An Evaluation of the Performance of a NoSQL Document Database in a Simulation of a Large Scale Electronic Health Record (EHR) System

A Thesis submitted by

Mehmet Zahid Ercan
BBA, MIT

For the award of

Doctor of Philosophy

2017

Abstract

Electronic Healthcare Record (EHR) systems can provide significant benefits by improving the effectiveness of healthcare systems. Research and industry projects focusing on storing healthcare information in NoSQL databases has been triggered by practical experience demonstrating that a relational database approach to managing healthcare records has become a bottleneck. Previous studies show that NoSQL databases based on consistency, availability and partition tolerance (CAP) theorem have significant advantages over relational databases such as easy and automatic scaling, better performance and high availability. However, there is limited empirical research that has evaluated the suitability of NoSQL databases for managing EHRs. This research addressed this identified research problem and gap in the literature by investigating the following general research: How can a simulation of a large EHR system be developed so that the performance of NoSQL document databases comparative to relational databases can be evaluated?

Using a Design Science approach informed by a pragmatic worldview, a number of IT artefacts were developed to enable an evaluation of performance of a NoSQL document oriented database comparative to a relational database in a simulation of a large scale EHR system. These were healthcare data models (NoSQL document database, relational database) for the Australian Healthcare context, a random healthcare data generator and a prototype EHR system. The performance of a NoSQL document database (Couchbase) was evaluated comparative to a relational database (MySQL) in terms database operations (insert, update, delete of EHRs), scalability, EHR sharing and data analysis (complex querying) capabilities in a simulation of a large scale EHR system, constructed in the cloud environment of Amazon Web Services (AWS). Test scenarios consisted of a number of different configurations ranging from 1, 2, 4, 8 and 16 nodes for 1Million, 10 Million, 100 Million and 500 Million records to simulate database operations in a large scale and distributed EHR system environment.

The Couchbase NoSQL document database was found to perform significantly better than the MySQL relational database in most of the test cases in terms of database operations -insert, update, delete of EHRs, scalability and EHR sharing. However, the MySQL relational database was found to perform significantly better than the

Couchbase NoSQL document database for the complex query test that demonstrates basic analysis capabilities. Furthermore, the Couchbase NoSQL document database used significantly more disk space than the MySQL relational database to store the same number of EHRs.

This research made a number of important contributions to knowledge, theory and practice. The main theoretical contribution to design theory was the design and evaluation of a prototype EHR system for simulating database management operations in a large scale EHR system environment. The prototype EHR system was underpinned by the development of two data models with data structures designed for a NoSQL document database and a relational database and a random healthcare data generator which were based on Australian Healthcare data characteristics and statistics. The design of a data model for EHRs for a NoSQL document database using an aggregated document modelling approach provided an important contribution to data modelling theory for NoSQL document databases using de-normalisation and document aggregation. The design of a random healthcare data generator was another important contribution to design theory and was based on a data distribution algorithm (multinomial distribution and probability theory) informed by National Health Data Dictionary and published Australian Healthcare statistics. The prototype EHR system allowed this study to demonstrate through a simulated performance evaluation that a NoSQL document database has significant and proven performance advantages over relational databases in most of the database management test cases. Hence this study demonstrated the utility and efficacy of a NoSQL document database in the simulation of a large scale EHR system. This research has made a number of important contributions to practice. Foremost is that the IT artefacts (namely, a data model for storing EHRs in a NoSQL document database, a random healthcare data generator and a prototype EHR system) developed and evaluated in this research can be readily adopted by practitioners. Another important practical contribution of this research is that it is based on the open source availability of NoSQL database and relational database alternatives. Hence, this research can provide a sound basis for lower-income countries as well higher-income countries to establish their own cost-effective national EHR systems without the restrictions, limitations, complexity or complications of similar proprietary relational database systems.

Certification of Thesis

This thesis is entirely the work of Mehmet Zahid ERCAN except where otherwise acknowledged. The work is original and has not previously been submitted for any other award, except where acknowledged.

Student and supervisors signatures of endorsement are held at USQ.

Dr. Michael Lane

Principal Supervisor

Prof. Raj Gururajan

Associate Supervisor

List of Publications

CONFERENCE PROCEEDINGS

Ercan, MZ, Lane, M 2014, An Evaluation of NoSQL Databases for Electronic Health Record Systems. Proceedings of the 25th Australasian Conference on Information Systems, Auckland, New Zealand.

Acknowledgements

First and foremost, I would like to express my humbleness and gratitude for being a part of the humanity who have been given the health, energy, time, and knowledge to be able to work on this research.

I acknowledge, with deep gratitude, the inspiration and encouragement by the great leaders of mankind who emphasised the value of seeking knowledge and advocation of knowledge and truth.

I would like to thank my principal supervisor Dr Michael Lane. I am deeply indebted and grateful to him for his extensive guidance, patience, continuous support, and encouragement. This thesis would not have been completed without his support and motivation.

I also would like to thank my associate supervisor Prof Raj Gururajan for guiding me in the right direction and for his support over the duration of this PhD Thesis. I also would like to acknowledge and thank Chris O'Reilly and Ms Sohrab for their editorial assistance and the Australian Government for supporting this research through the Research Training Scheme (RTS).

Finally, special thanks go to my family, friends, colleagues and business partners for their endless patience and encouragement. I acknowledge that they have given so much of their own time and effort to support me and to take care of the things that I could not find time to do while completing this PhD Thesis.

Table of Contents

Abstract i
Certification of Thesis iii
List of Publications iv
Acknowledgements v
List of Figures x
List of Tables xii
List of Abbreviations xv
Chapter 1 – Introduction 1
1.1 Chapter Introduction 1
1.2 Background and Motivation 1
1.3 Research Problem and Research Questions 4
1.4 Research Paradigm and Methodological Approach 5
1.5 Research Design and Scope 7
1.6 Planned Research Contributions 9
1.7 Outline of the Thesis 10
1.8 Definition of Key Terms 11
1.9 Chapter Summary 12
Chapter 2 - Literature Review 14
2.1 Introduction 14
2.2 Electronic Health Records (EHR) 15
2.2.1 EHR systems 16
2.2.2 EHR Systems in Australia 17
2.2.3 Electronic Health Record sharing functionality in EHR systems 17
2.2.4 Importance of EHR systems for Healthcare 18
2.2.5 Technological issues affecting EHR systems 18
2.3 NoSQL databases 20
2.3.1 Types of NoSQL databases 21
2.4 Theoretical Background 25
2.4.1 Relational Database Theory 26
2.4.2 Advantages of NoSQL document databases over relational databases 27
2.4.3 NoSQL Data Modelling versus Relational Data Modelling 29
2.4.4 Determining EHR data elements for NoSQL and Relational Data Models 33
2.4.5 CAP Theorem and NoSQL Databases 34
2.4.6 ACID Properties and NoSQL Databases 35
2.5 Suitability of NoSQL databases for EHR systems 37

2.5.1 CAP Theorem and NoSQL Databases in EHR systems 38
2.6 Previous Research on Performance and Scalability of NoSQL Databases 39
2.6.1 Previous Research on Evaluation of NoSQL Databases in Healthcare 40
2.7 Literature Gap and Research Focus 41
2.8 Conceptual Model and Research Questions 42
2.9 Conclusion 44
Chapter 3 - Methodology 47
3.1 Introduction 47
3.2 Research Philosophy 48
3.2.1 Methodological Approach 51
3.3 Overall Research Design 54
3.3.1 Identify research problem and need to conduct research 57
3.4 Research Plan 61
3.5 Evaluating Design Science Research Approach 65
3.6 Planned Research Contribution 67
3.7 Conclusion 68
Chapter 4 - Development of IT Artefacts 71
4.1 Introduction 71
4.2 Identification of Australian Healthcare Data Set Requirements 72
4.3 Development of Relational and NoSQL Data Models 75
4.3.1 Relational EHR Data Model 75
4.3.2 NoSQL EHR Data Model 79
4.4 Identification of Relevant Australian Healthcare Statistics 80
4.4.1 Separations 81
4.4.2 Age Group and Sex 84
4.4.3 Indigenous Status 85
4.4.4 Mode of Admission 86
4.4.5 Urgency of Admission 86
4.4.6 Principal Diagnosis 87
4.5 Development of Random Healthcare Data Generator 88
4.5.1 Data distribution algorithm 89
4.5.2 Validation of the random data generation algorithm 92
4.6 Development of Prototype EHR System 94
4.7 Conclusion 96
Chapter 5 – Simulation and Evaluation 98
5.1 Introduction 98
5.2 Database Selection 99

5.3 Setting up the distributed test environment and scenarios 100
5.3.1 Establishing cloud environment 100
5.3.2 Test scenarios 101
5.4 Running the tests 102
5.4.1 Simulation of data insertion 103
5.4.2 Simulation of update operations 117
5.4.3 Simulation of delete operations 130
5.4.4 Simulation of EHR sharing through retrieval of patient EHRs 142
5.4.5 Data Size 145
5.4.6 Query Capabilities 146
5.5 Conclusion 147
Chapter 6 – Discussion and Evaluation of this Research 149
6.1 Introduction 149
6.2 Discussion of Key Findings 150
6.2.1 Development of Relational and NoSQL Data Models - Research Question 1 150
6.2.2 Random Healthcare Data Generator – Research Question 2 151
6.2.3 EHR System Prototype – Research Question 3 152
6.2.4 Performance evaluation for basic database operations (insert, update, delete) for NoSQL and relational databases – Research Question 4 153
6.2.5 Scalability capabilities of NoSQL document database and relational database – Research Question 5 156
6.2.6 EHR Sharing Simulation – Research Question 6 159
6.2.7 Complex Query – Research Question 7 159
6.2.8 Data Size 160
6.3 Evaluation of this Research using Design Science Guidelines 162
6.3.1 Design of IT Artefacts in this Study 162
6.3.2 Problem Relevance of this Study 162
6.3.3 Design Evaluation of IT Artefacts in this Study 163
6.3.4 Research Contributions 164
6.3.5 Research Rigour 165
6.3.6 Design as a Search Process in this Study 166
6.3.7 Communication of this Research 166
6.4 Conclusion 168
Chapter 7 – Conclusion 169
7.1 Introduction 169
7.2 Summary of Study 170
7.2.1 Research Problem 170

7.2.2 Research Methodology – Design and Evaluation Activities.....171
7.3 Summary of Key Findings for each Research Question Investigated 174
7.4 Research Contributions to Theory and Practice.....177
7.4.1 Contribution to Theory.....177
7.4.2 Contribution to Practice 179
7.5 Limitations and Future Research 180
7.6 Summary 182
List of References 184
List of Appendices 199
Appendix A. Separation statistics, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016)) 200
Appendix B. Separations, by state or territory of usual residence and establishments, 2014–15 (Adopted from (AIHW 2016)) 202
Appendix C. Separations per 1,000 population, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016)).....203
Appendix D. Same-day and overnight separations per 1,000 population, states and territories, 2014–15 (Adopted from (AIHW 2016)) 204
Appendix E. Separations by mode of admission, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016)).....205
Appendix F. Admitted Patient Care National Minimum Dataset Details (Adopted from (AIHW 2015)).....206
Appendix G. JSON representation of aggregate oriented data model 209

List of Figures

Figure 1.1 Structure of Chapter 1 1
Figure 1.2 Overview of the research activities undertaken in this research 9
Figure 2.1 Structure of Chapter 2 14
Figure 2.2: Google search trends NoSQL databases versus Relational Databases 21
Figure 2.3: Key-Value Store representation (Adapted from Sadalage (2014)). 22
Figure 2.4: A sample representation of data stored in a document store (Adapted from Sadalage (2014) 22
Figure 2.5: A sample data structure representation of column family type of NoSQL database (Adapted from Sadalage (2014) 23
Figure 2.6: Data structure representation for a graph database (Adapted from Sadalage (2014) 24
Figure 2.7. Comparison of NoSQL databases based on model complexity and scalability (Adapted from Hsieh (2014) 30
Figure 2.8: Diagram showing a sample user data in relational model (Adapted from Couchbase (2016) 32
Figure 2.9: Initial result set for querying a sample user data in relational model (Adapted from Couchbase (2016) 32
Figure 2.10: Result set for querying a sample user data in document data model (Adapted from Couchbase (2016) 32
Figure 2.11: Related and Nested Document Database Models compared to Relational Database Model for sample user data (Adapted from Segleau (2016)) 33
Figure 2.12: Comparison of the three main data model types, Key-Value, Column Family, and Document Oriented, used in NoSQL databases with relational databases in terms of CAP Theorem (Adapted from Fernando (2016) 36
Figure 2.13: Conceptual model of artefacts built and evaluated and associated research activities conducted to achieve main objectives of this study 43
Figure 3.1 Structure of Chapter 3 47
Figure 3.2: Framework for theory development in Design Science Research (Adapted from Kuechler and Vaishnavi (2008) 53
Figure 3.3. Design Science Research Model (Adapted from Hevner, 2004) 55
Figure 3.4. Research Phases used to conduct this research. 62
Figure 3.5. DSR Knowledge Contribution Framework (Gregor & Hevner, 2013) 68
Figure 4.1 Structure of Chapter 4 72
Figure 4.2. Entity Relationship Diagram for relational data model 78
Figure 4.3. A sample section of NoSQL EHR data model 80
Figure 4.4. The relationship between IT artefacts (Random Healthcare Data Generator, Prototype EHR System) and database nodes 96
Figure 5.1. Structure of Chapter 5 99
Figure 5.2. Average number of records inserted per second with standard deviations for Couchbase and MySQL in single-node configuration. 106
Figure 5.3. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 2-node configuration. 108
Figure 5.4. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 4-node configuration. 111
Figure 5.5. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 8-node configuration. 114
Figure 5.6. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 16-node configuration. 116

Figure 5.7. Average number of records updated per second with standard deviations for Couchbase and MySQL in single-node configuration. 119
Figure 5.8. Average number of records updated per second with standard deviations for Couchbase and MySQL in 2-node configuration. 122
Figure 5.9. Average number of records updated per second with standard deviations for Couchbase and MySQL in 4-node configuration. 125
Figure 5.10. Average number of records updated per second with standard deviations for Couchbase and MySQL in 8-node configuration. 127
Figure 5.11. Average number of records updated per second with standard deviations for Couchbase and MySQL in 16-node configuration. 129
Figure 5.12. Average number of records deleted per second with standard deviations for Couchbase and MySQL in single-node configuration. 132
Figure 5.13. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 2-node configuration. 135
Figure 5.14. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 4-node configuration. 137
Figure 5.15. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 8-node configuration. 140
Figure 5.16. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 16-node configuration. 142
Figure 5.17. The average number of EHR sharing operations per second for Couchbase and MySQL. 145
Figure 5.18. Size of the data by the number of records stored for Couchbase and MySQL. 146
Figure 6.1. Structure of Chapter 6 150
Figure 6.2. Average number of operations per second per node count for Couchbase database. 157
Figure 6.3. Average number of operations per second per node count for MySQL database. 158
Figure 7.1 Structure of Chapter 7 170

List of Tables

Table 2.1. Typical Use Cases and Example Applications for NoSQL database types (Adapted from (Gudivada, Rao & Raghavan 2016) 25
Table 2.2. Comparison of Key Differences between NoSQL Databases and Relational Databases 27
Table 2.3. Comparison of healthcare data and NoSQL database characteristics (Adapted from Goli-Malekabadi, Sargolzaei-Javan and Akbari (2016)).....31
Table 2.4. Comparison of EHR requirements and NoSQL database features that address these requirements.....38
Table 3.1 Summary of Research Paradigm Perspectives used in Information Systems (adapted from Aljafari and Khazanchi (2013)).....49
Table 3.2 Design Science Activities/Steps Taken Distilled from Literature (adopted from Alturki, Gable & Bandara (2011).....56
Table 3.3. Guidelines for assessment of DSR adapted from Hevner et al. (2004) 67
Table 4.1. Data elements by categories in the selected datasets: Admitted Patient Care and Non-admitted Patient Emergency Care (AIHW 2015).....75
Table 4.2. Separation statistics for 2014-2015 based on age and sex (Adopted from AIHW 2016) 85
Table 4.3. Separation statistics for 2014-2015 based urgency of admission (Adopted from AIHW 2016).....87
Table 4.4. Separation statistics for 2014-2015 based principal diagnosis (Adopted from AIHW 2016).....88
Table 4.5. An example table for lower-upper boundaries for age group statistics. ...90
Table 4.6. An example table for lower-upper boundaries for sex statistics.....91
Table 4.7. An example combined table for lower-upper boundaries for age group and sex statistics.....91
Table 4.8. Validation test for random data generation algorithm based on actual urgency of admission statistics.....93
Table 5.1 Configuration Scenarios for Performance Tests 101
Table 5.2 Execution time statistics in milliseconds for data insert operations on single-node Couchbase database.....104
Table 5.3 Number of insert operations per second on single-node Couchbase database 104
Table 5.4 Execution time statistics in milliseconds for data insert operations on single MySQL database instance. 105
Table 5.5 Number of transactions per second for data insertion on single MySQL database instance.....105
Table 5.6 Execution time statistics in milliseconds for data insert operations on 2-node Couchbase cluster.....107
Table 5.7 Number of insert operations per second on 2-node Couchbase database 107
Table 5.8 Execution time statistics in milliseconds for data insert operations on 2-node MySQL cluster 108
Table 5.9 Number of insert operations per second on 2-node MySQL database 108
Table 5.10 Execution time statistics in milliseconds for data insert operations on 4-node Couchbase cluster.....109
Table 5.11 Number of insert operations per second on 4-node Couchbase database 109
Table 5.12 Execution time statistics in milliseconds for data insert operations on 4-node MySQL cluster 110

Table 5.13 Number of insert operations per second on 4-node MySQL database ..110
Table 5.14 Execution time statistics in milliseconds for data insert operations on 8-node Couchbase cluster.....112
Table 5.15 Number of insert operations per second on 8-node Couchbase database 112
Table 5.16 Execution time statistics in milliseconds for data insert operations on 8-node MySQL cluster 113
Table 5.17 Number of insert operations per second on 8-node MySQL database ..113
Table 5.18 Execution time statistics in milliseconds for data insert operations on 16-node Couchbase cluster.....115
Table 5.19 Number of insert operations per second on 16-node Couchbase database 115
Table 5.20 Execution time statistics in milliseconds for data insert operations on 16-node MySQL cluster 115
Table 5.21 Number of insert operations per second on 16-node MySQL database 116
Table 5.22 Execution time statistics in milliseconds for data update operations on single-node Couchbase database.....118
Table 5.23 Number of update operations per second on single-node Couchbase database 118
Table 5.24 Execution time statistics in milliseconds for data update operations on single node MySQL cluster.....119
Table 5.25 Number of update operations per second on single node MySQL database 119
Table 5.26 Execution time statistics in milliseconds for data update operations on 2-node Couchbase cluster.....120
Table 5.27 Number of update operations per second on 2-node Couchbase cluster121
Table 5.28 Execution time statistics in milliseconds for data update operations on 2-node MySQL cluster 121
Table 5.29 Number of update operations per second on 2-node MySQL database.121
Table 5.30 Execution time statistics in milliseconds for data update operations on 4-node Couchbase cluster.....123
Table 5.31 Number of update operations per second on 4-node Couchbase cluster123
Table 5.32 Execution time statistics in milliseconds for data update operations on 4-node MySQL cluster 124
Table 5.33 Number of update operations per second on 4-node MySQL database.124
Table 5.34 Execution time statistics in milliseconds for data update operations on 8-node Couchbase cluster.....125
Table 5.35 Number of update operations per second on 8-node Couchbase cluster126
Table 5.36 Execution time statistics in milliseconds for data update operations on 8-node MySQL cluster 126
Table 5.37 Number of update operations per second on 8-node MySQL database.127
Table 5.38 Execution time statistics in milliseconds for data update operations on 16-node Couchbase cluster.....128
Table 5.39 Number of update operations per second on 16-node Couchbase cluster 128
Table 5.40 Execution time statistics in milliseconds for data update operations on 16-node MySQL cluster 129
Table 5.41 Number of update operations per second on 16-node MySQL database 129

Table 5.42 Execution time statistics in milliseconds for delete operations on single-node Couchbase database....131
Table 5.43 Number of delete operations per second on single-node Couchbase cluster....131
Table 5.44 Execution time statistics in milliseconds for data delete operations on single-node MySQL cluster....132
Table 5.45 Number of delete operations per second on single-node MySQL database....132
Table 5.46 Execution time statistics in milliseconds for delete operations on 2-node Couchbase cluster....133
Table 5.47 Number of delete operations per second on 2-node Couchbase cluster 134
Table 5.48 Execution time statistics in milliseconds for data delete operations on 2-node MySQL cluster....134
Table 5.49 Number of delete operations per second on 2-node MySQL database..135
Table 5.50 Execution time statistics in milliseconds for delete operations on 4-node Couchbase cluster....136
Table 5.51 Number of delete operations per second on 4-node Couchbase cluster 136
Table 5.52 Execution time statistics in milliseconds for data delete operations on 4-node MySQL cluster....137
Table 5.53 Number of delete operations per second on 4-node MySQL database..137
Table 5.54 Execution time statistics in milliseconds for delete operations on 8-node Couchbase cluster....138
Table 5.55 Number of delete operations per second on 8-node Couchbase cluster 138
Table 5.56 Execution time statistics in milliseconds for data delete operations on 8-node MySQL cluster....139
Table 5.57 Number of delete operations per second on 8-node MySQL database..139
Table 5.58 Execution time statistics in milliseconds for data delete operations on 16-node Couchbase cluster....140
Table 5.59 Number of delete operations per second on 16-node Couchbase cluster....141
Table 5.60 Execution time statistics in milliseconds for data delete operations on 16-node MySQL cluster....141
Table 5.61 Number of delete operations per second on 16-node MySQL database 141
Table 5.62 Execution time statistics in milliseconds for EHR sharing simulation..143
Table 5.63 Number of operations per second for EHR sharing simulation....144
Table 5.64 Size of the data by the number of records stored for Couchbase and MySQL....146
Table 6.1. Average number of operations per second by the number of nodes and operation type for Couchbase database and MySQL database. 153
Table 6.2. Average execution times by the number of nodes and operation type for Couchbase database and MySQL database.....154
Table 6.3. Average number of operations per second for the number of stored records and operation type for Couchbase database and MySQL database.155
Table 6.4. Percentage change in average number operations per second per change in node count by operation type for Couchbase database and MySQL database. 156

List of Abbreviations

1NF	First Normal Form
2NF	Second Normal Form
ACID	Atomicity, Consistency, Isolation and Durability
ACIS	Australasian Conference on Information Systems
AIHW	Australian Institute of Health and Welfare
AWS	Amazon Web Services
BASE	Basic Availability, Soft State, Eventual Consistency
CAP	Consistency, Availability, and Partition-tolerance
CDAC	Centre for Development of Advanced Computing
CPU	Central Processing Unit
DIGHT	Distributed Infrastructure for Global Electronic Health Record Technology
DSR	Design Science Research
DSRM	Design Science Research Methodology
DSS	Data Set Specification
EBS	Elastic Block Storage
EC2	Elastic Compute Cloud
EHR	Electronic Health Record
EPR	Electronic Patient Records
GB	Gigabyte
ICD10	International Statistical Classification of Diseases and Related Health Problems, Tenth Edition
IO	Input Output
IOPS	Input Output Per Second
IS	Information Systems
ISO	The International Organization for Standardization

IT	Information Technology
JSON	JavaScript Object Notation
MIS	Management Information Systems
NCRS	National Care Record Service
NEHTA	National E-Health Transition Authority
NHDD	National Health Data Dictionary
NHDID	National Healthcare Document ID
NHS	National Health Service
NMDS	National Minimum Data Sets
NSP	National Switch Point
PCEHR	Personally Controlled Electronic Health Record
RDBMS	Relational Database Management Systems
SICS	Swedish Institute of Computer Science
SSD	Solid-state Drive
TPC	Transaction Processing Performance Council
XML	Extensible Markup Language
YCSB	Yahoo Cloud Serving Benchmark

Chapter 1 – Introduction

1.1 Chapter Introduction

This first chapter of the thesis provides the foundation for this research. First, the background and the motivation for undertaking this research are discussed. Then a description of the research problem is provided and the research questions that were investigated in this study are introduced. Next, the research paradigm and methodological approach that guided the conduct of this research are outlined. Finally, the scope and planned contributions of this research are presented, followed by a general outline of the thesis chapters. Figure 1.1 outlines the structure of this chapter.

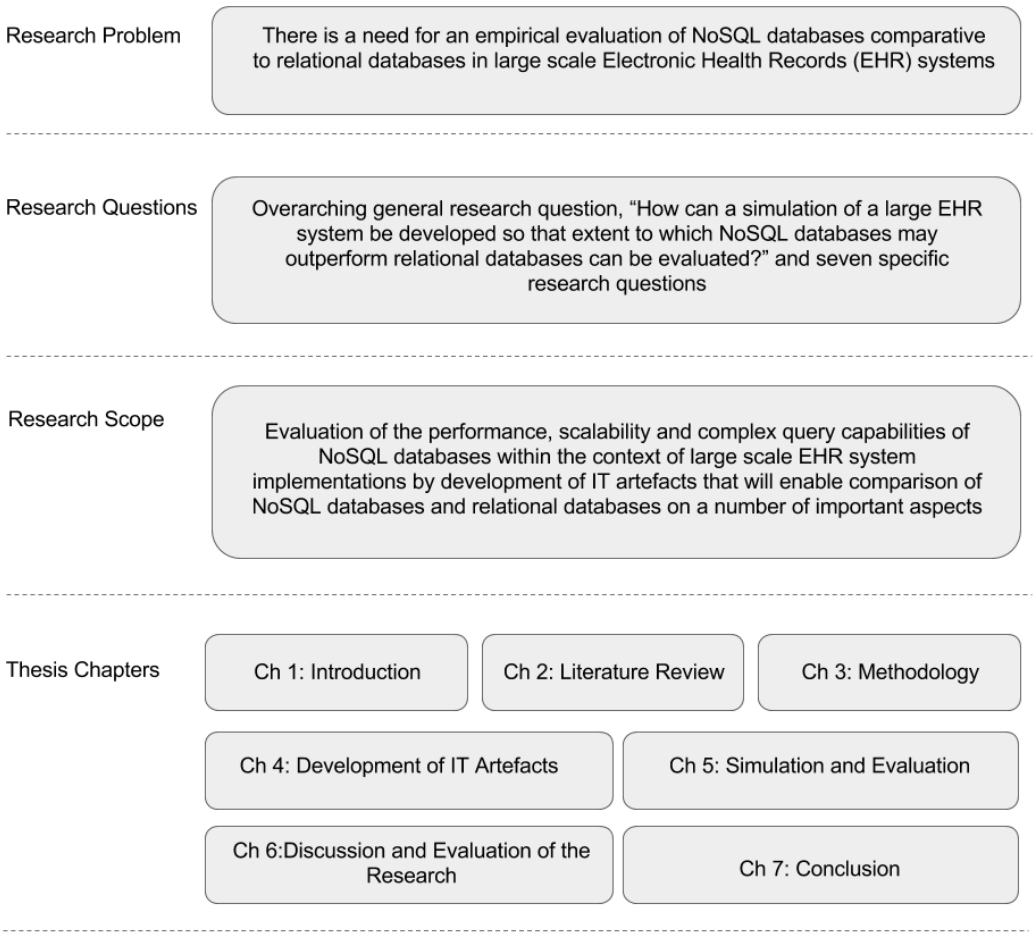


Figure 1.1 Structure of Chapter 1

1.2 Background and Motivation

Electronic Health Record (EHR) systems and healthcare data sharing between healthcare providers remain a significant challenge for many countries, despite many developments in database technology and network infrastructure (Bacelar-Silva et al. 2011; Hoerbst et al. 2010; Pearce & Haikerwal 2010). Many countries such as Australia, Finland, Germany and Turkey are working on establishing nationwide e-health platforms that will facilitate data sharing. However, issues about data standards, scalability, high volumes of data storage, data processing and the cost of EHR system implementations are particularly challenging for governments and healthcare providers (Bacelar-Silva et al. 2011; Drejhammar 2010; Grimson 2001; Hoerbst et al. 2010; Jin, Deyu & Xianrong 2011; Pearce & Haikerwal 2010; Schmitt & Majchrzak 2012; Vest 2012).

Data intensive information systems such as healthcare systems require database management systems in order to function properly (Mengchen 2011; Vera et al. 2015). The size and heterogeneity of data stored and managed in modern distributed systems, including healthcare systems, are increasing exponentially (Floratou et al. 2012; Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Lee, Tang & Choi 2013).

Most EHR systems are based on relational databases which struggle to accommodate the expanding size and evolving structure and use of healthcare data that requires data management systems that support scalability, high availability and data model flexibility which cannot be provided by relational databases (Blobel 2006; Dolin et al. 2006; Freire et al. 2016; Guo et al. 2004; Jin, Deyu & Xianrong 2011; Orfanidis, Bamidis & Eaglestone 2004; Schmitt & Majchrzak 2012). Furthermore, large scale EHR systems have significant potential for improving clinical decision support, population health management, discovering patterns and developing new treatments using efficient parallel data analytics over large volumes of healthcare data (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Hermon & Williams 2014). However, managing and analysing large scale healthcare data requires new data management tools and methods (Raghupathi & Raghupathi 2014; Sun & Reddy 2013)

The need to scale databases beyond the capabilities of relational databases running on a single large computer system has driven the introduction of new scalable database systems (Borkar, Carey & Li 2012; Helland 2011; Konstantinou et al. 2011). These

new systems are referred to as “NoSQL” databases. While the name is not entirely agreed upon, NoSQL stands for “Not Only SQL” (Cattell 2011).

Relational database management systems have limitations due to scalability and infrastructure cost issues (Borkar, Carey & Li 2012; Cattell 2011). NoSQL database systems which have emerged in response to these limitations are mostly open-source and can run on commodity hardware architectures (Jin, Deyu & Xianrong 2011; Konishetty et al. 2012; Valduriez 2011). NoSQL database systems can scale horizontally with no single point of failure or bottlenecks because of a shared-nothing architecture (Borkar, Carey & Li 2012; Konishetty et al. 2012).

In a shared-nothing architecture, servers have their own resources, thus they do not share RAM, processor or storage capability (Borkar, Carey & Li 2012; Cattell 2011). This enables horizontal scaling, the distribution of data and processing operations over many servers to achieve large numbers of read/write operations per second (Cattell 2011).

NoSQL databases offer low-cost solutions that provide high availability and address scalability issues. NoSQL database systems have been heavily influenced by Google’s Bigtable and Amazon’s Dynamo systems and can easily scale up to accommodate large datasets (Borkar, Carey & Li 2012; Schram & Anderson 2012). Some NoSQL databases have been developed and used commercially by companies such as Google and Amazon. However, there are many open source NoSQL database systems based on similar approaches, including HBase, MongoDB, CouchDB, Cassandra, Couchbase, etc. (Schram & Anderson 2012).

NoSQL database systems are already used in some large commercial applications by technology company leaders such as Google, Amazon, LinkedIn and Facebook. NoSQL database systems can support the management of more complex and heterogeneous data sources and offer high scalability and high availability that relational database systems cannot provide (Borkar, Carey & Li 2012; Cattell 2011; Konstantinou et al. 2011). Furthermore, open source NoSQL database systems have a significant advantage in terms of implementation and software licence costs over relational database systems. This is another reason to use NoSQL database systems to address the shortcomings of commercial relational database management systems (Escriva, Wong & Sirer 2012).

Implementation of EHR systems in many countries are in progress. Countries such as Turkey, Australia, China and the UK are following an e-health transition strategy. The use of information systems in healthcare facilities is increasing as it is promoted by national strategies and working groups (Australian Digital Health Agency 2015; Nøhr et al. 2005). Increasing diffusion of information systems to deliver healthcare and the increasing size and heterogeneity of healthcare data has resulted in a bottleneck for storage, retrieval, high availability and analysis aspects of relational databases. NoSQL database systems might be the solution to this bottleneck (Jin, Deyu & Xianrong 2011; Schmitt & Majchrzak 2012).

While there are significant advantages in using NoSQL database systems, there is limited research which has compared the performance of NoSQL databases in terms of database operations (insert, update, delete), scalability and data analysis (complex querying) capability with relational database systems in a healthcare domain. There are numerous white papers, blog entries and comments mentioning the advantages of NoSQL database systems over relational databases. However, there are very few empirical studies that compare NoSQL database systems and relational database systems, particularly in the context of healthcare (Parker, Poe & Vrbsky 2013). Hence, there is a significant need to evaluate these different types of database systems (NoSQL versus relational) in a healthcare context.

1.3 Research Problem and Research Questions

The identification of the gap in the literature provided the motivation for conducting this research. The research problem that was addressed in this PhD thesis can be defined as:

There is a need for an empirical evaluation of the performance of NoSQL document databases in terms of database operations, scalability, data sharing and data analysis aspects comparative to relational databases in large scale Electronic Health Records (EHR) systems based on a healthcare data model.

Based on this research problem, the general research question investigated in this study is framed as follows.

General RQ: How can a simulation of a large EHR system be developed so that the performance of NoSQL document databases comparative to relational databases can be evaluated?

In order to investigate this general research question, seven specific research questions are investigated to evaluate the feasibility of NoSQL document databases for managing distributed EHRs in an Australian healthcare context.

RQ1: How can a NoSQL document data model and a relational data model be developed for an EHR system that are in line with documents published by healthcare authorities in Australia?

RQ2: How can a random healthcare data generator be developed that will generate EHRs that are representative of the characteristics of Australian healthcare data based on statistics available in the public domain?

RQ3: How can a prototype EHR system be developed that will facilitate database operations and measure performance and scalability for NoSQL document databases and relational databases?

RQ4: How do NoSQL document databases perform compared to relational databases in executing basic database operations such as insert, delete and update on electronic health records?

RQ5: How do NoSQL document databases scale compared to relational databases in electronic health record systems?

RQ6: How do NoSQL document databases perform compared to relational databases in supporting electronic health record sharing through patient record retrieval in a distributed EHR system?

RQ7: How do NoSQL document databases perform compared to relational databases in executing complex queries on electronic health records?

1.4 Research Paradigm and Methodological Approach

This research addresses the gap identified in the literature and proposes a solution to the research problem described in the previous section by evaluating a NoSQL document database system in terms of the performance of database operations,

scalability, data sharing and data analysis (complex querying) capabilities comparative to a relational database in the context of the healthcare domain using healthcare-specific data models and data characteristics. This performance evaluation of a NoSQL database in the healthcare domain is based on the development of IT artefacts that are specifically built for healthcare applications rather than using a generic performance-measurement approach.

Researchers in the Information Systems (IS) discipline have increasingly used Design Science Research (DSR) as an approach to understand and provide solutions to real world problems. Design Science Research is identified as a problem-solving paradigm, as opposed to a problem understanding paradigm. There has been a number of research projects based on a DSRM which consist of design and evaluation of artefacts to justify the contributions to theory and practice in the IS discipline (Hevner et al. 2004; Peffers et al. 2007). Since the purpose of this research project is to develop IT artefacts and evaluate these IT artefacts as a suggested solution to a particular IS problem, a Design Science Research (DSR) is an appropriate research paradigm and methodology for collecting data to answer the specific research questions framed by the research problem and main objectives of this study.

Hevner et al (2004) argue that the understanding of a problem domain and providing a real world solution can be achieved by designing, building and evaluating an artefact (Hevner et al. 2004). The term artefact is a broad term. IT artefacts are diverse, with many possible manifestations and forms and may be composed from hardware, software and process information based on predefined rules, logic, structures, routines and values embedded in them (Zhang, Scialdone & Ku 2011).

Although various steps for conducting Design Science Research have been suggested there are commonly-agreed steps such as identification of the problem, design of the artefacts and evaluation of artefacts as solutions to a specific problem (Alturki, Gable & Bandara 2011; Gregor & Jones 2007; Rossi & Sein 2003).

Therefore, in this research, identification of the problem and solution requirements within the healthcare context are determined by designing building and evaluating IT artefacts to provide a solution to the problem being addressed in this study. Then these IT artefacts are used to evaluate the performance of a NoSQL document database in the healthcare domain comparative to relational databases.

1.5 Research Design and Scope

This research focuses on the evaluation of the performance of a NoSQL document database in terms of database operations, scalability and data analysis capabilities (complex querying) comparative to a relational database within the context of large scale EHR system implementations. This evaluation is achieved by development of IT artefacts that enable a performance evaluation of a NoSQL document database comparative to a relational database on a number of important data management aspects, namely, the performance of basic database operations (insert, update, delete, retrieval), scalability, data sharing and data analysis capabilities (complex querying). Previous empirical studies guided the performance comparison of a NoSQL document database with a relational database. The performance evaluation conducted in this study made use of commonly executed workload scenarios for evaluating database performance using the Yahoo Cloud Serving Benchmark (YCSB) tool (Cooper et al. 2010) that is discussed in detail in section 2.6.

In order for the performance evaluation of databases for a particular domain such as healthcare to be a realistic representation, it is required to establish a relevant data structure. The structure of the data for a specific domain such as healthcare can directly affect the overall performance of the underlying database. The required data sets and data elements of the required data structure are modelled across tables and fields for a relational database and the required data structure similarly determines the type of the NoSQL database to be used. Furthermore, the complexity of database operations and the overall database size are also directly determined by the required data structure (Nance et al. 2013; Swaroop & Vijit Gupta 2016). Therefore selecting data sets and data elements for performance comparison of a NoSQL document database with a relational database needed to be aligned with the context of the study, healthcare to achieve more accurate results.

As the healthcare data context for this study was the Australian healthcare system, data sets and data elements that may represent EHRs in the Australian healthcare domain are identified and data models are established. These healthcare related data models are based on the Australian National Health Data Dictionary which is used mostly for data collection and administrative purposes—and discussed later in subsequent chapters. Although healthcare systems might have many more data sets and data

elements in reality, the Australian National Health Data Dictionary establishes a sound basis for determining the minimum (basic) required data sets and elements to be stored in an EHR system (AIHW 2015).

Following the identification of the data elements of a data model for storing EHRs, healthcare statistics that represent the data characteristics for the Australian health system are identified, and provide the basis for populating an EHR data model with healthcare data.

After these steps, IT artefacts are developed that underpin the simulation of a large scale EHR system to enable the evaluation of NoSQL document databases in the healthcare domain comparative to relational databases in this research. These IT artefacts are a Random Healthcare Data Generator and prototype EHR system. The Random Healthcare Data Generator, provides simulated healthcare data that is representative of Australian healthcare data based on Australian healthcare statistics. The second artefact, the prototype EHR system, facilitated conducting tests for the evaluation of a NoSQL document database comparative to a relational database in a healthcare domain for a number of important database operations such as insert, update, delete of EHRs, scaling, EHR sharing and execution of complex database queries.

EHR sharing is one of the important aspects of EHR systems (Narayan, Gagne & Safavi-Naini 2010). Sharing EHRs between healthcare providers and a national EHR system has various challenges including authentication, security, and privacy concerns, however, from a technical and operational perspective, EHR sharing requires the retrieval of EHRs from the underlying source database in all cases whether it is a single database or a distributed multi-system architecture (Bergmann et al. 2007; Jin et al. 2009; Zhang & Liu 2010). In this research, the EHR sharing was limited in scope to the retrieval of EHRs of a particular person as a key operation that needs to be evaluated in terms of performance.

The focus of this research is the performance evaluation of a NoSQL document database comparative to a relational database in the context of healthcare. Therefore "EHR sharing" as a term is used throughout this thesis to identify the operation of data retrieval of patient’s EHR for the purpose of demonstrating the performance of the databases for an EHR sharing scenario.

EHR sharing includes identification of EHRs (multiple records) of a particular person, retrieving these records, and -for the relational database- adding the corresponding values for the foreign key fields (joining the lookup tables) to make sure the output of the operation is an meaningful aggregation of multiple EHRs.

The overall research design, research steps and their relationship with the research questions investigated in this study are presented in Figure 1.2.

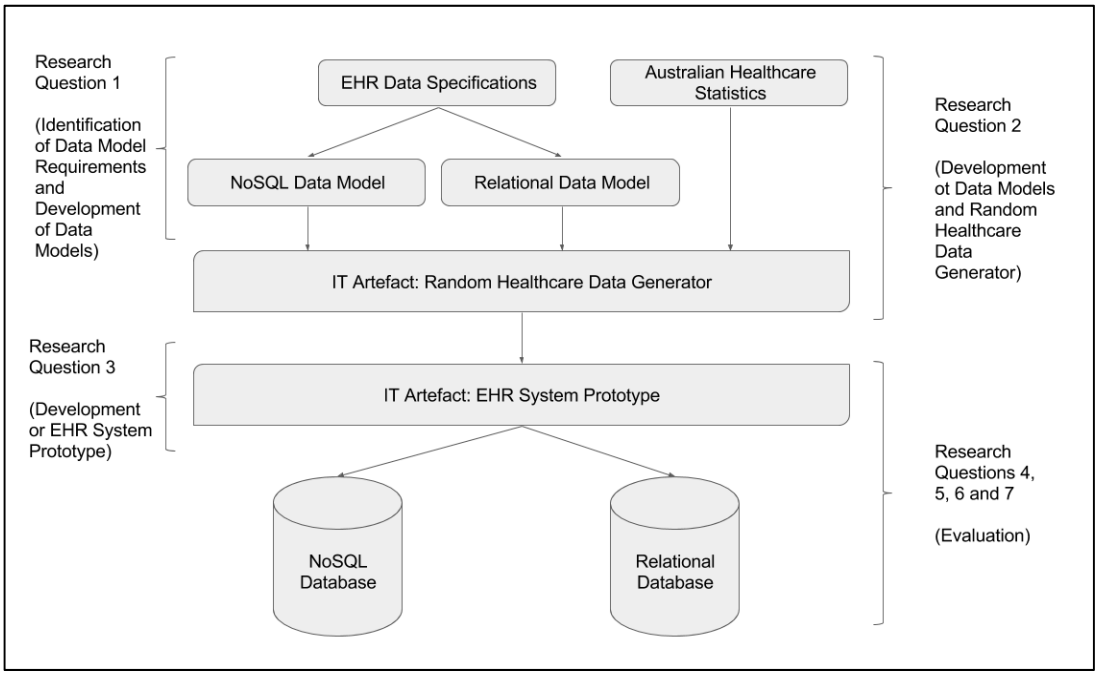


Figure 1.2 Overview of the research activities undertaken in this research

The scope of this research is limited to seven research questions concerned with the development of specific IT artefacts which then enabled an evaluation of the performance, scalability, data sharing and analysis capabilities (complex querying) of a NoSQL document database comparative to a relational database in a specific healthcare domain. Related topics such as privacy, interoperability, encryption, standards, etc. are independent and comprehensive areas of research within the healthcare domain and are out of scope for the main objectives and specific research questions investigated in this research.

1.6 Planned Research Contributions

The results of the design and evaluation of IT artefacts should be communicated to facilitate the accumulation of knowledge that is relevant, and which address a real

world problem (Hevner et al. 2004; Peffers et al. 2007). This research will contribute to theory and practice in a number of ways. Foremost, this research will address the gap in the literature by developing purpose-built IT artefacts and providing an empirical performance evaluation of database operations, scalability, data sharing and complex query capabilities of a NoSQL document database comparative to a relational database in a simulation of a large scale EHR system. This will contribute to design theory and knowledge by applying existing knowledge about NoSQL databases in a specific industry section, the healthcare domain, so that a number of artefacts can be built and evaluated to address a specific and important problem—data management in distributed EHR systems.

1.7 Outline of the Thesis

Following this introduction chapter, a comprehensive literature review is presented in Chapter 2. The previous literature on Electronic Health Records (EHR) systems and NoSQL databases was reviewed and identified the research problem and gap in the literature that is addressed by this study. The literature review is then used to frame the theoretical basis of this study. The scope and focus of this study is defined through an overarching general research question and seven specific research questions that were investigated.

Chapter 3 describes and justifies the choice of the research paradigm, Design Science, that in turn, guided the methodological approach used in this research. This chapter then discusses the steps taken to conduct this research using a Design Science Research Methodological approach and describes why this methodological approach is relevant and appropriate for this research. The research design and the steps undertaken in the design and evaluation of a number of IT artefacts are described and justified.

In Chapter 4, development of IT artefacts that underpinned the performance evaluation are described and discussed. First, existing public information about healthcare data sets and data elements for the Australian healthcare domain are investigated and the identified data sets and data elements that established the basis for developing the IT artefacts are presented. Then, all steps undertaken in the development process for each IT artefact are described in detail. A relational healthcare data model and a NoSQL healthcare document data model are developed based on the identified requirements.

Then, the steps undertaken in the design and implementation of the Random Healthcare Data Generator and Electronic Health Record System Prototype artefacts are described.

Chapter 5 presents the results of the performance evaluation of a NoSQL document database comparative to a relational database in a distributed EHR system. Following the selection of a relational database and a NoSQL document database for the purpose of this research, the Random Healthcare Data Generator artefact is used to generate simulated healthcare records data and populate a NoSQL document data model and a relational data model. A prototype EHR system is used to conduct and measure the performance of various test scenarios. Then, the results of the test scenarios used to evaluate the performance of a NoSQL document database comparative to a relational database are presented.

In Chapter 6 the key findings regarding each of the seven research questions in relation to existing literature are discussed in turn. The development of the IT artefacts to enable the performance evaluation of the selected NoSQL document database and selected relational database in a simulation of a large scale EHR system are discussed in relation to research questions 1, 2, 3 and the existing literature. The detailed test results for the selected NoSQL document database and the selected relational database are discussed in relation to research questions 4, 5, 6 and 7 and the existing literature. This is followed by an evaluation of the research activity undertaken in this study using well-established design science assessment guidelines (Gill & Hevner 2013; Hevner et al. 2004).

In the last chapter of this PhD thesis, Chapter 7, the research problem and general research question that was addressed in this study is restated, and the overall study design and research activities undertaken to conduct this study are summarised. Then the key research findings of this study are summarised in relation to each of the research questions addressed in this study. Next, the key contributions of this study to theory and practice are discussed. Finally, the limitations of this study are acknowledged and suggestions are provided for future research that builds on this study.

1.8 Definition of Key Terms

In this section, the key terms that are used throughout this thesis are defined in terms of the context and scope of this research.

NoSQL: A new kind of database system that is emerged as a response to the need to overcome the limitations of relational databases mainly in terms of scalability and availability. The term is usually defined as an acronym for "Not Only SQL". (Cattell 2011).

NoSQL Document Database: A type of NoSQL database in which the data is stored as documents, mainly in a format like JSON.

Relational Database: Database systems store data in interrelated tables using normalisation which is introduced by Codd (1970).

Electronic Health Record (EHR): A digital record that holds the patient's healthcare-related data.

EHR System: The system that is responsible for managing EHRs for patients. In this research an EHR System refers to a large-scale, mainly national system that manages the collection and storage of EHRs for patients from birth to death.

EHR Sharing: It is one of the important features of EHR Systems that enables the sharing EHRs between an EHR system and healthcare providers. In the context of this research, the performance of EHR sharing is evaluated as a technical operation that involves the data retrieval of a patient’s EHRs for the purpose of simulating the sharing of EHRs in a large scale distributed EHR system.

1.9 Chapter Summary

This chapter introduced to the background to this study. The motivation for conducting for this study was described and justified in terms of the research problem that was addressed in this study. A general research question was framed within the context of the research problem identified. This general research question is broken down into the seven research questions investigated in this study. The research paradigm and methodological approach that guided the conduct of this research was outlined and justified. The delimitations of scope and the planned contributions of this research are highlighted. Then, finally, an outline of each subsequent thesis chapter is provided.

.

Chapter 2 - Literature Review

2.1 Introduction

This chapter provides an extensive overview of the relevant literature in order to demonstrate the gap in the literature and to provide the context and justification for the specific research problem being investigated. Then, the review of the literature provides a theoretical and conceptual foundation for this research, underpinning how this research can make a contribution to existing theory and practice. Figure 2.1 presents the structure of this chapter.

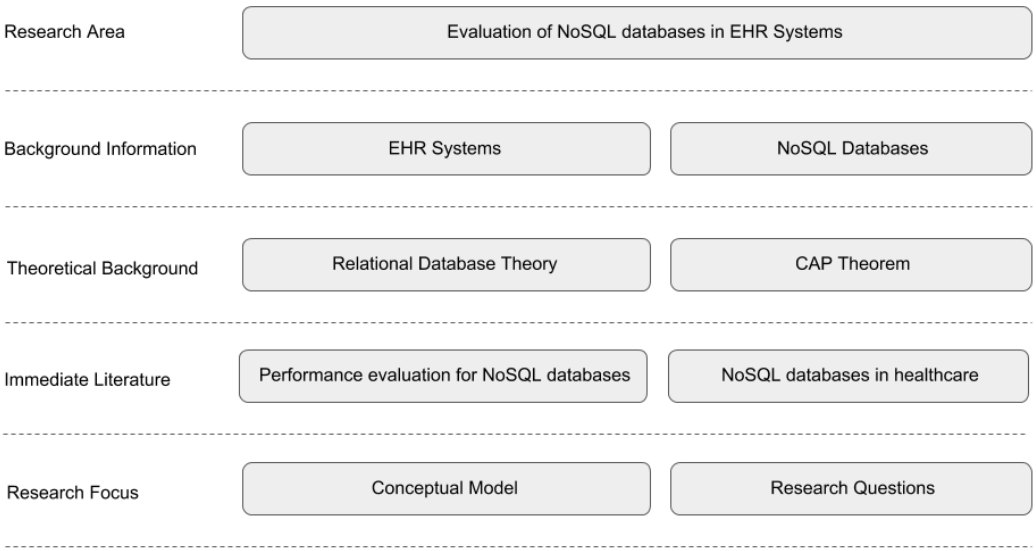


Figure 2.1 Structure of Chapter 2

The volume of healthcare data worldwide has increased rapidly in recent years. Furthermore, the diversity of healthcare data is expanding due to widespread dissemination of personal medical records systems digitally (Raghupathi & Raghupathi 2014). The emergence of technologies such as sensors and digitized 3D imaging etc is playing a greater role in healthcare and generating increased volumes and variety of healthcare data (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Raghupathi & Raghupathi 2014). Healthcare data is generally stored in relational databases. However, relational databases have limitations with regards to the current data and information needs of the healthcare sector as a whole. Hence, new and emerging database systems known as NoSQL databases could be a better fit for

managing distributed healthcare data sharing (Freire et al. 2016; Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016).

In this chapter, concepts of Electronic Health Records (EHR) and NoSQL database systems are introduced and explained, as these technologies provide the foundation for the IT artefacts which are developed and evaluated in this study. Recent research on these types of systems is reviewed and the important roles that EHRs and NoSQL database systems can play in the healthcare systems are presented. Then, Consistency, Availability and Partition Tolerance (CAP) theorem is discussed in terms of NoSQL document databases in healthcare; and the evaluation of the performance of NoSQL document databases is discussed in terms of basic database operations, scalability, data sharing and complex query capabilities in a healthcare domain. Finally this chapter identifies the gap in the literature regarding the evaluation of the performance of NoSQL document databases in EHR systems and justifies how this gap in the literature will be addressed in an overarching research question and a specific set of research questions.

2.2 Electronic Health Records (EHR)

The practice of storing healthcare information electronically emerged several decades ago in the 1990s because paper-based records could no longer meet the requirements of an advanced health care system (van Ginneken 2002). Electronically stored healthcare information has been identified by a number of different names such as Electronic Patient Records (EPR), Computerised Patient Records, Electronic Medical Records and Electronic Health Records (EHR) (ISO 2011; Narayan, Gagne & Safavi-Naini 2010).

While these terms might sometimes be used interchangeably, the National Health Service (NHS) suggests that EPR is “the record of the periodic care provided mainly by one institution”. On the other hand, EHR is defined as the collection of a patient’s health and healthcare information, from birth to death. According to these definitions, EHR is described as a collection of EPRs for a single individual (NHS 1998).

The International Organization for Standardization (ISO) defined EHR as “a repository of information regarding the status of a subject of care in a computer

processable form and, transmitted securely, accessible by multiple authorized users" (ISO 2004).

Based on these definitions, and in the context of this research, an electronic record that holds a patient's lifetime health-related information will be referred to as an EHR; and systems that handle operations on EHRs will be referred to as EHR systems.

2.2.1 EHR systems

EHR systems play an important role in improving healthcare service delivery by increasing quality and effectiveness of health services (Narayan, Gagne & Safavi-Naini 2010; van der Linden et al. 2009). When EHR systems are implemented at the national level and facilitates accumulation of healthcare data, EHR systems through data sharing can enable enhanced decision-making by health practitioners and health managers—including identification of effective treatments and pattern analysis (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Kruse et al. 2016; Raghupathi & Raghupathi 2014). In order to establish a foundation for the many significant benefits that can be realised, EHR systems should be designed to handle increasing data volume and diversity and facilitate sharing of healthcare data (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016). Therefore, this research focuses on large scale EHR systems that can be comparable to a national system rather than a system for a single healthcare service provider, such as a single hospital.

Many countries have developed their own national EHR system architecture. For example, Turkey has a national system called "Saglik-NET" which collects and centrally stores a wide range of medical data (Dogac et al. 2011; Kose et al. 2008). In the Netherlands, the data is stored locally and a central system called a "National Switch Point (NSP)" handles the links to the data and allows access to information by various services in the health network (Bacelar-Silva et al. 2011). Austria and Germany are also establishing their own nationwide EHR systems (Hoerbst et al. 2010).

In England, The National Care Record Service (NCRS) enables access to patients' EHRs in a national system called "Spine" (Bacelar-Silva et al. 2011). Authorised professionals can access summary records of patients which include basic information such as date of birth, name, contact information, allergies, etc.

2.2.2 EHR Systems in Australia

In Australia, there is a significant effort underway in establishing a Personally Controlled Electronic Health Record (PCEHR) system (Vest 2012). This system is now named as "My Health Record". The Australian Digital Health Agency (ADHA) (previously known as The National E-Health Transition Authority (NEHTA)) is working on establishing governing standards for the My Health Record system. The Australian Digital Health Agency is responsible for digital health activities in Australia and also provides all stakeholders of digital health, including healthcare professionals, patients and implementers with relevant resources and information (Australian Digital Health Agency 2015).

Furthermore, as a part of its role, the Australian Digital Health Agency publishes documents guiding the community and software vendors on technical information for infrastructure, integration, and clinical document and messaging standards (Australian Digital Health Agency 2015; Pearce & Haikerwal 2010).

2.2.3 Electronic Health Record sharing functionality in EHR systems

The literature emphasises the importance of the information sharing function of an EHR system in improving healthcare outcomes. Iakovidis (1998) suggests that the purpose of an EHR system is to support continuity of care; and van der Linden et al. (2009) note that the primary purpose of an EHR system is the support of continuing, efficient and quality integrated health care. Narayan et al. (2010) suggest that a life-time health record system is established to keep track of all healthcare-related information of individuals from birth to death to allow efficient, consistent and universal sharing of health information. Previous studies also suggest that additional purposes of an EHR system include providing support in development of health policies, medical education and advanced research (Heard 2006; Iakovidis 1998; Murphy, Hanken & Waters 1999).

Harvard Business Review suggests that having comprehensive EHR systems and universal access to these systems are necessary for the best medical care in the 21st century, as well as delivering advances in health care (for example, precision medicine) (Pearl 2017). EHR sharing functionality has become a topic of interest for researchers due to the significant benefits that can be realised. Furthermore sharing EHRs with remote locations and even between physicians and pharmacies have also

been a subject of previous research to explore the extent of its usefulness. (Ibrahim, Mahmood & Singhal 2016; Keller et al. 2015; Pussewalage & Oleshchuk 2016).

Therefore, EHR sharing has the potential for significantly improving healthcare outcomes both at the patient level and at the national level. Furthermore, EHR sharing enables a platform that would facilitate providing valuable information to inform healthcare policy, medical practice and training, and medical research. EHR sharing in this research is delimited in scope to EHR sharing through patient EHR retrieval in a large scale EHR system.

2.2.4 Importance of EHR systems for Healthcare

According to the US Institute of Medicine, an EHR system improves patient safety, supports efficient patient care delivery and improves the efficiency of healthcare services (Englebardt & Nelson 2002; Kohn, Corrigan & Donaldson 2000). Schiff et al. (2003) note that patient safety and quality of healthcare can be increased by sharing EHRs amongst healthcare facilities.

Halamka et al (2005) demonstrated that an uncoordinated approach to managing medical records leads to a significant waste of time and medical errors. Previous literature suggests that implementing a fully-functioning national EHR system with the participation of all healthcare organisations could lead to a USD77.8 billion benefit for the United States (Halamka et al. 2005; Schiff et al. 2003; Walker et al. 2005; Yasnoff et al. 2004).

Brazil and Switzerland has adopted e-health strategies to facilitate interoperability and sharing of healthcare information across healthcare service providers inspired by the positive outcomes of the EHR system implementations of other countries to achieve better quality of care and greater efficiency (Chaim, Oliveira & Araújo 2017; De Pietro & Francetic 2017).

2.2.5 Technological issues affecting EHR systems

Establishing a nationwide EHR system requires a significant investment, as well as extensive system design and project management (Hoerbst et al. 2010; Pearce & Haikerwal 2010; Vest 2012). Poorly designed architecture not only poses a substantial failure risk for the implementation of EHR systems, but can also result in significant losses of financial and human resources (Pearce & Haikerwal 2010).

There are a number of obstacles and challenges that exist in relation to establishing large scale national EHR systems described previously in the literature, such as standardisation of vocabulary, security, privacy and data quality (Gunter & Terry 2005). In addition to these matters which are extensively covered in various publications, Orfanidis, Bamidis and Eaglestone (2004) claim that the expanding size of healthcare data also creates an obstacle for EHR systems. Blobel (2006) suggests that an EHR system which allows the exchange of health information should be scalable, flexible and portable, with Internet access.

Patients' records can contain different types of documents such as full-text reports, test results, images, prescriptions, etc. This heterogeneous nature of healthcare data, together with increasing size and the requirement of scalability for EHR systems, are also considered to be a major bottleneck for EHR system implementations. Most current EHR systems are based on relational databases which struggled to support unstructured data types (Dolin et al. 2006; Guo et al. 2005; Guo et al. 2004; Jin, Deyu & Xianrong 2011; Schmitt & Majchrzak 2012; Takeda et al. 2000).

Healthcare data is changing over time and the value that can be derived from an EHR system cannot be underestimated. Database systems used in healthcare should be able to support efficient parallel processing over large volumes of data to discover patterns, support decision-making, development of effective treatments and management of health policies (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016). Moreover, database systems should be flexible enough to adapt to the changing structure of healthcare data, provide high availability, and be easy to maintain in order to achieve a great range of benefits (Freire et al. 2016). These requirements cannot be fulfilled entirely by relational database systems and a new approach to data management should be considered in the healthcare domain to address the shortcomings of relational databases in relation to changing demands of healthcare data.

A recent popular term, 'big data', has been defined by multiple of characteristics of data such as volume, velocity and variety based on a report by Laney (2001) relating to the challenges and opportunities of increased data . Volume refers to the scale or quantity of data, velocity is the speed of data and the term 'variety' is used to emphasise various forms of data generally coming from different sources. Healthcare data clearly shows these characteristics of big data (Kruse et al. 2016; Raghupathi &

Raghupathi 2014). Existing literature demonstrates that rich healthcare data and EHR systems have significant potential for improving clinical decision support; and for improving population health management using vast data analytics (Hermon & Williams 2014; Raghupathi & Raghupathi 2014). However, it is difficult or impossible to manage and analyse large healthcare datasets with traditional or common data management tools and methods (Raghupathi & Raghupathi 2014; Sun & Reddy 2013).

Challenges and changing requirements for data management in general has led to the emergence of new forms of data-related systems to handle 'big data' in multiple aspects, including capturing, transformation, management, analysis and so on (MarkLogic 2014; Raghupathi & Raghupathi 2014; Stonebraker & Cattell 2011). A new category of non-relational databases, NoSQL databases, has emerged as a response to meeting big data management requirements (MarkLogic 2014; Mason 2015; Sadalage & Fowler 2012; Stonebraker & Cattell 2011).

2.3 NoSQL databases

NoSQL is a term often used to describe the category of non-relational databases (Li & Manoharan 2013; Sadalage & Fowler 2012). A NoSQL database, also known as a distributed data store, is capable of scaling large datasets with no single point of failure (Ferreira, Calil & Mello 2013). Data may span server nodes, racks, and even multiple data centres. The emergence of NoSQL databases has been heavily influenced by a seminal whitepaper published by Google about its BigTable system and Amazon's related system called Dynamo (Cattell 2011; Featherston 2010; Li & Manoharan 2013; Wu 2011).

NoSQL database technology depends on horizontal scalability which enables increased performance and capacity by increasing the number of nodes, rather than increasing the computer power of a single node (Abramova & Bernardino 2013; Yassien & Desouky 2016). Thus, NoSQL databases offer the high performance required for managing large data sets (Aboutorabi et al. 2015). NoSQL databases also offer another significant advantage over traditional relational databases by providing more data model flexibility for the types of data stored. In most cases there are no strict pre-defined schema requirements for NoSQL databases, in contrast to relational databases (Aboutorabi et al. 2015; Li & Manoharan 2013).

NoSQL database systems have attracted a lot of attention from industry and researchers due to the demand for distributed database systems capable of delivering high performance access to large volumes of data across geographical locations without requiring significant effort for scaling and tuning (Ferreira, Calil & Mello 2013). Figure 2.2 shows how NoSQL is trending in terms of attention in contrast to relational databases between 2004 and 2017, based on Google searches.

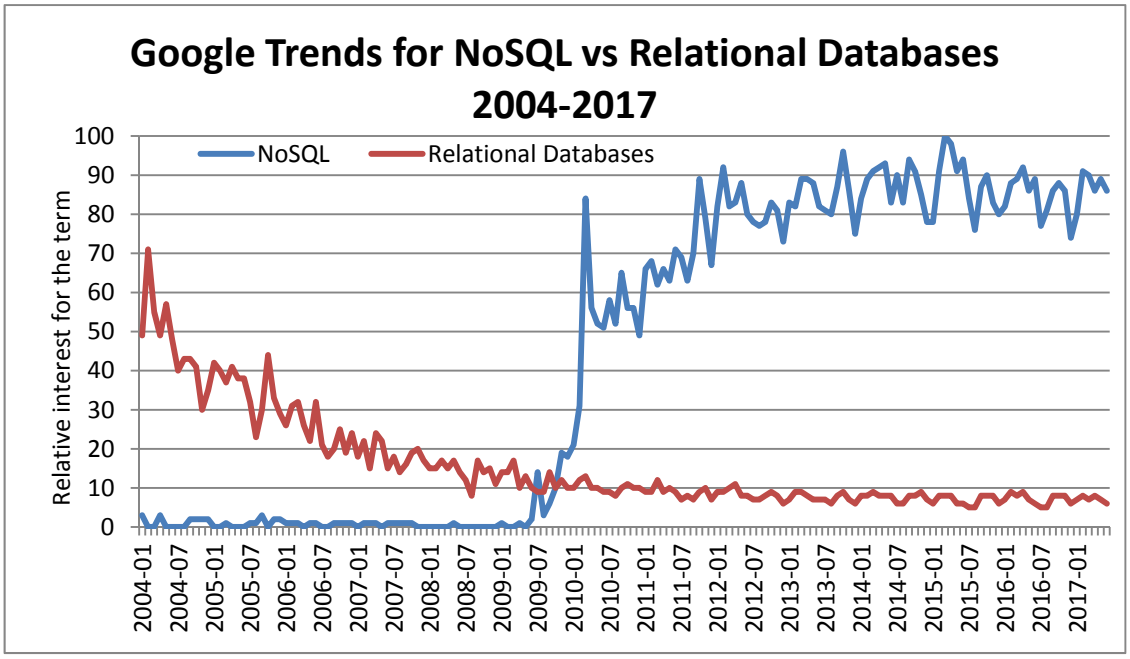


Figure 2.2: Google search trends NoSQL databases versus Relational Databases

2.3.1 Types of NoSQL databases

There were more than 200 NoSQL databases available as at December 2017. Based on the data model used, NoSQL databases can be grouped into four main categories: (1) Key-value store; (2) Document store; (3) Column-family, and (4) Graph database (Abramova & Bernardino 2013; Edlich 2017; Haseeb & Pattun 2017; Leavitt 2010; Yassien & Desouky 2016).

Although there are some other categories used for NoSQL database types such as multimodel, NoSQL databases mainly demonstrate characteristics of one or more of the four main categories listed above. Each NoSQL database type has its own data structure, strength, and typical use cases which are discussed in the following subsections.

2.3.1.1 Key-Value Store

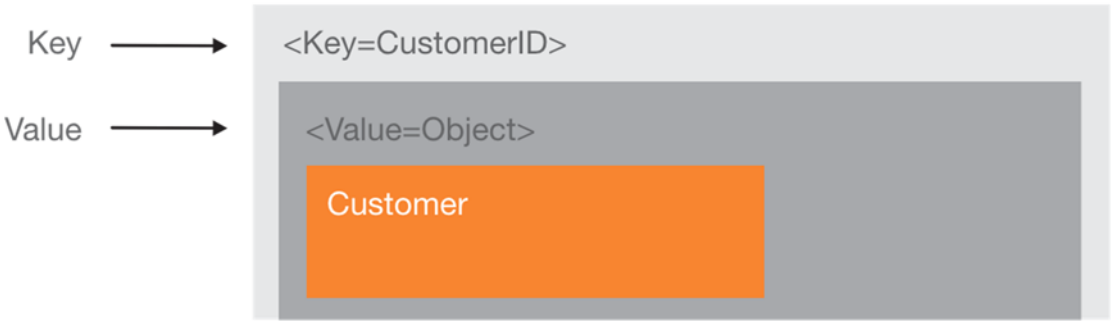


Figure 2.3: Key-Value Store representation (Adapted from Sadalage (2014)).

In key-value stores, all data is stored as key-value pairs, in which the keys are unique values that are used to access the information stored in values (Moniruzzaman & Hossain 2013; Sadalage & Fowler 2012; Sumbaly et al. 2012; Yassien & Desouky 2016). A key-value store is the simplest form of NoSQL databases to allow fast retrieval of values which can be a string, list or any other object. Redis, Amazon SimpleDB, Voldemort and DynamoDB are examples of key-value stores (Leavitt 2010; Moniruzzaman & Hossain 2013).

2.3.1.2 Document Store

Figure 2.4: A sample representation of data stored in a document store (Adapted from Sadalage (2014)

Document stores are essentially similar to key-value stores. However, the values are usually documents in known formats such as XML or JSON. Documents (values) can be a structured or unstructured document, as well as de-normalised (aggregate) database entries. Document stores are also known as 'aggregate databases'. Contents of documents may vary between records. This allows flexibility for the types of data stored in document stores, a feature which has become increasingly important for healthcare data given its increasing volume and variety. Well-known examples of document stores are MongoDB and Couchbase (Abramova & Bernardino 2013; Dede et al. 2013; Li & Manoharan 2013; MarkLogic 2014; Moniruzzaman & Hossain 2013).

2.3.1.3 Column Family

For the column family type of NoSQL databases, data is stored in columns, however the columns are not required to be defined at the beginning and there may be countless numbers of columns which may also be organized in groups called supercolumns (Leavitt 2010; Yassien & Desouky 2016).

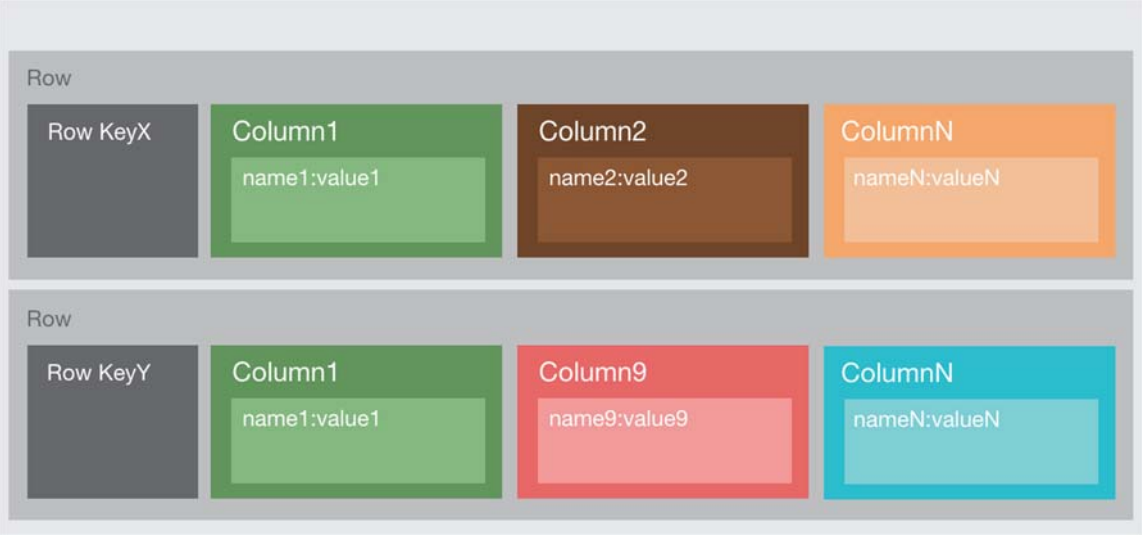


Figure 2.5: A sample data structure representation of column family type of NoSQL database (Adapted from Sadalage (2014)

The design of column family type NoSQL databases are mainly influenced by the work described in Google's Bigtable paper; and Cassandra and HBase are examples of known column family type NoSQL databases which have been implemented in practice (Li & Manoharan 2013; MarkLogic 2014).

2.3.1.4 Graph Databases

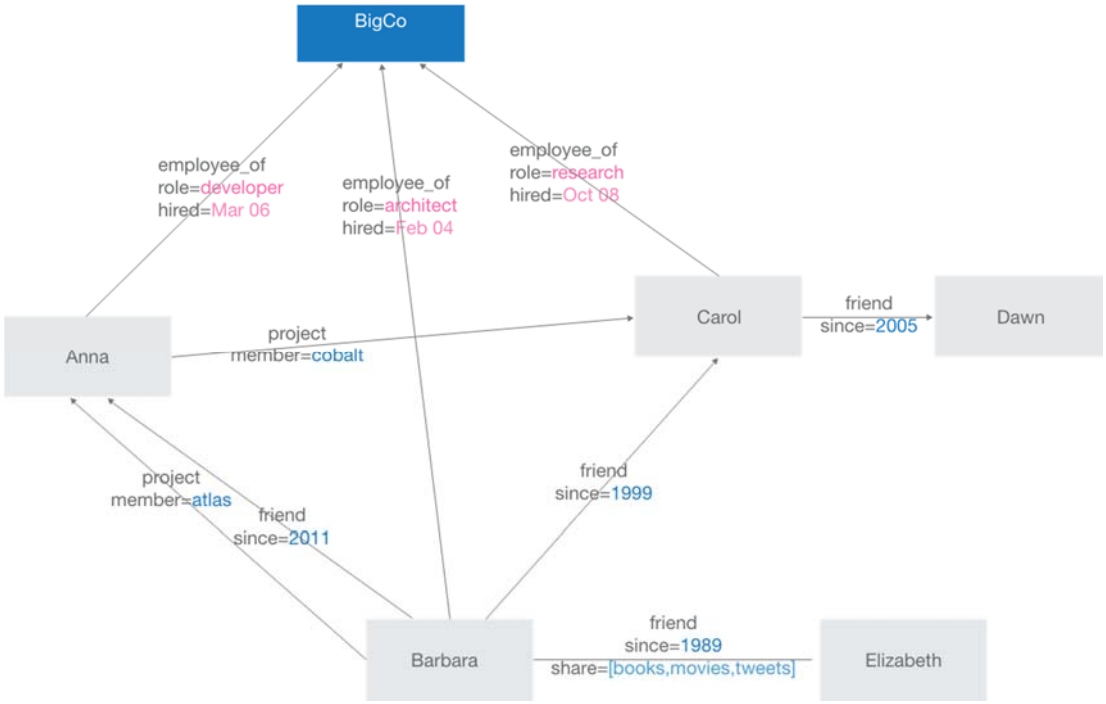


Figure 2.6: Data structure representation for a graph database (Adapted from Sadalage (2014)

Graph databases are examples of data stores that can store and handle graph type of data such as social network relations. Neo4j and InfoGrid are examples of graph databases (Abramova & Bernardino 2013; Moniruzzaman & Hossain 2013; Yassien & Desouky 2016).

Typical use cases and example applications for these four main types of NoSQL databases are presented in Table 2.1.

NoSQL Database Type	Typical Use Case	Examples
Key-Value	Real-time processing of extremely large data, horizontal scalability, high reliability and high availability, primary query mechanism is key-based lookup.	Session management, real-time bidding, online trading
Document	Applications that need flexible schema, semi-structured, nested hierarchical data	Healthcare records and derivative securities
Column Family	Applications requiring flexible, evolving database schema, tolerance to network failure and temporary data inconsistency	Mixed content management, stock trading
Graph	Applications with queries that require graph traversals	Social media applications, recommendation engines

Table 2.1. Typical Use Cases and Example Applications for NoSQL database types (Adapted from (Gudivada, Rao & Raghavan 2016)

The data model and NoSQL database type that will be used for a particular system depends on the use case; and each of these NoSQL database types have their own characteristics, as summarised in Table 2.1. Therefore, it is essential to determine the most feasible data model and NoSQL database type for specific system applications. However, understanding the architecture of scalable databases and deciding the most suitable candidate database technology that satisfies the needs of an application is a challenging tasks due to the complexity of comparing different types of NoSQL databases (Gorton, Klein & Nurgaliev 2015). The data model and NoSQL database type considered to be most suitable for EHR systems are discussed in detail later in this chapter.

2.4 Theoretical Background

The kernel theories and practice knowledge are discussed in the subsequent subsections that informed the design theory developed in the research process and

activities undertaken in this study to build and evaluate artefacts to solve a real world problem.

2.4.1 Relational Database Theory

The relational model and database theory known as a product of E.F. Codd has existed since the 1970s. Since then, the relational model has been adopted widely in industry and many of the current modern day commercial database systems are influenced by the work of Codd (Suciu 2001; Yassien & Desouky 2016). In database theory, Codd (1970) suggests that the data stored in large shared data banks can be defined and organised based on interrelationships of data, and redundancy and consistency problems can be eliminated by normalisation of data. Normalisation is a procedure for organising data into relational views, eliminating the need for copies of the same data and establishing a link between data groups using primary keys (Abiteboul, Hull & Vianu 1995; Codd 1970).

Although there have been many supportive theories and models developed by Bernstein (1976), Fagin (1977), Mendelzon (1984) and Papadimitriou (1979), the fundamentals of relational databases have remained unchanged for decades (Chen 1976; Suciu 2001; Ullman 1987). However, the link between theory and relevance in practice in relation to database systems has weakened over time (Abelló, Ferrarons & Romero 2011; Badia & Lemire 2011; Suciu 2001; Vianu 2001). The emergence of high-speed networks, fast commodity hardware and the increasing amounts of unstructured or semi-structured data has created the necessity for relational database theory and designs to be adapted in order to meet the needs of today's business environment. The impedance mismatch between relational data structures and the in-memory data structures of an application has driven the need for different ways of storing data that are not restricted by a relational model (Sadalage 2014). Using NoSQL databases allows applications to be developed without having to convert in-memory structures to relational structures of a relational database. Valduriez (2011), Jin, Deyu and Xianrong (2011) and Konishetty (2012) explored the principles underpinning distributed database management systems and the practical implementations of NoSQL databases have helped in establishing a better link between theory and what is required to meet the needs of practice in terms of data management (Badia & Lemire 2011; Konishetty et al. 2012; Valduriez 2011; Vianu 2001).

2.4.2 Advantages of NoSQL document databases over relational databases

Database systems are crucial for all sorts of data-intensive applications which store and manage huge amounts of data. Modern applications such as high-traffic web sites or large enterprise systems require new approaches to data storage in order to achieve higher performance and higher availability than is possible with traditional relational database management systems (RDBMS). This is particularly the case when it involves high concurrent numbers of transactions and large amounts of data (Klein et al. 2014; Mengchen 2011; Parker, Poe & Vrbsky 2013). NoSQL databases have emerged as a response to this requirement and they have significant differences compared to relational databases, which are summarised in Table 2.2 below.

Relational Databases	NoSQL Databases
Structured	Semi- or Non-structured
Difficult and Manual Scaling	Easy and Automatic Scaling
Share resources	Shared-nothing
Possible Single Point of Failure	High Availability
Strong Consistency	Weak Consistency
Mostly Commercial, Expensive	Many Open Source Alternatives

Table 2.2. Comparison of Key Differences between NoSQL Databases and Relational Databases

NoSQL databases adopt a shared-nothing architecture which enables easy scalability. Furthermore, NoSQL databases favour availability and partition tolerance over consistency, as opposed to strong consistency approach used in relational databases. NoSQL databases have many open source alternatives and they have flexible data schema which allows handling of semi- or non-structured data. Previous studies suggest that NoSQL databases have many technical and financial advantages over relational databases for large scale data intensive applications due to these differences (Borkar, Carey & Li 2012; Klein et al. 2014; Manyam et al. 2012; Meijer & Bierman 2011). The main advantages of NoSQL databases are presented in the five following sub-sections: (1) Performance; (2) Scalability; (3) High Availability; (4) Flexible Data Model; and (5) Open Source Availability.

2.4.2.1 Performance

One of the advantages of NoSQL databases is high performance in terms of higher number of operations per second and lower execution times of database operations. NoSQL databases can achieve higher performance than relational databases due to simpler and mostly de-normalised data structures and their distributed nature, and the performance difference can be significant in larger datasets depending on the use case (Aboutorabi et al. 2015; Freire et al. 2016).

2.4.2.2 Scalability

A significant advantage of NoSQL databases is that they allow scaling up to large datasets without any changes in the overall structure of data model or architecture (Ferreira, Calil & Mello 2013). Hardware requirements and costs can grow in a linear manner as storage requirements grow. Therefore, cost-effective scaling up is made possible and high initial investment in hardware requirements is avoided (Lakshman & Malik 2010).

Relational database systems mostly rely on purchasing more expensive and powerful servers in order to increase capacity. In contrast, distributed data storage systems such as NoSQL database systems are based on a shared-nothing approach (Stonebraker & Cattell 2011). Capacity can be increased by adding more commodity servers dynamically. The redistribution of the data occurs on the fly and seamlessly without reconfiguration or a decrease in performance. This aspect is one of the most important advantages of NoSQL database systems over relational database systems (Pokorny 2011).

2.4.2.3 High Availability

Furthermore, achieving high availability by maintaining a number of replications, enabling high performance on transactions using distributed algorithms is also another major advantage of distributed data storage systems such as NoSQL database systems have over relational database systems (Featherston 2010; Mengchen 2011). In order to achieve this, NoSQL database systems trade-off consistency for availability, an aspect which is discussed in more detail later in this chapter (Dede et al. 2013).

2.4.2.4 Flexible Data Model

Data modelling in relational databases relies on tables and relations between tables, a pre-defined set of columns for each table and strict requirements for data stored in

each column. However, NoSQL databases have flexible data models where all types of structured, semi-structured and unstructured data can be stored and processed, which eliminates the requirement for a pre-defined data structure and schema. Therefore, NoSQL databases are often referred to as schema-less (Freire et al. 2016).

This feature gives the flexibility to handle the changing structure of the data stored in NoSQL databases, which is an important requirement for managing healthcare data. Changing requirements and improved technologies increasingly means different forms of healthcare data need to be stored; and any change in information requirements can be easily implemented without any changes in database structure in NoSQL databases (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016).

2.4.2.5 Open Source Availability

In addition to these advantages of NoSQL databases over relational databases, it is also important to note that there are many open source NoSQL database alternatives available in the marketplace. This may help in reducing the overall cost of implementation by achieving lower cost per terabyte and making customisation of a database system possible as open source NoSQL database solutions provide access to the source code (Leavitt 2010; Stonebraker & Cattell 2011).

2.4.3 NoSQL Data Modelling versus Relational Data Modelling

Data modelling is an important topic when considering the suitability of NoSQL databases for healthcare applications such as EHR systems. Each type of NoSQL database has its own strengths and weaknesses, therefore it is necessary to determine the type and data model suitable for the use case. Figure 2.7 compares the four main types of NoSQL database in terms of their ability to accommodate varying degrees of scalability and complexity of a data model. Key value stores can accommodate the greatest level of scalability, while graph databases can accommodate the greatest level of complexity in a data model. Document databases can address the data model complexity of EHR systems while providing an appropriate level of scalability.

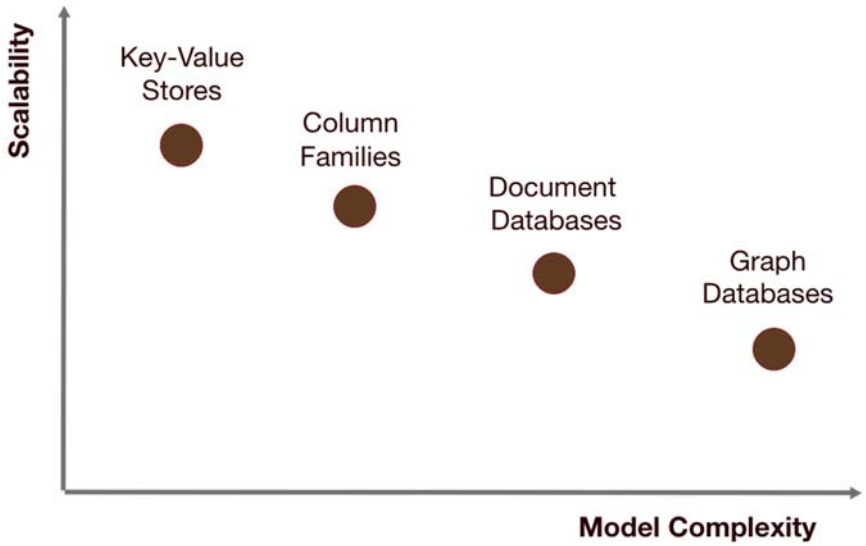


Figure 2.7. Comparison of NoSQL databases based on model complexity and scalability (Adapted from Hsieh (2014)

Goli-Malekabadi, Sargolzaei-Javan and Akbari (2016) evaluated four main types of NoSQL databases to determine which NoSQL database would provide the best approach for storing healthcare data. The match between the characteristics of healthcare data and the characteristics of each type of NoSQL database are summarised in Table 2.3.

Healthcare data characteristics	NoSQL database characteristics			
	Key-Value	**Document**	**Column Family**	**Graph**
Mostly document based	Storing key and value	Storing of documents	Storing key and its value	Storing nodes and relationships
Different types of data	Flat data models	Storing different types	Storing different types	Storing different types
Frequent read and write	Suitable for frequent write operations	Suitable for frequent read and write operations	Suitable for frequent read from different columns	-
Query in several fields	Query by key	Query by any field	Query by limited number of columns	Query by nodes

Table 2.3. Comparison of healthcare data and NoSQL database characteristics (Adapted from Goli-Malekabadi, Sargolzaei-Javan and Akbari (2016))

Consistent with suggested examples provided by Gudivada, Rao and Raghavan (2016), Goli-Malekabadi, Sargolzaei-Javan and Akbari (2016) also concluded that document databases are suitable for storing healthcare records.

A whitepaper by Couchbase (2016), provides a good example of how a document data model compared to a relational data model can be developed for storing a user as an entity as shown in Figure 2.8, Figure 2.9 and Figure 2.10.

Storing imaginary user data in a relational data model requires normalisation and would require six rows in three tables as visualised in Figure 2.8.

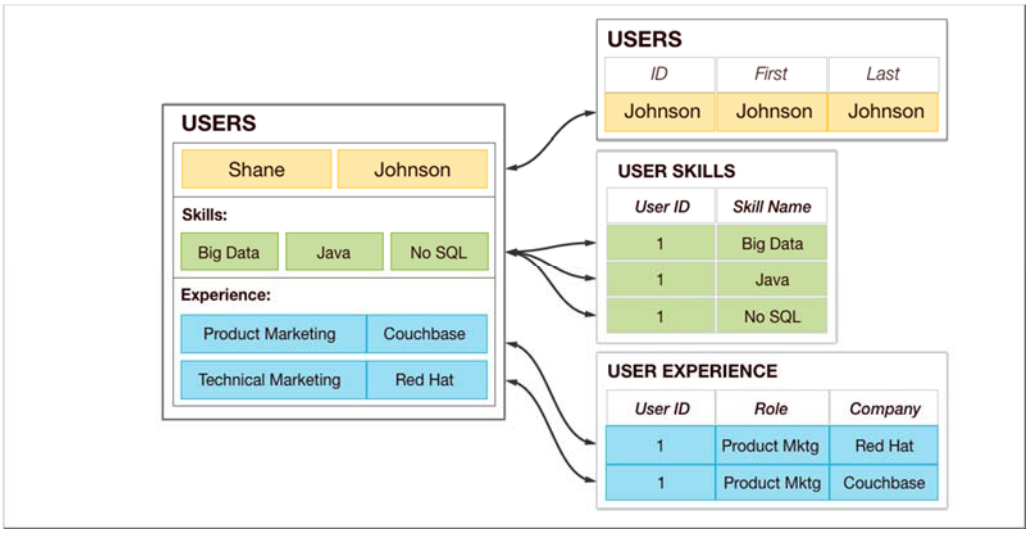


Figure 2.8: Diagram showing a sample user data in relational model (Adapted from Couchbase (2016)

As the data is split into three tables, reading this data would require generation of the following result set in Figure 2.10 that has 6 rows and duplicate values and requires filtering to achieve intended results.

Shane	Jhonson	Big Data	Product Marketing	Couchbase
Shane	Jhonson	Big Data	Technical Marketing	Red Hat
Shane	Jhonson	Java	Product Marketing	Couchbase
Shane	Jhonson	Java	Technical Marketing	Red Hat
Shane	Jhonson	No SQL	Product Marketing	Couchbase
Shane	Jhonson	No SQL	Technical Marketing	Red Hat

Figure 2.9: Initial result set for querying a sample user data in relational model (Adapted from Couchbase (2016)

In contrast, in a document-oriented NoSQL database, the same sample user data can be stored in one JSON document, as presented in Figure 2.10, and can be queried as a single record—which eliminates overheads and simplifies application development.

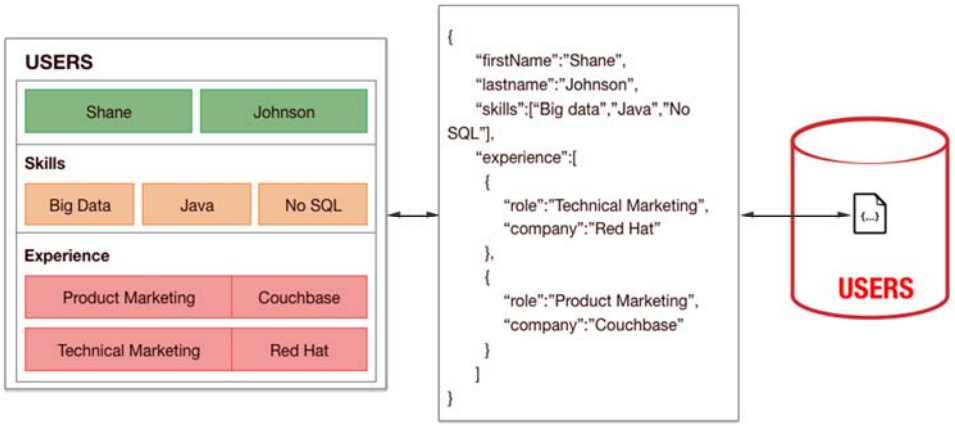


Figure 2.10: Result set for querying a sample user data in document data model (Adapted from Couchbase (2016)

Vera et al. (2015) compared conceptual data models for relational databases with conceptual data models for document-oriented NoSQL databases. They named the

nested data model for a NoSQL database (shown in Figure 2.11) an embedded document model.

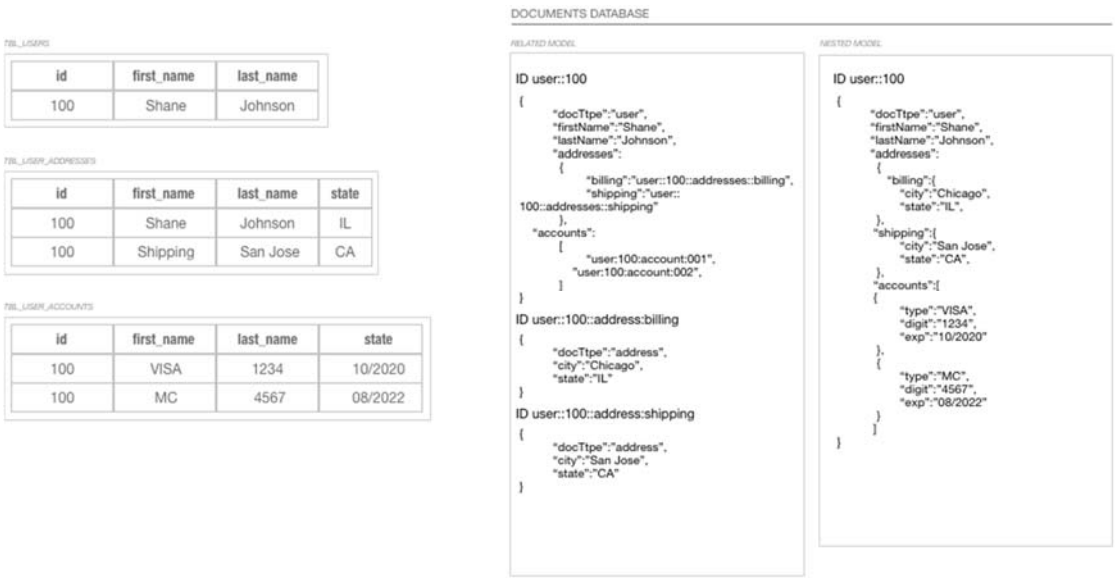


Figure 2.11: Related and Nested Document Database Models compared to Relational Database Model for sample user data (Adapted from Segleau (2016))

As this research focuses on evaluating the performance of database operations, scalability, EHR sharing and data analysis (complex querying) capability, an embedded document data model is considered as an appropriate data modelling approach. An embedded document model allows all of the required details saved into one document, thus eliminating the relations for document sections. Therefore, NoSQL data model used in this research is based on an aggregate oriented, embedded document model. This research focuses on one NoSQL database type that is most suitable for storing EHR data in conducting a performance evaluation of a NoSQL document database in terms of basic database operations and scalability, EHR sharing and data analysis (complex querying) comparative to a relational database.

2.4.4 Determining EHR data elements for NoSQL and Relational Data Models

An important requirement for data models is that they need to be established based on the type of information needed be stored. As this research focuses on the Australian healthcare domain, relevant datasets and data elements related to Australian healthcare domain were identified. The Australian Institute of Health and Welfare has published the National Health Data Dictionary (NHDD) on their website, which helps in

establishing standards for data collection and reporting for Australian healthcare providers (AIHW 2015). In the NHDD, national minimum data sets, along with their attributes, are defined and these guided the establishment of an appropriate document data model and a relational data model in line with the main aims of this research.

There are various archetype-based EHR models such as openEHR, ISO 13606 and HL7-CDA (Frade et al. 2013; Sundvall et al. 2017).

The literature suggest that using multilevel archetype-based models such as openEHR that involve complex data structures causes difficulties on the database operations of storing, retrieving and querying of EHRs (Frade et al. 2013; Freire et al. 2016). Furthermore previous studies have compared performance of NoSQL and relational databases using archetype-based EHR models by storing EHRs as XML documents in relational databases due to the complexity of the document structure of EHRs and requirement of data transformation in order to store non-relational data in a relational database (Sundvall et al. 2017).

Therefore, in this study, a simpler data structure justified as this allowed for a meaningful comparison of the performance of both a NoSQL document database and a relational database and is sufficient to cover the NHDD minimum datasets used instead of archetype-based systems.

After establishing data models for NoSQL and relational databases based on the NHDD minimum data sets, relevant publicly-available healthcare statistics are identified (AIHW 2015, 2016). Using these statistics, random healthcare data is generated to populate the data models based on the NHDD minimum data sets which reflect the data characteristics of the Australian healthcare domain.

2.4.5 CAP Theorem and NoSQL Databases

NoSQL database systems have received much attention from the research community (Cattell 2011; Escriva, Wong & Sirer 2012; Floratou et al. 2012; Lee, Tang & Choi 2013; Schram & Anderson 2012). The previous literature suggests that current research focuses on the scalability, fault-tolerance and performance advantages of the NoSQL/distributed database systems, while criticising the weak consistency approach of these types of database systems (Agrawal, Das & El Abbadi 2011). The issue of consistency with NoSQL databases is explained in the context of CAP (consistency,

availability, and partition-tolerance) theorem (Agrawal, Das & El Abbadi 2011; Bermbach & Tai 2011).

CAP theorem, introduced by Eric Brewer in 2000, suggests that there is always a trade-off between consistency, availability and partition-tolerance. In the context of CAP theorem, consistency means that each server returns the right response to each request; availability means that each request will eventually receive a response; and partition-tolerance means that the service can continue operating normally even when communication between some of the nodes are lost. The underlying idea in this theorem is that the communication between servers is prone to network errors and failures, thus it is not possible to have all three features (consistency, availability, partition tolerance) working together perfectly (Gilbert & Lynch 2012).

2.4.6 ACID Properties and NoSQL Databases

Gray (1981) suggested a number of properties for database systems to achieve reliable transaction processing, commonly known as Atomicity, Consistency, Isolation and Durability (ACID). Atomicity means a transaction is either completed entirely or failed, i.e. there is no partial completion in any transaction. Consistency is the property that guarantees that every transaction changes a database into a valid new state, incorporating all rules, constraints and triggers, etc. Isolation means that each transaction happens totally independent of each other and transactions do not affect each other while being executed. Durability is the property that means if a transaction has been completed, the new state of a database is guaranteed to be durable regardless of any potential failures such as power loss, network errors, etc. afterwards (Gray 1981; Sattar, Lorenzen & Nallamaddi 2013).

Due to their distributed nature without a coordinator or master node, and based on the CAP theorem, NoSQL databases cannot offer strong consistency models like relational databases can do. Therefore, while having many advantages such as high availability and easy scalability, NoSQL databases cannot have all strong ACID properties. NoSQL databases focus on the BASE principal instead, which stands for **B**asically **A**vailable, **S**oft state and **E**ventually consistent. The BASE principal implies that the system can continue working as usual in case of a failure due to the distributed nature of NoSQL databases. For NoSQL databases, the BASE principal ensures that even though there is no guarantee of consistency at any given point of time, data will

eventually be consistent at some point in time (Bailis & Ghodsi 2013; Moniruzzaman & Hossain 2013).

Figure 2.12 summarises data models in terms of two possible combinations of CAP theorem to categorise the strengths of three main types of NoSQL databases comparative to relational databases. This emphasises the trade off against the strengths of a particular type of database that is made when choosing either NoSQL database or a relational database.

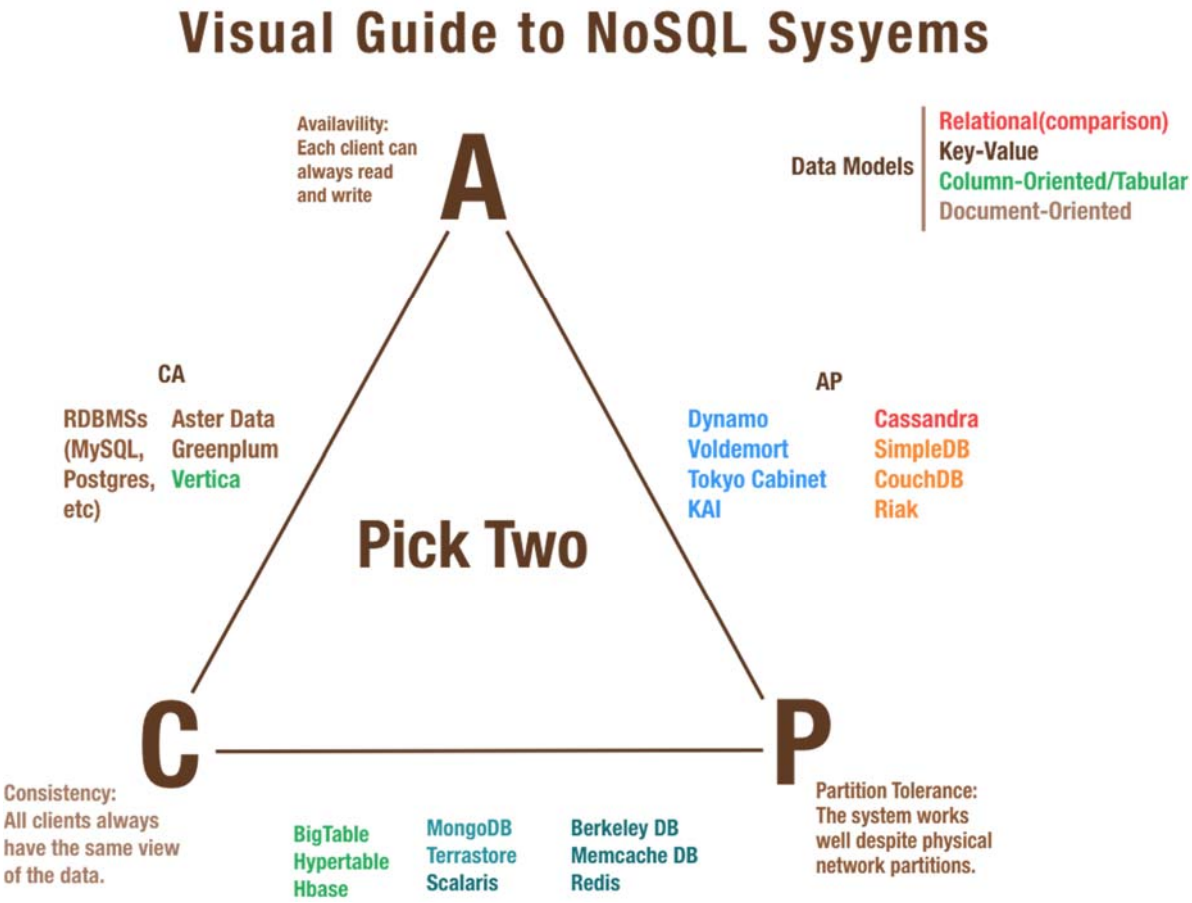


Figure 2.12: Comparison of the three main data model types, Key-Value, Column Family, and Document Oriented, used in NoSQL databases with relational databases in terms of CAP Theorem (Adapted from Fernando (2016)

Figure 2.12 shows that the strengths of relational data models are in being able to deliver consistency and, to a lesser extent, availability; whereas strengths of NoSQL key value, column oriented, tabular and document oriented data models are in being

able to deliver consistency and partition tolerance or availability and partition tolerance.

In an eventually consistent NoSQL database, data read by clients immediately after being updated may be an out-dated version as all nodes have not been updated at once. However, some NoSQL databases such as Cassandra offers different levels of consistency and users can select the level of consistency they require for each transaction. Furthermore, previous studies have shown that the inconsistency windows for many NoSQL databases are less than a second. Therefore, eventual consistency model suggested by NoSQL databases is claimed to be sufficient in most use cases (Bailis & Ghodsi 2013)

Google published a paper on 'Spanner', Google's globally distributed database system, which mentions the possibility of achieving transaction control, consistency and replication without sacrificing high-availability. Furthermore, there are other papers suggesting that it might be possible to achieve consistency and high availability together to an extent that distributed databases can match the properties of current relational databases (Bailis et al. 2013; Corbett et al. 2013).

2.5 Suitability of NoSQL databases for EHR systems

There is no unanimous agreement in the literature on the overall superiority of NoSQL databases over relational databases such as their generic suitability for data-intensive applications. However, past empirical research demonstrates that the type and the requirements of the application dramatically determines the suitability of the use of NoSQL databases (Badia & Lemire 2011; Jin, Deyu & Xianrong 2011; Parker, Poe & Vrbsky 2013; Vianu 2001).

Table 2.4 summarises the main requirements of EHR systems and highlights how NoSQL database system features can address these requirements. Clearly the features of NoSQL database systems align well with the main requirements of EHR systems.

EHR requirement	NoSQL database feature
Size of healthcare data increased over time, data size is a bottleneck for EHR systems	NoSQL databases based on horizontal scalability allows easy and automatic scaling
Healthcare data includes free-text notes, images and other complex data. Heterogeneity of healthcare data requires new solutions	NoSQL databases accommodate Flexible data models which allow unstructured or semi-structured data to be stored easily
Healthcare data should always be accessible for continuity of healthcare services	NoSQL databases provide high availability through their distributed nature and replication of data
Healthcare data sharing requires access to EHRs from multiple locations which requires a high-performance system to respond data access request in a timely manner	NoSQL databases offer higher performance compared to relational databases in many use cases because of their distributed and shared nothing architecture, and simplified method of data access.

Table 2.4. Comparison of EHR requirements and NoSQL database features that address these requirements

2.5.1 CAP Theorem and NoSQL Databases in EHR systems

Relational databases with strong consistency features are more suitable for an update-intensive database application where consistency is very important, such as a stock exchange system that handles financial transactions from all over the world and where milliseconds in processing time matter.

However, the other two aspects of CAP theorem, availability and partition-tolerance, are particularly important in the healthcare context. Schmitt and Majchrzak (2012) suggest that the nature and purpose of healthcare data requires high availability and distributed data management to enable access to healthcare information whenever needed, even in an event of crisis when data centres fail (Schmitt & Majchrzak 2012). Other literature emphasise significant benefits of EHR systems when data analysis using parallel processing is possible, such as medical research involving pattern recognition and effective treatment (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016). Therefore, partition-tolerance is another key aspect for managing and improving the effective use of healthcare data. Availability and partition-tolerance are

particular strengths of NoSQL databases, hence, NoSQL databases have a good fit with two key requirements of healthcare systems.

2.6 Previous Research on Performance and Scalability of NoSQL Databases

The review of the literature identified increasing research activity focused on distributed databases and the comparison of NoSQL database systems with relational database systems focusing on topics such as basic performance comparisons on single nodes for MongoDB and Microsoft SQL Server, distributed and scalable searches of scientific XML data, distributed spatial data context for product and price search, large-scale text analysis, scalable transactions on NoSQL database systems, and querying NoSQL database systems (Aji et al. 2013; Atzeni et al. 2013; Chen & Hsu 2013; Dede et al. 2011; Dey, Fekete & Röhm 2013; Oliveira et al. 2013; Parker, Poe & Vrbsky 2013; Ruan, Zhang & Plale 2013).

Yahoo Cloud Serving Benchmark (YCSB) is a benchmarking tool for comparing performance of databases using pre-defined sets and rules (Cooper et al. 2010). However, previous research has shown a preference for developing custom benchmarking tools for comparison of performance and scalability between NoSQL databases and relational databases in specific domains which limits the external validity and ability to replicate the findings of such studies. For example, Shi et al. (2010) in evaluating performance of cloud databases used a specific approach to benchmarking which focused on the architecture and query capabilities of these databases, rather than using a well-known performance evaluation benchmark, YCSB. Lungu and Tudorica (2013) and Aboutorabi et al. (2015) also developed custom benchmarking applications to compare performance of NoSQL and relational databases in order to identify and use the most efficient data access methods for each database. It is also observed that latency and throughput are two well regarded and widely used metrics for the comparison of NoSQL databases and relational databases in recent research (Swaroop & Vijit Gupta 2016). It is also observed that latency and throughput are two main metrics used in the comparison of NoSQL databases and relational databases in recent research (Swaroop & Vijit Gupta 2016). Throughput is measured as an average number of database operations completed per second and latency is measured as an average execution time for each database operation. These

two metrics form the basis of the performance evaluation of a NoSQL document database comparative to a relational database in a large scale EHR system.

2.6.1 Previous Research on Evaluation of NoSQL Databases in Healthcare

Research and industry projects focusing on storing healthcare information in NoSQL databases are being driven by practical experience (Jin, Deyu & Xianrong 2011). This demonstrates that the relational approach for storing healthcare records has become a bottleneck for healthcare systems as the structure and size of the healthcare data have changed considerably over time. There is also an increased emphasis on better utilising healthcare information to deliver better healthcare outcomes and NoSQL databases can play an important role in achieving this aim because of their strong support for a distributed database environment. Medical databases can contain heterogeneous data including text, images, free-text physician notes, logs from medical devices, etc. which are difficult to handle and manage using traditional relational databases in terms of size and structure (Jin, Deyu & Xianrong 2011; Schmitt & Majchrzak 2012).

One of the early developments in the use of NoSQL databases in the area of healthcare is the project called DIGHT (**D**istributed **I**nfrastructure for **G**lobal Electronic **H**ealth Record **T**echnology) of Swedish Institute of Computer Science (SICS) and Centre for Development of Advanced Computing (CDAC), which aims to develop a distributed EHR system for lifelong health records for about one billion Indian citizens (Alnuem et al. 2011; Drejhammar 2010). The DIGHT project focused on developing a customised NoSQL database at a time when there were very few NoSQL databases available in the marketplace. (CDAC 2009).

Although there are industry examples of NoSQL databases being used in healthcare applications, there is limited empirical research on the use of NoSQL databases in healthcare. Lee at el. (2013), Jin et al. (2011) and Schmitt and Majchrzak (2012) have contributed to emerging research on the use of NoSQL databases in EHR systems by evaluating NoSQL databases for distributed storage of healthcare data in terms of data model and performance. Klein et al. (2015) compared various types of NoSQL databases using synthetic EHR data of one million patients. Freire et al. (2016) conducted a similar study to this research by comparing the performance of NoSQL and relational databases with a relatively small dataset using archetype-based EHR data. However, the design of comparative performance evaluation of NoSQL

databases and relational databases reported in previous empirical studies do not completely reflect a nationwide EHR system in size.

2.7 Literature Gap and Research Focus

The discussion in the preceding sections of this chapter demonstrates that the requirements of evolving healthcare data needs cannot be satisfied by relational databases and NoSQL databases have significant potential to provide EHR systems with necessary functionality and capabilities. Moreover, given that the document store type of NoSQL databases are determined as being highly suitable for storing healthcare data, there is a lack of previous research that has compared the performance of NoSQL document databases and relational databases in a large scale realistic EHR system environment.

Li and Manoharan (2013) compared the performance of Microsoft SQL Server with multiple NoSQL databases that are based on a key-value store implementation at a relatively small scale. They found that performance varied depending on the database operation and that not all NoSQL databases perform better than the Microsoft SQL Server database.

The literature review also suggests that both relational databases and NoSQL databases have their suitable domains and use cases. In this regard, there are multiple papers suggesting that the determination of which type of database is better for a particular use case is directly related to the requirements of a particular use case and the required data model. For instance, Nance et al. (2013) mention that the problem that an organisation is trying to solve will determine whether to choose a NoSQL database or a relational database. Swaroop and Vijit Gupta (2016) suggest that the selection of data model and appropriate database depends on the use case.

Limited research exists that has focused on exploring the possibility of establishing a healthcare data model using a NoSQL database. Other research merely tries to evaluate basic database performance by comparing the performance of NoSQL databases with relational databases. Inadequate attention has been given in prior research to establishing a healthcare data model and then testing the performance with realistic large-scale healthcare data sets. Clearly, this may lead to results which deviate from what can be found in a real-world scenario (Hadjigeorgiou 2013; Jin, Deyu &

Xianrong 2011; Lee, Tang & Choi 2013; Sattar, Lorenzen & Nallamaddi 2013; Schmitt & Majchrzak 2012).

2.8 Conceptual Model and Research Questions

Figure 2.13 provides a conceptual model of IT artefacts that will be built and evaluated; and activities which will be undertaken in order to provide a solution to a real world problem identified previously in the review of the literature. Therefore, this research focuses on first building a number of IT artefacts. This involves establishing a NoSQL document data model and a relational data model for storing electronic health records and then building a random healthcare data generator to generate synthetic EHR records. Then a prototype EHR system is built that will enable a performance evaluation of a NoSQL document database comparative to a relational database for basic database operations and scalability, EHR sharing and data analysis capabilities (complex querying) in a simulation of a large scale EHR system. Thus, the main objectives of this study can be achieved by evaluating the performance, scalability, EHR sharing and data analysis capabilities of NoSQL document databases and relational database comparatively to demonstrate the feasibility of using NoSQL document databases in large scale EHR systems.

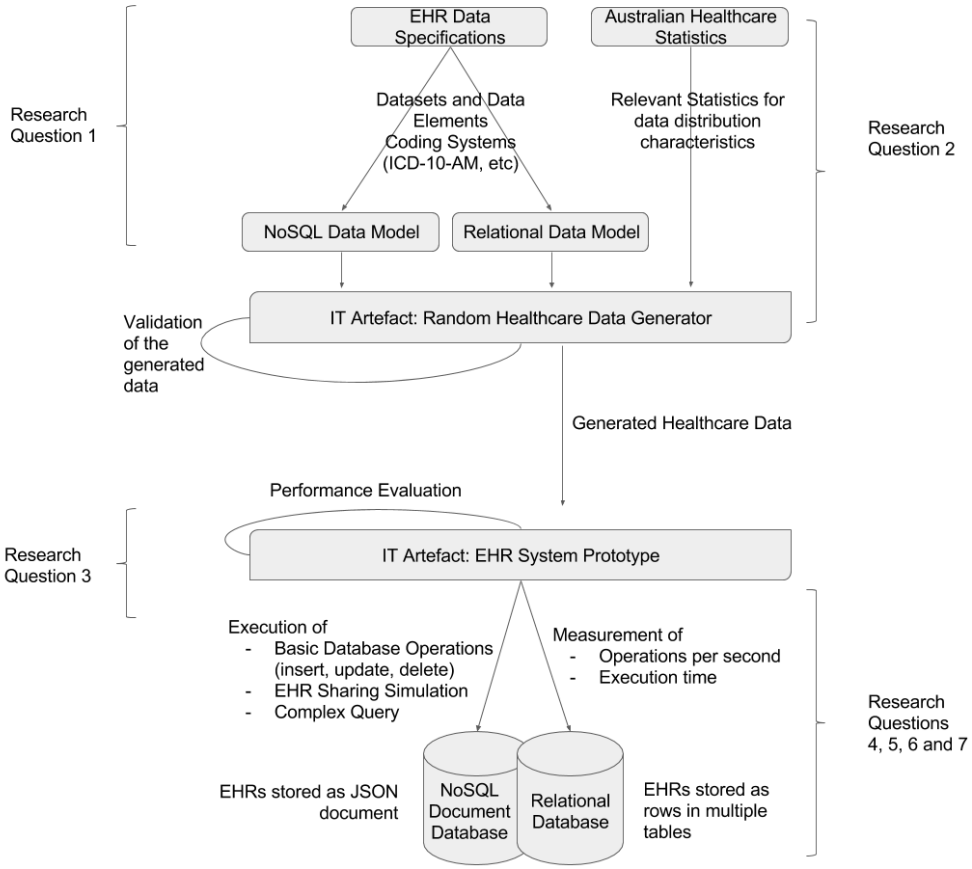


Figure 2.13: Conceptual model of artefacts built and evaluated and associated research activities conducted to achieve main objectives of this study

The research problem that provided the motivation for this study is addressed by the following overarching general research question.

General RQ: How can a simulation of a large EHR system be developed so that the performance of NoSQL document databases comparative to relational databases can be evaluated?

In order to investigate this general research question, the following specific research questions are investigated for proof of concept, in an Australian Healthcare context using a Design Science methodology.

RQ1: How can a NoSQL document data model and a relational data model be developed for an EHR system that are in line with documents published by healthcare authorities in Australia?

RQ2: How can a random healthcare data generator be developed that will generate EHRs that are representative of the characteristics of Australian healthcare data based on statistics available in the public domain?

RQ3: How can a prototype EHR system be developed that will facilitate database operations and measure performance and scalability for NoSQL document databases and relational databases?

RQ4: How do NoSQL document databases perform compared to relational databases in executing basic database operations such as insert, delete and update on electronic health records?

RQ5: How do NoSQL document databases scale compared to relational databases in electronic health record systems?

RQ6: How do NoSQL document databases perform compared to relational databases in supporting electronic health record sharing through patient record retrieval in a distributed EHR system?

RQ7: How do NoSQL document databases perform compared to relational databases in executing complex queries on electronic health records?

2.9 Conclusion

This chapter reviewed existing knowledge on Electronic Health Records (EHR), EHR systems, relational databases and NoSQL database systems as the parent literature for this study. The immediate literature identified the importance of NoSQL document databases and their suitability for large scale EHR systems were identified and discussed as the main focus of this study. The theoretical background to this study is discussed in terms of the descriptive and prescriptive theory and practical knowledge that informs a design science approach. The relevant theory and practical knowledge that provided the foundation for this study is discussed in terms of the development and evaluation of IT artefacts to achieve the main objectives of this study. This discussion was guided by kernel theories, design theory and current practice

knowledge. This included reviewing and identifying appropriate data modelling approaches for the NoSQL document databases and relational databases that were used in this study. CAP theorem is discussed in terms of its guidance in choosing the most suitable NoSQL databases for electronic health records management. Then, evaluation of the performance of NoSQL document databases is discussed in terms of relevant database performance metrics within the context of healthcare. Previous literature regarding the evaluation of the performance of NoSQL databases in EHR systems highlighted the current gap in the literature. The need for empirical research that addresses this gap is identified and discussed.

In summary, the literature review demonstrates a significant gap in literature. There is little empirical work has been conducted to establish a reliable and complete sample healthcare data model for EHR systems using NoSQL document databases. Furthermore, there is a lack of a robust evaluation of the performance, scalability, EHR sharing and data analysis (complex querying) capabilities of NoSQL databases comparative to relational databases for large scale healthcare-specific applications such as a national EHR system.

Based on this identified gap, an overarching research question and a specific set of research questions are presented in this chapter.

This research conducted an empirical evaluation of a NoSQL document database in large scale Electronic Health Records (EHR) systems in comparison to a relational databases based on a healthcare data model to address these research questions and the identified gap in the literature. In this regard, this research contributes to the existing knowledge by evaluating a NoSQL document database in a particular domain, healthcare.

Although NoSQL databases have significant potential for offering better solutions than relational databases in large scale implementations in many sectors, including healthcare, organisations tend to stay away from exploring them. Organisations in general are unfamiliar with NoSQL databases and tend to think that they are not knowledgeable enough to pick the correct type of NoSQL databases for their use case (Nance et al. 2013).

Therefore, by addressing the gap identified in the literature with an empirical evaluation of the performance of a NoSQL document database in EHR systems in the healthcare domain, this research also aims to contribute to practice as the key findings of this study will help professionals to choose the most suitable database for their use cases in this domain.

Chapter 3 - Methodology

3.1 Introduction

In this chapter, the research paradigm and the research design that guided the methodological approach used to conduct this study is described and justified. Although a recent methodological approach in the Information Systems discipline, Design Science, has been used extensively and is well-established in other reference disciplines such as economics, engineering and computing science. Moreover, there is a growing body of literature in Information Systems that provides substantial guidance on how to conduct research using a Design Science methodological approach in a rigorous and relevant manner.

This chapter begins by describing and justifying the choice of Design Science as the research paradigm and philosophy that underpins the research design of this study. Then Design Science is justified as a sound methodological approach that meets the main objectives of this study. Then, the research plan is presented, which explains how this research was conducted in six phases. Next, the research design is assessed using design science evaluation principals. This is followed by the planned contribution of this research using a design science approach. The structure of this chapter is presented in Figure 3.1.

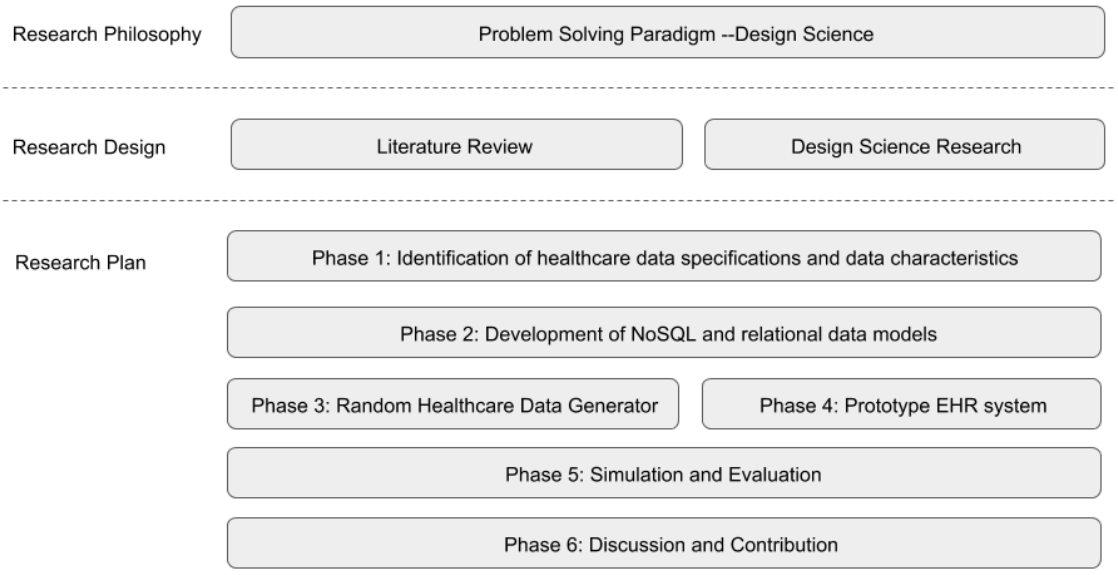


Figure 3.1 Structure of Chapter 3

3.2 Research Philosophy

The choice of the scientific paradigm that underpins the conduct of an empirical study is determined by the philosophical belief of a researcher and ultimately determines the choice of a methodological approach (Weber 2004). A scientific paradigm is understood to be the distinct worldview of the researcher based on certain ontological, epistemological and methodological assumptions (Niehaves 2007). There are a number of classifications of scientific paradigms. In Information Systems the scientific paradigms used have been predominately positivist and interpretivist. More recently, there has been an increasing focus on design science as a suitable alternative research paradigm with the IT artefact as the key concept that is built and evaluated to provide a solution to a real world problem (Gregor & Hevner 2013; Hevner et al. 2004). Furthermore, a scientific paradigm can be used to classify two distinct types of research: behavioural science research and design science research (Hevner et al. 2004; March & Smith 1995). Positivist and interpretivist paradigms adopted in Information Systems research have tended to focus on behavioural science research which is a "problem understanding paradigm"; while design science research is clearly a "problem solving paradigm" (Hevner et al. 2004; March & Smith 1995). The choice of a research paradigm in turn determines the ontology, epistemology and methodology that will be used in a study. Furthermore, the choice of a research paradigm determines how knowledge and theory is generated and communicated, the role between theory and practice, the rigor versus relevance debate and role of the researcher as a participant in the research (see Table 3.1).

	Positivist	Interpretive	Critical	Scientific realism	Design Science
Ontology	A physical world where a single reality exists	A social world where multiple realities are constructed through human interactions.	A social world where multiple realities are historically constructed and re-constructed.	An objective physical and social world independent of humans.	Multiple world states where reality is socio-technologically constructed
Epistemology	Objective reality is investigated through structured instruments that follow rigorous empirical testing	Subjective reality is investigated through accessing meanings that humans assign to them while addressing cultural and contextual elements	Subjective reality that is embedded in social and historical practices is generated through critical evaluation of social systems.	Universal laws and principles searched are based on distinct logic of discovery and logic of justification	Objectively constrained reality that is contextually constructed is revealed through iterative circumscription
Dominant Methodology	Quantitative	Qualitative	Ethnography and historical studies	Mix of methods	Mix of methods
Axiology – Values	True knowledge: generating generalizable theories	Situated knowledge: understanding IS phenomena in the social world	Historical knowledge: understanding the IS phenomena by analyzing the historical dynamics among humans, tech and organizations	Fallible knowledge: knowledge is continuously revised and updated	Design knowledge: shaping the IS phenomena in the real world through creating artefacts
Relationship between theory and practice	Theory is used to produce desired state of affairs in the physical world.	Theory cannot be wholly used to predict future situations	Social theory and social research are understood as social critique.	It is possible to discover universal laws that govern the external world	Design theory is used to build predictably functioning artefacts.
Role of researcher	Passive/value neutral observer	Participant observer who enacts social reality	Participant observer who initiates change in social relations and practices.	Objective, impartial observer, passive, value-neutral	Participant observer at early stages then more value neutral observer later
Methodology guidelines	Dubé and Paré (2003); Straub, Boudreau and Gefen (2004)	Klein and Myers (1999); Lee (1989)	Myers and Klein (2011)		Hevner et al. 2004

Table 3.1 Summary of Research Paradigm Perspectives used in Information Systems (adapted from Aljafari and Khazanchi (2013))

From an ontological perspective, design science research by definition changes the state of the world through the introduction of novel artefacts that attempt to solve real world problems. Hence, alternative world states are acceptable for design science research as an artefact is built and evaluated to solve a real world problem. In this research, a number of IT artefacts were built to solve a real world problem, the performance evaluation of a NoSQL document database in terms of basic database operations and scalability, data sharing and data analysis capability comparative to a relational database.

Epistemologically in design science research, an artefact is developed and evaluated and its behaviour and outcomes are the results of interactions between components of the problem domain. Descriptions of the interactions are information and to the extent that an artefact behaves predictably, the information is true. In other words, the functionality that an artefact enables in providing a solution to real world problem is information; and measurable in the build and evaluation phases. In this research the functionality of three IT artefacts, a healthcare data modal (NoSQL document database, relational database), a random healthcare generator and a prototype EHR system provided the functionality that enabled the performance evaluation of a NoSQL document database comparative to a relational database and provided measurable information in building and evaluation phases of this research.

Methodological approach in design science research is developmental where the impact of an artefact(s) is measurable in a composite system. One or more range of methodological approaches can be used in design science, depending on the nature of the problem which is being solved through building and evaluating artefacts. In this research, an experimental design was used to enable a performance evaluation of a NoSQL document database in terms of basic database operations, and scalability, data sharing and data analysis capability comparative to a relational database in a simulation of a large scale EHR system.

From an axiological perspective, design science research values creative manipulation and control of the environment in the problem domain. Artefacts are built and evaluated as solutions to a specified problem which leads to improvements and better understanding of a problem domain where knowledge is not static and is constantly evolving. In this research a number of IT artefacts were built to enable a realistic

performance evaluation of a NoSQL document database in terms of basic database operations, and scalability, data sharing and data analysis capability comparative to a relational database in a simulation of a large scale EHR system. This provided a better understanding of the suitability of a NoSQL document database as a viable alternative to a relational database as a data management technology for a large scale EHR system.

One of the key strengths of design science as a research paradigm is that the relationship between theory and practice is grounded in design theory and appropriate kernel theories and practice knowledge that inform building and evaluating functional artefacts to provide solutions to real world problems. In this research appropriate kernel theories and practice knowledge informed the building and evaluation of functional IT artefacts, two healthcare data models, a random healthcare data generator and a prototype EHR system. These IT artefacts provided proof of concept that a NoSQL document database as a viable solution to address the shortcomings of relational databases in meeting the data management needs of large scale distributed EHR systems

The role of the researcher in a design science project is that of a participant observer who becomes a more value-neutral observer in the later phases of evaluation of an artefact as a solution to a real world problem and its contribution more broadly to theory and practice in the problem domain. In this study, the researcher was a participant observer who was actively involved in the design and implementation of a number of IT artefacts. Then, the researcher role became that of a more value neutral observer. These IT artefacts were then used to provide a rigorous and relevant performance evaluation of a NoSQL document database in terms of basic database operations, and scalability, data sharing and data analysis capability comparative to a relational database in a simulation of a large scale EHR system.

The conduct and evaluation of design science research in Information Systems should be based on well-established methodology guidelines (Gregor & Hevner 2013; Hevner et al. 2004; Venable, Pries-Heje & Baskerville 2012). In this study, a well established set of methodological steps for conducting a design science research were used.

3.2.1 Methodological Approach

The Information Systems (IS) discipline has seen increased research activity using Design Science Research (DSR) as a sound theoretical and methodological approach that emphasises both rigor and relevance. In DSR, an effective solution is suggested by designing and building an artefact and then the utility, quality and efficacy of an artefact in providing a solution to a particular IS problem is evaluated. Although there is increasing research activity using a DSR approach to invent or build new systems in the IS discipline, establishing a theoretical background and theorising using a DSR approach is still a challenging task (Chatterjee 2015). DSR is positioned to solve real world problems through design. Thus, the IS community has engaged in considerable discussion on the 'relevance versus rigor' debate and DSR is increasingly seen as a viable approach to ensure both rigor and relevance in IS research.

Hevner et al. (2004), Gregor and Hevner (2013) and Goldkuhl (2004) have contributed to the effort of establishing guidelines for conducting rigorous and relevant DSR. Previous literature suggests that DSR aims to provide solutions to IS problems by building and evaluating artefacts which involve a design phase. Therefore, the research process in DSR uses kernel (reference) theories and well-established practice referred to as justificatory knowledge to underpin and inform the design phase of artefacts (Hevner et al. 2004; Kuechler & Vaishnavi 2008; Walls, Widmeyer & El Sawy 1992). Thus, kernel theories and practice knowledge are translated as inputs into the process of the development of a design theory that is relevant and may be explanatory or predictive resulting in the creation of an artefact(s) to solve a real world problem (Gregor & Jones 2007; Kuechler & Vaishnavi 2012). The evaluation of an artefact can then, in turn, to lead to refinement and enrichment of kernel theories and existing practice knowledge. Thus, design science research, from a theoretical perspective, provides a process for not only describing how to design an artefact but also for understanding why an artefact should work. The evaluation of designed artefacts in terms of their utility, quality and efficacy in solving a real world problem (Hevner et al. 2004; Gregor & Hevner 2013) leads to evidence and confirmation that an intended result is based on presumed cause and effect. The relationship between kernel theories, design theories and design process results in artefacts that provide solutions to real world problems, is depicted in Figure 3.2.

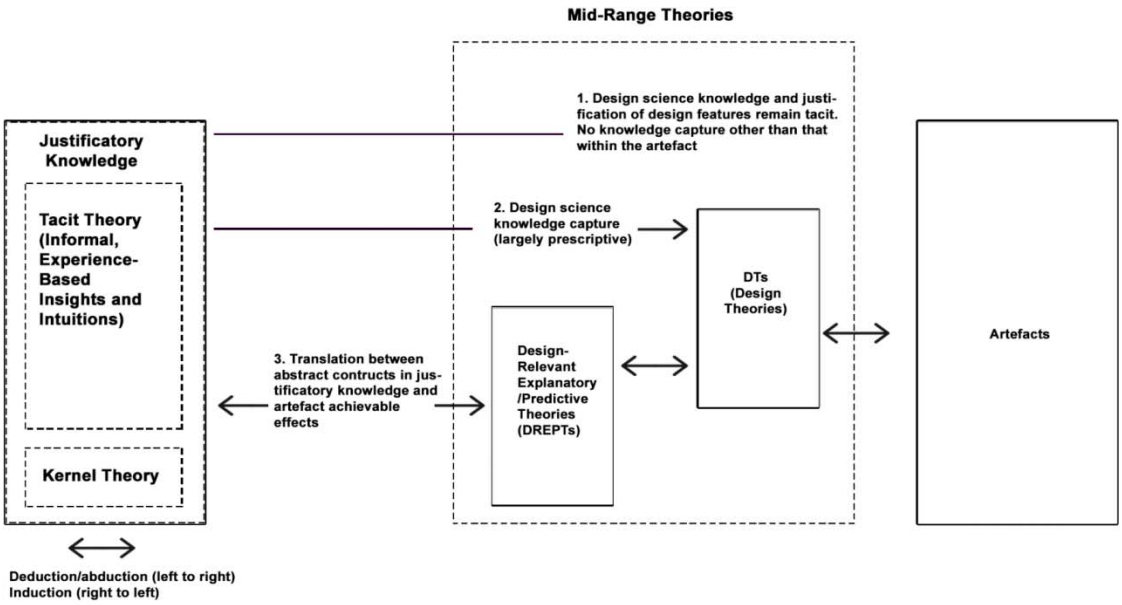


Figure 3.2: Framework for theory development in Design Science Research (Adapted from Kuechler and Vaishnavi (2008)

This research provides evidence to determine to what extent NoSQL databases can provide a solution to the technological issues affecting large scale EHR systems by evaluating designed artefacts. The design phase was informed by kernel theories and practice knowledge which provided descriptive knowledge to guide the design and development of artefacts. The kernel theories and practice knowledge used in this study are data modelling approaches, Australian healthcare data elements and statistics, CAP theorem and database performance metrics. Relational database theory is used to guide the establishment of a relational data model; important NoSQL data modelling concepts such as de-normalisation and aggregation informed the establishment of a NoSQL document database data model. CAP theorem and practice knowledge helped to determine the choice of a NoSQL database that is suitable for EHR systems. Design theory and practical knowledge informed the non-trivial and innovative adaption of known knowledge and solutions regarding NoSQL document databases to a new problem context, data management in large scale EHR systems (Gregor & Hevner 2013; Kuechler & Vaishnavi 2012). Thus, a design theory was developed from the design and evaluation of the utility, quality and efficacy of NoSQL databases in large scale EHRs. Utility is the defining characteristic of an artefact which can be evaluated in terms of a number of dimensions including functionality,

performance and reliability (Helfert, Donnellan & Ostrowski 2012; Hevner et al. 2004). In this research, the utility of the artefact is primarily evaluated by assessing the database performance, scalability, data sharing and data analysis capability of a NoSQL database comparative to a relational database in a large scale EHR system.

3.3 Overall Research Design

The research design of an empirical study is guided by the philosophical stance and worldview adopted by a researcher. This study is guided by the design science research paradigm which is a problem-solving paradigm which in turn determined the choice of the methodological approach used to collect data to provide answers to the seven research questions investigated in this research and to the IT artefacts which are built and evaluated in this study.

Although theorising is not always easy when the research involves creative work, there are a number of research papers which provide clear guidance to researchers on how to rigorously conduct research that employs a design science research methodology (DSRM) (Chatterjee 2015; Gregor & Hevner 2013; Hevner et al. 2004). March and Smith (1995). Hevner et al. (2004) has written extensively about design science as a legitimate research methodology and provided guidelines on how to follow the steps of design science research in a rigorous manner (Alturki, Gable & Bandara 2011; Gregor & Jones 2007; March & Smith 1995).

In the IS discipline, it is possible that the design theories may be seen to have different forms compared to other disciplines. The design, construction and use of artefacts based on information technology (IT) to solve real world problems are increasingly seen to be central to ensuring the relevance of IS research and still maintaining research rigour (Alturki, Gable & Bandara 2011; Chatterjee 2015; Gregor & Jones 2007). The term artefact may include such things as software, formal logic, rigorous mathematics, and so on. Hevner et al (2004) argues that the understanding of a problem domain and also its solution are achieved in the process of building, and in the application and evaluation of the designed artefact (Hevner et al. 2004) (see Figure 3.3 for a conceptual overview of this approach). The business needs guide the development and evaluation of artefacts that are relevant in addressing a real world problem; and the existing knowledge in terms of theories and frameworks and methodologies ensures rigor in the conduct of a DSR project (Hevner et al. 2004).

Furthermore, Chatterjee (2015) suggests that DSR needs to demonstrate that the IT artefacts have quality, efficacy and utility.

Since the purpose of this research project is evaluate a suggested solution to a particular research problem which is underpinned by developing a number of IT artefacts, an experimental design used in the simulation of a large scale EHR system is an appropriate methodology for collecting data to answer the specific research questions framed by this research. Various steps have been suggested to achieve similar goals within a DSRM context (Alturki, Gable & Bandara 2011).

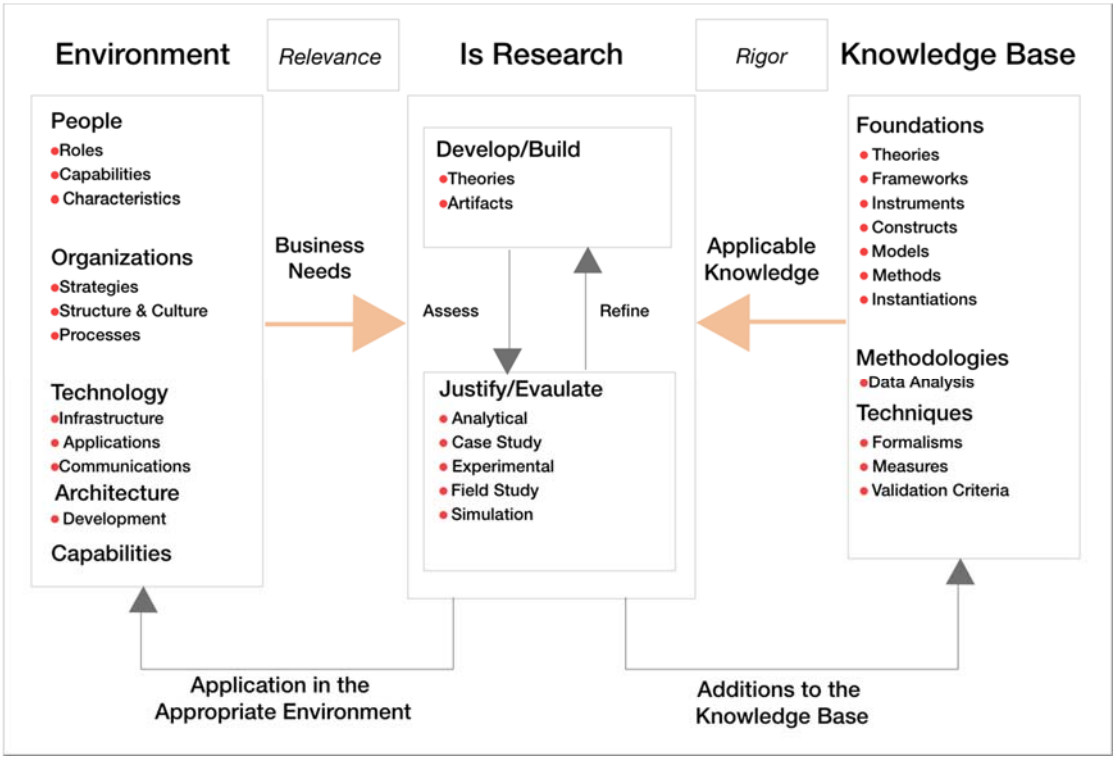


Figure 3.3. Design Science Research Model (Adapted from Hevner, 2004)

Alturki et al (2011) derived a summary table of prescribed steps for conducting Design Science Research based on a number of DSR articles, see Table 3.2. While it has been suggested that the steps for a DSRM are as simple as (1) Build and (2) Evaluate, others such as Gregor and Jones (2007) suggest six compulsory and two optional steps for a DSRM. Rossi & Sein (2003), on the other hand, identified five Design Science Steps in a DSRM, which are: (1) Identify a need; (2) Build; (3) Evaluate,;(4) Learn; and (5) Theorise (Rossi & Sein 2003). These five steps were followed in using a DSRM approach in this study.

Author/Year	Steps	Design Science Activities/Steps/Tasks presented in the paper							
(Nunamaker Jr, Chen, & Purdin, 1991)	5	Construct a conceptual framework	Develop a system architecture	Analyse and design the system	Build the (prototype) system	Observe and evaluate the system			
(Walls et al., 1992)	2	Design Product				Design Process			
	7	Meta-requirements	Meta-design	Kernel theories	Testable design product hypotheses	Design method	Kernel theories	Testable design process hypotheses	
(S.T. March & Smith, 1995)	2	Build				Evaluate			
(Rossi & Sein, 2003)	5	Identify a need	Build	Evaluate	Learn	Theorise			
(Hevner et al., 2004)	7	Design as an Artifact	Problem Relevance	Design Evaluation	Research Contributions	Research Rigour	Design as a Search Process	Communication of Research	
(Vaishnavi & Kuechler, 2004)	5	Awareness of a problem	Suggestion	Development	Evaluation	Conclusion			
(Aken, 2004)	4	Choosing a case	Planning and implementing interventions	Reflecting on the results	Developing design knowledge to be tested and refined in subsequent cases				
(Cole, Purao, Rossi, & Sein, 2005)	4	Problem Definition	Intervention	Evaluation	Reflection and Learning				
(Venable, 2006)	4	Solution technology invention	Theory building	Artificial evaluation	Naturalistic evaluation				
(Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007)	6	Problem identification and motivation	Define the objectives for a solution	Design and development	Demonstration	Evaluation	Communication		
(Gregor & Jones, 2007)	8	Compulsory						Optional	
		The purpose and scope	Constructs	Principles of form and function	Artifact mutability	Testable propositions	Justificatory knowledge	Principles of implementation	Expository instantiation
(Salvatore T. March & Storey, 2008)	6	Identification and clear description of a relevant organizational IT problem	Demonstration that no adequate solutions exist in the extant knowledge-base	Development and presentation of a novel IT artifact that addresses the problem	Rigorous evaluation of the IT artifact enabling the assessment of its utility	Articulation of the value added to the knowledge-base and to practice	Explanation of the implications for IT management and practice		
(Pries-Heje et al., 2008a)	4	Risk Evaluation in DS Research							
		Risk identification	Risk analysing	Risk treatment	Risk monitoring				
(Pries-Heje et al., 2008b)	8	Evaluation Activity							
		Ex Ante Naturalistic Design process	Ex Ante Naturalistic Design product	Ex Ante Artificial Design process	Ex Ante Artificial Design product	Ex Post Naturalistic Design process	Ex Post Naturalistic Design product	Ex Post Artificial Design process	Ex Post Artificial Design product
(Baskerville, Pries-Heje, & Venable, 2009)	7	A specific problem is identified and delineated	This problem must then be expressed as a specific set of requirements	The specific problem are systemically abstracted and translated into a general problem	General solution design (a class of solutions) for the general problem	General design requirements are compared with the specific problem for fit	A declarative search is then made for the specific components that will provide a workable instance of a solution to the general requirements.	An instance of the specific solution is constructed and deployed into the social system	

Table 3.2 Design Science Activities/Steps Taken Distilled from Literature (adopted from Alturki, Gable & Bandara (2011)

Gregor and Hevner (2013) also suggest a schema for publication of the results of Design Science Research which includes the following sections: (1) Introduction; (2) Literature Review; (3) Method; (4) Artefact Description; (5) Evaluation; (6) Discussion; and (7) Conclusions (Gregor & Hevner 2013). This schema guided the overall structure of this PhD thesis and also informed the structure of methodological approach of this study for conducting a rigorous and relevant DSR project.

3.3.1 Identify research problem and need to conduct research

An extensive review of the existing literature suggests that relational databases are not well adapted to modern day data-driven applications; and database design remains a critical problem in modern distributed systems such as EHR systems, that needs to be solved (Badia & Lemire 2011; Floratou et al. 2012). Recent developments in distributed and horizontally-scalable database systems, namely NoSQL databases, have been discussed previously in the literature review chapter. However, there are a few empirical studies such as Floratou et al. (2012) and Cattell (2011) that compare the performance of NoSQL and relational databases. The suitability of a database system depends on the purpose of the application rather than basic data access performances as discussed in the previous chapter. Furthermore, the results of comparisons between NoSQL databases and relational databases are highly associated with the types, versions and capabilities of the system. It is worthwhile to note that developments in distributed database systems are quite rapid, thus the development in such domains is often referred to as an 'explosion' (Phanishayee et al. 2012).

The research problem is defined and scoped in terms of a general over-arching research question and seven specific research questions:

Problem Definition – General Research Question: How can a simulation of a large EHR system be developed so that the performance of NoSQL document databases comparative to relational databases can be evaluated?

Seven research questions investigated the building and evaluation of IT artefacts as a solution to a real world problem and determined the scope of this study. These were:

RQ1: How can a NoSQL document data model and a relational data model be developed for an EHR system that are in line with documents published by healthcare authorities in Australia?

RQ2: How can a random healthcare data generator be developed that will generate EHRs that are representative of the characteristics of Australian healthcare data based on statistics available in the public domain?

RQ3: How can a prototype EHR system be developed that will facilitate database operations and measure performance and scalability for NoSQL document databases and relational databases?

RQ4: How do NoSQL document databases perform compared to relational databases in executing basic database operations such as insert, delete and update on electronic health records?

RQ5: How do NoSQL document databases scale compared to relational databases in electronic health record systems?

RQ6: How do NoSQL document databases perform compared to relational databases in supporting electronic health record sharing through patient record retrieval in a distributed EHR system?

RQ7: How do NoSQL document databases perform compared to relational databases in executing complex queries on electronic health records?

Hence, the unit of analysis in this research is a database management system in the context of a large scale EHR system. The dependent variable is the performance of a database management system which involved a comparative analysis and evaluation of a NoSQL document database management system versus a relational database management system in a simulation of a large scale EHR system. The independent variables in this study are the type of database management system, basic database operations (insert, update, delete), scalability, EHR sharing and data analysis (complex querying) capability.

Methodological approach used in this research

The research paradigm adopted by the researcher, Design Science, and the nature of the research problem being investigated determined the methodological approach used in this research (Hevner et al. 2004; Venable, Pries-Heje & Baskerville 2012). In Design Science research, one or more of a number of different methods can be used based on whether the evaluation is naturalistic or artificial and ex ante or ex post. An artificial evaluation using a simulation suited the main objectives of this study as purely technical artefacts were built and evaluated; and this approach provided the desired rigor with control of the key variables in an efficient and cost effective manner (Venable, Pries-Heje & Baskerville 2012). Given that this research focuses on providing a technological solution to a real world problem, an experimental design that utilised simulation was an appropriate methodological approach. A performance evaluation of a NoSQL document database in terms of basic database operations, and scalability, data sharing and data analysis capability comparative to a relational database in a simulation of a large scale EHR system was conducted for the evaluation of the artefacts designed and built to solve a real world problem.

A number of artefacts were designed and built based on existing knowledge and design theories, kernel theories and practice knowledge of the researcher (Kuechler & Vaishnavi 2012) to simulate a large scale EHR system in order to evaluate the performance of a NoSQL database comparative to a relational database in relation to data management of EHRs. Two data models with data structures designed for storing EHRs in a NoSQL document database and a relational database were developed. A random healthcare data generator that creates synthetic healthcare data based on the characteristics of Australian healthcare data and Australian Healthcare Statistics was developed to generate electronic health records representative of Australian healthcare statistics in sufficient volume for a simulation of a large scale EHR system. A prototype EHR system was developed to manage and capture metrics for the performance of database operations in a large scale EHR system simulation. The database operations of a NoSQL document database and a relational database that were evaluated for comparative performance included basic database operations (insert, update, delete), scalability, EHR sharing and data analysis (complex querying) capabilities. This prototype EHR system enabled the comparative evaluation of the performance of a NoSQL database with a relational database in a large scale EHR

system simulation. The research plan describes how each of these research activities was undertaken in the build and evaluation phases of this study.

Reliability and Validity

Reliability and validity are fundamental cornerstones of a rigorous research approach (Creswell 2013; Golafshani 2003). Reliability ensures that any significant results must be more than a one-off finding and must be inherently repeatable (Golafshani 2003). The results should be consistent with theoretical expectations and the researcher's interpretations. Other researchers must be able to replicate the study under the same conditions and generate similar results. In order to ensure the reliability of this study, the researcher provided a detailed description of how this research was conducted, and how the artefacts were built and evaluated in relation to existing knowledge in the problem domain.

Validity encompasses the entire research process and establishes how the key results meet all of the requirements of a scientific method (such as an experimental design and simulation) used in an empirical study (Creswell 2013; Golafshani 2003). Internal validity is ensured through the rigorous application of a structured design of a research method. Internal validity can be defined as the ability of a method to accurately measure what it is intended or supposed to measure. In this study, the method used an experimental design to implement a simulation of database performance testing in a large scale EHR system environment. This involved building and evaluating a number of key artefacts in order to conduct a simulation of database performance testing in a large scale EHR system. Internal validity is an important priority in conducting rigorous research, however, in an applied discipline such as Information Systems where relevance is also a high priority it is equally important to strengthen and emphasise external validity (Calder, Phillips & Tybout 1982; Green 1977; Gregor & Hevner 2013; Victora, Habicht & Bryce 2004).

External validity is concerned with the extent to which the results of a study can be generalised to other situations (Creswell 2013). No method can be completely successful in ensuring external validity; hence, research results can be called significant but not absolute truths. External validity increases the likelihood that the key results and findings of a study can be translated into practice (Gregor & Hevner 2013; Victora, Habicht & Bryce 2004). This research focused on building and

evaluating technology based models and instantiated artefacts (data models for a NoSQL document database and a MySQL relational database, Random Healthcare Data Generator, Prototype EHR system) in a simulation of a large scale EHR system. Hence, the research activities and results can be reprocessed or re-calibrated so as to circumvent differences in context and produce generalizable results for different contexts such as a different country or industry setting. Furthermore, external validity is strengthened in this study by demonstrating practical utility and efficacy of artefact(s) using a design science approach so that the key findings of this study can be translated into IT practice in the healthcare domain (Gregor & Hevner 2013; Victora, Habicht & Bryce 2004).

3.4 Research Plan

This research consisted of six phases in order to build and evaluate a number of artefacts which addressed the identified problem by investigating and providing answers to seven research questions (See Figure 3.4). These six phases are described and justified in this section to explain how this DSR project is conducted.

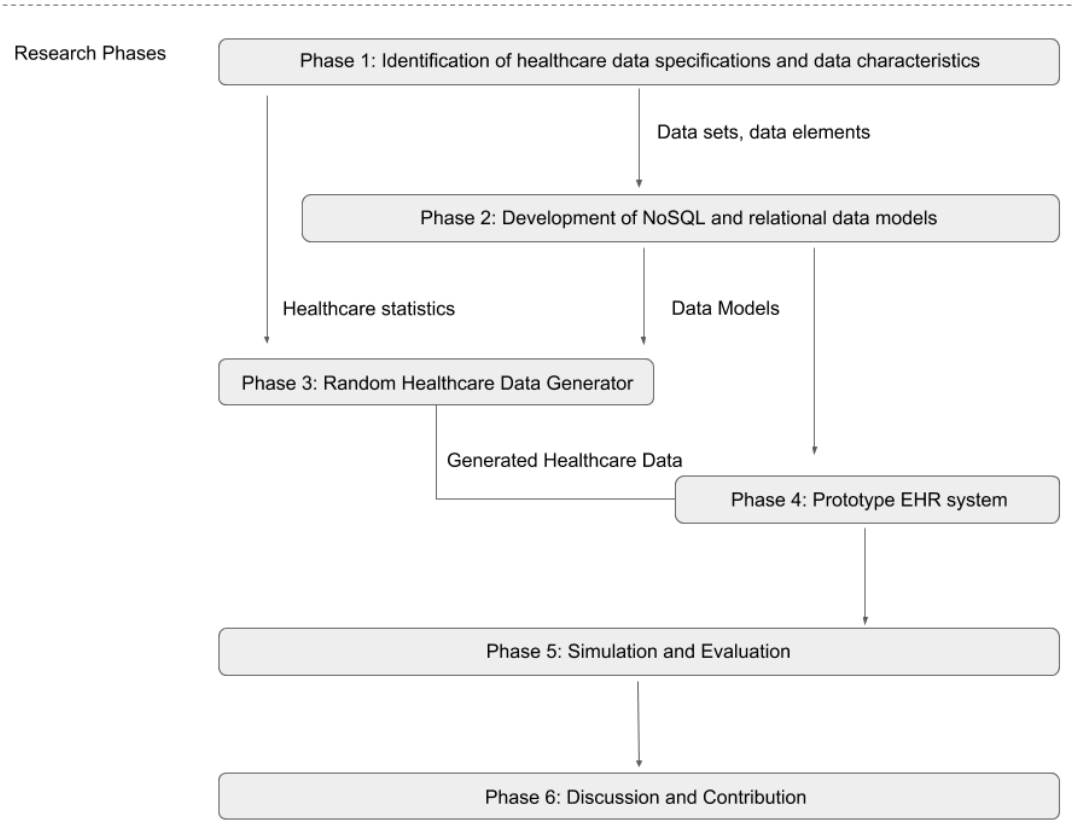


Figure 3.4. Research Phases used to conduct this research

Just like any purpose-built system, such as accounting, enterprise resource planning, and so on, EHR systems have their own data characteristics. In order to be able to evaluate a NoSQL document database in a healthcare domain properly, the basic requirements and specifications of the healthcare data used in the evaluation phases of this study need to be identified.

The first phase (PH1) of this research consists of the identification of healthcare data specifications and data characteristics. Using an appropriate data model that reflects actual real-world needs is essential to achieve higher quality research outcomes. Therefore, identification of healthcare data specifications such as coding systems, standards, minimum data sets, etc. is the first step in this project.

The Australian Institute of Health and Welfare (AIHW) published the National Health Data Dictionary (NHDD) Version 16.2 in 2015. The NHDD is publicly available and includes definitions for data elements and national minimum data sets (NMDS), as well as coding standards to be used such as ICD-10-AM. Information in NHDD, including Admitted Patient Care NMDS, Non-Admitted Patient Emergency Department Care NMDS, Outpatient Care NMDS and health care client identification data set specification (DSS) guided establishment of the specifications and the scope of the healthcare data which are the main input for establishing data models in this research (AIHW 2015).

Additionally, the statistical distribution of the characteristics of each data element may affect performance, data structure and overall evaluation results. Therefore, the next step in this phase of this research was to identify characteristics of healthcare data. Healthcare data statistics are publicly available from the Australian Institute of Health and Welfare. This information is sourced from AIHW publications and guided the first phase of the research (AIHW 2016).

In the second phase (PH2), data models were developed for both a NoSQL document database and a relational databases using known best practices in the industry, such as normalisation, indexing and query optimisation, foreign and primary keys for the relational data model and de-normalisation and aggregation for the NoSQL document data model. The outcome of this phase is relational and non-relational healthcare data

models consisting of the data sets and data elements identified in the previous phase of the project. In this phase, one relational and one NoSQL database system were selected based on their features and availability at that time. Data models developed in this phase are then applied to the selected relational and NoSQL document databases to create the underlying data structures. At the end of phase two (PH2), the outcome is a relational EHR data model and a non-relational EHR data model. The structure of these two data models were developed using relevant data modelling theory and practice. The data elements in these two EHR data models were determined by analysing AIHW publications and selecting two data sets and associated data elements, as well as analysing the documentation outlining the characteristics of Australian healthcare data based on statistics available to the public domain.

In the third phase (PH3) of this research the first IT (software) artefact, a Random Healthcare Data Generator, is built. This artefact is based on the healthcare data characteristics identified in the first phase. The purpose of this artefact is to generate random healthcare data which is used for testing and comparing the performance of a NoSQL document database comparative to a relational database. Thus, the complexity of accessing real healthcare data including the ethical issues concerned with gaining permission to access real world healthcare data was avoided. A feedback mechanism was built into the Random Healthcare Data Generator to validate the quality of the generated healthcare data. This ensured that the data generated by the Random Healthcare Data Generator reflects the Australian healthcare data characteristics.

The outcome of the third phase is a random healthcare data generator. This artefact is a fundamental outcome and also a significant contribution to future researchers who would like to use randomly-generated healthcare data in their research to avoid the security and privacy issues associated with accessing real world healthcare data.

In the fourth phase (PH4), another IT artefact, the prototype EHR system was built to simulate a large scale EHR system accessing and sharing EHR data generated by the IT artefact built in phase three. This artefact will facilitate the simulation of the data sharing process for EHR applications. Thus this artefact is the application interface to evaluate NoSQL databases and compare the performance of NoSQL database with relational databases in an EHR application specific role.

After completing these four design and build phases, IT artefacts to be used for the performance measurement of NoSQL databases in EHR systems in comparison to relational databases in EHR systems were ready to be deployed in a simulation of a large scale EHR system. The prototype EHR system is developed in phase four and is a significant contribution of this study as it facilitated basic database operations, scalability, EHR data sharing, and data analysis (complex querying) for a NoSQL document database and a relational database to complete the evaluation.

The fifth phase (PH5) was concerned with the performance measurement and comparison of a NoSQL document database with a relational database in the healthcare domain based on four important criteria, (1) basic database performance; (2) scalability; (3) EHR data sharing performance; and (4) complex data query performance.

For basic database performance testing, database operations such as insert, update and delete were evaluated using performance indicators such as execution time and operations per second. EHR system prototype artefact handles the execution of database operations, as well as recording the metrics required for the performance evaluation. The database nodes are run on the Amazon Web Services Elastic Compute Cloud (EC2) platform (Amazon 2016). This cloud computing platform enabled easy scaling and configuration of database and client nodes to allow execution of these tests on a range of number of nodes that facilitated a scalability comparison of both databases.

Random Healthcare Data Generator artefact developed in phase four (PH4) is used to generate healthcare data, and Prototype EHR System artefact handles database operations for both databases (NoSQL, Relational) using generated healthcare data. This artefact also enabled the execution of EHR data sharing simulation which requires querying both databases for all EHRs of a particular person.

In addition to the basic database operations, scalability, and EHR data sharing tests, a complex query is run against both relational and NoSQL databases and the data analysis (complex querying) performance of a NoSQL database and a relational database is evaluated and compared.

In the last phase (PH6), all outcomes derived from the performance testing completed in the previous phase (PH5) are individually analysed and discussed, and then collectively analysed and discussed to provide an overall comparison which led to a conclusion as to which type of database is better—NoSQL database versus a relational database in a distributed EHR system.

3.5 Evaluating Design Science Research Approach

Hevner et al. (2004) suggested a number of guidelines on how to evaluate the quality of a design science research. Table 3.3 describes each of these guidelines and discusses in the third column how this research meets each of these guidelines to ensure quality of a research project using a Design Science approach.

Guideline	Description	Discussion
Design as an Artefact	Design Science research must produce a viable artefact in the form of a construct, a model, a method, or an instantiation.	There are multiple artefacts produced as a result of this research. The NoSQL data model for EHR is a model which was a key input to two IT artefacts instantiated, Random Healthcare Data Generator and EHR prototype which are designed to enable evaluation of the proposed solution to a particular problem.
Problem Relevance	The objective of this Design Science research is to develop technology-based solutions to important and relevant business problems.	Increasing size and complexity of healthcare data causes bottlenecks and operational issues in many cases and NoSQL databases have many advantages over relational databases in healthcare domain. Hence, this research develops IT artefacts that simulate a large scale EHR system running on a NoSQL database and a relational database.
Design Evaluation	The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods.	The artefacts are evaluated by conducting a series of performance test using different scenarios. These tests demonstrate basic database operations performance, scalability, EHR sharing performance and complex query performance of a NoSQL database comparative to a relational database
Research Contributions	Effective Design Science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations, and/or design methodologies.	This research project identified a gap in the existing literature, the lack of a comprehensive evaluation of the suitability of using NoSQL databases in large scale EHR systems and seeks to address the identified research problem using a Design Science Research approach. The research also designed, built and evaluated an NoSQL data model for EHR systems, Random Healthcare Data Generator and a prototype EHR system that have significant theoretical and practical contributions.
Research Rigour	Design Science research relies upon application of rigorous methods in construction and evaluation of the design artefact.	The research method and steps used to conduct this study were based on well established Design Science Research principles and structures

Guideline	Description	Discussion
Design as Search	The search for an effective artefact design requires utilising available means to reach desired ends while satisfying laws in the problem environment.	The artefacts are developed based on existing knowledge both theoretical and practical utilising widely accepted best practices in database design and management such as normalisation and data aggregation and performance testing of databases with appropriate evaluation methods and metrics
Communication of Research	Design Science research must be presented effectively both to technology-oriented, as well as management-oriented audiences.	The research outcomes are of interest and useful to both technology-oriented management-oriented audiences by means of an open source solution that compromised of a number of artefacts including data models and instantiated artefacts to simulate database operations in a large scale EHR. Furthermore, the solution is cost effective and flexible. The PhD thesis will be made available to the public after a standard 12 months embargo period to allow the researcher and supervisory team to publish the key findings of this study in research and practice journals. The key artefacts developed in this study will also be made available through GitHub or similar websites.

Table 3.3. Guidelines for assessment of DSR adapted from Hevner et al. (2004)

3.6 Planned Research Contribution

Although Design Science Research is a valid and widely-accepted methodology, theorising and expressing the theoretical contributions remains a challenge (Chatterjee 2015). Design science research contributes to IS knowledge in different ways, namely: invention, improvement, exaptation and routine design, depending on solution maturity and application domain maturity (Gregor & Hevner 2013). Figure 3.4 shows how these contribution types are positioned.

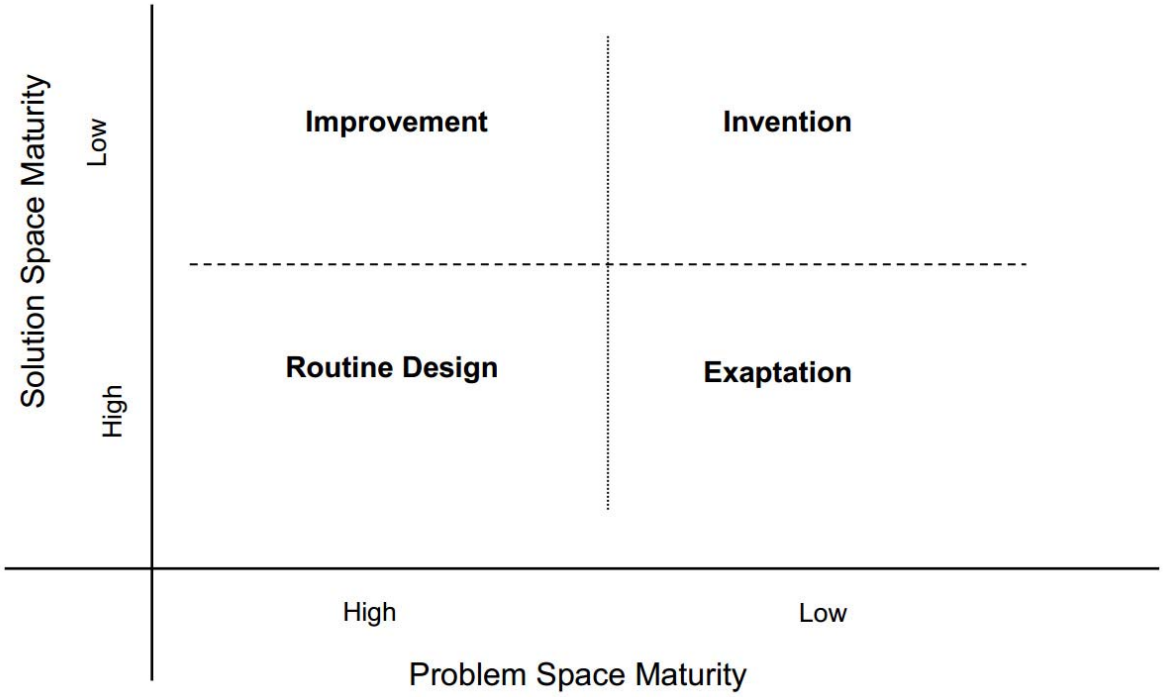


Figure 3.5. DSR Knowledge Contribution Framework (Gregor & Hevner, 2013)

Exaptation type of research contributes to knowledge by adapting new technologies which have emerged in response to problems in other fields or disciplines into a new field. Testing and refining prior ideas new fields enables exaptation of these ideas (Gregor & Hevner 2013). Brendt et al (2003) present an example of an exaptation type of contribution. They have adapted data warehouse development methods to healthcare in their CATCH data warehouse research project which constitutes an IS research example of adapting emerging database methods that are primarily in the scope of computer science domain into healthcare field by developing an IT artefact to test and refine the proposed solution (Berndt, Hevner & Studnicki 2003). Similarly, this research will make a contribution to theoretical and practical knowledge by way of the exaptation of a new and emerging database technology, a NoSQL document database to data management of EHRs in a large scale EHR system.

3.7 Conclusion

This chapter presented and justified the Design Science research paradigm and philosophy adopted by the researcher which, in turn, determined the research design and the choice of an experimental methodology that utilised simulation as an appropriate approach for conducting this Design Science research. In order to address

the research problem and general research question identified as a gap in the literature, the main objectives of this study were specified in seven research questions which are systematically addressed in the chosen methodological approach using a six phase research design.

The purpose of this research is to determine the suitability and feasibility of NoSQL systems in the healthcare domain by considering healthcare-specific data models and data characteristics, and developing and evaluating IT artefacts that are specifically built for healthcare applications rather than using a generic performance-measurement approach. In this regard, this research was well suited for using a DSR approach. The choice of an experimental method, a simulation of the database performance in a large scale EHR system in order to evaluate the performance of a NoSQL document database comparative to a relational database, was described and justified. The research problem, general research question and seven specific research questions were restated in this chapter. The scope of the study was clearly delineated by defining the unit of analysis and independent and dependent variables that were investigated in the seven research questions. The rigour and relevance of this design science project was described in terms of how reliability and validity were ensured in the research design and methodological approach.

A design science research framework and a set of guidelines for conducting design science (Hevner et al. (2004)) was used to demonstrate that the research design used in this study is a rigorous and relevant approach that followed sound design science guidelines. The research design used six phases in order to build and evaluate artefacts to solve a real world problem. This included identification of the data sets and data elements of healthcare data (PH1) and establishment of data models for a NoSQL document database and a relational database based on these requirements (PH2). This is followed by the identification of relevant Australian public healthcare statistics and development of a Random Healthcare Data Generator artefact to generate synthetic healthcare data based on the data models and the statistics (PH3). The next phase of the research involved the development of a prototype EHR system that facilitated the performance measurement of database operations, scalability, EHR sharing and data analysis (complex querying) capabilities in a simulation of a large scale EHR system (PH4 and PH5). In the final phase (PH6), an evaluation of the performance of a NoSQL document database comparative to a relational database was conducted and

the outcomes are discussed in terms of the existing literature to determine the theoretical and practical contributions of this research.

Chapter 4 - Development of IT Artefacts

4.1 Introduction

A Design Science Research Methodology is based on multiple steps, as explained in the methodology chapter 3. One of these steps is the development of artefacts. In this chapter, the development of the IT artefacts that are practical outcomes of this research are described and discussed. The IT artefacts developed in this research that are described and discussed are: (1) a Relational Healthcare Data Model; (2) a NoSQL Healthcare Data Model; (3) a Random Healthcare Data Generator which is used to populate EHRs; and (4) a Prototype Electronic Health Record (EHR) System.

The first step in development of the first two IT artefacts in this research is the identification of healthcare data specifications and data characteristics in the Australian healthcare domain which provide the representative data sources for the Random Healthcare Data Generator to generate electronic health records that are representative of the Australian healthcare domain and statistics. In the following sections, data elements that reflect the actual real-world needs of the Australian healthcare domain, along with their respective specifications such as coding systems, standards, minimum data sets, etc., are identified.

Following the identification of the required datasets and data elements of the Australian Healthcare domain, healthcare data models for a NoSQL document database and a relational database were developed. For the relational data model, known best practices in the data management industry, such as normalisation, indexing and query optimisation, foreign and primary keys are utilised. For NoSQL document data model, an appropriate type of NoSQL document database is selected and the data model is established considering the effect on performance while executing database operations. This includes the coding systems and all values from lookup tables being embedded into the data model.

In subsequent sections of this chapter, the steps taken to develop the two primary IT (software) artefacts for this research, namely, a random healthcare data generator and a prototype EHR system are described and discussed. The random healthcare data generator generates healthcare data based on the publicly available statistics of the Australian healthcare domain, and prototype EHR system acts as a basic system that

manages the simulation of an EHR data sharing environment to conduct performance evaluation and comparison of a NoSQL document database with a relational database in such an environment. The structure of this chapter is presented in Figure 4.1.

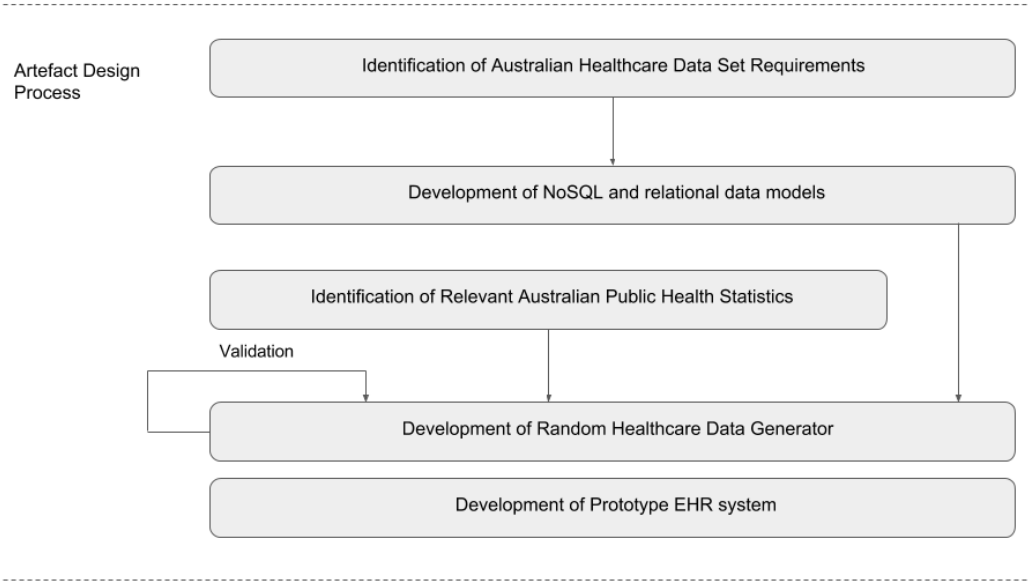


Figure 4.1 Structure of Chapter 4

4.2 Identification of Australian Healthcare Data Set Requirements

The first step in developing the data models required for this study was to identify required data elements and their attributes. This research focuses on the Australian healthcare domain. Therefore, datasets and data elements related to the Australian healthcare domain are identified and described in this section. These are essential inputs for the Random Healthcare Data Generator to populate the data models with EHR data that is representative of the Australian healthcare domain.

The Australian Institute of Health and Welfare has published the National Health Data Dictionary (NHDD) on their website which helps in establishing standards for data collection and reporting by Australian healthcare providers (AIHW 2015). In the NHDD, national minimum data sets, along with their attributes, are defined.

In this research two of the national minimum datasets are used as they are designed to be comprehensive enough for covering most of the basic healthcare data which is suitable for the context of this research—electronic health record systems. These two dataset are (1) Admitted Patient Care Dataset and (2) Non-admitted Patient

Emergency Department Care Dataset. Each of these datasets has multiple data elements under various categories.

Recently, Admitted Patient Care Dataset has been amalgamated from (1) Admitted Patient Care; (2) Admitted Patient Mental Health Care; and (3) Admitted Patient Palliative Care datasets (AIHW 2015). Thus, it is concluded that this dataset is comprehensive enough to cover a broad range of healthcare activities that would be stored in an EHR system for the purpose of this research. Full details of this dataset are shown in Appendix F.

The Admitted Patient Care Dataset consists of admission details, information about establishment (healthcare provider), demographic information about the patient and other relevant data. The Non-admitted Patient Emergency Department Care dataset is designed for cases related to emergency healthcare services and includes data elements related to patient details, as well as urgency status and similar episode-related data elements (AIHW 2015).

In this research, the scope is limited to mandatory data elements in Admitted Patient Care Datasets and Non-admitted Patient Emergency Department Care datasets in the process of establishing the data model and generating healthcare data based on Australian Health Care statistics. A total number of 49 unique data elements and their respective categories are listed in Table 4.1.

Person	Person identifier
	Area of usual residence
	Country of birth
	Date of birth
	Indigenous status
	Sex
	Medicare Eligibility status
	Address
	Record—identifier
Emergency Department Stay	Physical departure date
	Physical departure time
	Presentation date
	Presentation time
	Transport mode (arrival)
	Type of visit
	Urgency related group major diagnostic block
Patient	Compensable status
	Hospital insurance status
Episode of admitted patient care	Admission date
	Admission mode
	Admission urgency status
	Condition onset flag
	Intended length of hospital stay
	Number of days of hospital-in-the-home care,
	Number of leave days
	Patient election status
	Procedure
	Separation date
	Separation mode
Episode of care	Inter-hospital contracted patient status
	Mental health legal status
	Number of psychiatric care days
	Principal diagnosis
	Source of funding, patient funding source
	Funding eligibility indicator
Establishment	Australian state/territory identifier
	Geographic remoteness
	Organisation identifier (state/territory)
	Region identifier
	Sector
	Organisation identifier
Injury Event	Activity type
	External cause
	Place of occurrence
Non-admitted patient service	Episode end date
	Episode end status
	Episode end time

	Service episode length
Hospital Service	Care type

Table 4.1. Data elements by categories in the selected datasets: Admitted Patient Care and Non-admitted Patient Emergency Care (AIHW 2015)

4.3 Development of Relational and NoSQL Data Models

Based on the data elements identified in the previous section, two different EHR data models—a relational data model for relational databases and an aggregate oriented data model for NoSQL databases—were established.

4.3.1 Relational EHR Data Model

Relational database theory and relational data models are fundamentally based on Codd's normalisation approach (Codd 1970). Therefore, data elements are categorised based on their characteristics, repetition status and the requirement of a lookup list in order to execute normalisation.

Fields in the datasets, at their initial setup, were not in their normalised form. If the datasets were used in their initial states this would have resulted in redundant data providing unnecessary duplication, a state not representative of best database management practice. In addition, normalisation helps in making additions and deletions of EHRs easier, which is not possible if these two datasets were kept as is in de-normalised form.

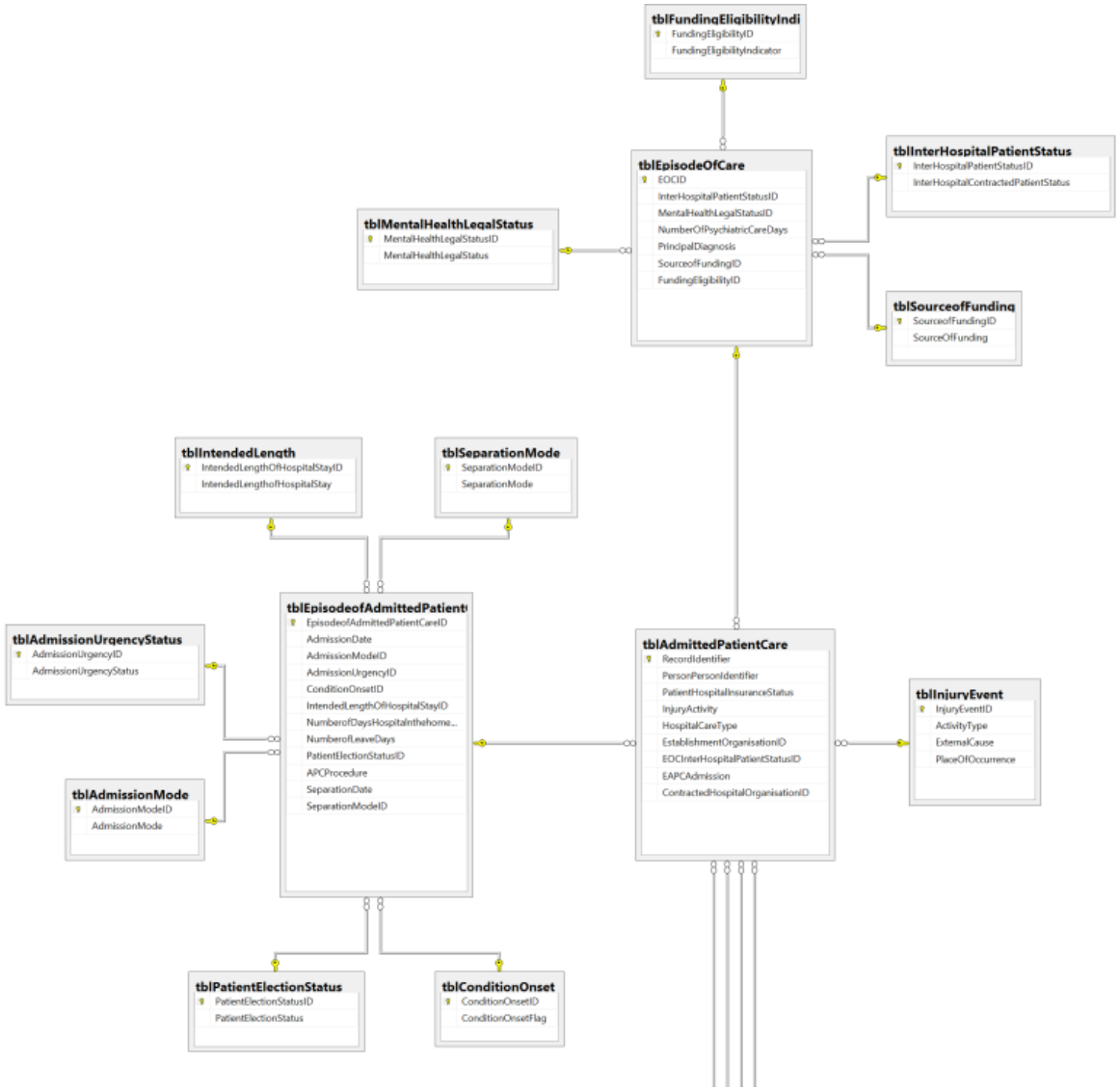
For example, there are person details in Admitted Patient Care dataset. A person can be an admitted patient at one time and a non-admitted patient at another time. In addition to Person, there are additional fields such as establishment which need to be stored as well. There are different establishments, therefore, it would make sense to split this data into a separate table having an establishment identifier and details such as [Establishment ID, Organisation ID, Australian state/territory identifier, Geographic remoteness, Region identifier, Sector].

Considering the data statistics and the main tables, the relational database was first brought into 1NF (first normal form). In 1NF, the database had only atomic values and there were no repeating groups. Each record was unique.

The relational database in 1NF still had partial functional dependency in tables such as recurring values for region identifier, sex, etc. In order to remove them, the

relational database was normalized further and is in 2NF at its current state. All the non-key fields now depend on all components of the primary key.

With the existing relational database in its 2NF, it has few transitional dependencies, but removing them will need more tables and more joins that will slow down the performance of the EHR relational database. Based on the statistics needed, it does not appear that the latest state will cause any critical insert, update or delete anomalies and the results produced will be efficient. Therefore, the data model for the relational database was considered to be most efficient if kept in 2NF. The Entity Relation Diagram for the relational EHR data model is shown in Figure 4.2.


tblFundingEligibilityIndi
FundingEligibilityID
FundingEligibilityIndicator
tblEpisodeOfCare
EOCID
InterHospitalPatientStatusID
MentalHealthLegalStatusID
NumberOfPsychiatricCareDays
PrincipalDiagnosis
SourceofFundingID
FundingEligibilityID
tblInterHospitalPatientStatus
InterHospitalPatientStatusID
InterHospitalContractedPatientStatus
tblMentalHealthLegalStatus
MentalHealthLegalStatusID
MentalHealthLegalStatus
tblSourceofFunding
SourceofFundingID
SourceOfFunding
tblIntendedLength
IntendedLengthOfHospitalStayID
IntendedLengthofHospitalStay
tblSeparationMode
SeparationModeID
SeparationMode
tblEpisodeofAdmittedPatient
EpisodeofAdmittedPatientCareID
AdmissionDate
AdmissionModeID
AdmissionUrgencyID
ConditionOnsetID
IntendedLengthOfHospitalStayID
NumberofDaysHospitalInthehome...
NumberofLeaveDays
PatientElectionStatusID
APCProcedure
SeparationDate
SeparationModeID
tblAdmissionUrgencyStatus
AdmissionUrgencyID
AdmissionUrgencyStatus
tblAdmissionMode
AdmissionModeID
AdmissionMode
tblAdmittedPatientCare
RecordIdentifier
PersonPersonIdentifier
PatientHospitalInsuranceStatus
InjuryActivity
HospitalCareType
EstablishmentOrganisationID
EOCInterHospitalPatientStatusID
EAPCAdmission
ContractedHospitalOrganisationID
tblInjuryEvent
InjuryEventID
ActivityType
ExternalCause
PlaceOfOccurrence
tblPatientElectionStatus
PatientElectionStatusID
PatientElectionStatus
tblConditionOnset
ConditionOnsetID
ConditionOnsetFlag

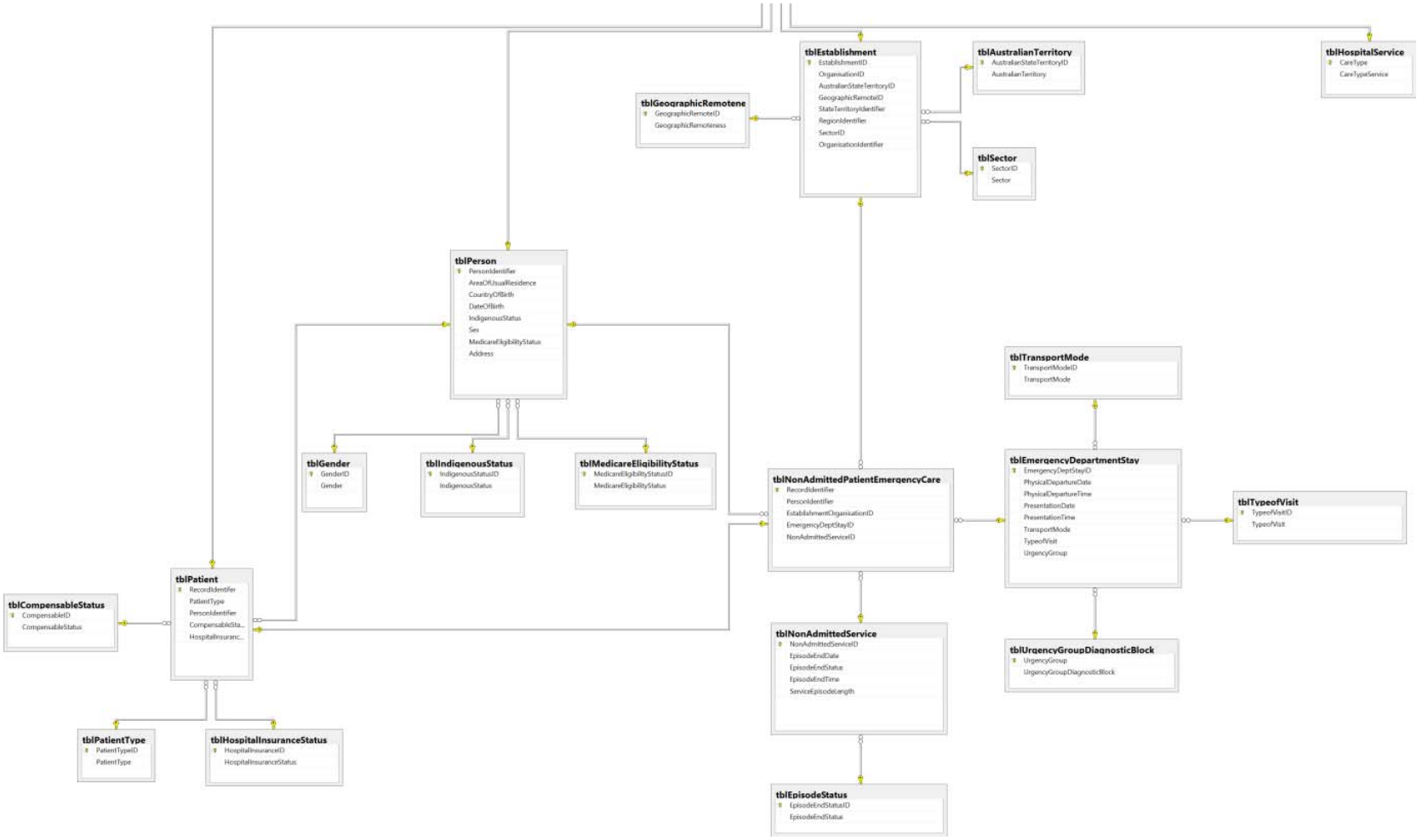


Figure 4.2. Entity Relationship Diagram for relational data model

4.3.2 NoSQL EHR Data Model

Modelling healthcare data for NoSQL databases may have a significant effect on performance. Therefore, establishing the correct data model for storing EHRs first requires a comparison between available data model types for NoSQL databases. NoSQL databases are mainly grouped into four categories in terms of how data is stored, key-value stores, column family stores, document databases and graph databases (Abramova & Bernardino 2013; Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016). In a recent study by Goli-Malekabadi et al. (2016) they evaluated these four categories of NoSQL databases for the purpose of storing healthcare data. Due to the nature of the healthcare data, document databases were identified as the best option for a NoSQL healthcare data model (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016).

In order to maximise the benefits and performance of document-based NoSQL databases, a data model should be kept in an un-normalised state which reduces processing complexity (Borkar et al. 2016). An aggregate oriented data model satisfies the requirements for healthcare data model and also provides high performance using de-normalisation and aggregation of all relevant data into a single document (Borkar et al. 2016; Sadalage & Fowler 2012). In an aggregate oriented data model, joins between multiple entries are avoided by including these linked data into the original document, whether it is a one-to-one or one-to-many relation. This approach is also called Embedded Documents (Vera et al. 2015).

Based on the Australian healthcare data characteristics, storing EHRs in an aggregate oriented data model in a document-based NoSQL system was selected as the best possible option for an EHR system which was discussed and justified in section 2.4.3 of Chapter 2 based on the previous relevant literature.

An aggregate oriented document-based data model can be established in multiple formats. The document oriented-databases can store documents in formats such as JSON, XML and BSON, however, JSON is becoming a standard format for document storage, processing and sharing and many NoSQL databases, such as MongoDB and Couchbase, natively support JSON format for data storage and retrieval (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Vohra 2015). Therefore, the minimum

Australian healthcare datasets are converted into an aggregate oriented JSON data model for storage in a NoSQL document oriented database.

A sample JSON formatted data model is shown in Figure 4.3 and a full JSON representation of the EHR datasets is shown in Appendix G.


```
{
    "Person": {
        "Person identifier": "123456789",
        "Area of usual residence": {
            "METeOR identifier": "469909",
            "code": "31701144631446",
            "value": "Darling Heights"
        },
        "Country of birth": {
            "METeOR identifier": "459973",
            "code": "5101",
            "value": "Myanmar"
        },
        "Date of birth": "01012000",
        "Indigenous status": {
            "METeOR identifier": "291036",
            "code": "4",
            "value": "Neither Aboriginal nor Torres Strait Islander origin"
        },
        "Sex": {
            "METeOR identifier": "287316",
            "code": "1",
            "value": "Male"
        },
        "Medicare Eligibility status": {
            "METeOR identifier": "481841",
            "code": "1",
            "value": "Eligible"
        },
        "Address": "",
        "Record-identifier": "abcd-1234"
    }
}
```


Figure 4.3. A sample section of NoSQL EHR data model

After completing these phases, the data models required for storing healthcare data in both relational and NoSQL databases for the purposes of this research are established.

4.4 Identification of Relevant Australian Healthcare Statistics

Following the identification of required data sets and data elements, and the establishment of appropriate data models for relational and NoSQL databases, an underpinning basis for healthcare data generation needs to be established prior to the development of Random Healthcare Data Generator. The healthcare data that is used in the large scale EHR system simulation for this research is generated based on the statistics that are available from the Australian Healthcare System (AIHW 2016)

Generating random data based on Australian healthcare statistics ensures that the distribution of the data is similar to real life scenarios. This is particularly important for the relational database, and indexes are directly affected by the distribution—thus the data distribution characteristics might have a significant effect on performance of database operations.

For the purpose of this research, the statistics that are only relevant to the EHR datasets and data elements that are mentioned in the previous sections are identified and included in this section.

In a relational database approach, the datasets and data elements are represented in the database schema as tables and have various fields/columns. Many of the columns in tables are multi-value attributes that give rise to further tables.

Australian healthcare statistics published by AIHW contain information about the data distribution for the data elements of the datasets that are the subject of this research (AIHW 2016). These Australian healthcare statistics are discussed in the following 6 sub-sections: (1) separations; (2) age group and sex; (3) indigenous status; (4) mode of admission; (5) urgency of admission; and (6) principal diagnosis.

4.4.1 Separations

This section describes the statistics that underpin the random generation of data for Admitted Patient Care EHRs based mainly on Separations. A separation is an episode of care for admitted patients which is considered a stay between admission and discharge, transfer or death. The Separation statistics provided in this section comprise the following tables/fields:

- Episode of Admitted Patient Care-> Intended length of hospital stay (same-day/overnight)
- Episode of Admitted Patient Care->Separation mode
- Establishment -> Australian state/territory identifier
- Establishment->Sector
- Person -> Area/state of usual residence

Establishment sector can be public or private. Separation Mode can be one of the following:

- Discharge/transfer to (an)other acute hospital
- Discharge/transfer to a residential aged care service, unless this is the usual place of residence
- Discharge/transfer to (an)other psychiatric hospital
- Discharge/transfer to other health care accommodation (includes mothercraft hospitals)
- Statistical discharge - type change
- Left against medical advice/discharge at own risk
- Statistical discharge from leave
- Died
- Other (includes discharge to usual residence, own accommodation/welfare institution, e.g. prisons, hostels and group homes providing primarily welfare services).

In 2014–15, there were about 10.2 million separations in Australia's public and private hospitals: about 59% (6.0 million) of these occurred in public hospitals; 94% of separations were for acute care and 4% for rehabilitation care (AIHW 2016). Public hospitals accounted for about 70% of overnight separations and 52% of same-day separations. For the 4.2 million separations from private hospitals, about 23% of separations (941,000) occurred in private free-standing day hospital facilities and the remainder were in other private hospitals (that can provide overnight care).

In 2014–15, overnight separations made up almost 48% of separations in public hospitals and 30% in private hospitals (AIHW 2016). The proportion of overnight separations that were in public hospitals varied among states and territories, ranging from 64% in Queensland to 76% in New South Wales. The proportion of separations that were for same-day care varied among states and territories and between public and private hospitals.

For public hospitals, the proportion of same-day separations ranged from 46% in New South Wales to 69% in the Northern Territory (AIHW 2016). For private free-standing day hospitals and other private hospitals combined, it ranged from 67% in Victoria to 74% in New South Wales.

For 2014–15, about 98% of separations (9.9 million) were for people who were hospitalized in their state or territory of residence (AIHW 2016). However, in the Australian Capital Territory, almost 81% of hospital separations were for Australian Capital Territory residents, with most of the remainder (18%) being for residents of New South Wales.

Statistics for Separation Rates

In 2014–15, there were about 240 separations per 1,000 population in public hospitals and 164 per 1,000 in private hospitals (AIHW 2016). Separations per 1,000 population in public hospitals ranged from 208 in Tasmania to 598 in the Northern Territory. For private hospitals, separations per 1,000 population ranged from 143 in New South Wales to 207 in Queensland.

Statistics for Same Day Separations

The number of same-day separations may not be comparable among the states and territories due to variations in admission practices. Therefore, these data should be interpreted with caution.

In 2014–15, there were about 241 same-day separations per 1,000 population. Public hospitals accounted for about 125 same-day separations per 1,000 population and private hospitals accounted for 116 per 1,000 (AIHW 2016).

Rates of same-day separations in public hospitals ranged from 102 per 1,000 in New South Wales to 408 per 1,000 in the Northern Territory. For private hospitals, rates of same-day separations ranged from 105 per 1,000 in New South Wales to 144 per 1,000 in Queensland.

Statistics for Overnight Separations

In 2014–15, there were about 164 overnight separations per 1,000 population. Public hospitals accounted for about 115 overnight separations per 1,000 population and private hospitals accounted for about 49 per 1,000 (AIHW 2016).

Rates of overnight separations in public hospitals ranged from 100 per 1,000 in Tasmania to 190 per 1,000 in the Northern Territory. For private hospitals, rates of overnight separations ranged from 38 per 1,000 in New South Wales to 64 per 1,000 in Queensland.

Detailed tables about the information provided in this section are provided in Appendices B, C, D, and E.

4.4.2 Age Group and Sex

The information in this section provides statistics on people who received Admitted Patient Care. This involves the age group and sex of the patient, indigenous status of the patient, remoteness area of usual residence of the patient and socioeconomic status of the area of usual residence of the patient. The statistics provided in this section comprise the following tables/fields:

- Person -> Sex
- Person -> Date of birth

In 2014–15, 53% of separations were for women and girls. In 2014–15, people aged 65 and over accounted for 41% of separations (AIHW 2016). For people aged 65 to 74, separations increased by 27% overall, an average increase of 6.0% each year.

In 2014–15, overall there were over 5.3 million separations for females, compared with about 4.8 million separations for males (AIHW 2016). In particular, women accounted for 65% of separations for people aged 15 to 44 (the age range that includes most separations for childbirth). Females also accounted for more patient days than males (15.2 million and 13.6 million patient days, respectively).

People aged 65 and over (who make up about 15% of the population) accounted for 41% of separations and 49% of patient days in 2014–15 (AIHW 2016). People aged 85 and over accounted for about 7% of separations and 13% of patient days in 2014–15.

Separations and patient days, by age group and sex, all hospitals, 2014–15

Age group (years)	Separations			Patient days		
	Males	Females	Persons	Males	Females	Persons
0–4	219,720	159,389	379,117	676,746	538,567	1,215,359
5–9	84,680	63,642	148,323	130,481	100,350	230,832
10–14	64,532	55,277	119,811	119,825	119,752	239,583
15–19	101,356	139,281	240,640	234,719	331,471	566,206
20–24	124,197	231,036	355,237	349,992	501,704	851,795
25–29	127,805	300,140	427,946	388,299	715,378	1,103,678
30–34	150,466	359,946	510,414	449,280	890,928	1,340,210
35–39	165,743	312,328	478,071	478,181	767,095	1,245,276
40–44	216,614	302,106	518,720	569,595	683,730	1,253,325
45–49	250,427	292,103	542,535	634,240	659,974	1,294,220
50–54	319,136	350,723	669,860	773,310	783,345	1,556,660
55–59	370,644	372,601	743,248	901,259	853,899	1,755,161
60–64	448,671	410,311	858,985	1,103,734	974,989	2,078,726
65–69	537,433	458,878	996,312	1,368,543	1,163,734	2,532,278
70–74	493,250	426,272	919,525	1,316,016	1,197,705	2,513,758
75–79	461,650	395,739	857,390	1,357,846	1,314,180	2,672,027
80–84	361,038	332,447	693,486	1,251,830	1,380,710	2,632,541
85+	310,456	380,202	690,658	1,471,907	2,199,902	3,671,809
Total[(a)]	**4,807,825**	**5,342,450**	**10,150,367**	**13,575,816**	**15,177,442**	**28,753,539**

(a) Total includes separations for which the date of birth was not reported.

Table 4.2. Separation statistics for 2014-2015 based on age and sex (Adopted from AIHW 2016)

4.4.3 Indigenous Status

The information in this section provides statistics on people based on indigenous status who received Admitted Patient Care. The statistics that will be provided in this section comprise the following tables/fields:

- Person -> Indigenous Status

In 2014–15, there were about 443,000 separations reported for Aboriginal and Torres Strait Islander people (AIHW 2016). About 90% of separations for Indigenous Australians were from public hospitals, compared with 57% of separations for other Australians.

Indigenous Australians were hospitalised at about 2.4 times the rate for other Australians (950 and 393 separations per 1,000 population, respectively).

4.4.4 Mode of Admission

The information in this section provides statistics on the mode of admission for the patients who were admitted to hospital. The statistics provided in this section comprise of the following tables/fields:

- Episode of Admitted Patient Care -> Admission Mode

Patients may have the following modes of admission:

- Admitted patient transferred from another hospital
- Statistical admission: care type change—where a new admitted patient episode is created as a result of a change in the clinical intent of care (for example, a patient's care may move from a focus on acute care to a focus on rehabilitation or palliative care) within the same hospital
- New admission to hospital—this term refers to all other planned and unplanned admissions (that is, the patient was not transferred from another hospital or had a Statistical admission in the same hospital).

In 2014–15, most separations in both public and private hospitals had a mode of admission of new admission to hospital (94% and 96%, respectively) (AIHW 2016). Public hospitals had a higher proportion of patients transferred from another hospital than private hospitals (4.7% and 3.0%, respectively). For public hospitals, Western Australia had the highest proportion of patients transferred from another hospital and the Northern Territory had the lowest (6.3% and 0.1%, respectively).

Public hospitals also reported higher proportions of the type of admission called 'Statistical admissions: care type change' than private hospitals (1.6% and 0.6%, respectively). For public hospitals, the Australian Capital Territory had the highest proportion of patients with this 'statistical admission'.

A detailed table on admission modes is shown in Appendix E.

4.4.5 Urgency of Admission

The information in this section provides statistics on the urgency of admission for patients who were admitted to hospital. The statistics that will be provided in this section comprise the following tables/fields:

• Episode of Admitted Patient Care -> Admission Urgency Status

Admissions to hospital were categorised in 2014–15 as Emergency (required within 24 hours) or Elective (required at some stage beyond 24 hours). Emergency/elective status is not assigned for some admissions (for example, obstetric care and planned care, such as dialysis).

Statistics:

Separations by urgency of admission, public and private hospitals

	2014–15
Public hospitals	
Emergency	2,514,638
Elective	2,384,343
Not assigned	1,080,644
Not reported[(a)]	713
Total	**5,980,338**
Private hospitals	
Emergency	213,810
Elective	3,441,036
Not assigned	508,984
Not reported[(a)]	6,199
Total	**4,170,029**

Table 4.3. Separation statistics for 2014-2015 based urgency of admission (Adopted from AIHW 2016).

4.4.6 Principal Diagnosis

This section presents information on the reasons for patients' hospital admissions, which are described by the principal diagnosis—that is the diagnosis established after study (for example, at the completion of the episode of care) and chiefly responsible for occasioning the episode of admitted patient care. In some cases, the principal diagnosis is described in terms of a treatment for an ongoing condition (for example, care involving dialysis). The statistics provided in this section comprise the following tables/fields:

• Episode of Care -> Principal Diagnosis

Statistics

In 2014–15, more than one-quarter of separations in public and private hospitals had a principal diagnosis in the Z00-Z99 chapter of ICD-10-AM—which includes Care

involving dialysis (over 1.3 million separations), and Care involving use of rehabilitation procedures, radiotherapy, chemotherapy and palliative care (AIHW 2016).

The relative distribution of separations by ICD-10-AM chapter varied across public and private hospitals (AIHW 2016). For example, about 84% of separations for certain infectious and parasitic diseases and 82% of separations for injury, poisoning and certain other consequences of external causes were from public hospitals. For diseases of the eye and adnexa, about 73% of separations were from private hospitals.

Separations, by principal diagnosis in ICD-10-AM chapters, public and private hospitals, 2014–15

Principal diagnosis		Public hospitals	Private hospitals	Total
A00–B99	Certain infectious and parasitic diseases	125,953	24,284	150,237
C00–D48	Neoplasms	292,316	348,034	640,350
D50–D89	Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism	102,411	58,872	161,283
E00–E89	Endocrine, nutritional and metabolic diseases	97,936	58,849	156,785
F00–F99	Mental and behavioural disorders	204,767	190,846	395,613
G00–G99	Diseases of the nervous system	156,787	119,100	275,887
H00–H59	Diseases of the eye and adnexa	103,378	279,692	383,070
H60–H95	Diseases of the ear and mastoid process	33,148	30,726	63,874
I00–I99	Diseases of the circulatory system	339,253	150,866	490,119
J00–J99	Diseases of the respiratory system	338,772	99,193	437,965
K00–K93	Diseases of the digestive system	463,856	544,265	1,008,121
L00–L99	Diseases of the skin and subcutaneous tissue	117,422	47,032	164,454
M00–M99	Diseases of the musculoskeletal system and connective tissue	207,396	326,791	534,187
N00–N99	Diseases of the genitourinary system	271,558	199,016	470,574
O00–O99	Pregnancy, childbirth and the puerperium	353,721	136,954	490,675
P00–P96	Certain conditions originating in the perinatal period	54,605	11,141	65,746
Q00–Q99	Congenital malformations, deformations and chromosomal abnormalities	27,187	11,352	38,539
R00–R99	Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified	517,019	240,540	757,559
S00–T98	Injury, poisoning and certain other consequences of external causes	532,237	118,393	650,630
Z00–Z99	Factors influencing health status and contact with health services	1,638,320	1,174,076	2,812,396
	Not reported	2,296	7	2,303
Total		**5,980,338**	**4,170,029**	**10,150,367**

Table 4.4. Separation statistics for 2014-2015 based principal diagnosis (Adopted from AIHW 2016).

4.5 Development of Random Healthcare Data Generator

In the previous sections, the identification of Australian Healthcare data elements, the development of Australian Healthcare data models for relational and NoSQL databases and identified publicly available Australian healthcare statistics used to

populate these data elements are described. Based on this information, a Random Healthcare Data Generator is developed as an artefact in this research.

The Random Healthcare Data Generator is used to populate a NoSQL document database and a MySQL relational database with synthetic EHR data in a simulation of a large scale EHR system. This Random Healthcare Data Generator was developed to generate random, anonymised healthcare data having characteristics and distribution similar to the publicly available Australian healthcare data. This simulated Australian Healthcare data enabled a simulation of a large scale EHR system so that the performance of NoSQL databases could be assessed and compared to relational databases in a distributed EHR system.

The Random Healthcare Data Generator uses publicly available healthcare statistics discussed in the preceding section as an input to generate data for the datasets selected. The artefact is developed in C# .NET. This input, tables of healthcare statistics, are stored as in-memory arrays in the artefact to allow randomised generation of values for the relevant data elements using a data distribution algorithm which is described in detail in the next subsection.

The data distribution algorithm used in the Random Healthcare Data Generator based on the Australian healthcare statistics identified in the previous section is described in the following subsection.

Following the establishment of the distribution algorithm used in the Random Healthcare Data Generator, trial data generation was conducted and results for this data generation are validated against the original healthcare data statistical tables. This was to make sure the data distribution algorithm is performing correctly and data generation is valid and the results represent the Australian healthcare domain.

4.5.1 Data distribution algorithm

Establishing a data distribution algorithm that enables generating data based on Australian healthcare statistics is a fundamental requirement for the Random Healthcare Data Generator artefact. Publicly available Australian healthcare data statistics described in the previous section is mostly presented in data element pairs, such as age and sex, establishment type and principal diagnosis, and so on. These Australian healthcare statistics, in general, can be described using a multinomial

distribution and probability theory. For instance, for any given patient X, the probability of being in the age group of 0 to 4, P(0-4), can be roughly calculated as 0.0373500, which is the number of persons in the relevant age group divided by the total number of persons. Thus, using a multinomial distribution simulation, a random number between 0 and 1 can be generated and then mapped to a value according to the probability distribution (Chen 2010; Siegrist 1997)

A mathematically equivalent representation of this probability distribution helps avoid probability calculations for each statistical table. This can be achieved by generating a list of possible values and their respected weights. The weighted values are cumulatively added and expressed as minimum-maximum value pairs for each item in the list. For instance, if there are two items in the values list with respected weights of 10 and 90, the first item will have values of 1 and 10, and the second item will have values of 11 and 100 as lower and upper boundaries. Then a random number is generated based on the minimum of the first and maximum of the last item in the list, 1 and 100 in this example. Then the item for which the random number falls within the lower and upper boundaries is selected. For instance, if the random number is 50, then item 2 will be selected. Given enough rounds of random generation, distribution of the results will be similar to the distribution of the sample set given at the beginning.

This algorithm may further be expanded to a second dependent value to accommodate Australian statistics by simply executing as multiple steps.

A simple example will be explained using the values expressed in Table 4.5.

Age Group	Lower boundary	Upper boundary
0-4	1	379117
5-9	379118	527441

Table 4.5. An example table for lower-upper boundaries for age group statistics.

In the first step, a random number will be generated between 1 and 527441 to determine the age group using the above table. For instance, if the number is 500000, then the age group will be selected as 5-9. In the next step, sex distribution for the age group of 5-9 will be used in accordance with the distribution presented in Table 4.6.

Sex	Lower boundary	Upper boundary
Male	1	84680
Female	84681	148322

Not specified	148323	148323

Table 4.6. An example table for lower-upper boundaries for sex statistics.

Subsequently, a random number between 1 and 148323 will be generated. If the number is 10000, for instance, value “Male” will be selected as sex. As a result, generated value will be a male within the age group of 5-9.

Another possible approach would be combining the possible value pairs, which renders into a table as shown in Table 4.7:

Age-Sex	Lower boundary	Upper boundary
0-4, Male	1	219720
0-4, Female	219721	379109
0-4, Not specified	379110	379117
5-9, Male	379118	463797
5-9, Female	463798	527439
5-9, Not specified	527440	527440

Table 4.7. An example combined table for lower-upper boundaries for age group and sex statistics.

Using this combined table, only one round of random number generation is sufficient to determine the selected values of sex and age, thus this is a more efficient approach for these elements. However, there are multiple statistics based on one type of variable, for example, the establishment type being public or private is correlated with both urgency status and principal diagnosis. It is not practical to generate a combined distribution table for more than two related elements. Therefore, the combined table approach will only be applied wherever feasible.

The generation and selection procedure described above may be programmatically represented in two different ways: (1) store upper and lower boundaries in a structure and select the relevant value by scanning the list for upper-lower boundaries that the generated number falls in between. This can be executed either using probabilities or actual values, or (2) generate an array with the size of the highest upper boundary of actual numbers and assign the relevant value to each of the array elements according to their distribution, then directly accessing the value after random number generation (arr[number]).

In the first method, a very low memory usage occurs, despite the selection requiring comparison operation in each execution, while in the second method more memory will be required—however the selection of the value will be faster due to direct access.

In the context of this research, data generation speed is considered a more important aspect than the total memory size. Therefore, a small .NET application is developed to test the alternative methods.

The statistics presented in the previous section has 28 million as a total number at most. Therefore the test application has been developed to test 30 million lookup operations.

For the first approach, a dictionary having the key of cumulative probabilities and the value of corresponding string value for 8 possible values is created. It took an average of 60 seconds for 30 million random value selections based on the random number between 0 and 1 and the value in the dictionary relevant to that particular random number.

For the second approach, an array or 30 million items has been created and, for each item, a string value of "value + item number" has been assigned. In the test environment, it is seen that 1.2 GB of RAM is used for the resulting array which is below the .NET maximum size limit of 2 GB, and it took less than 6 seconds for 30 million lookups. The total memory consumption is estimated using `GC.GetTotalMemory` function and the result of this function has been confirmed using Windows Task Manager.

The computer used in this test had the following specification: Intel Core i7-4510U CPU @ 2.6 GHz, 2 cores, 4 virtual processors, 64-bit Windows 8.1 OS, 8GB of RAM, and 512 GB SSD disk.

Therefore, it is concluded that the second approach of creating an array and assigning values number of times relative to their actual distribution is the simplest yet far more effective approach. Furthermore, reducing memory size is also possible by using percentage values of distribution rather than the actual numbers wherever appropriate.

4.5.2 Validation of the random data generation algorithm

The data distribution method demonstrated in the previous section needs validation in order to make sure that the data generated by the random data generator artefact is actually similar in characteristics to the Australian Healthcare statistics used as input.

This validation is confirmed by generating a sample set of data based on one of the tables provided in the previous section and comparing the test results with the actual input. Urgency status table is used as a test input which has the values based on establishment status and urgency status. A combined table is established and given as an input to the random data generator.

Data has been generated three times, and the average of group totals are calculated based on the results as well as the difference between the actual input values and the results.

Table 4.8 shows the actual input value, calculated values based on three tests, their averages, difference between the actual and generated results and the percentage value of the difference based on the actual values.

Item Name	Actual value	Test 1 value	Test 2 value	Test 3 value	Test avg value	Difference	Difference (%)
Private Hospital - Elective	3441036	3440924	3440394	3438426	3439915	-1121	-0.03%
Private Hospital - Emergency	213810	213688	214240	213163	213697	-113	-0.05%
Private Hospital - Not assigned	508984	508627	508651	509528	508935	-49	-0.01%
Private Hospital - Not reported(a)	6199	6203	6236	6076	6172	-27	-0.44%
Public Hospital - Elective	2384343	2386138	2384633	2386577	2385783	1440	0.06%
Public Hospital - Emergency	2514638	2515068	2512879	2513928	2513958	-680	-0.03%
Public Hospital - Not assigned	1080644	1079013	1082642	1081907	1081187	543	0.05%
Public Hospital - Not reported(a)	713	706	692	762	720	7	0.99%

Table 4.8. Validation test for random data generation algorithm based on actual urgency of admission statistics.

In this validation test, total memory used for 10150367 items was 403 MB and the average run time for three tests was 2828 milliseconds, using the same computer having the specifications described in the array creation test.

Results of the data generation test were satisfactory as the highest discrepancy was less than 1% which occurred in a very small value of 713, which represents a small

minority in a total result set of more than 10 million items. This validation test demonstrates that the method used to generate random data is quite accurate in a high number of generated items, and more than 99% accurate in a lower number of items. Therefore, it is concluded that the random data generator developed in this research can serve the purpose of generating Australian healthcare data that is representative of Australian healthcare data statistics.

4.6 Development of Prototype EHR System

Following the development of the Random Healthcare Data Generator, a prototype EHR System was required to manage the simulation of database operations as well as data sharing and complex query for a performance comparison of a NoSQL database with a relational database in a distributed EHR system environment. Couchbase database has been selected as the NoSQL document database and MySQL database has been selected as the relational database for this research and this selection is discussed later in Chapter 5. This artefact is developed using C# .Net Framework 4.0. MySQL Connector/Net version 6.9.9 and Couchbase SDK 1.3.9 is used from relevant vendors.

The Prototype EHR System is responsible for receiving the generated healthcare data from the Random Healthcare Data Generator and assigning the data a unique identifier and inserting the data into NoSQL and relational databases.

This artefact also facilitates data sharing simulation which requires querying the relevant EHRs from both databases. Performance related measurements are also included as part of this artefact to minimize the measurement of operational overhead for data manipulation in order to enable more accurate comparison between NoSQL and relational databases in an EHR system.

For the purpose of this research, EHRs are generated by the Random Healthcare Data Generator and then processed by the Prototype EHR System. Each EHR is converted into INSERT, UPDATE, DELETE T-SQL statements for relevant operations on the relational database and into a JSON document for inserting or updating on the NoSQL database by the Prototype EHR System artefact. GET, SET, and GETVIEW operations are used for Couchbase database and T-SQL insert, update, delete and select statements are used for MySQL database.

A unique key is assigned to each EHR which will enable data lookup for data sharing function as well as update operations on the records. This unique key is called National Healthcare Document ID (NHDID) in this research.

A separate in-memory list is created for storing execution times for each database and a stopwatch is started at the beginning of the database operation and stopped after the execution. The resulting value, i.e. time in milliseconds of database operation, is added to the corresponding in-memory list called ConcurrentBag.

There are lists to store execution times for the following operations: (1) Insert; (2) Update; (3) Delete; and (4) Query. Separate lists are created for both relational and NoSQL databases and the results are saved to filesystem after each test.

This artefact is implemented as a Windows Service and it interacts with the selected databases and the file system to store the test results. The execution of the tests are triggered simultaneously on all client nodes using built-in REST endpoints of the artefacts by making HTTP requests with query string of requested test scenario parameters such as the operation, record count, etc. to be tested. Log files are generated in each client node and then merged into a single log file to be analysed. The relation between Random Healthcare Data Generator, Prototype EHR system, log files and database nodes are presented in Figure 4.4.

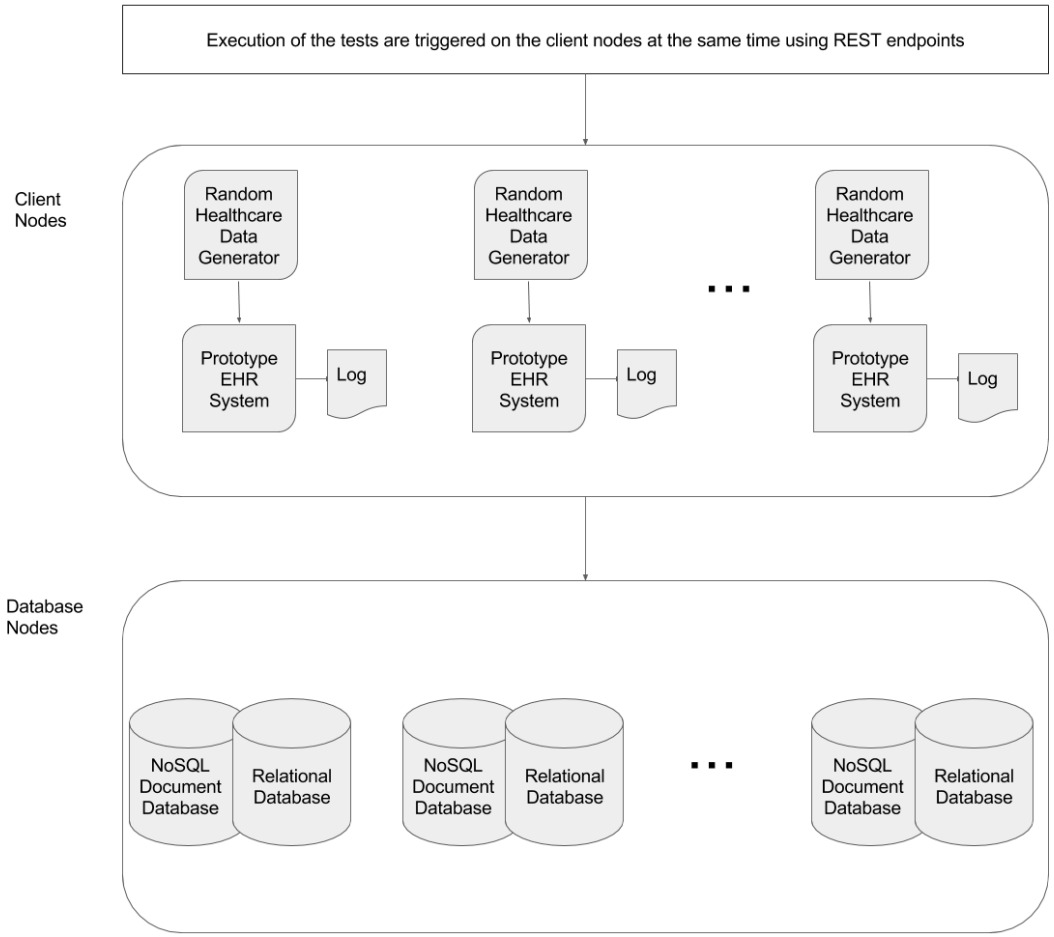


Figure 4.4. The relationship between IT artefacts (Random Healthcare Data Generator, Prototype EHR System) and database nodes

The following methods are used in the Prototype EHR System: (1) Insert; (2) Update by NHDID; (3) Delete by NHDID; and (4) Query – Patient Identifier. For the purpose of querying the databases for data sharing purposes, Patient ID is sent to the Prototype EHR System as input and all relevant healthcare records are queried against the databases and returned to the client. This simulates the scenario of a physician or an emergency crew accessing the previous healthcare history of a particular patient from a distributed EHR system.

4.7 Conclusion

This chapter described the design and development of the IT artefacts used in this research to evaluate the performance of a NoSQL database in managing EHRs in a distributed environment comparative to a relational database as a crucial step in the

Design Science Research Methodology used in this study. Minimum data sets and relevant data element specifications have been identified for the Australian healthcare domain and a relational and a NoSQL healthcare data model has been established for the purpose of this research.

Based on the data elements identified for the NoSQL and relational data models, and relevant healthcare statistics that are made publicly available by the Australian Institute of Health and Welfare, the Random Healthcare Data Generator artefact is developed to generate synthetic EHRs that will represent the data characteristics of the Australian healthcare domain. Sample healthcare data was generated by executing the Random Healthcare Data Generator and the data generated by this artefact was validated against the original public healthcare statistics. It is observed that the generated healthcare data demonstrates a similar statistical distribution to the real-world data. Therefore, the Random Healthcare Data Generator artefact has been found to be valid and sufficient for the requirements of this research.

Finally, another important artefact for the purposes of this study—prototype EHR system—was developed. This artefact handled the database operations and recorded associated performance metrics in a simulation of a large scale EHR system in order to conduct a performance evaluation of a NoSQL database comparative to a relational database in such an environment. The relational and NoSQL data models, Random Healthcare Data Generator and distributed prototype EHR system (software artefacts) developed and discussed in this chapter enabled the evaluation and comparison of the performance of insert, update and delete operations and EHR sharing, as well as scalability and complex query capabilities of NoSQL databases and relational databases in an Australian healthcare context.

Chapter 5 – Simulation and Evaluation

5.1 Introduction

The next step in a Design Science Research Methodology following the development of the IT artefacts is the evaluation of the IT artefacts solving an identified real world problem. Therefore, this chapter reports on the main results of the evaluation of the performance, scalability, EHR sharing and data analysis capability of NoSQL databases in comparison to relational databases. After designing appropriate relational and document data models with data structures for storing EHRs, this PhD study developed two software artefacts: a Random Healthcare Data Generator to generate synthetic EHRs; and a Prototype EHR System to simulate the database operations in a large scale EHR system, as described in the previous chapter four.

An appropriate relational database (MySQL) and an appropriate NoSQL database (Couchbase) were chosen because of their suitability for a performance evaluation in a distributed EHR system, which was one of main objectives of this research. A cloud environment on Amazon Elastic Compute Cloud was established as the testing platform for the evaluation of the performance of NoSQL database comparative to a relational database in a distributed EHR system by creating nodes using pre-configured images. Using this environment helped to provide easy scaling and configuration of database nodes. The simulated healthcare data required for the performance, scalability, EHR sharing and complex query testing was generated by the Random Healthcare Data Generator artefact. The tests are performed by sending this synthetic data to the prototype EHR system. This IT artefact managed the execution of database operations and recording of the performance metrics for the tests chosen to evaluate a NoSQL database comparative to a relational database in a distributed EHR system environment. Structure of this chapter is presented in Figure 5.1.

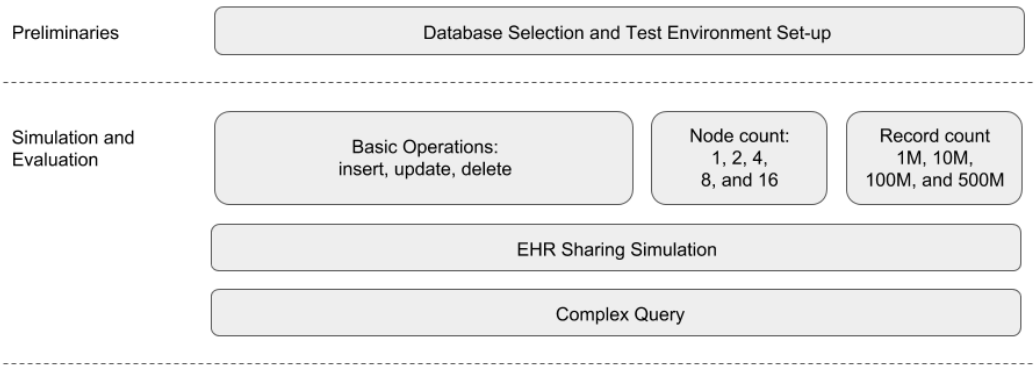


Figure 5.1. Structure of Chapter 5

5.2 Database Selection

There are numerous products available for both relational and NoSQL databases that include commercial, free and open-source databases. Oracle, Microsoft SQL Server, MySQL, PostgreSQL and DB2 are the most widely used relational databases (SolidIT 2016). MongoDB, Cassandra and Couchbase are well-known NoSQL databases that are backed by either a commercial company or a well-established community (Avalon Consulting 2016).

This research assessed the performance of a NoSQL document oriented database comparative to a relational database using simulation generated EHRs with tests run on the Amazon Web Services Elastic Compute Cloud environment. Therefore, a relational database alternative that is capable of running in a clustered architecture and supporting sharding is considered suitable. MySQL has been chosen as a relational database to be tested in this research as it is a widely adopted database alternative that satisfies the requirements of this study and also performs well compared to other candidate relational databases that are freely available (Oracle 2011; Souley & Mohammed 2013). For the MySQL database, the Record Identifier and Person Identifier fields in the main tables such as tblPerson, tblPatient, tblAdmittedPatientCare, tblNonAdmittedPatientEmergencyCare are used as shard keys for data distribution across multiple nodes.

For the NoSQL database, a document oriented database is a better choice as the EHR is actually a document consisting of healthcare related information given the business domain for testing is a simulation of a large scale EHR system. The suitability of

document databases for the purpose of this research has been discussed thoroughly in Chapter 2. There are a number of suitable alternatives for NoSQL Databases such as Couchbase and MongoDB to store documents. Couchbase was chosen as the NoSQL document database to be tested in this research because of the following superiorities over other document oriented databases: (1) shared-nothing architecture requiring less number of nodes for similar scenarios; (2) built-in managed in-memory caching architecture; and (3) views for incremental MapReduce operations (Vohra 2015).

In addition to the technical reasoning behind the database selection process, Couchbase is already being used in healthcare practice by the Turkish Ministry of Health in its National EHR System and MySQL is used by eClinicalWorks, one of the largest internet-based EHR systems in the United States (Oracle 2016). Therefore, MySQL and Couchbase are considered best choices to provide a performance comparison of NoSQL databases and relational databases for this research in the healthcare domain.

MySQL database and Couchbase database have different architectures for storing and managing data and it is acknowledged by the researcher that relational databases and NoSQL databases are not directly comparable in technical specifications. However the tests conducted in this research are concerned with a performance evaluation of basic database operations and scalability, EHR sharing and data analysis (complex querying) capability of the two selected databases, a NoSQL document database (Couchbase) and a relational database (MySQL). Hence, the performance comparison of these two databases using the same test scenarios is suitable and justifiable based on previous literature to achieve the outcome intended in this research.

5.3 Setting up the distributed test environment and scenarios

Following the selection of the database alternatives, a cloud environment was set up for the execution of tests in a range of scenarios.

5.3.1 Establishing cloud environment

Amazon Elastic Compute Cloud (EC2) has a marketplace that can provide and easily deploy readily-available and configured servers running particular software or services (Amazon 2016). These marketplace server images contain already installed software using best practices and configured by either professionals or the relevant

software vendor. Furthermore, Amazon EC2 has options to use SSD disks with predefined IOPS (input output per second) values which allows fine tuning of servers to a desired level of IO bandwidth, in addition to a wide range of CPU and memory options available in server configuration selection step (Amazon 2016). Therefore, a testing environment using readily available server images on Amazon EC2 was set up for all test scenarios which are identified and described in the next sub-section. MySQL 5.6.27 and Couchbase 4.1 Community Edition versions are used in the tests.

5.3.2 Test scenarios

TPC-H benchmarking method is widely used, along with YCSB framework, to evaluate database performances (Barata, Bernardino & Furtado 2014; Thanopoulou, Carreira & Galhardas 2012). However, these methods and frameworks use a predefined set of tables and queries to be executed in various server and client configurations (Meinel et al. 2015). On the contrary, this research focuses on Australian healthcare domain where all tables, fields and data are carefully selected to simulate a real-life healthcare data storage and sharing environment, thereby achieving one of the main objectives of this research.

TPC-H benchmarking has a well-established client, server, number of records and other similar configuration scenarios, besides the dataset and query definitions. Therefore, the following configuration alternatives for different aspects such as number of nodes, number of rows, etc. are derived from two sources, namely, (1) real life healthcare statistics and (2) TPC benchmarking in order to establish a wide spectrum of valuable tests for the purposes of this research (see Table 5.1).

Configuration of scenarios for each database (Couchbase, MySQL)		**Number of health records stored for each scenario**				**Number of scenarios**
Number of database nodes for each scenario	1	1M	10M	100M	*	3
	2	1M	10M	100M	*	3
	4	1M	10M	100M	*	3
	8	1M	10M	100M	500M	4
	16	1M	10M	100M	500M	4

Table 5.1 Configuration Scenarios for Performance Tests

By establishing these distributed EHR system configurations and scenarios setups, performance and scalability is compared between the selected relational and NoSQL databases.

The following parameters are measured: (1) Execution time in milliseconds; (2) Operations per second; and (3) Data size.

The following operations are tested: (1) Insert; (2) Update; (3) Delete; (4) EHR Sharing; and (5) Complex Query.

All results from these test scenarios are presented and described in the subsequent sections.

5.4 Running the tests

For database nodes, 64-bit m4.2xlarge EC2 instances are created which have 8 virtual CPUs (hyper threaded cores) with 32 GB of RAM assigned to each. These instances have EBS-optimised (high performance) storage and high network throughput. 256 GB SSD disks have been mounted on each EC2 instance with 768 IOPS guaranteed IO rate.

For clients, c4.4xlarge EC2 instances are created. These instances have been chosen due to their higher frequency 16 virtual CPUs and 30 GBs of RAM in order to facilitate faster data generation and storage of generated data in memory before sending it to the prototype EHR system for storing into databases and collecting statistical data about execution time, etc.

In the following sub-sections mainly execution times and number of operations per second values are presented for data insertion, update and delete operations in a range of configurations having a different number of database nodes.

Conducting tests for 500M records was not possible due to disk and memory limitations of single, 2- and 4-node Couchbase clusters. Therefore, 500M records tests were conducted for 8- and 16-node configurations only.

5.4.1 Simulation of data insertion

The data generated by the random healthcare generator has been saved in an in-memory data structure using a generated key which is also used as a Patient – Record identifier, a unique identifier for each EHR.

As a first step, a single node **Couchbase database** has been deployed as the designated EC2 database instance. Then EHRs are inserted into the database using a .NET API provided by Couchbase. The execution time for each insert operation and number of records inserted for each second have been saved into an in-memory structure called ConcurrentBag in .NET. This test has been conducted for 1M, 10M, and 100M insert operations.

Similarly, the **MySQL database** deployed on an identical instance has been used to insert EHRs. T-SQL insert statements have been generated for each EHR for inserting relevant data into relevant relational tables. The set of insert statements are executed against the MySQL database in a single transaction for each EHR. Therefore, multiple insert statements for various tables comprise a single EHR and, thus, a single transaction. Although it is possible and can be more efficient to do bulk insert operations in relational databases, this performance test scenario required individual insert operations to be executed for each EHR as the test simulates healthcare service providers sending single EHR data for a particular patient visit to the EHR system. For this relational database scenario, the number of transactions per second and average response time for a transaction is measured.

5.4.1.1 Insert operations on single node

The data insertion operation has been executed by 8 parallel client threads as this has been identified as the maximum number of possible threads to insert data to a single Couchbase node due to the following limitations. The client threads generated around 900 Mbits of network traffic and the Couchbase node had a 1 Gbit network connection. Therefore, a network limitation has occurred. In addition to this network limitation, the disk write queue for Couchbase has been steady at this level and the drain rate was equal to the fill rate. Any higher value of insertion caused the fill rate to be higher than the drain rate when tested on the same machine, which would eventually cause out-of-memory errors.

Based on these limitations it is concluded that 8 concurrent client threads that will insert pre-generated EHRs to Couchbase database is the optimal configuration for the single node scenario.

The results for execution times and number of operations per second for the NoSQL Couchbase database are presented in Table 5.2 and Table 5.3 respectively.

	Execution time in milliseconds					
Number of records	**Mean**	**SD**	**Min**	**Max**	**Median**	**CI (95%)**
1M	0.153702	0.007504	0.142053	0.170902	0.154140	0.002133
10M	0.150956	0.016277	0.131827	0.314699	0.147973	0.001437
100M	0.148133	0.010493	0.129679	0.251630	0.147457	0.000305
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.2 Execution time statistics in milliseconds for data insert operations on single-node Couchbase database

	Number of operations per second					
Number of records	**Mean**	**SD**	**Min**	**Max**	**Median**	**CI (95%)**
1M	20000	603	18790	22728	19972	171
10M	20186	2011	8154	44335	20239	178
100M	20070	2180	768	28703	20138	63
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.3 Number of insert operations per second on single-node Couchbase database

Couchbase seems to be able handle about 20,000 concurrent executions successfully and has a mean response time of around 0.15 milliseconds in a single node configuration. Insertion of 500M records was not possible due to the total available disk and memory in this single node configuration. These values are consistent with the benchmarks published by Couchbase which state 1 million operations per second can be achieved using 50 Couchbase nodes (Biyikoglu 2016).

Then, the data insertion operation has been executed on the MySQL database using the same number of client threads. Connection pooling is used to optimize

performance. The results for execution times and number of operations per second for MySQL database are presented in Table 5.4 and Table 5.5 respectively.

	Execution time in milliseconds					
Number of rec.	Mean	SD	Min	Max	Median	CI (95%)
1M	35.522702	10.016451	18.958060	73.857323	33.808805	1.386193
10M	62.360366	10.341411	24.321896	110.490316	61.849190	0.373079
100M	82.523779	16.061316	33.404720	669.784922	80.987136	0.160361
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.4 Execution time statistics in milliseconds for data insert operations on single MySQL database instance.

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	5308	1207	1937	7428	5364	167
10M	4809	673	2373	8203	4786	24
100M	3861	633	1281	7733	3840	6
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.5 Number of transactions per second for data insertion on single MySQL database instance.

MySQL was able to handle around 5000 transactions per second while the size of the database is relatively small, then started to slow down as the size of the database increased to 10 million and 100 million records stored, as shown in Table 5.4. While the average response time was around 35 milliseconds for 1M insert operations, it gradually increased to an average of 82 milliseconds when the number of records inserted was 100 million, as presented in Table 5.5.

Each insert transaction contained multiple insert statements for various tables for a particular EHR for MySQL. Although the actual number of T-SQL insert statements executed per second was more than 50,000 on average, the number of EHRs saved into the MySQL database was 5,308 on average.

In comparison to Couchbase, MySQL performed slower in both number of records inserted and the average response time. Figure 5.2 shows the average number of

records inserted per second for both Couchbase and MySQL databases comparatively in a single-node configuration.

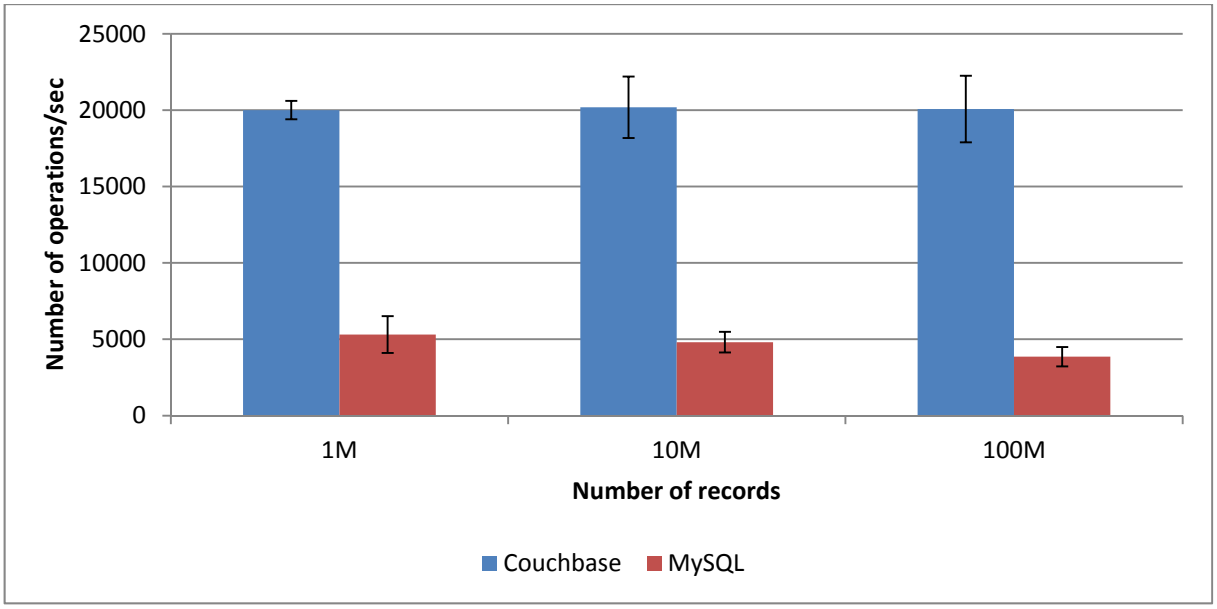


Figure 5.2. Average number of records inserted per second with standard deviations for Couchbase and MySQL in single-node configuration.

Furthermore, the MySQL database slows down when the total number of records increases. The number of records inserted per second decreases, as presented in Table 5.5, and the average execution time increases by more than 100%—as shown in Table 5.4.

5.4.1.2 Insert operations on two nodes

Based on the limitations mentioned for the single node configuration, 16 concurrent client threads have been used for data insertion on two database nodes. Average response time and average number of records inserted per second for the NoSQL Couchbase database 2 node cluster is shown in Table 5.6 and Table 5.7 respectively.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.273550	0.059515	0.154106	0.346323	0.290606	0.021103
10M	0.156874	0.007060	0.142867	0.199279	0.156266	0.000866
100M	0.153920	0.011270	0.134470	0.379438	0.152916	0.000462
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.6 Execution time statistics in milliseconds for data insert operations on 2-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	29962	4405	17591	39124	29037	1562
10M	37709	2641	20554	58350	37463	324
100M	38378	3016	19417	71109	37994	124
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.7 Number of insert operations per second on 2-node Couchbase database

Couchbase was able to handle around 50% more insert operations per second for the insertion of 1M records and performed even better when inserting higher number of records. The number of records that can be inserted per second in a 2-node Couchbase cluster for 100M records was almost double the number for a single node configuration. In addition to that, the average response time was similar in both configurations.

However, the situation was not similar for MySQL database. Average number of records inserted per second and the average response time for MySQL database 2 node cluster are shown in Tables 5.8 and 5.9.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	40.638814	11.995656	11.298356	70.990462	42.213457	1.299020
10M	57.137738	10.976489	16.457215	366.366510	58.107731	0.445253
100M	90.821534	17.925550	34.821367	544.997372	90.003622	0.253464
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.8 Execution time statistics in milliseconds for data insert operations on 2-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	6917	741	4388	8912	6965	80
10M	6632	473	4060	8547	6644	19
100M	4749	779	2142	7946	4725	11
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.9 Number of insert operations per second on 2-node MySQL database

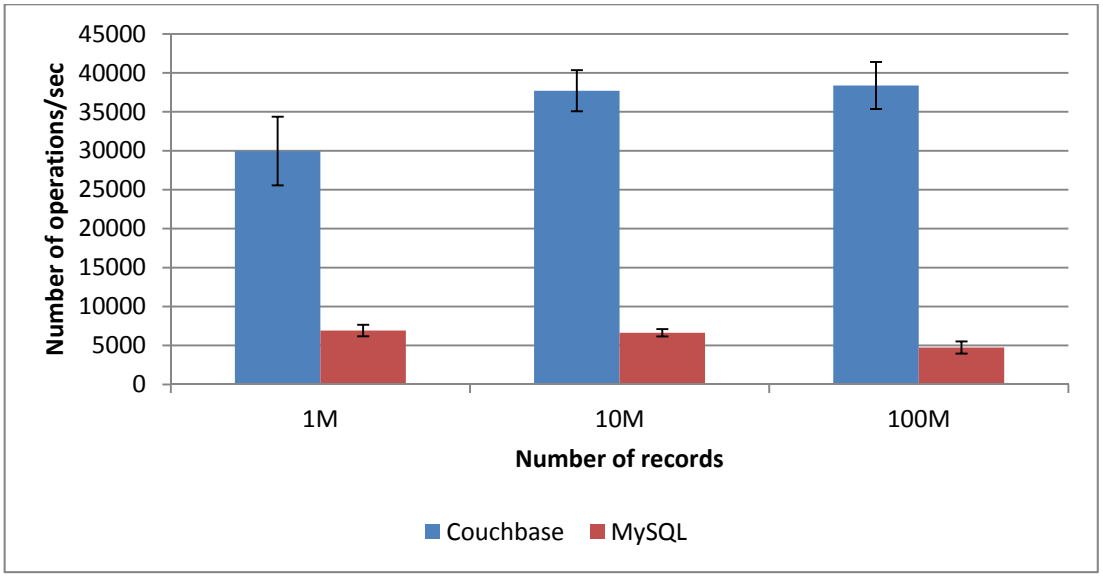


Figure 5.3. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 2-node configuration.

Although there is an improvement in the average number of records inserted per second for MySQL database, it is not as significant —as can be seen in Couchbase scenario (see Figures 5.1 and 5.2). Response times were similar for single node and 2-node MySQL configurations, but the improvement was around 25% in number of records inserted per second. MySQL also seemed to become slower when the total number of inserted records became higher, as shown in Table 5.8.

5.4.1.3 Insert operations on 4 nodes

The number of client threads were adjusted to the point that increasing the number of clients did not have any positive effect on performance. As a result, 32 concurrent client threads have been used for data insertion on a 4-node configuration, each running parallel operations and there is connection pooling in place. The average execution time and the number of records inserted per second for a Couchbase 4 node cluster is shown in Table 5.10 and Table 5.11.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.206894	0.044502	0.149733	0.284753	0.203810	0.019731
10M	0.166319	0.006498	0.149453	0.193299	0.166342	0.000787
100M	0.169957	0.007292	0.153517	0.210167	0.169195	0.000501
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.10 Execution time statistics in milliseconds for data insert operations on 4-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	73106	4760	63695	82238	72826	2110
10M	71800	3204	55043	93055	71456	388
100M	88963	4268	69680	114960	88399	293
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.11 Number of insert operations per second on 4-node Couchbase database

In a 4-node configuration, Couchbase has performed better than it did for a 2-node configuration and a near linear scalability is demonstrated. The number of records inserted per second was almost double compared to a 2-node configuration; and response times were still well under one millisecond level, similar to the level observed for a 2-node configuration.

MySQL also performed better in a 4-node configuration than a 2-node configuration, however, the improvement was not as significant as the Couchbase scenario. The average execution time and the number of records inserted per second for MySQL database 4 node cluster is shown in Table 5.12 and Table 5.13.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	47.387782	21.499094	19.288564	92.749640	39.307242	3.367510
10M	69.316874	23.359773	22.240973	150.120026	68.531566	2.106103
100M	91.120392	19.308699	30.360149	172.532989	97.877772	1.151259
500M	121.847202	21.247962	46.127519	467.244158	118.454975	0.154861

Table 5.12 Execution time statistics in milliseconds for data insert operations on 4-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	8222	1218	5459	11151	8273	191
10M	6454	1008	3491	8950	6578	159
100M	4902	736	2745	7969	4698	44
500M	4329	597	1926	11063	4348	4

Table 5.13 Number of insert operations per second on 4-node MySQL database

For MySQL in a 4-node configuration, there has been an improvement of up to 20% in the number of records inserted per second and the response times were similar to the previous configurations in MySQL. The average response time was over 120 milliseconds for the 500M insertion test in 4-node MySQL configuration, as shown in Table 5.12 and the number of records inserted per second in the same test is lower

than all 2-node configuration tests (see Tables 5.13 and 5.9). Figure 5.4 presents the average number of records inserted per second for both databases comparatively.

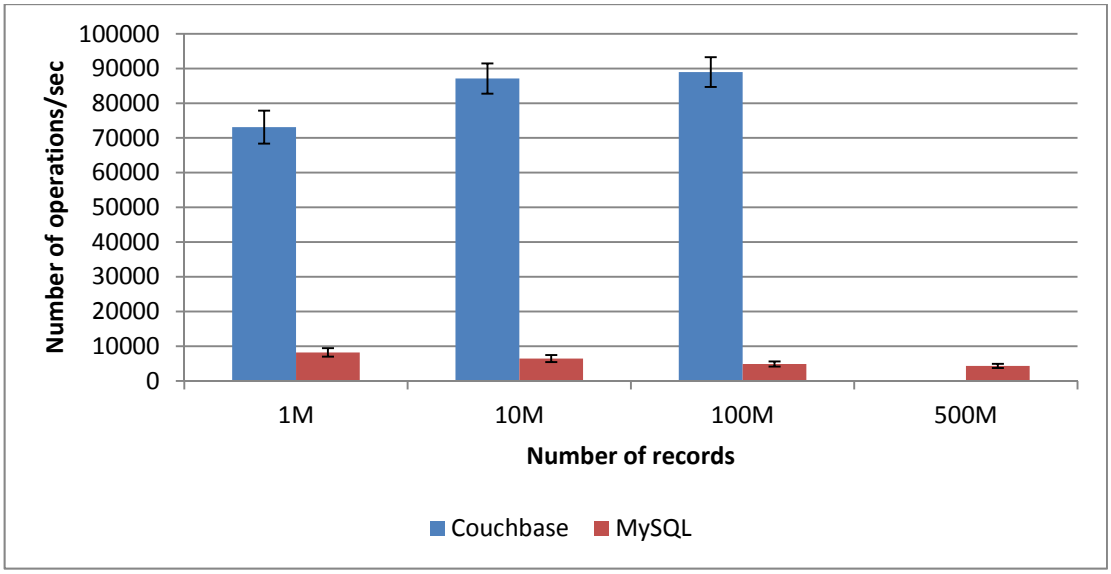


Figure 5.4. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 4-node configuration.

These results were consistent with the previous results which demonstrated that MySQL was getting slower as a result of a higher number of records. A number of potential reasons for performance degradation of records inserts with relational databases as the number of records increased include (1) the percentage of the data cached in memory reduces as the total size of data increases which causes more disk reads; (2) index fragmentation occurs for primary and foreign indexes; and (3) number of joins and number of concurrent operations effects the overall performance (Hadjigeorgiou 2013; Schmidt 2001; Souley & Mohammed 2013).

5.4.1.4 Insert operations on 8 nodes

The number of client threads has been adjusted for the maximum performance for Couchbase and MySQL respectively. Sixty-four concurrent client threads running parallel executions is configured for Couchbase and connection pooling for about 100 connections per client node is configured for MySQL.

While a consistent linear scalability is observed for Couchbase, average execution time for insert queries has increased in an 8-node configuration. Couchbase started responding slower than a 4-node configuration. Execution times for 4-node and 8-

node insert operations are shown in Tables 5.10 and 5.14 respectively. However, while the number of record inserts was elevated, the average response time was still under a millisecond. Couchbase seemed to perform slower in the 10M records test. This might be related to some internal process blocking the execution for a small period of time as the data files in the Couchbase folder seemed to expand very quickly at that stage, however, it was not significant enough to cause Couchbase to perform worse than for the 4-node configuration.

The average response time and the number of records inserted per second for an 8 node Couchbase cluster is shown in Table 5.14 and Table 5.15 respectively.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.232728	0.054227	0.154467	0.335185	0.214209	0.026136
10M	0.550048	0.214875	0.113364	0.866355	0.553557	0.049101
100M	0.262563	0.147796	0.099792	0.812308	0.207399	0.013026
500M	0.211525	0.158407	0.066649	1.766618	0.147815	0.006818

Table 5.14 Execution time statistics in milliseconds for data insert operations on 8-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	145581	11711	120755	160910	150835	5645
10M	113238	17299	86487	158969	110648	3953
100M	158681	15975	98637	202064	159892	1408
500M	154561	17069	90052	212758	156950	735

Table 5.15 Number of insert operations per second on 8-node Couchbase database

For MySQL in the 8-node configuration, the improvement was around 100% for a higher number of records. Response times and the number of records inserted per second were better than a 4-node configuration, particularly for a high number of records. Execution times and the number of records inserted per second for MySQL database 8 node cluster is shown in Table 5.16 and Table 5.17.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	45.083707	71.569724	18.000552	970.512634	37.768134	10.13464
10M	76.885133	33.871741	22.083073	984.841746	80.966209	2.089371
100M	81.334755	13.768478	34.191082	993.905688	81.127703	0.279595
500M	80.352173	11.349797	29.416558	1000.506708	80.048438	0.170435

Table 5.16 Execution time statistics in milliseconds for data insert operations on 8-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	11647	1416	7412	15039	11845	200
10M	10611	899	7621	13916	10540	55
100M	9978	669	7436	12902	9969	14
500M	9518	636	6741	12264	9503	10

Table 5.17 Number of insert operations per second on 8-node MySQL database

Although MySQL seemed to handle 500M records better in an 8-node configuration compared to a 4-node configuration based on the average number of insert operations per second presented in Table 5.13 and Table 5.17, it slowed down when the number of records increased, however, decrease in performance was not as significant as the previous configurations.

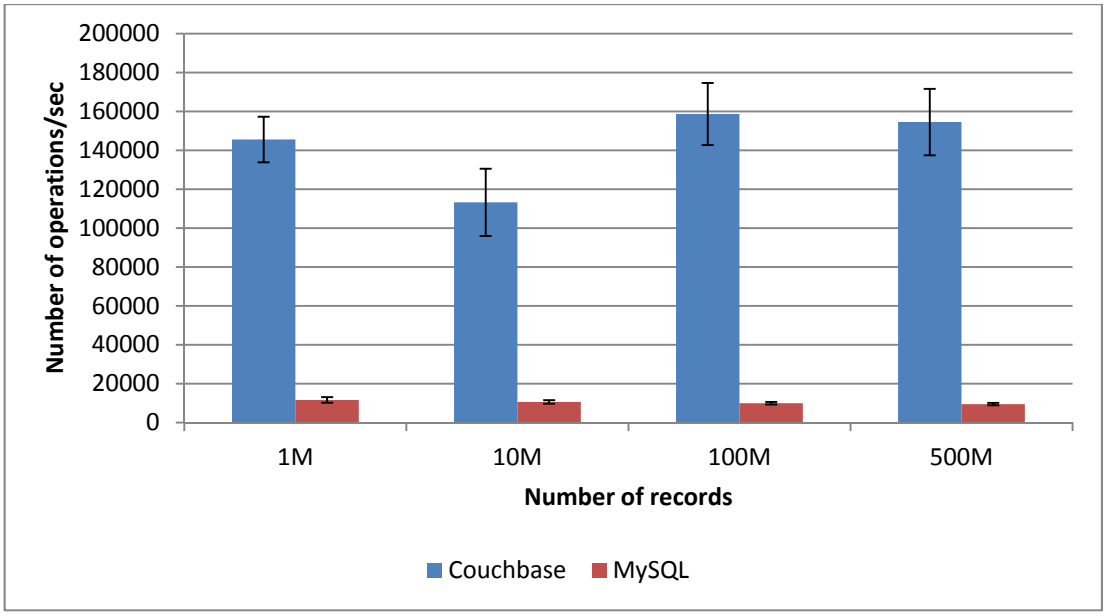


Figure 5.5. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 8-node configuration.

The number of records inserted per second was around 10,000 for the tests for MySQL, while it was around 150,000 for Couchbase—as evidenced in Figure 5.5. The performance gap between Couchbase and MySQL for number of records inserted has been found to be greater when the configuration of the database cluster involved a higher number of nodes (see Figure 5.5).

5.4.1.5 Insert operations on 16 nodes

Finally, the insertion tests have been executed on a 16-node configuration with the number of client threads adjusted accordingly. The performance improvement for Couchbase seemed to be better when the number of nodes increased from 8 to 16. Average response times and the number of records inserted per second for a 16-node Couchbase cluster is shown in Table 5.18 and Table 5.19.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.175442	0.023516	0.138094	0.213225	0.172074	0.011006
10M	0.202226	0.039837	0.139622	0.316384	0.191291	0.014363
100M	0.324992	0.107322	0.137578	0.710155	0.319529	0.013396
500M	0.249412	0.393963	0.089050	4.415888	0.150457	0.022646

Table 5.18 Execution time statistics in milliseconds for data insert operations on 16-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	281894	10307	257482	294589	284242	4824
10M	394801	38556	329670	457613	397620	13901
100M	290595	24159	228787	420065	287591	3015
500M	325040	16630	261629	376375	326831	956

Table 5.19 Number of insert operations per second on 16-node Couchbase database

While there is an improvement of more than 100% in insertion tests for 10M and 500M for Couchbase, the improvement in MySQL was around 60%. However, response times for MySQL increased significantly in this 16 node cluster test scenario, particularly for higher numbers of records. The average execution time and the number of records inserted for MySQL database 16 node cluster are shown in Table 5.20 and Table 5.21.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	69.10552	73.230766	23.385589	993.105480	60.412052	10.369854
10M	105.28047	38.616143	30.794788	1052.56921	104.408717	2.382029
100M	120.56583	22.304410	49.284543	1013.35603	117.729079	0.452933
500M	122.51623	21.722253	51.768716	1050.60043	119.463696	0.326193

Table 5.20 Execution time statistics in milliseconds for data insert operations on 16-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	17589	1804	12708	22359	17715	255
10M	16579	1135	12827	20581	16513	70
100M	15418	886	12066	18972	15398	18
500M	14332	842	10699	17795	14320	13

Table 5.21 Number of insert operations per second on 16-node MySQL database

EHR insertion tests are concluded with these 16 node cluster results and Couchbase demonstrated linear scalability, achieving 394,801 records per second for insertion (see Figure 5.6) and average response times of under a millisecond consistently, having a maximum response time of 4 milliseconds. However, in comparison, MySQL was only able to insert a maximum of 17,589 records per second (see Figure 5.6) while sometimes having a maximum of around 1000 milliseconds (1 second) response time, as shown in Tables 5.16 and 5.20.

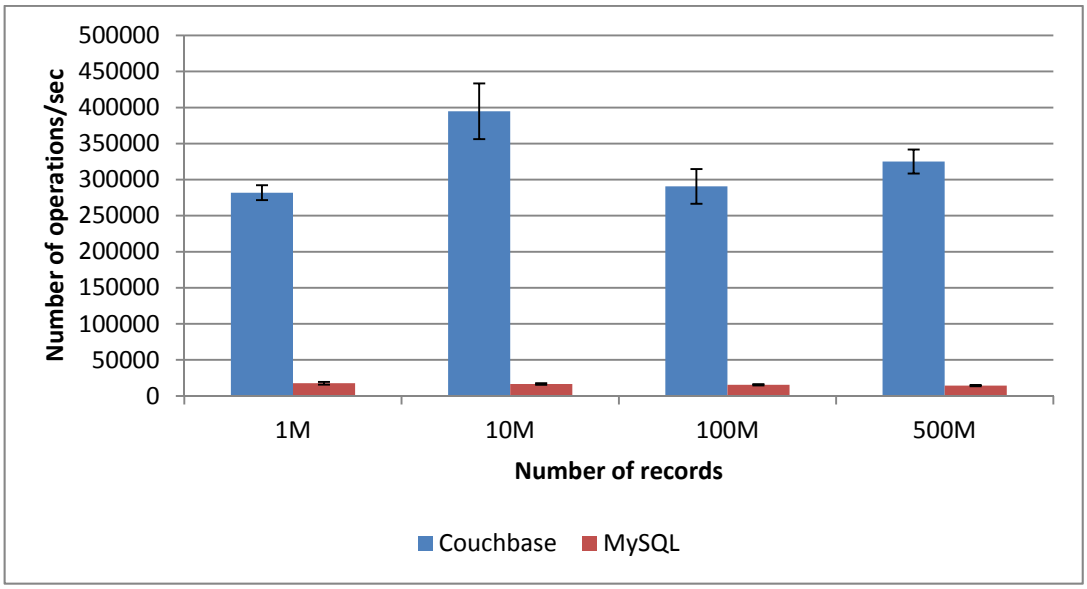


Figure 5.6. Average number of records inserted per second with standard deviations for Couchbase and MySQL in 16-node configuration.

5.4.2 Simulation of update operations

Once performance testing of different ranges of records written to the databases using insert operations was completed, testing of update operations was executed on randomly selected records. Update Records Tests are run on various configurations with different number of nodes and different number of stored records. Although total number of stored records are different, 1 million update operations are executed on each test and execution time for each update operation and the number of update operations per second are measured. Records on Couchbase and MySQL have been updated using the record identifier key field.

Updates have been executed using the same SET operation on Couchbase, which was used to insert records. The key and the updated version of the relevant EHR are used as parameters of the SET command.

However, update operations on the MySQL database are performed differently. EHRs stored on MySQL were saved into multiple tables and thus update operations are planned to change a random part of the data, affecting only one or some of the tables. Therefore, update operations on the MySQL database were expected to run efficiently. Furthermore, conducting tests for 500M records was not possible for 1, 2, and 4-node configurations due to memory and disk limitations.

5.4.2.1 Update operations on single node

As update operations are similar to insert operations for the Couchbase database, the same number of client threads was used for data update operations while executing tests on Couchbase.

The results for update operations on Couchbase are presented in Table 5.22 and Table 5.23.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.149371	0.006808	0.136411	0.163198	0.149073	0.001955
10M	0.154444	0.007620	0.140854	0.204384	0.154060	0.001034
100M	0.157729	0.009444	0.140996	0.182085	0.159075	0.002684
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.22 Execution time statistics in milliseconds for data update operations on single-node Couchbase database

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	20138	356	19292	20774	20188	102
10M	21022	1891	16291	27752	20353	256
100M	19714	499	18145	20500	19819	141
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.23 Number of update operations per second on single-node Couchbase database

The single node Couchbase database seems to be able to handle a similar number of update operations as in insert operation tests. An average of around 20,000 update operations were achieved, while execution time for update operations was around 0.15 milliseconds, which is similar to the value achieved for the insert operations as presented in Tables 5.22 and 5.23.

MySQL has demonstrated a similar performance on update operations compared to insert operations, which was shown in Table 5.4, however, the average execution time was better than for insert operations. Average execution time and number of operations for MySQL are shown in Table 5.24 and Table 5.25 respectively.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	31.489434	135.192489	8.934383	1015.55274	11.963705	36.547649
10M	34.099787	125.445826	9.720098	1133.77026	18.670848	28.098334
100M	47.443898	48.144115	18.207891	509.45553	40.081116	8.854346
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.24 Execution time statistics in milliseconds for data update operations on single node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	4879	1656	1374	10697	4487	448
10M	3883	993	663	6844	3747	222
100M	3006	1020	932	6497	2973	188
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.25 Number of update operations per second on single node MySQL database

A comparison of the average number of records updated per second for Couchbase versus MySQL databases on a single node configuration is shown in Figure 5.7.

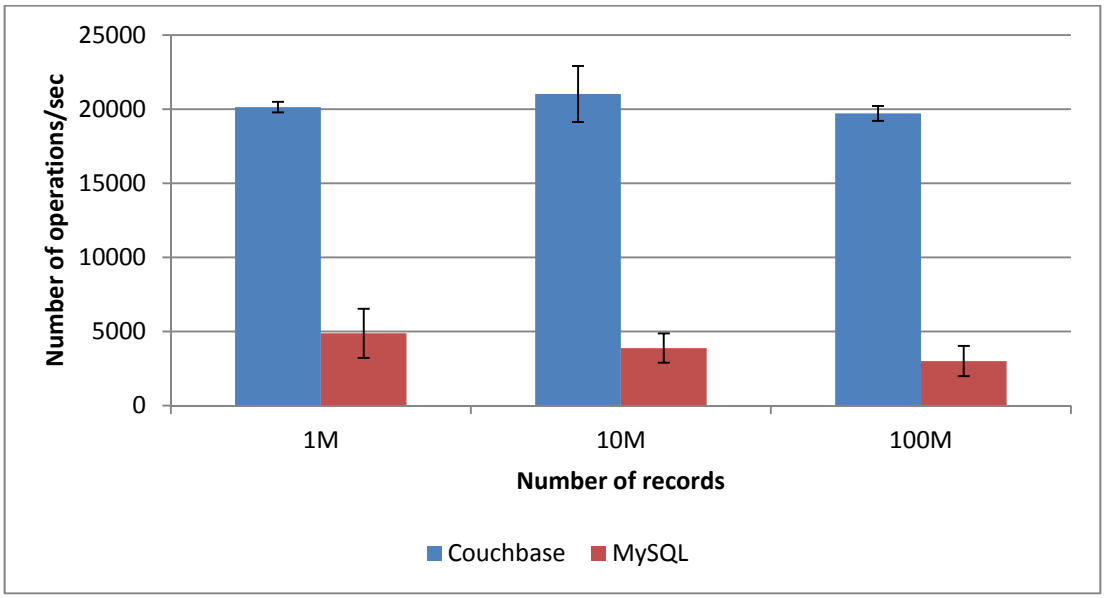


Figure 5.7. Average number of records updated per second with standard deviations for Couchbase and MySQL in single-node configuration.

5.4.2.2 Update operations on two nodes

Update operations have been executed on both Couchbase and MySQL on a 2-node cluster configuration. It was not possible to conduct tests for 500M records on both databases due to memory and disk limitations of a 2-node cluster.

The change in performance between single node and 2-node cluster configuration tests was similar for both databases. Average execution time and average number of update operations for Couchbase is presented in Table 5.26 and Table 5.27.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.1549886	0.005126	0.146910	0.166523	0.155630	0.002116
10M	0.1539261	0.006375	0.140988	0.165904	0.154813	0.002631
100M	0.1556281	0.007388	0.140134	0.175461	0.156387	0.002099
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.26 Execution time statistics in milliseconds for data update operations on 2-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	38361	2032	34807	42763	37742	839
10M	38868	1474	36563	42134	38424	609
100M	38053	1212	35552	42165	37690	345
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.27 Number of update operations per second on 2-node Couchbase cluster

Execution times were around 0.15 milliseconds for update operations, which is similar to the single-node configuration test and the average number of records updated per second has been improved by around 90% with an increase from 20,000 to 38,000 on average. Similar improvement was observed for MySQL as well. Average execution time and the number of records updated per second for MySQL is shown in Table 5.28 and Table 5.29.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	34.579299	76.314712	9.894826	550.418276	18.771078	19.545105
10M	30.178145	117.718718	6.737242	878.199817	13.050963	32.131025
100M	35.877078	123.591271	12.857745	1061.08678	20.568325	29.253577
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.28 Execution time statistics in milliseconds for data update operations on 2-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	9120	2630	4933	16447	8630	674
10M	8530	2470	4676	15276	8358	674
100M	7111	1675	4897	12944	6872	397
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.29 Number of update operations per second on 2-node MySQL database

Execution time for update operations were about 32 milliseconds on average for a 2 node MySQL database cluster, as shown in Table 5.29. However, the improvements for execution time and number of records updated per second were better for higher numbers of records stored compared to a single node configuration as shown in the results presented in Table 5.25. Although the performance improvement for Couchbase was around 90%, it was 130% for MySQL in the scenario having 100M records stored. It is also observed that the performance of MySQL for update records operations has decreased for the higher number of records stored in the database as was the case for the insert operations test.

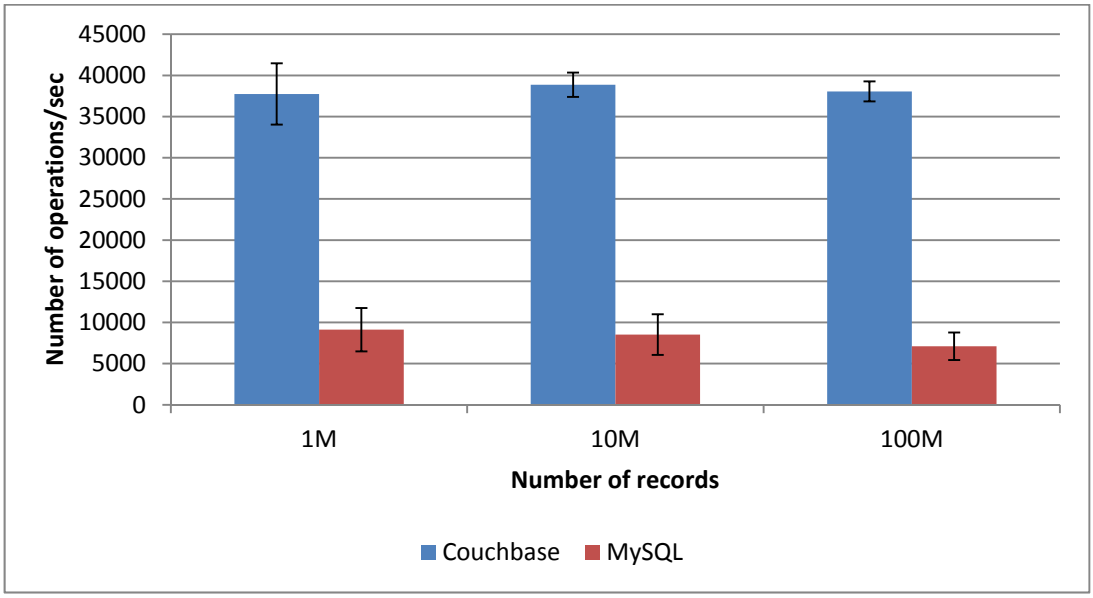


Figure 5.8. Average number of records updated per second with standard deviations for Couchbase and MySQL in 2-node configuration.

Furthermore, performance for MySQL seems to be better for update operations compared to insert operations as insert operations involve data addition to multiple tables (see Tables 5.9 and 5.29), while in a relational database such as a MySQL database, update operations change only some of the tables. Therefore, it can be expected for a relational database such as MySQL that partial updates could be executed slightly faster than inserts.

5.4.2.3 Update operations on 4 nodes

In the 4-node configuration, update operations are executed using a higher number of client threads. A higher number of client threads have been used for MySQL compared

to Couchbase to achieve maximum performance. As in insert operation tests, number of clients has been increased to the point that the marginal performance increase is zero for any additional client threads. Both MySQL and Couchbase demonstrated linear scalability and the performance increased significantly for the 4-node scenario compared to the 2-node scenario (see average number of records updated for 2-node configuration and 4-node configuration in Figure 5.8 and Figure 5.9 respectively).

The average execution time and number of records updated per second for three test scenarios for a 4 node Couchbase cluster is presented in Table 5.30 and Table 5.31.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.157392	0.003790	0.149789	0.164877	0.157317	0.001499
10M	0.162747	0.030386	0.143539	0.528638	0.154635	0.002757
100M	0.159543	0.012444	0.147914	0.217290	0.157191	0.005026
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.30 Execution time statistics in milliseconds for data update operations on 4-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	94043	2322	90880	99682	93582	919
10M	94730	6218	72179	138721	94799	564
100M	93208	949	91458	95280	93131	383
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.31 Number of update operations per second on 4-node Couchbase cluster

It is observed that Couchbase was able to respond consistently when performing data update operations. The average response times were around 0.16 milliseconds and the number of records updated per second increased by more than 100% compared to 2-node configuration (see Tables 5.27 and 5.31). A similar performance improvement is also observed for a 4 node MySQL database cluster. The average execution time and the average number of update operations for MySQL is presented in Table 5.32 and Table 5.33.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	48.740752	186.48895	6.521561	1166.43691	7.428957	55.380385
10M	35.334547	160.76209	6.511512	1105.30389	7.366239	38.332368
100M	20.189276	6.128679	11.161256	42.254836	19.82484	1.909831
500M	15.640881	49.518282	8.110318	512.282202	10.30020	9.677833

Table 5.32 Execution time statistics in milliseconds for data update operations on 4-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	21453	4526	14181	31169	20481	1344
10M	20464	4510	11653	31034	20738	1075
100M	16675	2756	11381	24808	16849	859
500M	14397	4299	5556	23426	14321	840

Table 5.33 Number of update operations per second on 4-node MySQL database

MySQL has performed significantly better, having around 130% increase in the number of records updated per second compared to a two node MySQL database cluster. Furthermore, mean execution time has dropped to 15.6 milliseconds in the 500M records test. The average number of records updated per second decreases by the number of records—stored which is also consistent with the results of the insert tests for MySQL.

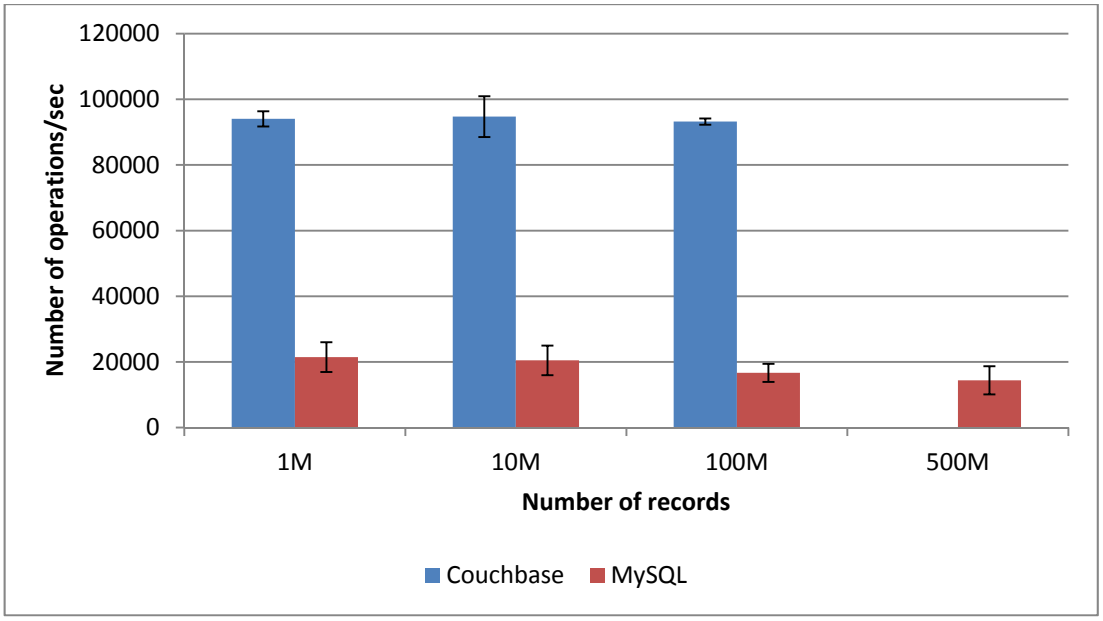


Figure 5.9. Average number of records updated per second with standard deviations for Couchbase and MySQL in 4-node configuration.

5.4.2.4 Update operations on 8 nodes

Update operations were executed for Couchbase and MySQL on an 8-node configuration. Couchbase seems to have increased response times, however it preserved near linear scalability for the average number of records updated. MySQL has also demonstrated a good scalability performance with around double the number of records updated per second and even smaller response times. The average execution time and the average number of update operations for Couchbase is presented in Table 5.34 and Table 5.35.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.122928	0.036114	0.052263	0.188523	0.124145	0.012041
10M	0.130890	0.013708	0.105906	0.168181	0.130123	0.004506
100M	0.101203	0.015948	0.075321	0.133392	0.097706	0.005564
500M	0.133138	0.017851	0.097745	0.172253	0.133185	0.005867

Table 5.34 Execution time statistics in milliseconds for data update operations on 8-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	168031	13908	145000	206844	166779	4637
10M	172153	11082	148334	197933	172841	3643
100M	176259	7457	163512	208036	175891	2602
500M	169740	11128	148469	200237	170156	3658

Table 5.35 Number of update operations per second on 8-node Couchbase cluster

Couchbase was able to handle a maximum number of around 200,000 update operations per second at times, however, mean was around 170,000 records per second—which is about an 88% increase from the 4-node configuration performance, as shown in Tables 5.31 and 5.35. However, the average execution times for all 8-node tests were between 0.10 and 0.13 milliseconds as presented in Table 5.34—which is still well below the one millisecond level.

MySQL has also demonstrated around 90% performance increase compared to a 4-node configuration. The average execution time and the average number of records updated for MySQL is shown in Table 5.36 and Table 5.37.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	17.797026	21.342206	7.067944	138.624876	11.23545	6.411908
10M	14.251006	20.487896	6.949741	138.621010	8.413221	5.822596
100M	13.284667	25.954801	6.805793	182.998415	7.995805	7.620615
500M	12.881169	10.818246	7.613767	94.863891	10.41849	2.347703

Table 5.36 Execution time statistics in milliseconds for data update operations on 8-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	39465	7177	22305	53751	37398	2156
10M	37307	9062	20332	54522	34504	2576
100M	30596	7897	11801	49157	31149	2319
500M	28471	8325	13800	44291	29359	1807

Table 5.37 Number of update operations per second on 8-node MySQL database

Although MySQL has performed significantly better in terms of average execution time compared to a 4-node configuration as shown in Tables 5.36 and 5.32, this was still 100 times higher than the execution times for Couchbase. MySQL also demonstrated consistently lower performance on higher number of stored records, while a higher number of records had no significant effect on the performance of Couchbase for update operations or insert operations.

It is also observed that the performance degradation between 100M and 500M stored records test that occurred in 8-node MySQL cluster is not as significant as the degradation observed between 100M and 500M stored records test in a 4-node configuration. The number of records stored has no significant effect for Couchbase (see Figure 5.9 and Figure 5.10).

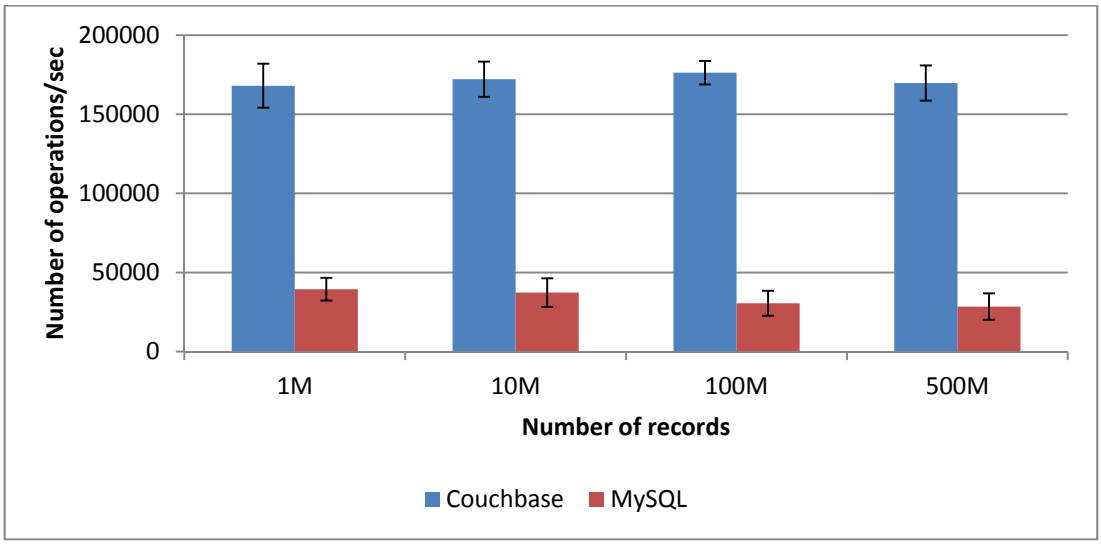


Figure 5.10. Average number of records updated per second with standard deviations for Couchbase and MySQL in 8-node configuration.

5.4.2.5 Update operations on 16 nodes

In the final update operations test, 16 nodes have been configured to handle the update operations load. The results of the update operations for a 16 node Couchbase cluster were quite similar to the 8-node test for Couchbase, demonstrating a near linear scalability. The average execution time and the average number of records updated per second for Couchbase is shown in Table 5.38 and Table 5.39.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.265266	0.021060	0.218065	0.298339	0.270838	0.010151
10M	0.145795	0.030815	0.112220	0.240975	0.139003	0.015324
100M	0.139815	0.031652	0.100709	0.225849	0.128105	0.015256
500M	0.119642	0.009769	0.105036	0.138452	0.116983	0.004858

Table 5.38 Execution time statistics in milliseconds for data update operations on 16-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	327908	22875	299620	387371	323316	11026
10M	335469	28615	299567	432730	328361	14230
100M	344284	22672	322870	421041	339613	10927
500M	367093	34777	340071	484526	354592	17294

Table 5.39 Number of update operations per second on 16-node Couchbase cluster

Couchbase achieved a mean of around 367,000 records updated per second, with a maximum of 484,000 records updated per second for a 16 node cluster, as shown in Table 5.39. The response time for each record update was affected significantly, being around 0.12-0.14 milliseconds for most of the cases (see Table 5.38).

MySQL showed around a 60% increase in the average number of records updated per second while almost preserving the average execution times for a 16 node cluster. The

average execution time for a record update and the average number of records updated per second for MySQL is shown in Table 5.40 and Table 5.41.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	19.404544	38.095892	7.051670	260.722528	9.378034	11.445273
10M	14.129704	19.517664	7.006595	138.898273	8.766027	5.546859
100M	13.977971	25.736622	7.038544	171.317033	8.229623	7.642833
500M	12.136113	7.475673	7.792066	52.773793	10.33582	1.622320

Table 5.40 Execution time statistics in milliseconds for data update operations on 16-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	65647	9972	46935	89341	64160	2996
10M	63051	10322	43342	84786	61671	2933
100M	49967	10412	27370	74785	48167	3092
500M	45108	11970	26132	68690	45159	2598

Table 5.41 Number of update operations per second on 16-node MySQL database

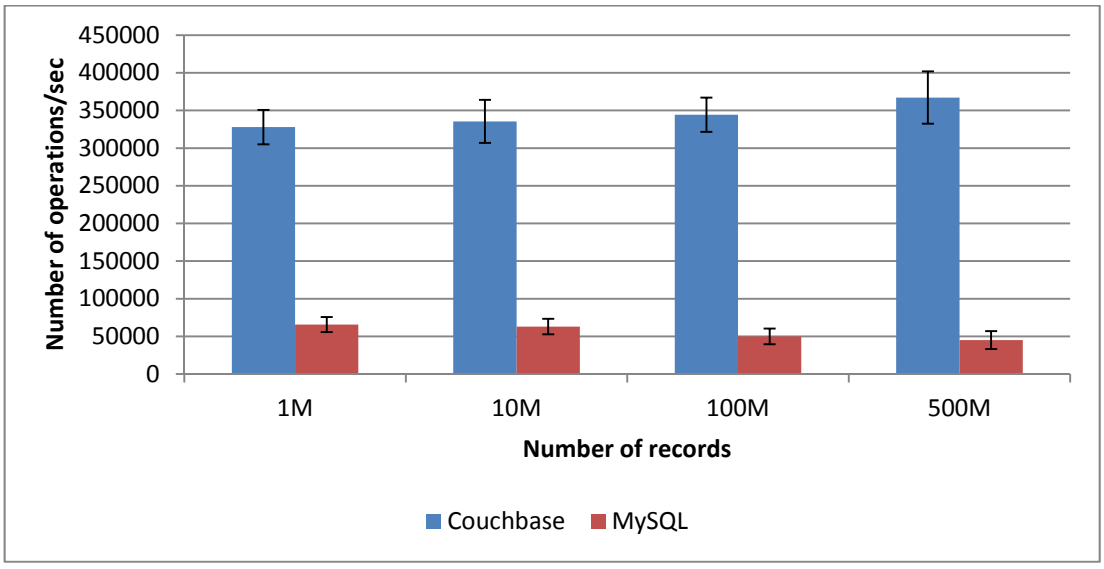


Figure 5.11. Average number of records updated per second with standard deviations for Couchbase and MySQL in 16-node configuration.

The average number of records updated per second for Couchbase and MySQL in a 16-node configuration are presented comparatively in Figure 5.11 which demonstrates that the performance of the Couchbase database is not affected by the number of records stored, however, the performance of MySQL database degrades for the tests which involve a higher number of records stored.

5.4.3 Simulation of delete operations

Following the insert and update operations, tests for delete operations are executed using a number of different configurations in terms of the number of nodes for both Couchbase and MySQL databases. The number of operations per second and execution time were measured and reported for each configuration (number of nodes). The number of client threads was adjusted based on the configuration to maximize the performance of both databases.

The number of records stored in the databases has an effect on the total items stored in memory (active items cached in RAM), therefore, deletion test was executed in a number of different test scenarios using different numbers of nodes and different numbers of records stored. However, conducting tests for 500M records was not possible for 1, 2, and 4-node configurations due to memory and disk limitations.

5.4.3.1 Delete operations on single node

In contrast to insert and update operations, delete operations do not send large amounts of data to databases. Therefore, a network bandwidth bottleneck did not apply to delete operations. A higher number of client threads are used to delete data for both Couchbase and MySQL. It is found that any additional number of client threads above 32 did not cause any improvement for the measured values on Couchbase database. The number of records mentioned in the test results indicates the total number of records stored in the relevant database before delete operations are executed.

For MySQL, delete operations demonstrated slower performance compared to update operations which only affect some tables. For MySQL, a delete operation results in an operation on all related tables to delete a single EHR. Conversely, for Couchbase—because of the way EHR data is stored in the data model of a document database—a

significantly higher number of records can be deleted per second compared to insert and update operations.

The average execution time in milliseconds and the average number of records deleted per second for Couchbase are presented in Table 5.42 and Table 5.43 respectively.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.183945	0.048137	0.145628	0.281925	0.158758	0.030585
10M	0.175649	0.034750	0.145874	0.268797	0.164056	0.022079
100M	0.268597	0.021036	0.242225	0.324657	0.264606	0.015048
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.42 Execution time statistics in milliseconds for delete operations on single-node Couchbase database

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	74141	9469	60385	84215	77279	6017
10M	78003	8022	60214	89268	79102	5097
100M	65088	2943	63439	72516	63735	2105
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.43 Number of delete operations per second on single-node Couchbase cluster

Couchbase is able to delete around 70,000 records per second, which is a higher value compared to single-node insert and update operations as shown in Tables 5.3, 5.23 and 5.43. The average response times were around 0.18 milliseconds for 1M and 10M records stored, however, it was 0.27 milliseconds when the number of records stored was 100M (see Table 5.42). In contrast, MySQL has demonstrated significantly lower performance when deleting records. The average execution times and the number of records deleted for MySQL in a single-node configuration is shown in Table 5.44 and Table 5.45.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	26.324059	75.464050	10.442080	980.387595	19.05796	11.672151
10M	35.588544	57.655111	13.632901	963.092700	29.82108	6.794830
100M	43.264956	73.023233	13.595287	1087.97424	24.52064	6.905395
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.44 Execution time statistics in milliseconds for data delete operations on single-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	3165	1122	1734	7363	2854	174
10M	2188	896	724	7163	2001	106
100M	1364	425	83	2203	1401	40
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.45 Number of delete operations per second on single-node MySQL database

The average number of records deleted per second for Couchbase and MySQL databases in a single node configuration is presented in Figure 5.12.

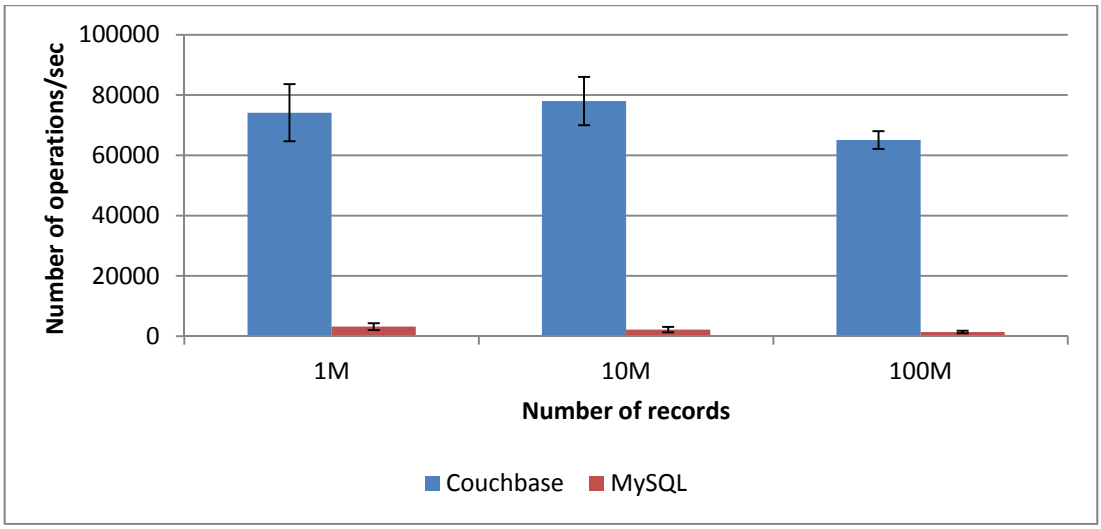


Figure 5.12. Average number of records deleted per second with standard deviations for Couchbase and MySQL in single-node configuration.

The average number of delete operations per second on MySQL was significantly lower compared to insert and update operations (see Tables 5.5, 5.25 and 5.45). The average response time was similar to the single-node insert and update operations (see Tables 5.4, 5.24 and 5.44), however, a maximum of about 1000 milliseconds is observed at times. Furthermore, a minimum of 83 delete operations per second is also observed for 100M records test. These results demonstrate that delete record operations are executed much slower in MySQL compared to insert and update operations; and MySQL has also slowed down when the total number of records stored increase for a delete operation test.

5.4.3.2 Delete operations on two nodes

Similar to the previous delete records operations tests, the number of client threads is adjusted for the maximum possible performance for both Couchbase and MySQL. Couchbase could handle 64 client threads, however, MySQL slowed down on any additional threads over 16.

Both Couchbase and MySQL have performed better for 2-node configurations, however, the increase in the number of records deleted per second was higher for Couchbase. MySQL demonstrated a performance increase of around 58% for delete records operations, while the performance increase in delete records operations for Couchbase was about 71%. The average execution times and the number of records deleted for Couchbase in a 2-node configuration is shown in Table 5.46 and Table 5.47.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.248538	0.038607	0.193474	0.353560	0.248568	0.023330
10M	0.309964	0.080658	0.233900	0.502251	0.276355	0.046571
100M	0.203230	0.035639	0.173989	0.274356	0.188020	0.023942
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.46 Execution time statistics in milliseconds for delete operations on 2-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	132559	8035	116942	151199	133318	4856
10M	129047	9765	119584	158947	127272	5638
100M	110994	8538	98544	127541	111623	5736
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.47 Number of delete operations per second on 2-node Couchbase cluster

Couchbase was able to delete around 130,000 records per second in 1M and 10M records tests, however, it slowed down to 110,000 for 100M records test as presented in Table 5.47. The average execution time was slightly higher than the previous tests, increasing to around 0.25 on average (see Table 5.46).

Although performance on a 2-node configuration was better for MySQL compared to a single-node configuration, it was not more than 60% for the average number of delete operations per second. The performance improvement observed between single node and 2-node configuration for 100M records stored was only 33%. The average execution time for a record deletion operation and the average number of delete operations per second for MySQL are shown in Table 5.47 and Table 5.49.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	34.992382	118.04716	12.825939	1069.02556	17.74394	25.011804
10M	27.609866	40.411903	10.374970	490.412335	22.82184	6.680461
100M	42.959680	88.981196	16.944667	1667.97833	25.98346	7.922546
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.48 Execution time statistics in milliseconds for data delete operations on 2-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	5118	1110	3091	8005	5081	235
10M	3910	1550	653	7470	4065	256
100M	1826	351	647	2766	1869	31
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.49 Number of delete operations per second on 2-node MySQL database

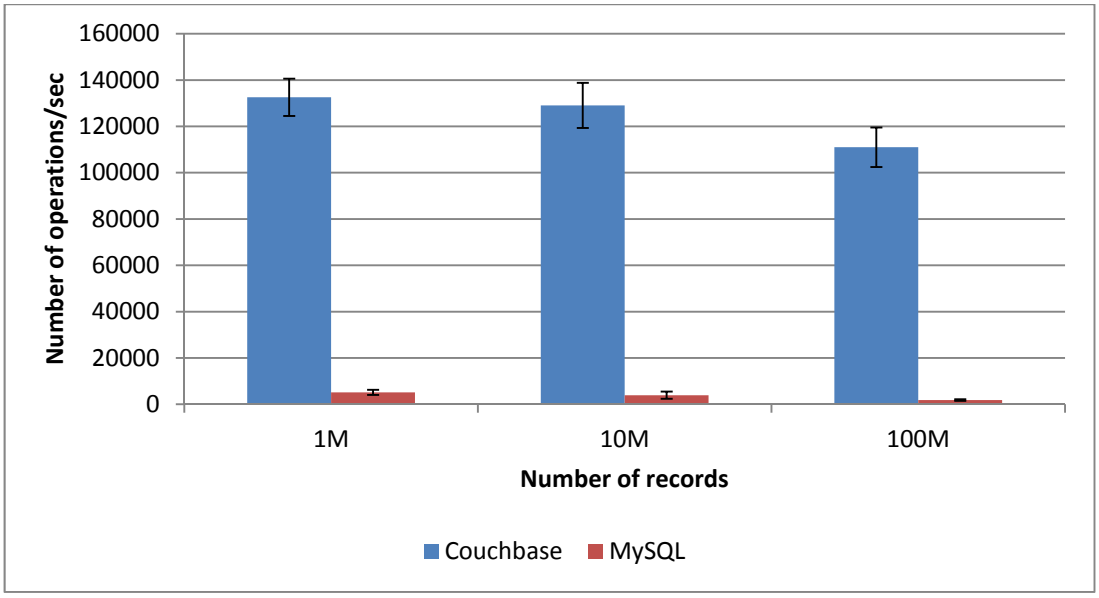


Figure 5.13. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 2-node configuration.

Similar to single-node delete test, Couchbase has significantly outperformed MySQL in the delete operation test run on a 2-node configuration as well, as shown in Figure 5.13. The number of delete operations that MySQL database could execute also decreased as the number of the total records stored increased.

5.4.3.3 Delete operations on 4 nodes

Tests for delete operations have been executed on Couchbase and MySQL in a 4-node configuration and both databases have demonstrated an increased performance compared to their respective 2-node configurations. However, the gap between

Couchbase and MySQL was significant. Couchbase was able to handle about 190,000 delete operations per second, while the average for MySQL was around 6,000.

The average execution times and the number of records deleted for Couchbase in a 4-node configuration are shown in Table 5.50 and Table 5.51.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.165020	0.004263	0.158787	0.170642	0.165178	0.003564
10M	0.258348	0.025178	0.229078	0.312813	0.255580	0.015215
100M	0.208812	0.010120	0.183337	0.219864	0.211394	0.006430
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.50 Execution time statistics in milliseconds for delete operations on 4-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	230106	10959	219588	252439	226823	9162
10M	170398	11197	154325	193210	168531	6766
100M	179125	9985	163825	204118	177329	6344
500M	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.51 Number of delete operations per second on 4-node Couchbase cluster

Couchbase was able to perform executions of delete record operation well under the millisecond level and performing around 250,000 executions per second at times while, conversely, MySQL was only able to achieve a maximum of 23,500 executions for the same delete records operation test scenario (See Table 5.53). The average execution time and the average number of delete operations per second for MySQL is shown in Table 5.52 and Table 5.53.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	20.780450	96.923433	6.782082	1105.81183	9.586021	15.429978
10M	22.811614	87.611905	7.945793	1039.52318	13.61861	10.536997
100M	33.875315	110.50461	12.918372	1002.45094	18.04231	21.596977
500M	32.807602	124.53444	9.446510	910.307648	14.99437	24.460885

Table 5.52 Execution time statistics in milliseconds for data delete operations on 4-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	12028	3510	5710	23571	11216	559
10M	6459	2731	2757	17069	5622	328
100M	2664	477	1578	4087	2678	93
500M	1917	473	547	3244	1859	93

Table 5.53 Number of delete operations per second on 4-node MySQL database

The average number of records deleted per second for Couchbase and MySQL databases in a 4-node configuration is presented in Figure 5.14.

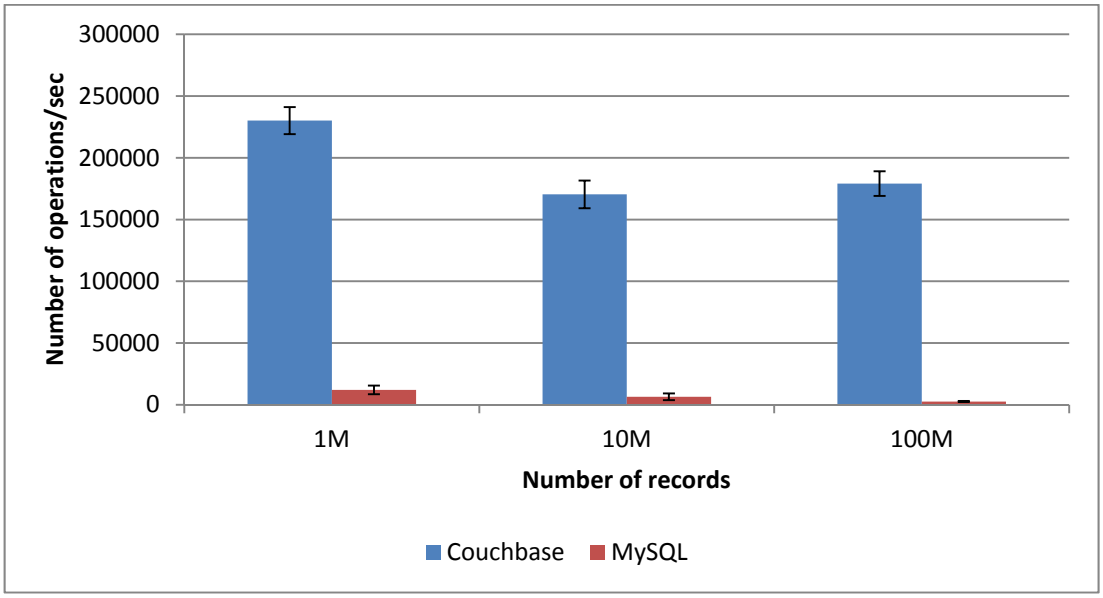


Figure 5.14. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 4-node configuration.

Although there is a significant increase in the number of records deleted per second, execution times for delete operations were at around 1000 millisecond level at times for MySQL, as shown in Table 5.52. Furthermore, the average number of records deleted per second significantly decreased as the total number of records stored in MySQL increased from 1 Million, 10 Million through 100 Million in the delete record test scenario (see Table 5.53).

5.4.3.4 Delete operations on 8 nodes

The Delete records operations tests were executed on an 8-node configuration for both MySQL and Couchbase database. The average execution times for a delete record operation and the number of records deleted for Couchbase in an 8-node configuration is shown in Table 5.54 and Table 5.55.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.154722	0.005239	0.144562	0.167790	0.154121	0.002031
10M	0.153492	0.005035	0.144792	0.164437	0.153596	0.001992
100M	0.153437	0.006596	0.143241	0.167675	0.152423	0.002463
500M	0.154561	0.005164	0.145423	0.167968	0.152939	0.001964

Table 5.54 Execution time statistics in milliseconds for delete operations on 8-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%
1M	306003	34014	180033	348455	313337	13189
10M	275208	25591	219976	333088	274434	10123
100M	289799	23689	236241	326736	286070	8846
500M	309081	29751	250820	382439	311216	11317

Table 5.55 Number of delete operations per second on 8-node Couchbase cluster

Couchbase was able to delete around 300,000 records per second and the average response time was consistent at 0.15 milliseconds, as shown in Tables 5.54 and 5.55.

In contrast, response times for MySQL were quite variable, ranging between 6 and 1119 milliseconds (see Table 5.56). The average execution time and the average number of delete operations per second for MySQL is shown in Table 5.56 and Table 5.57.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	20.491357	14.894490	7.661936	150.323271	17.65293	2.647627
10M	20.999945	70.741330	8.602682	1110.51504	13.38604	8.572796
100M	19.931986	70.963426	7.694629	1113.40727	12.98694	7.547685
500M	24.597611	104.83407	6.802336	1119.83917	11.95452	13.866245

Table 5.56 Execution time statistics in milliseconds for data delete operations on 8-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	23722	4648	12640	35251	23868	826
10M	13663	3986	6153	23516	13396	483
100M	6182	2365	1581	15492	5515	252
500M	3550	1162	623	7736	3324	154

Table 5.57 Number of delete operations per second on 8-node MySQL database

MySQL was able to delete 23,700 records per second for 1M records stored, however, this number significantly drops to 3,500 for 500M records. The performance increase in MySQL was around 100% by the means of number of records deleted per second compared to the 4-node configuration.

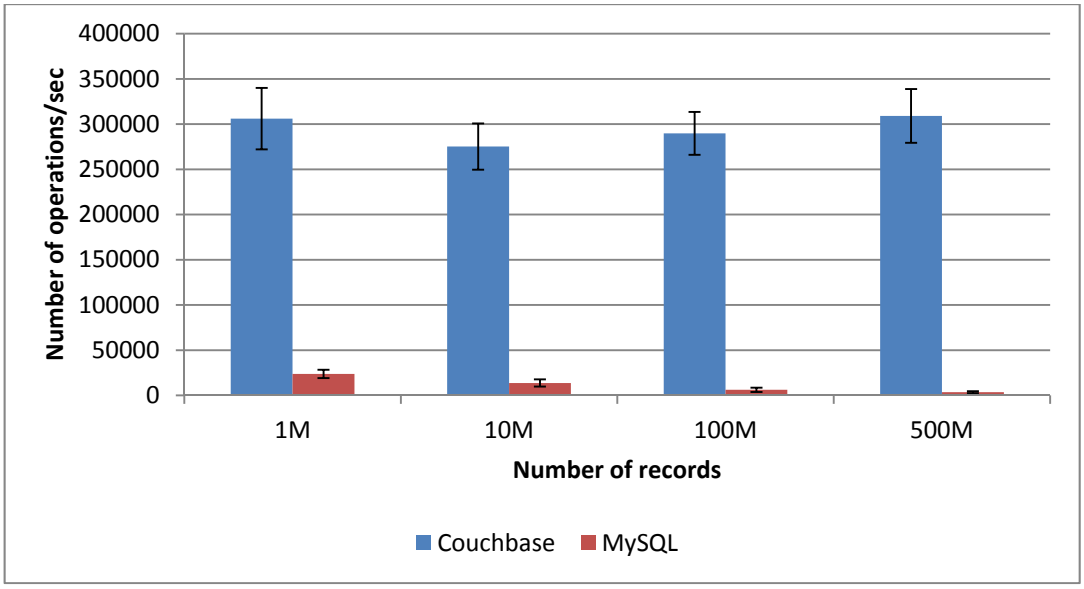


Figure 5.15. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 8-node configuration.

Although Couchbase has performed around 50% better in this 8-node configuration compared to the 4-node configuration, Couchbase has significantly outperformed MySQL in delete records operations, as shown in Figure 5.15.

5.4.3.5 Delete operations on 16 nodes

Final performance tests for delete operations were executed on a 16-node configuration. Response times for Couchbase were similar to the previous configurations and even better in some cases. The average number of records deleted per second has increased 67% on average with 81% improvement for 500M records stored. The average execution times and the number of records deleted for Couchbase in a 16-node configuration is shown in Table 5.58 and Table 5.59.

	Execution time in milliseconds					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	0.323179	0.022473	0.271631	0.366165	0.321886	0.013580
10M	0.197337	0.010646	0.183903	0.216813	0.194736	0.006764
100M	0.217867	0.047175	0.171414	0.346600	0.201658	0.028508
500M	0.181790	0.003608	0.176479	0.189423	0.181767	0.002773

Table 5.58 Execution time statistics in milliseconds for data delete operations on 16-node Couchbase cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	462038	34723	407005	539753	460242	20983
10M	467789	41466	421811	561765	452283	26346
100M	479979	32315	456126	582952	469497	19528
500M	559991	24641	533711	617140	553382	18941

Table 5.59 Number of delete operations per second on 16-node Couchbase cluster

Couchbase was able to delete more than 550,000 records per second, which eventually resulted in the test being completed in less than 2 seconds (see Table 5.59). MySQL has also performed better with improvements above 100% for higher number of records stored. The average execution time and the average number of delete operations per second for MySQL are shown in Table 5.60 and Table 5.61.

	Execution time in milliseconds					
Number of recs	Mean	SD	Min	Max	Median	CI (95%)
1M	23.251341	54.000756	8.653422	566.343375	15.14417	9.599112
10M	25.940480	86.689914	7.397691	973.351270	16.48033	15.409899
100M	20.173944	75.880972	7.468597	1113.42958	12.92544	9.968792
500M	27.509758	124.20507	7.505406	1135.97802	9.998260	26.629623

Table 5.60 Execution time statistics in milliseconds for data delete operations on 16-node MySQL cluster

	Number of operations per second					
Number of records	Mean	SD	Min	Max	Median	CI (95%)
1M	38681	5718	26347	53431	38781	1017
10M	24444	4311	15562	37263	24549	766
100M	14126	3600	8466	29219	13300	473
500M	7310	2638	3450	15123	6435	566

Table 5.61 Number of delete operations per second on 16-node MySQL database

The average number of records deleted per second for Couchbase and MySQL databases in a 16-node configuration is shown comparatively in Figure 5.16.

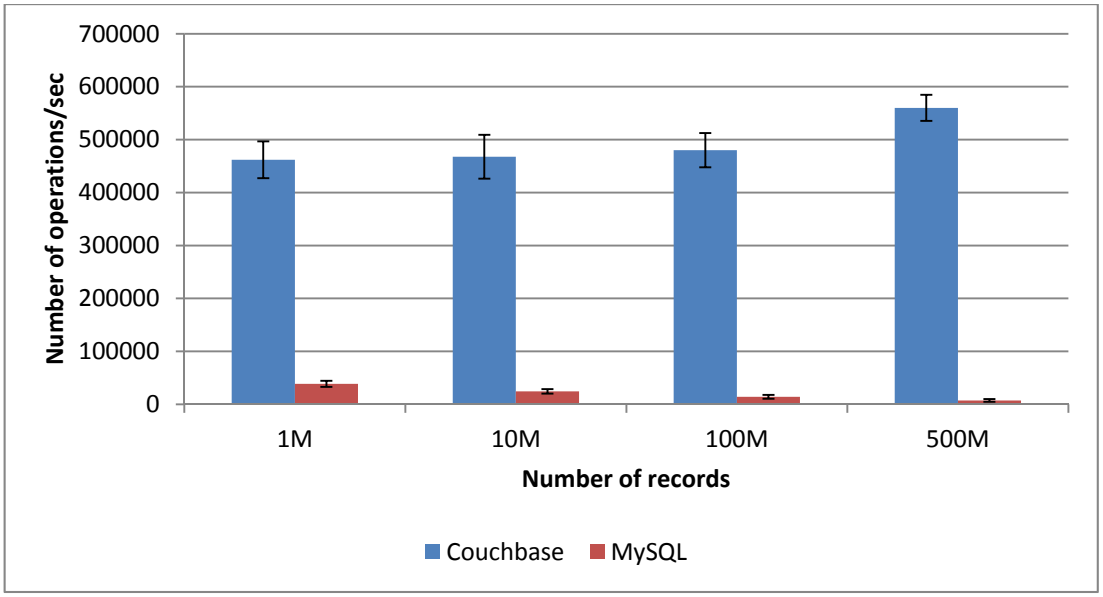


Figure 5.16. Average number of records deleted per second with standard deviations for Couchbase and MySQL in 16-node configuration.

MySQL was able to execute delete operations in about 25 milliseconds on average. However, for higher number of records, the execution time was more than one second at times—as presented in Table 5.60. The maximum number of delete operations executed per second across all tests was 53,431 for MySQL, while this number was 617,140 for Couchbase—as shown in Table 5.61 and 5.59 respectively.

5.4.4 Simulation of EHR sharing through retrieval of patient EHRs

This research focused on healthcare specific data while executing the performance tests of a NoSQL database comparative to a relational database in a distributed EHR system environment. In the previous tests reported in this chapter, single record operations such as insert, update and delete are executed using the generated healthcare data to measure the performance. In this section, results of simulation of a data retrieval operation of patient's EHRs that supports EHR sharing functionality for a NoSQL database comparative to a relational database are presented.

EHR sharing requires finding multiple EHRs for a single person (Bergmann et al. 2007; Huang et al. 2009; Narayan, Gagne & Safavi-Naini 2010). Therefore, instead of

finding records by record identifiers, records are queried by person identifiers. This operation requires finding the relevant EHRs using a person identifier, sourcing these EHRs and all required values from lookup tables where applicable to generate a response containing the required EHRs. This EHR sharing test is executed using a single query on a MySQL database using multiple joins, and is executed using a two-step operation on a Couchbase database that is to (1) find record identifiers by person identifier and (2) obtain relevant documents.

This EHR sharing test was executed on the cluster having the highest number of (16) nodes with the highest number of (500M) records stored to simulate realistic EHR data sharing on a large scale EHR system.

For MySQL, a single query returning all relevant rows based on the patient identifier is used. The query included the lookup tables to identify the values stored by keys, such as Indigenous Status, Sex and Admission Mode to enable generating a meaningful response. It is aimed at returning a response containing the values corresponding to the identifiers to make sure interoperability and human-readability is achieved. For Couchbase, as documents already contain the relevant values as objects, there was no other join or enrichment required to achieve this.

However, EHRs are stored as documents identified by EHR Identifiers in a NoSQL database (Couchbase). This led to the requirement of establishing an index to query EHR identifiers using person identifiers. The results for the EHR sharing simulation are presented in Table 5.62 and Table 5.63.

	Execution time in milliseconds					
Database	Mean	SD	Min	Max	Median	CI (95%)
MySQL	22.7549	101.0860	2.0435	528.4742	2.8469	39.9883
Couchbase	17.7056	31.0887	4.5703	197.9042	11.0846	10.0778

Table 5.62 Execution time statistics in milliseconds for EHR sharing simulation

	Number of operations per second					
Database	Mean	SD	Min	Max	Median	CI (95%)
MySQL	11429	2307	6298	16131	11598	913
Couchbase	14624	3344	8240	21403	14288	1084

Table 5.63 Number of operations per second for EHR sharing simulation

The average number of EHR sharing operations per second is significantly lower and the average execution time is also significantly higher for Couchbase compared to the previous insert, update and delete record tests. However, MySQL performed similar to other database operations of insert, update and delete in both of the measured parameters. Therefore the performance difference between Couchbase and MySQL was minimal in the EHR sharing test. This was primarily caused by the time-consuming operation of identifying the document keys of the EHRs for a particular person. After identifying the document keys, the rest of the process execution time was taken up with retrieving the required documents by these keys.

The results for Couchbase are consistent with the YCSB benchmark published on the Couchbase blog in July 2016 and the results of a study by Borkar et al. (2016), in which a special hardware used. The YCSB benchmarking results in Borkar et al. (2016) study demonstrated that the number of operations per second is significantly lower for YCSB Workload E compared to YCSB Workload A. YCSB Workload E contains complex operations comparable to EHR sharing simulation and YCSB Workload A is a mixed load of 50/50 reads and writes (Borkar et al. 2016; Zhu 2016). Therefore, the results for EHR sharing simulation in this study is consistent with the previous findings of a similar YCSB testing (Borkar et al. 2016).

Despite being significantly higher than the insert, update and delete tests, the average response times for Couchbase database was still lower than the average response times for MySQL database in this EHR sharing test. Figure 5.17 demonstrates the comparative performance in terms of number of operations per seconds for Couchbase and MySQL databases.

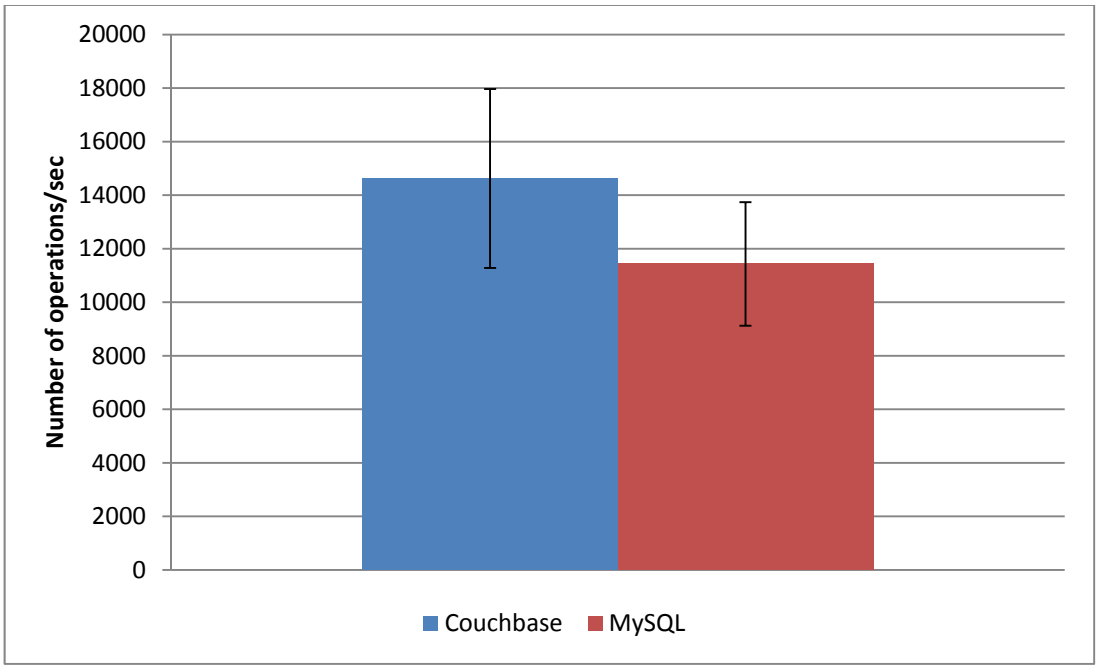


Figure 5.17. The average number of EHR sharing operations per second for Couchbase and MySQL.

5.4.5 Data Size

The size of the data stored in both MySQL and Couchbase 16 node cluster configurations has been measured for 1M, 10M, 100M and 500M records respectively. Data size for Couchbase has been measured after manually executing a compaction operation. However, it is worth noting that Couchbase requires at least 30% more space than the original data size to function properly. This is due to the design of the Couchbase data handling process that triggers a compaction for the data files at the configured fragmentation threshold which is 30% by default.

The size for the data files grow linearly by the number of the records for both databases, however, the number of nodes did not have any significant effect on file size. Therefore, data sizes for both databases are shown in Table 5.64 in GBs.

Number of records	Couchbase Data Size (GB)	MySQL Data Size (GB)
1M	1.9	0.88
10M	20	8
100M	190	90
500M	940	440

Table 5.64 Size of the data by the number of records stored for Couchbase and MySQL

It is observed from Table 5.64 that Couchbase needed more than double the amount of space required by MySQL to store the same amount of EHRs for a 16 node cluster configuration. Data sizes for both databases are also shown comparatively in Figure 5.18 for the range of different numbers of records used in this EHR sharing simulation.

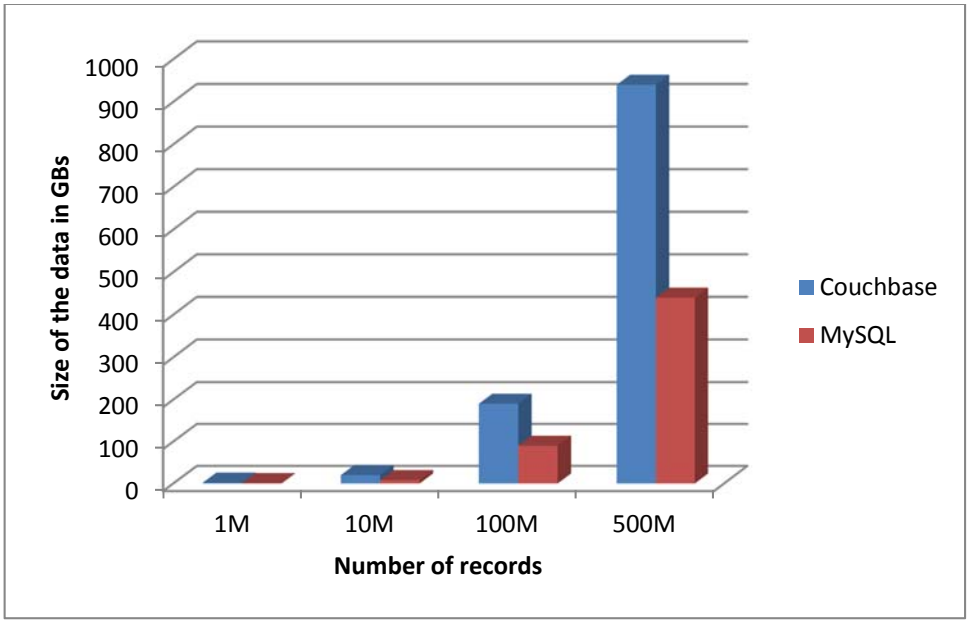


Figure 5.18. Size of the data by the number of records stored for Couchbase and MySQL.

5.4.6 Query Capabilities

For the last test in the evaluation of a NoSQL document database comparative to a relational database in a distributed EHR system environment in this research, a complex query was executed on both the Couchbase database and the MySQL

database when the number of stored records was 500M and the number of nodes was 16.

The query is expected to return average date of birth per principal diagnosis. The result of this query was achieved using a 'group by' query in MySQL. On the other hand, a map function that selects the relevant data with a built-in reduce function called "_stats" is used for the same purpose in the Couchbase database. Views in Couchbase database are similar to materialised views in relational databases. A view is essentially a distributed index created as a result of a Map Reduce operation that can then be queried (Borkar et al. 2016). Materialised views are not supported in MySQL database and therefore query tests in this research do not include materialised view approach in relational databases. However this test is to evaluate complex query capabilities and it is valid to compare the Couchbase initial view creation to the MySQL query returning the same results.

It took 4375 seconds for Couchbase database to create the initial view, however, as this is a one-time operation any other subsequent updates on the data are almost instantly reflected in the views. On the other hand, this query can be executed on the MySQL database without any need for particular indexes for the selected fields and the MySQL database was able to return the query result in 5149 seconds. An index was created on the MySQL database for principal diagnosis which also contains record identifier. Index creation took 601 seconds and after the creation of index, original query took 3884 seconds to execute.

The Couchbase database and MySQL database demonstrated similar execution times for this complex query. However, after the initial view creation, querying the view on the Couchbase database takes less than 100 milliseconds on average. This duration is similar to the average execution time for the EHR sharing test. Therefore, it is observed that the Couchbase database is better at performing pre-defined queries after the initial view creation has been conducted, while MySQL performs better on ad-hoc queries when there are relevant indexes in place.

5.5 Conclusion

This chapter presented the key results of the evaluation phase of a simulation of a large scale EHR system in this study. The selection of a specific NoSQL database and a specific relational database, and a cloud environment to conduct the simulations tests

to evaluate the performance of a NoSQL database comparative to a relational database was determined. Couchbase was selected as the NoSQL database and MySQL was selected as the relational database for the purpose of this research. Amazon Web Services (AWS) was used to setup the test cloud environment for the evaluation of the performance of a NoSQL database comparative to a relational database in a simulation of a large scale EHR system.

Test scenarios have been identified for relevant database operations and for a range of different configurations of number of nodes and numbers of records to simulate a large scale EHR system. Test scenarios simulate the real life EHR system functionalities. Insert tests have been conducted to simulate receiving EHR data from healthcare facilities and saving into databases. Update and delete tests simulate the relevant operations by healthcare facilities. Furthermore, one of the main aspects of EHR systems, EHR sharing, is simulated by querying records using person identifiers. In addition, data sizes for databases are presented comparatively and a sample complex query has been executed on both databases to understand and compare complex query capabilities for both databases.

All test results are summarised in tables and provide descriptive statistics for the average number of operations per seconds and the average response time for both types of databases.

Test results in this chapter demonstrated that the Couchbase database outperformed MySQL database in most of the performance tests. Furthermore, Couchbase database was able to demonstrate near linear scalability which was better than the scalability capabilities demonstrated by the MySQL database. Couchbase has also performed slightly better than MySQL database in data retrieval operation for EHR sharing simulation. However, test results also suggest that the MySQL database has better analysis performance for ad-hoc queries and stores data more efficiently using less storage space compared to the Couchbase database. In the following chapter, chapter 6, the results of this chapter and chapter 4 are discussed in detail in relation to each of the research questions investigated in this study and the existing literature. Then this study as a whole is evaluated and discussed using design science research guidelines.

Chapter 6 – Discussion and Evaluation of this Research

6.1 Introduction

In this chapter, the results from the EHR system simulation are discussed in relation to each of the seven research questions investigated in this study. Then the research activities conducted in this study are evaluated and discussed using seven guidelines for evaluating design science research. For research question one, the key findings regarding database selection and development of the data models for each selected database are then discussed. For research question 2, the key findings regarding the development of the Random Healthcare Data Generator artefact that was used to populate each database and its underlying data structures in order for the EHR System Prototype simulation tests to be conducted are discussed. Next, for research question 3, the key findings regarding the development of the EHR System Prototype simulation tests conducted for the selected NoSQL database (Couchbase) and the selected relational database (MySQL database) are discussed. For research question 4, the key findings regarding the performance evaluation of database operations (insert, update, delete records) for the selected NoSQL database and selected relational database are compared and discussed. For research question 5, the key findings regarding the performance evaluation of scalability for the selected NoSQL database and selected relational database are compared and discussed. For research question 6, the performance evaluation of EHR sharing for the selected NoSQL database and selected relational database are compared and discussed. For research question 7, the performance evaluation of complex querying for the selected NoSQL database and selected relational database are compared and discussed. Finally, to conclude this chapter, the research activities conducted to complete this design science study are evaluated and discussed using seven design science research guidelines (Hevner et al 2004). The structure of this chapter is shown in Figure 6.1.

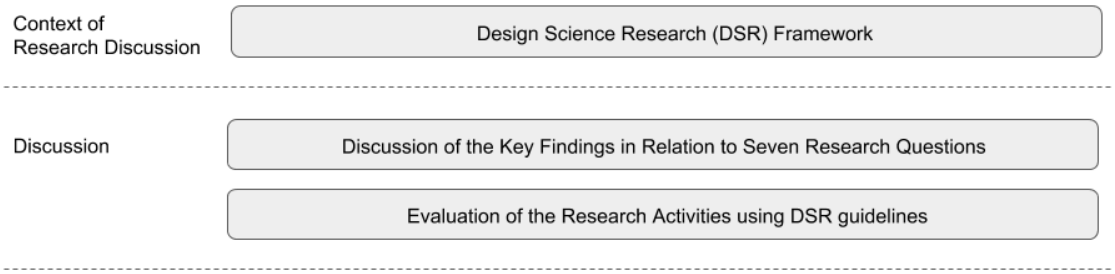


Figure 6.1. Structure of Chapter 6

6.2 Discussion of Key Findings

The key results from chapters 4 and 5 are discussed in relation to each of the research questions investigated in this study and the existing literature in the following subsections.

6.2.1 Development of Relational and NoSQL Data Models - Research Question 1

RQ1: How can a NoSQL document data model and a relational data model be developed for an EHR system that are in line with documents published by healthcare authorities in Australia?

In this research, relational and NoSQL (document) data models which provided the data structures for storing EHRs in a NoSQL document database and a relational database were developed based on the activities undertaken and described in Chapter 4 – Development of IT artefacts. Data sets and data elements are based on the National Health Data Dictionary published by the Australian Institute of Health and Welfare (AIHW 2015). The data model for the relational MySQL database was developed as multiple tables based on normalisation theory (Codd 1970) (see Chapter 4 for relational database data model). The data model for the NoSQL Couchbase database was developed based on a document model as aggregate oriented, nested document model (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Gudivada, Rao & Raghavan 2016) as discussed in Chapter 2 Sections 2.3.1 and 2.4.3.

The document data model contained all information as nested objects in JSON format along with all required code and values. However, the values for codes that are a part of a medical coding system, such as ICD-10-AM for principal diagnosis (AIHW 2016), are stored in separate lookup tables for relational databases. The different design of each data model caused the overall data size to be smaller in relational databases compared to NoSQL databases—an aspect which is discussed in more detail later in this chapter.

6.2.2 Random Healthcare Data Generator – Research Question 2

RQ2: How can a random healthcare data generator be developed that will generate EHRs that are representative of the characteristics of Australian healthcare data based on statistics available in the public domain? The first software artefact developed for this research was the Random Healthcare Data Generator. This artefact generated the test data used in this research. One of the key contributions of this research, this artefact, is able to generate synthetic healthcare data which eliminates the possible ethical issues of obtaining access to patient EHRs.

The Australian Institute of Health and Welfare (AIHW) publishes the healthcare statistics for Australian health system, which are the main inputs for the Random Healthcare Data Generator artefact (AIHW 2016). This is a fundamental requirement to generate synthetic data that will show similar data distribution characteristics to the healthcare statistics for the Australian health system.

The data generated as the output of this artefact has been compared with the source data that is publicly available from healthcare statistics provided by AIHW. The distribution characteristics of the generated data were shown to be similar to the source data (Australian healthcare statistics) as described in Chapter 4 section 4.5.2.

Therefore, it is observed that the data generation for a particular dataset is possible based on the relevant statistical publications. This is an approach that is applicable to domains where such statistics and data dictionaries are available.

EHRs are created and then inserted into each database. Thus, the data models established as a response to the previous research question provide the data structures for storing EHRs in each database (NoSQL database, Relational database).

6.2.3 EHR System Prototype – Research Question 3

RQ3: How can a prototype EHR system be developed that will facilitate database operations and measure performance and scalability for NoSQL document databases and relational databases?

The second software artefact, a prototype EHR system, was developed to act as an intermediary between the Random Healthcare Data Generator IT artefact and the selected NoSQL document database (Couchbase) and the selected relational database (MySQL). This IT artefact was used to evaluate the comparative performance of two selected databases (NoSQL document database; MySQL database) through execution of database operations and collection of relevant metrics about the executions, namely execution time and number of executions per second.

The data generated by the Random Healthcare Data Generator is processed by this prototype EHR system and inserted into NoSQL and relational databases for a number of different node configurations. This simulates a national EHR system that receives data from a healthcare service provider and stores that data in a database. This artefact executed insert record, update record and delete record operations and collected relevant metrics on the performance of each database operation.

Another key function of the prototype EHR system is to enable simulation of the EHR sharing operation. This operation requires identification of all EHR documents of a particular person in the NoSQL database or joining all relevant tables to find out all relevant data in multiple tables in the relational database, and returning the results to the client. This functionality simulates the scenario of a healthcare service provider—either a hospital or emergency services—requesting EHRs for a particular patient. This artefact has also collected the same database performance metrics for EHR sharing simulation tests.

Furthermore, this EHR system prototype has also executed complex query tests on both NoSQL and relational databases and recorded the performance metrics for the complex query simulation tests.

6.2.4 Performance evaluation for basic database operations (insert, update, delete) for NoSQL and relational databases – Research Question 4

RQ4: How do NoSQL document databases perform compared to relational databases in executing basic database operations such as insert, delete and update on electronic health records?

Results of the performance evaluation of a NoSQL database comparative to a relational database (presented in chapter 5) is based on two commonly used database performance metrics: (1) average number of database operations per second; and (2) average execution time (milliseconds) of a database operation (Barata, Bernardino & Furtado 2014; Meinel et al. 2015; Thanopoulou, Carreira & Galhardas 2012). These two metrics are calculated to measure the performance of insert, update and delete records operations using different number of records stored and node count combinations. A summary table for the average number of operations per second by node count for each operation, independent of number of total records stored, is presented in Table 6.1.

Number of Nodes	INSERT		UPDATE		DELETE	
	Couchbase	MySQL	Couchbase	MySQL	Couchbase	MySQL
1 Node	20,085	4,659	20,292	3,923	72,411	2,239
2 Nodes	35,350	6,099	38,427	8,254	124,200	3,618
4 Nodes	77,956	5,977	93,993	18,247	193,210	5,767
8 Nodes	143,015	10,438	171,546	33,960	295,023	11,779
16 Nodes	323,082	15,980	343,688	55,943	492,449	21,140
Average	119,898	8,631	133,589	24,065	235,459	8,909

Table 6.1. Average number of operations per second by the number of nodes and operation type for Couchbase database and MySQL database.

The Couchbase database outperformed the MySQL database in all node configurations for insert, update and delete operations. While the best performance for the Couchbase database is observed in delete operations, the MySQL database performed better in update operations compared to other types of operations. The Couchbase database was able to execute 5 to 26 times more operations per second compared to the MySQL database. The results presented in Table 6.1 also indicate that the Couchbase database has an ability to scale out at a rate exponential to relational databases, which has significant performance implications that are discussed in the next section.

The second measure of database performance was average execution time in milliseconds for each insert, update and delete record operation. A summary table for this measure is presented in Table 6.2.

Number of nodes	INSERT		UPDATE		DELETE	
	Couchbase	MySQL	Couchbase	MySQL	Couchbase	MySQL
1 Node	0.15	60.14	0.15	37.68	0.21	35.06
2 Nodes	0.19	62.87	0.15	33.54	0.25	35.19
4 Nodes	0.18	82.42	0.16	29.98	0.21	27.57
8 Nodes	0.31	70.91	0.12	14.55	0.15	21.51
16 Nodes	0.24	104.37	0.17	14.91	0.23	24.22
Average	0.22	76.14	0.15	26.13	0.21	28.71

Table 6.2. Average execution times by the number of nodes and operation type for Couchbase database and MySQL database.

Couchbase database was able to complete the execution of insert, update and delete operations in less than a millisecond on average. Conversely, MySQL was able to execute insert operations in 76 milliseconds, update operations in 26 milliseconds and delete operations in 29 milliseconds on average. Furthermore, in some tests, MySQL completed executions of some operations in around 1,000 milliseconds. In contrast, the maximum duration for executions on Couchbase database was 4.4 milliseconds and it also demonstrated a predictable high performance during the tests. The difference in average execution times between Couchbase database and MySQL database was highly significant as Couchbase database was able to execute operations more than 300 times faster than the MySQL database for some tests.

Insert, update and delete operations are all single EHR operations. Couchbase database has performed significantly better than MySQL database in both average number of operations per second and average execution time measures in these operations. The results are consistent with previous studies where NoSQL databases demonstrate a better performance for different workloads in various setups (Biyikoglu 2016; Cooper et al. 2010; Freire et al. 2016; Li & Manoharan 2013). The results for the operations dealing with multiple EHRs such as EHR sharing simulation are discussed in the next sections.

In terms of the number of records stored while executing the operations on the databases, Couchbase and MySQL have demonstrated different results. Table 6.3

summarises the average number of operations per second for different numbers of stored records in both databases. The table shows the mean value of average number of operations per second for 8 and 16 nodes, for which storing 500M records were possible.

Number of records	INSERT		UPDATE		DELETE	
	Couchbase	MySQL	Couchbase	MySQL	Couchbase	MySQL
1M	213,738	14,618	247,970	52,556	384,021	31,202
10M	254,020	13,595	253,811	50,179	371,498	19,053
100M	224,638	12,698	260,272	40,281	384,889	10,154
500M	239,801	11,925	268,416	36,790	434,536	5,430
Average	233,049	13,209	257,617	44,951	393,736	16,460

Table 6.3. Average number of operations per second for the number of stored records and operation type for Couchbase database and MySQL database.

Couchbase database was able to handle the number of operations per second on or above the overall average for 1M, 10M, 100M and 500M stored records in all operation types. However, for MySQL database, the average numbers of operations per second for 100M and 500M stored records were lower than the overall average number of operations per second for MySQL database. For instance, the overall average number of insert operations for MySQL was 13,209, while it was 12,698 for 100M records test and 11,925 for 500M records test.

This demonstrates that the performance for MySQL database decreases by the number of stored records in all cases, while the total number of records stored does not have any significant impact on the Couchbase database. This is consistent with previous studies showing similar performance degradation on relational databases over time—as mentioned in the previous chapter (Hadjigeorgiou 2013; Schmidt 2001; Souley & Mohammed 2013).

In summary, Couchbase database demonstrated a predictable and significantly higher performance than MySQL database in operations dealing with a single EHR. Moreover, Couchbase database also preserved the high performance in a higher number of records in contrast to MySQL which slowed down considerably as the number of records stored increased from 1M, 10M, 100 M and 500 M.

6.2.5 Scalability capabilities of NoSQL document database and relational database – Research Question 5

RQ5: How do NoSQL document databases scale compared to relational databases in electronic health record systems?

In addition to performance evaluation of basic database operations for the selected NoSQL document database and relational databases, scalability capabilities of each database are also compared as an important part of this research.

The tests were executed on different numbers of nodes to evaluate scalability capabilities of both databases. The Couchbase database had identical nodes, however, MySQL database had a specified number of data nodes in addition to SQL and API nodes. Therefore, for instance, when Couchbase database had 4 nodes, MySQL database had 5 nodes, 4 data nodes and one node running API and SQL node which handles the coordination and execution of T-SQL statements with data nodes. Thus, MySQL requires more hardware than Couchbase in order to run on the same number of data nodes.

Table 6.1 in section 6.2.4 summarises the average number of operations per second for various number of nodes which evaluates the scalability capabilities of Couchbase and MySQL databases for insert, update and delete operations. Both the Couchbase database and MySQL database demonstrated scalability capabilities, which are presented as a percentage change in the number of average operations that can be executed for each increase in the node count in Table 6.4.

Change in node count	**INSERT**		**UPDATE**		**DELETE**	
	Couchbase	MySQL	Couchbase	MySQL	Couchbase	MySQL
1 -> 2	75.99%	30.90%	89.37%	110.41%	71.52%	61.57%
2 -> 4	120.53%	-2.01%	144.60%	121.08%	55.56%	59.40%
4 -> 8	83.45%	74.66%	82.50%	86.11%	52.70%	104.25%
8 -> 16	125.91%	53.08%	100.35%	64.73%	66.92%	79.47%
Average	101%	39%	104%	96%	62%	76%

Table 6.4. Percentage change in average number operations per second per change in node count by operation type for Couchbase database and MySQL database.

Couchbase demonstrated a linear scalability for insert and update operations, as the average number of operations per second is increased by around 103% on average

when the node count is doubled. The improvement in performance was 62% on average for delete operations, however, the number of delete operations was significantly higher than the insert and update operations on each configuration.

The average numbers of operations for all node counts per operation type for the Couchbase database are plotted in Figure 6.2.

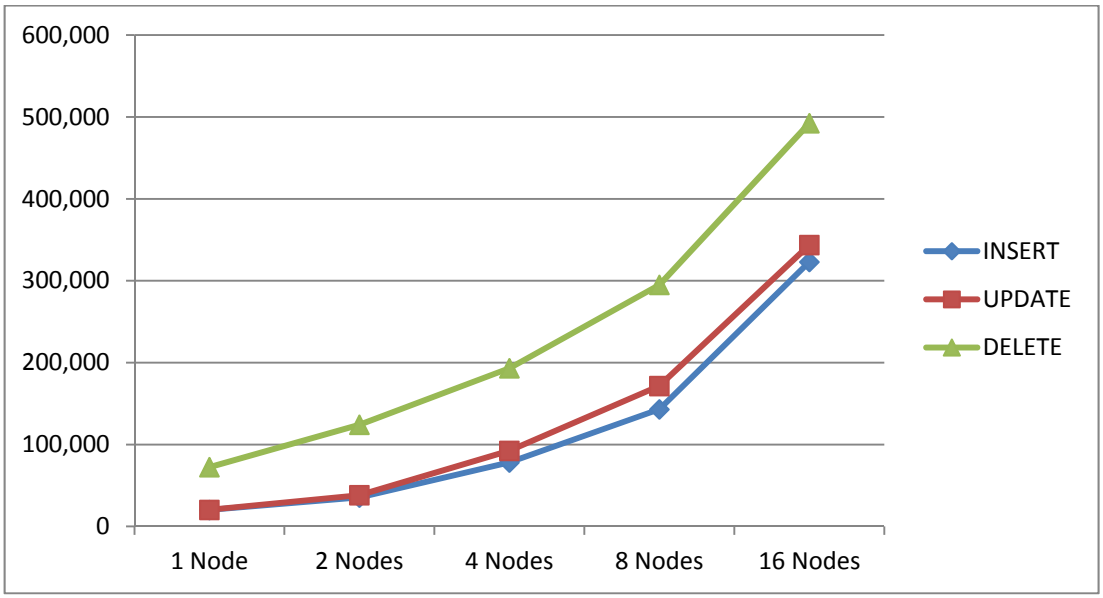


Figure 6.2. Average number of operations per second per node count for Couchbase database.

On the other hand, MySQL also demonstrated some scalability, but the increase in performance was 39% for insert operations on average when the node count is doubled. The increase in performance was 96% for update operations which required change in some tables, while the other operations require changes in all relevant tables as the values in a single EHR are stored across multiple tables after normalisation process is applied to the relational database. The improvement in delete operations was better than the improvement for insert operations for the relational database. However, improvements as the number of nodes was increased to scale up the relational database operations were not as good as the improvement seen for the Couchbase database. The average numbers of operations per second for all node counts per operation type for the MySQL database are plotted in Figure 6.3.

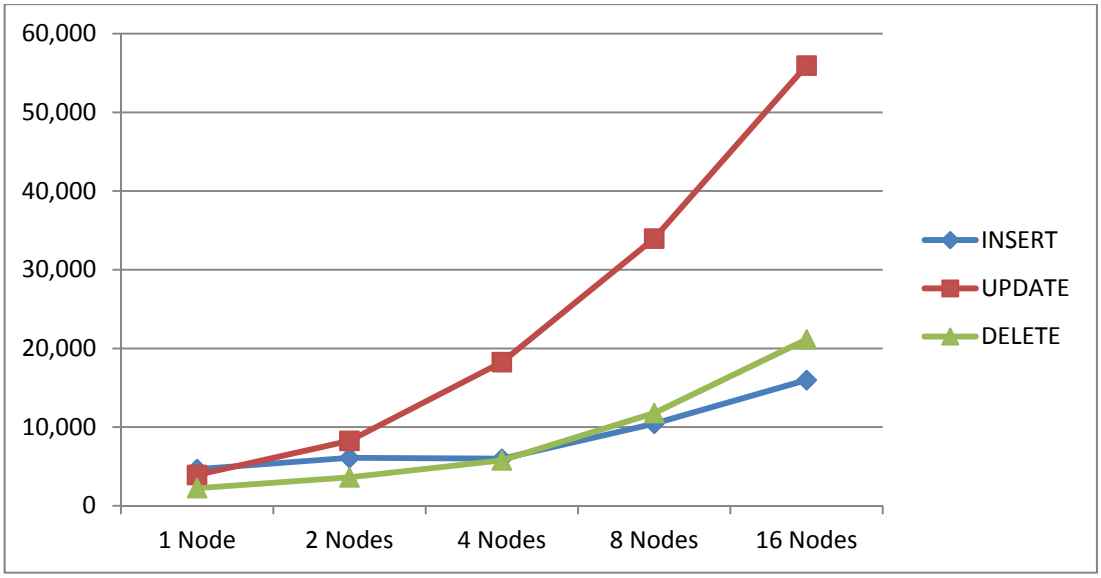


Figure 6.3. Average number of operations per second per node count for MySQL database.

The average improvements in performance as the Couchbase database and the MySQL database are scaled up across an increasing number of nodes is presented in Table 6.4 and in Figure 6.2 and Figure 6.3. The trends in Table 6.4 and Figures 6.2 and Figure 6.3 clearly demonstrate that Couchbase database has a near linear scalability for insert and update operations, while MySQL can scale up in update operations only. The improvement was 89% on average for Couchbase database when the number of nodes doubled; and 70% for MySQL. Therefore, Couchbase has demonstrated a better overall scalability on a much larger number of database operations executed per second.

Furthermore, while Couchbase has no limitation in maximum number of nodes mentioned in its documentation, maximum number of vbuckets—which are the storage files and therefore building blocks for data storage for Couchbase—is limited to 1024 (nodes). In contrast, the MySQL Cluster with NDB engine can only scale up to 255 nodes, including data and SQL nodes. As a result, Couchbase database is capable of scaling to a much higher number of nodes than the MySQL database.

6.2.6 EHR Sharing Simulation – Research Question 6

RQ6: How do NoSQL document databases perform compared to relational databases in supporting electronic health record sharing through patient record retrieval in a distributed EHR system?

Despite the significant performance difference between the Couchbase database and MySQL database for insert, update and delete operations, both databases demonstrated similar performance for the EHR sharing simulation.

The data retrieval for EHR sharing simulation requires more operations on databases compared to insert, update and delete operations. A single person can have multiple EHRs, and EHR sharing requires access to this person's previous EHRs. Therefore, in Couchbase tests, it is necessary to identify the EHR document keys for a particular person before fetching these EHRs. This leads to a two-step operation involving a query by a person identifier rather than the document key. Furthermore, in order to include all the necessary information, such as the values (meanings) of the codes to make it inter-operable and human readable, the result set is generated by joining multiple tables in MySQL database.

A two-step operation is a requirement for Couchbase database and joining lookup tables to fetch the full result set is a requirement for MySQL database. These requirements cause higher execution times resulting in a lower average number of operations per second for both databases. Couchbase database had an average execution time of 17.8 milliseconds, while MySQL database was able to execute the simulation queries in 22.7 milliseconds on average. The average number of operations for a Couchbase database was 14,624 and 11,429 for a MySQL database.

As a result, Couchbase database was able to handle 28% more executions and responded 29% faster than MySQL database in this EHR sharing simulation.

6.2.7 Complex Query – Research Question 7

RQ7: How do NoSQL document databases perform compared to relational databases in executing complex queries on electronic health records?

In order to assess the performance of NoSQL databases comparative to relational databases when the test scenario involves a complex ad-hoc query, a sample query has

been established to simulate the high-level statistic that shows the average date of birth for principal ICD10 –medical diagnosis- codes. The query is established as a T-SQL statement for MySQL database and a view for Couchbase database.

The tests have demonstrated that initial view creation for Couchbase took longer than MySQL query execution duration. However, after the creation of the view, Couchbase database was able to respond to subsequent queries within the same view in around 100 milliseconds. Therefore, it is concluded that if the complex query is ad-hoc and query conditions are changing, the MySQL database can perform better than the Couchbase database. However, if the query is pre-defined, the Couchbase database is able to respond to queries in a relatively small execution time after the initial view creation has occurred.

6.2.8 Data Size

In addition to the main research questions, an additional aspect, the data size of each database (NoSQL versus relational), is compared within the scope of this research. The total size for both the Couchbase database and the MySQL database is for 1M, 10M, 100M and 500M EHRs respectively for the different test scenarios. It is observed that the total size has grown linearly by the increasing number of records for both databases in the simulated EHR system environment.

However, Couchbase database needed more disk space to store the same number of EHRs compared to MySQL database. The main reason for this difference is the data structure underpinning the databases. Couchbase database has stored the EHRs as JSON documents including all values resulting in a readable and understandable, complete document. On the other hand MySQL database has stored EHRs using a normalised approach in multiple tables.

For instance, Person section of the EHR documents stored in a JSON object format takes 553 bytes (without spaces) as shown below:

```
"Person": {
        "Person identifier": "123456789",
        "Area of usual residence": {
                "METeOR identifier": "469909",
                "code": "31701144631446",
                "value": "Darling Heights"
        },
        "Country of birth": {
                "METeOR identifier": "459973",
                "code": "5101",
                "value": "Myanmar"
        },
        "Date of birth": "01012000",
        "Indigenous status": {
                "METeOR identifier": "291036",
                "code": "4",
                "value": "Neither Aboriginal nor Torres Strait Islander origin"
        },
        "Sex": {
                "METeOR identifier": "287316",
                "code": "1",
                "value": "Male"
        },
        "Medicare Eligibility status": {
                "METeOR identifier": "481841",
                "code": "1",
                "value": "Eligible"
        },
        "Address": "Address Information Sample",
},
```


The same data stored in a MySQL database in a Person table is shown below (column headers for information only, not are included in data size):

Person Identifier	Area Of Usual Residence	Country Of Birth	Date Of Birth
123456789	31701144631446	5101	2000-01-01
Indigenous Status	**Sex**	**Medicare Eligibility Status**	**Address**
4	1	1	Address Information Sample

Based on the table statistics available in MySQL, one row in the Person table requires around 190 bytes of disk storage on average. All sections in EHRs have similar differences between MySQL database storage and Couchbase database storage.

Therefore, in summary, Couchbase database requires around 100% more disk size compared to MySQL database to store the same number of EHRs.

6.3 Evaluation of this Research using Design Science Guidelines

Design Science research is defined as a 'problem solving paradigm' As discussed in Chapter 3, this research fits in the Design Science research paradigm in terms of its main outcome being a solution to a particular problem for which four IT artefacts are designed and implemented to enable the evaluation of the solution (Hevner et al. 2004; March & Smith 1995). In this section, research activities and contributions of this PhD Thesis are presented and discussed in the context of seven Design Science guidelines proposed in a seminal MISQ paper on Design Science (Hevner et al. 2004).

6.3.1 Design of IT Artefacts in this Study

The first guideline is that Design Science research needs to produce an artefact. This artefact can be a construct, a model, a method, or an instantiation (Hevner et al. 2004, p. 347).

In the course of this research four artefacts are developed as instantiations. A Random Healthcare Data Generator overcame the ethical issues and operational issues related to accessing sufficient healthcare data for the purposes of this study. This allowed a simulation of database operations such as insert, update and delete records and scalability and complex querying to be evaluated across multiple nodes (1 to 16) and large numbers of EHRs (1 Million to 500 Million). The second artefact Prototype EHR System managed the simulation of these database operations for a NoSQL document database (Couchbase) and a relational database (MySQL) in a large scale EHR system, including capturing the performance metrics of these database operations for each database. These artefacts are built on two data models developed in this study with data structures designed for storing EHRs in each database (Couchbase, MySQL). These data models were defined in the context of the Australian healthcare domain. These artefacts can be applied to similar research areas requiring performance testing that need synthetic healthcare data or an EHR sharing environment.

6.3.2 Problem Relevance of this Study

The second guideline is that "the main objective of design-science research is to develop technology-based solutions for important and relevant business problems" (Hevner et al. 2004, p. 347). An extensive review of the literature demonstrated that the modern day requirements for storing healthcare data has changed significantly from the requirements of previous decades (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Kruse et al. 2016). The size and heterogeneity of healthcare data has changed significantly over time and traditional relational databases cause a bottleneck in healthcare information systems (Freire et al. 2016; Jin, Deyu & Xianrong 2011; Lee, Tang & Choi 2013; Raghupathi & Raghupathi 2014; Schmitt & Majchrzak 2012). There is limited research in the area and most of the previous studies lack enough scope to identify and solve the problem of healthcare data management and storage. This research evaluated the performance of a NoSQL document database (Couchbase) solution comparative to a relational database (MySQL) using synthetic healthcare data in a simulation of large scale EHR system. The research problem is identified and extensively discussed in Chapter 2. In this research the proposed solution to this research problem is developed and evaluated using IT artefacts which constitute a 'technology-based solution' that is relevant and can inform real world practice.

6.3.3 Design Evaluation of IT Artefacts in this Study

The third guideline is that "the utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods" (Hevner et al. 2004, p. 347). The performance of a NoSQL database in healthcare data management is evaluated using the artefacts developed in this research to enable a simulation of a large scale EHR system. A simulation is considered a viable way to evaluate the utility, quality and efficacy of a design artefact (Hevner et al. 2004; Gill & Hevner 2013). The evaluation is based on generating healthcare data and measuring and evaluating the performance of database operations such as insert, update and delete records, scalability, EHR sharing and complex querying comparatively for NoSQL and relational databases. Therefore, a realistic prototype EHR system was developed to facilitate database operations for a NoSQL document database (Couchbase) and a relational database (MySQL) on EHRs. The simulated EHR data is provided by the Random Healthcare Data Generator at the scale of 1 Million, 10 Million, 100 Million and 500 Million records. These two artefacts were built on two data models developed for this study with data structures designed for a NoSQL document database and

MySQL relational database with data elements based on Australian Healthcare data and statistics. The database performance metrics used for evaluation of the performance of a NoSQL document database comparative to a MySQL relational database are well established metrics based on the execution times of database operations (per second) and the number of records handled per second for database operations. Data size used by each database for different configurations of number of nodes (1, 2, 4, 8, 16) and number of records (1M, 10M, 100M, 500M) was also evaluated. This approach was justified as being suitable to evaluate the performance of a NoSQL document database comparative to a relational database in large scale EHR Systems based on Australian healthcare data specifications.

The performance, scalability, EHR sharing and analysis capabilities for the selected NoSQL document database and relational database are evaluated comparatively using different configurations and test scenarios to demonstrate the benefits of using NoSQL databases in healthcare data management in large scale EHR systems.

6.3.4 Research Contributions

The fourth guideline is that "effective design-science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations, and/or design methodologies" (Hevner et al. 2004, p347). This research has made a number of important contributions to theory and practice. The main focus of this study was to evaluate NoSQL databases in the context of the Australian healthcare domain, which required developing IT artefacts and a simulation environment. Therefore, the principal contributions are the Random Healthcare Data Generator and NoSQL based EHR System prototype artefacts for the simulation of a large scale EHR system running on a cloud computing platform, AWS. These IT artefacts enabled the researchers to conduct the performance evaluation of database operations, scalability, EHR sharing and complex querying for a Couchbase database comparatively to a MySQL relational database. According to Hevner et al. (2004), the artefact(s) must provide a solution to unsolved problems. As discussed in Chapter 3, Gregor and Hevner (2013) identified a number of ways that design science research contributes to the Information Systems domain of knowledge (Gregor & Hevner 2013). This research contributes to knowledge by way of exaptation, adapting a new and emerging technology—NoSQL databases—which have been emerged in response to significant

data management problems in other fields or disciplines to another field and industry sector, healthcare. Therefore, this research applies existing knowledge about NoSQL document databases to the healthcare domain by developing artefacts to enable a performance evaluation of a NoSQL document database for data management in large scale EHR systems, which aligns with the research contribution of an exaptation (Gregor & Hevner 2013).

A comparison of the performance and scalability features, EHR sharing and analysis capabilities for a NoSQL document database and a relational database are evaluated using quantitative performance measures, along with descriptive statistics of the test results. Therefore, a solid comparative evaluation of the performance of database operations such as insert, update and delete, scalability, record sharing and complex querying for a NoSQL document database and a relational database are established for the healthcare domain. This also leads to an EHR system design based on a NoSQL document database to solve the healthcare data storage and handling problems identified and discussed in Chapter 1 and Chapter 2. Thus, this research has made significant and important contributions to both theory and practice.

6.3.5 Research Rigour

The fifth guideline is that "Design Science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact" (Hevner et al. 2004, p. 347). In this research, the empirical work has been carried out by developing and then applying and evaluating a solution to the research problem using the Design Science steps suggested and discussed in Chapter 3 Methodology (Gregor & Hevner 2013). Following the identification of the problem based on literature review, IT artefacts are constructed based on Australian healthcare dataset requirements and publicly available Australian healthcare statistics.

In the evaluation phase of the IT artefacts, quantitative data was collected using well-established performance metrics. The results of all performance tests are presented comparatively for a NoSQL document database and a relational database and discussed extensively. Sufficient details about the environment, test cases and other technical information are provided to allow other researchers to replicate the research. Best practice, based on previous literature, was followed when establishing the data

models and executing the simulation tests. The results of the evaluation are discussed in section 6.2 of this chapter.

6.3.6 Design as a Search Process in this Study

The sixth guideline is that "the search for an effective artefact requires utilizing available means to reach desired ends while satisfying laws in the problem environment" (Hevner et al. 2004, p. 347). The IT artefacts developed in this research are the results of a detailed search process. All datasets and data elements, as well as coding references used in the Random Healthcare Data Generator, are based on the National Health Data Dictionary (NHDD) published by the Australian Institute of Health and Welfare and publicly available national healthcare statistics (AIHW 2015, 2016). The required information is gathered from these sources to generate synthetic healthcare data needed for the database operations simulation tests in a large scale EHR system. After a detailed review of the available NoSQL databases and relational databases, Couchbase, a document database was chosen as the most suitable NoSQL database and MySQL was chosen as the most suitable relational database for the purposes of this study within the scope of a PhD study.

The prototype EHR system is based on the data models developed using the NHDD definitions and data structures designed to meet the unique requirements for each selected database (NoSQL document database, Couchbase; relational database, MySQL). Details of the research activities undertaken to design and evaluate these artefacts are described and justified in Chapter 3. The artefacts were designed and developed to maximise the alignment with the problem identified and seven research questions specified in Chapter 2, and the requirements of the healthcare domain. In this regard, the IT artefacts developed and evaluated in this research constitute valuable and promising solutions to the research problem.

6.3.7 Communication of this Research

The seventh guideline is that "Design Science research must be presented effectively both to technology-oriented and management-oriented audiences" (Hevner et al. 2004, p. 347). The researcher, although working fulltime as an IT practitioner with his own IT Consulting company, has published and presented one research paper from this study on the feasibility of NoSQL databases for data management of EHRs at the 25th

Australasian Conference on Information Systems in December 2014 in Auckland, New Zealand. This research paper has been uploaded to ResearchGate and has gained considerable interest from both academics and practitioners, with over 1500 reads as at September 2017. As a result of this paper, the researcher has been contacted by a number of practitioners seeking advice on how to implement a NoSQL database solution for EHR systems. This paper has also been cited by other academics, as evidenced by the citations statistics in Scopus (5 citations) and Google Scholar (9 citations).

This PhD thesis will be published in the public domain after a standard 12 months embargo. Consequently, this research will be freely available to other researchers who wish to build on the foundations established in this study.

The source code for two key artefacts developed in this research, Random Healthcare Data Generator and Prototype Electronic Health Records System will be made available in the public domain via GitHub for interested researchers and practitioners to download and adapt and use for their own purposes.

As an outcome of this research, a NoSQL based EHR system for large scale (particularly national) implementation is proposed and evaluated and found to be feasible. In many aspects, including cost reduction and high availability, this solution would be of benefit to the healthcare industry. A significant step has already been taken with this research, as demonstrated in the development of IT artefacts to enable a simulation of a large scale EHR system. Moreover, the evaluation of the database performance of a NoSQL document database versus a relational database provides a proof of concept that works in a realistic test environment. Therefore, key findings of this research would benefit both technology-oriented audiences and management-oriented audiences.

6.4 Conclusion

In this chapter, the key findings of this study are discussed in relation to each of the IT artefacts developed in this study and research questions 1, 2 and 3. The development of two primary IT artefacts for this research, a Random Healthcare Data Generator and a Prototype EHR system, are underpinned by the development of another two IT artefacts, two data models with data structures designed for storing EHRs in a NoSQL document database and a relational database. The Random Healthcare Data Generator enabled this research to generate EHRs representative of Australian Healthcare characteristics and statistics at scale of 1 Million, 10 Million, 100 Million and 500 Million records. These randomly generated healthcare data sets were used by the Prototype EHR system to facilitate a performance evaluation of a NoSQL document database versus a relational database in a simulation of a large scale EHR system. The key findings regarding the evaluation of the performance of a NoSQL document database comparative to a relational database in large scale EHR system simulation are discussed in relation to research questions 4, 5 6 and 7. The performance evaluation focused on database operations including insert, update and delete of EHRs, scalability, EHR sharing and complex querying. The test scenarios were configured by the Prototype EHR system for 1, 2, 4, 8 and 16 nodes and 1 Million, 10 Million, 100 Million and 500 Million records to simulate the performance evaluation in a large scale EHR system. The detailed discussion of the key findings regarding research questions 4, 5, 6 and 7 demonstrated that a Couchbase database has performed better than a MySQL database in most of the tests, however, MySQL database has superior analysis performance for ad-hoc queries and stores the data using less space compared to Couchbase database.

Then the research as a whole is evaluated using Design Science research assessment guidelines (Hevner et al. 2004). Each of these assessment criteria is discussed in relation to the relevant stage of this study. This research satisfies the design science assessment guidelines and contributes to theory and practice by suggesting a feasible solution to a real world research problem.

Chapter 7 – Conclusion

7.1 Introduction

Data management is a significant challenge in data intensive applications (Cattell 2011; Konishetty et al. 2012; Valduriez 2011). EHR systems and their underlying data management systems are attracting increased attention from academics and industry as high availability, high performance and scalability are sought-after features in healthcare information systems (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Klein et al. 2014; Kruse et al. 2016; Raghupathi & Raghupathi 2014). However, literature suggests that the feasibility of using NoSQL databases depends on the actual use case and there is limited empirical research that has empirically evaluated the usage of NoSQL databases in the healthcare domain (Hadjigeorgiou 2013; Li & Manoharan 2013; Nance et al. 2013).

This research investigated the feasibility of the usage of NoSQL databases in large scale EHR systems using a Design Science Research Approach. Results of this empirical research were conclusive, as the selected NoSQL document database, Couchbase, was shown to outperform its chosen relational database alternative, MySQL, in most of the test cases for database operations and also demonstrated significantly better scalability capabilities.

In this final chapter, the research focus and key findings are presented. Then, the research activities are summarised and presented in relation to each of the research questions. This is followed by a discussion of the contributions this research has made to theory and practice. The limitations of this study are acknowledged and future areas of research are highlighted. Finally, in the concluding section of this chapter, a brief summary of this PhD study is presented. The structure of this chapter is shown in Figure 7.1.

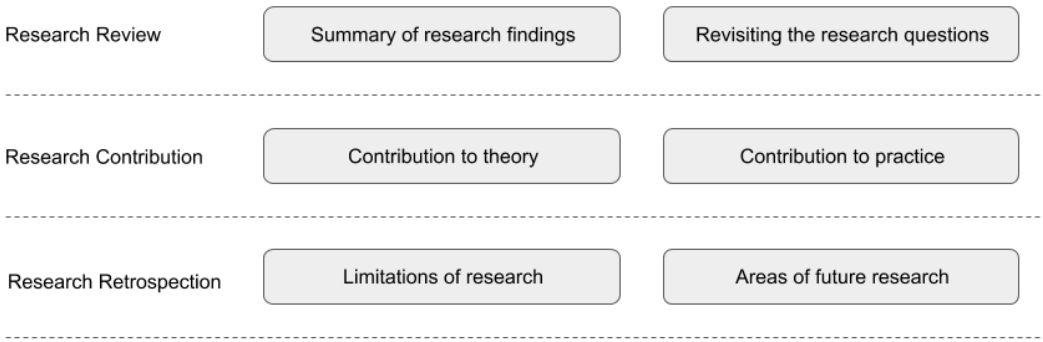


Figure 7.1 Structure of Chapter 7

7.2 Summary of Study

7.2.1 Research Problem

In a broad sense, the aim of this research was to explore the feasibility of the usage of NoSQL document databases in large scale EHR systems. Hence this study addresses the following general research question:

How can a simulation of a large EHR system be developed so that the performance of NoSQL document databases comparative to relational databases can be evaluated?

Past empirical research suggests that the expanding size of healthcare systems in general is a major obstacle for EHR systems. Moreover, for EHR systems to be able facilitate exchange of health information, these types of systems should be scalable and flexible (Blobel 2006; Freire et al. 2016; Lee, Tang & Choi 2013; Orfanidis, Bamidis & Eaglestone 2004). Furthermore, the heterogeneous nature of healthcare data is also considered a bottleneck for EHR system implementations. Most of the current EHR systems are based on relational databases which do not support a flexible data schema (Dolin et al. 2006; Guo et al. 2005; Guo et al. 2004; Jin, Deyu & Xianrong 2011; Schmitt & Majchrzak 2012; Takeda et al. 2000).

As discussed in Chapter 2, data storage systems are crucial for all sorts of data intensive applications which increasingly need to store and manage huge amounts of data. Modern applications such as high-traffic web sites or large enterprise systems require new approaches to data storage in order to achieve higher performance and higher availability than is possible with traditional relational database management

systems (RDBMS). This is particularly the case when it also involves unstructured data or when flexible data models are a requirement. Therefore, using NoSQL document databases has significant potential to lead to better EHR applications in terms of scaling, flexibility and high availability (Jin, Deyu & Xianrong 2011; Lee, Tang & Choi 2013; Schmitt & Majchrzak 2012).

Previous studies suggest that NoSQL databases have many technical and financial advantages for large scale data intensive applications (Borkar, Carey & Li 2012; Manyam et al. 2012; Meijer & Bierman 2011; Mengchen 2011). However, there is no unanimous agreement in the literature on the overall superiority of NoSQL databases over traditional relational databases in all cases, or generic suitability for data-intensive applications.

Previous research in the healthcare domain on this topic is largely limited to evaluating basic database performance of NoSQL databases in comparison to relational databases. However, inadequate attention has been given to establishing a healthcare data model and testing the performance with realistic healthcare data sets in terms of size for a large scale implementation to validate the comparison between NoSQL databases and relational databases. Clearly, this may lead to results which deviate from what would be obtained in a real-world scenario.

Hence, first this research attempted to demonstrate how a large scale EHR system can be established using NoSQL databases by selecting the right NoSQL database type, document store and establishing a realistic healthcare data model. Secondly, this research aimed to demonstrate how well NoSQL document databases perform compared to relational databases in terms of performance, scalability, data sharing and analysis capabilities in a real life-like scenario. Relational databases currently are predominately used in healthcare.

7.2.2 Research Methodology – Design and Evaluation Activities

Design Science Research was described and justified as a suitable research paradigm and methodological approach for this study. Design Science is defined as being a problem solving paradigm and, as a methodological approach, establishes a solid basis for contributing to the existing literature by developing and evaluating IT artefacts to derive useful and relevant conclusions by providing solutions to real world problems.

To address the identified gap in the literature, using a Design Science approach, in the first phase, healthcare data requirements for the EHR system were determined based on Australian healthcare minimum data sets; and then data models are established for a NoSQL document database and a relational database using relevant data modelling practices for NoSQL document databases and relational databases. Then, an IT artefact referred to as a Random Healthcare Data Generator was developed to generate synthetic EHR data based on publicly available Australian healthcare statistics.

NoSQL database types were evaluated in the context of the requirements of data models for storing EHRs in the healthcare domain, and a document based NoSQL database, Couchbase, was found to be suitable for the main objectives and specific research questions investigated in this research. In order to conduct a comparative evaluation, a relational database, MySQL, was selected because of its ability to run as a cluster. Furthermore MySQL databases are already being used in healthcare practice and are supported by a number of significant vendors.

Following these steps, a second artefact, an EHR system prototype, was developed as the facilitating system between the Random Healthcare Data Generator and the underlying NoSQL document database and relational database, Couchbase and MySQL. This artefact enabled this study to conduct a simulation of a large scale EHR system to evaluate the performance of a NoSQL document database comparative to a relational database. This artefact is designed to handle database operations including insert, update, delete operations, EHR sharing and complex querying in a simulation of a large scale EHR system. This artefact is designed to run in a distributed environment to enable sufficient concurrent client operations; and also was responsible for measuring the performance metrics for the operations and logging of the results. The details of the establishment of the two EHR data models for Couchbase and MySQL and the development of these two artefacts were presented in Chapter 4.

Through the development of these artefacts, a crucial step in a Design Science Research approach is achieved (Hevner et al. 2004; March & Smith 1995; Rossi & Sein 2003). Then, the next step, the evaluation, was executed by running a number of tests for database operations such insert, update and delete of EHRs, scalability, EHR sharing and complex queries for both Couchbase database and MySQL database. This enables this study to determine whether NoSQL databases are superior to relational

databases in multiple aspects of EHR in a distributed data management environment (Hevner et al. 2004).

In Chapter 5 the detailed results of the evaluation phase are presented for each test executed on different numbers of nodes (1, 2, 4, 8, 16), as well as different data sizes (1M, 10M, 100M and 500 M records) for both databases. The key findings from the results of these tests are discussed in detail in Chapter 6. The Couchbase database has been found to perform better than MySQL database in most of the tests, however, the MySQL database showed its strength in data analysis capabilities, particularly for ad-hoc queries and it also required less disk storage space than the Couchbase database to store the same number of EHRs.

In terms of performance, the Couchbase database outperformed the MySQL database in all node configurations for insert, update and delete operations, as well as in EHR sharing simulation that involves retrieval of all EHRs for a particular patient. The Couchbase database has demonstrated better response times and the average number of executions per seconds was significantly higher than the MySQL database for insert, update and delete operations. For the EHR sharing simulation the Couchbase database also performed better than the MySQL database, however, the difference was less significant.

Furthermore, the Couchbase database demonstrated 30% better scalability than MySQL database. It should be noted that the MySQL database has limitations in terms of design on scaling, while Couchbase database is designed to scale better.

The MySQL database performed better on ad-hoc complex queries than the Couchbase database, which is another aspect of this research. Although the Couchbase database can respond to pre-defined queries (views) almost instantly, the time taken by Couchbase to execute a complex query for the first time was much longer than for the MySQL database. Couchbase needed to generate the view for a query first up, which took more time than for the MySQL database to return a response to a query. Moreover, there is no requirement to define the query beforehand for MySQL database.

The results presented and discussed in Chapter 6 clearly demonstrate that NoSQL document databases are promising alternatives to be used as an underlying primary

data store for large scale EHR implementations. However, for further analysis of healthcare data, a relational database or a data warehouse could be a better option as NoSQL databases did not perform better on the data analysis of EHRs when ad-hoc queries are executed. It is also worth noting that the same number of EHRs took around double the amount of disk space for storage when using the Couchbase database compared to the MySQL database. This result was due to the differences in the approach and design of the two EHR data models.

Following the discussion of the test results, this research in terms of process (research activities) and product (output – artefacts) is evaluated using assessment guidelines for Design Science Research proposed by Hevner et al (2004), to assess the alignment of the research design of this study with the key objectives of a Design Science Research Approach.

In the following section, the key findings in relation to each research question are summarised to demonstrate how each research question was addressed in this study.

7.3 Summary of Key Findings for each Research Question Investigated

RQ1: How can a NoSQL document data model and a relational data model be developed for an EHR system that are in line with documents published by healthcare authorities in Australia?

The National Health Data Dictionary (NHDD) published by the Australian Institute of Health and Welfare is used to determine the healthcare data that is stored in EHRs for the purpose of this study (AIHW 2015). The NHDD contains National Minimum Data Sets, which are used as the basis for establishing the key data elements in the data models. A total number of 49 unique data elements are identified and the details are presented in Section 4.2. Based on previous studies, a document data model was determined as suitable for storing healthcare data in a NoSQL document database. Thus, an aggregate oriented data model for a NoSQL document database was established and a relational data model for a relational database was established for this research, as outlined in Section 4.3 of chapter 4.

RQ2: How can a random healthcare data generator be developed that will generate EHRs that are representative of the characteristics of Australian healthcare data based on statistics available in the public domain?

A Random Healthcare Data Generator artefact is developed using publicly available statistics for the Australian healthcare system. Multiple statistics are identified as relevant for populating the two established data models (Document, Relational) with EHRs based on minimum data sets. A multinomial distribution drawing on probability theory was used to generate random data based on these statistics. The Australian Healthcare statistics used to generate the characteristics of EHRs are presented in Section 4.4 of Chapter 4 and the details of the design of the Random Healthcare Data Generator artefact are presented in Section 4.5 of Chapter 4.

RQ3: How can a prototype EHR system be developed that will facilitate database operations and measure performance and scalability for NoSQL document databases and relational databases?

After the Random Healthcare Data Generator was developed, another artefact, an EHR system prototype, is developed to facilitate the simulation of a large scale EHR system and database operations to be executed in the context of this research. This artefact was responsible for data sharing executions on both NoSQL and relational databases and measurement of the metrics for evaluation. Details of the design of this artefact are presented in Section 4.6 of chapter 4.

RQ4: How do NoSQL document databases perform compared to relational databases in executing basic database operations such as insert, delete and update on electronic health records?

The Couchbase database was selected as the NoSQL document database and the MySQL database was selected as the relational database to be evaluated in a simulation of a large scale EHR system in this research. For the basic database operations of insert, update and delete record, the Couchbase database demonstrated 5 to 26 times better performance compared to the MySQL database. In addition to this significant difference in performance, the Couchbase database also demonstrated predictable performance in terms of response time and average number of executions

per second, while MySQL showed a higher range of variation in terms of response time and average number of executions.

RQ5: How do NoSQL document databases scale compared to relational databases in electronic health record systems?

In addition to the results regarding the performance of basic database operations, the Couchbase database has also demonstrated better scalability than the MySQL database. The improvement in average number of executions per second was 89% for the Couchbase database and 70% for the MySQL database when the number of nodes was doubled. In addition to that, the MySQL database had design limitations on the number of nodes that can be added to a cluster.

RQ6: How do NoSQL document databases perform compared to relational databases in supporting electronic health record sharing through patient record retrieval in a distributed EHR system?

The Couchbase database performed better than the MySQL database in the data retrieval for EHR sharing simulation test. However, the difference between databases was around 29%—which was not as significant difference as the results of the basic database operations tests due to the complex nature of retrieving multiple EHRs for EHR sharing purpose as discussed in Section 6.2.6 of chapter 6.

RQ7: How do NoSQL document databases perform compared to relational databases in executing complex queries on electronic health records?

This research question was addressed by running complex query tests. It is possible to generate views with the Couchbase database using complex map-reduce codes written in JavaScript. These views enable the results of a query to be displayed almost instantly after the initial execution of the code (Couchbase 2015). However, initial execution of the view code for the complex query test took more time than the same query executed on MySQL database because the required indexes needed to be created. The Couchbase database was able to provide fast access to the results of pre-defined queries, however, this can only be achieved after the initial view creation process was completed which takes much longer to run than the same query in the MySQL relational database. Thus, the Couchbase database demonstrated limited capabilities when running arbitrary ad-hoc queries compared to the MySQL database.

Therefore, it is concluded from this study that relational databases or data warehouse applications could be used as complementary systems to NoSQL document databases for data analysis purposes.

7.4 Research Contributions to Theory and Practice

7.4.1 Contribution to Theory

Healthcare is one of the important domains of electronic data exchange. The literature review suggests that the current problems of electronic data exchange in healthcare have not been effectively and completely addressed. Thus, more practical and relevant research is needed to address this important topic by developing and evaluating real world solutions. First and foremost, this research provides a sound basis for other potential researchers to study healthcare data sharing issues by creating awareness of the potential opportunities and challenges of using emerging technologies such as NoSQL databases and by developing and evaluating a solution to a particular problem. It has been identified that there is a need for more clarity on whether it is better to use NoSQL document databases for large scale EHR system implementations in terms of performance, scalability, data sharing and data analysis aspects in the healthcare domain.

In this research, a framework is established for evaluating the performance of NoSQL document database systems in terms of performance, scalability, data sharing and data analysis features in the context of large scale EHR systems. The researcher believes that a document-based NoSQL database model is a more appropriate approach that meets the current and emerging data requirements of EHRs rather than the traditional approach based on the relational database model. This study has contributed to closing the gap in the literature by conducting an extensive empirical evaluation of the promising NoSQL document database technology in the important field of healthcare.

There are number of theoretical contributions that are specified for Design Science research in terms of the IT artefact (Gregor & Hevner (2013). This research made a number of important theoretical contributions that were achieved by adapting new technologies that have emerged in other fields into a new field, which is referred to as an ‘exaptation’. These included the development and evaluation of a number of important and interdependent artefacts which were essential in achieving the main objectives of this research. These were a data model for storing EHRs in a NoSQL

document database, a random Healthcare data generator for generating synthetic EHR data, and a prototype EHR system to facilitate database operations and EHR sharing in a simulation of a large scale EHR system.

A document based NoSQL data model is established in the course of this research based on the Australian National Health Data Dictionary published by the Australian Institute of Healthcare and Welfare (AIHW 2015). This data model is designed based on best practice and uses an aggregate oriented approach where EHRs are stored in JSON format (Goli-Malekabadi, Sargolzaei-Javan & Akbari 2016; Vera et al. 2015). This document data model provides an important theoretical contribution to the kernel theory of data modelling. This research used an aggregate oriented approach for the data modelling of EHRs for NoSQL document databases which are considered to be highly suitable for the data management of EHRs (Gudivada, Rao & Raghavan 2016).

The second IT artefact developed and evaluated for this research is a Random Healthcare Data Generator. The method used to develop and evaluate this artefact made an important contribution to the design theory of building and evaluating an artefact for generating synthetic EHR data to simulate a large scale EHR system (AIHW 2015, 2016).

This IT artefact can help future research to avoid ethical issues in dealing with domain-specific data when privacy is a concern, such as is the case with healthcare data. This approach can be applied to any domain, including healthcare, when the data model can be established based on published industry data sets and elements; and statistics are available to identify data characteristics of data sets and data elements in a particular industry domain. The development and evaluation of the random healthcare data generator as a critical component in the simulation of a large scale EHR system is a significant contribution of this research. This IT artefact for which the source code will be made available in the public domain via Github enables researchers to easily generate random data at big data scale based on public data sets and their elements with characteristics that are statistically representative a range of domains where such required information is available in the public domain.

The main contribution of this research is the development and evaluation of a third IT artefact, an EHR system prototype that enabled the performance evaluation of NoSQL document databases comparative to relational databases. This artefact enabled the

simulation of a large scale EHR system for the main purpose of evaluating the performance of a NoSQL document database. The prototype EHR system managed the execution of database operations for insert, update and delete operations, EHR data sharing, and execution of complex queries, as well as capturing the measurement of performance metrics for each test case. It is expected that the key findings of this research will encourage both academics and practitioners to adapt, test and use NoSQL technologies in healthcare-related research and applications. The main theoretical contribution of this empirical research to design theory was the design and evaluation of a prototype EHR system for simulating database management operations in a large scale EHR system environment. This artefact demonstrates through a simulated performance evaluation that a NoSQL document database has significant and proven performance advantages over relational databases in most of the database management test cases. Hence this study demonstrated the utility and efficacy of a NoSQL document database in the simulation of a large scale EHR system.

7.4.2 Contribution to Practice

This research has made a number of important contributions to practice foremost is that the IT artefacts (namely, a data model for storing EHRs in a NoSQL document database, a random healthcare data generator and a prototype EHR system) developed and evaluated in this research can be readily adopted by practitioners. The research activities undertaken to develop and evaluate these artefacts is described and justified in this PhD Thesis which will be made available in the public domain after a one year embargo. The design of the data model for storing EHRs in a NoSQL document database and the source code for the random healthcare data generator and the source code for the prototype EHR system will be made available online via a GitHub repository.

Another important practical contribution of this research is that it is based on the open source availability of many NoSQL database alternatives. Hence, this research will encourage developing and under-developed countries to establish their own cost-effective national EHR systems without the restrictions, limitations, complexity or complications of similar proprietary relational database systems. The approach and solutions of this research will also help healthcare providers with multiple establishments delivering healthcare services to different locations to develop their

own central data storage and data sharing systems without the requirement of big initial investments and difficult implementation processes to achieve high availability.

7.5 Limitations and Future Research

Both healthcare and database systems are major areas of research involving many different aspects. Therefore, it is not possible to cover all related matters regardless of how comprehensive a research is. The security and privacy concerns on EHR sharing, issues about interoperability, data standardisation, coding systems and many other healthcare related topics are out of the scope of this PhD study. Furthermore, ACID or BASE properties, replication and consistency considerations, compression and encryption features of database systems are also not covered in this research.

One of the major limitations of this project is the identification of the data model. Although EHRs could contain many different types of information, structured or free-text, this study is based on the minimum data sets and mandatory data elements published by the Australian Institute of Health and Welfare. While establishing a sound basis on what needs to be included in the proposed data model, it also limits the overall data model to an administrative perspective for data collection rather than a medical perspective. However, as discussed in Chapter 2, NoSQL databases offer flexible data models. Thus, the effect of including medical data elements which are more complex in nature and require significantly more data storage space than the current data model is expected to have minimal impact on the outcome of the NoSQL document database test results. Flexible data models of NoSQL databases would also be a prime area for future research to more extensively test the performance of NoSQL document databases using healthcare data that includes medical imaging results and free-text physician notes.

The researcher also acknowledges that the selection of the database system for both NoSQL and relational databases can be considered as a limitation of this research. There are numerous commercial vendor offerings of NoSQL database systems and the number of available NoSQL database systems is increasing rapidly. Therefore, before the completion of this research, some alternative database solutions or newer versions of the database systems selected in this research may have emerged.

Furthermore, the researcher acknowledges that a NoSQL document database and a relational database cannot be considered as fully equivalent for a comparison as their underlying technical design are different. However, the same configurations were used for each of the test scenarios for Couchbase database and MySQL database to make the outcome as accurate as possible in terms of an overall performance comparison in the context of their fitness for the use case of data management of EHRs from an IS perspective.

Another limitation of this research is that the test environment was the cloud environment (Elastic Compute Cloud) provided by Amazon Web Services. The tests could also have been conducted in a number of different environments, such as using local hardware or other cloud vendors, using different or newer versions of database software, different database configuration and tunings or even different databases. It is not practically possible to conduct the tests on all possible environments and configurations within the scope of a three year PhD program. Therefore, the selection of software and test configurations in this research is justified within reasonable grounds where possible. It is also noteworthy to mention that the improvements of database systems and cloud environments are rapid and it is not always possible to keep up with the speed that these types of technology evolve and advance over the duration of a three year PhD-level research program.

In this research it is observed that NoSQL databases document cannot perform significantly better than relational databases when executing arbitrary ad-hoc queries required for further data analysis using relevant applications such as Business Intelligence tools. Therefore, exploring or enhancing data analysis capabilities of NoSQL document databases seems to be an area worthy of further research given the increasing uptake of NoSQL databases in mainstream IT practice. Moreover, using a relational database or a data warehouse application together with a NoSQL document database might be a promising way of implementing large-scale and distributed data management systems requiring significant operational capabilities, as well as rich query environment. Thus, extracting data from NoSQL databases using pre-defined queries and aggregating or summarising the raw data and saving the results into a relational database or a data warehouse application is another area that can be evaluated and is an area worthy of future research in healthcare—or any other domain for that matter. This can be achieved using batch or streaming processing and analytics

as tools such as Kafka to support the transformation of the data between SQL tables and JSON documents effectively (Mitchell & Tucker 2017).

Furthermore, the research also mentions the flexible data model and distributed parallel processing capability of NoSQL databases that have great potential of the technology from a clinical perspective by enabling clinical decision support and effective management of heterogeneous unstructured clinical data both of which may be areas of future research.

Moreover, in this research the performance evaluation of EHR sharing as a technical operation that was limited in scope to the data retrieval of patient's EHRs in the simulation of a large scale EHR system. The literature also highlights the importance of addressing security and privacy concerns with data encryption in considering EHR sharing in conjunction with the emerging database technologies like NoSQL document databases. While privacy and security concerns associated with EHR sharing and data encryption as a technical solution is beyond the scope of this research it is another area of EHR sharing worthy of further research to complement and build on foundations established by this research.

7.6 Summary

This research has demonstrated the feasibility and potential benefits of using NoSQL databases in large scale EHR systems through the evaluation of the performance of a NoSQL document database comparative to a relational database in the healthcare domain, which made important contributions to both theory and practice. A Design Science Research approach was used to undertake and complete this research. The performance evaluation of a NoSQL database was conducted by developing and evaluating IT artefacts specifically designed for achieving the main objectives of this study by investigating seven research questions. The research activity and outputs of this study were evaluated using the DSR evaluation guidelines identified in the relevant literature and the results and contribution of this research are presented in the context of the Design Science paradigm. NoSQL document databases have promising features and their performance, scalability, data sharing and analysis capabilities were evaluated thoroughly in this research.

The research has demonstrated that NoSQL document databases outperform relational databases in a simulation of a large scale EHR system for basic database operations such as insert, update, and delete, and EHR sharing. However the selected relational database, MySQL, has shown superior performance in executing complex ad-hoc queries compared to the selected NoSQL document database, Couchbase.

The gap in the literature is addressed by the empirical evaluation of performance and scalability of a NoSQL document database compared to a relational database in large scale EHR systems context. Furthermore, as a practical contribution, the data models and IT artefacts developed in this research also provide guidance to industry and enable researchers to conduct similar researches using the approaches and artefacts presented in this research.

Therefore, the research has met its objectives and the outcomes of this research provides a solid basis for industry and researchers to undertake future research activities complementing the usage of NoSQL document databases in large scale EHR systems, such as data warehousing applications, encryption and privacy protection approaches.

List of References

Abelló, A, Ferrarons, J & Romero, O 2011, 'Building cubes with MapReduce', *proceedings of the ACM 14th international workshop on Data Warehousing and OLAP* ACM, pp. 17-24.

Abiteboul, S, Hull, R & Vianu, V 1995, *Foundations of databases*, vol. 8, Addison-Wesley.

Aboutorabi, SH, Rezapour, M, Moradi, M & Ghadiri, N 2015, 'Performance evaluation of SQL and MongoDB databases for big e-commerce data', *proceedings of the 2015 International Symposium on Computer Science and Software Engineering (CSSE)* pp. 1-7.

Abramova, V & Bernardino, J 2013, 'NoSQL databases: MongoDB vs cassandra', *proceedings of the International C* Conference on Computer Science and Software Engineering* ACM, Porto, Portugal, pp. 14-22.

Agrawal, D, Das, S & El Abbadi, A 2011, 'Big data and cloud computing: current state and future opportunities', *proceedings of the 14th International Conference on Extending Database Technology* ACM, pp. 530-3.

AIHW 2015, *National Health Data Dictionary. Version 16.2*, AIHW, Canberra, ISBN 978-1-74249-690-0, <http://www.aihw.gov.au/publication-detail/?id=60129550408>.

AIHW, *Admitted patient care 2014–15: Australian hospital statistics*, 2016, Canberra.

Aji, A, Wang, F, Vo, H, Lee, R, Liu, Q, Zhang, X & Saltz, J 2013, 'Hadoop GIS: a high performance spatial data warehousing system over mapreduce', *Proc. VLDB Endow.*, vol. 6, no. 11, pp. 1009-20.

Aljafari, R & Khazanchi, D 2013, 'On the veridicality of claims in design science research', *proceedings of the System Sciences (HICSS), 2013 46th Hawaii International Conference on* IEEE, pp. 3747-56.

Alnuem, M, Samir, ELM, Youssef, A & Emam, A 2011, 'Towards Integrating National Electronic Care Records in Saudi Arabia', *proceedings of the 2011 World Congress in Computer Science, Computer Engineering, and Applied Computing.* Las Vegas, Nevada, USA.

Alturki, A, Gable, G & Bandara, W 2011, 'A design science research roadmap', *Service-Oriented Perspectives in Design Science Research*, pp. 107-23.

Amazon 2016, *Amazon EC2 - Virtual Server Hosting* viewed 1 July 2016, <https://aws.amazon.com/ec2/>.

Atzeni, P, Jensen, CS, Orsi, G, Ram, S, Tanca, L & Torlone, R 2013, 'The relational model is dead, SQL is dead, and I don't feel so good myself', *SIGMOD Rec.*, vol. 42, no. 2, pp. 64-8.

Australian Digital Health Agency 2015, *Australian Digital Health Agency*, <http://www.digitalhealth.gov.au >.

Avalon Consulting 2016, *Benchmark: MongoDB 3.2 vs. Couchbase Server 4.5 for Query and Read/Write Performance*, viewed 1 August 2016, <http://info.couchbase.com/2016_Benchmark_MongoDB_3_2_vs_Couchbase_Server_4_5_HP_TOP.html>.

Bacelar-Silva, GM, Vicente, CMO, David, M & Antunes, L 2011, 'Comparing security and privacy issues of EHR: Portugal, the Netherlands and the United Kingdom', *proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies* ACM, Barcelona, Spain, pp. 1-4.

Badia, A & Lemire, D 2011, 'A call to arms: revisiting database design', *ACM SIGMOD Record*, vol. 40, no. 3, pp. 61-9.

Bailis, P & Ghodsi, A 2013, 'Eventual consistency today: limitations, extensions, and beyond', *Commun. ACM*, vol. 56, no. 5, pp. 55-63.

Bailis, P, Fekete, A, Ghodsi, A, Hellerstein, JM & Stoica, I 2013, 'HAT, not CAP: towards highly available transactions', *proceedings of the 14th USENIX conference on Hot Topics in Operating Systems* USENIX Association, Santa Ana Pueblo, New Mexcio, pp. 24-.

Barata, M, Bernardino, J & Furtado, P 2014, 'Survey on Big Data and Decision Support Benchmarks', *proceedings of the International Conference on Database and Expert Systems Applications* Springer, pp. 174-82.

Bergmann, J, Bott, OJ, Pretschner, DP & Haux, R 2007, 'An e-consent-based shared EHR system architecture for integrated healthcare networks', *International journal of medical informatics*, vol. 76, no. 2, pp. 130-6.

Bermbach, D & Tai, S 2011, 'Eventual consistency: How soon is eventual? An evaluation of Amazon S3's consistency behavior', *proceedings of the 6th Workshop on Middleware for Service Oriented Computing* ACM, p. 1.

Berndt, DJ, Hevner, AR & Studnicki, J 2003, 'The Catch data warehouse: support for community health care decision-making', *Decision support systems*, vol. 35, no. 3, pp. 367-84.

Bernstein, PA 1976, 'Synthesizing third normal form relations from functional dependencies', *ACM Trans. Database Syst.*, vol. 1, no. 4, pp. 277-98.

Biyikoglu, C, 2016, 'Couchbase Server Hits One Million Writes Per Second with Just 50 Nodes of Google Compute Engine', viewed 28 May 2016,

<https://cloudplatform.googleblog.com/2015/05/Couchbase-Server-Hits-One-Million-Writes-Per-Second-with-Just-50-Nodes-of-Google-Compute-Engine.html>.

Blobel, B 2006, 'Advanced and secure architectural EHR approaches', *International journal of medical informatics*, vol. 75, no. 3, pp. 185-90.

Borkar, D, Mayuram, R, Sangudi, G & Carey, M 2016, 'Have Your Data and Query It Too: From Key-Value Caching to Big Data Management', *proceedings of the 2016 International Conference on Management of Data* ACM, San Francisco, California, USA, pp. 239-51.

Borkar, VR, Carey, MJ & Li, C 2012, 'Big data platforms: What's next?', *XRDS*, vol. 19, no. 1, pp. 44-9.

Calder, BJ, Phillips, LW & Tybout, AM 1982, 'The concept of external validity', *Journal of Consumer Research*, vol. 9, no. 3, pp. 240-4.

Cattell, R 2011, 'Scalable SQL and NoSQL data stores', *SIGMOD Rec.*, vol. 39, no. 4, pp. 12-27.

CDAC 2009, *DIGHT*, viewed 10.10.2011, <http://dight.sics.se/>.

Chaim, RM, Oliveira, EC & Araújo, APF 2017, 'Technical specifications of a service-oriented architecture for semantic interoperability of EHR — electronic health records', *proceedings of the 2017 12th Iberian Conference on Information Systems and Technologies (CISTI)* pp. 1-6.

Chatterjee, S 2015, 'Writing My next Design Science Research Master-piece: But How Do I Make a Theoretical Contribution to DSR?', *proceedings of the Twenty-Third European Conference on Information Systems* Münster, Germany.

Chen, F & Hsu, M 2013, 'A performance comparison of parallel DBMSs and MapReduce on large-scale text analytics', *proceedings of the 16th International Conference on Extending Database Technology* ACM, Genoa, Italy, pp. 613-24.

Chen, PP-S 1976, 'The entity-relationship model\—toward a unified view of data', *ACM Trans. Database Syst.*, vol. 1, no. 1, pp. 9-36.

Chen, Y 2010, *Introduction to probability theory*, The lecture notes on information theory. Duisburg-Essen University.

Codd, EF 1970, 'A relational model of data for large shared data banks', *Communications of the ACM*, vol. 13, no. 6, pp. 377-87.

Cooper, BF, Silberstein, A, Tam, E, Ramakrishnan, R & Sears, R 2010, 'Benchmarking cloud serving systems with YCSB', *proceedings of the 1st ACM symposium on Cloud computing* ACM, Indianapolis, Indiana, USA, pp. 143-54.

Corbett, JC, Dean, J, Epstein, M, Fikes, A, Frost, C, Furman, JJ, Ghemawat, S, Gubarev, A, Heiser, C, Hochschild, P, Hsieh, W, Kanthak, S, Kogan, E, Li, H,

Lloyd, A, Melnik, S, Mwaura, D, Nagle, D, Quinlan, S, Rao, R, Rolig, L, Saito, Y, Szymaniak, M, Taylor, C, Wang, R & Woodford, D 2013, 'Spanner: Google's Globally Distributed Database', *ACM Trans. Comput. Syst.*, vol. 31, no. 3, pp. 1-22.

Couchbase 2015, *View Basics*, viewed 01.12.2016, <http://docs.couchbase.com/admin/admin/Views/views-basics.html>.

Couchbase 2016, *Why NoSQL*, <https://www.couchbase.com/resources/why-nosql>.

Creswell, J 2013, 'Standards of validation and evaluation', *Qualitative inquiry and research design: choosing among five approaches*, vol. 2, pp. 201-21.

De Pietro, C & Francetic, I 2017, 'E-health in Switzerland: The laborious adoption of the federal law on electronic health records (EHR) and health information exchange (HIE) networks', *Health Policy*.

Dede, E, Fadika, Z, Gupta, C & Govindaraju, M 2011, 'Scalable and Distributed Processing of Scientific XML Data', *proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing* IEEE Computer Society, pp. 121-8.

Dede, E, Govindaraju, M, Gunter, D, Canon, RS & Ramakrishnan, L 2013, 'Performance evaluation of a MongoDB and hadoop platform for scientific data analysis', *proceedings of the 4th ACM workshop on Scientific cloud computing* ACM, New York, New York, USA, pp. 13-20.

Dey, A, Fekete, A & Röhm, U 2013, 'Scalable transactions across heterogeneous NoSQL key-value data stores', *Proc. VLDB Endow.*, vol. 6, no. 12, pp. 1434-9.

Dogac, A, Yuksel, M, Avci, A, Ceyhan, B, Hülür, U, Eryilmaz, Z, Mollahaliloglu, S, Atbakan, E & Akdag, R 2011, 'Electronic health record interoperability as realized in the Turkish health information system', *Methods of information in medicine*, vol. 50, no. 2, p. 140.

Dolin, RH, Alschuler, L, Boyer, S, Beebe, C, Behlen, FM, Biron, PV & Shvo, AS 2006, 'HL7 clinical document architecture, release 2', *Journal of the American Medical Informatics Association*, vol. 13, no. 1, pp. 30-9.

Drejhammar, F 2010, January 2010, 'Designing a Trusted Distributed Long-Term Archive for Health Records', *ERCIM News*, no. 80,

Dubé, L & Paré, G 2003, 'Rigor in information systems positivist case research: current practices, trends, and recommendations', *MIS quarterly*, pp. 597-636.

Edlich, S 2017, *NoSQL Database List*, viewed December 2017, <http://nosql-database.org/>.

Englebardt, SP & Nelson, R 2002, *Health care informatics: An interdisciplinary approach*, Mosby Incorporated.

Escriva, R, Wong, B & Sirer, EG 2012, 'HyperDex: a distributed, searchable key-value store', *proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication* ACM, Helsinki, Finland, pp. 25-36.

Fagin, R 1977, 'Multivalued dependencies and a new normal form for relational databases', *ACM Trans. Database Syst.*, vol. 2, no. 3, pp. 262-78.

Featherston, D 2010, *Cassandra: Principles and Application*, viewed 27.08.2012, <http://disi.unitn.it/~montreso/ds/papers/Cassandra.pdf>.

Fernando, SF 2016, *NoSQL ?*, <https://www.linkedin.com/pulse/nosql-suzanne-fiona-fernando>.

Ferreira, GdS, Calil, A & Mello, RdS 2013, 'On Providing DDL Support for a Relational Layer over a Document NoSQL Database', *proceedings of the International Conference on Information Integration and Web-based Applications & Services* ACM, Vienna, Austria, pp. 125-32.

Floratou, A, Teletia, N, DeWitt, DJ, Patel, JM & Zhang, D 2012, 'Can the elephants handle the NoSQL onslaught?', *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 1712-23.

Frade, S, Freire, SM, Sundvall, E, Patriarca-Almeida, JH & Cruz-Correia, R 2013, 'Survey of openEHR storage implementations', *proceedings of the Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems* pp. 303-7.

Freire, SM, Teodoro, D, Wei-Kleiner, F, Sundvall, E, Karlsson, D & Lambrix, P 2016, 'Comparing the Performance of NoSQL Approaches for Managing Archetype-Based Electronic Health Record Data', *PloS one*, vol. 11, no. 3, p. e0150069.

Gilbert, S & Lynch, N 2012, 'Perspectives on the CAP Theorem', *Computer*, vol. 45, no. 2, pp. 30-6.

Gill, TG & Hevner, AR 2013, 'A fitness-utility model for design science research', *ACM Transactions on Management Information Systems (TMIS)*, vol. 4, no. 2, p. 5.

Golafshani, N 2003, 'Understanding reliability and validity in qualitative research', *The qualitative report*, vol. 8, no. 4, pp. 597-606.

Goldkuhl, G 2004, 'Design theories in information systems-a need for multi-grounding', *JITTA: Journal of Information Technology Theory and Application*, vol. 6, no. 2, p. 59.

Goli-Malekabadi, Z, Sargolzaei-Javan, M & Akbari, MK 2016, 'An effective model for store and retrieve big health data in cloud computing', *Computer Methods and Programs in Biomedicine*, vol. 132, pp. 75-82, NLM, item: 27282229.

Gorton, I, Klein, J & Nurgaliev, A 2015, 'Architecture Knowledge for Evaluating Scalable Databases', *proceedings of the 2015 12th Working IEEE/IFIP Conference on Software Architecture* pp. 95-104.

Gray, J 1981, 'The transaction concept: virtues and limitations (invited paper)', *proceedings of the Seventh International conference on Very Large Data Bases - Volume 7* VLDB Endowment, Cannes, France, pp. 144-54.

Green, LW 1977, 'Evaluation and measurement: some dilemmas for health education', *Am J Public Health*, vol. 67, no. 2, pp. 155-61, NLM, item: 402085.

Gregor, S & Jones, D 2007, 'The anatomy of a design theory', *Journal of the Association for Information Systems*, vol. 8, no. 5, pp. 312-35.

Gregor, S & Hevner, AR 2013, 'POSITIONING AND PRESENTING DESIGN SCIENCE RESEARCH FOR MAXIMUM IMPACT', *MIS quarterly*, vol. 37, no. 2, pp. 337-55.

Grimson, J 2001, 'Delivering the electronic healthcare record for the 21st century', *International journal of medical informatics*, vol. 64, no. 2, pp. 111-27.

Gudivada, VN, Rao, D & Raghavan, VV 2016, 'Renaissance in database management: navigating the landscape of candidate systems', *Computer*, vol. 49, no. 4, pp. 31-42.

Gunter, TD & Terry, NP 2005, 'The Emergence of National Electronic Health Record Architectures in the United States and Australia: Models, Costs, and Questions', *Journal of Medical Internet Research*, vol. 7, no. 1, p. e3, PMC, item: PMC1550638.

Guo, J, Takada, A, Tanaka, K, Sato, J, Suzuki, M, Suzuki, T, Nakashima, Y, Araki, K & Yoshihara, H 2004, 'The development of MML (Medical Markup Language) version 3.0 as a medical document exchange format for HL7 messages', *Journal of medical systems*, vol. 28, no. 6, pp. 523-33.

Guo, J, Takada, A, Niu, T, He, M, Tanaka, K, Sato, J, Suzuki, M, Suzuki, T, Nakashima, Y & Araki, K 2005, 'Enhancement of MML medical data exchange standard for a localized Chinese version', *Journal of medical systems*, vol. 29, no. 5, pp. 555-67.

Hadjigeorgiou, C 2013, 'Rdbms vs nosql: Performance and scaling comparison', MSc thesis, The University of Edinburgh.

Halamka, J, Aranow, M, Ascenzo, C, Bates, D, Debor, G, Glaser, J, Goroll, A, Stowe, J, Tripathi, M & Vineyard, G 2005, 'Health care IT collaboration in Massachusetts: the experience of creating regional connectivity', *Journal of the American Medical Informatics Association*, vol. 12, no. 6, pp. 596-601.

Haseeb, A & Pattun, G 2017, 'A review on NoSQL: Applications and challenges', *International Journal of Advanced Research in Computer Science*, vol. 8, no. 1.

Heard, S 2006, 'Electronic Heatlh Records', in M Conrick (ed.), *Health Informatics: Transforming healthcare with technology*, Melbourne, pp. 222-332.

Helfert, M, Donnellan, B & Ostrowski, L 2012, 'The case for design science utility and quality-Evaluation of design science artifact within the', *Systems, Signs & Actions*, vol. 6, no. 1, pp. 46-66.

Helland, P 2011, 'If you have too much data, then 'good enough' is good enough', *Commun. ACM*, vol. 54, no. 6, pp. 40-7.

Hermon, R & Williams, PA 2014, 'Big data in healthcare: What is it used for?', *proceedings of the 3rd Australian eHealth Informatics and Security Conference* University, Joondalup Campus, Perth, Western Australia, <http://ro.ecu.edu.au/aeis/22/>.

Hevner, AR, March, ST, Park, J & Ram, S 2004, 'Design science in information systems research', *MIS quarterly*, vol. 28, no. 1, pp. 75-105.

Hoerbst, A, Kohl, CD, Knaup, P & Ammenwerth, E 2010, 'Attitudes and behaviors related to the introduction of electronic health records among Austrian and German citizens', *International journal of medical informatics*, vol. 79, no. 2, p. 81.

Hsieh, D 2014, *NoSQL Data Modeling*, <http://www.ebaytechblog.com/2014/10/10/nosql-data-modeling/>.

Huang, L-C, Chu, H-C, Lien, C-Y, Hsiao, C-H & Kao, T 2009, 'Privacy preservation and information security protection for patients' portable electronic health records', *Computers in Biology and Medicine*, vol. 39, no. 9, pp. 743-50.

Iakovidis, I 1998, 'Towards personal health record: current situation, obstacles and trends in implementation of electronic healthcare record in Europe', *International journal of medical informatics*, vol. 52, no. 1, pp. 105-15.

Ibrahim, A, Mahmood, B & Singhal, M 2016, 'A secure framework for sharing Electronic Health Records over Clouds', *proceedings of the 2016 IEEE International Conference on Serious Games and Applications for Health (SeGAH)* pp. 1-8.

ISO 2004, *TS 18308 Health Informatics-Requirements for an Electronic Health Record Architecture*.

ISO 2011, *ISO 18308:2011 Preview: Health informatics -- Requirements for an electronic health record architecture*, <https://www.iso.org/standard/52823.html>.

Jin, J, Ahn, G-J, Hu, H, Covington, MJ & Zhang, X 2009, 'Patient-centric authorization framework for sharing electronic health records', *proceedings of the 14th ACM symposium on Access control models and technologies* ACM, Stresa, Italy, pp. 125-34.

Jin, Y, Deyu, T & Xianrong, Z 2011, 'Research on the distributed electronic medical records storage model', *proceedings of the IT in Medicine and Education (ITME), 2011 International Symposium on* IEEE, pp. 288-92.

Keller, ME, Kelling, SE, Cornelius, DC, Oni, HA & Bright, DR 2015, 'Enhancing Practice Efficiency and Patient Care by Sharing Electronic Health Records', *Perspectives in Health Information Management*, vol. 12, no. Fall, p. 1b, PMC, item: PMC4632871.

Klein, HK & Myers, MD 1999, 'A set of principles for conducting and evaluating interpretive field studies in information systems', *MIS quarterly*, pp. 67-93.

Klein, J, Gorton, I, Ernst, N, Donohoe, P, Pham, K & Matser, C 2014, *Quality Attribute-Guided Evaluation of NoSQL Databases: An Experience Report*, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Klein, J, Gorton, I, Ernst, N, Donohoe, P, Pham, K & Matser, C 2015, 'Performance evaluation of nosql databases: A case study', *proceedings of the 1st Workshop on Performance Analysis of Big Data Systems* ACM, pp. 5-10.

Kohn, LT, Corrigan, JM & Donaldson, MS 2000, *To err is human: building a safer health system*, vol. 627, National Academies Press.

Konishetty, VK, Kumar, KA, Voruganti, K & Rao, GVP 2012, 'Implementation and evaluation of scalable data structure over HBase', *proceedings of the International Conference on Advances in Computing, Communications and Informatics* ACM, Chennai, India, pp. 1010-8.

Konstantinou, I, Angelou, E, Boumpouka, C, Tsoumakos, D & Koziris, N 2011, 'On the elasticity of NoSQL databases over cloud management platforms', *proceedings of the 20th ACM international conference on Information and knowledge management* ACM, Glasgow, Scotland, UK, pp. 2385-8.

Kose, I, Akpinar, N, Gurel, M, Arslan, Y, Ozer, H, Yurt, N, Kabak, Y, Yuksel, M & Dogac, A 2008, 'Turkey's national health information system (NHIS)', *proceedings of the eChallenges Conference* pp. 22-4.10.

Kruse, CS, Goswamy, R, Raval, Y & Marawi, S 2016, 'Challenges and Opportunities of Big Data in Health Care: A Systematic Review', *JMIR Medical Informatics*, vol. 4, no. 4, p. e38, PMC, item: PMC5138448.

Kuechler, B & Vaishnavi, V 2008, 'On theory development in design science research: anatomy of a research project', *European Journal of Information Systems*, vol. 17, no. 5, pp. 489-504.

Kuechler, W & Vaishnavi, V 2012, 'A framework for theory development in design science research: multiple perspectives', *Journal of the Association for Information Systems*, vol. 13, no. 6, p. 395.

Lakshman, A & Malik, P 2010, 'Cassandra: a decentralized structured storage system', *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 2, pp. 35-40.

Laney, D 2001, '3D data management: Controlling data volume, velocity and variety', *META Group Research Note*, vol. 6, p. 70.

Leavitt, N 2010, 'Will NoSQL Databases Live Up to Their Promise?', *Computer*, vol. 43, no. 2, pp. 12-4.

Lee, AS 1989, 'A scientific methodology for MIS case studies', *MIS quarterly*, pp. 33-50.

Lee, KKY, Tang, WC & Choi, KS 2013, 'Alternatives to relational database: Comparison of NoSQL and XML approaches for clinical data storage', *Computer Methods and Programs in Biomedicine*, vol. 110, no. 1, pp. 99-109.

Li, Y & Manoharan, S 2013, 'A performance comparison of SQL and NoSQL databases', *proceedings of the 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)* pp. 15-9.

Lungu, I & Tudorica, BG 2013, 'The development of a benchmark tool for nosql databases', *Database Systems Journal BOARD*, vol. 13.

Manyam, G, Payton, MA, Roth, JA, Abruzzo, LV & Coombes, KR 2012, 'Relax with CouchDB — Into the non-relational DBMS era of bioinformatics', *Genomics*, vol. 100, no. 1, pp. 1-7.

March, ST & Smith, GF 1995, 'Design and natural science research on information technology', *Decision support systems*, vol. 15, no. 4, pp. 251-66.

MarkLogic 2014, *The NoSQL Generation: Embracing the Document Model*, http://cdn.marklogic.com/wp-content/uploads/2014/12/nosql-generation-embracing-document-model.pdf>.

Mason, RT 2015, 'NoSQL databases and data modeling techniques for a document-oriented NoSQL database", *proceedings of the Informing Science & IT Education Conference (InSITE)* pp. 259-68.

Meijer, E & Bierman, G 2011, 'A co-Relational Model of Data for Large Shared Data Banks', *Queue*, vol. 9, no. 3, pp. 30-48.

Meinel, C, Polze, A, Oswald, G, Strotmann, R, Seibold, U & Schulzki, B 2015, *HPI Future SOC Lab: Proceedings 2013*, Univ.-Verlag.

Mendelzon, AO 1984, 'Database states and their tableaux', *ACM Trans. Database Syst.*, vol. 9, no. 2, pp. 264-82.

Mengchen, Y 2011, *Cassandra to back applications*, Indiana Univerisity, viewed 27.08.2012,

<http://salsahpc.indiana.edu/b534projects/sites/default/files/public/1_cassandra%20to%20back%20applications_Yu,%20Mengchen.pdf>.

Mitchell, T & Tucker, D 2017, *Couchbase and Apache Kafka – Bridging the gap between RDBMS and NoSQL*, <http://www.dataversity.net/slides-couchbase-apache-kafka-bridging-gap-rdbms-nosql/>.

Moniruzzaman, A & Hossain, SA 2013, 'Nosql database: New era of databases for big data analytics-classification, characteristics and comparison', *International Journal of Database Theory and Application*, vol. 6, no. 4, pp. 1-14.

Murphy, G, Hanken, MA & Waters, K 1999, *Electronic health records: Changing the vision*, Saunders WB Co.

Myers, MD & Klein, HK 2011, 'A set of principles for conducting critical research in information systems', *MIS quarterly*, pp. 17-36.

Nance, C, Losser, T, Iype, R & Harmon, G 2013, 'Nosql vs rdbms-why there is room for both', *proceedings of the Southern Association for Information Systems Conference* Savannah, GA, USA.

Narayan, S, Gagne, M & Safavi-Naini, R 2010, 'Privacy preserving EHR system using attribute-based infrastructure', *proceedings of the 2010 ACM workshop on Cloud computing security workshop* ACM, Chicago, Illinois, USA, pp. 47-52.

NHS 1998, *Information for health: an information strategy for the modern NHS 1998-2005: a national strategy for local implementation*, NHS Executive.

Niehaves, B 2007, 'On epistemological diversity in design science: New vistas for a design-oriented is research', *proceedings of the Twenty-Eighth International Conference on Information Systems, Montreal* Citeseer.

Nøhr, C, Andersen, SK, Vingtoft, S, Bernstein, K & Bruun-Rasmussen, M 2005, 'Development, implementation and diffusion of EHR systems in Denmark', *International journal of medical informatics*, vol. 74, no. 2, pp. 229-34.

Oliveira, MGd, Alves, ALF, Leite, DFB, Rocha, JH, Filho, JAMA & Baptista, CdS 2013, 'Introducing spatial context in comparative pricing and product search', *proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems* ACM, Luxembourg, Luxembourg, pp. 127-34.

Oracle 2011, *MySQL Reference Architectures for Massively Scalable Web Infrastructure,*
MySQL Best Practices for Innovating on the Web, viewed 1 July 2016, <http://www.oracle.com/us/products/mysql/wp-high-availability-webrefarchs-362556.pdf>.

Oracle 2016, *MySQL Customer: eClinicalWorks*, viewed 10 July 2016, <https://www.mysql.com/customers/view/?id=803>.

Orfanidis, L, Bamidis, PD & Eaglestone, B 2004, 'Data quality issues in electronic health records: an adaptation framework for the Greek health system', *Health informatics journal*, vol. 10, no. 1, pp. 23-36.

Papadimitriou, CH 1979, 'The serializability of concurrent database updates', *J. ACM*, vol. 26, no. 4, pp. 631-53.

Parker, Z, Poe, S & Vrbsky, SV 2013, 'Comparing NoSQL MongoDB to an SQL DB', *proceedings of the 51st ACM Southeast Conference* ACM, Savannah, Georgia, pp. 1-6.

Pearce, C & Haikerwal, MC 2010, 'E-health in Australia: time to plunge into the 21st century', *Med J Aust*, vol. 193, no. 7, pp. 397-8.

Pearl, RM 2017, *What Health Systems, Hospitals, and Physicians Need to Know About Implementing Electronic Health Records*, <https://hbr.org/2017/06/what-health-systems-hospitals-and-physicians-need-to-know-about-implementing-electronic-health-records>.

Peffers, K, Tuunanen, T, Rothenberger, MA & Chatterjee, S 2007, 'A Design Science Research Methodology for Information Systems Research', *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45-77.

Phanishayee, A, Andersen, DG, Pucha, H, Povzner, A & Belluomini, W 2012, 'Flex-KV: Enabling High-performance and Flexible KV Systems', *proceedings of the First Workshop on Management of Big Data Systems* San Jose, CA.

Pokorny, J 2011, 'NoSQL databases: a step to database scalability in web environment', *proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services* ACM, Ho Chi Minh City, Vietnam, pp. 278-83.

Pussewalage, HSG & Oleshchuk, VA 2016, 'An attribute based access control scheme for secure sharing of electronic health records', *proceedings of the 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)* pp. 1-6.

Raghupathi, W & Raghupathi, V 2014, 'Big data analytics in healthcare: promise and potential', *Health Information Science and Systems*, vol. 2, p. 3, PMC, item: PMC4341817.

Rossi, M & Sein, MK 2003, 'Design research workshop: a proactive research approach', *Presentation delivered at IRIS*, vol. 26, pp. 9-12.

Ruan, G, Zhang, H & Plale, B 2013, 'Exploiting MapReduce and data compression for data-intensive applications', *proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery* ACM, San Diego, California, pp. 1-8.

Sadalage, P, 2014, 'NoSQL Databases: An Overview', viewed 2 October 2014, <https://www.thoughtworks.com/insights/blog/nosql-databases-overview>.

Sadalage, PJ & Fowler, M 2012, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*, Addison-Wesley Professional.

Sattar, A, Lorenzen, T & Nallamaddi, K 2013, 'Incorporating NoSQL into a database course', *ACM Inroads*, vol. 4, no. 2, pp. 50-3.

Schiff, GD, Klass, D, Peterson, J, Shah, G & Bates, DW 2003, 'Linking laboratory and pharmacy: opportunities for reducing errors and improving care', *Archives of Internal Medicine*, vol. 163, no. 8, pp. 893-900.

Schmidt, LF 2001, *System and method for enhanced performance of a relational database management system through the use of application-specific memory-resident data*, Google Patents, <https://www.google.com/patents/US6304867>.

Schmitt, O & Majchrzak, TA 2012, 'Using Document-Based Databases for Medical In-formation Systems in Unreliable Environments', *proceedings of the 9th International ISCRAM Conference* Vancouver, Canada.

Schram, A & Anderson, KM 2012, 'MySQL to NoSQL: data modeling challenges in supporting scalability', *proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity* ACM, Tucson, Arizona, USA, pp. 191-202.

Segleau, D 2016, *NoSQL Data Modeling Using JSON Documents – A Practical Approach*, <https://www.slideshare.net/Dataversity/slides-nosql-data-modeling-using-json-documents-a-practical-approach>.

Shi, Y, Meng, X, Zhao, J, Hu, X, Liu, B & Wang, H 2010, 'Benchmarking cloud-based data management systems', *proceedings of the Second International Workshop on Cloud Data Management* ACM, Toronto, ON, Canada, pp. 47-54.

Siegrist, K 1997, *The Multinomial Distribution*, University of Alabama, Department of Mathematical Sciences, <http://www.math.uah.edu/stat/bernoulli/Multinomial.html>.

SolidIT 2016, *DB-Engines Ranking of Relational DBMS*, viewed 1 August 2016, <http://db-engines.com/en/ranking>.

Souley, B & Mohammed, D 2013, 'Performance analysis of query optimizers under varying hardware components in rdbms', *Journal of Computer Engineering & Information Technology*, vol. 2, no. 3.

Stonebraker, M & Cattell, R 2011, '10 rules for scalable performance in 'simple operation' datastores', *Commun. ACM*, vol. 54, no. 6, pp. 72-80.

Straub, D, Boudreau, M-C & Gefen, D 2004, 'Validation guidelines for IS positivist research', *The Communications of the Association for Information Systems*, vol. 13, no. 1, p. 63.

Suciu, D 2001, 'On database theory and XML', *SIGMOD Rec.*, vol. 30, no. 3, pp. 39-45.

Sumbaly, R, Kreps, J, Gao, L, Feinberg, A, Soman, C & Shah, S 2012, 'Serving large-scale batch computed data with project Voldemort', *proceedings of the 10th USENIX conference on File and Storage Technologies* USENIX Association, San Jose, CA, pp. 18-.

Sun, J & Reddy, CK 2013, 'Big data analytics for healthcare', *proceedings of the Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* ACM, pp. 1525-.

Sundvall, E, Wei-Kleiner, F, Freire, SM & Lambrix, P 2017, 'Querying Archetype-Based Electronic Health Records Using Hadoop and Dewey Encoding of openEHR Models', *Stud Health Technol Inform*, vol. 235, pp. 406-10, NLM, item: 28423824.

Swaroop, P & Vijit Gupta, K 2016, 'NoSQL Paradigm and Performance Evaluation', *Scientific Society of Advanced Research and Social Change*, vol. 3.

Takeda, H, Matsumura, Y, Kuwata, S, Nakano, H, Sakamoto, N & Yamamoto, R 2000, 'Architecture for networked electronic patient record systems', *International journal of medical informatics*, vol. 60, no. 2, pp. 161-7.

Thanopoulou, A, Carreira, P & Galhardas, H 2012, 'Benchmarking with TPC-H on off-the-shelf Hardware', *ICEIS (1)*, pp. 205-8.

Ullman, JD 1987, 'Database theory—past and future', *proceedings of the Sixth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems* ACM, pp. 1-10.

Valduriez, P 2011, 'Principles of distributed data management in 2020?', *proceedings of the Database and Expert Systems Applications* Springer, pp. 1-11.

van der Linden, H, Kalra, D, Hasman, A & Talmon, J 2009, 'Inter-organizational future proof EHR systems: A review of the security and privacy related issues', *International journal of medical informatics*, vol. 78, no. 3, pp. 141-60.

van Ginneken, AM 2002, 'The computerized patient record: balancing effort and benefit', *International journal of medical informatics*, vol. 65, no. 2, pp. 97-119.

Venable, J, Pries-Heje, J & Baskerville, R 2012, 'A comprehensive framework for evaluation in design science research', *proceedings of the International Conference on Design Science Research in Information Systems* Springer, pp. 423-38.

Vera, H, Wagner Boaventura, MH, Guimaraes, V & Hondo, F 2015, 'Data Modeling for NoSQL Document-Oriented Databases', *proceedings of the CEUR Workshop* pp. 129-35.

Vest, JR 2012, 'Health Information Exchange: National and International Approaches', *Advances in health care management*, vol. 12, p. 3.

Vianu, V 2001, 'A web odyssey: From Codd to XML', *proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* ACM, pp. 1-15.

Victora, CG, Habicht, JP & Bryce, J 2004, 'Evidence-based public health: moving beyond randomized trials', *Am J Public Health*, vol. 94, no. 3, pp. 400-5, NLM, item: 14998803.

Vohra, D 2015, *Pro Couchbase Development: A NoSQL Platform for the Enterprise*, Apress.

Walker, J, Pan, E, Johnston, D, Adler-Milstein, J, Bates, DW & Middleton, B 2005, 'The value of health care information exchange and interoperability', *HEALTH AFFAIRS-MILLWOOD VA THEN BETHESDA MA-*, vol. 24, p. W5.

Walls, JG, Widmeyer, GR & El Sawy, OA 1992, 'Building an information system design theory for vigilant EIS', *Information systems research*, vol. 3, no. 1, pp. 36-59.

Weber, R 2004, 'Editor's comments: the rhetoric of positivism versus interpretivism: a personal view', *MIS quarterly*, vol. 28, no. 1, pp. iii-xii.

Wu, J 2011, *Decentralized storage system-Cassandra*, <http://salsahpc.indiana.edu/b534projects/sites/default/files/public/1_Apache%20Cassandra_Wu,%20Jiang.pdf>.

Yasnoff, WA, Humphreys, BL, Overhage, JM, Detmer, DE, Brennan, PF, Morris, RW, Middleton, B, Bates, DW & Fanning, JP 2004, 'A consensus action agenda for achieving the national health information infrastructure', *Journal of the American Medical Informatics Association*, vol. 11, no. 4, pp. 332-8.

Yassien, AW & Desouky, AF 2016, 'RDBMS, NoSQL, Hadoop: A Performance-Based Empirical Analysis', *proceedings of the 2nd Africa and Middle East Conference on Software Engineering* ACM, Cairo, Egypt, pp. 52-9.

Zhang, P, Scialdone, M & Ku, M-C 2011, 'IT artifacts and the state of IS research', *proceedings of the International Conference on Information Systems* Shanghai.

Zhang, R & Liu, L 2010, 'Security Models and Requirements for Healthcare Application Clouds', *proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing* pp. 268-75.

Zhu, Q, 2016, 'YCSB Benchmark - Couchbase + Tegile/Cisco UCS', viewed 19 July 2016, <https://blog.couchbase.com/2016/july/ycsb-benchmark--couchbase-tegile>.

List of Appendices

Appendix A. Separation statistics, public and private hospitals, states and territories, 2014–15

Appendix B. Separations, by state or territory of usual residence and establishments, 2014–15

Appendix C. Separations per 1,000 population, public and private hospitals, states and territories, 2014–15

Appendix D. Same-day and overnight separations per 1,000 population, states and territories, 2014–15

Appendix E. Separations by mode of admission, public and private hospitals, states and territories, 2014–15

Appendix F. Admitted Patient Care National Minimum Dataset Details

Appendix G. JSON representation of aggregate oriented data model

Appendix A. Separation statistics, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016))

	NSW	Vic	Qld	WA	SA	Tas	ACT	NT	Total
Separations									
Public hospitals									
Public acute hospitals	1,808,679	1,587,510	1,202,496	599,474	420,870	118,419	100,784	132,283	5,970,515
Public psychiatric hospitals	5,319	441	302	1,249	1,425	1,087	. .	. .	9,823
Total public hospitals	*1,813,998*	*1,587,951*	*1,202,798*	*600,723*	*422,295*	*119,506*	*100,784*	*132,283*	*5,980,338*
Private hospitals									
Private free-standing day hospital facilities	254,859	223,434	228,431	143,825	76,091	n.p.	n.p.	n.p.	940,703
Other private hospitals	929,680	785,903	804,526	336,915	239,765	n.p.	n.p.	n.p.	3,229,326
Total private hospitals	*1,184,539*	*1,009,337*	*1,032,957*	*480,740*	*315,856*	n.p.	n.p.	n.p.	*4,170,029*
Public acute and private hospitals	*2,993,218*	*2,596,847*	*2,235,453*	*1,080,214*	*736,726*	n.p.	n.p.	n.p.	*10,140,544*
All hospitals	**2,998,537**	**2,597,288**	**2,235,755**	**1,081,463**	**738,151**	**n.p.**	**n.p.**	**n.p.**	**10,150,367**
Overnight separations									
Public hospitals									
Public acute hospitals	978,234	671,847	558,108	275,409	221,688	54,875	47,316	41,243	2,848,720
Public psychiatric hospitals	5,118	439	302	1,237	1,044	1,071	. .	. .	9,211
Total public hospitals	*983,352*	*672,286*	*558,410*	*276,646*	*222,732*	*55,946*	*47,316*	*41,243*	*2,857,931*
Private hospitals									
Private free-standing day hospital facilities	69	4	0	1,813	0	n.p.	n.p.	n.p.	1,886
Other private hospitals	310,023	332,192	316,233	139,249	91,852	n.p.	n.p.	n.p.	1,240,837
Total private hospitals	*310,092*	*332,196*	*316,233*	*141,062*	*91,852*	n.p.	n.p.	n.p.	*1,242,723*
Public acute and private hospitals	*1,288,326*	*1,004,043*	*874,341*	*416,471*	*313,540*	n.p.	n.p.	n.p.	*4,091,443*
All hospitals	**1,293,444**	**1,004,482**	**874,643**	**417,708**	**314,584**	**n.p.**	**n.p.**	**n.p.**	**4,100,654**
Same-day separations									
Public hospitals									
Public acute hospitals	830,445	915,663	644,388	324,065	199,182	63,544	53,468	91,040	3,121,795
Public psychiatric hospitals	201	2	0	12	381	16	. .	. .	612

Total public hospitals	*830,646*	*915,665*	*644,388*	*324,077*	*199,563*	*63,560*	*53,468*	*91,040*	*3,122,407*
Private hospitals									
Private free-standing day hospital facilities	254,790	223,430	228,431	142,012	76,091	n.p.	n.p.	n.p.	938,817
Other private hospitals	619,657	453,711	488,293	197,666	147,913	n.p.	n.p.	n.p.	1,988,489
Total private hospitals	*874,447*	*677,141*	*716,724*	*339,678*	*224,004*	n.p.	n.p.	n.p.	*2,927,306*
Public acute and private hospitals	*1,704,892*	*1,592,804*	*1,361,112*	*663,743*	*423,186*	n.p.	n.p.	n.p.	*6,049,101*
All hospitals	**1,705,093**	**1,592,806**	**1,361,112**	**663,755**	**423,567**	**n.p.**	**n.p.**	**n.p.**	**6,049,713**
Same-day separations as % of total									
Public hospitals									
Public acute hospitals	45.9	57.7	53.6	54.1	47.3	53.7	53.1	68.8	52.3
Public psychiatric hospitals	3.8	0.5	0.0	1.0	26.7	1.5	. .	. .	6.2
Total public hospitals	*45.8*	*57.7*	*53.6*	*53.9*	*47.3*	*53.2*	*53.1*	*68.8*	*52.2*
Private hospitals									
Private free-standing day hospital facilities	100.0	100.0	100.0	98.7	100.0	n.p.	n.p.	n.p.	99.8
Other private hospitals	66.7	57.7	60.7	58.7	61.7	n.p.	n.p.	n.p.	61.6
Total private hospitals	*73.8*	*67.1*	*69.4*	*70.7*	*70.9*	n.p.	n.p.	n.p.	*70.2*
Public acute and private hospitals	*57.0*	*61.3*	*60.9*	*61.4*	*57.4*	n.p.	n.p.	n.p.	*59.7*
All hospitals	**56.9**	**61.3**	**60.9**	**61.4**	**57.4**	**n.p.**	**n.p.**	**n.p.**	**59.6**

Appendix B. Separations, by state or territory of usual residence and establishments, 2014–15 (Adopted from (AIHW 2016))

	State or territory of hospitalisation									
State or territory of usual residence	**NSW**	**Vic**	**Qld**	**WA**	**SA**	**Tas**	**ACT**	**NT**	**Total**	**Separations per 1,000 population**
Public hospitals										
New South Wales	1,781,294	32,744	12,460	682	1,825	252	17,940	395	1,847,592	226.7
Victoria	4,145	1,541,375	3,038	731	2,080	378	317	372	1,552,436	249.1
Queensland	12,113	1,689	1,177,069	678	494	263	203	735	1,193,244	244.7
Western Australia	622	688	795	594,432	334	72	43	3,462	600,448	230.9
South Australia	771	2,317	631	333	414,846	70	57	3,014	422,039	225.4
Tasmania	308	2,226	393	114	72	118,318	19	47	121,497	212.1
Australian Capital Territory	3,537	260	245	30	58	28	81,717	25	85,900	227.8
Northern Territory	241	348	552	220	1,883	8	6	123,926	127,184	572.9
Other Australian territories[(a)]	1,230	1,590	0	293	0	0	0	3	3,116	n.p.
Not elsewhere classified/Not reported[(b)]	9,737	4,714	7,615	3,210	703	117	482	304	26,882	. .
Total public hospitals	*1,813,998*	*1,587,951*	*1,202,798*	*600,723*	*422,295*	*119,506*	*100,784*	*132,283*	*5,980,338*	*240.2*
Private hospitals										
New South Wales	1,160,016	10,285	37,252	228	1,828	n.p.	n.p.	n.p.	1,218,177	146.8
Victoria	8,583	993,317	1,693	226	1,628	n.p.	n.p.	n.p.	1,005,689	159.6
Queensland	4,384	1,197	991,070	386	260	n.p.	n.p.	n.p.	997,487	200.4
Western Australia	467	622	397	479,387	109	n.p.	n.p.	n.p.	481,100	184
South Australia	264	713	394	124	310,367	n.p.	n.p.	n.p.	312,097	157.7
Tasmania	300	1,908	362	46	70	n.p.	n.p.	n.p.	91,041	152.7
Australian Capital Territory	2,700	231	288	19	42	n.p.	n.p.	n.p.	38,517	102.4
Northern Territory	373	479	804	151	1,296	n.p.	n.p.	n.p.	16,721	79.7
Other Australian territories[(a)]	6,531	25	0	134	0	n.p.	n.p.	n.p.	6,690	n.p.
Not elsewhere classified/Not reported[(b)]	921	560	697	39	256	n.p.	n.p.	n.p.	2,510	. .
Total private hospitals	*1,184,539*	*1,009,337*	*1,032,957*	*480,740*	*315,856*	n.p.	n.p.	n.p.	*4,170,029*	*164.4*
All hospitals	**2,998,537**	**2,597,288**	**2,235,755**	**1,081,463**	**738,151**	**n.p.**	**n.p.**	**n.p.**	**10,150,367**	**404.6**

(a) Includes Cocos (Keeling) Islands, Christmas Island and Jervis Bay Territory.

(b) Includes Resident overseas, At sea and No fixed address.

Appendix C. Separations per 1,000 population, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016))

	NSW	Vic	Qld	WA	SA	Tas	ACT	NT	Total
Public hospitals									
Public acute hospitals	221.9	254.8	246.6	230.5	224.8	206.1	267.2	598.0	239.8
Public psychiatric hospitals	0.7	0.1	0.1	0.5	0.9	2.1	. .	. .	0.4
Total public hospitals	*222.6*	*254.9*	*246.7*	*231.0*	*225.6*	*208.3*	*267.2*	*598.0*	*240.2*
Private hospitals									
Private free-standing day hospital facilities	30.8	35.5	45.6	55.3	36.8	n.p.	n.p.	n.p.	37.0
Other private hospitals	112.2	124.6	161.8	128.5	122.9	n.p.	n.p.	n.p.	127.4
Total private hospitals	*143.0*	*160.2*	*207.4*	*183.9*	*159.7*	n.p.	n.p.	n.p.	*164.4*
All hospitals	*365.5*	*415.0*	*454.0*	*414.9*	*385.3*	n.p.	n.p.	n.p.	*404.6*

Appendix D. Same-day and overnight separations per 1,000 population, states and territories, 2014–15 (Adopted from (AIHW 2016))

Same-day separations

	NSW	Vic	Qld	WA	SA	Tas	ACT	NT	Total
Public hospitals	101.5	147.0	131.7	124.6	106.6	108.2	143.1	408.3	125.1
Private hospitals	105.2	108.3	143.7	130.0	112.9	n.p.	n.p.	n.p.	115.5
All hospitals	**206.7**	**255.3**	**275.4**	**254.6**	**219.5**	**n.p.**	**n.p.**	**n.p.**	**240.6**

Overnight separations

	NSW	Vic	Qld	WA	SA	Tas	ACT	NT	Total
Public hospitals	121.1	107.9	115.0	106.4	119.0	100.1	124.1	189.7	115.1
Private hospitals	37.7	51.9	63.6	53.9	46.8	n.p.	n.p.	n.p.	48.9
All hospitals	**158.9**	**159.8**	**178.6**	**160.3**	**165.8**	**n.p.**	**n.p.**	**n.p.**	**164.0**

Appendix E. Separations by mode of admission, public and private hospitals, states and territories, 2014–15 (Adopted from (AIHW 2016))

	NSW	Vic	Qld	WA	SA	Tas	ACT	NT	Total
Public hospitals									
New admission to hospital[(a)]	1,673,721	1,496,384	1,135,367	555,553	396,114	113,267	94,519	130,465	5,595,390
Admitted patient transferred from another hospital	96,949	75,239	42,316	37,559	20,886	3,157	3,439	196	279,741
Statistical admission: care type change	35,065	15,849	25,115	7,611	4,288	1,975	2,826	1,622	94,351
Not reported	8,263	479	0	0	1,007	1,107	0	0	10,856
Total public hospitals	*1,813,998*	*1,587,951*	*1,202,798*	*600,723*	*422,295*	*119,506*	*100,784*	*132,283*	*5,980,338*
Private hospitals									
New admission to hospital[(a)]	1,134,239	970,013	998,018	469,438	308,854	n.p.	n.p.	n.p.	4,001,858
Admitted patient transferred from another hospital	43,556	35,171	25,686	8,438	6,337	n.p.	n.p.	n.p.	123,647
Statistical admission: care type change	5,412	4,153	9,253	2,864	533	n.p.	n.p.	n.p.	23,646
Not reported	1,332	0	0	0	132	n.p.	n.p.	n.p.	20,878
Total private hospitals	*1,184,539*	*1,009,337*	*1,032,957*	*480,740*	*315,856*	*n.p.*	*n.p.*	*n.p.*	*4,170,029*
All hospitals									
New admission to hospital[(a)]	2,807,960	2,466,397	2,133,385	1,024,991	704,968	n.p.	n.p.	n.p.	9,597,248
Admitted patient transferred from another hospital	140,505	110,410	68,002	45,997	27,223	n.p.	n.p.	n.p.	403,388
Statistical admission: care type change	40,477	20,002	34,368	10,475	4,821	n.p.	n.p.	n.p.	117,997
Not reported	9,595	479	0	0	1,139	n.p.	n.p.	n.p.	31,734
Total	**2,998,537**	**2,597,288**	**2,235,755**	**1,081,463**	**738,151**	**n.p.**	**n.p.**	**n.p.**	**10,150,367**

(a) *New admission to hospital* is equivalent to *Other* in the mode of admission definition. It refers to all planned and unplanned admissions except transfers from other hospitals and statistical admissions.

Appendix F. Admitted Patient Care National Minimum Dataset Details (Adopted from (AIHW 2015))

Admitted patient care NMDS 2014-15

Metadata item type: Data Set Specification

METeOR identifier: 535047

Registration status: Health, Standard 11/04/2014

DSS type: National Minimum Data Set (NMDS)

Scope: The purpose of the Admitted patient care national minimum data set (APC NMDS) is to collect information about care provided to admitted patients in Australian hospitals.

The scope of the APC NMDS is episodes of care for admitted patients in all public and private acute and psychiatric hospitals, free standing day hospital facilities and alcohol and drug treatment centres in Australia. Hospitals operated by the Australian Defence Force, corrections authorities and in Australia's off-shore territories may also be included. Hospitals specialising in dental, ophthalmic aids and other specialised acute medical or surgical care are included.

Hospital boarders and still births are not included as they are not admitted to hospital. Posthumous organ procurement episodes are also not included.

Collection and usage attributes

Statistical unit: Episodes of care for admitted patients

Collection methods: Data are collected at each hospital from patient administrative and clinical record systems. Hospitals forward data to the relevant state or territory health authority on a regular basis (e.g. monthly).

National reporting arrangements

State and territory health authorities provide the data to the Australian Institute of Health and Welfare for national collation, on an annual basis.

Metadata items in this Data Set Specification

Metadata item	Obligation	Max Occurs
Elective surgery waiting times cluster	Conditional	99
Activity when injured	Mandatory	99
Additional diagnosis	Conditional	99
Admission date	Mandatory	1
Admitted patient election status	Mandatory	1
Area of usual residence (SA2)	Mandatory	1
Australian postcode (address)	Mandatory	1
Australian State/Territory identifier (establishment)	Mandatory	1
Care type	Mandatory	1
Condition onset flag	Mandatory	99
Contract establishment identifier	Mandatory	1
Country of birth	Mandatory	1
Date of birth	Mandatory	1
Duration of continuous ventilatory support	Conditional	1
Establishment number	Mandatory	1
Establishment sector	Mandatory	1
External cause	Mandatory	99
Funding source for hospital patient	Mandatory	1
Geographic remoteness—admitted patient care	Mandatory	1
Hospital insurance status	Mandatory	1
Indigenous status	Mandatory	1

Intended length of hospital stay	Mandatory	1
Inter-hospital contracted patient	Mandatory	1
Length of stay in intensive care unit	Conditional	1
Medicare eligibility status	Mandatory	1
Mental health legal status	Mandatory	1
Mode of admission	Mandatory	1
Mode of separation	Mandatory	1
Number of days of hospital-in-the-home care	Mandatory	1
Number of qualified days for newborns	Conditional	1
Person identifier	Mandatory	1
Place of occurrence of external cause of injury (ICD-10-AM)	Mandatory	99
Principal diagnosis—episode of care	Mandatory	1
Procedure	Mandatory	99
Record identifier (80 character maximum)	Mandatory	1
Region code	Mandatory	1
Separation date	Mandatory	1
Sex	Mandatory	1
Source of referral to public psychiatric hospital	Conditional	1
Total leave days	Mandatory	1
Total psychiatric care days	Mandatory	1
Urgency of admission	Mandatory	1
Weight in grams (measured)	Conditional	1

Appendix G. JSON representation of aggregate oriented data model


```
{
	"Person": {
		"Person identifier": "123456789",
		"Area of usual residence": {
			"METeOR identifier": "469909",
			"code": "31701144631446",
			"value": "Darling Heights"
		},
		"Country of birth": {
			"METeOR identifier": "459973",
			"code": "5101",
			"value": "Myanmar"
		},
		"Date of birth": "01012000",
		"Indigenous status": {
			"METeOR identifier": "291036",
			"code": "4",
			"value": "Neither Aboriginal nor Torres Strait Islander origin"
		},
		"Sex": {
			"METeOR identifier": "287316",
			"code": "1",
			"value": "Male"
		},
		"Medicare Eligibility status": {
			"METeOR identifier": "481841",
			"code": "1",
			"value": "Eligible"
		},
		"Address": "",
		"Record—identifier": "abcd-1234"
	},
	"Emergency Department Stay": {
		"Physical departure date": "01082016",
		"Physical departure time": "1120",
		"Presentation date": "01082016",
		"Presentation time": "1000",
		"Transport mode (arrival)": "",
		"Type of visit": "",
		"Urgency related group major diagnostic block": ""
	},
	"Patient": {
		"Compensable status": "",
		"Hospital insurance status": {
			"METeOR identifier": "270253",
			"code": "9",
			"value": "Unknown"
		}
```

```
	},
	"Episode of admitted patient care": {
		"Admission date": "01082016",
		"Admission mode": {
			"METeOR identifier": "269976",
			"code": "3",
			"value": "Other"
		},
		"Admission urgency status": "",
		"Condition onset flag": "",
		"Intended length of hospital stay": "2",
		"Number of days of hospital-in-the-home care,": "0",
		"Number of leave days": "0",
		"Patient election status": {
			"METeOR identifier": "326619",
			"code": "1",
			"value": "Public"
		},
		"Procedure": [],
		"Separation date": "02082016",
		"Separation mode": {
			"METeOR identifier": "270094",
			"code": "9",
			"value": "Other (includes discharge to usual residence, own
accommodation/welfare institution (includes prisons, hostels and group homes
providing primarily welfare services))"
		}
	},
	"Episode of care": {
		"Inter-hospital contracted patient status": "",
		"Mental health legal status": "",
		"Number of psychiatric care days": "1",
		"Principal diagnosis": {
			"METeOR identifier": "514273",
			"code": "V00",
			"value": "Pedestrian conveyance accident"
		},
		"Source of funding, patient funding source": "",
		"Funding eligibility indicator": ""
	},
	"Establishment": {
		"Australian state/territory identifier": {
			"METeOR identifier": "269941",
			"code": "3",
			"value": "Queensland"
		},
		"Geographic remoteness": {
			"METeOR identifier": "539871",
			"code": "1",
			"value": "Inner regional Australia"
```

```
		},
		"Organisation identifier (state/territory)": "",
		"Region identifier": "",
		"Sector": {
			"METeOR identifier": "269977",
			"code": "1",
			"value": "Public"
		},
		"Organisation identifier": "12345"
	},
	"Injury Event": {
		"Activity type": {
			"METeOR identifier": "514277",
			"code": "V00",
			"value": "Pedestrian conveyance accident"
		},
		"External cause": {
			"METeOR identifier": "514295",
			"code": "Y93.9",
			"value": "Activity, unspecified"
		},
		"Place of occurrence": {
			"METeOR identifier": "514302",
			"code": "Y92.41",
			"value": "Street and highway as the place of occurrence of the
external cause"
		}
	},
	"Non-admitted patient service": {
		"Episode end date": "01082016",
		"Episode end status": "",
		"Episode end time": "1220"
	},
	"Hospital service": {
		"Care type": {
			"METeOR identifier": "491557",
			"code": "1",
			"value": "Acute care"
		}
	}
}
```