Dissertation Project

A Comparison of NoSQL and Indexing Solutions for Big Data

Author: Callum George William Guthrie

Supervisor: Dr. Albert Burger

Heriot Watt University

Edinburgh, Scotland

A dissertation submitted in partial fulfilment of the requirements for the degree of Bachelor of Science.

May 2016

Contents

Introduction

The era of Big Data is upon us bringing with it a range of new challenges “Without big data, you are blind and deaf and in the middle of a freeway.”[3] . The importance which accompany these challenges encouraged the formulation of new approaches for cleaning, processing and using enormous amounts of data. (Section 1.1)

Data is being collated and stored every second of every day and the value of doing so has never been greater. Billion dollar companies such as Google and Amazon dominate the market in data collection and pride themselves in knowing everything about everything. Former CEO of Google, Eric Schmidt famously said in 2010 “We know where you are. We know where you’ve been. We can more or less know what you’re thinking…” [6]. Thus the power of data collection has led to the development of a range of technologies designed to meet the needs of Big Data.

The purpose of the following research, by way of investigation, is to deliver an insightful examination of a subset of new technologies which deliver high performance querying of large datasets. The ultimate aim of the research is to gain an understanding of these technologies and achieve a level of mastery which permits a thorough scrutiny of their application to Big Data.

There are a number of different indexing solutions available however for the means of this project to encapsulate a comprehensive examination a focus will be on leading NoSQL solutions, modern search and analytics engines and for comparative reasons a conventional relational database management system. The following technologies which will be used for the project:

- MongoDB (Section 1.6.1)
- Neo4j (Section 1.6.2)
- Apache Cassandra (Section 1.6.3)

- MySQL (Section 1.6.4)

0.1 Objectives

The three key objectives and main intended outcomes for the project are:

1. Investigate the strengths and weaknesses of the functionality each technology provides.
2. Compare and contrast the analytical capabilities of each technology by way querying prototype models.
3. Conduct a comparative analysis to investigate the scalability of each technology.

0.2 Motivation

My interest in the field of data science stems from university modules I have undertaken as part of my BSc Computer Science degree such as 'Database Management Systems' and a course I am currently studying 'Data Mining and Machine Learning'. The material involved in these courses gave me an insight in to the field of data science and provided me with the opportunity to get a hands on feel for the manipulation, cleansing and processing of a variety of real life data sets and database systems.

Whilst studying for my degree I have undertaken modules which have stipulated a working knowledge of MySQL as a prerequisite therefore my comprehension of MySQL is proficient. One of the main attractions to undertaking this project was to be given the opportunity to learn about a number of next generation database management systems. It was important for me to undertake a project in which I will be able to apply my learning and findings to progress in a career path within the data science field.

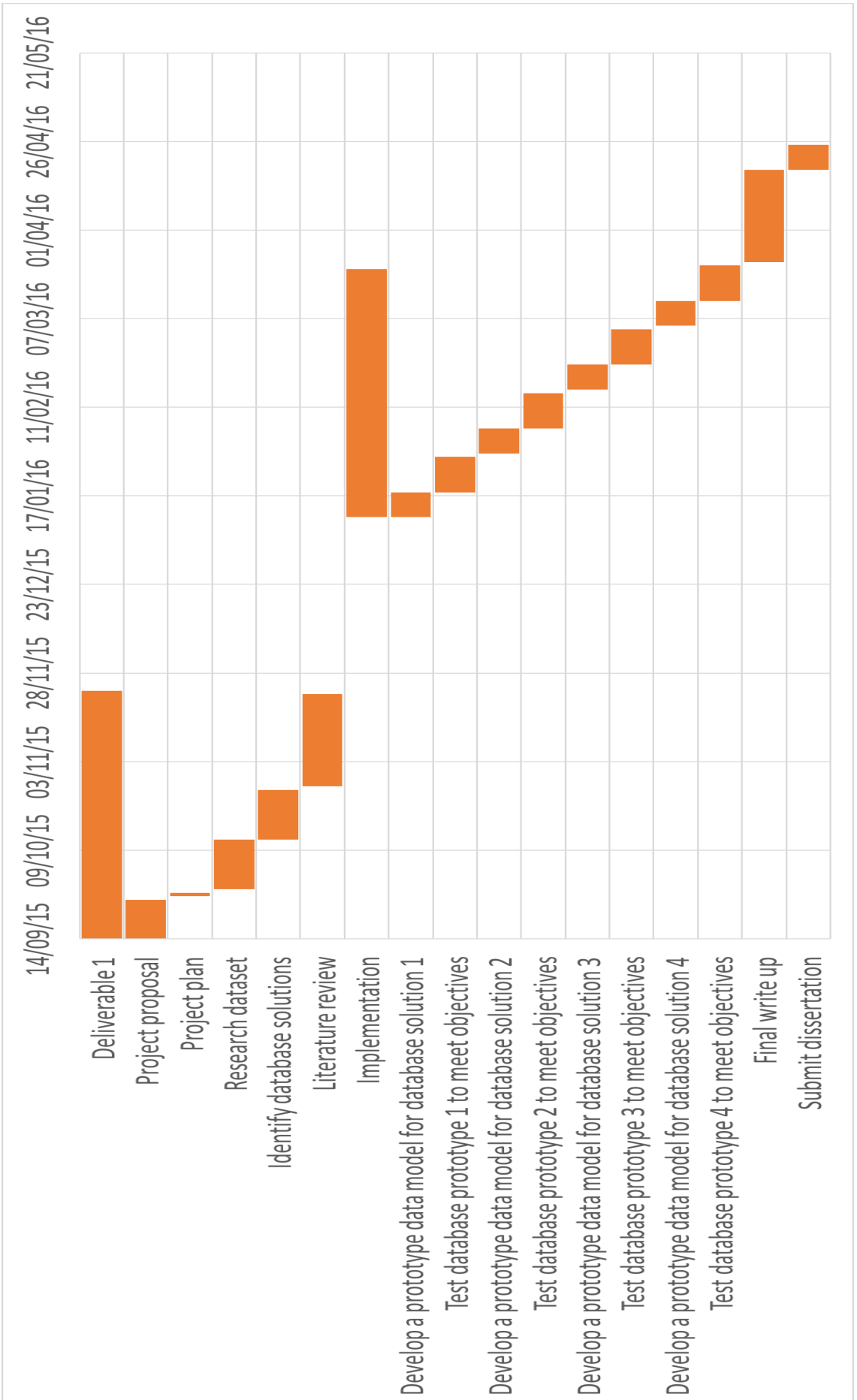
0.3 Definitions

The data source being used in this document comes from the biological field and therefore relies on an apprehension of basic concepts and terms. The below table of definitions provides an overview in to some of these terms to develop a level of understanding and aid the reader of the document. The table also includes the definitions of generally unknown terms and phrases which will be discussed throughout this project.

Term	**Definition**
Edinburgh Mouse Atlas Project (EMAP)	The combined research projects of Dr Duncan Davidson and Prof Richard Baldock.
EMAP **anatomy**	A freely available, structured, **stage specific** list of 13,000+ terms that describe visible anatomical structures in the developing mouse embryo.
Edinburgh Mouse Atlas Project Abstract (EMAPA)	A refined and algorithmically developed **non-stage specific** anatomical ontology representation of the EMAP anatomy.
Edinburgh Mouse Atlas of Gene Expression (EMAGE)	A database of in situ gene expression data in the developing mouse embryo
Theiler Stage (TS)	Each stages defines the development of a mouse embryo by a set of organism structure criteria.
Not only SQL (NoSQL)	A non-relational database environment which is useful for very large sets of distributed data. Allows rapid, ad-hoc organisation and analysis of extremely high-volume, disparate data types.
Ontology	Refers to the science of describing the kinds of entities in the world and how they are related.
Web Ontology Language (OWL)	A language representation standard for designing and authoring Web ontologies produced from the World Wide Web Consortium W3C.
OBO	A flat file format ontology representation language

0.4 Project Plan

The below gantt chart illustrates the time scale in which deadlines and deliverables will be met throughout the project.


14/09/15
09/10/15
03/11/15
28/11/15
23/12/15
17/01/16
11/02/16
07/03/16
01/04/16
26/04/16
21/05/16
Deliverable 1
Project proposal
Project plan
Research dataset
Identify database solutions
Literature review
Implementation
Develop a prototype data model for database solution 1
Test database prototype 1 to meet objectives
Develop a prototype data model for database solution 2
Test database prototype 2 to meet objectives
Develop a prototype data model for database solution 3
Test database prototype 3 to meet objectives
Develop a prototype data model for database solution 4
Test database prototype 4 to meet objectives
Final write up
Submit dissertation

Chapter 1

Background Theory

The intended outcome of this literature review is to deliver a thorough explanation of the various technologies used throughout the project and to provide details on any additional subject matter aspects which will aid understanding of the project.

A general summation of Big Data and its application is given in section 1.1 which is followed by a discussion on the pre-processing of datasets discussed in section 1.2. A high level overview of the NoSQL concept is examined in section 1.4. The various types of dataset classification which are utilised in the project are given in section 1.5.

1.1 Big Data

Big Data is a broad, evolving term bound to a complex and powerful application of analytical insight which over recent years has had a variety of definitions. In simplistic terms Big Data can be described as extremely large datasets that may be studied computationally to reveal patterns, trends, and associations for ongoing discovery and analysis.

The 2011 McKinsey Global Institute (MGI), a multinational management consultancy firm, report "Big data: The next frontier for innovation, competition, and productivity" outlines the potential effects big data will have on a number of industries. The report suggests that with the increasing "exponential" growth of data volume that simply recruiting a "few data-orientated managers" will be a temporary fix rather than a lasting solution. MGI suggest that if companies in a variety of sectors such as the healthcare and retail industry were to take advantage of the value which big data brings could see potentially huge returns. "...a retailer using big data to the full could increase its operating margin by more than 60

percent...healthcare were to use big data creatively and effectively to drive efficiency and quality, the sector could create more than $300 billion in value every year" [18]

1.1.1 5vs Model

In 2001, Gartner analyst Doug Laney delivered the original 3vs model which define the challenges and opportunities which have arisen from the increase in data volumes. Laney categorises big data in to three dimensions; Volume, Velocity and Variety, with the increase of each encapsulating the challenges currently faced today of big data management. The

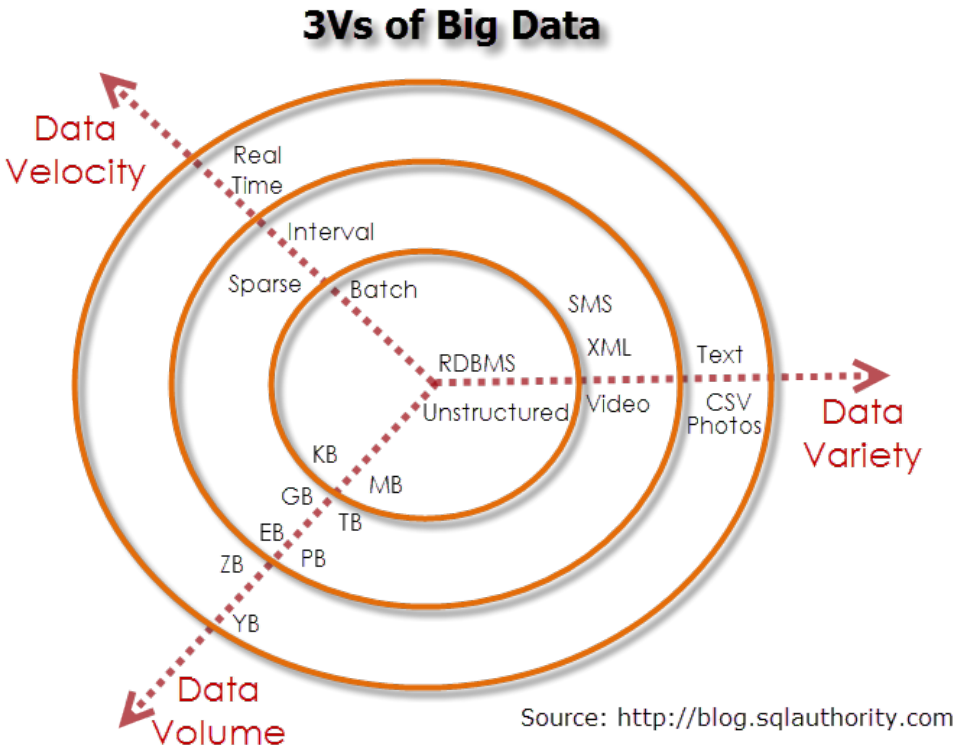


Figure 1.1: 3vs data model

characteristics of each property illustrated in figure 1.1 are defined as: **Volume** - Refers to the vast amounts of data generated every second. With the creation and storage of large quantities of data, the scale of this data becomes progressively vast. **Velocity** - Refers to the speed at which new data is generated and the speed at which data moves around emphasising the "timeliness" of the big data. In order to fully profit from the commercial value of big data, data collection and data examination must be conducted promptly . **Variety** - This characteristic alludes to the various types of data we can now use; semi-structured and unstructured. Examples being "audio, video, webpage and text as well as traditional structured data" [19].

Big Data is a term becoming increasingly common in business and society. Overcoming obstacles and implementing effective, actionable Big Data strategies is key for successful big data management. IBM has introduced a fourth V: **Veracity** Data inconsistencies and incompleteness result in data uncertainty and unreliability. This creates a new challenge; keeping data organised. [19]

The final and considered by many to be the most important V of big data is **Value**. "All the volumes of fast-moving data of different variety and veracity have to be turned into value" [12] One of the biggest challenges faced by organisations is having the ability to turn data into something useful.

The amount of data being produced has dramatically increased from when Laney first introduced the 3vs model in 2001. This is in no small part due to the availability and accessibility of the internet. In 1995 the internet had on average 45 million users, 1% of the worlds population. This figure increased to over 1 billion people with internet access worldwide in 2005, and by 2010 nearly 2 billion which was 30% of the worlds population. The latest figures show that in 2015 the penetration of the internet reached 3 billion people, 40% of the entire population. Social media sites such as Facebook, Twitter, Snapchat, Instagram and Pinterest generate millions of with Facebook boasting 1 million links shared, 2 million friend requests sent and 3 million messages sent on average every twenty minutes [10]. These statistics show the continual growth of internet accessibility as a whole and with this growth comes the challenge of an amalgamation of the benefit of high volumes of captured analytical data and using this data to add value to an organisation.

1.2 Extract Transform Load

This project will require the extraction, manipulation and processing of a data source from one data model to another. This process is commonly known as Extract Transform Load (ETL). Section 1.2.1 discusses each stage involved in the ETL procedure followed by section 1.2.2 which examines the ETL implementation methodology and its relevance to this project.

1.2.1 Process

A basic definition of the Extract Transform Load (ETL) process is pulling data from one database, refactoring the composition of the data and putting the data into another database.

While the name ETL implies there are 3 main categorisation stages - extract, transform, load - the procedure in its entirety is a much broader and expansive process which encompass these stages. Despite this the procedure is split in to these three stages. Figure 1.2 illustrates the ETL process with data coming from a source; a file or database management system for example then being transformed in to the required format for a successful load.

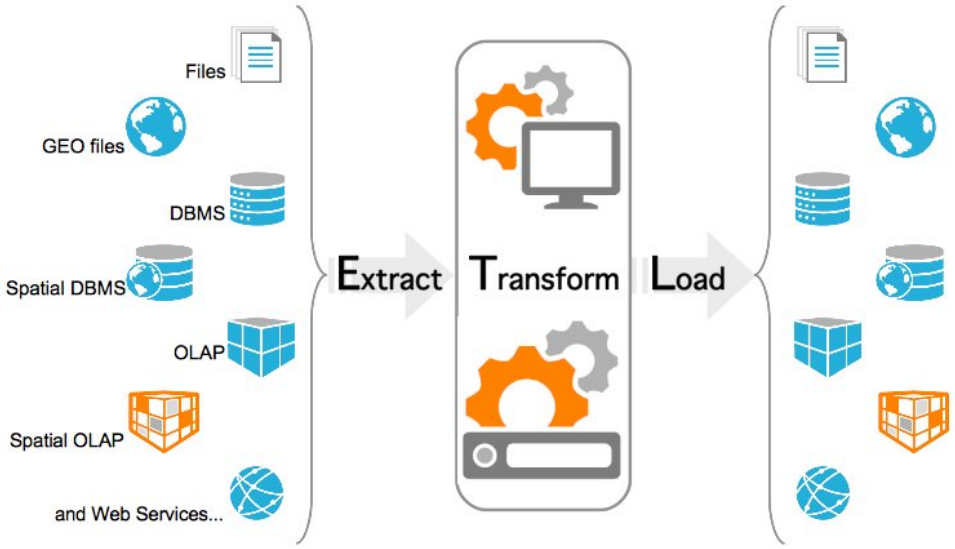


Figure 1.2: ETL process

Extract is the first step in the ETL procedure in which data is read from a source system, usually a database but not restricted to, and makes it available for processing. The main objective of the extract stage is to retrieve all the required data from a source system using as little resources as possible [5]. It is common for data to be extracted from source systems with different organisations and formats to that of the target system. The extract stage provides an opportunity to *cleanse* the data from the source system as often there will be redundant or irrelevant data which is not required.

Transform is where the extracted data is manipulated from its previous state and converted into a target system format. The step involves the application of a set of rules or functions to transform the data from the source to the target. As well as the applied rules and functions the transformation step is responsible for the validation of records ensuring unacceptable records are removed accordingly. “The most common processes used for transformation are conversion, clearing the duplicates, standardizing, filtering, sorting, translating and looking up or verifying if the data sources are inconsistent.” [4].

Load completes the three step procedure and is where data is written into the target

system. There are multiple ways in which data is loaded into a system using the ELT methodology. One of which and the most obvious is to physically insert the data. For example if the target repository is a SQL database insert the data as a new row using the relevant *Insert* statement. An alternative to loading the process manually is that some ETL tool implementations have the capability to "...link the extraction, transformation, and loading processes for each record from the source." [4]. Depending on the technique applied the load step of the process can become the most time consuming.

1.2.2 Tool Implementation

Apache Jena (Jena) is a free and open source Java framework for building Semantic Web - discussed in section 1.3.1 - and Linked Data applications. [2] The Jena framework consists of different APIs interacting together to process Resource Description Framework (RDF) data. I will be using Jena to implement the ETL procedure for the data sources discussed in section **??** as Jena provides an extensive Java library collection for RDF and OWL data representation languages.

W3C defines RDF as "...a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed. RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a "triple"). Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications." [9]

1.3 Knowledge Representation Languages

A Knowledge Representation (KR) language can be defined as: for any given interpretation of a sentence or string of text the KR must have the ability to effectively and unambiguously express knowledge in a both a human and computer manageable form.

There are a number of options and possibilities for communicating data and information which range from binary representation to meta markup languages such as Extensible Markup Language (XML) for example. Markup languages which are easily read by humans such as XML which as a result of its rigid set of rules lends its self to both humans and

machines. Comparatively binary notation which uses 1's and 0's to represent data, while relatively cheap in terms of computing power the ability to comprehend this notation requires a unique and specific skill set.

1.3.1 Semantic Web

The Semantic Web is an extension of the Web through standards by the World Wide Web Consortium (W3C). "The standards promote common data formats and exchange protocols on the Web, most fundamentally the Resource Description Framework (RDF)." [1]

The Semantic Web has two main intended outcomes. The first is about the standardised formats of data pulled from variety of sources, whereas the original Web concentrated on the interchange of documents [1]. The second outcome of the Semantic Web is the language for recording the relation between data and objects in the real world. "That allows a person, or a machine, to start off in one database, and then move through an unending set of databases which are connected not by wires but by being about the same thing." [1].

1.3.2 Web Ontology Language OWL

The Web Ontology Language OWL is a language representation standard for designing and authoring Web ontologies produced from the World Wide Web Consortium W3C. [8]. The OWL file format is designed to be used by applications required to process the content of the information and to be humanly readable. "[OWL] is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications."[8]. The OWL languages are characterised by formal semantics and are built upon the W3C standard RDF format - discussed in section 1.2.2

1.3.3 OBO

The OBO flat file format is an ontology representation language. "The concepts it models represent a subset of the concepts in the OWL description logic language, with several extensions for meta-data modelling and the modelling of concepts that are not supported in DL languages." [22]

[22] outlines the intended outcome of the file format aiming to achieve the following criteria :

- Human readability

- Ease of parsing
- Extensibility
- Minimal redundancy

1.4 NoSQL

NoSQL is labeled as a next generation database known to most as “Not only SQL” [21]. This definition however insinuates its defiance against the industry standard SQL. It was originally developed in 1998 by Carlo Strozzi; a member of the Italian Linux society, with the intention of being a non-relational, widely distributable and highly scalable database. Strozzi named the database management system NoSQL to merely state it does not express queries in the traditional SQL format. Sadalage and Fowler believe the definition we commonly refer NoSQL as comes from a 2009 conference in San Fransisco held by Johan Oskarsson, a software developer. Sadalage and Fowler recall Oskarssons desire to generate publicity surrounding the event and in an attempt to do so devised the twitter hashtag “NoSQL Meetup”. The main attendees at the conference debrief session were Cassandra, CouchDB, HBase and MongoDB and so the association stuck. [21]

There are are number of key features which encompass the NoSQL framework and encapsulate the essence of its popularity. Many of the NoSQL databases boast their capacity of working in a cluster environment - a cluster being two or more connected computers working collaboratively. Thus delivering a range of options for consistency and distribution [21]

NoSQL solutions are not bound by a definitive schema structure. This permits the ability to freely adapt database records or add custom fields for example without considering structural changes. This is extremely effective when dealing with varying data types and data sets, in comparison to the traditional relational database model which when tackling this issue often resulted in ambiguous field names. [21]

1.5 Database Classification

One of the first decisions to be made when when selecting a database is the characteristics of the data you are looking to leverage. [14] There are a multitude of options available with many different classifications. The following sections discuss a subset of these which are relevant to

this project.

1.5.1 Distributed Database

A distributed database (DDB) comprises of two or more data files located at different sites and servers on a computer network. [23] The advantage of using a DD is that as the database is distributed, multiple users can access a portion of the database at different locations locally and remotely without obstructing one another's work. It is pivotal for the DD database management system to periodically synchronise the scattered databases to make sure that they all have consistent data. [23] For example if a user updates or deletes data in one location is is essential this change is mirrored on all databases. This ability to remotely access a database from all across the world lends itself to not only multinational companies for example but also startup businesses which recruit the expertise of others from various locations.

1.5.2 Document-Oriented Database

Document-orientated database (DODB) are designed for storing, retrieving and managing document files such as XML, JSON and BSON. The documents stored in a DODB model are data objects which describe the data in the document, as well as the data itself. Figure 1.3

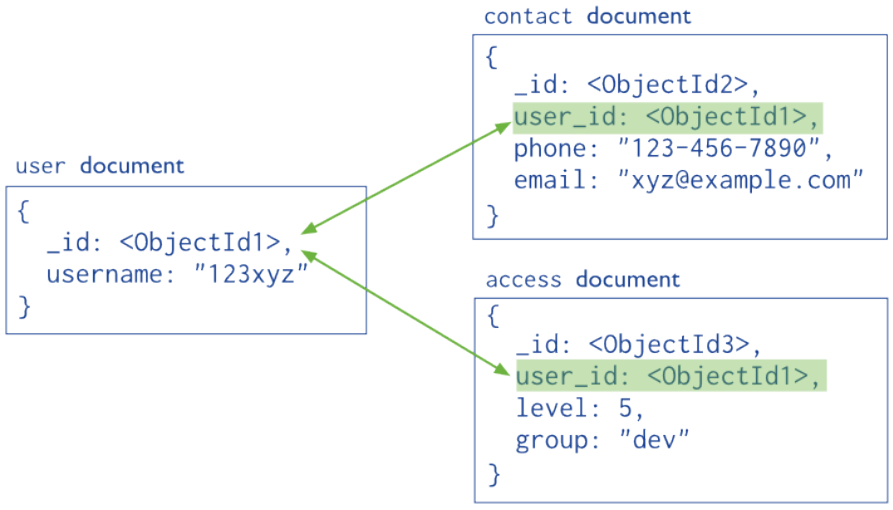


Figure 1.3: MongoDB document

illustrates an example document stored in a DODB specifically in MongoDB. The data is a recognisable JSON format and the joins of the document are between common variable values

within each document.

1.5.3 Graph-Orientated Database

A graph-oriented database (GODB), is a form of NoSQL database solution that uses graph theory to store, map and query relationships. A graph is a collection of nodes connected by relationships. "Graphs represent entities as nodes and the ways in which those entities relate to the world as relationships." [20] The formation of the graph database structure is extremely useful and eloquent as it permits clear modelling of a vast and often ecliptic array of data types. [20] An example of data represented in a graph structure is the Twitter relationship model. Figure 1.4 illustrates the nodes involved in a standard tweet and the relationship link

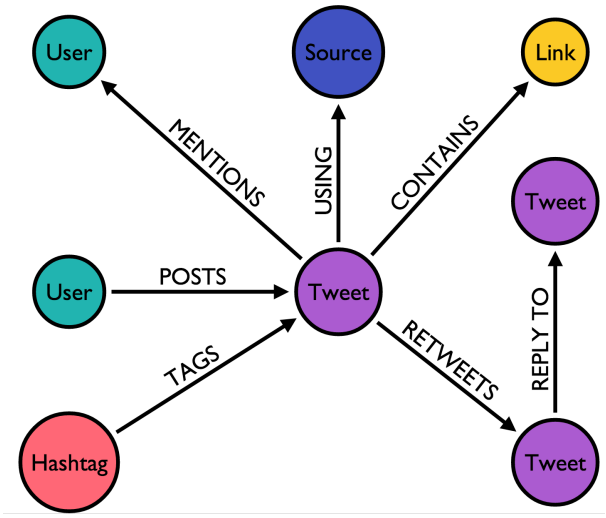


Figure 1.4: Example tweet data relationship

between them. The labeled nodes indicate the various operations which are involved in one the tweet. One interpretation of the figure 1.4 example is that a user posts a tweet, using the Twitter App which mentions another user and includes a hashtag and link.

1.5.4 Relational Database

A relational database (RDB) is a collection of data items organised as a set of tables, records and columns from which data can be accessed or reassembled in many different ways [15]. The connected tables are known as relations and contain one or more columns which comprise of data records called rows. Relations can also be instantiated between the data rows to form functional dependencies.

- One to One: One table record relates to another record in another table.
- One to Many: One table record relates to many records in another table.
- Many to One: More than one table record relates to another table record.
- Many to Many: More than one table record relates to more than one record in another table.

1.5.5 Column-Orientated Database

A column-orientated database (CODB) is a database management system that stores data tables as columns of data rather than as rows of data. The main objective of a CODB is to write and read data from the hard disk efficiently in an attempt to speed up querying time. A CODB has the ability to self index which uses less disk space than RDBMS which holds the same data. A CODB can also be highly compressed, resulting in aggregate functions such as MIN, MAX and SUM to be performed at a extremely high rate. [17].

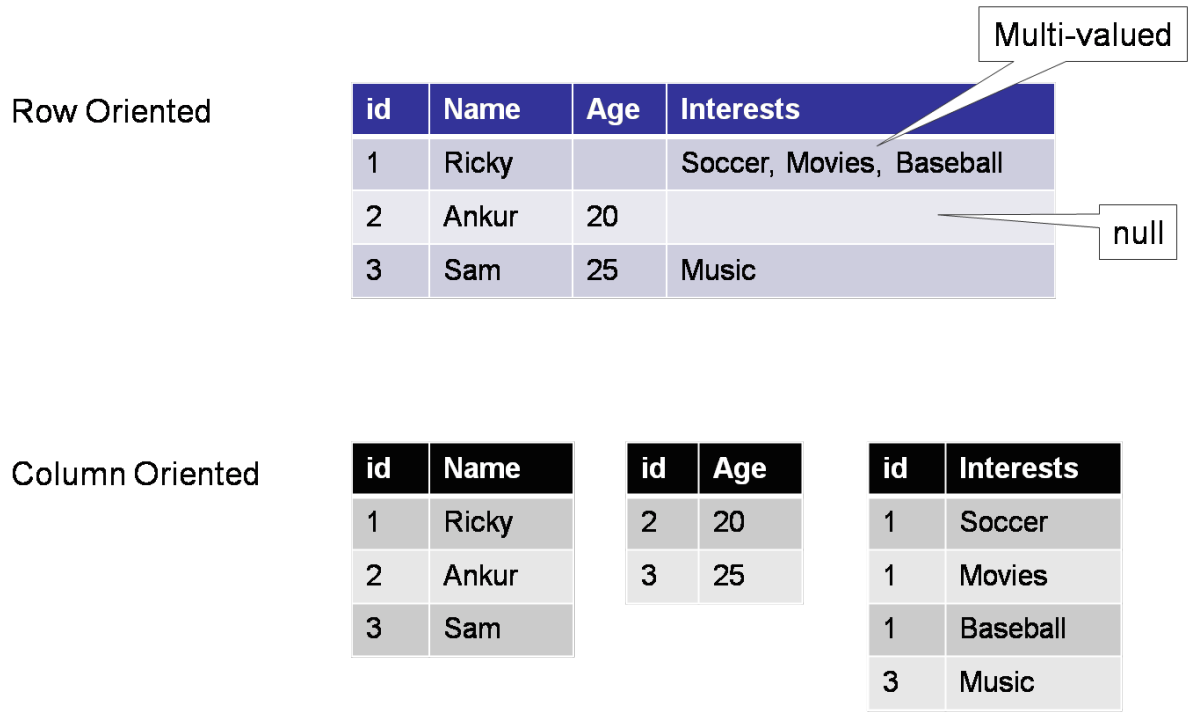


Figure 1.5: Column orientated database example

Figure **??** illustrates the comparison of a RDB model against a CODB model. Within the row based model the data contains both multiple values per record and null values. However

in the CODB model null values are not required as each record contains a minimum and maximum of one value.

1.6 Technology Evaluation

The technologies being evaluated in this project are outlined below.

1.6.1 MongoDB

MongoDB is an open source cross-platform DODB. The premise for using MongoDB is simplicity, speed and scalability [16]. Its ever growing popularity, specifically amongst programmers, stems from the unrestrictive and flexible DODB data model which gives you the ability to query on all fields and boasts instinctive mapping of objects in modern programming languages. [16] The database design of MongoDB is based on the JSON file format named BSON.

A record in MongoDB is known as a document; a data structure composed of field and value pairs. The values of fields can include other documents, arrays and arrays of other documents. The key features of using MongoDB are its high performance data persistence, provide high availability and automatic scaling [16].

1.6.2 Neo4j

Neo4j is an open-source NoSQL GODB which imposes the Property Graph Model throughout its implementation. The team behind the development of Neo4j describe it as an "An intuitive approach to data problems" [7]. One of the reasons in which Neo4j is favoured predominantly amongst database administrators and developers is its efficiency and high scalability. This is in part due to its compact storage and memory caching for the graphs. "Neo4j scales up and out, supporting tens of billions of nodes and relationships, and hundreds of thousands of ACID transactions per second." [7]

The key features of Neo4j which lends itself to users, developers and database administrators are its ability to establish relationships on creating, the equality of relationships permits the addition of new relationships being created after initial implementation at no performance cost and its use of memory caching for graphs which allows efficient scaling.

1.6.3 Apache Cassandra

Apache Cassandra is an open source column-orientated DDB that is designed for storing and managing vast amounts of data across multiple servers. “Apache Cassandra is a highly scalable, high-performance distributed database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure.” [11]. Apache Cassandra define the key features of their database management system as “continuous availability, linear scale performance, operational simplicity and easy data distribution across multiple data centres and cloud availability zones.” [11]. Figure 1.6 illustrates an example record stored in a Cassandra database.

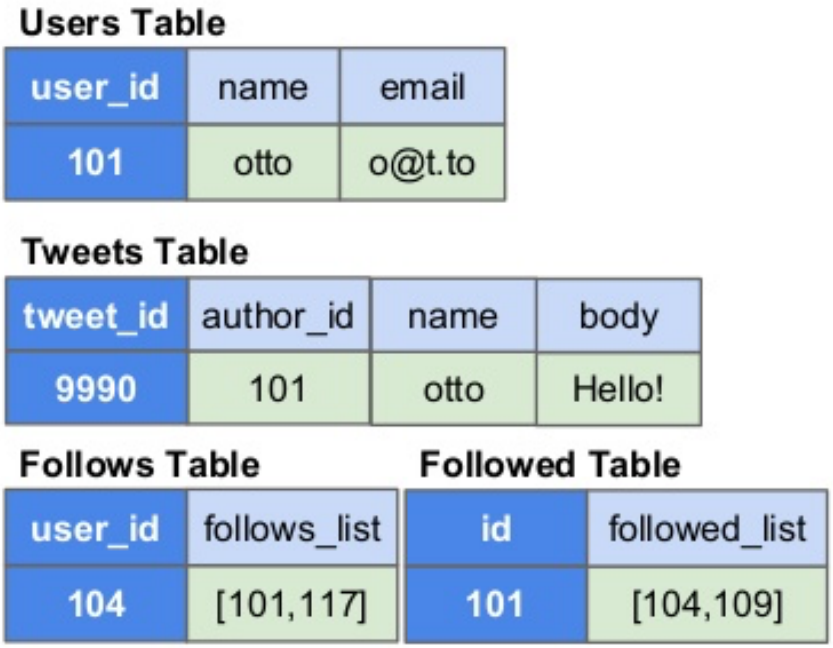


Figure 1.6: Example Cassandra record

1.6.4 MySQL

MySQL is a freely available open source RDB that uses Structured Query Language (SQL). MySQL is commonly used for web applications with its speed and reliability being a key feature. The MySQL database stores data in tables - a collection of related data - which consists of columns and rows. MySQL runs as a server and allows multiple users to manage and create numerous databases.

SQL is a programming language used to communicate with databases through queries.

SQL queries are used to perform tasks such as update or retrieve data in a database. The queries are in the form of command line language which include keyword statements such as select, insert and update.

Chapter 2

Data Source and Representation

The dataset used as a resource to populate the database solutions is called EMAP; a freely available anatomical ontology of the developmental stages of mouse embryos. The EMAP dataset was chosen for this project as my supervisor has much experience in the field and would be able to assist me with any queries I had regarding the data. The size and granularity of the EMAP dataset also meets the criteria which will be required to test the database solutions, explore the limitations of each database comparatively and pose insight into the overall performance of each database.

2.1 EMAP

The *Edinburgh Mouse Atlas Project* (EMAP) is an ongoing research project to develop a digital atlas of mouse development. The objective of the EMAP is to implement a digital model of mouse embryos for each time stage in development [13]. The collated model embryo data is then used to form a database from which further research can be conducted and experiments can be mapped.

Each time step in the digital model are named *Theiler Stages* inspired by the research conducted by Karl Theiler. A Theiler Stage defines the development of a mouse embryo by the form and structure of organisms and their specific anatomical structural features. There are 26 individual Theiler Stages which define the growth and evolution of the mouse embryo. The Theiler Stage scheme comprises of both the anatomical developmental stage definition and the estimated length of time since conception. Each Theiler Stage also provides a brief description of the anatomy and any significant changes between the current and previous stage.

Theiler proposed using this scheme as embryos at the same developmental age can have evolved at different rates and therefore exhibit different structural characteristics EMAP has developed a collection of three dimensional computer models which illustrate and summarise each Theiler Stage.[13]

The anatomy generated at each Theiler Stage has an associated ontological representation. Each provides an alternative aspect of the evolution of a mouse embryo which corresponds with a respective Theiler Stage. The abbreviated term EMAP carries a certain amount of ambiguity as it is refers to the name of the project, and one of the stages in the implemented anatomy. For the purpose of this project the main anatomy to be utilised is the aggregated non stage specific Edinburgh Mouse Atlas Project Abstract (EMAPA) anatomy.

2.1.1 EMAP anatomy

The EMAP anatomy ontology was originally developed to deliver a stage-specific anatomical structure for the developing laboratory mouse. As the EMAP research has progressed, the ontology has followed suit, and is continually under development.

The original EMAP anatomy ontology consists of a series of relational components organised in a hierarchical tree structure which utilise “part-of” relations and subdivisions which encompass each Theiler Stage [13]. The intention behind the implementation of the original ontology structure was to “...describe the whole embryo as a tree of anatomical structures successively divided into non-overlapping named parts” [13].

Each of the Theiler Stage components has an appropriately named term label, known as the *short name* which describes each respective component. Each Theiler Stage also has a *full name* which comes in the form of the components entire hierarchical path [13]. Neither the *full* nor *short* anatomical name of each component are required to be distinct and can appear in several Theiler Stages. Therefore to avoid ambiguity each component can be addressed by a unique identifier. The unique identifier is in the form of the relevant anatomy followed by a number (EMAP:number). For example “choroid plexus” is the short name of TS20/mouse/body region/head/brain/choroid plexus and has a has unique identifier of EMAP:4218.

The EMAP hierarchical structure facilitates the need for basic “data annotation and integration” however a combination of the lack of hierarchical views, missing or poorly represented Theiler Stages and label name ambiguity exposed the limitations of the EMAP

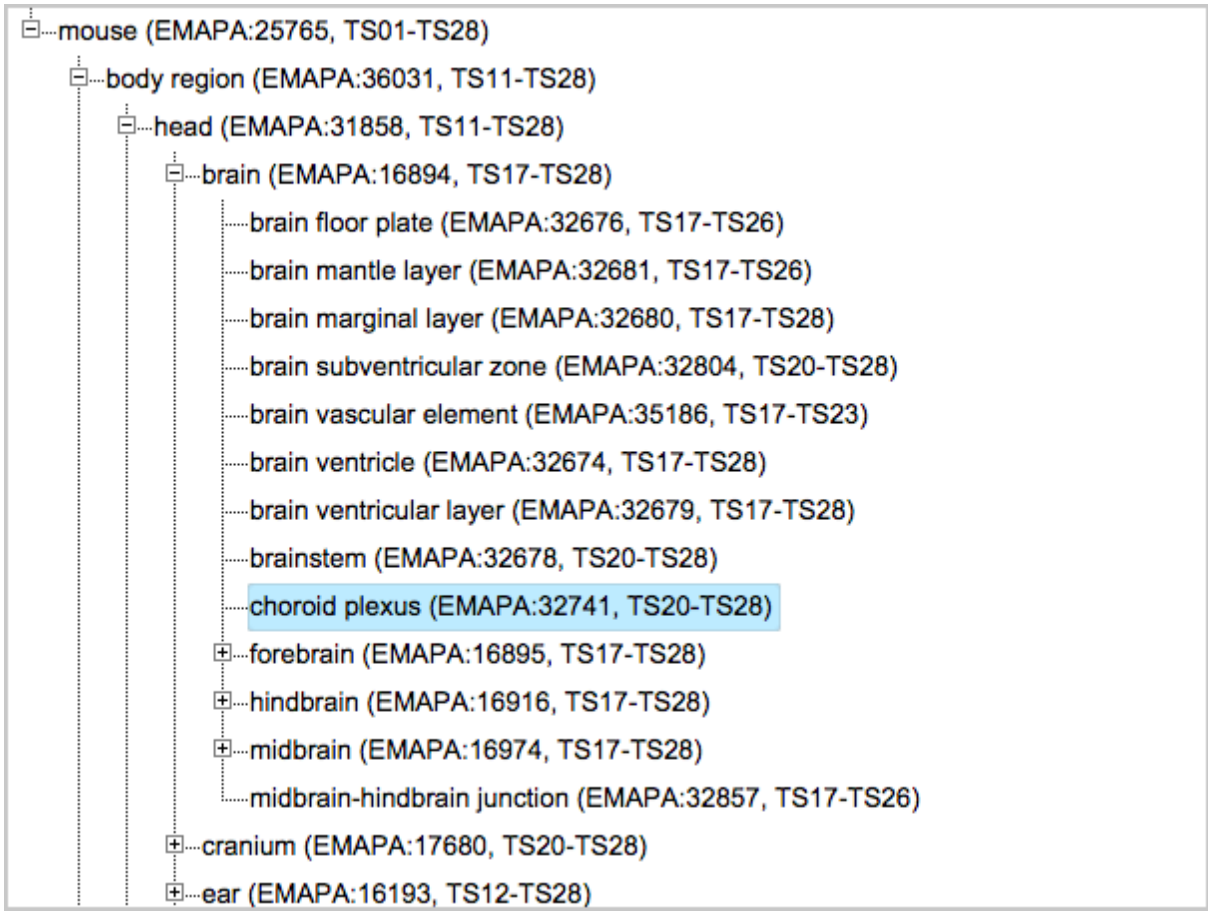


Figure 2.1: EMAPA data structure

structure. As a result the need for a hybrid "abstract" version of the anatomy was identified and subsequently developed; EMAPA. [13] Thus the EMAPA anatomy will be the main data source for this work.

The research surrounding the EMAP resource is continually being developed, thus the growth of the project as a whole is progressively increasing with the richness of data at the heart [13]. The EMAPA ontology discussed below in section 2.1.2 is now considered the primary data source thus the EMAP dataset is available in a combined EMAP and EMAPA standard ontological format developed by the Open Biological Ontologies (OBO) consortium (Section 1.3.3).

2.1.2 EMAPA anatomy

EMAPA is a refined and algorithmically developed non-stage specific anatomical ontology *abstract* representation of the EMAP anatomy. The EMAPA implementation replaces the EMAP hierarchical tree structure for a *directed acyclic graph* structure; a graph in which it is

impossible to start at some vertex v and follow a sequence of edges that eventually loops back to v again. Thus enabling the ability to represent multiple parental relationships and other forms of "is-a" relations where appropriate [13].

Each anatomical component in the EMAPA anatomy is identified as a single term, coupled with the appropriate start and end Theiler Stage at which the component is considered to be present in the developing embryo. [13] With the aim of enhancing user experience, the EMAPA anatomy implements an alternative naming convention from the EMAP anatomy replacing full path names for components to *"print names"*. Using the above example for comparison, "EMAP:4218" in the EMAP anatomy becomes "TS20 brain choroid plexus" in EMAPA. This naming convention supplements the requirement of uniqueness and is easily comprehensible.

The EMAPA ontology is available in a standard ontological format developed by the Open Biological Ontologies (OBO) consortium (Section 1.3.3) and is also available in a Web Ontology Language (OWL); a standard produced from W3C discussed in section 1.3.2.This enhanced version of the ontology EMAPA, is now considered to be the primary EMAP anatomy ontology thus will be the main source for the work on this project.

2.1.3 EMAGE anatomy

EMAGE is a database consisting primarily of image data of *in situ* gene expression data of the developing mouse embryo. The data is sourced from in the community and which is then taken by curators who monitor the EMAGE project and implement it in a standardised way that allows data query and exchange. The description includes a text-based component but the unique aspect of EMAGE is its spatial annotation focus. [13]

"Sites of gene/enhancer expression in EMAGE are described by denoting appropriate regions in the EMAP virtual embryos where expression is detected (and not detected) and also describing this information with an accompanying text-based description, which is achieved by referring to appropriate terms in the anatomy ontology" [13]

The EMAGE anatomy provides an alternative view of the EMAP anatomical ontology. The main data source for this project will be EMAPA however should the need for a data set which holds larger data then the EMAGE data set will be used.

Chapter 3

Evaluation Strategy

The purpose of this chapter is to discuss the methods used to carry out the primary research for this project. In order to achieve the objectives outlined in section 0.1 a formal set of requirements have been established to meet each intended target. Section 3.1 details how each objective will be evaluated and examples are provided of how each technology will be measured by way of competency questions.

The requirement section below is a high level view of what and how the data solutions will be analysed throughout the project. In order to get a detailed and finalised set of requirements an initial prototype model will be developed which will give an insight in to what will be expected to be achieved from the project. It is likely that the requirements below while they may form the basis of the project objectives will change drastically once the prototype model has been developed.

3.1 Requirements

The first objective of the project is to investigate the strengths and weaknesses of the functionality each technology provides. In order to evaluate the functional limitations of each technology, prototype data models will be developed for each solution. The solutions will then be loaded with the EMAPA dataset. Competency questions will be devised which will return a simple yes or no answer. Example competency questions can be found at section 3.1.1. These questions will be translated in to the relevant query language for each of the datasets. The competency questions will range from a beginner level where it will be expected that all database solutions will be able to return the intended value to an advanced level. The

difficulty of writing these queries will also be recorded and used to analyse the performance of the database solution. The time taken for the query to run will also be recorded and used for analysis

1. Is this query possible in database X?

   For example the first query in the EMAPA competency question table asks : "For structure X, find all the ancestors." If this query is possible in a given database, move on to the next query in the list and continue until failure. Once this step has been complete for all database solutions, by way of comparison evaluate the strength of each database.

Objective two of the project is to compare and contrast the analytical capabilities of each technology. This will follow on from the tasks undertaken in the first objective. This objective will also identify and analyse the difficulties which may or may not have arisen in the first objective on the ease of writing the query.

1. If the query is possible in a given database how rich a return of the dataset can it provide.
2. Does any database offer any querying capabilities/functionality which compared with another which aids the query in any way.

The final key objective to be evaluated in this project is to conduct a comparative analysis of the scalability of each technology. This will be achieved by loading the data sources in to each prototype data model. The specific requirements which will measure the performance of each database are :

1. Ease of ETL implementation - The focus of this requirement will be to evaluate any challenges which were faced during the ETL process.
2. Did the data model handle the volume of data and were there any issues as a result.

3.1.1 Data Source Competency Questions

EMAPA	**EMAGE**
For structure X, find all the ancestors.	Where is gene X expressed?
For structure X, find all the decedents.	What is expressed in structure X?
For structure X, find all structures in the same group (this only makes sense in limited situations, e.g., find all "hair").	Which genes are expressed in both structures X and Y? (This is so-called co-expression).
Find all the structures in Theiler Stage X.	Which genes are most commonly co-expressed? (This is getting towards BI and is possibly out with the scope of your thesis, but it does no harm to discuss it.
Find all the structures in Theiler Stages X to Y (e.g., 17-19).	
What stages does structure X appear in?	
Given search term X return the "best match" structure (e.g., if Isearch on "heart" I would expect the output to be heart, heart atrium, heart septum heart mesentery)	
Given an EMAPA ID return the name of the structure.	
Given a name return the EMAPA ID.	

Chapter 4

Risk Assessment

The following table is a summation of the potential risks which may occur during the various stages of this project and the mitigation plans in place.

Risk	**Mitigation Plan**
Corrupt computer hard drive	Regular data back ups will occur on an external hard drive. Multiple file storage services such as Google Drive and Dropbox will be used for the document files.
Postural problems	The appropriate use of a chair and sitting position will be adopted at all times
Visual problems	Optical eye wear such as prescribed glasses will be worn when using the computer equipment
Fatigue and stress	A project plan has been developed which will allow a level control and organisation

Bibliography

[1] Introduction to the semantic web, nov 2014.

[2] Apache jena -, nov 2015.

[3] Best 10 big data quotes of all time | smartdata collective, nov 2015.

[4] Etl - extract transform load, nov 2015.

[5] Etl (extract-transform-load) | data integration info, nov 2015.

[6] Google's schmidt: We know what you're thinking, nov 2015.

[7] Neo4j, the world's leading graph database, nov 2015.

[8] Owl 2 web ontology language document overview (second edition), nov 2015.

[9] Rdf - semantic web standards, nov 2015.

[10] Statistic brain | market research, rankings, financials, percentages, nov 2015.

[11] What is apache cassandra?, nov 2015.

[12] Why only one of the 5 vs of big data really matters | the big data hub, nov 2015.

[13] KAUFMAN M. H. DUBREUIL C. BRUNE R. M. BURGER A. BALDOCK R. A. BARD, J. B. and D. R. DAVIDSON. An internet-accessible database of mouse developmental anatomy based on a systematic nomenclature. *Mechanisms of Development*, 74(1-2):111–120, 1998.

[14] A. BRUST. Rdbms vs. nosql: How do you pick? | zdnet, nov 2013.

[15] J. L. HARRINGTON. *Relational database design clearly explained*. Morgan Kaufmann Publishers, first edition, 2002.

[16] MEMBREY P. HOWS, D. and E. PLUGGE. *MongoDB basics*. Apress, 2014, first edition, 2014.

[17] S. LAMBA C. KUMARDWIVEDI, A. and S. SHUKLA. Performance analysis of column oriented database vs row oriented database. *International Journal of Computer Applications*, 50:31–34, 2012.

[18] CHUI M. BROWN B. BUGHIN J. DOBBS R. ROXBURGH C. MANYIKA, J. and A. HUNG BYERS. *Big Data*. McKinsey Global Institute, first edition, 2011.

[19] Yin Zhang Victor CM Leung Min Chen, Shiwen Mao. *Big Data: Related Technologies, Challenges and Future Prospects*. Springer, 2014, 2014.

[20] WEBBER J. ROBINSON, I. and E. EIFREM. *Graph databases*. O'Reilly, 2013.

[21] P. J. SADALAGE and M. FOWLER. *NoSQL distilled*. Addison-Wesley, first edition, 2013.

[22] AITKEN S. MOREIRA D. A. MUNGALL C. SEQUEDA J. SHAH N. H. TIRMIZI, S. and D. P. MIRANKER. Mapping between the obo and owl ontology languages. *J Biomed Sem*, 2:S3, 2011.

[23] M. T. □ZSU and P. VALDURIEZ. *Principles of distributed database systems*. Springer, first edition, 2011.