

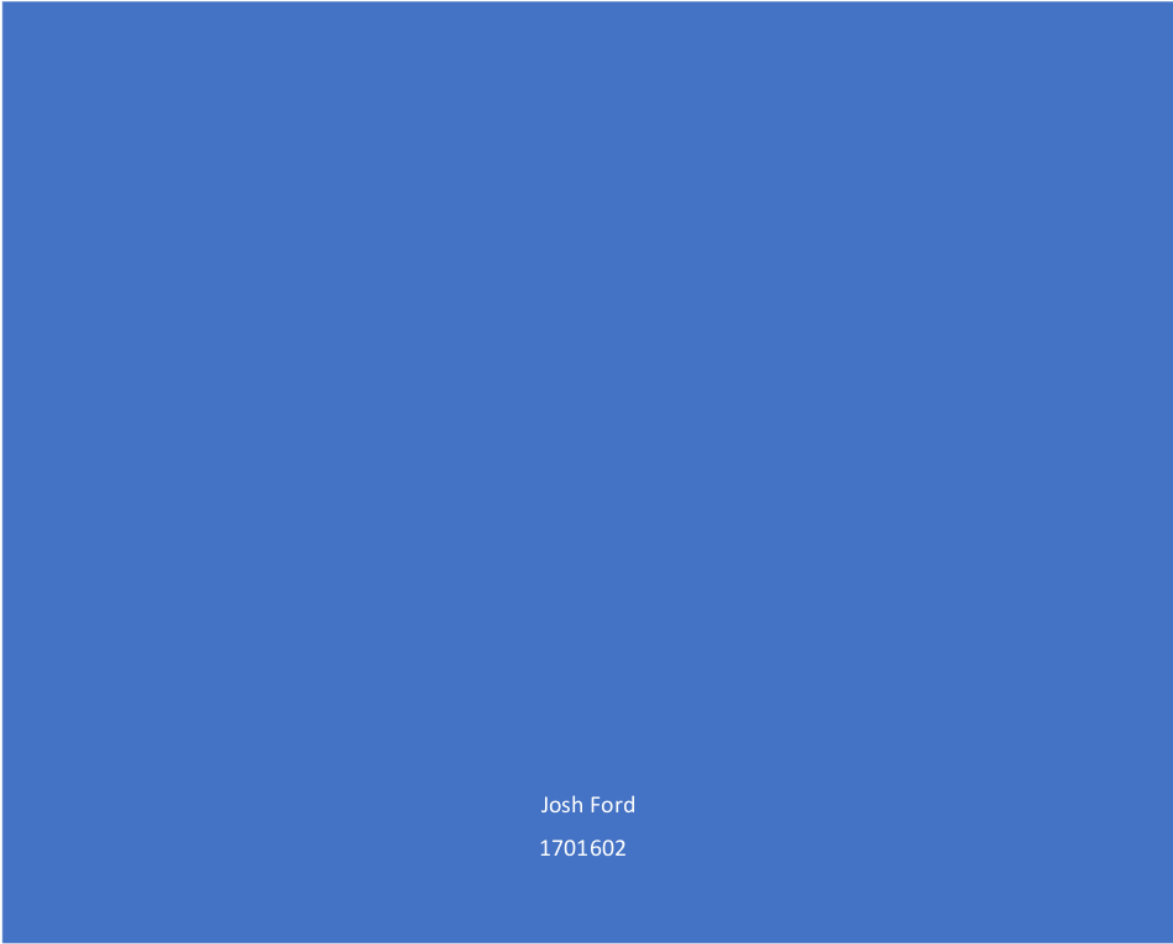
Josh Ford 1701602

by Joshua Ford

FILE	4098_JOSHUA_FORD_JOSH_FORD_1701602_245230_1243021536.DOCX (44.04K)		
TIME SUBMITTED	10-MAY-2020 02:08PM (UTC+0100)	WORD COUNT	3028
SUBMISSION ID	126889722	CHARACTER COUNT	14632



BIG DATA, ASSIGNMENT 2



Josh Ford
1701602

Contents

1. Introduction	2
2. Key-Value databases	2
3. Document orientated database	4
4. Graph databases	6
5. Conclusion	7
6. References	8

1. Introduction

This report will look at the benefits, drawbacks and features of both key-value databases and document databases, then looking at the main features of both and how they work. While looking at them both it will look at how they can be designed and examples that show these databases in action and examples of when these would be used. These examples will be applications and the different applications that would need the databases and why. Finally, it will look at a social media site that requires a database and why a graph database would be suitable. This will involve looking at its features, users and why it makes it suitable for a social media site to use.

2. Key-Value databases

A key-value database is a database that uses a key value method to store details, this means that it is stored in key-value pairs which serves as a unique ID [1]. The two values can be anything and this allows for horizontal scaling which other databases do not allow.[1]

The benefits of using a key value database is that when something needs to be looked up in the database, the data being looked for will be stored under one attribute so will be easier to find [2]. This also goes on to say that another benefit is that in other databases such as relational databases, the data must be in tables and there is a schema that is used. In key value databases there is no schema used and the data is represented by a key value or key value pair, the key being anything that is set by the user [3]. A drawback of using a key value database is that because the data is stored under key value, you cannot find data under the value part of the data being saved [4]. This means that if you save a value under the ID, you cannot search for that data, only the value in which it is stored under.

There are many datatypes that can be stored in key value databases. An example would be the database Redis which is a key value database. This can store many datatypes including lists, which are collections of strings, hashes, which are maps composed of fields associated with values, and bitmaps which are simply images [5]. By being able to store the many different data types mentioned, Redis and key value databases are very useful and can be used for many circumstances because of its use of storing data such as strings, Int's, bitmaps and many more.

The choices for the key of a dataset would be anything set by the user, because there is no schema used, the key can be whatever is set by the user [1]. The key could be the name of the object that is being stored or be more specific, it could just be set as an ID for the set of objects that is being stored. Hash functions are used in key value databases as ways to find key values using addresses, these addresses will be made up of the values and keys of that piece of data [6]. The reason this is done is so that the database does not need to waste any time or performance looking for that key because it already knows where it is because of the address having part of the value and key. Another feature of key value databases that is similar are namespaces. Namespaces are used for if you have two datasets with similar names and similar values, this is where namespaces come in. Datasets can be stored in namespaces which can then help with situations such as this, the two similar datasets can be stored under different namespaces [7]. An example of this could be two tracks stored in a key value database with the same track name and artist, these could be from different

albums. The namespace could then be added and named after the album. This means that when it is searched for then, using the namespace, it can be found correctly.

An example of a suitable application that would benefit from these features of a key value application would be an application that would be storing images. The application which would have people storing images and letting other people see these images on their own feed. This would be done with the key of the images, then a key which would have the user which posted it. Once user post images on to the application, they would then need to be displayed from the database, for when someone else tries to view someone else's image that they have posted. When the image is fetched from the database it would look for the image that is needed and then the user that posted it. it could be a way of completing this, but the database would have to go and look for both keys. A way of improving this would be to use a hash function. This would be done by creating a hash function that collected the image id aswell as the user id in one retrieve. The hash would have the image id which has the value, which would be the image and then the second key which would have the user id which would have the value of the user id. This would then know what to look for when retrieving the two keys using the hash function. This would also save performance and time because the database knows what keys to look for.

As shown in the table below it shows that the two keys image ID and User ID, their values are image 1 etc, and user 1 etc. As explained, the database would have to look for the keys image id and user id and when found it can then display or retrieve the value of the key.

Image ID	User ID
Image 1	User 1
Image 2	User 2

The naming of the keys or important because the less work the database does, means better performance. With the keys being image id and user id, this is important because with the application required to get images and the correct ones posted by the user, it is important to only have the keys required and to have the two keys that are required. This is because when other images are added or similar, the creator does not want to add in other keys to cater for the new content. By having the keys named after something that could be used for all content will be important for the creator for ease of use and the performance.

An SQL database could still be used for this application because the way in which the key value database would be set up, with the user ID, the ID would have the value of the ID of the user and not actually the details of the user. An SQL database could be used for store users details so that they can be used on the application aswell.

With an SQL database having rows and columns, this would be suitable for storing user information which would be stored under one single ID.

User ID	name	username	DOB
User 1	Joe	Joe123	12/04/99
User 2	Jack	Jack1212	22/12/98

This table above shows an example of what the SQL database would look like if created. It shows the users information stored under the user ID of user 1, which also for the key value database table, the users in user ID were saved under user ID. Which means that if implemented, the user ID from the key value table would be referring to the information of the user in the SQL table.

3. Document orientated database

A document-oriented database is a type of database that stores data in collections. The documents are stored as either XML, JSON or BSON [8]. A collection has documents, each document contains a set of fields, with each field being able to be indexed individually [8]. This would mean that if you were to create a document with a certain field and the user wanted to create an index that showed all the fields from the different documents added, this could be done. Many documents can be created inside of a collection, these documents could be the store for a products having fields for the information such as name, price, description etc. A new document could then be created inside of a collection for each new product. The collection houses the documents.

The benefits of this type of database is that it offers flexibility because it allows the data being stored to evolve [9], this means that there is not set schema meaning that the layout of the database or document could change. This would be useful for developers who are creating an application and is constantly changing and so is the data being stored.

With having flexibility is a benefit, it can also be a drawback because of no schema or set structure. This could be a drawback because if a user was entering data for multiple users and they wanted to create a field for date of birth and some users do not want to enter, the field would just be left blank. This means that when looking at the data for the different entries for date of birth It will show the entries and with the users that did not enter, rather than a null or n/a it will not show anything. This could be a problem if rounding up data and could make a problem for the user. Another drawback in document databases is that there are no foreign keys meaning that it is very hard to create relationships between the data that is being stored. This is a problem because if products were being stored, and some data would be used to both fields and documents, it could not be done easy.

An example of an application that would use this type of database would be an application that was selling products. It would be done by having the database and within the database having collections for the products, orders etc. The first collection would be for the products.

As shown below, it would have a collection for the product documents which the documents would have fields for its name, price, and description. As stated, before because it does not have a schema there is no set structure. This is shown in document 2 where it does not have a description field, this is not required, and it is also not against the schema.

Collection 1 {

Document1

Type: product
Name: book
Price (£): 3.99
Description: maths book

Document 2

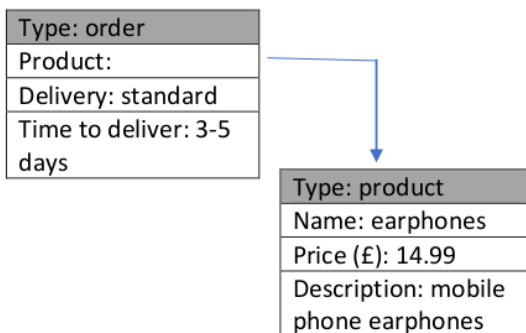
Type: product
Name: jumper
Price (£): 8.99

Document 3

Type: product
Name: earphones
Price (£): 14.99
Description: mobile phone earphones

}

To add to this, a technique called **normalisation** could be used to enhance it even more. Denormalization allows data from other documents to be imbedded into another document [10]. This could be because, if one document were referring to another document, the database would have to read and find both documents. Denormalization helps with this issue by embedding that data inside of that document rather than another document. An example of this would be to extend the example explained above.



The example above shows a document which contains details about an order. The field which asks about the product, rather than referring to another document, it just embeds

the information in the document itself. This means that the information is there under that field already.

4. Graph databases

Graph databases are built to store and navigate relationships [11]. Graph databases use nodes to store data entities and edges to store relationships between entities [11]. With the use of nodes and edges, this would be useful for the use of a social media site because the nodes which store data would be the people or profiles of people, this would be where their information would be stored. This would be for the many nodes which could be stored. The relationship between nodes would then be an example of mutual friends and if the two nodes that connect the edge would be friends. If the two nodes that were connected by an edge meant that they were friends, if another node then had an edge or relationship with that node, it could then see its edges with other nodes. This would be an example of nodes/users seeing mutual friends and who they are friends with. This would be a good reason to use a graph database because of this reason alone. It would allow users to store their data in their own node which could then be connected to another node by an edge to see other friends they are friends with.

Another good reason to use a graph database is because a graph database allows to add new nodes and relationships without compromising existing network [12]. This means that when the application allows user to join the site, it will not affect any other nodes. This is the same for creating relationships, if a new user wanted to add a new friend or add a new relationship, this will not effect any other relationships, including the one of the node in which the new user is trying to connect to. These points are useful because this allows for expansion of the site and allows for users to keep on joining without effecting any existing data.

Another reason to use it would be because of its high query writes[12]. The high query writes will be consistent for small queries and larger queries such as ones including several nodes[12]. This backs up the advantage of more nodes being able to join without effecting other data. To go with this, the query speeds will not affect complex queries which would work for more users and relationships. Carrying on about performance, when nodes make direct connections, 'relating data becomes a simple matter of following connections'[12], this means that join indexing is avoided, which enhances performance.

Graph databases also have good security measures in place, one being that it can recognise relationship patterns such as more than one people associated with an email account or multiple people sharing the same IP address from different addresses [11]. This is a useful feature to have because it has a base layer of security as standard which will be useful for a social media site because social media can be a very powerful tool.

The final reason to use this database is because it is a very well used database, it has many users such as eBay, Adobe, UBS, US army, Microsoft, and many, many more[13]. This is good because it shows that it is a very well used and trusted database model. Also meaning that because it is so well used, there will be lots of documentation about how to best use it and the ways in companies have found ways to get the best out of the database model.

5. Conclusion

To conclude this report, it has been shown the benefits, drawbacks and features of both key value and document databases and how they work. It was clear to how these would work when used with an application and how the features of the two can be used to enhance features of an application to improve on ease of use and performance. It was clear that these different databases have different ways of handling data and storing data depending on the type of data being stored. It was also learnt that different databases are useful for different situations, such as a graph database for social media sites, document databases for storing products and things with many fields associated to that object, and key value databases for example storing individual items and these being associated with items from other keys. Also, with key value, it was learnt that they are also very useful for applications that store data belonging to users and then need to be displayed depending on the users, examples such as snapchat, twitter with its tweets, Instagram with its photos aswell as Pinterest[14].

6. References

- [1] 'What Is a Key-Value Database?' [Online]. Available: <https://aws.amazon.com/nosql/key-value/>. [Accessed: 04-May-2020].
- [2] O. Osemwegie, K. Okokpujie, N. Nkordeh, C. Ndujiuba, S. John, and U. Stanley, 'Performance Benchmarking of Key-Value Store NoSQL Databases', *Int. J. Electr. Comput. Eng.*, vol. 8, no. 6, pp. 5333–5341, 2018.
- [3] A. El Alami, M. Bahaj, and Y. Khourdifi, 'Supply of a key value database redis in-memory by data from a relational database', in *19th IEEE Mediterranean Electrotechnical Conference, MELECON 2018 - Proceedings*, 2018, pp. 46–51.
- [4] 'Everything you need to know about NoSQL databases - DEV Community '. [Online]. Available: <https://dev.to/lmolivera/everything-you-need-to-know-about-nosql-databases-3o3h>. [Accessed: 04-May-2020].
- [5] 'What is a Key-Value Database? | Database.Guide'. [Online]. Available: <https://database.guide/what-is-a-key-value-database/>. [Accessed: 04-May-2020].
- [6] N. Cassee and A. Wijs, 'Analysing the performance of GPU hash tables for state space exploration', in *Electronic Proceedings in Theoretical Computer Science, EPTCS*, 2017, vol. 263, pp. 1–15.
- [7] M. Teresa González-Aparicio, M. Younas, J. Tuya, and R. Casado, 'Testing of transactional services in NoSQL key-value databases', *Futur. Gener. Comput. Syst.*, vol. 80, pp. 384–399, 2018.
- [8] H. Hashem and D. Ranc, 'Evaluating NoSQL document oriented data model', in *Proceedings - 2016 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016*, 2016, pp. 51–56.
- [9] 'What Is a Document Database?' [Online]. Available: <https://aws.amazon.com/nosql/document/>. [Accessed: 06-May-2020].
- [10] 'MongoDB: Normalization vs Denormalization - DEV Community '. [Online]. Available: <https://dev.to/damcosset/mongodb-normalization-vs-denormalization>. [Accessed: 06-May-2020].
- [11] 'What Is a Graph Database?' [Online]. Available: <https://aws.amazon.com/nosql/graph/>. [Accessed: 06-May-2020].
- [12] 'Guest View: Relational vs. graph databases: Which to use and when? - SD Times'. [Online]. Available: <https://sdtimes.com/databases/guest-view-relational-vs-graph-databases-use/>. [Accessed: 06-May-2020].
- [13] 'Neo4j Customers - Neo4j Graph Database Platform'. [Online]. Available: <https://neo4j.com/customers/>. [Accessed: 06-May-2020].
- [14] 'Who's using Redis? – Redis'. [Online]. Available: <https://redis.io/topics/whos-using-redis>. [Accessed: 06-May-2020].

FINAL GRADE

60/100

GENERAL COMMENTS

Instructor

KV DB (10): 10
KV Des (20): 11

Doc DB (20): 14
Doc Des (20): 10

Graph DB (20): 15

Feedback

=====

The example provided for the KV DB is a little hard to follow, but the basic concept seems to be correct.

Slight confusion in the discussion of Doc DBs, or at least it was not very clear. A slightly more interesting design was anticipated

For the graph DB you have covered a number of interesting points and described a key reason for using it in detail.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5



Comment 1

Surely this is one of the advantages of a Doc DB because you don't waste storage on values you that are not required

PAGE 6



Comment 2

denormalisation?

PAGE 7

PAGE 8

PAGE 9
