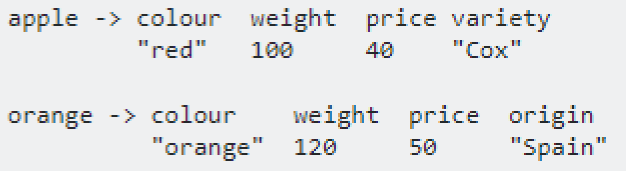
Mini Test Question 1

(a) In this small report I will be looking at the key differences between structured and semi-structured databases, as well as giving examples to show how these are supported in NoSQL document and column stores using code from Cassandra and MongoDB. Structured databases have relational keys and the data within them is organised and put into tables with rows and columns. Semi-structured databases are databases which are not relational but do have some organisational properties.

The key differences between structured and semi-structured databases are that structured databases are done in such a way that avoids complexity and their data structure is organised logically. (Soltesz, n.d.) However with semi-structured databases the data is not stored as logically as with a structured database and as such the model is a lot more complex (Team, 2008). Data retrieval within a structured database is a lot easier as the user can query the entire database easily (Soltesz, n.d.). With a semi-structured databases the data is not as constrained and therefore is a lot harder to query (Team, 2008).

Columns stores store tables as the name suggests uses columns rather than rows to store data. The reason why it is stored this way it because it allows for high scalability and performance which make it ideal for storing large amounts of data (Wodehouse, n.d.). An example of this can be seen in Figure 1.

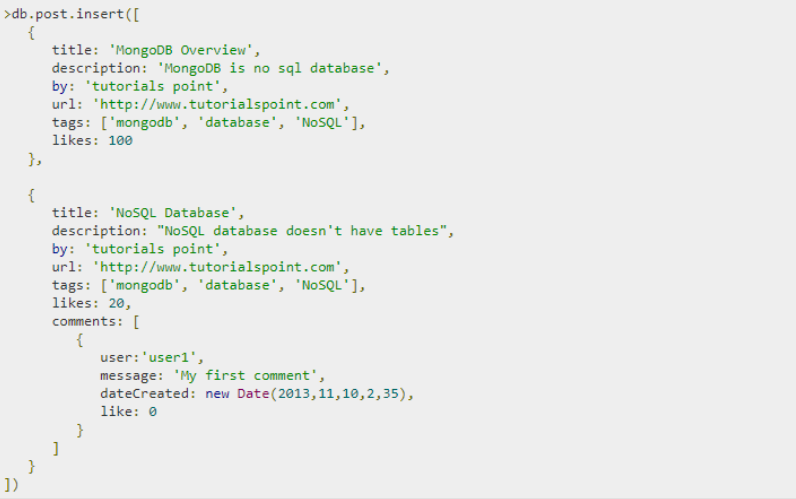


```
apple -> colour  weight  price variety
         "red"   100     40    "Cox"

orange -> colour    weight  price  origin
          "orange"  120     50     "Spain"
```

Figure 1: Cassandra Column Store code example (Stack Overflow, 2012)

With a document store the normal key concept is taken a step further to make each document, which has its own data, have its own unique key, which is used to query the specific document. This particular type of store is good for semi-structured data and structured data as each document can have its data structured or semi-structured (Wodehouse, n.d.). An example of this can be seen in Figure 2.



```
>db.post.insert([
   {
      title: 'MongoDB Overview',
      description: 'MongoDB is no sql database',
      by: 'tutorials point',
      url: 'http://www.tutorialspoint.com',
      tags: ['mongodb', 'database', 'NoSQL'],
      likes: 100
   },

   {
      title: 'NoSQL Database',
      description: "NoSQL database doesn't have tables",
      by: 'tutorials point',
      url: 'http://www.tutorialspoint.com',
      tags: ['mongodb', 'database', 'NoSQL'],
      likes: 20,
      comments: [
         {
            user:'user1',
            message: 'My first comment',
            dateCreated: new Date(2013,11,10,2,35),
            like: 0
         }
      ]
   }
])
```

Figure 2: MongoDB Document Store code example (www.tutorialspoint.com, n.d.)

(b) Column stores that have had their data ran through compression algorithms and the data being kept in a compressed format have had significant improvements in their query performance (Abadi, Madden and Hachem, 2018). By default data stored in a column store compared to a row store in a relational database is more compressible, this is due to compression algorithms being more effective on data that has low information entropy (Abadi, Madden and Hachem, 2018). If a database needs to store information on customers, such as their phone numbers, the phone numbers will all be stored together and as a result they will be more compressible due to the fact that they can be run-length encoded (Abadi, Madden and Hachem, 2018). To speed up query processing column stores skip reading columns that are not required by the query, this is called column elimination and reduces query execution time especially in databases which are large, for example if a databases has 100 columns and the query requires only 5 of them, the query only looks at those 5 which is a 95% performance increase over relational databases (Docs.microsoft.com, 2017).

References

Abadi, D., Madden, S. and Hachem, N. (2018). Column-stores vs. row-stores: how different are they really?. [online] pp.1-14. Available at: http://db.csail.mit.edu/projects/cstore/abadi-sigmod08.pdf [Accessed 13 May 2018].

Docs.microsoft.com. (2017). Columnstore indexes - Query performance. [online] Available at: https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-query-performance?view=sql-server-2017 [Accessed 13 May 2018].

Soltesz, D. (n.d.). What are the Advantages of a Relational Database Model? | Techwalla.com. [online] Techwalla. Available at: https://www.techwalla.com/articles/what-are-the-advantages-of-a-relational-database-model [Accessed 13 May 2018].

Stack Overflow. (2012). Why many refer to Cassandra as a Column oriented database?. [online] Available at: https://stackoverflow.com/questions/13010225/why-many-refer-to-cassandra-as-a-column-oriented-database [Accessed 13 May 2018].

Team, E. (2008). Semi-Structured Model | Online Learning. [online] Learn.geekinterview.com. Available at: http://www.learn.geekinterview.com/it/data-modeling/semi-structured-model.html [Accessed 13 May 2018].

Wodehouse, C. (n.d.). SQL vs. NoSQL Databases: What’s the Difference?. [online] upwork. Available at: https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/ [Accessed 13 May 2018].

www.tutorialspoint.com. (n.d.). MongoDB Insert Document. [online] Available at: https://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm [Accessed 13 May 2018].