Central Hospital Case Study Part 2 - NoSQL

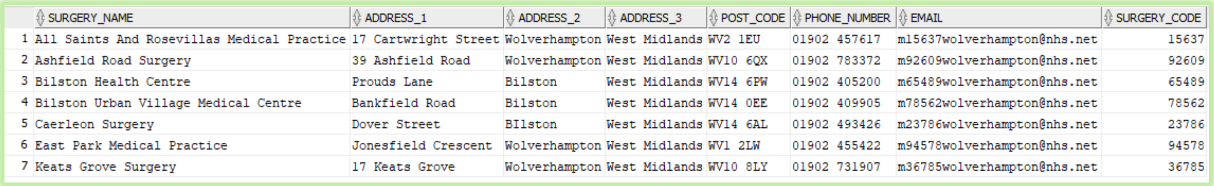
Contents

Section five - NoSQL Database

Design Decisions

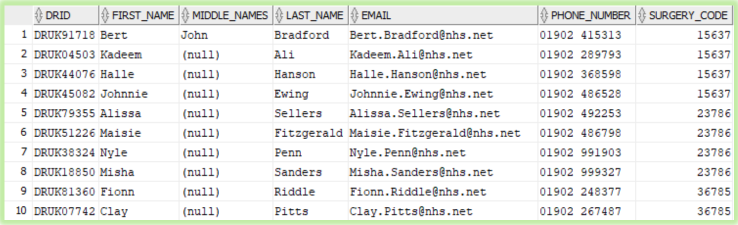
From the Central Hospital scenario provided, eight separate tables were created for the Oracle SQL DBMS. When reimagining the same scenario, but through the lens of utilising a NoSQL DBMS such as MongoDB, it was clear that simply creating a collection for each table would not be sufficient. In order to maximise the functionality of the MongoDB system several of the original tables were merged with foreign keys being eliminated. Typically, smaller tables that existed separately have been incorporated into larger tables inside arrays. For example, where there was a surgeries table and a separate table that contained doctors linked by a foreign key to the surgery that they belonged too, within the MongoDB Surgeries collection each surgery document has an array of doctor elements, where each element is a record from the original doctor table.

Original Surgery Table



	SURGERY_NAME	ADDRESS_1	ADDRESS_2	ADDRESS_3	POST_CODE	PHONE_NUMBER	EMAIL	SURGERY_CODE
1	All Saints And Rosevillas Medical Practice	17 Cartwright Street	Wolverhampton	West Midlands	WV2 1EU	01902 457617	m15637wolverhampton@nhs.net	15637
2	Ashfield Road Surgery	39 Ashfield Road	Wolverhampton	West Midlands	WV10 6QX	01902 783372	m92609wolverhampton@nhs.net	92609
3	Bilston Health Centre	Prouds Lane	Bilston	West Midlands	WV14 6PW	01902 405200	m65489wolverhampton@nhs.net	65489
4	Bilston Urban Village Medical Centre	Bankfield Road	Bilston	West Midlands	WV14 0EE	01902 409905	m78562wolverhampton@nhs.net	78562
5	Caerleon Surgery	Dover Street	BIlston	West Midlands	WV14 6AL	01902 493426	m23786wolverhampton@nhs.net	23786
6	East Park Medical Practice	Jonesfield Crescent	Wolverhampton	West Midlands	WV1 2LW	01902 455422	m94578wolverhampton@nhs.net	94578
7	Keats Grove Surgery	17 Keats Grove	Wolverhampton	West Midlands	WV10 8LY	01902 731907	m36785wolverhampton@nhs.net	36785

Original Doctors Table



	DRID	FIRST_NAME	MIDDLE_NAMES	LAST_NAME	EMAIL	PHONE_NUMBER	SURGERY_CODE
1	DRUK91718	Bert	John	Bradford	Bert.Bradford@nhs.net	01902 415313	15637
2	DRUK04503	Kadeem	(null)	Ali	Kadeem.Ali@nhs.net	01902 289793	15637
3	DRUK44076	Halle	(null)	Hanson	Halle.Hanson@nhs.net	01902 368598	15637
4	DRUK45082	Johnnie	(null)	Ewing	Johnnie.Ewing@nhs.net	01902 486528	15637
5	DRUK79355	Alissa	(null)	Sellers	Alissa.Sellers@nhs.net	01902 492253	23786
6	DRUK51226	Maisie	(null)	Fitzgerald	Maisie.Fitzgerald@nhs.net	01902 486798	23786
7	DRUK38324	Nyle	(null)	Penn	Nyle.Penn@nhs.net	01902 991903	23786
8	DRUK18850	Misha	(null)	Sanders	Misha.Sanders@nhs.net	01902 999327	23786
9	DRUK81360	Fionn	(null)	Riddle	Fionn.Riddle@nhs.net	01902 248377	36785
10	DRUK07742	Clay	(null)	Pitts	Clay.Pitts@nhs.net	01902 267487	36785

Example MongoDB Surgery document

```
{
   "Surgery_Name": "All Saints And Rosevillas Medical Practice",
   "Address_1": "17 Cartwright Street",
   "Address_2": "Wolverhampton",
   "Address_3": "West Midlands",
   "Post_Code": "WV2 1EU",
   "Phone_Number": "01902 457617",
   "Email": "m15637wolverhampton@nhs.net",
   "Surgery_Code": 15637,
   "Doctors": [
                 {       "DRID": "DRUK91718", "First_Name": "Bert", "Middle_Names": "John","Last_Name": "Bradford",
                         "Email": "Bert.Bradford@nhs.net","Phone_Number": "01902 415313"},
                 {       "DRID": "DRUK04503", "First_Name": "Kadeem", "Last_Name": "Ali",
                         "Email": "Kadeem.Ali@nhs.net", "Phone_Number": "01902 289793"},
                 {       "DRID": "DRUK44076", "First_Name": "Halle", "Last_Name": "Hanson",
                         "Email": "Halle.Hanson@nhs.net", "Phone_Number": "01902 368598"},
                 {       "DRID": "DRUK45082", "First_Name": "Johnnie", "Last_Name": "Ewing",
                         "Email": "Johnnie.Ewing@nhs.net", "Phone_Number": "01902 486528"}
   ]
 }
```

Following a similar pattern, the columns from the consultations table were added to a patient document/record as an array. Equally, the same occurred for the test table, this was also added to the patient document/record as an array. From a logical perspective a patient can exist without a test, or alternatively with several vice versa for consultations.

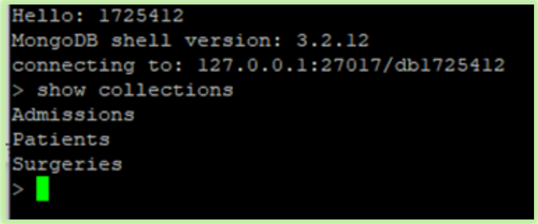
The final collection that was created was the admissions collection, this collection required the most amount of reworking from its original Oracle DBMS state. As it exists now, each document/record represents a single patient admission in the hospital, as such the document/record includes an array of PPE assessments carried out against that patient. Within the array of PPE assessments there is a second array which refers to the member of staff who completed the assessment and this includes the main columns from the hospital staff table.

Data creation and import

All collections/documents were created in notepad++ before then being imported into MongoDB. The sample data that has been used is 90% identical to the original sample data used in part one of this assessment. For example, where there were seven records in the surgery table and twenty eight records in the doctors table, the new collection contains seven surgery documents, each with an array of four doctor documents.

In the following sections you will see an example of each of the raw text files, followed by the import statement and result. Finally, you will see for each of the collections an example of a single record from the MongoDB DBMS.

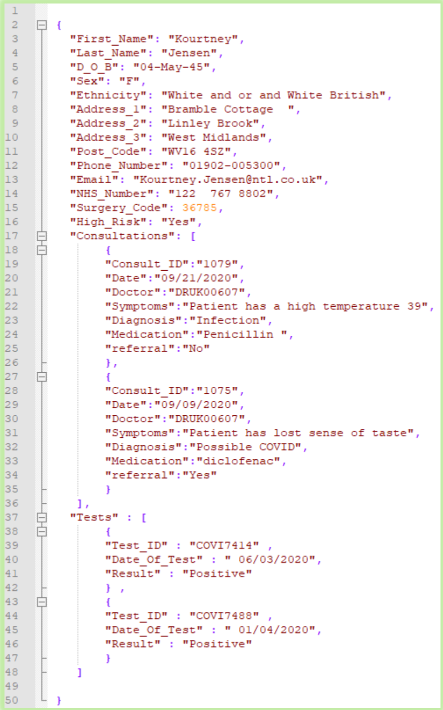
After importing all collections into MongoDB, the following command was run to verify the collections were in place. `“show collections”`



```
Hello: 1725412
MongoDB shell version: 3.2.12
connecting to: 127.0.0.1:27017/db1725412
> show collections
Admissions
Patients
Surgeries
>
```

Patients

Raw data

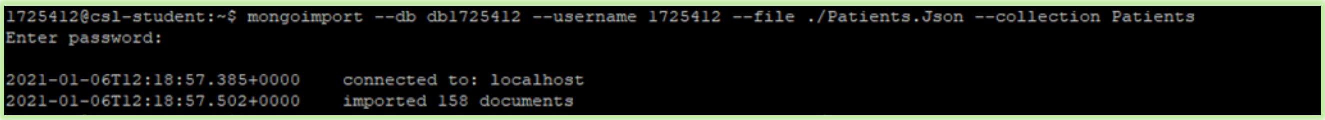


```

{
  "First_Name": "Kourtney",
  "Last_Name": "Jensen",
  "D_O_B": "04-May-45",
  "Sex": "F",
  "Ethnicity": "White and or and White British",
  "Address_1": "Bramble Cottage  ",
  "Address_2": "Linley Brook",
  "Address_3": "West Midlands",
  "Post_Code": "WV16 4SZ",
  "Phone_Number": "01902-005300",
  "Email": "Kourtney.Jensen@ntl.co.uk",
  "NHS_Number": "122  767 8802",
  "Surgery_Code": 36785,
  "High_Risk": "Yes",
  "Consultations": [
       {
       "Consult_ID":"1079",
       "Date":"09/21/2020",
       "Doctor":"DRUK00607",
       "Symptoms":"Patient has a high temperature 39",
       "Diagnosis":"Infection",
       "Medication":"Penicillin ",
       "referral":"No"
       },
       {
       "Consult_ID":"1075",
       "Date":"09/09/2020",
       "Doctor":"DRUK00607",
       "Symptoms":"Patient has lost sense of taste",
       "Diagnosis":"Possible COVID",
       "Medication":"diclofenac",
       "referral":"Yes"
       }
    ],
  "Tests" : [
       {
       "Test_ID" : "COVI7414" ,
       "Date_Of_Test" : " 06/03/2020",
       "Result" : "Positive"
       } ,
       {
       "Test_ID" : "COVI7488" ,
       "Date_Of_Test" : " 01/04/2020",
       "Result" : "Positive"
       }
    ]

}
```

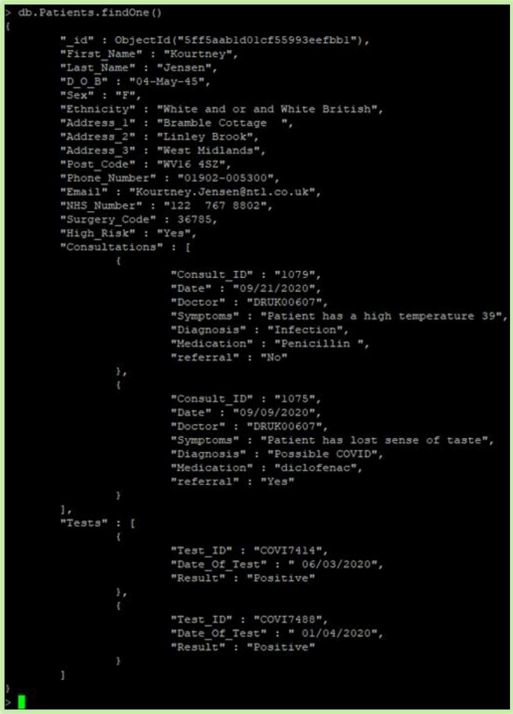
Creation statement



```
1725412@csl-student:~$ mongoimport --db db1725412 --username 1725412 --file ./Patients.Json --collection Patients
Enter password:

2021-01-06T12:18:57.385+0000    connected to: localhost
2021-01-06T12:18:57.502+0000    imported 158 documents
```

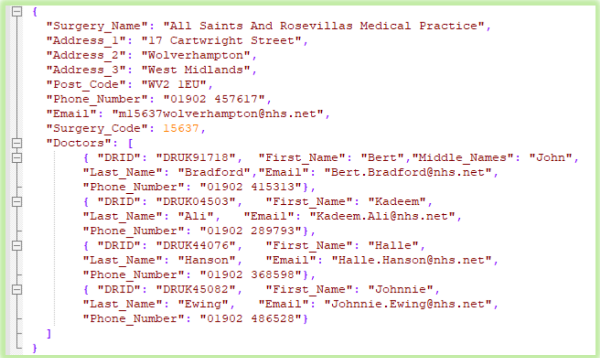
Example record generated by the command "db.Patients.findOne()"



```
> db.Patients.findOne()
{
        "_id" : ObjectId("5ff5aab1d01cf55993eefbb1"),
        "First_Name" : "Kourtney",
        "Last_Name" : "Jensen",
        "D_O_B" : "04-May-45",
        "Sex" : "F",
        "Ethnicity" : "White and or and White British",
        "Address_1" : "Bramble Cottage  ",
        "Address_2" : "Linley Brook",
        "Address_3" : "West Midlands",
        "Post_Code" : "WV16 4SZ",
        "Phone_Number" : "01902-005300",
        "Email" : "Kourtney.Jensen@ntl.co.uk",
        "NHS_Number" : "122  767 8802",
        "Surgery_Code" : 36785,
        "High_Risk" : "Yes",
        "Consultations" : [
                {
                        "Consult_ID" : "1079",
                        "Date" : "09/21/2020",
                        "Doctor" : "DRUK00607",
                        "Symptoms" : "Patient has a high temperature 39",
                        "Diagnosis" : "Infection",
                        "Medication" : "Penicillin ",
                        "referral" : "No"
                },
                {
                        "Consult_ID" : "1075",
                        "Date" : "09/09/2020",
                        "Doctor" : "DRUK00607",
                        "Symptoms" : "Patient has lost sense of taste",
                        "Diagnosis" : "Possible COVID",
                        "Medication" : "diclofenac",
                        "referral" : "Yes"
                }
        ],
        "Tests" : [
                {
                        "Test_ID" : "COVI7414",
                        "Date_Of_Test" : " 06/03/2020",
                        "Result" : "Positive"
                },
                {
                        "Test_ID" : "COVI7488",
                        "Date_Of_Test" : " 01/04/2020",
                        "Result" : "Positive"
                }
        ]
}
>
```

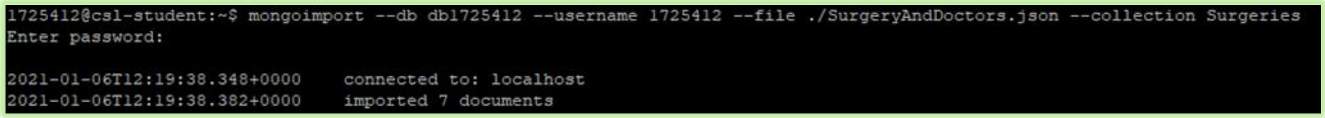
Surgeries & Doctors

Raw Data



```
{
  "Surgery_Name": "All Saints And Rosevillas Medical Practice",
  "Address_1": "17 Cartwright Street",
  "Address_2": "Wolverhampton",
  "Address_3": "West Midlands",
  "Post_Code": "WV2 1EU",
  "Phone_Number": "01902 457617",
  "Email": "m15637wolverhampton@nhs.net",
  "Surgery_Code": 15637,
  "Doctors": [
       { "DRID": "DRUK91718",  "First_Name": "Bert","Middle_Names": "John",
       "Last_Name": "Bradford","Email": "Bert.Bradford@nhs.net",
       "Phone_Number": "01902 415313"},
       { "DRID": "DRUK04503",   "First_Name": "Kadeem",
       "Last_Name": "Ali",   "Email": "Kadeem.Ali@nhs.net",
       "Phone_Number": "01902 289793"},
       { "DRID": "DRUK44076",   "First_Name": "Halle",
       "Last_Name": "Hanson",   "Email": "Halle.Hanson@nhs.net",
       "Phone_Number": "01902 368598"},
       { "DRID": "DRUK45082",   "First_Name": "Johnnie",
       "Last_Name": "Ewing",   "Email": "Johnnie.Ewing@nhs.net",
       "Phone_Number": "01902 486528"}
  ]
}
```

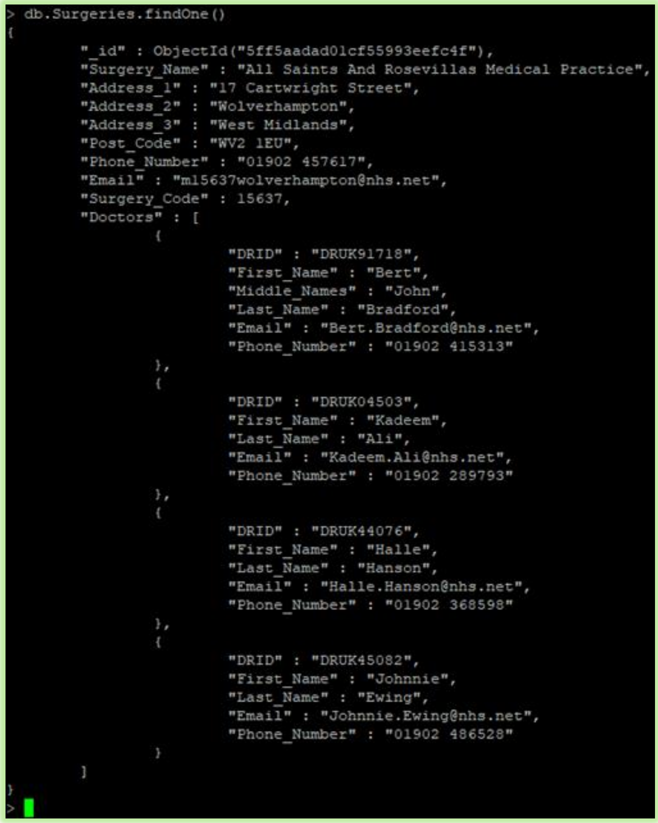
Creation statement



```
1725412@csl-student:~$ mongoimport --db db1725412 --username 1725412 --file ./SurgeryAndDoctors.json --collection Surgeries
Enter password:

2021-01-06T12:19:38.348+0000    connected to: localhost
2021-01-06T12:19:38.382+0000    imported 7 documents
```

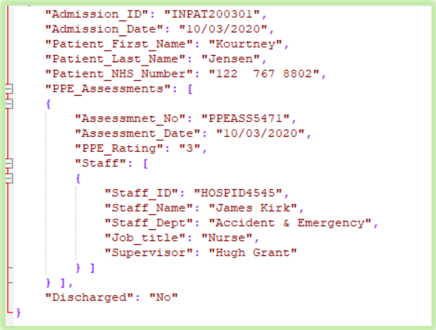
Example record generated by the command “`db.Surgeries.findOne()`”



```
> db.Surgeries.findOne()
{
        "_id" : ObjectId("5ff5aadad01cf55993eefc4f"),
        "Surgery_Name" : "All Saints And Rosevillas Medical Practice",
        "Address_1" : "17 Cartwright Street",
        "Address_2" : "Wolverhampton",
        "Address_3" : "West Midlands",
        "Post_Code" : "WV2 1EU",
        "Phone_Number" : "01902 457617",
        "Email" : "m15637wolverhampton@nhs.net",
        "Surgery_Code" : 15637,
        "Doctors" : [
                {
                        "DRID" : "DRUK91718",
                        "First_Name" : "Bert",
                        "Middle_Names" : "John",
                        "Last_Name" : "Bradford",
                        "Email" : "Bert.Bradford@nhs.net",
                        "Phone_Number" : "01902 415313"
                },
                {
                        "DRID" : "DRUK04503",
                        "First_Name" : "Kadeem",
                        "Last_Name" : "Ali",
                        "Email" : "Kadeem.Ali@nhs.net",
                        "Phone_Number" : "01902 289793"
                },
                {
                        "DRID" : "DRUK44076",
                        "First_Name" : "Halle",
                        "Last_Name" : "Hanson",
                        "Email" : "Halle.Hanson@nhs.net",
                        "Phone_Number" : "01902 368598"
                },
                {
                        "DRID" : "DRUK45082",
                        "First_Name" : "Johnnie",
                        "Last_Name" : "Ewing",
                        "Email" : "Johnnie.Ewing@nhs.net",
                        "Phone_Number" : "01902 486528"
                }
        ]
}
>
```

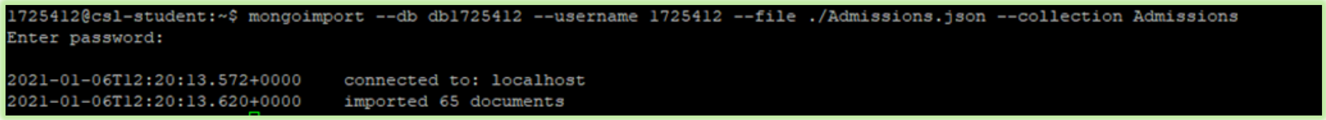
Admissions

Raw Data



```
    "Admission_ID": "INPAT200301",
    "Admission_Date": "10/03/2020",
    "Patient_First_Name": "Kourtney",
    "Patient_Last_Name": "Jensen",
    "Patient_NHS_Number": "122  767 8802",
    "PPE_Assessments": [
    {
        "Assessmnet_No": "PPEASS5471",
        "Assessment_Date": "10/03/2020",
        "PPE_Rating": "3",
        "Staff": [
        {
            "Staff_ID": "HOSPID4545",
            "Staff_Name": "James Kirk",
            "Staff_Dept": "Accident & Emergency",
            "Job_title": "Nurse",
            "Supervisor": "Hugh Grant"
        } ]
    } ],
    "Discharged": "No"
}
```

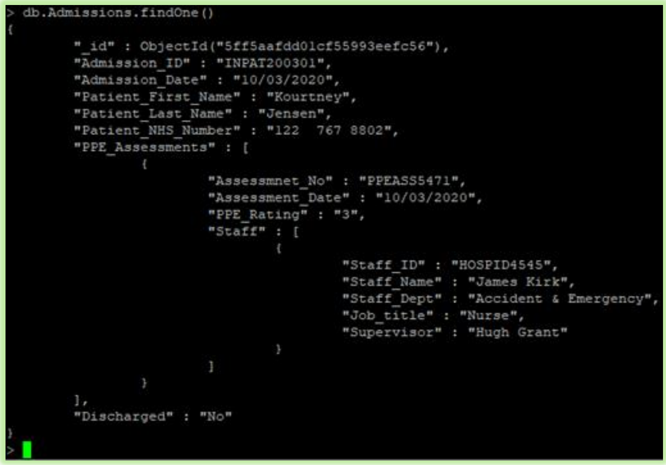
Creation statement



```
1725412@csl-student:~$ mongoimport --db db1725412 --username 1725412 --file ./Admissions.json --collection Admissions
Enter password:

2021-01-06T12:20:13.572+0000    connected to: localhost
2021-01-06T12:20:13.620+0000    imported 65 documents
```

Example record generated by the command “`db.Admissions.findOne()`”



```
> db.Admissions.findOne()
{
        "_id" : ObjectId("5ff5aafdd01cf55993eefc56"),
        "Admission_ID" : "INPAT200301",
        "Admission_Date" : "10/03/2020",
        "Patient_First_Name" : "Kourtney",
        "Patient_Last_Name" : "Jensen",
        "Patient_NHS_Number" : "122  767 8802",
        "PPE_Assessments" : [
                {
                        "Assessmnet_No" : "PPEASS5471",
                        "Assessment_Date" : "10/03/2020",
                        "PPE_Rating" : "3",
                        "Staff" : [
                                {
                                        "Staff_ID" : "HOSPID4545",
                                        "Staff_Name" : "James Kirk",
                                        "Staff_Dept" : "Accident & Emergency",
                                        "Job_title" : "Nurse",
                                        "Supervisor" : "Hugh Grant"
                                }
                        ]
                }
        ],
        "Discharged" : "No"
}
>
```

NoSQL Queries

Query one

Description

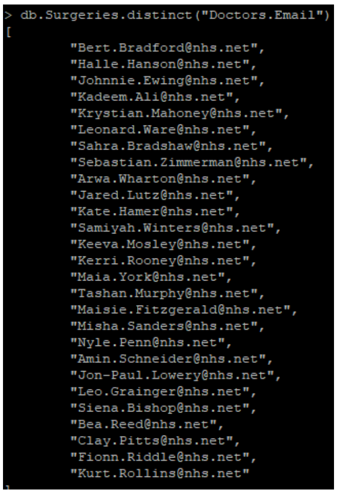
This query looks at the Surgeries collection and returns a list of the email addresses for each of the registered doctors using the distinct command, eliminating and duplicate emails found for any doctor that may have a document/record at multiple surgeries. In a real-world scenario this could be used to gather the email addresses for means of distributing a mailshot with important information.

Code

The command to run the query is

```
db.Surgeries.distinct("Doctors.Email")
```

Results



```
> db.Surgeries.distinct("Doctors.Email")
[
        "Bert.Bradford@nhs.net",
        "Halle.Hanson@nhs.net",
        "Johnnie.Ewing@nhs.net",
        "Kadeem.Ali@nhs.net",
        "Krystian.Mahoney@nhs.net",
        "Leonard.Ware@nhs.net",
        "Sahra.Bradshaw@nhs.net",
        "Sebastian.Zimmerman@nhs.net",
        "Arwa.Wharton@nhs.net",
        "Jared.Lutz@nhs.net",
        "Kate.Hamer@nhs.net",
        "Samiyah.Winters@nhs.net",
        "Keeva.Mosley@nhs.net",
        "Kerri.Rooney@nhs.net",
        "Maia.York@nhs.net",
        "Tashan.Murphy@nhs.net",
        "Maisie.Fitzgerald@nhs.net",
        "Misha.Sanders@nhs.net",
        "Nyle.Penn@nhs.net",
        "Amin.Schneider@nhs.net",
        "Jon-Paul.Lowery@nhs.net",
        "Leo.Grainger@nhs.net",
        "Siena.Bishop@nhs.net",
        "Bea.Reed@nhs.net",
        "Clay.Pitts@nhs.net",
        "Fionn.Riddle@nhs.net",
        "Kurt.Rollins@nhs.net"
]
```

Query two

Description

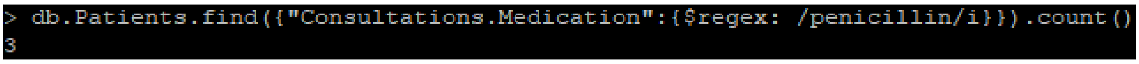
This query looks at all of the consultations that have taken place and counts the number of times that Penicillin has been prescribed as a medication to the patient base. A query like this may be useful so identify any trends with volumes of medications prescribed.

Code

The command to run the query is

```
db.Patients.find
        (
        {"Consultations.Medication":{$regex: /penicillin/i}}
        ).count()
```

Results



```
> db.Patients.find({"Consultations.Medication":{$regex: /penicillin/i}}).count()
3
```

Query three

Description

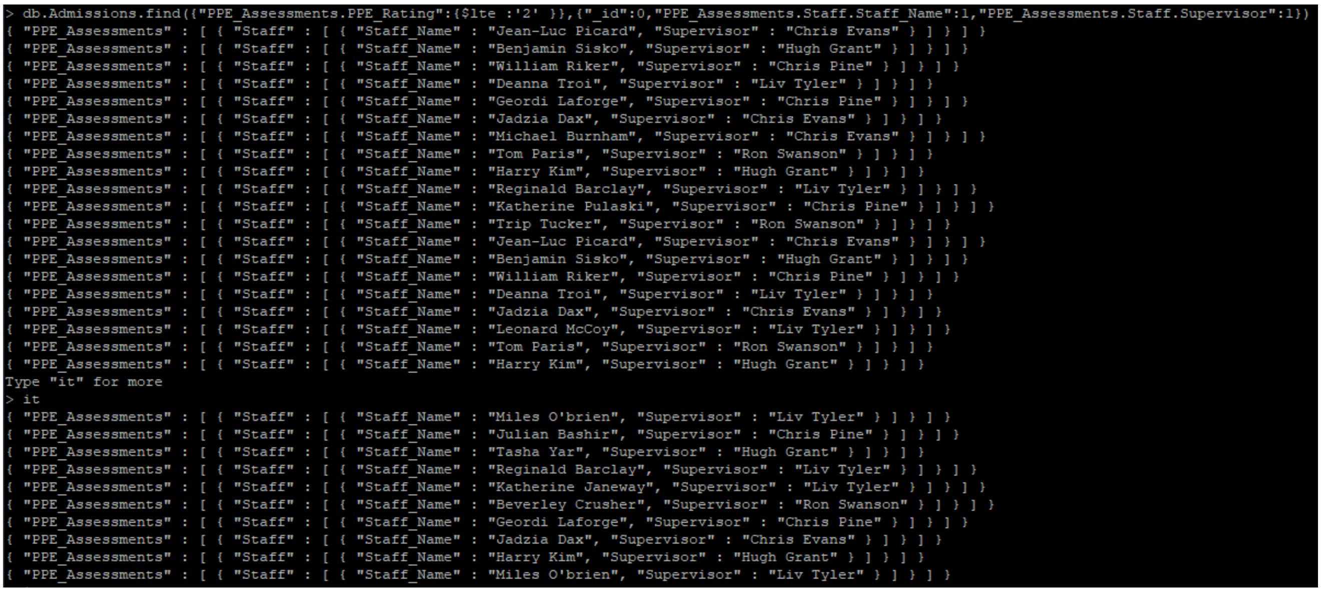
This query is more complex in that it searches through multiple layers of nested documents within the Admissions collection. The criteria it is searching for is where a PPE assessment has occurred and a rating of 2 or lower has been recorded. Then the query returns just the staff name and supervisor name for the matching assessments.

Code

The command to run the query is

```
db.Admissions.find
      (
      {"PPE_Assessments.PPE_Rating":{$lte:'2'}},
      {"_id":0,"PPE_Assessments.Staff.Staff_Name":1,
      "PPE_Assessments.Staff.Supervisor":1}
      )
```

Results



```
> db.Admissions.find({"PPE_Assessments.PPE_Rating":{$lte :'2' }},{"_id":0,"PPE_Assessments.Staff.Staff_Name":1,"PPE_Assessments.Staff.Supervisor":1})
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Jean-Luc Picard", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Benjamin Sisko", "Supervisor" : "Hugh Grant" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "William Riker", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Deanna Troi", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Geordi Laforge", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Jadzia Dax", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Michael Burnham", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Tom Paris", "Supervisor" : "Ron Swanson" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Harry Kim", "Supervisor" : "Hugh Grant" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Reginald Barclay", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Katherine Pulaski", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Trip Tucker", "Supervisor" : "Ron Swanson" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Jean-Luc Picard", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Benjamin Sisko", "Supervisor" : "Hugh Grant" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "William Riker", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Deanna Troi", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Jadzia Dax", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Leonard McCoy", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Tom Paris", "Supervisor" : "Ron Swanson" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Harry Kim", "Supervisor" : "Hugh Grant" } ] } ] }
Type "it" for more
> it
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Miles O'brien", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Julian Bashir", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Tasha Yar", "Supervisor" : "Hugh Grant" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Reginald Barclay", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Katherine Janeway", "Supervisor" : "Liv Tyler" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Beverley Crusher", "Supervisor" : "Ron Swanson" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Geordi Laforge", "Supervisor" : "Chris Pine" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Jadzia Dax", "Supervisor" : "Chris Evans" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Harry Kim", "Supervisor" : "Hugh Grant" } ] } ] }
{ "PPE_Assessments" : [ { "Staff" : [ { "Staff_Name" : "Miles O'brien", "Supervisor" : "Liv Tyler" } ] } ] }
```

Section six - Report

Introduction

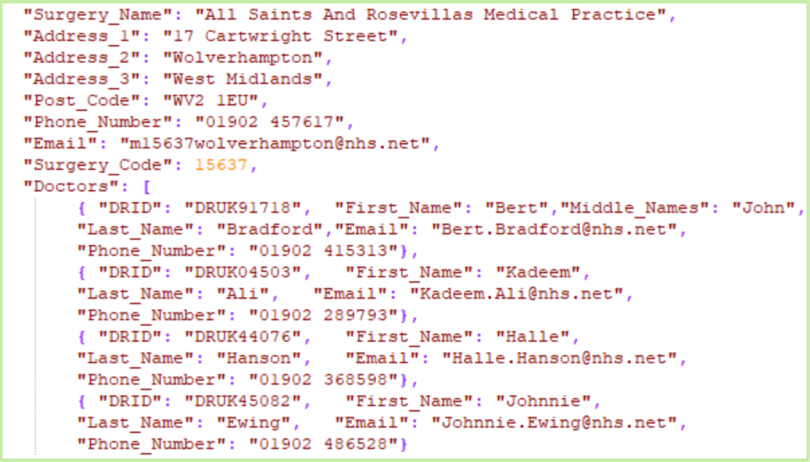
Within the scope of this assignment a given scenario involving Hospitals, Doctors, GP Surgeries and Patients among other related entities has been implemented in both a relational DBMS (Database Management System) and similarly in a NoSQL DBMS. Each of the two approaches has aimed to follow the specification and requirements set out within the scenario. The choice of DBMS used in the relational enactment was Oracle via their SQL Developer software while, the choice of NoSQL DBMS used was the shell version 3.2 of MongoDB.

Relational Design

The scenario called for quite detailed and usually related information to be stored such as patient records, consultations with a doctor, COVID tests and their subsequent results, hospital admissions, PPE assessments. The first step was to identify the tables, and attributes needed. In total nine tables were identified with almost fifty unique attributes. Following this, the primary keys and foreign keys were established along with choices made regarding the corresponding relationships and cardinalities, for example, a surgery has a one-to-many relationship with its patients, one surgery can have many patients while a patient can only have one surgery. Similarly, another relationship that was identified that required the creation of a link entity was that of the Consultation. A doctor carries out many consultations and a patient can have multiple consultations. In addressing this, a Consultation table was created to reduce the many-to-many relationship into two individual one-to-many relationships.

NoSQL Design

Chronologically the formation of the NoSQL based implementation of the Central Hospital Scenario, took place after the relational implementation. This was of some benefit in that the attributes and entities had already been identified earlier. However, it would not have been a practical approach to simply create a collection within the MongoDB Shell for each of the relational tables similarly creating a document for each record. Instead, making use of one of the advantages of a NoSQL approach the number of relational tables was reduced from nine down to just three collections by way of merging key attributes within documents. The consultation table and COVID test table were added as array elements to a patient document. By storing them within the patient record as key value pairs it allowed for dynamic resizing of the document. Similarly, as shown below, the separate Doctors table and the Surgeries table were also merged into a single collection where the Doctors that belonged to a Surgery existed in the Surgery document as an array of key value pairs.



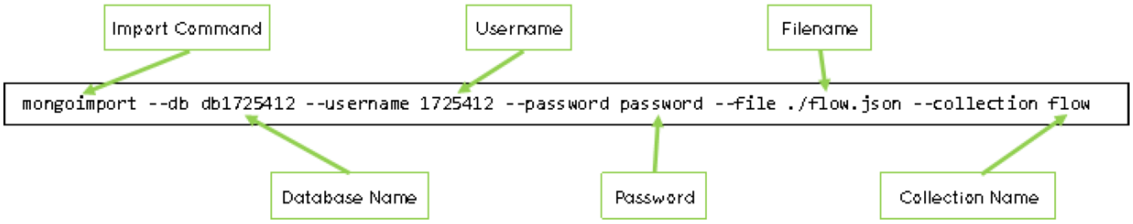
```
"Surgery_Name": "All Saints And Rosevillas Medical Practice",
"Address_1": "17 Cartwright Street",
"Address_2": "Wolverhampton",
"Address_3": "West Midlands",
"Post_Code": "WV2 1EU",
"Phone_Number": "01902 457617",
"Email": "m15637wolverhampton@nhs.net",
"Surgery_Code": 15637,
"Doctors": [
     { "DRID": "DRUK91718",  "First_Name": "Bert","Middle_Names": "John",
     "Last_Name": "Bradford","Email": "Bert.Bradford@nhs.net",
     "Phone_Number": "01902 415313"},
     { "DRID": "DRUK04503",   "First_Name": "Kadeem",
     "Last_Name": "Ali",   "Email": "Kadeem.Ali@nhs.net",
     "Phone_Number": "01902 289793"},
     { "DRID": "DRUK44076",   "First_Name": "Halle",
     "Last_Name": "Hanson",   "Email": "Halle.Hanson@nhs.net",
     "Phone_Number": "01902 368598"},
     { "DRID": "DRUK45082",   "First_Name": "Johnnie",
     "Last_Name": "Ewing",   "Email": "Johnnie.Ewing@nhs.net",
     "Phone_Number": "01902 486528"}
```

Oracle Implementation

The execution of the relational DMBS was relatively simple and straightforward. The sample data used was generated across several spreadsheet worksheets, in the same form as the tables to which it would be populating. Then using the built-in features of the SQL Developer tool, it was possible to import the sample data from the spreadsheet and create the necessary tables. By utilising this approach, it was also possible to perform some low-level data cleansing insofar that the data generated in the spreadsheets was able to be done so that there would be no incorrect dates or values. Following the basic data import and table creation, it was then possible to set the relationships and both primary and foreign keys within the software. As part of the assessment, it was required to query the data as well as generate triggers and materialised views. All of these steps were accomplished without issue. Additionally, other more obscure queries were run to ensure the tables could be joined with outer joins, inner joins and full joins and still return the correct data when queried.

MongoDB Implementation

Unfortunately, the implementation within MongoDB was not quite as straightforward as with the SQL Developer software. MongoDB utilises key value pairs to make up the data being stored. In order to import data and create a collection using the MongoDB shell you would typically type a single command made up of key elements shown below.



```
mongoimport --db db1725412 --username 1725412 --password password --file ./flow.json --collection flow
```

This requires that the JSON file you are trying to import is correctly formatted and contains the correct syntax. There are multiple examples of the JSON files used earlier in this document.

Once the collections had been created and populated, it was possible to test the data and generate numerous queries and corresponding results. For grouped results this was done by means of what is known as the aggregation pipeline.

Reflective Advantages & Disadvantages

Certainly, when reviewing the process of initial database creation, the SQL Developer software offered a strong advantage over the MongoDB shell. This was clearly manifest in the ease of user experience when importing data. The SQL Developer software gave clear informative errors when they occurred, usually displayed in a window. Whereas with MongoDB, when importing data, if an error occurred, usually down to a missing bracket, then you would receive a vague error, sometimes without a line number or column identification for debugging purposes. Alternatively, it is not entirely fair to say that MongoDB is not as straightforward as its relational counterpart. A correctly formatted collection of considerable size, several hundred thousand documents, can be imported in a very short timeframe with a single command line entry (shown above).

The relational DBMS implementation of the Central Hospital case study showed advantages in the readability of the output of query results. From a legibility perspective the relational records themselves can objectively be easier to read on the page, especially when you consider that you are able to output specific columns and rename them easily.

A powerful advantage offered by the NoSQL MongoDB approach is the horizontal flexibility offered, it is possible to add additional attributes to a single record without having to completely redesign the entire schema, this coupled with the object-oriented approach allows a user to nest objects with individual attributes inside other records. An example of this can be seen in the Admissions collection of the Central Hospital case study. Within an admissions record, there is an array of PPE Assessment objects, then within each PPE Assessment object there is another Staff object containing pertinent information about the member of staff. This information is accessed via the common object-oriented navigation method of outer/parent element preceded by a dot then followed by the inner/child object or element.

```
“db.admissions.record.ppe_assessment.staff.supervisor”
```

With the advantages listed above, both for the relational approach and NoSQL approach, it is worth noting that one of the key reasons the advantages have been identified is that they are disadvantages in the alternate implementation. The NoSQL MongoDB query results and records are not easy to read from a user experience perspective. Also, the schema restrictions in place around a relational DBMS can be quite limiting and require extensive reworking should the systems users decide that some element of new data is required to be stored.

Data security and availability

There are several perspectives with which you can explore the security of the data being held in each of the two implementations. From a higher level, each system requires that a user accessing the DBMS has a valid account to do so. This account would typically comprise of a username and password, the identifiable weakness being in the user’s choice of password, or perhaps not logging out of the system when leaving the workstation that they are using to access the DBMS. Looking at the data security from a lower level, i.e., a record or attribute basis with a relational approach by means of use of foreign keys it has a stronger level of security in cases of record deletion. Ultimately allowing for greater referential integrity rules to be implemented and adhered to. Conversely, the NoSQL MongoDB implementation, does not offer as many varied tools to the database administrator to enforce the desired business rules for the stakeholders. However, it is possible to have triggers and these would appear to be the primary tool available.

Both DBMS solutions make use of locking to ensure concurrency of data, with record or documents being locked at various levels whilst operations and transactions are being carried out. Equally, both allow for a read only lock option where other users can view but not amend the specific locked record or document.

Conclusion and Recommendation

Looking at a scaling this scenario out to a size that includes 51,000+ patients and greater than 565,150 infection data points each year, the limits of data were explored in each of the two proposed solutions. Neither Oracle or MongoDB would have any significant issues in retaining and processing the proposed volume of data, so from that perspective there is no inference in a preference to be observed. The next consideration is from a data querying and reporting viewpoint, Oracle undoubtedly has the advantage here offering a greater degree of flexibility in report generation. In query processing times both systems are similar, perhaps with MongoDB having a slight advantage. However, if this perceived increase in query runtime is something that a user would be aware off is something that would require further research.

Overall, based on the work completed the relational DBMS would be recommended for scaling the given Central Hospital scenario into an operational system.