Toby bluck

BLU16274423

Word Count

1863

SCDT44 CW2

DATABASE SYSTEMS

Table of Contents

[Database requirements: 1](#_Toc48772788)

[User requirements: 2](#_Toc48772789)

[Functional requirements for the database: 2](#_Toc48772790)

[Performance requirements: 2](#_Toc48772791)

[DBMS – 3](#_Toc48772792)

[MySQL 3](#_Toc48772793)

[mongoDB 3](#_Toc48772794)

[mariaDB 3](#_Toc48772795)

[Queries - 3](#_Toc48772796)

[Rationale choice 4](#_Toc48772797)

[Normalisation – 5](#_Toc48772798)

[Original dataset: 5](#_Toc48772799)

[Staff: 5](#_Toc48772800)

[Patient: 5](#_Toc48772801)

[Appointment: 5](#_Toc48772802)

[1NF: 6](#_Toc48772803)

[2NF: 6](#_Toc48772804)

[3NF: 6](#_Toc48772805)

[Process: 7](#_Toc48772806)

[Database Design: 7](#_Toc48772807)

[ERD- 7](#_Toc48772808)

[DFD- 8](#_Toc48772809)

[Data implementation: (exporting of the SQL) 9](#_Toc48772810)

[Queries: 14](#_Toc48772811)

# Database requirements:

Needs to record the following-

* Name of staff
* Staff position
* Area of expertise
* Patients they have
* Patient names
* Patient address
* Phone number
* Insurance number
* Appointment ID
* Patient name
* Nurse attending
* Doctor/physician attending
* Procedures carried out
* Cost of procedure
* Date and start and end of appointment
* Total time
* Room# used
* Available/ unavailable
* Medication & description & brand

## User requirements:

GUI   
Be able to look at all the records   
Sort the record by A-Z, Date, Cost, or numerical order.   
Input queries to get specific data out

## Functional requirements for the database:

Accessibility to data- the data stored within the database should be accessible to all users. However, this should be based on a read only basis. All systems on the network should be able to access the network. Furthermore, the data should be able to be exported into an excel format or similar.

Integrity of data- data should be validated on a regular basis to find any changes to appointment times or room No. or any other factor that could be changed. New data should be validated by other users to ensure correctness of details and that any new users using the database should be verified and old users have permissions revoke. (Hunt et al., 2016)

## Performance requirements:

Optimization- optimization of the tables configuration, this can be done through normalisation, distribution of the data into correct fields to ensure integrity is maintained, also generalised configuration of the database can be done to help improve the experience.

(Defining Database Performance, n.d.)

Scalability- the design of the database should keep scalability in mind, this ties into DBMS as management systems can affect the overall scalability of the solution depending on the amount of data being stored. For the scalability of this database, essential requirements for performance would be the availability and accessibility 24/7.

(Designing and Developing for Performance, n.d.)

# DBMS –

## MySQL

MySQL is an open source and will be used as the RDBMS (Relation Database Management System) this allows multiple tables to be used which can be used for referential integrity across the database itself. Furthermore, support for SQL queries allows for relevant queries to pull information requested from all tables and display them. (MySQL - Introduction, n.d.)

Security – overall MySQL is extremely secure and flexible and is renown in the industry for it and its high scalability. (McLean, 2017)

Performance – able to handle large amounts of data and also using queries to pull results is extremely quick. (McLean, 2017)

## mongoDB

mongoDB is an open source RDBMS it differs slightly from MySQL but in principle similar. This functions on using documents and collections opposed to rows and tables. This is used to create databases which is able to handle a vast amount of data which can be varying and complex data structures. Due to its high scalability and open source it is a solid option. Due to the way the data is stored and formatted it is retaining its natural form and also is readable for users of the database. (Why Use MongoDB & When to Use It?, 2020) (MongoDB - Overview, n.d.)

Performance – mongoDB once utilised correctly has outstanding performance this is partly due to it being able to handle unstructured data easily. In comparison to MySQL, mongoDB is considerably faster with select, update and insert statements. (Shah and Shah, 2019)

## mariaDB

mariaDB this uses MySQL as a backbone to create a RDBMS, this is a solid choice as many of the original developers of MySQL created MariaDB. This is fundamentally MySQL but with additional features stacked on top of it. Due to the popularity of SQL this easy to use as it uses the same querying as SQL, furthermore the overall support to run on all different forms of OS and also varying programming languages as well as storage engines allows unparalleled flexibility. (About MariaDB Server - MariaDB.org, n.d.), (MariaDB - Introduction, n.d.)

Performance – overall MariaDB focuses on amplifying the speed in comparison of MySQL which is proven true. It can handle both uncompressed and compressed data within a database and other forms of information slightly to considerably faster than MySQL dependant on what it is being utilised for. MariaDB has provided charts to show the performance comparison. (Lindstrom, 2014)

Security – solid encryption of DB and management of encryption keys allows for varying in encryption choices uses. Access/role based control allows for management of the distribution of data to users or roles available allowing for a solid confidentiality on information.

## Queries -

Depending on the DBMS chosen they have varying capabilities with handling queries and the formatting of them. MySQL and MariaDB with be fundamentally similar due to MariaDB having many of the original SQL developers.

With MongoDB not using SQL like the other DBMS the queries are constructed differently,

SQL uses SELECT \* appointment for selecting everything while MongoDB would use db.appointment.find( {} ) although these are different formulated queries, if the syntax is correct MongoDB is incredible fast opposed to MySQL but it is less readable than the simple querying of mySQL. Overall MongoDB for updating is approximately 50% faster than MySQL for updating and inserting into the database MongoDB clearly out preforms MySQL.

However, this is only done by using large amounts of data for the performance this was carried out with 5000 inserts into a table on each DBMS. The results will change when scaling the amount of data being inserted or updated.

(Gyorodi et al., 2015)

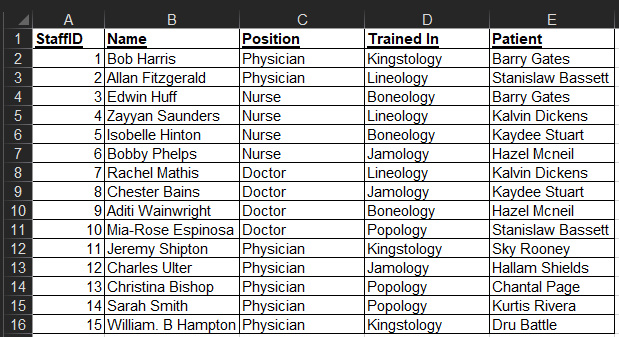
## Rationale choice

Overall, throughout the comparison and analysis of the DBMS and query evaluation. MySQL will be the optimum choice for the simplicity of queries as well as large support and solid security when being implemented and solid performance on the board overall.

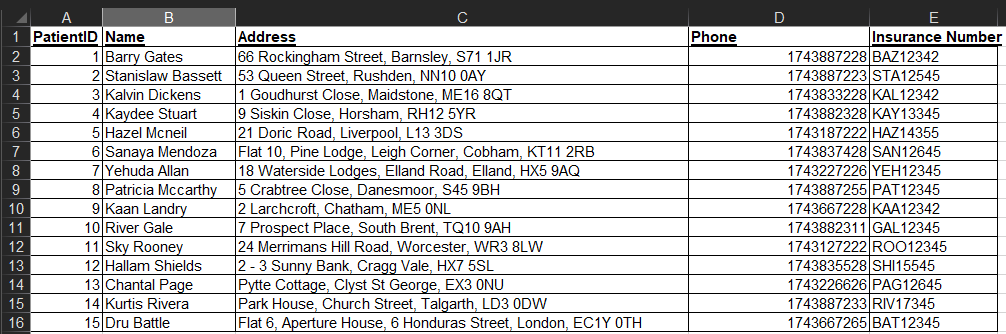
# Normalisation –

## Original dataset:

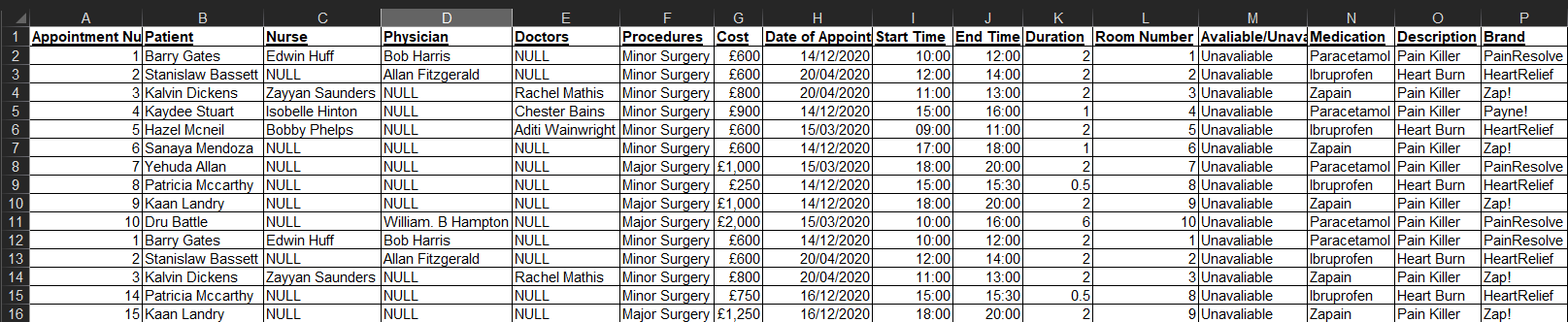
### Staff:



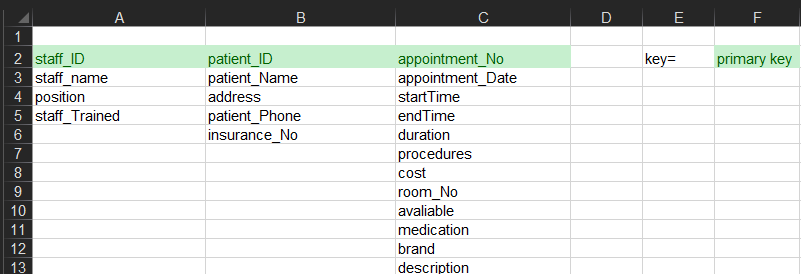
### Patient:



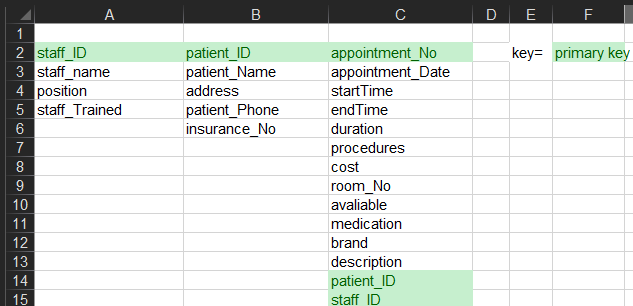
### Appointment:



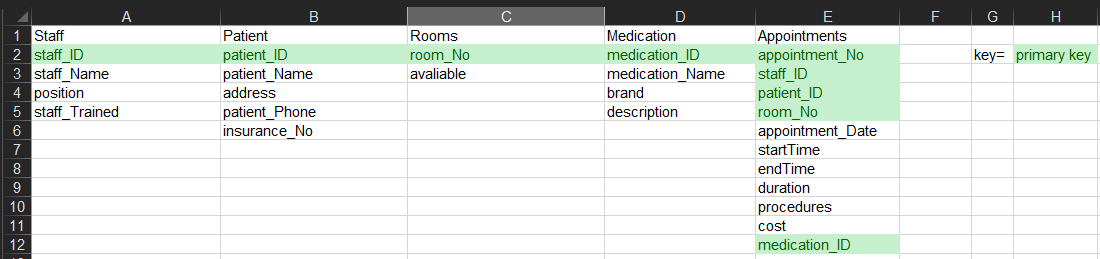
## 1NF:



## 2NF:



## 3NF:



## Process:

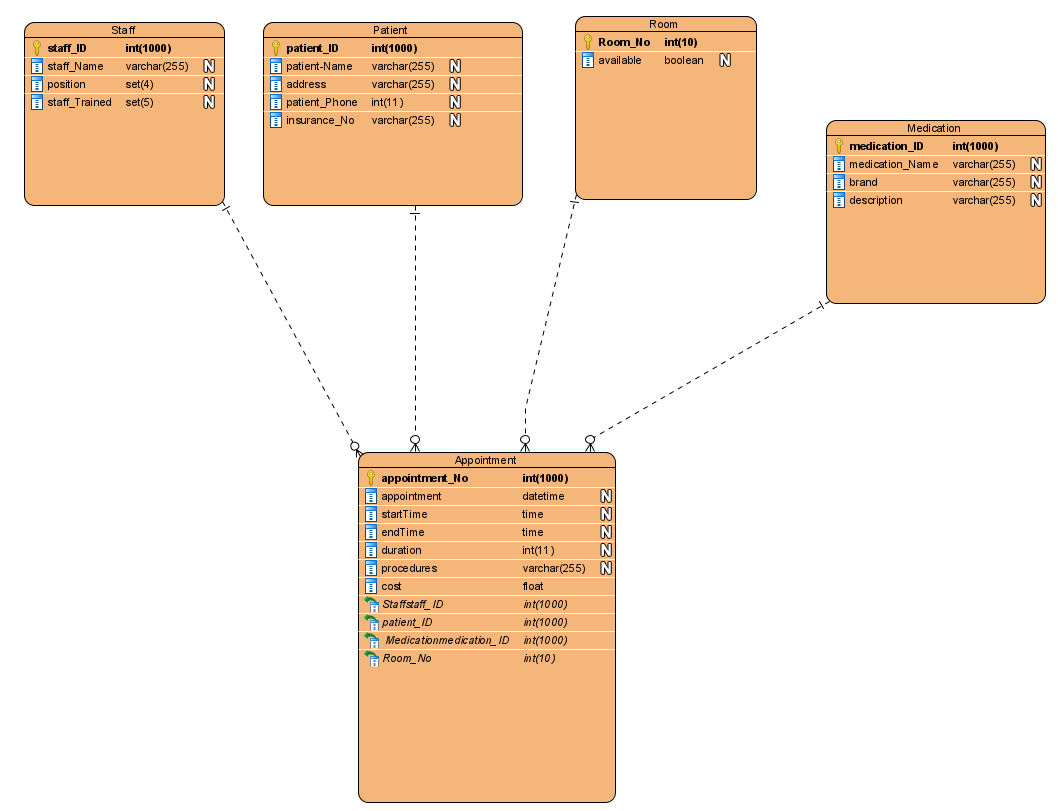
To create the 1NF the dataset needs to be analysed and broken down the dataset. They should only contain a single value and the names of all columns should have a unique name.

For the 2NF it must firstly be 1NF then allocate primary keys to each table that are unique identifiers which will remove any partial dependency in the 2NF staff\_ID, patient\_ID and appointment\_ID can be used to identify all information within the dataset.

Then for 3NF the 2NF was modified by changing breaking the dataset up so that any form of unique identifications would have their own table, which was then linked to the appointment tables which remove any non-prime attributes relying on each other.

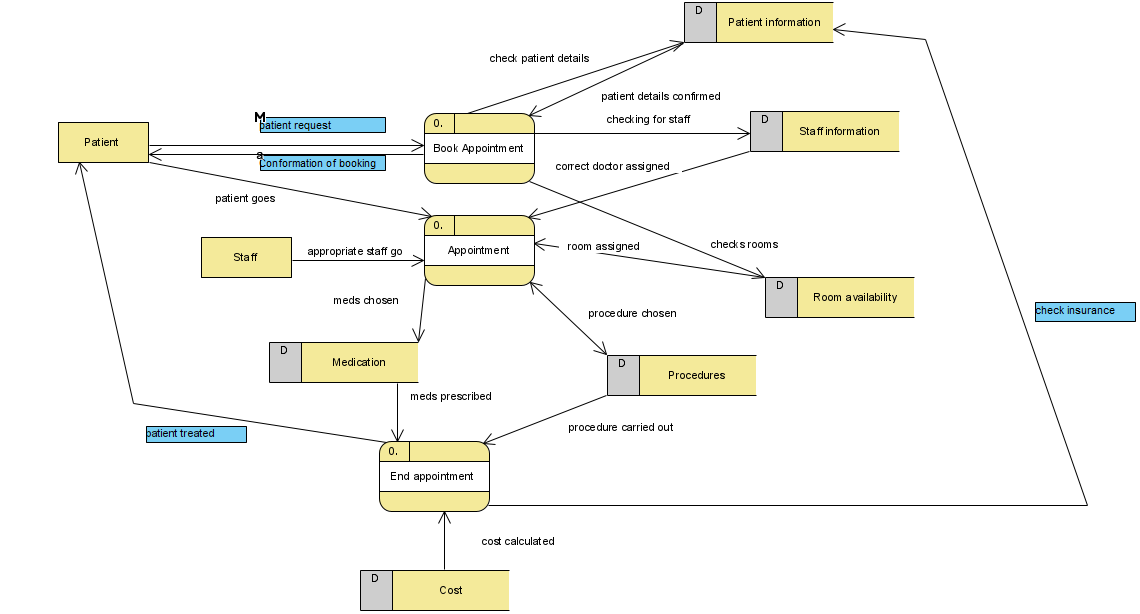
# Database Design –

## ERD:

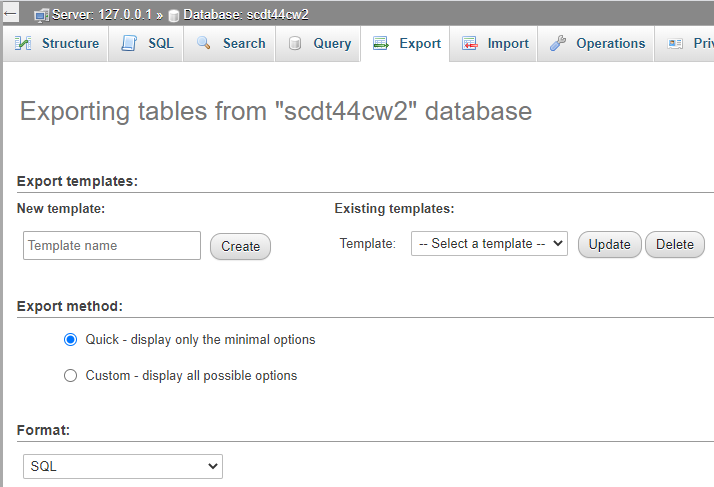


This ERD shows the relationships between the tables along the entities and the attributes they have. Each entity has a primary key which will be the unique identifier for the database and within the appointment entity there are foreign keys that are derived from the entities that are connected to them which uses the information from the relevant key its attributed to. The relationship within these entities is that there one to many, one staff, patient etc. to many appointments. (What is Entity Relationship Diagram (ERD)?, n.d.)

## DFD:

  
The DFD shows the traversing of the data within the database. This diagram shows the data that is stored, physical entities and the processes involved. For DFDs there are typically 3 levels, each level expands and demonstrates better breakdowns of the flow of information and a more detailed design however, there is more than 3 levels although they are less common. This DFD closely resembles a level 1 DFD which illustrates the flow of information which is specific to the logical workings of appointments. This could be made into a level 2 DFD and go into high detail on all the exchanging information but the complexity of a level 2 DFD does not match the overall complexity of the requirements or data. This DFD staff and patients are entities which are the source/destination of information. The rectangle with the D in grey represents the data store where information is stored. The rounded boxes with 0. Are called a process which is a function or action used on the data. The arrows represent the flow of data within this diagram. (What is Data Flow Diagram (DFD)? How to Draw DFD?, n.d.)

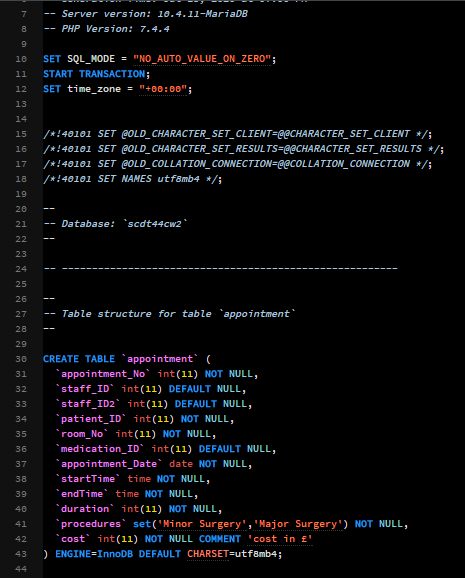
# Data implementation: (exporting of the SQL)



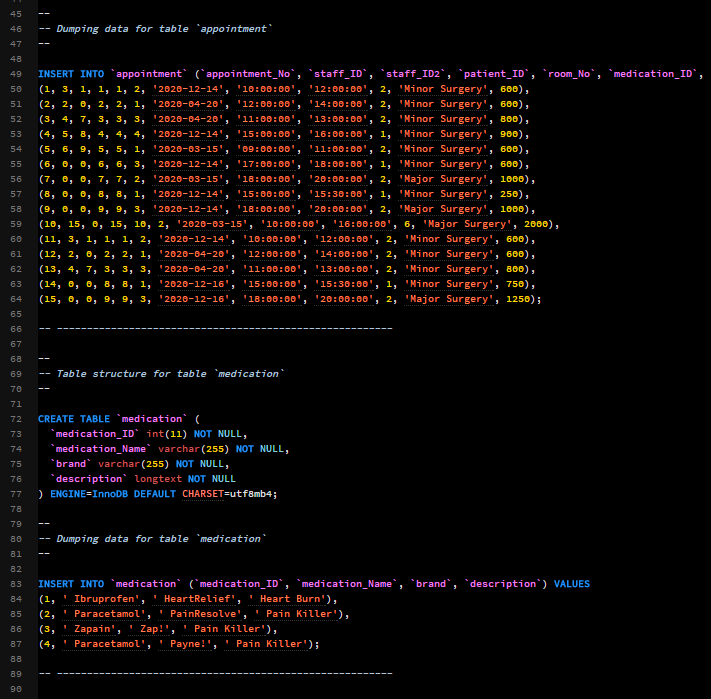
Exporting the whole database in SQL format from myPHPadmin.



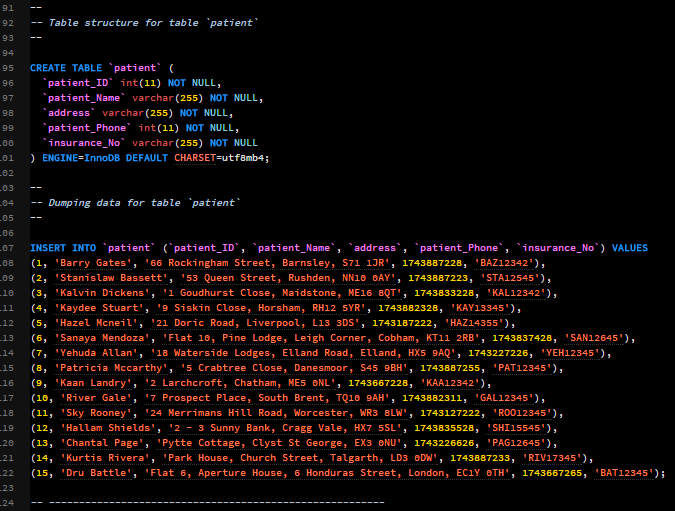
The file that had been created from the exporting of the database.



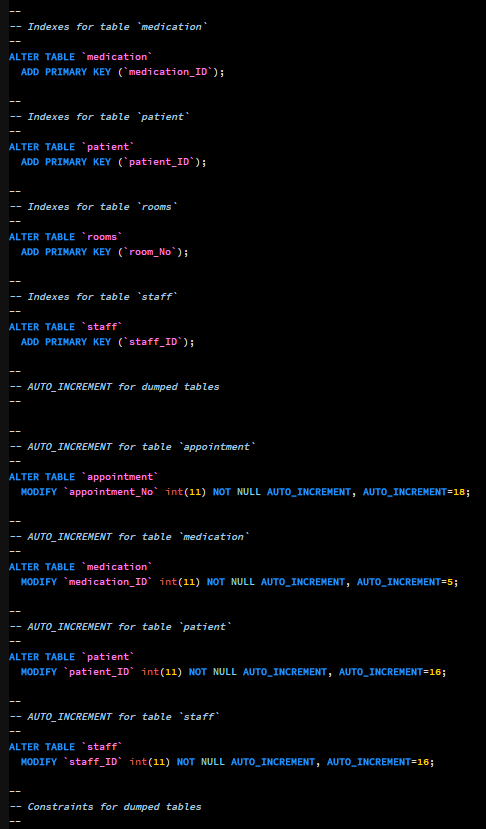
The exported SQL which has been opened in Brackets.



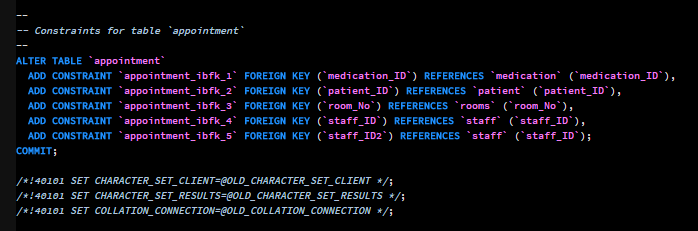
This shows the SQL code to structure the tables and the information which is presently stored in them at the time of the export.



This shows the structure and the information that has been inserted into the patients table.



Creation of primary keys for the tables as well auto-incrementing for them.



This is for the appointments table which is the creation of foreign keys to link tables i.e. patients to the appointment table to get the necessary information when finding information by using queries.

# Queries:

Using SQL queries allows you to find specific information within a database using the correct parameters to find the relevant data. Queries allow you to search one or more tables for any information as long as the formatting and syntax of the query is correct.

1. Query design:



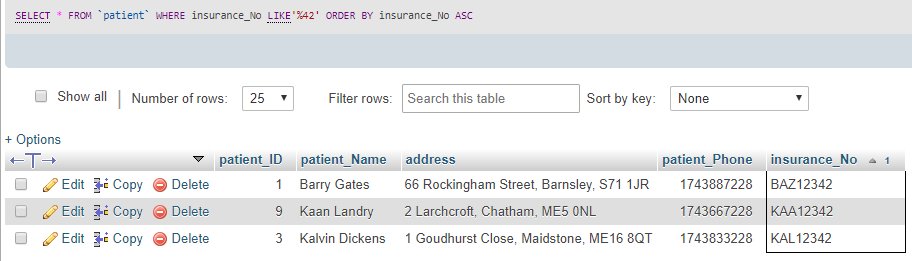
1. Query testing:

  
This query is being used to find the patient names and insurance numbers and then ordering them by ascending order

1. Query design:



1. Query testing:

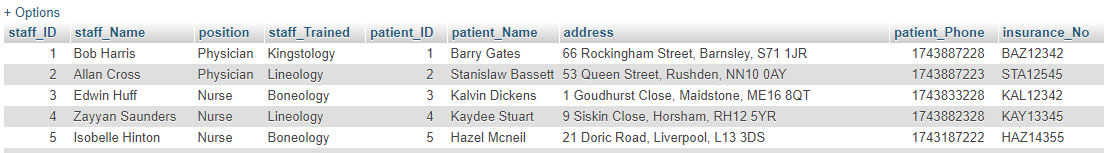


This query finds everyone is the patient table where the parameter is the insurance number/s that end in 42 (%42) then order the insurance numbers in ascending order. (SQL LIKE Operator, n.d.)

1. Query design:

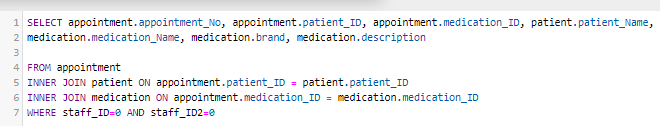


1. Query testing:

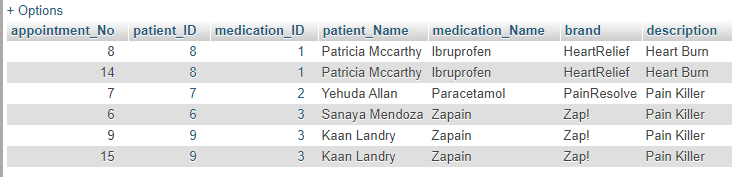


This query allows for all staff table and patient table to be linked together through INNER JOIN which allow for records with the dame values to be linked together in a query. (SQL INNER JOIN Keyword, n.d.)

1. Query design:



1. Query testing:

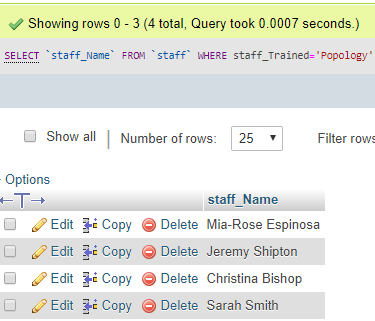


This query uses the appointment table and joins the medication and patient table using INNER JOIN and the parameter is where there is no value/ null value in the staff ID on the appointment. This allows for the appointment ID, patient ID, medication ID and the relevant information from the medication table. The results show the prescriptions of patients that have not had an appointment with a staff member.

1. Query design:



1. Query testing:



This query just displays the staff names which have been trained in the “Popology” department.

1. Query design:



1. Query testing:

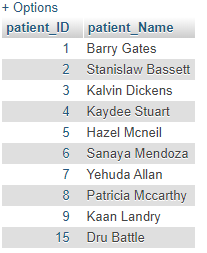


This query shows the updating of information without going into the table to manually find the row/s, UPDATE means that a record is going to be modified, SET is used to specify what will be changed in the row/s where the ID is equal to i.e. 3 etc. (SQL UPDATE Statement, n.d.)

1. Query design:

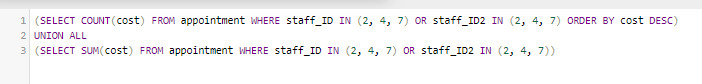
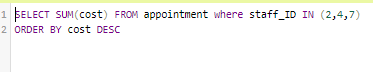


1. Query testing:

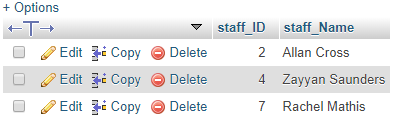


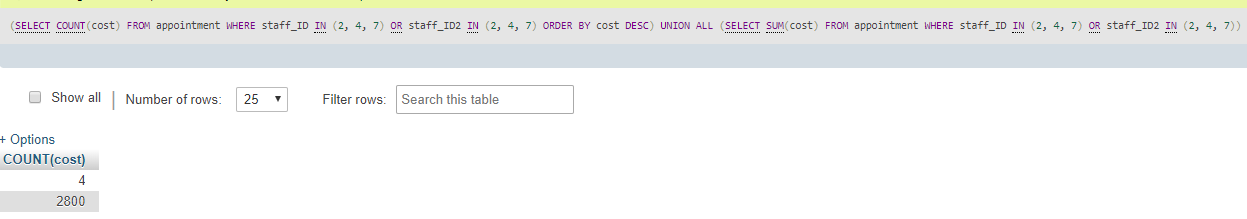
This query is showing patient IDs and names that where the staff ID is a value that isn’t null or 0. SELECT DISTINCT allows for only a selection of different values, none of them being the same, WHERE NOT allows for statements that are not true/false.

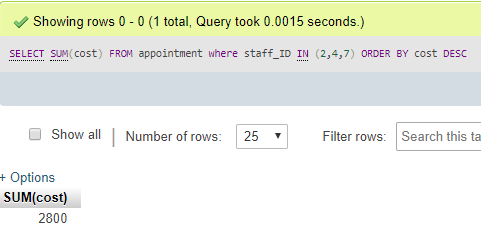
(SQL AND, OR, NOT Operators, n.d.), (SQL SELECT DISTINCT Statement, n.d.)

1. Query design:
2. 
3. 
4. 

8. Query testing:

a. 

b. 

c. 

This query is broken down into 3 parts but could be improved by using sub-queries, part a. COUNT is used to find the specific amount of row using parameters to identify them which is staff IDs relating to Lineology , alongside being ordered in descending order. (SQL COUNT(), AVG() and SUM() Functions, n.d.)

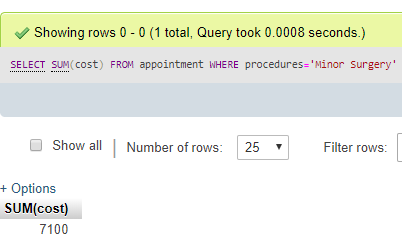
Part b. is using UNION ALL combines the values of multiple SELECT statements, which combines part a with a SUM(cost) which calculates the total of the column with a numeric value.

Part c. is the SUM(cost) which is the total amount on its own from the select values.

1. (SQL UNION, UNION ALL, n.d.)Query design:

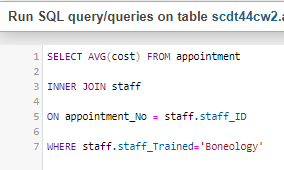


1. Query testing:



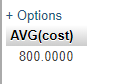
This query shows the total cost of all appointments in which the procedure carried out is “Minor Surgery” and gives the whole value in a single row.

1. Query design:



By default, SQL is organised by ascending order

1. Query testing:



This query is using the AVG(cost) which takes all the values from the cost column in the appointments table, which is joined to the staff table to find staff that has been trained in “Boneology” as a parameter. This allows for the staff ID to be found which is used for the appointment No which find all the relevant numeric values in the cost column.

# Reference/s –

Docs.oracle.com. n.d. Designing And Developing For Performance. [online] Available at: <https://docs.oracle.com/database/121/TGDBA/pfgrf\_design.htm#TGDBA94088>.

Gpdb.docs.pivotal.io. n.d. Defining Database Performance. [online] Available at: <https://gpdb.docs.pivotal.io/43250/admin\_guide/perf\_intro.html>.

Gyorodi, C., Gyorodi, R., Pecherle, G. and Olah, A. (2015) A comparative study: MongoDB vs. MySQL, 2015 13Th International Conference On Engineering Of Modern Electric Systems (EMES), .

Hunt, N., Renner, C., Embry, B., Wright, D., Myers, J., Arias, L., Barron, J., Webb, P., Prentiss, R. and Rodis, J., 2016. FUNCTIONAL And TECHNICAL REQUIREMENTS DOCUMENT FDP Expanded Clearinghouse Pilot Phase 2 – Web-Based System. [ebook] pp.4-7. Available at: <https://sites.nationalacademies.org/cs/groups/pgasite/documents/webpage/pga\_179129.pdf>.

Lindstrom, J. (2014) Performance evaluation of MariaDB 10.1 and MySQL 5.7.4-labs-tplc - MariaDB.org, Mariadb.Org, [Online]. Available at <https://mariadb.org/performance-evaluation-of-mariadb-10-1-and-mysql-5-7-4-labs-tplc/#:~:text=When%20client%20threads%20are%20between,MySQL%20performance%20for%20uncompressed%20tables>.

MariaDB.org. n.d. About Mariadb Server - Mariadb.Org. [online] Available at: <https://mariadb.org/about/>.

McLean, M., 2017. 8 Advantages Of Using Mysql. [online] DevOps.com. Available at: <https://devops.com/8-advantages-using-mysql/>.

MongoDB. 2020. Why Use Mongodb & When To Use It?. [online] Available at: <https://www.mongodb.com/why-use-mongodb#:~:text=Companies%20and%20development%20teams%20of,of%20both%20data%20and%20traffic.>.

Shah, M. and Shah, H., 2019. Mongodb Vs Mysql: A Comparative Study On Databases. [online] Simform.com. Available at: <https://www.simform.com/mongodb-vs-mysql-databases/#:~:text=MongoDB%20vs%20MySQL%3A%20Performance%20%26%20Speed,is%20more%20sensitive%20to%20workload.>.

SQL AND, OR, NOT Operators (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_and_or.asp>.

SQL COUNT(), AVG() and SUM() Functions (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_count_avg_sum.asp>.

SQL INNER JOIN Keyword (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_join_inner.asp>.

SQL LIKE Operator (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_like.asp>.

SQL SELECT DISTINCT Statement (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_distinct.asp>.

SQL UNION, UNION ALL (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_ref_union.asp>.

SQL UPDATE Statement (n.d.) W3schools.Com, [Online]. Available at <https://www.w3schools.com/sql/sql_update.asp>.

Tutorialspoint.com. n.d. Mariadb - Introduction. [online] Available at: <https://www.tutorialspoint.com/mariadb/mariadb\_introduction.htm>.

Tutorialspoint.com. n.d. Mongodb - Overview. [online] Available at: <https://www.tutorialspoint.com/mongodb/mongodb\_overview.htm>.

Tutorialspoint.com. n.d. Mysql - Introduction. [online] Available at: <https://www.tutorialspoint.com/mysql/mysql-introduction.htm>.

What is Data Flow Diagram (DFD)? How to Draw DFD? (n.d.) Visual-Paradigm.Com, [Online]. Available at <https://www.visual-paradigm.com/tutorials/data-flow-diagram-dfd.jsp>.

What is Entity Relationship Diagram (ERD)? (n.d.) Visual-Paradigm.Com, [Online]. Available at <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/>.