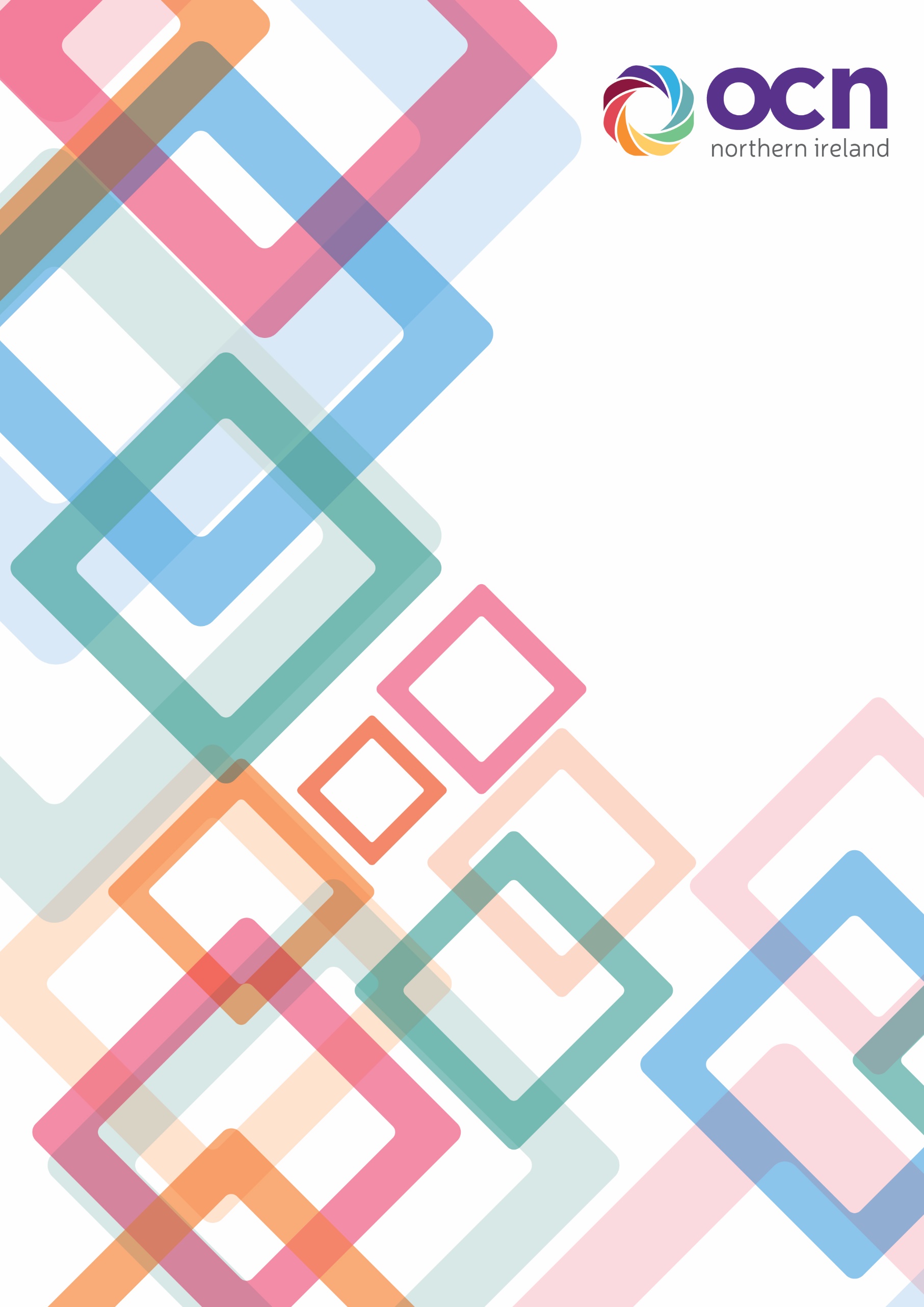
****

**Structured Query Language (SQL) Fundamentals**

**(Y/617/8669)**

**Learner Assessment  
Booklet**

**Learner Assessment Booklet**

**Level 4 Diploma in Software Testing**

**Unit:** Structured Query Language (SQL) Fundamentals

**Credit Value:** 13

**Unit Reference Number:** (Y/617/8669)

***This form is used to record and confirm that assessment has taken place and the learner has achieved the unit detailed on pages 5 & 6***

|  |  |
| --- | --- |
| **Learner Name:** | Ryan McKee |
| **Assessor name:** |  |
| **Internal Verifier name:** |  |
| **Course:** |  |
| **Course code:** |  |
| **Completion Date:** |  |

|  |
| --- |
| **Achievement of unit is confirmed: Yes/No**  **Signature of Assessor:**  **Date:** |

|  |
| --- |
| **I certify that all the work in this booklet is my own.**  **Learner Signature:**  **Date:** |

**Assessment Grid: *Completed by Assessor***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Learning Outcomes** | **Achieved**  **(Delete)** | **Criteria** | **Booklet Page** | **Assessment Date**  **(If Applicable)** | **Date of Verification**  **(If Applicable)** |
| **1** | Yes/No | 1.1  1.2  1.3  1.4 | **7**  **8**  **9 – 12**  **13** |  |  |
| **2** | Yes/No | 2.1  2.2  2.3 | **14**  **15**  **16** |  |  |
| **3** | Yes/No | 3.1  3.2  3.3 | **17 – 18**  **19 – 22**  **23 – 24** |  |  |
| **4** | Yes/No | 4.1  4.2  4.3  4.4 | **25 – 28**  **25 – 28**  **25 – 28**  **25 – 28** |  |  |
| **5** | Yes/No | 5.1  5.2  5.3 | **29 – 33**  **34 – 35**  **36 – 37** |  |  |
| **6** | Yes/No | 6.1  6.2  6.3 | **38 – 39**  **40**  **40** |  |  |
| **7** | Yes/No | 7.1  7.2  7.3  7.4  7.5 | **9 – 12**  **11**  **12**  **12**  **41 – 44** |  |  |
| **8** | Yes/No | 8.1  8.2 | **45 – 50**  **45 - 50** |  |  |

***Note to Assessors:***

***This assessment record book is a template that you may choose to use to record summative assessment activities. It is intended to offer a framework for recording summative assessment. It is not intended to be prescriptive about assessment tasks. Assessors should amend the assessments and the booklet accordingly, taking account of the needs of their specific learner group and any opportunities for using combined assessments that cover multiple assessment criteria or learning outcomes.***

**Assessment record: *Completed by Assessor***

|  |
| --- |
| **Assessor Assessment decisions:** |

**Signature of Assessor:**

**Date:**

**All the following Learning Outcomes and Assessment Criteria MUST be met:**

**This unit has eight learning outcomes**

|  |  |
| --- | --- |
| **Learning Outcomes** | **Assessment Criteria** |
| **The learner will:** | **The learner can:** |
| 1. Be able to perform database operations. | * 1. Explain with examples what is meant by the concept of a table and how it is used to work with data.   2. Explain what a Relational Database Management System (RDMS) is and the main types of databases available.   3. Explain using examples the Create, Retrieve, Update and Delete (CRUD) operations available within a database.   4. Explain and use different types of SQL language including Data Manipulation Language (DML), Data Control Language (DCL) and Data Definition Language (DDL) to perform database operations. |
| 1. Be able to create an RDMS composed of tables. | * 1. Explain the purpose of database normalisation and the forms available.   2. Explain the concept of primary, foreign and composite keys   3. Create an RDMS composed of tables to include:   a) database normalisation to the third normal form b) creation of appropriate keys and indices within tables to form relationships |
| 1. Be able to develop an RDMS within a database server. | * 1. Explain the main data types, why they are important and how they affect database storage.   2. Interpret and create an Entity Relationship Diagram (ERD) depicting tables and relationships.   3. Develop and execute at least three scripts to create an RDMS within a database server. |

|  |  |
| --- | --- |
| **Learning Outcomes** | **Assessment Criteria** |
| **The learner will:** | **The learner can:** |
| 1. Be able to develop SQL queries based on one table. | * 1. Use the SELECT statement to retrieve data from a table.   2. Demonstrate the removal of duplicate records from query results.   3. Demonstrate the filtering of data using at least three SQL operators including the WHERE clause.   4. Demonstrate the ordering of data using SQL ORDER BY clause. |
| 1. Be able to develop SQL queries based on more than one table. | * 1. Summarise using examples at least five types of joins.   2. Demonstrate the joining of tables using at least three methods including INNER JOIN.   3. Demonstrate the use of the following joins:   a) LEFT JOIN  b) RIGHT JOIN  c) OUTER JOIN |
| 1. Be able to use SQL aggregate functions. | * 1. Explain using examples key built-in aggregate functions including SUM, AVG and COUNT.   2. Use the GROUP BY clause to group data.   3. Split groups using HAVING clause. |
| 1. Be able to manipulate data within a database. | * 1. Explain the significance of referential integrity when carrying out Create, Read, Update and Delete (CRUD) operations.   2. Add new data to a table using SQL INSERT INTO.   3. Modify data within a table using SQL UPDATE.   4. Remove data from table using SQL DELETE.   5. Explain with examples stored procedures using Transact-SQL (T-SQL). |
| 1. Be able to export query results. | * 1. Research and explain different types of export techniques for data available within a database.   2. Export query results to a universal file format. |

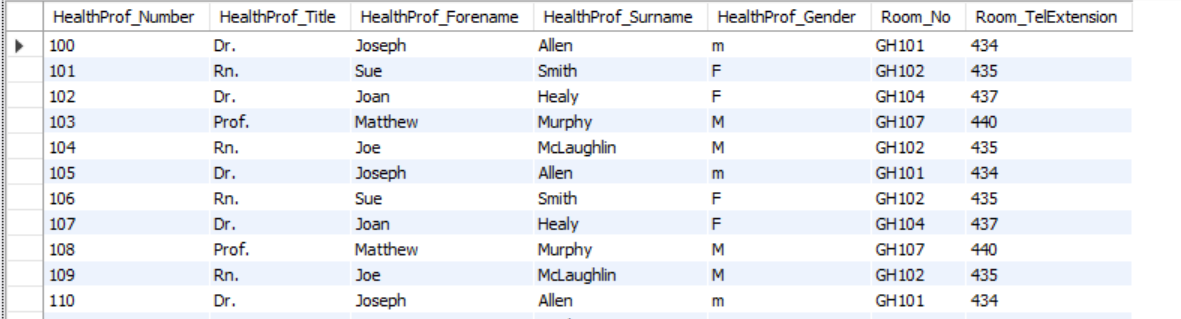
**ASSESSMENT CRITERIA (AC)**

**(AC 1.1)**

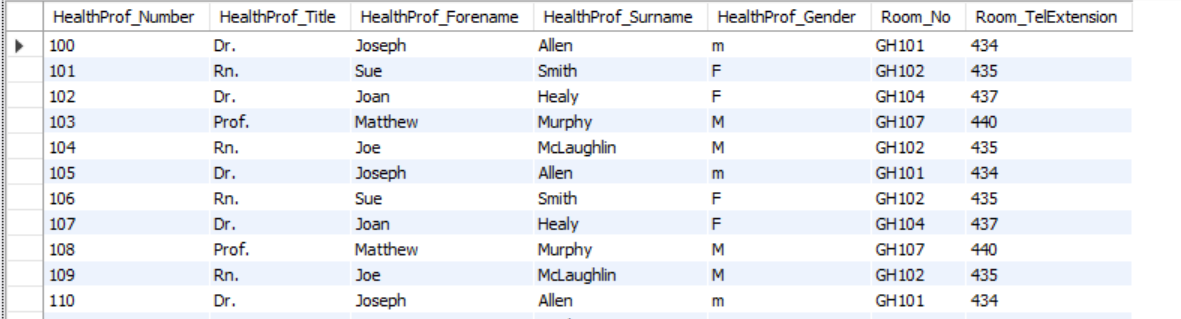
Explanation here on what a table is etc.

**(AC 1.1) Explain with examples what is meant by the concept of a table and how it is used to work with data.**

Tables are a database object that contain al the data in a database. In tables, data is logically ordered in a row-and-column format similar to in a spreadsheet, each row represents a unique record, and each column represents a afield in the record. For example, a table that contains employee data for a company might contain a row for each employee and columns representing employee information like employee number, name address job title and contact details like telephone and email.

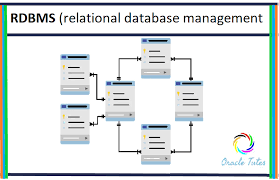


Another example of a table would be an appointment table. This table would store unique appointments in each of its records with each of columns representing information about the appointment like the health professional that will do the appointment which would be represented by a foreign key pointing to a record in the healthProf table shown in the above image the fields would also show additional information like appointment date, time room number and telephone number for the room.



**(AC 1.2) Explain what a relational database management system is and the main types of databases available**

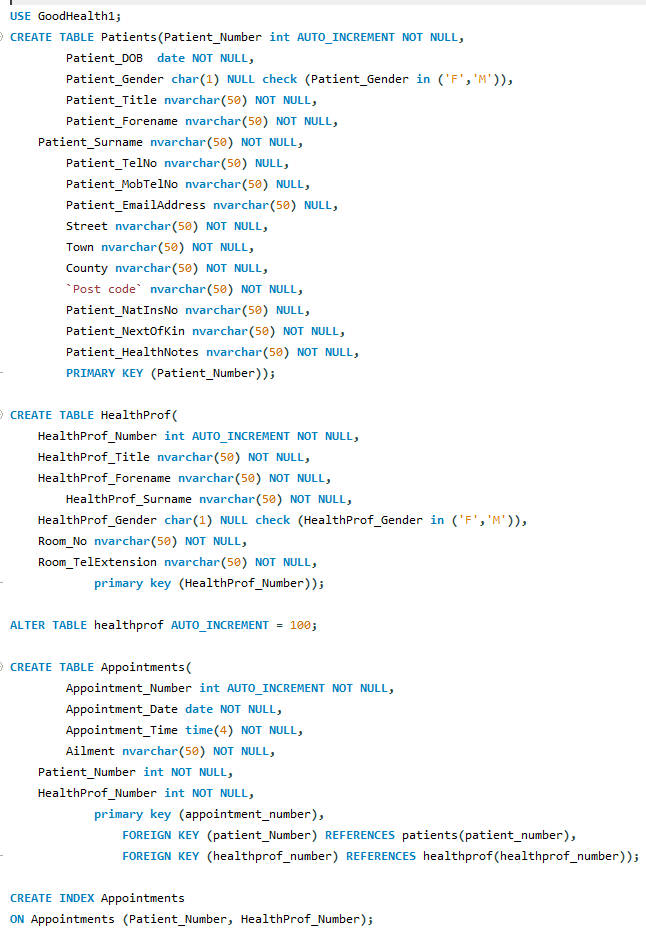
A relational database is a type of database that stores and provides access to datapoints that are related in a straightforward logical way using tables. This is done through the primary foreign key concept. This concept allows each table record to be uniquely identified using a unique key called a primary key which can be pointed to in other tables through a foreign key to establish a relationship between tables, The software used to store manage query and retrieve data stored in a relational database is called RDMS (relational database management system). This software covers three areas DDL (Data definition language), DML (Data manipulation language) and DCL (Data control language).

There are many other types of databases used other than relational database these include hierarchical database which resembles a tree structure where the relationships between records are pre-defined using one to one relationships between parent and child nodes and Network database which also have a hierarchical structure however uses a single-parent tree hierarchy that allows for many-to-many relationships allowing for one child to have more than one parent And finally the Object-Orientated database this database uses different types of relationships between two or more objects.

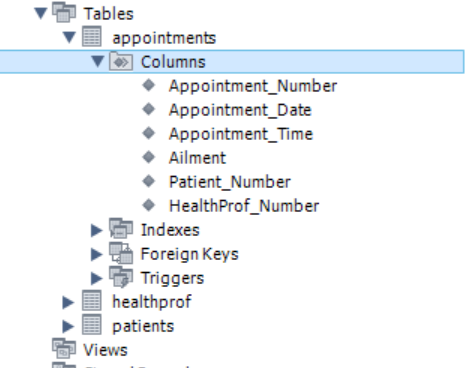
**(AC 1.3, 7.1, 7.2, 7.3, 7.4) Explain with examples the create retrieve update and delete operations available within a database.**

There are several operations in SQL that are used in order to manipulate data within a database. When doing these operations however a user needs to consider if he is keeping up the referential integrity of the database. Referential integrity is a database concept that is used to build and maintain logical relationships between tables to avoid logical corruption of data. It is very useful and important part in RDBMS the one rule to ensure a relational database is that a foreign key must have a matching primary key or must be null. This constraint is specified between two tables, and it maintains the correspondence between rows in these tables. It means the reference from a row in one table to another table must be valid.

**Create operation:**

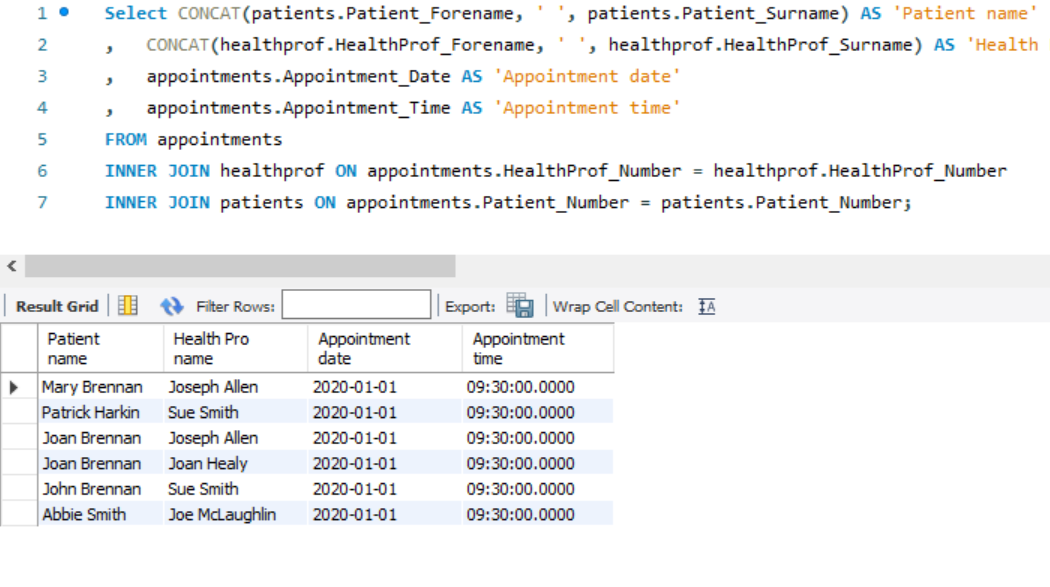
The create operation is used to define the tables within a database including the field names, field types, conditionals for those fields and whether primary, foreign, and composite keys exist, below is an example of a create statement script:

After this create statement is executed it outputs this to the schema view showing that the database has been created with the tables and columns that are defined in the statement above.



**Retrieve**

Select statements allow you to retrieve data from a database from tables you choose using constraints you select and then this data is presented in the form of a table view to make the information outputted easy to read.



**Insert Into**

The insert into operation allows you to add a new record into an existing database table, To ensure that referential integrity is upheld when doing an insert into statement it’s a good idea to check before you do an insert if the data you plan to insert already exists within the table this is to ensure that you do not add redundancy. After ensuring that the data does not exist you can do an insert like shown in the below example:

Before

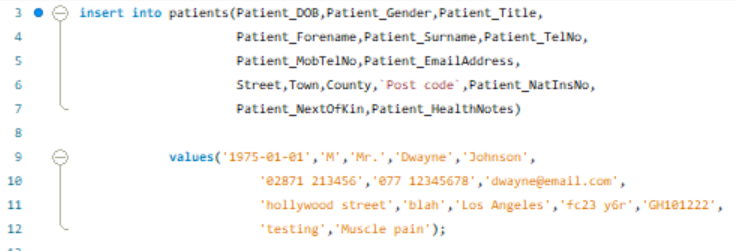
Before inserting a patient named Dwayne john I do a select to ensure that this customer does not already exist in the database.





During

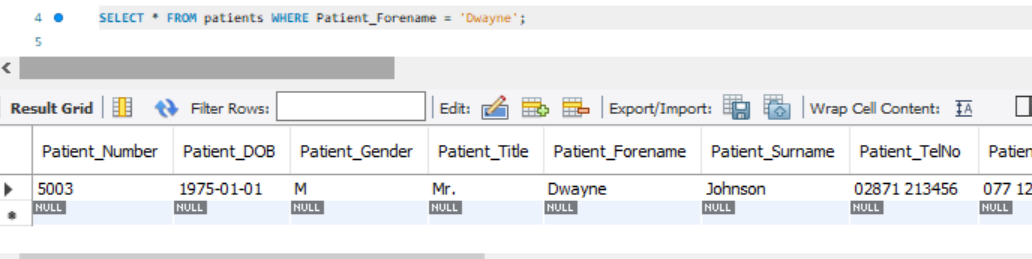
Since a patient called Dwayne does not exist we can add him.





After

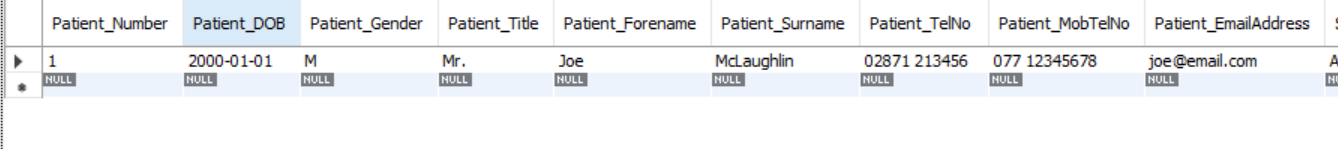
Now to confirm that Dwayne was added successfully I do a select where the forename is Dwayne

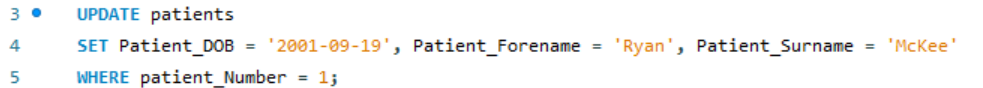


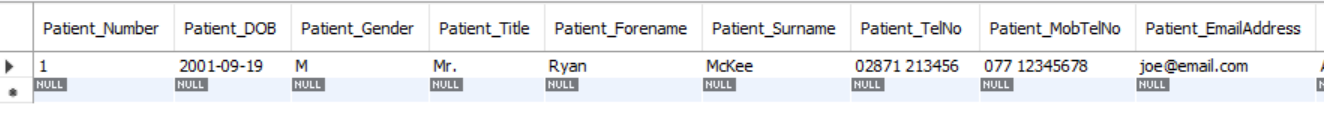
**Update**

The update operation allow you to change data already existing in record or multiple records, on records that you select, To ensure referential integrity is upheld you should check before the update the data of the record you are planning to update so you can decide if you need to update and what you should use in the where clause to ensure that you do not change a record that is different but has similar information.

Update example:

Before

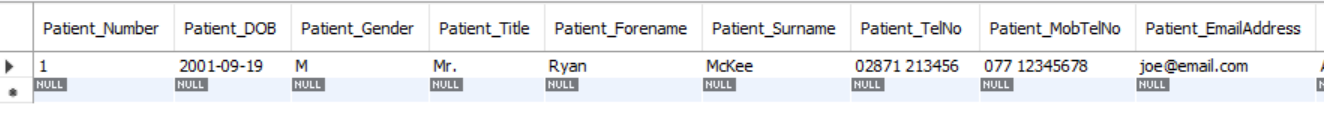
During

After

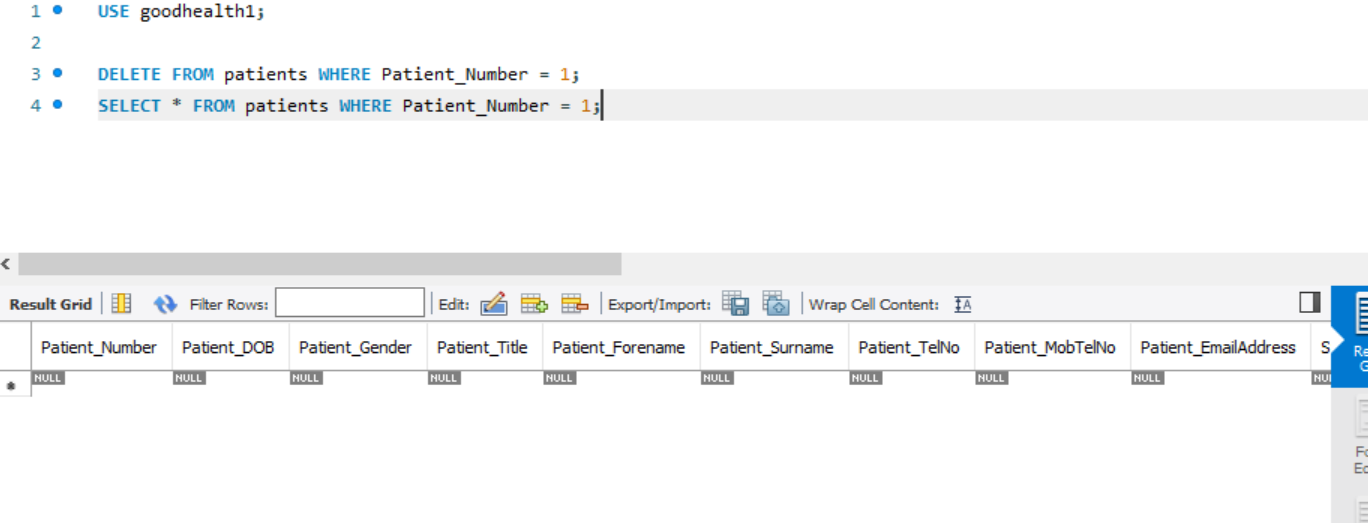
**Delete**

The delete operation is another operation this allows you to delete records from the database in a table based on the conditions you set. To ensure referential integrity it is a good idea to first view the record you are going to delete to if the record is the one you want to delete and decide what constraints you use to delete the record by then after you should also check if the item you wanted to delete was deleted.

Record with Patient number 1 before deletion:



Deletion query and Record with patient number 1 after deletion operation is performed:

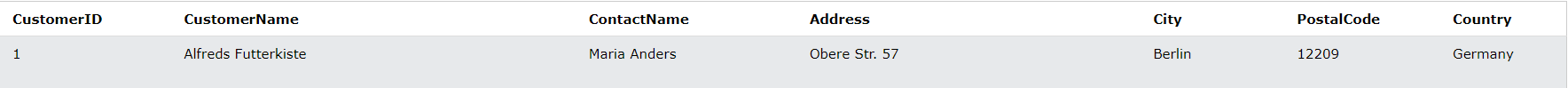


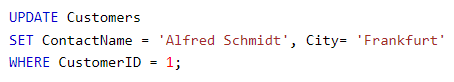
**(AC 1.4) Explain and use different types of SQL language including Data Manipulation Language (DML), Data Control Language (DCL) and Data Definition Language (DDL) to perform database operations.**

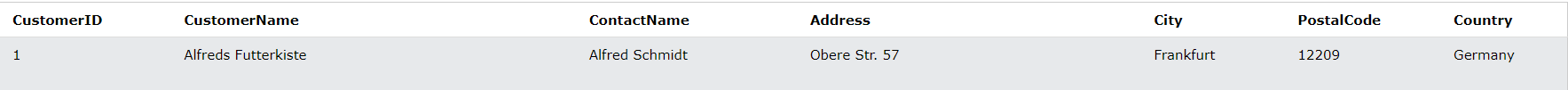
Using SQL queries you can query, filter, sort, join, group and modify the data in a database. Majority of these commands can be categorized into three categories these are data manipulation language (DML), data control language (DCL) and data definition language (DDL).

DML short for data manipulation language is typically the language used by data analysts this language allows them to execute queries in order to filter and retrieve data that they need in a view in a format that they require, this language type also allows them to insert new data, remove data. And update current data in the database as they please.

The following images contain an example of a DCL update statement on a customer table:

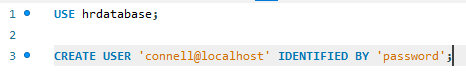
****

****

****

DCL is the next command type it stands for data control language and is typically used by admins of a database. DCL’s commands allow a server admin to decide who has access to information stored in a database and the language can be used to give or remove permissions from users to get or alter information in the database.

The following image shows an example of a grant command apart of DCL which is used to create a user within a database:





DDL short for data definition language is the portion of SQL that is used for the creation, alteration, updates and removal of database objects. There are many types of objects but these are the foundation of a database these include schemas, tables, views, sequences, catalogs, indexes, and aliases.

Below is an example of DDL language being used for the creation of a database itself. And then being removed.

**(AC 2)**

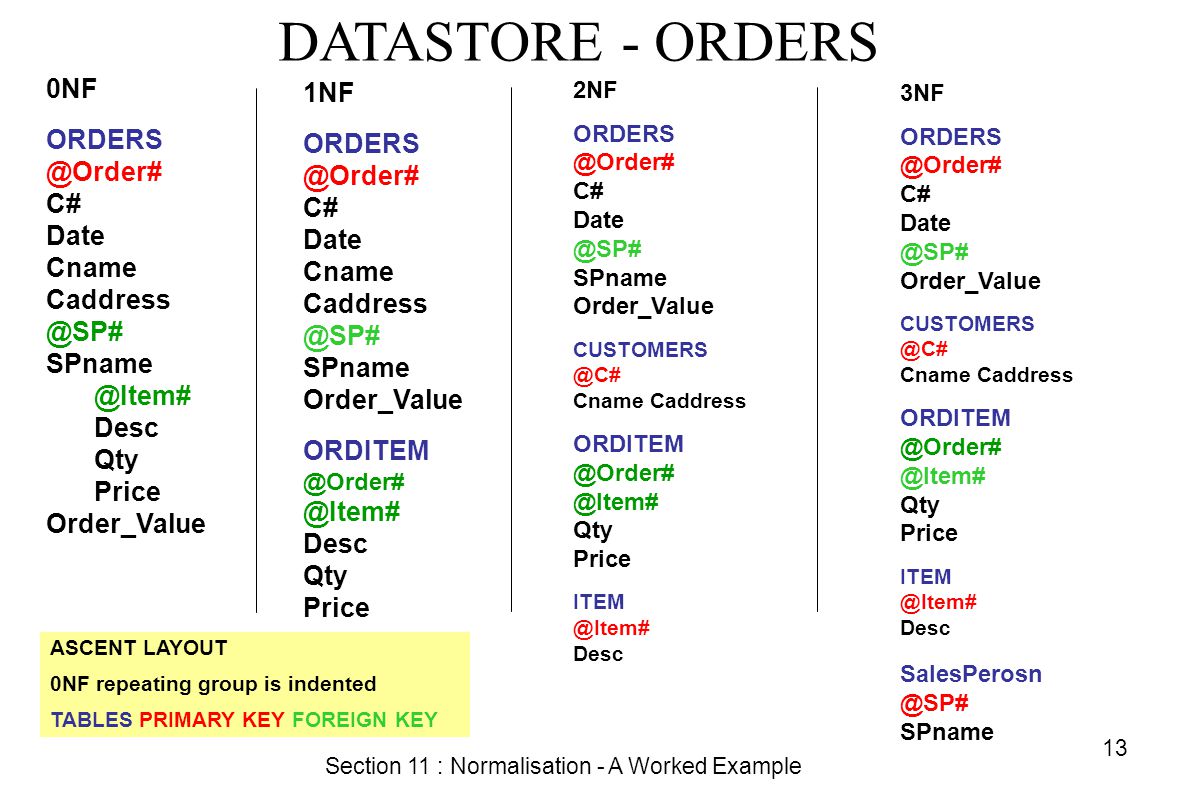
Be able to create an RDMS composed of tables.

**(AC 2.1) Explain the purpose of database normalisation and the forms available**

Database normalization is a technique of organizing data in a database. Normalisation is a systematic approach of decomposition tables to eliminate data redundancy and undesirable characteristics like insertion update and deletion anomalies. It is a multi-step process that puts data into a tabular form, removing duplicated data from the relation tables. Normalisation has many normal forms the main ones being the first 3 these are:

* 0NF: This form is the flat file here you just have the field names for the records within a flat file.
* 1NF: This form has 4 rules the first being that there should only be atomic fields, second is that values stored in a column should be of the same domain, third all columns must have unique names and finally the order of the data does not matter
* 2NF: In second normal form there are two rules that the database should already be in 1st normal form and that there should be no partial dependency’s
* 3NF: In third normal form each of the tables should be in 2nd normal form and should not have transitive dependencies.

Below is an example of a database being normalised from 0nf to 3nf:



**(AC 2.2) Explain the concept of primary, foreign and composite keys**

Primary, foreign and composite keys are the foundation of how relationships within a relational database work. Primary keys are the unique identifiers for each record within an entity, A primary key is typically assigned to each record in each table of a database. Foreign keys allow a record in a table to point towards a record in another table. A foreign key is the primary key in another table and using joins in SQL statements you can access the data within a table based on the foreign key in another. The final type of key is a composite key, this is a combination of two foreign keys to make a unique identifier for a record a composite key allows for the same functionality as a many to many relationships without breaking the relational database rule these tables are also known as junction tables.

**(AC 2.3) Create an RDMS composed of tables to include:**

a) database normalisation to the third normal form

b) creation of appropriate keys and indices within tables to form relationships

The image below shows An appointment booking system for a clinics normalisation process from 0NF to 3NF:

**(AC3)**

Be able to develop an RDMS within a database server.

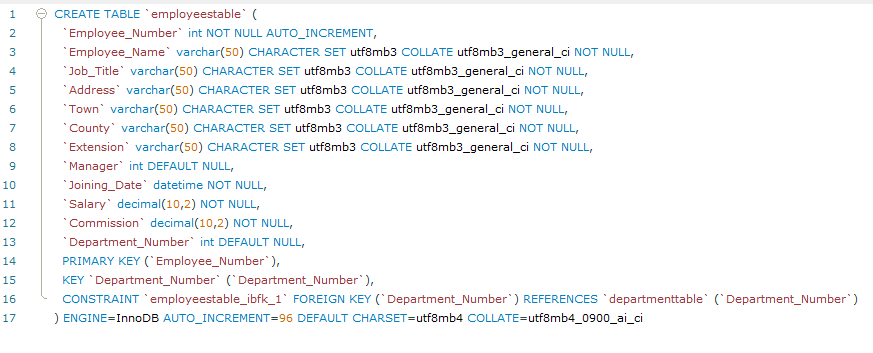
**(AC 3.1) Explain the main data types, why they are important and how they affect database storage.**

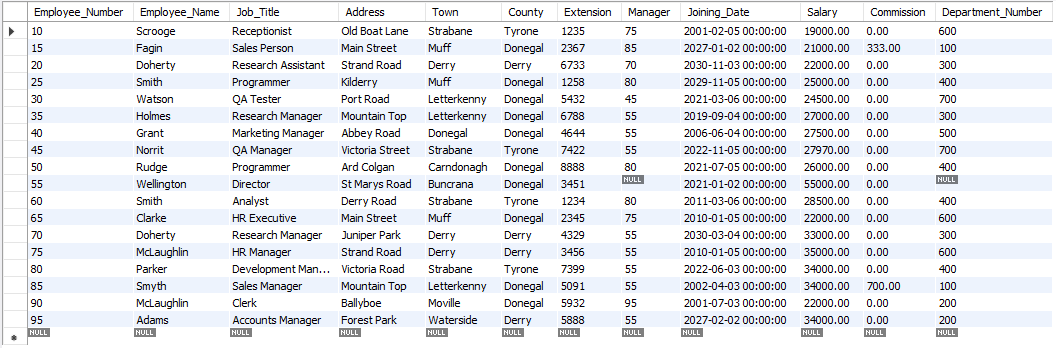
Within tables each field has a specified data types each of these types belong to one of the three main data type categories: string, numeric and date and time and then within these are sub types which are more specific to different requirements that may be needed for fields in a database.

The Strings data types has many sub types, The main ones being char which allows a fixed length string from anything from 0 to 255, varchar which allows a variable length from 0 to 65535 or from 0 to whatever the max length is that is entered into the parameters and nvarchar which is like varchar but has the length entered into the parameters is a fixed length not a variable length in the case of varchar.

Like with the strings type there are many numeric types like bit BOOLEAN which can store either 0 or 1. INT which stores a signed number from -2147483648 to 2147483647 and double which allows you to store a floating normal floating-point number. The parameters in for this take two items the size of the number that can be entered left of the decimal point and size right of the decimal point. The final Data type is date and time this has 5 types DATE, DATETIME, TIMESTAMP, TIME and YEAR, DATE takes just the data in format YYYY-MM-DD. DATETIME takes the year format from DATA but also takes time in format YYYY-MM-DD hh:mm:ss, TIMESTAMP takes a TIME value and autoincrements to show the time since it was last updated. TIME just takes time in format hh:mm:ss and YEAR takes a for digit year. Each of the formats talked about allow a person to create a database to fulfil specific requirements and add some validation to what can be entered into there database for example if a database designer wanted a field for a phone number they would probably have a datatype of char(11) which would make you enter a string of 11 characters long therefore making it slightly easier to validate what you are inputting is correct. As well as deciding datatypes based on requirements a database designer would also choose a datatype based on storage as different datatypes take up a different amount of storage within a database for example an INT data types field can take up as much as 4 bytes while a bigint takes up as much as 8 bytes so when deciding the data type that you use it is best that a person chooses the minimum size data type that fits the requirements.

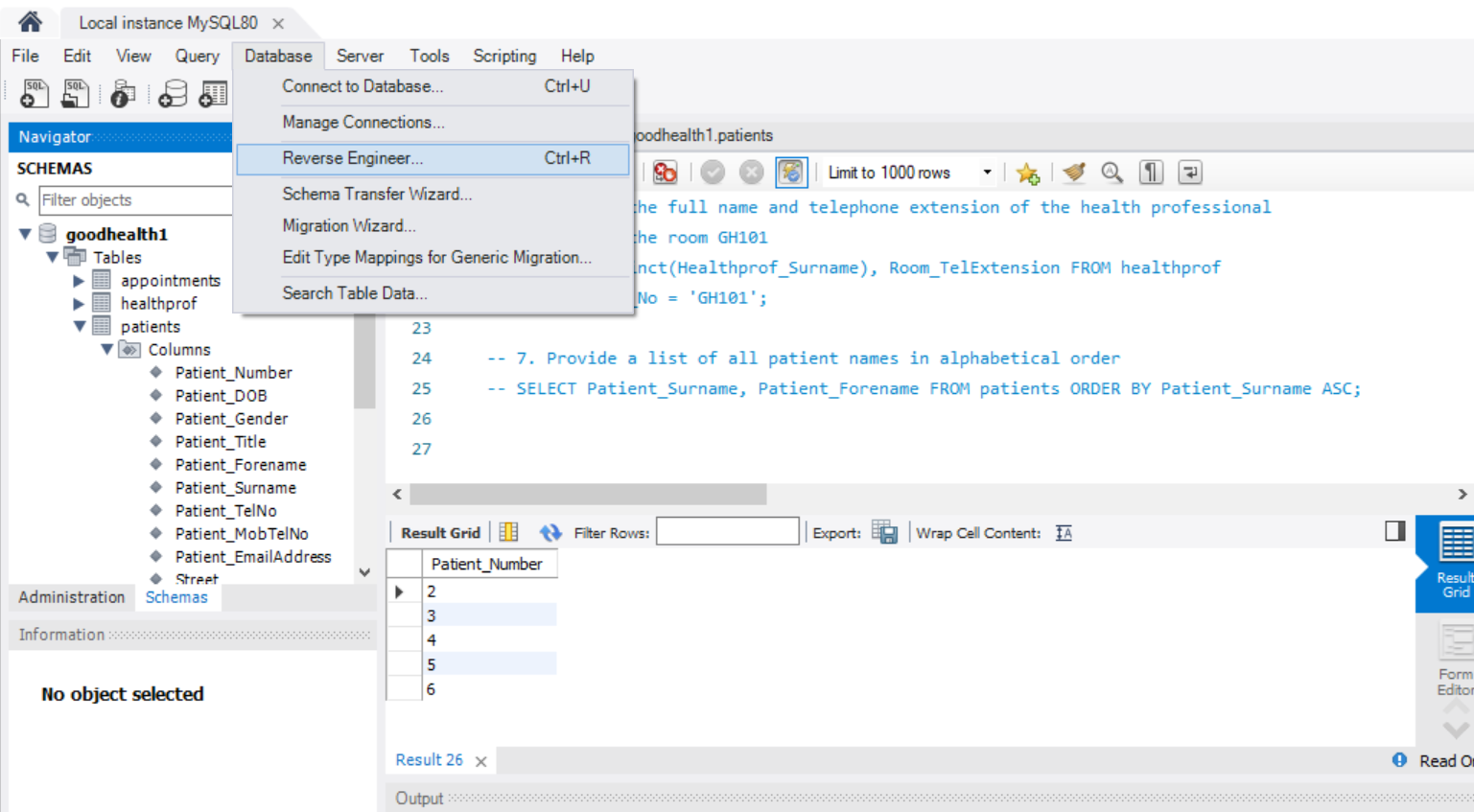
The following image contains examples of the different datatypes being used within a table in an employees database:





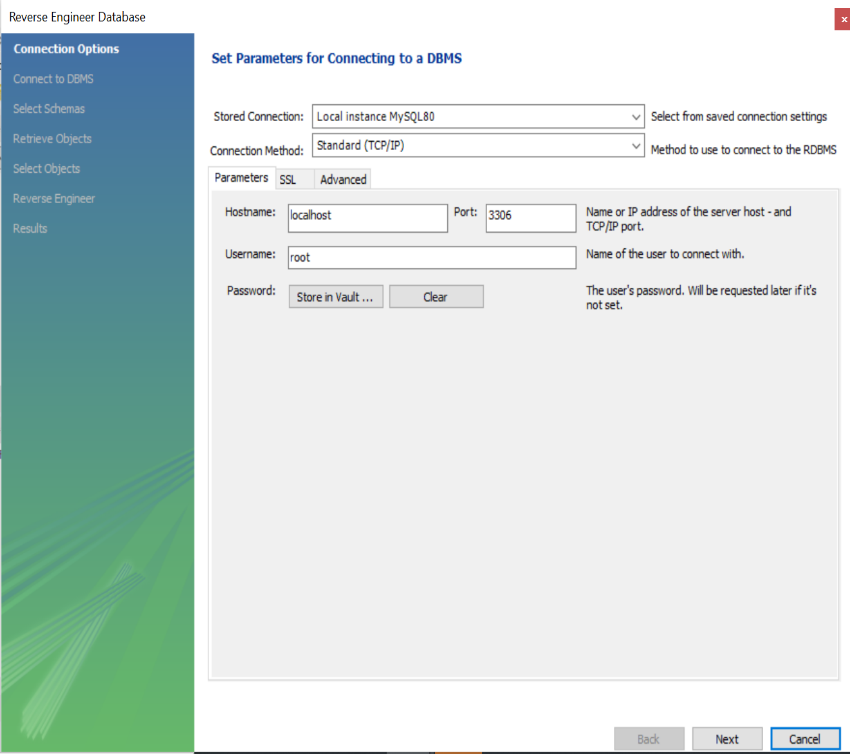
**(AC 3.2) Interpret and create an Entity Relationship Diagram (ERD) depicting tables and relationships**

To create an ER model using SQL workbench based on the database already created you just need to do the following steps.

****

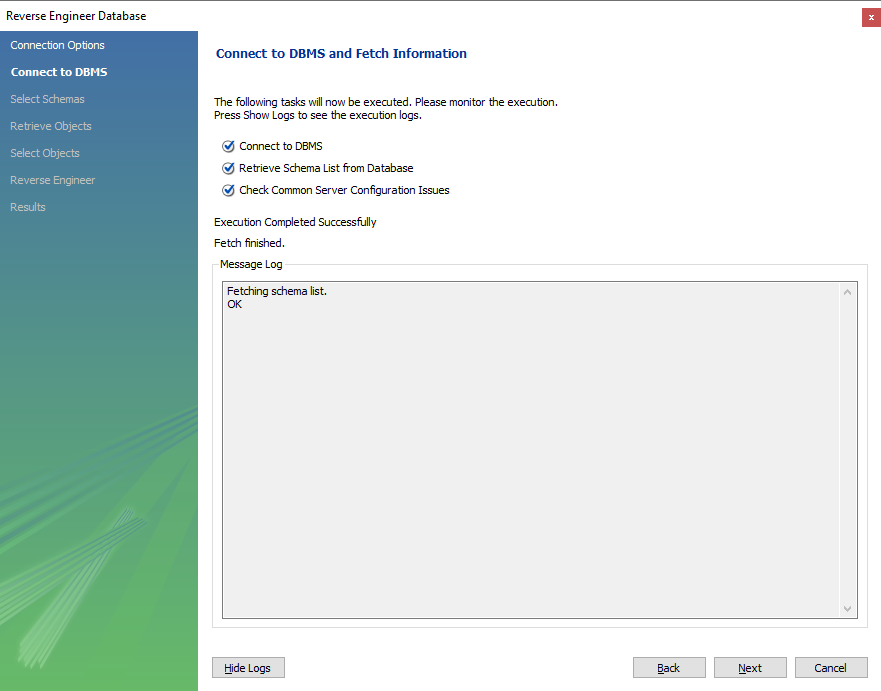
**Step 1:**

**To start you need to click on the database dropdown and select reverse engineer.**

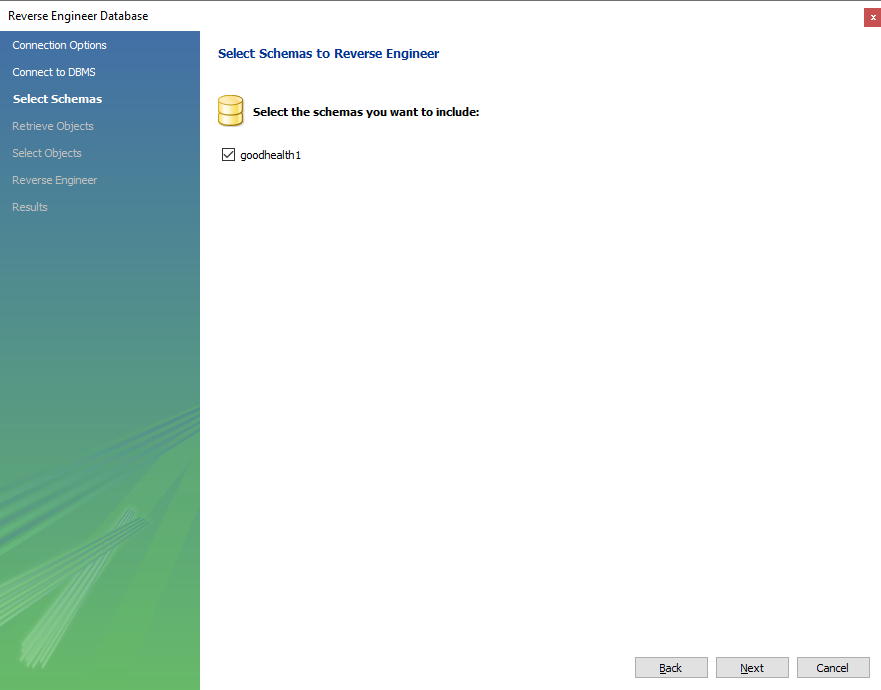
****

**Step 2:**

**After following the first step this panel should show up, You just leave the default settings already and click next.**

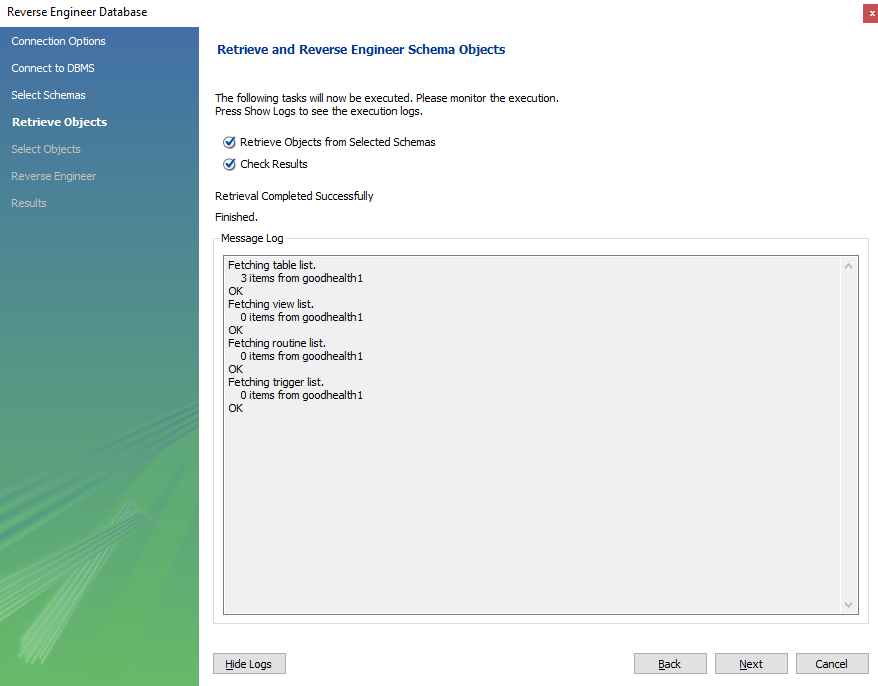
**Step 3:**

**The next form that shows up just finds all the databases available that you can reverse engineer you just need to select next.**

****

**Step 4:**

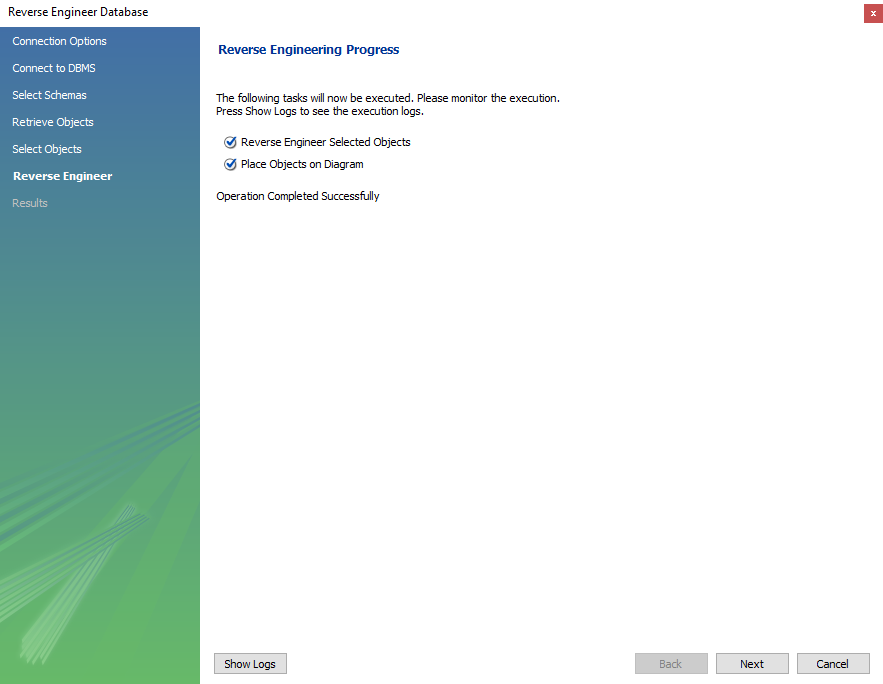
**This form you just select the database you want to create an ER model of in this case I choose the goodhealth database and finally clicked next.**

**Step 5:**

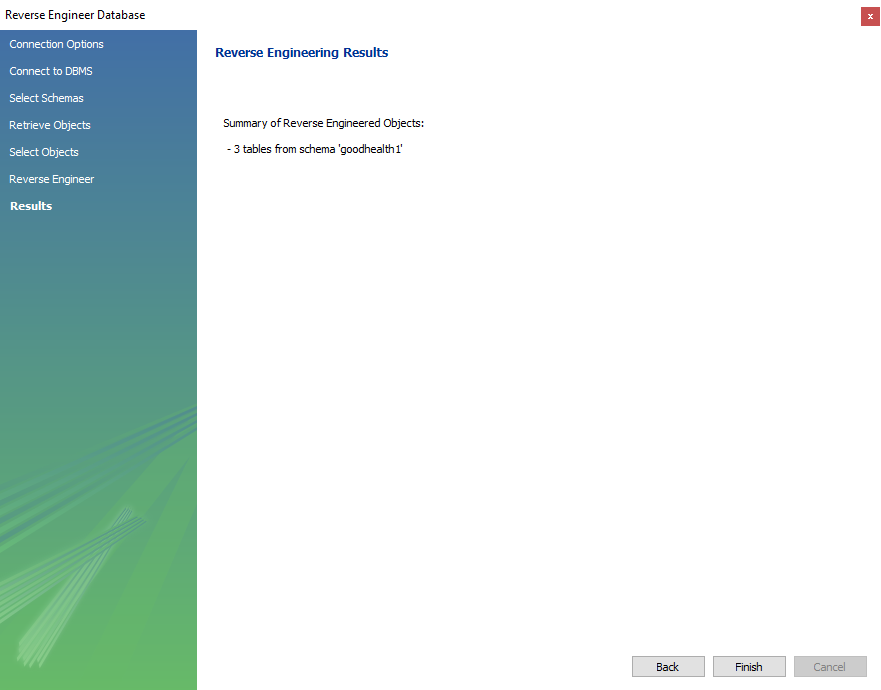
**This form just shows that the ER model is being mapped you just click next after they have been created.**

**Step 6:**

**Leave the defaults as is on this form and click next.**

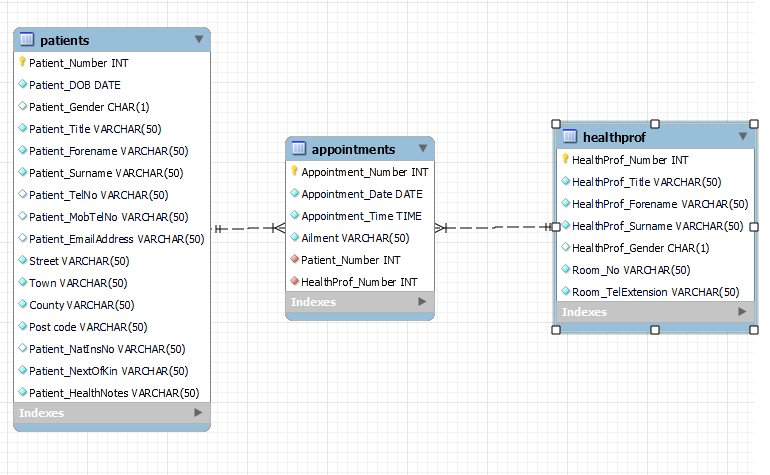
**Step 7:**

**Click next.**

**Step 8:**

**This form shows that the ER models have been created successfully.**

**Finally The ER models should show on workbench.**

****

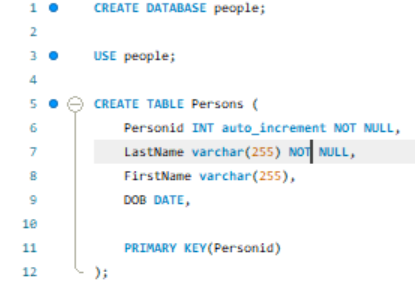
The above entity relationship diagram tells you that there are three tables in this database healthprof, appointments and patients. Both patients and healthprof have a 1 to many relationships to appointments making appointments a junction table. Appointments stores both the patients and healthprof ID’s as composite key.

**(AC 3.3) Develop and execute at least three scripts to create an RDMS within a database server.**

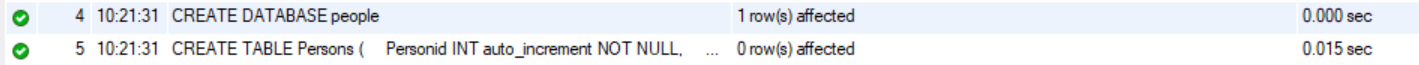
To develop and execute this database I used three scripts: A script for creating the actual database, one for inserting the data needed and finally one for retrieving all the data from the database to confirm the scripts worked as expected

Script to insert the tables

This script was to create the database this defined the tables, the columns within them and the column variable types.

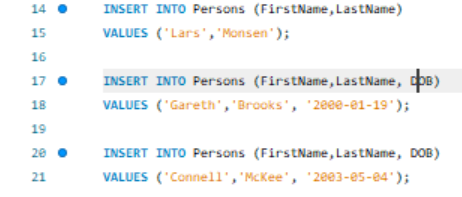


After the create script was executed a confirmation as shown below was outputted confirming that the database tables had been created successfully.

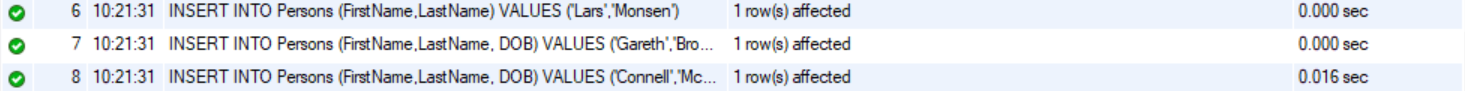


Script to insert data into the database

Script 2 was to insert all the data into the database. This script inserts records into the person’s table.



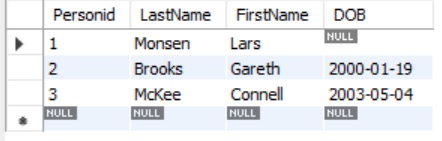
After the script was executed above multiple messages were outputted as shown below that showed that the inserts were successful.



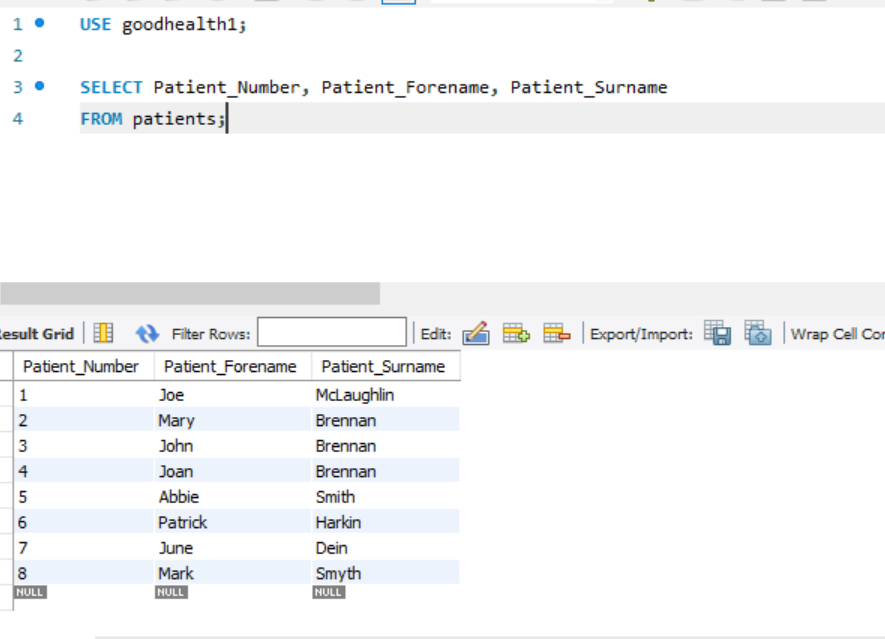
Validating that the inserts were successful

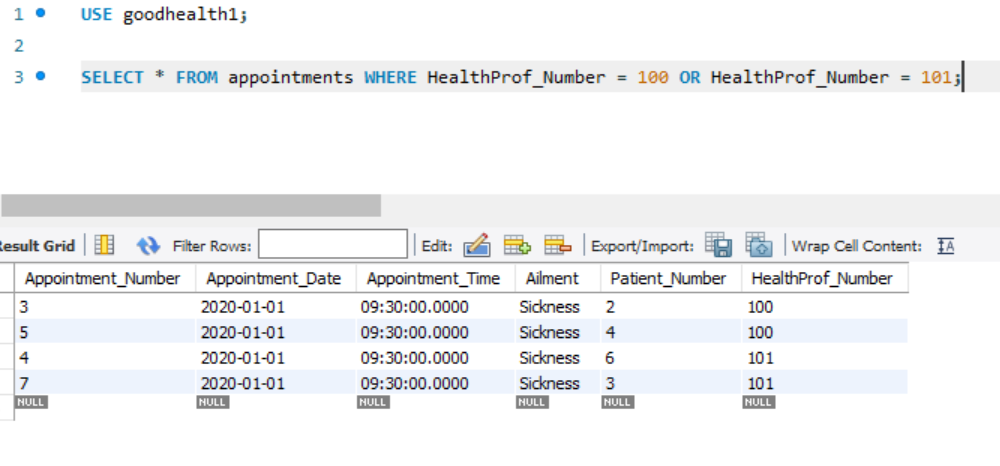
Script 3 was to select data from the persons table in to ensure that the data inserts in the above queries successfully executed.

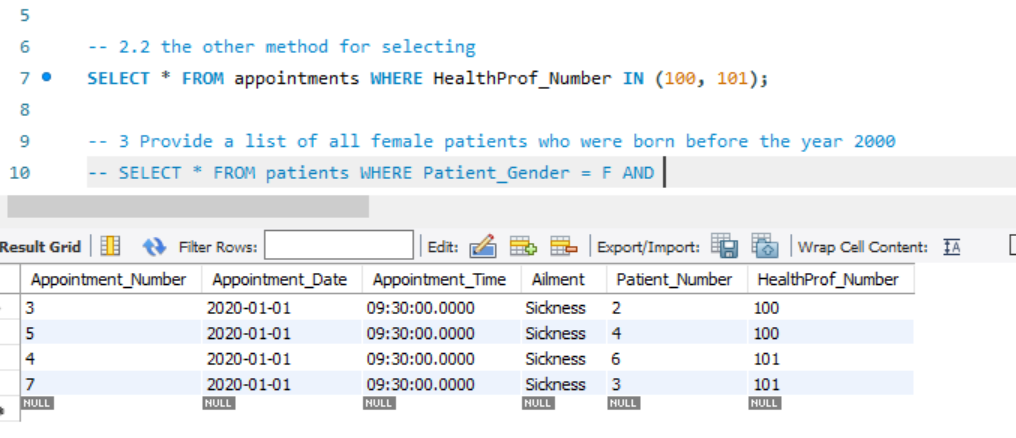


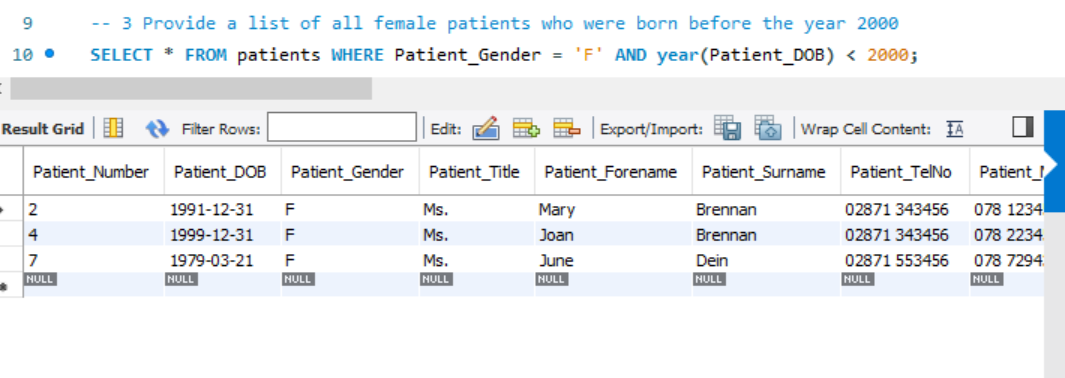
After the select is executed the table below is what shows showing all the three records that I tried to insert successfully inserted.

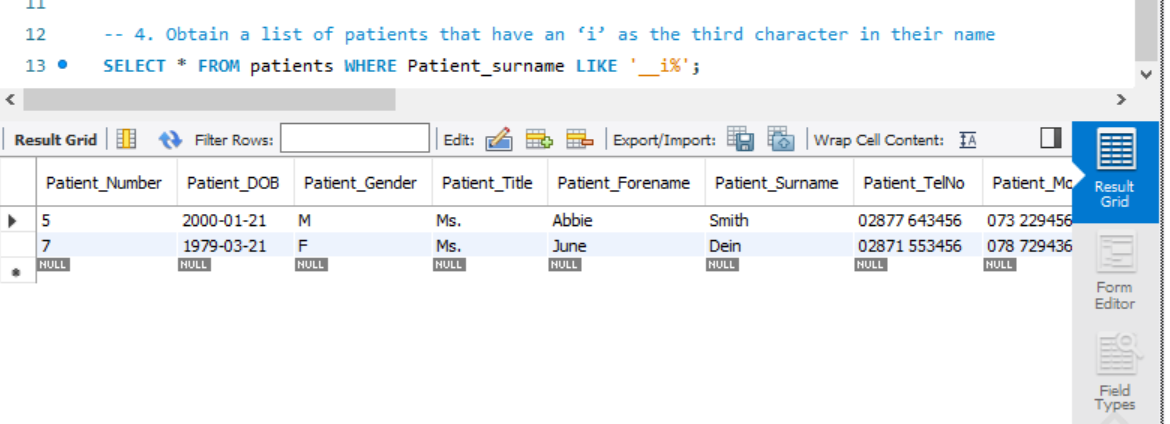
**(AC 4.1, 4.2, 4.3, 4.4) Be able to develop SQL queries based on one table**

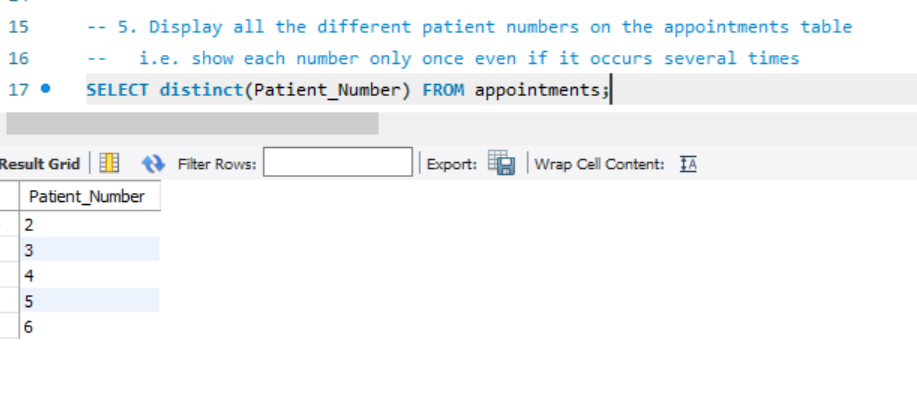
**Q1. Obtain the patient number and full name for all patients registered at the practice**

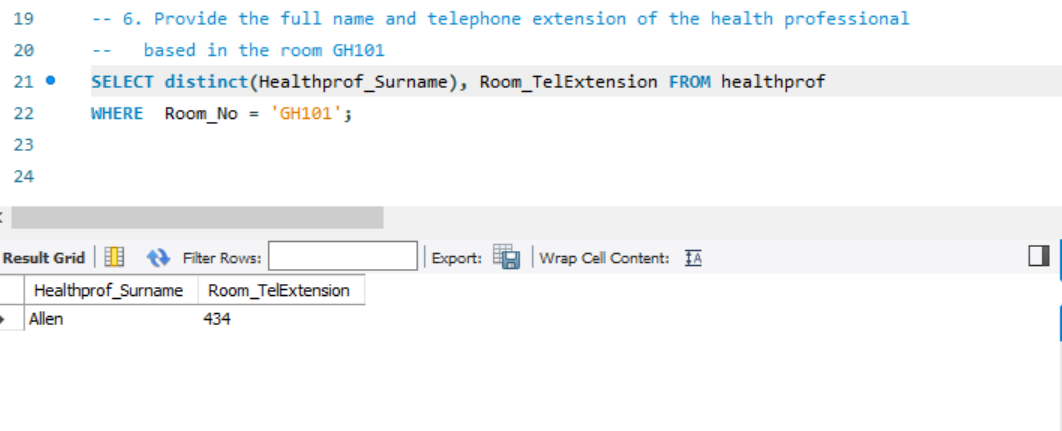
**Q2. Obtain all the details from the appointments table for health professional numbers 100 and 101. Do this in two different ways.**

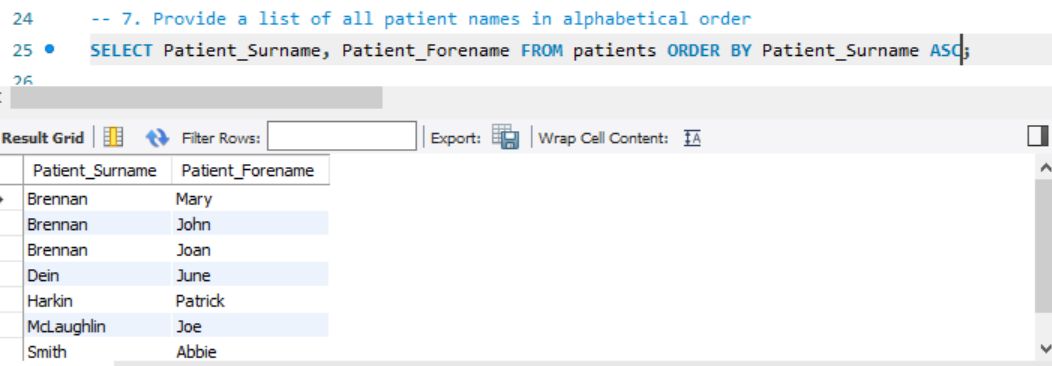
**Q. 2.2. Alternative method:**

**Q3. Provide a list of all female patients who were born before the year 2000**

**Q4. Obtain a list of patients that have an ‘i’ as the third character in their name**

**Q5. Display all the different patient numbers on the appointments table**

**Q6. Provide the full name and telephone extension of the health professional based in the room GH101**

**Q7. Provide a list of all patient names in alphabetical order**

**(AC 5)**

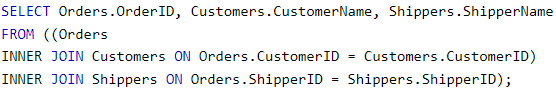
Be able to develop SQL queries based on more than one table.

**(AC 5.1) Summarise using examples at least five types of joins.**

There are 5 main types of joins in SQL, Inner joins, Left outer join, right outer join, self join and cross join each of these give the ability to access fields in different tables using primary foreign key connections although each of these implement joins in a different way and allow you to access different kinds of data.

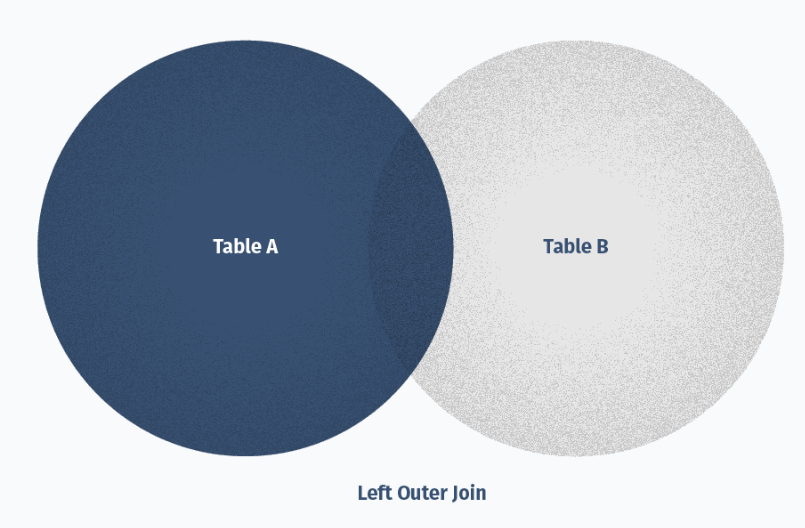
**Inner Join**

An inner join creates a result table by combining rows that have matching values in two or more tables that are being connected.

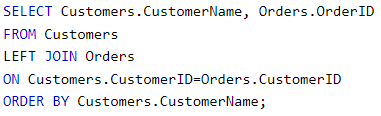
The below images are an example of a select statement utilizing an inner join so that fields from three tables customers, shippers and orders can be presented in one table output when the select is executed:

Since we used an inner join on this select there is no null values exist as a record can only be shown where an order has both a customer and a shipper ID.

**Left outer join**

****A left outer join creates a result table and includes into it all the records from the left table and only matching rows from the right table.

The following two images show an example of a select statement use a left join between a customer and order table.

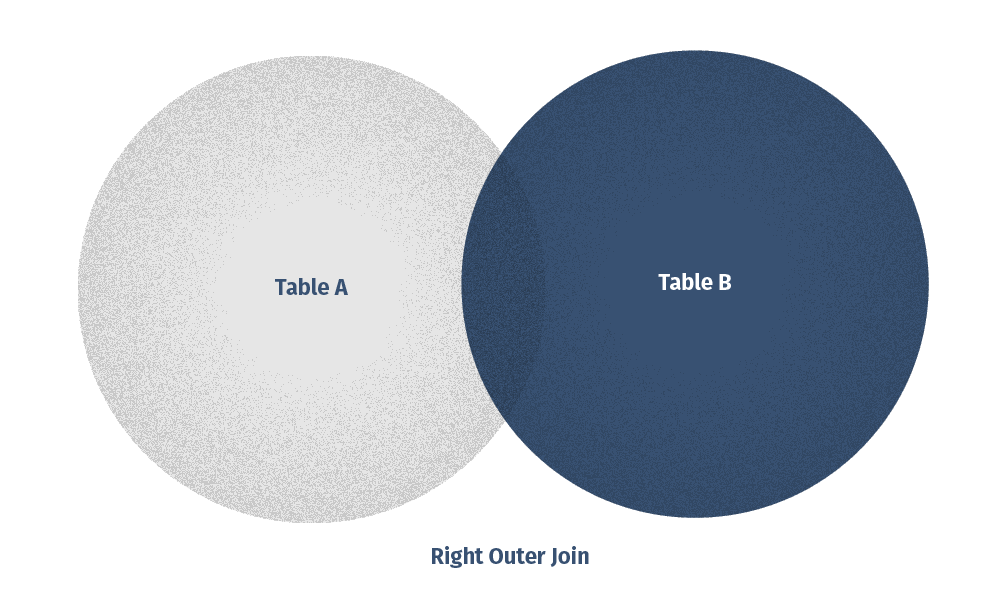




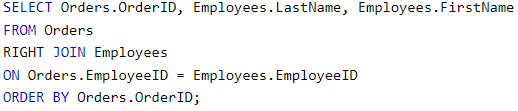
Since this is a left join and table a is customers with table b is orders every record from table a is shown in the output show above even if there is no order with that records foreign key that’s the reason the first record shown in the above table is null.

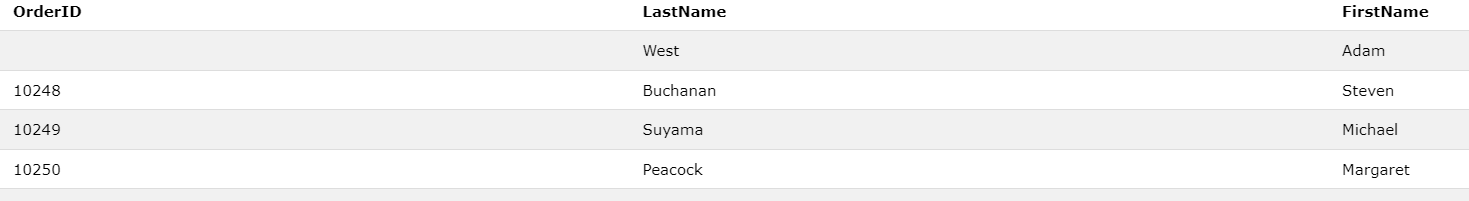
**Right outer join**

A right outer join is the opposite of a left outer join, it creates a result table and includes into it all the records from the right tables and only matching rows from the left table.



The following two images show an example of a select statement that uses a right join to information about orders and employees.

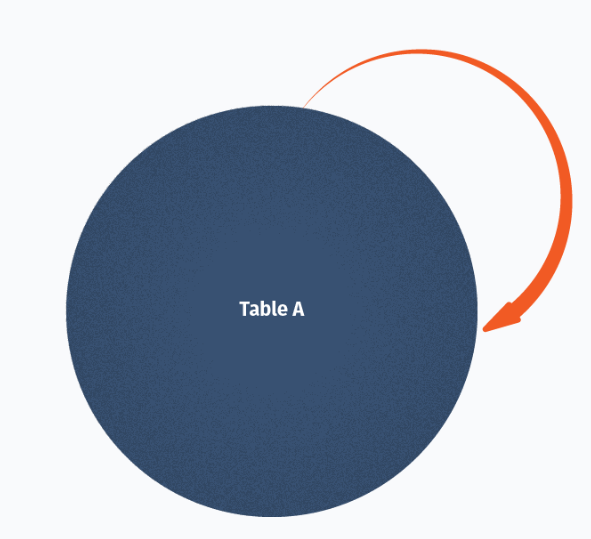
****

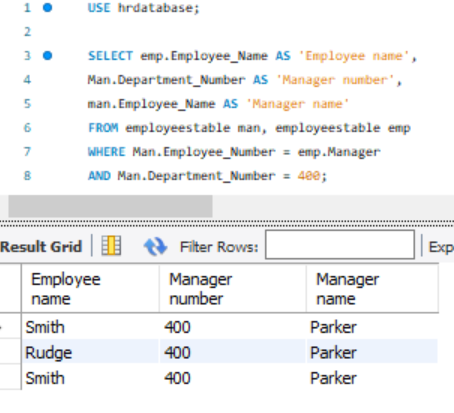


Since this is a right join the records above show names of all employees and Order id’s even if an employee does not have an order id.

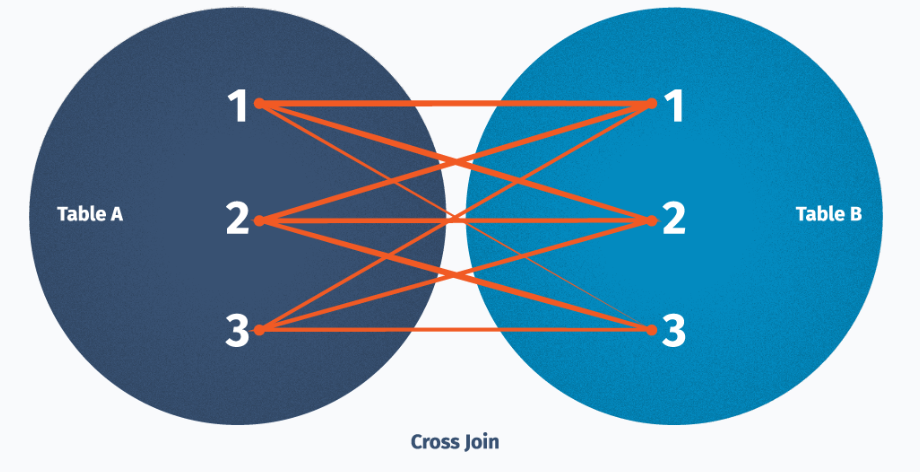
**Self-join**

SQL self-join joins the table to itself and allows comparing rows within the same tables

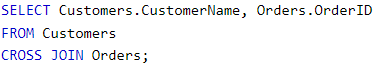
****

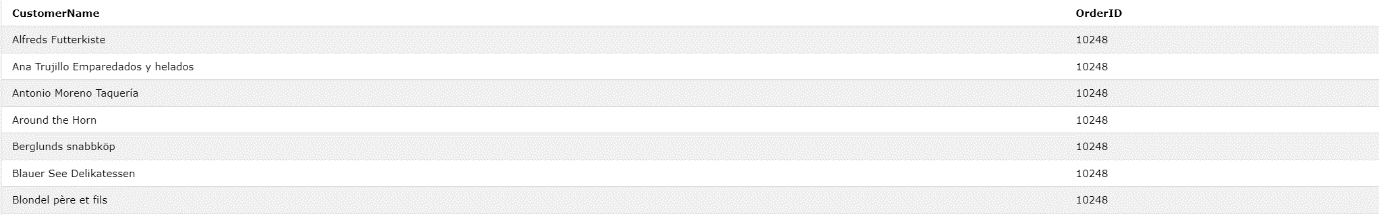
The below image shows an example of a self-join query that gets all records from patients and appointments for patients who have appointments.

Cross join

****SQL cross join creates a result table containing paired combination of each row of the first table with each row of the second table.

The following images contain an example of a cross join query and its output, This cross join gets customer name and order numbers.

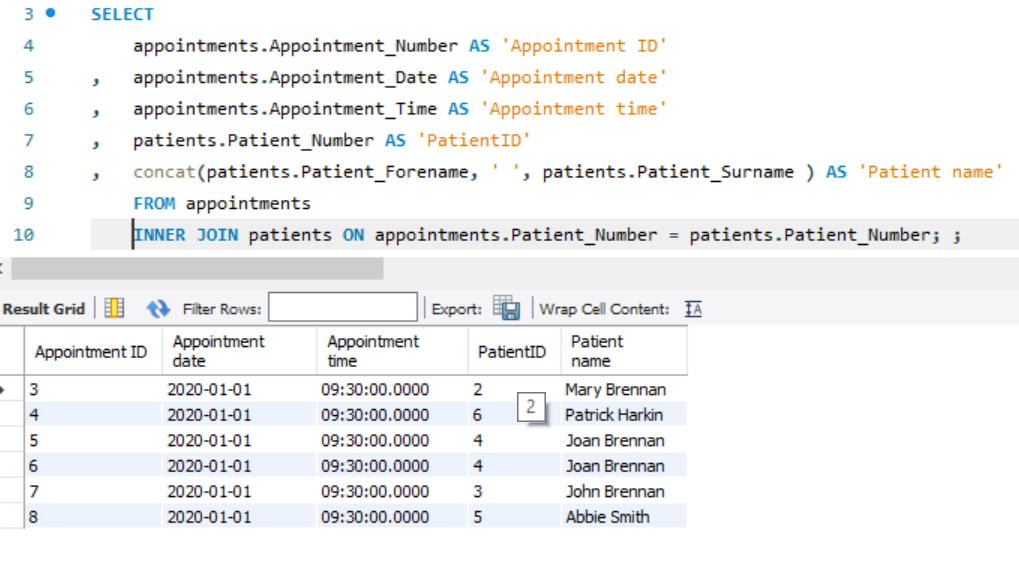
****

****

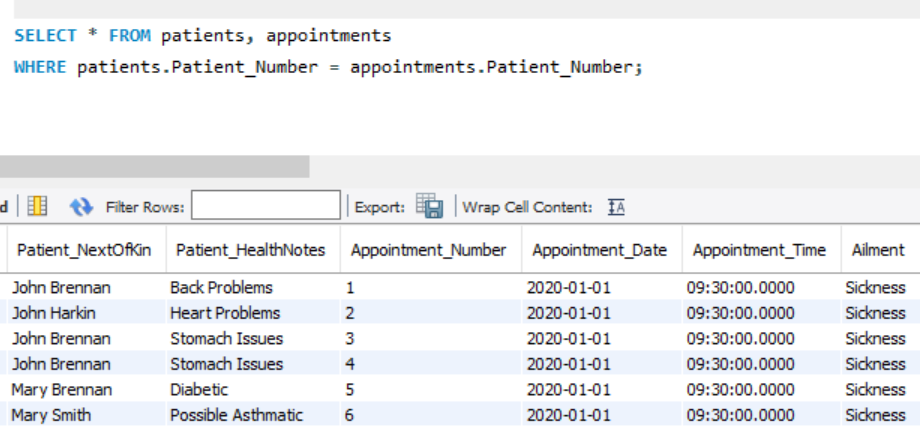
**(AC 5.2) Demonstrate the joining of tables using at least three methods including INNER JOIN. – Should do normal inner join, do an inner join with aliases**

The following images show some examples of different kinds of joins that can be used to gather information from multiple tables including an inner join.

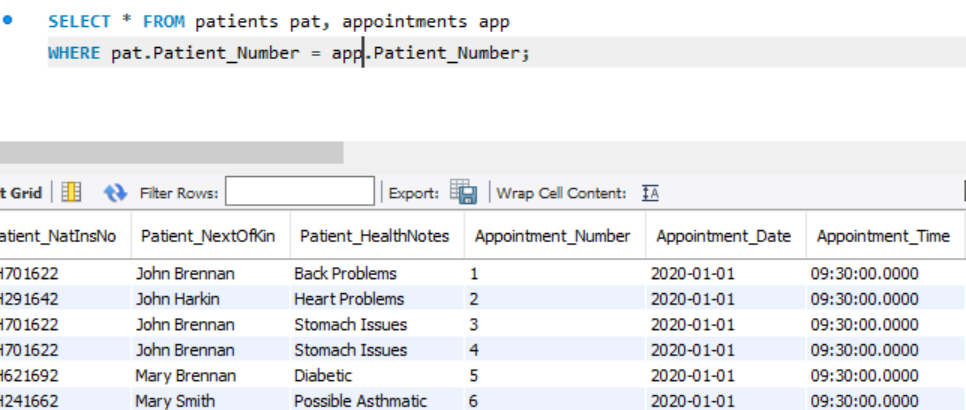
The first example of a join is an inner join:



The second example is a join using correlated names:

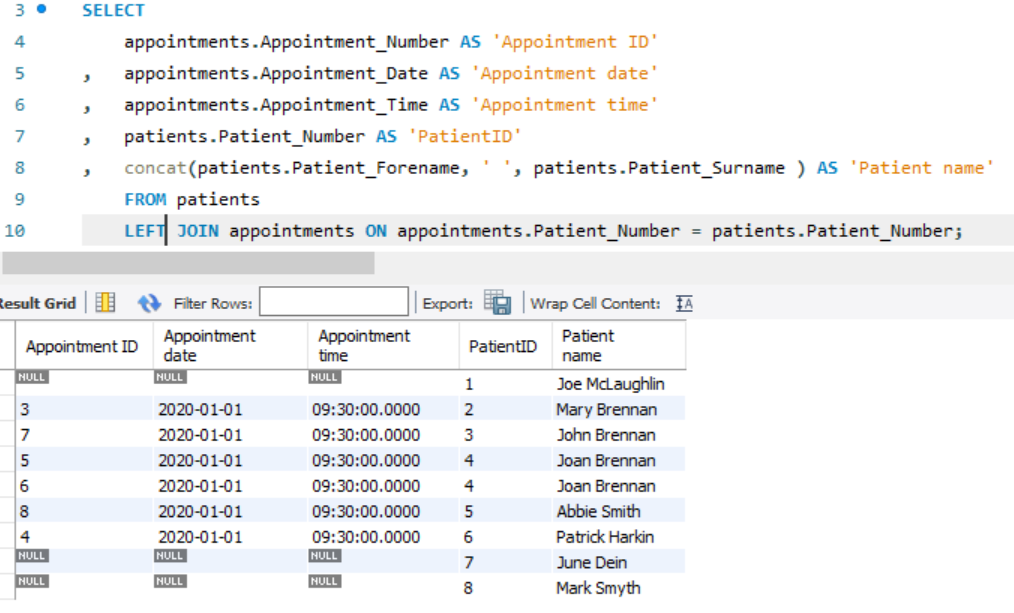
****

The final example is a join using aliases.

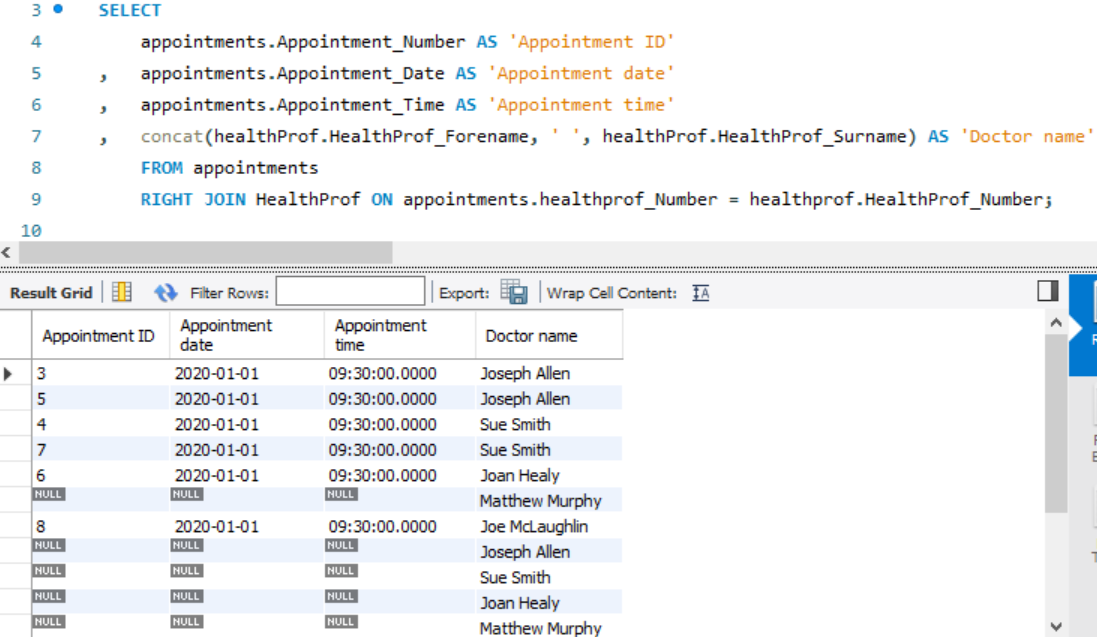
****

**(AC 5.3) Demonstrate the use of LEFT JOIN, Right Joins and outer joins.**

LEFT JOIN

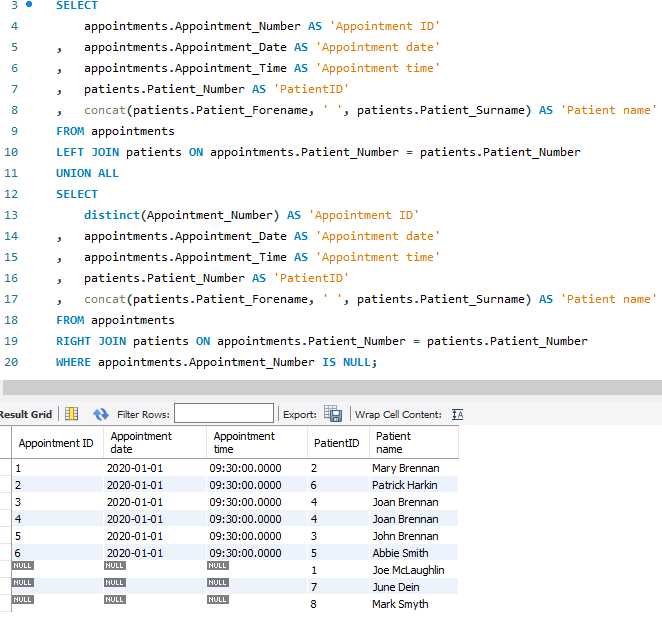


RIGHT JOIN



OUTER JOIN

As I am using mySQL there is no FULL OUTER JOIN keyword so to counter this I used to SELECT statements unionised one selecting the records using a left join and the other using a right join.

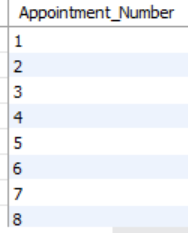
****

**(AC 6)**

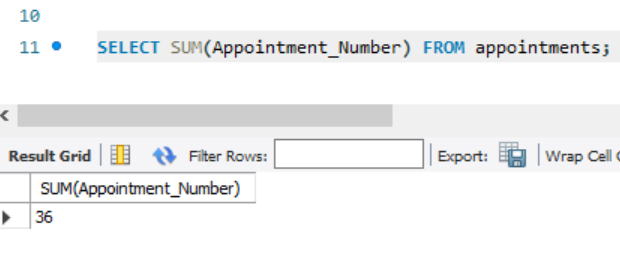
Be able to use SQL aggregate functions.

**(AC 6.1)** **Explain using examples key built-in aggregate functions including SUM, AVG and COUNT.**

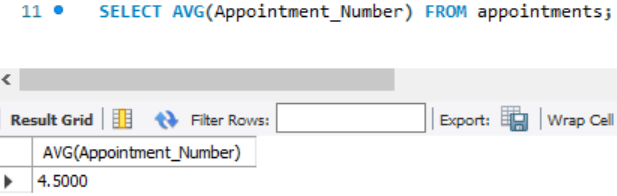
Built in aggregate functions perform calculations on multiple values and returns a single value. There are many built in aggregate functions in SQL that allow for the extraction of information that you are looking for easier. Sum is one of these functions in the parameters of sum it takes a field and all the values of these fields are added up together and outputted. For example, if I was to get the sum of appointments numbers from the table below:



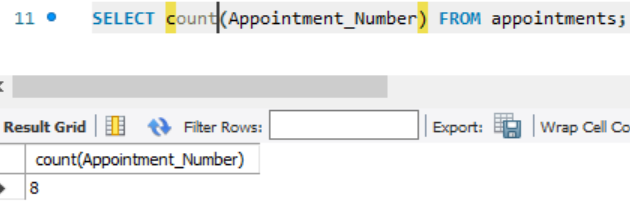
If I was to manually add these fields records together in the set 1,2,…,8 you would get 36 but instead if you were to use the SUM function in a select if will do the calculation for you.



The next function commonly used is AVG this will take all the record field values for the field named in the parameters and output the average value so if I was to do apply this function on the appointment number field as well you get 4.5.



The final of the main aggregate functions is the count function this function just counts the amount of records that are within a field that you put in the parameters.



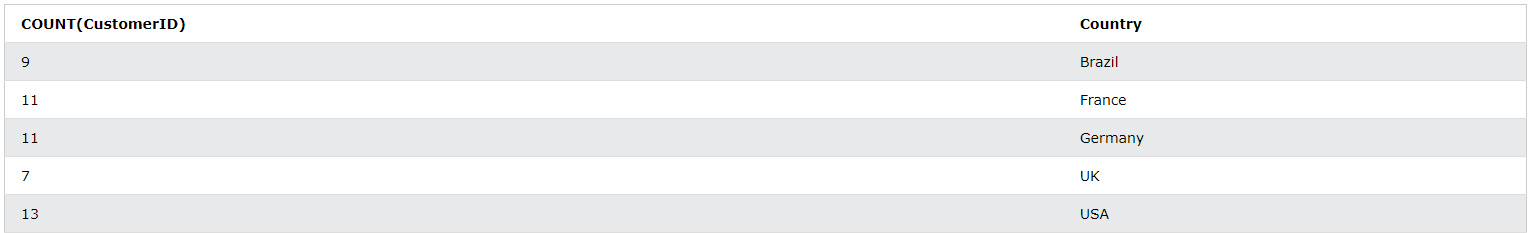
**(AC 6.2 & 6.3) Use the GROUP BY and HAVING clause to group data and split groups.**

The group by clause is a powerful function in SQL it allows you to group records together by the value in a field you give in the group by for example in the query below I group the records in the select by the patient ID this would be very useful in tables that are very large as it would make it easier to find data that you need.

The having clause is a powerful SQL feature that allows you to specify conditions which groups results appear in results, this clause is used in coalition with the where clause and was added to give SQL queries more versatility when filtering as where clause cannot be used with aggregate functions so having given a query an ability to use aggregate functions.

The following query shows a select which utilizes a having and group by clause. This clause selects the number of customers in a country where the customer is greater than 5.

****

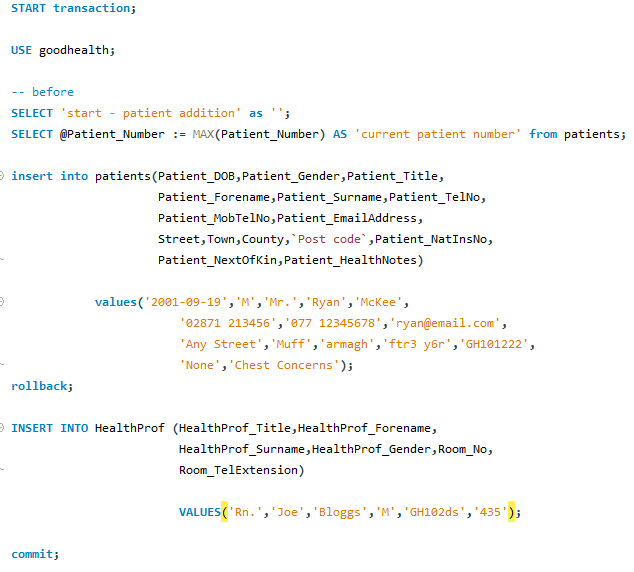
****After the above query is executed, this is the output:

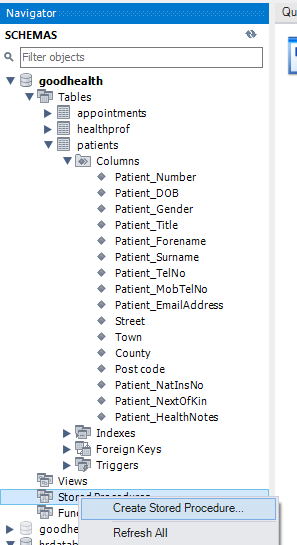
**(AC 7)**

Be able to manipulate data within a database.

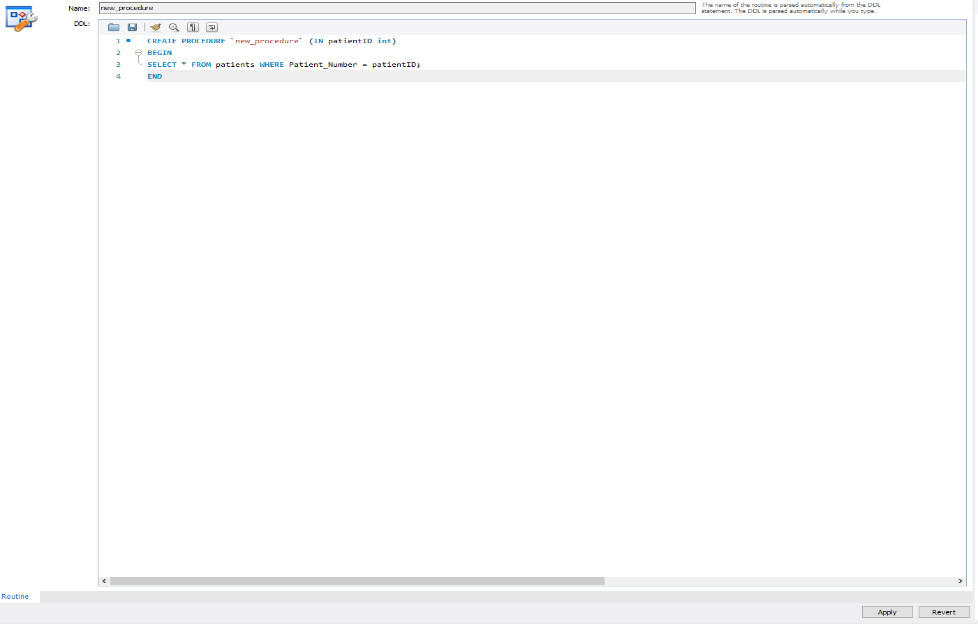
**(AC 7.5) Explain with examples stored procedures using Transact-SQL (T-SQL). – A stored procedure**

Stored procedures are a collection of transact SQL statements that are saved within the database for the purpose of making it easy to reduce redundancy when writing queries as you can easily call a stored procedure instead of constantly re-writing a commonly used query, Using stored procedures have other benefits also one of these is being a prevention measure to SQL injection which is the most common hacking technique used against SQL database in order to steal data. SQL injection arises as fields are available for user input allow SQL statements to pass through and query the database directly to steal information, they should not be able to access, The access that a hacker has to a database through the use of SQL injection can be taken away through the use of a stored query and a stored query can ensure that only a specific parameter can be inputted into a field of a query.

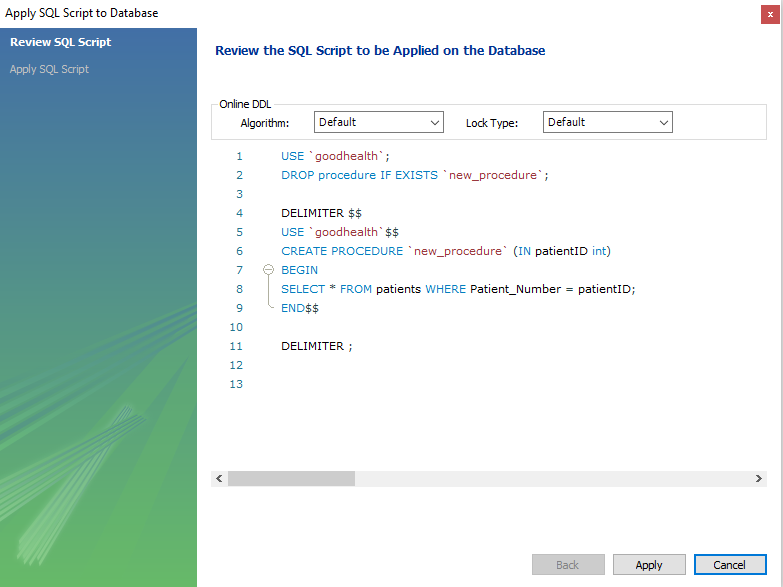
Below is an transaction SQL a transaction starts with start transition after this there are two main commands rollback and commit, if a commit command is used execute the commands before it however if a rollback command is used all the commands before it won’t be executed.

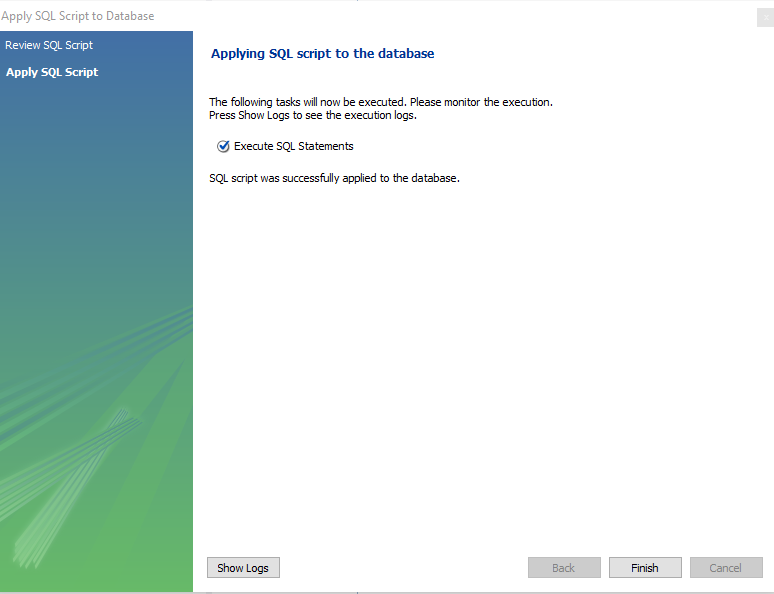
Now if you wanted to create a stored procedure in my SQL workbench it is simple to do so you have to first navigate to the schema pane on the left-hand side of the page and under the database stored procedure folder you should right click then click the create new stored procedure button

After selecting the above option, you should have a window open in workbench with a text editor where you can enter a procedure as shown below.



When you are done creating your query you need to save it you can do this by clicking the apply button in the bottom right of the pane.

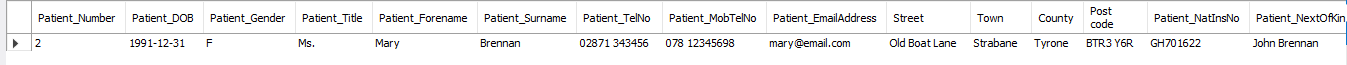
****

****After clicking next this form above will show up you just click apply

If you want to execute that stored procedure all you have to do is right click on the run button that shows up when you hover over, it.

****

**Since this procedure has a parameter in it a form opened when I tried run it in this case I just entered the patientID 2 and clicked the execute button and it created the query shown below the results view below that showing the record details for the patient with patient number of 2.**

****

**(AC 8)**

Be able to export query results.

**(AC 8.1 & 8.2) Research and explain different types of export techniques for data available within a database and Export query results to a universal file format**

A data analyst may want to export data from a database in order to perform data analysis and inference. There are many different ways in which someone can export data and many different file formats in which it can be exported.

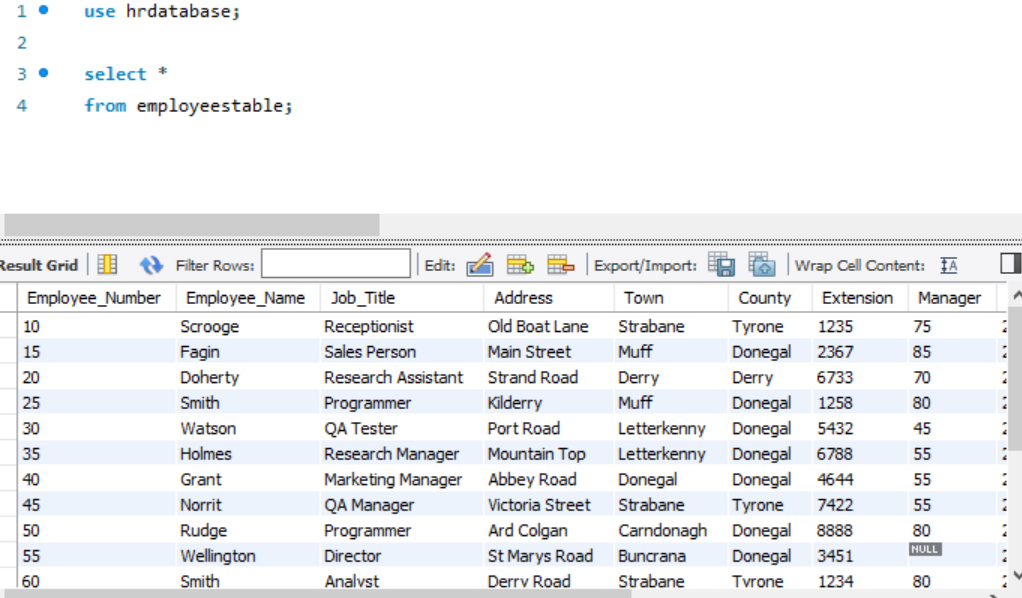
The most common file type is CSV (comma separated values) this file when exported puts columns in a database separated by ‘,’ otherwise known as the field separator and records are on new rows this is called a line break. export for this file is done by database tour engine with direct file access therefore making it very fast. CSV is so widely used because it has multiple advantages other file types don’t have like being easy to create, human readable because text is outputted in ascii format as its not encoded, they can be opened and edited by almost any text editor, and they are compact as they have a small file size. They do have their disadvantages like not being able to handle complex data, having no separation of numeric data and not being 100% standardised.

The next type of file type are JSON files like CSV files they have the same field separator ‘,’ and line separator of new lines. They are also fast to create because they also use the database tour engine. These files are saved with extension. json they have a large file size because of all the metadata saved unlike a csv and are less secure than a csv. The benefits of using json is that it is easy to integrate with PIs and that they use JavaScript data types. They also support hierarchical can relational data which csv does not without errors.

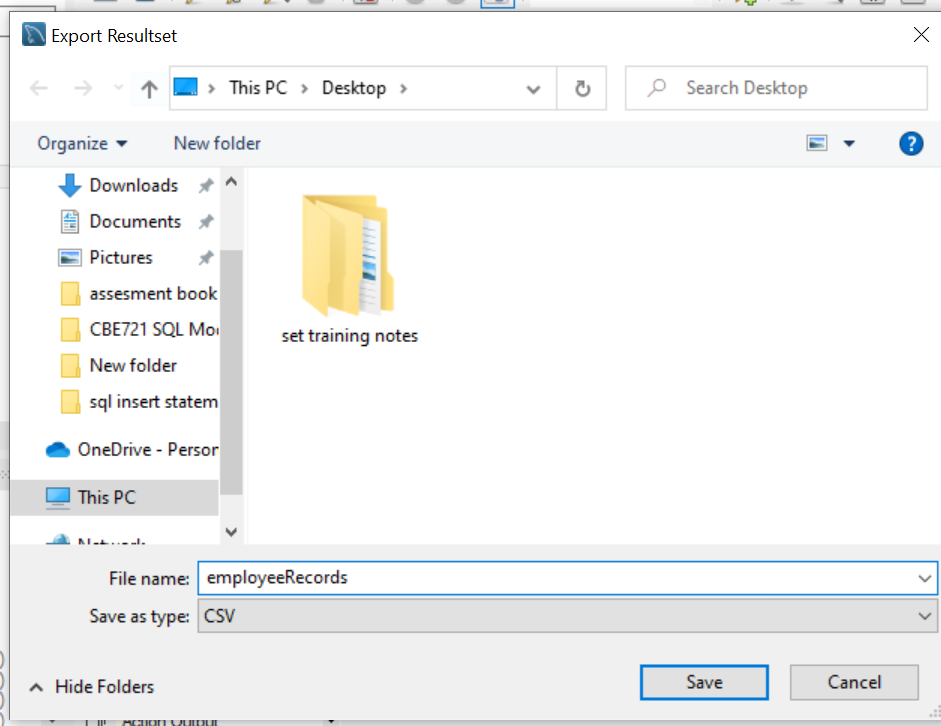
There are many techniques for exporting data within MySQL workbench one of the best techniques if to first execute the query that you that will get the data you to exports one you have an output you just need to click the export recorded set to external file button after clicking this it will open a form.

I used a select statement to get all the data from the employee table as that is what I want exported.

Method 1 for exporting:

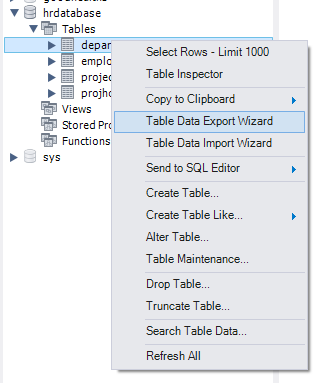
****

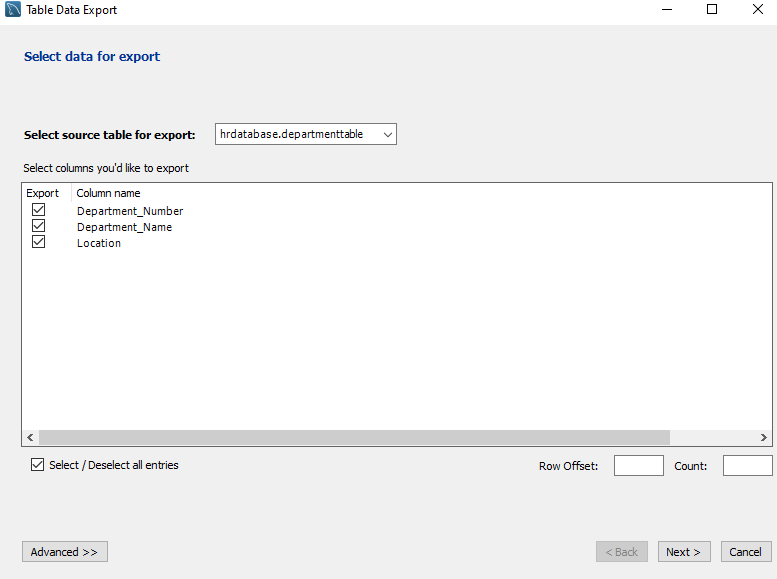
It then opened this form when I selected to save the file as a csv, You can select multiple different files types including ones I talked about in 8.1 like json depending on your preferences I then clicked save



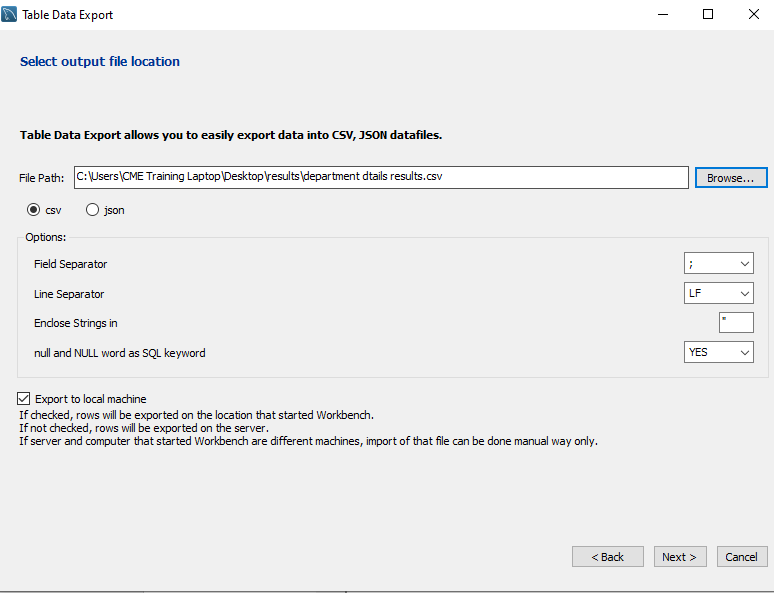
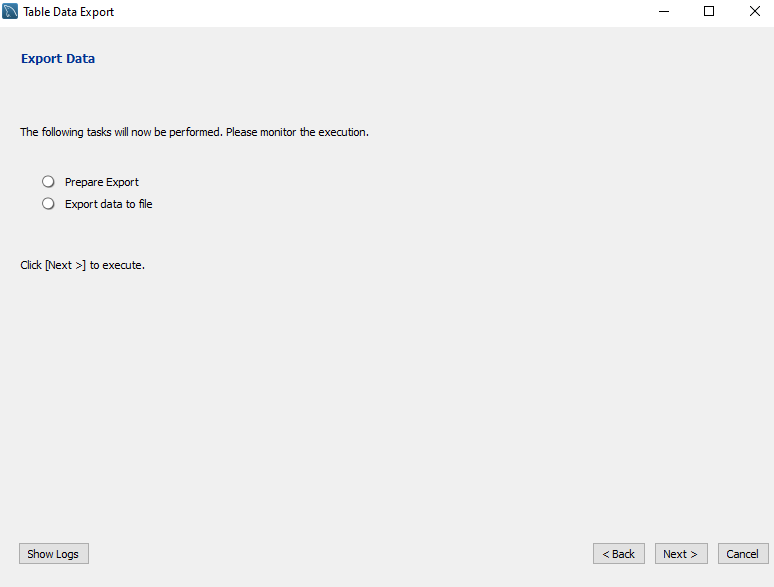
Method 2 for exporting

For the second method for exporting the data I just right clicked on the department table I wanted to export from the good health data base then clicked on the table data export wizard button.

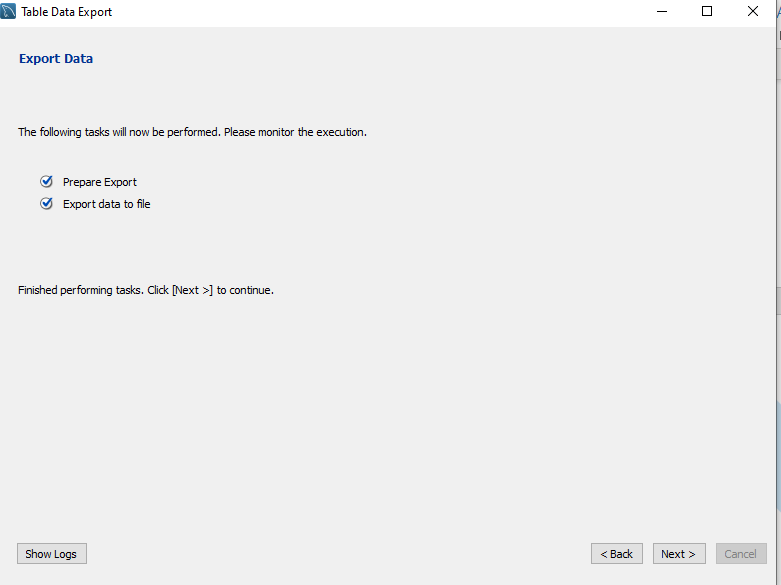


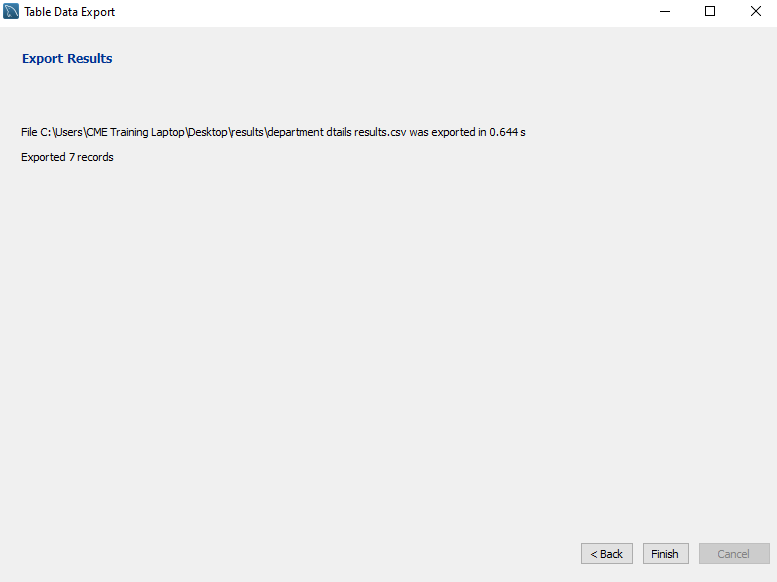
Next A form opened from here I just choose to export each of the columns from the table.

The next form that showed up is a form where I choose the location which the output would be exported to for this case I just exported the data to a results directory in my desktop.

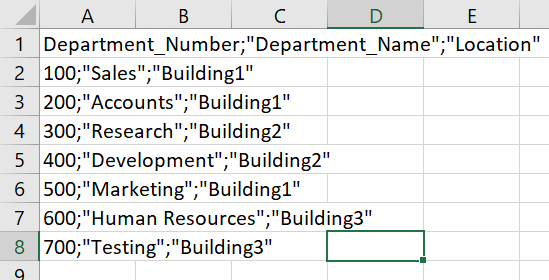
This form showed up next you just click next on this form.

After clicking next this next form shows that the data from the table has been successfully exported.

Finally, this form shows up to confirm that the export has complete successfully.



Export output

After exporting and opening the csv file in excel you should see the following.