Task 1.Normalized Forms

|  |  |  |  |
| --- | --- | --- | --- |
| UNF | 1NF | 2NF | 3NF |
| Student Details and Module Results |  |  |  |
| Name | Name (PK) | Name (PK) | Name (PK) |
| Tel number | Tel number | Tel number | Tel number |
| Email Address | Email Address | Email Address | Email Address |
| Module Taken | Module (PK) |  |  |
| Students start date | level | Module (PK) | Module (PK) |
| Result | Students start date | level | level |
| Student Type | Result | Students start date | Students start date |
|  | Student date | Result | Result |
|  | Name (FK) | Student date | Student date |
|  |  | Name (FK) | Name (FK) |
| Student Type |  |  |  |
| Type | Type (PK) | Type (PK) | Type (PK) |
| Discount | Discount | Discount | Discount |
|  |  |  |  |
| Modules |  |  |  |
| Module Title | Module Title(PK) | Module Title(PK) | Module Title (PK) |
| Level | Level | Level | Level (FK) |
| Duration | Duration | Duration |  |
| Class Tutor | Class Tutor(PK) | Class Tutor(PK) | Student (FK) (PK) |
| Class Day | Class Day | Class Day | Type (FK) (PK) |
| Class Time | Class Time | Class Time |  |
| Room | Room(FK) | Room(FK) | Module Title (PK) |
|  |  |  | Level |
|  |  |  | Duration |
|  |  |  | Class Tutor(PK) |
|  |  |  | Class Day |
|  |  |  | Class Time |
|  |  |  | Room(FK) |
| Rooms |  |  |  |
| Room Number | Room Number(PK) | Room Number(PK) | Room Number(PK) |
| Type | Type | Type | Type |
| Location | Location | Location | Location |
| Max No of Students | Max No of Students | Max No of Students | Max No of Students |
|  |  |  |  |
| Tutors |  |  |  |
| First Name | First Name(PK) | First Name(PK) | First Name(PK) |
| Surname | Surname(PK) | Surname(PK) | Surname(PK) |
| Specialism | Specialism(PK) | Specialism(PK) | Specialism(PK) |
| Hourly Rate | Hourly Rate | Hourly Rate | Hourly Rate |
| Day rate | Day rate | Day rate | Day rate |

1. Normalization is a database design procedure where data in tables are organized in a way to decreases redundancy and dependency of data. The procedure is done by dividing larger tables into smaller table’s linking them to each other using relationships. The rules guiding the normalization process which are done in stages from the first to a more advanced stage. Normalization achieves its best form mostly in the third normal form. (Connolly, 2004)

In the first normal form, the rule states that all data in the table must be atomic and unique values. In the Sidari summer database for instance under the Tutors table, the Specialism column contained data which aren’t atomic and unique. The Specialism had improvisation, trumpet, saxophone and theory. The first normal form rule breaks these tutors into new columns, that uniquely identify improvisation, trumpet, saxophone and theory to which tutors can be located. (Benyon-Davies, 2003)

The second normal form rule states that the table must first be in the first normal form and all non-key attributes should be fully dependent on the primary key. None-key attributes are columns in a table that are not used to uniquely identify an attribute in a table. In the Sidari Summer database, under the modules and classes table, the level dependent on the duration. In other to correct this, the school table was partitioned to create a new table known as module table which contain some students details. This made they non-key attributes depend on the primary key in both tables. (Connolly, 2004)

The third normal form rule states a relation is regarded in the third normal form if and only if it is first of all in the second normal form and every attribute on the table is dependent on the primary key and not on another non-key attribute. Looking at the current state of Sidari Summer all non-key attributes depend only on the primary key.

Normalization of all the samples of data to the third normal form helped create an ER diagram which showed how the entities relate to each other in the database.



Modul\_Result

Student Type

Student

M M 1

1 1

Module

M

M

1

1 1

Tutor

Room

Class

M M

M

1

Level

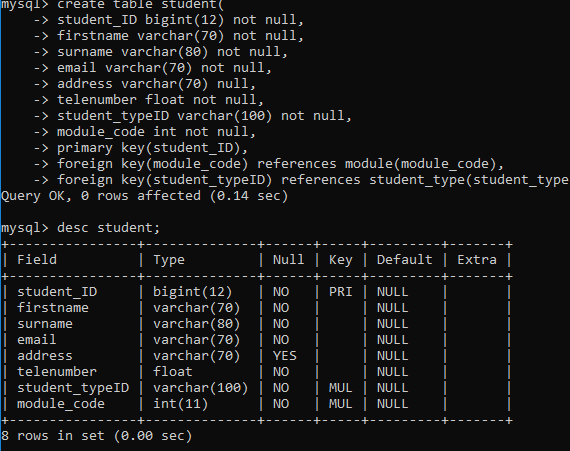
**Data dictionary**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Entity** | **Attributes** | **Data Type** | **Length** | **Constraint** | **Key** |
| Student | Student ID(PK) | Integer | 4\_\_ | Not Null | Primary Key |
| First \_name | character | 50 | Not null |  |
| Surname | character | 50 | Not null |  |
| Telephone Number | varchar | 50 | Not null |  |
| Email Address | varchar | 150 | Not null |  |
| Start Date | date | 5 | Not null |  |
| Student Type | varchar | 50 | Not null |  |
| Studennames Results | Results (PK) | character | 20 | Not Null | Primary key |
| Module Code (FK) | Integer | 5 | Not null | Foreign key |
| Student ID (FK) | Integer | 4 | Not null | Foreign key |
| Student Type | Student Type (PK) | varchar | 50 | Not Null | Primary Key |
| Discount | Integer | 5 | Not null |  |
| Module Result | Module Code (PK) | Integer | 5 | Not Null | Primary Key |
| Level code | Integer | 50 | Not null |  |
| Tutor (FK) | Integer | 3 | Not null | Foreign key |
| Module Title | varchar | 50 | Not null |  |
| Level | Level code(PK) | Integer | 2 | Not null | Primary Key |
| Duration | Integer | 3 | Not null |  |
|  |  |  | Not null |  |
| Classes | Class Code (PK) | Integer | 3 | Not Null | Primary Key |
| Tutor ID (FK) | Integer | 3 | Not Null | Foreign Key |
| Class Day | character | 150 | Not Null |  |
| Class Time | varchar | 4 | Not Null |  |
| Room Number (FK) | varchar | 3 | Not Null | Foreign Key |
| Room | Room Number(PK) | varchar | 3 | Not Null | Primary Key |
| Type | varchar | 150 | Not Null |  |
| Location | varchar | 150 | Not Null |  |
| Max Number of Students | Integer | 3 | Not Null |  |
| Tutor | Integer | Integer | 3 | Not Null | Primary Key |
| First Name | varchar | 50 | Not Null |  |
| Surname | varchar | 50 | Not Null |  |
| Specialism (FK) | varchar | 50 | Not Null | Foreign Key |
| Hourly Rate | Integer | 3 | Not Null |  |
| Day Rate | Integer | 4 | Not Null |  |

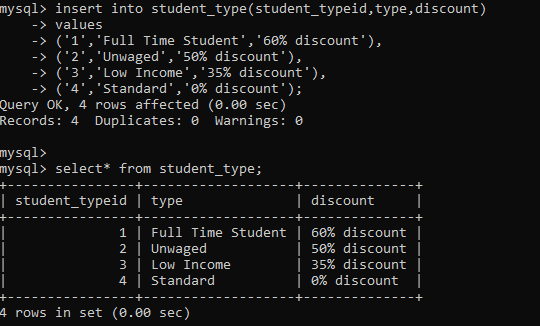
Task 2

Q1 Creating database

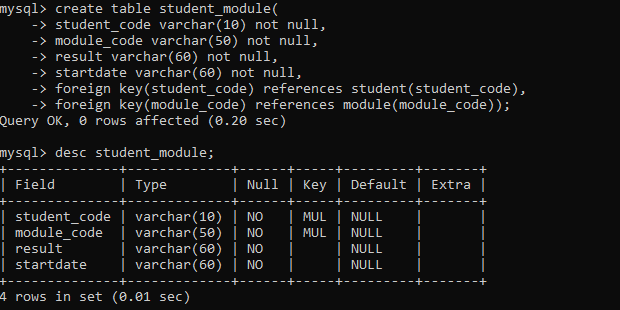
STUDENT



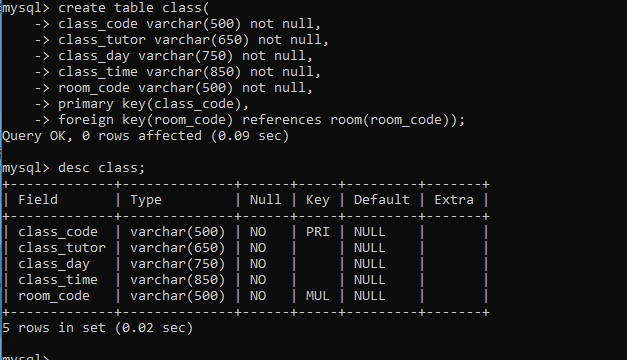
STUDENT TYPE



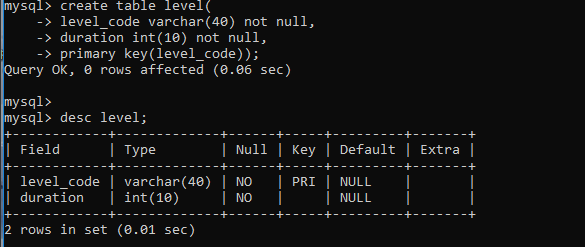
STUDENT\_MODULE



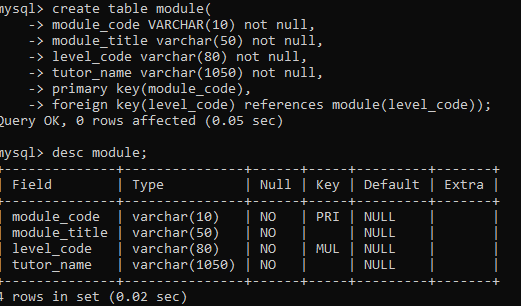
CLASS



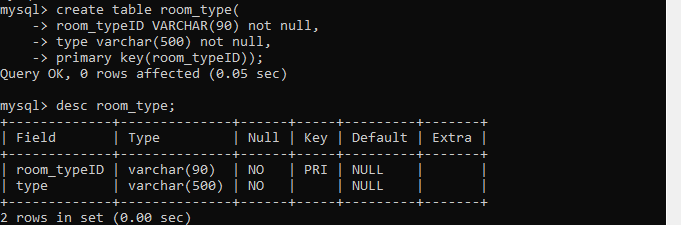
LEVEL



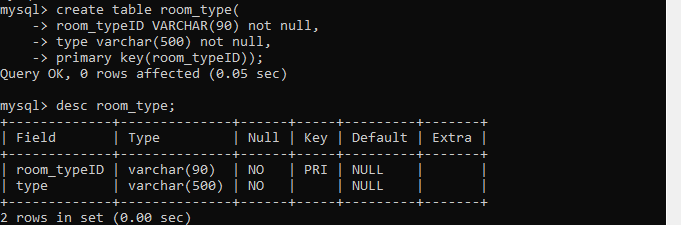
MODULE

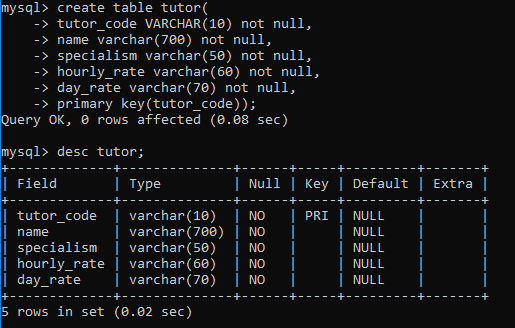


ROOM



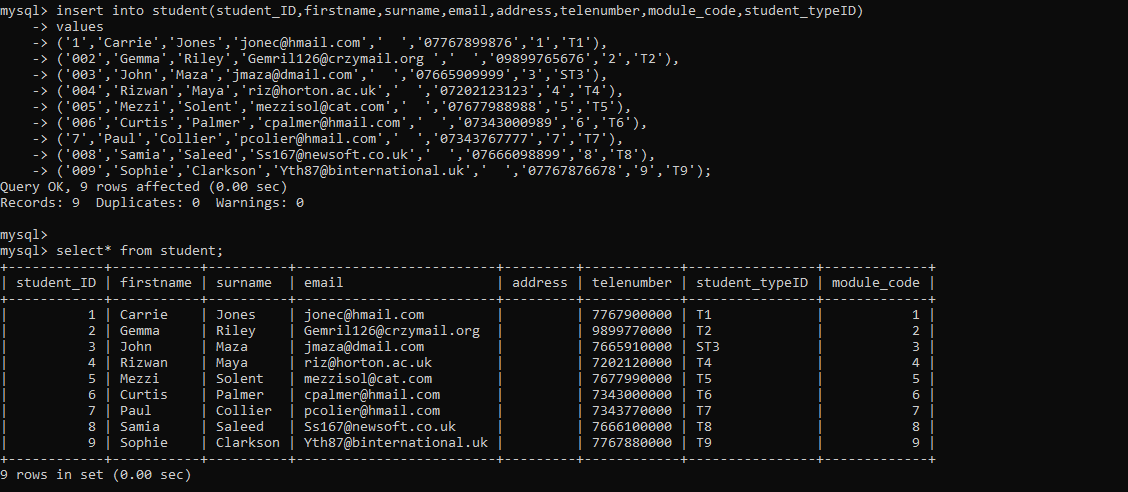
ROOM TYPE

TUTOR

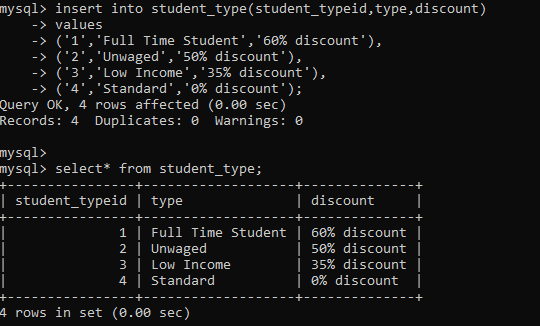


DATA ENTRY

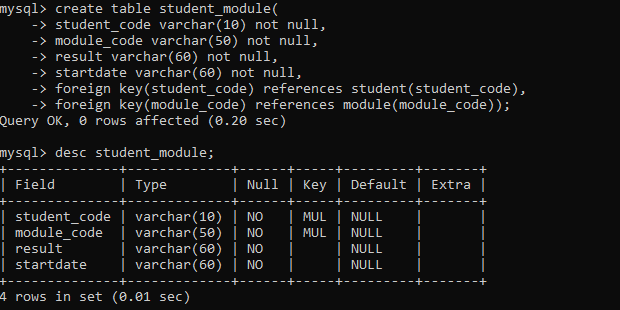
STUDENT



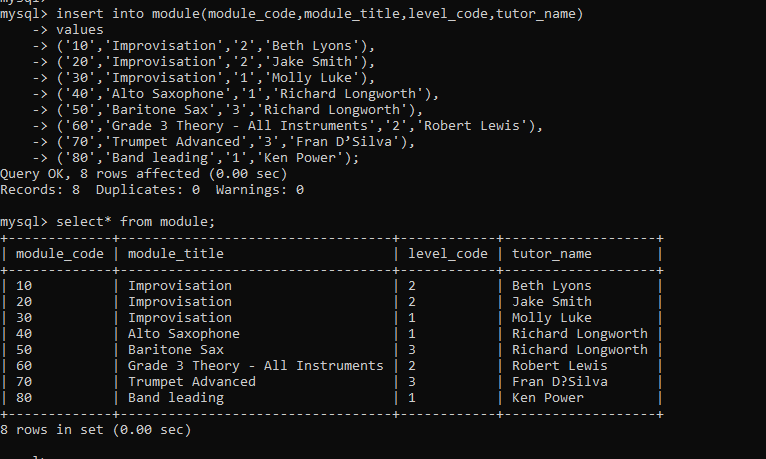
STUDENT TYPE



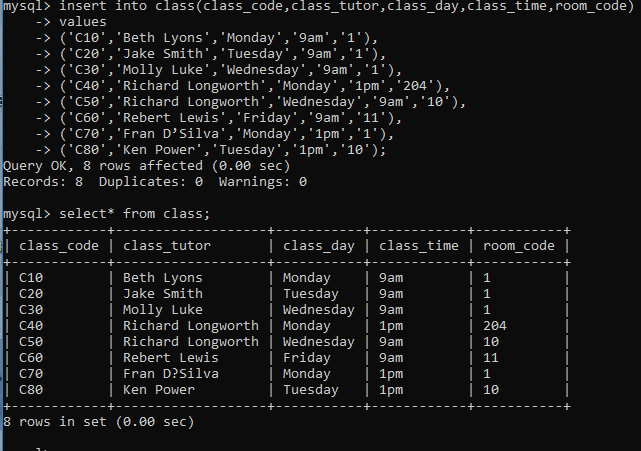
STUDENT MODULE



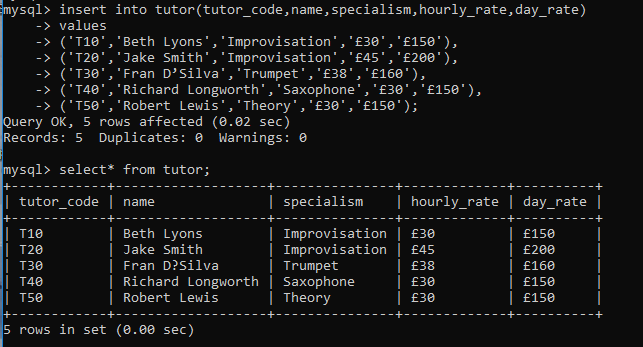
MODULE



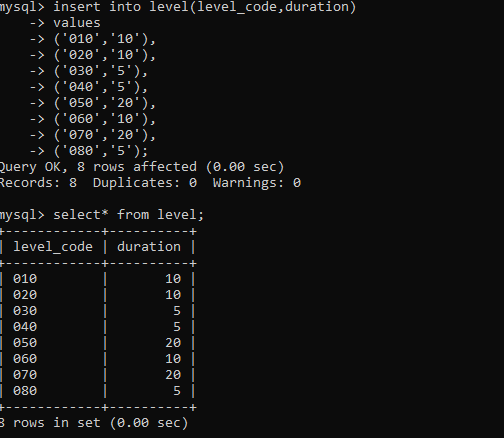
CLASS



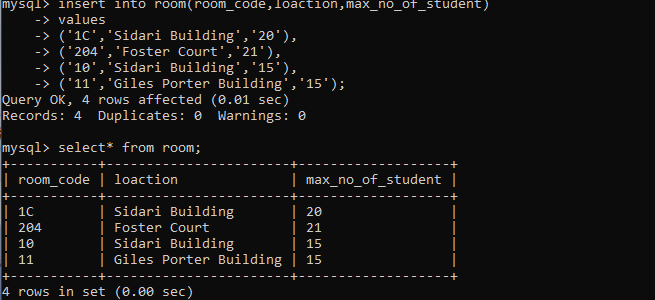
TUTOR



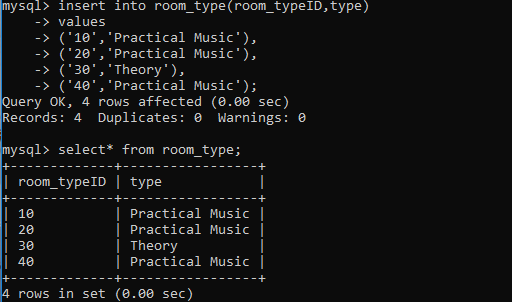
LEVEL



ROOM

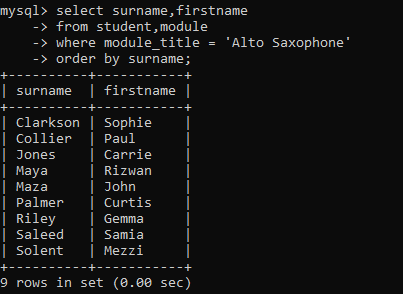


ROOM TYPE

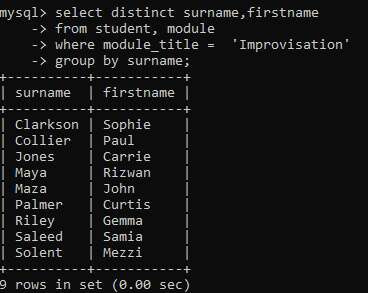


QUERIES

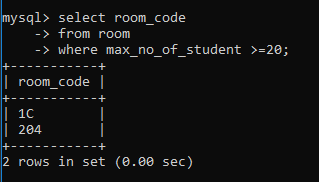
QUERY 8



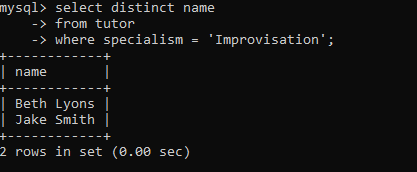
QUERY 9

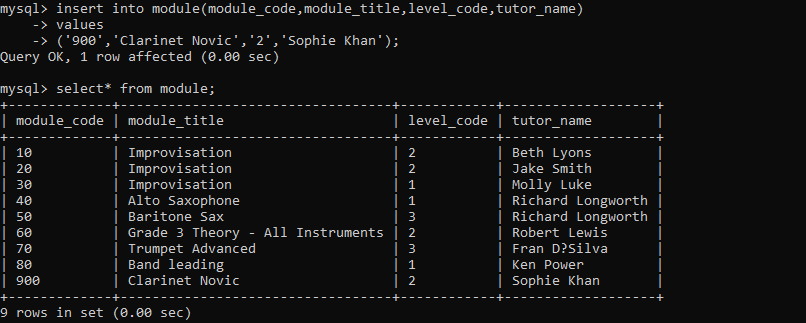


QUERY 10

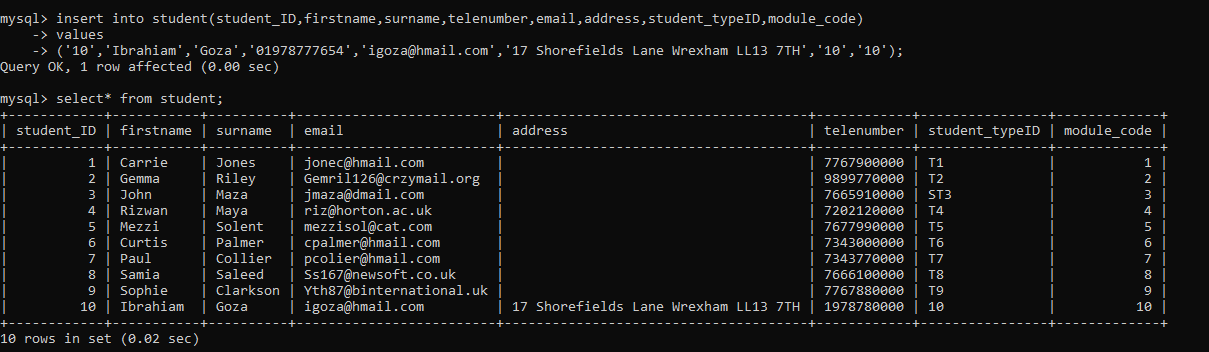


QUERY 12

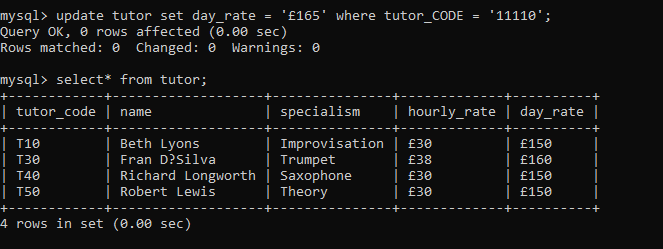
****

QUERY 13

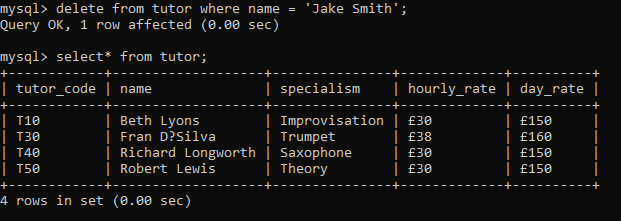
QUERY 14



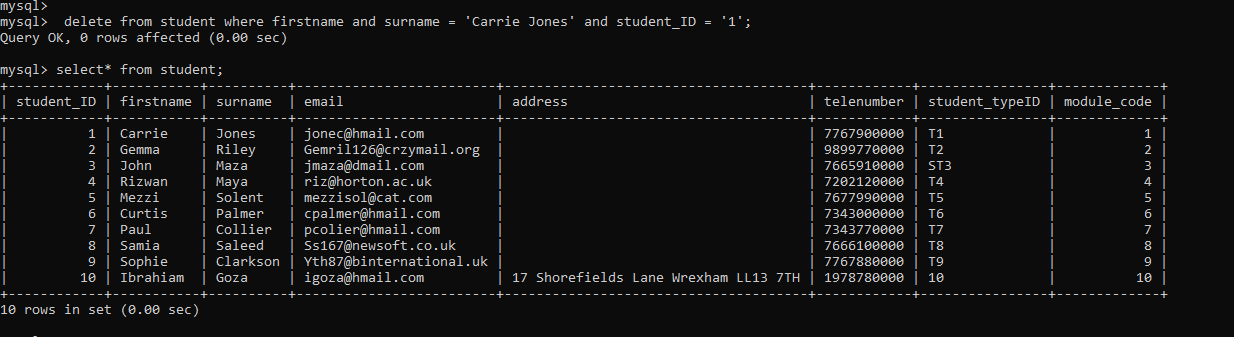
QUERY 15



QUERY 16



QUERY 17



Task3

Sidari summer school is a musical program takes place once in a year starting from May to September in summer takes place in the university room of Sidari University. Listed students are talented to study between 2 to 6 modules in every week and may house 70 students in every week who either stay in the universities hostels whiles students have left the hostels.

Sidari Summer School owns its present records on a different spreadsheet and on paper, but has recently taken pleasure in automating its management scheme having a database to retain accounts of Students, classes and instructors as it is developing more and general. The recent data of Sidari Summer school are in a composite organization which may create it very tough to generate a automate record.

Meeting System Requirements

Disbanding Names under the Entity Students\_Details in Surname and First\_Name, a primary key needed to be created to uniquely identify the table. That is the reason why I had Student\_Code as my primary key in the table.

Student Details and Module Result needed to be dissolved so I can have two tables from one table. I did that because Students\_Details can be an entity on its own likewise Module Results, that is why in the 3rd normal form we had Students Details and Module Result dissolved into Students\_Details and Student Module Results.

The same idea in the second bullet needed to be applied to Module and classes to have 2 tables from one bench. To help easily create the computerize records of Sidari.

Sidari Summer School had normalized from the first normal form over to the third normal form and with its EDR Diagram interlinked to get related information when needed to all entities.

By means of the data dictionary to state the data type, how voidable the data is and qualities of the various objects to see how the MySQL server would pile and know how to insert data.

Query statements were documented to choose, inset, generate, modernize and erase to go into and operate the database using MySQL on a Wamp Server.

How Requirements have been met

Management can have information on any entity whenever needed since they are all grouped in tables. The Name was divided into first name and surname and as well divided Modules and classes into two ways one of them. (Connolly, 2004)

# References

Benyon-Davies, P., 2003. *Database Systems,* s.l.: Palgrave Macmillan.

Connolly, T., 2004. *Database Systems Apractical Approach to Design Implementation and Management,* s.l.: Addison Wesley.