DipIT07 – Introduction to Database Systems

COURSEWORK – II

Student Id: NP03A180156

Student Name: Bijay Risal

B.Sc.(Hons)Computer Science

Submitted date: 2019/1/31

Submitted to: Dipson Shrestha

Marking Guidelines

SN	Title	Marks	Comments
A 1.1	Appropriateness of Research	/5	
A 1.2	Proper Listing of Clients of Researched Database	/10	
A 2.1	1NF description and example	/10	
A 2.2	2NF description and example	/5	
A 2.3	3NF description and example		
B 1.1	Identification of Entities and Attributes of Given Scenario	/15	
B 1.2	PK and FK Creation and Referential Integrity	/5	
C 1.1	ER Diagram	/10	
D 1.1	Proper use of Insert and Create Command	/10	
D 1.2	Proper use of Select Command	/5	
D 1.3	Proper use of Select Command	/5	
D 1.4	Proper use of Select Command	/5	
D 1.5	Proper use of Select Command	/5	
D 1.6	Proper use of Select and Group function Command	/10	
E	Viva	A/B/F	*F means fail in the

Acknowledgement:

This report for Introduction to Database Systems is the end result of researching about database itself and various features, where it is used and why it is used by on the internet, so I owe a great deal of gratitude to the people that gave me the opportunity to utilize the internet, especially my parents. I gratefully acknowledge the support of some of my friends and maybe the module leader too. This coursework could not have been completed without the generous duration given by the institution, so I thank the institution for that too.

Table of Contents

A. Research

A.1 MySQL for Small and Medium sized Organization

MySQL was firstly developed by MySQL AB, Swedish Company. Then it was acquired by Sun Microsystems in 2008 and then by oracle in 2010. Enterprises should obtain a commercial license and developers should have General Public License in order to use MySQL. MySQL is a relational database management System (RDBMS) based on Structured Query Language (SQL). It runs on virtually all platforms including Windows, Unix and Linux. So, it is platform independent. (Rouse, 2018)

MySQL System provides suitable environment for enterprises of medium and small size of organization or web pages. The number of user using MySQL is increasing day by day because of the features it has been providing to its users. For example: 58.7 percent of people working in small and medium sized Organizations of United Kingdom are using it. Irrespective of the exact definition it can quite clearly, compete effectively in the enterprise space. (Whitehorn, 2007)Some of the features that have made MySQL suitable for medium and small sized organization are:

1. Relational Database System:
   MySQL is a relational database system like the others. Being relational database data among table is established logically, no chance of data redundancy provides security and easy to use as it supports all type of data (ex: numbers, characters and data images etc…). MySQL enhances all the key Relational Databases features, including

   a) High performance b) Single-User and Multi-User c) Multi-Threaded d) Multi-Process e) Replication and Clustering f) BLOBs and UDFs (Binary Large Objects and User Defined Objects) g) Referential Integrity

h) ACID transactions (MySQL , 2019)

2. SQL compatibility:

   As it name suggests it is a Structured Query Language. SQL is a standardized language for querying, updating data and administrating of a database. Through the configuration setting SQL-mode, MySQL server behaves for the most part compatibility with various database systems. About the numbers of database systems there are SQL languages. MySQL closely follow to the current standard SQL although with significant barriers and large number of extensions.

3. Performance and Scalability:

   In the world of online, competition is always a screen touch or mouse click. It means rapid responses to customer inquiries and activities are therefore paramount. The database providing the service of web applications need to be fast to both simple and complex queries (read and write). Even of the workload (thousands of current connections) or data volume the extreme performance must be exhibited. One of the successful web properties need to be able to quickly adapt to a growing and expanding customer workload , without experiencing any interruption in customer service. The hallmark of MySQL is Performance and Scalability. So, many Web-businesses use MySQL. It utilizes a number of key strengths to deliver fast performance. (A MySQL Strategy Whitepaper, 2011)

4. Client/Server Architecture:

   MySQL is a client/server system. Many clients which communicate with the server; that is, the query data, save changes, etc. Users can run on the same computer as the server or on another computer (communication via a local network or the Internet). This allows a business to work on the database via different systems. (Search It Channel, 2019)

5. Enterprise Dashboard:

   Top feature of MySQL enterprise is Enterprise Dashboard. It saves DBA writing, maintain and checking scripts that compile, collect and report similar metrics. It assists DBAs in determining “how” and “where” to spend time. At a glance it provides view to health, availability, security and performance of all MySQL servers. With this feature it allows in easiness to scale to scale-out on MySQL without adding more DBAs.

6. MySQL Network Advisors and Rules:

   More than hundred Advisors Rules proactively monitors more than six hundred MySQL and OS specific metrics for problems before they occur. And it extends DBA skill set ti include MySQL expertise. MySQL network advisors rules find problems/tuning opportunities which DBAs cannot find themselves. With this feature it reduces downtime and performance issues by identifying mistakes and corrective actions before costly outages occur.

7. Customizable Monitors and Alerts

   This unique feature of MySQL saves time and effort by allowing DBAs to group and checks many servers as single entity. The user defined Advisor rules allows for small and medium sized organizations specific monitoring and alerts. Notifications (Simple Network Management Protocol and Simple Mail Transfer Protocol) can be tailored for individual or group recipients. With this feature it allows for customized SLAs across large MySQL deployments. (Kumar, 2009)

8. Platform Independence:

   MySQL can be executed under number of operating systems. MySQL runs over 20 platforms including Linux, Solaris, and Microsoft windows, BSDI, AIX, OS X, SGI Iris and Mac OS delivering a solution on the platform of their own choice and giving organizations complete flexibility.

9. Reduce TCO:

   MySQL is a proven cost-effective database solution which reduces the cost of database infrastructure by more than 90% than others.

   MySQL reduces the TCO (Total Cost of Ownership) of database software because of following reasons:

   - Reduces database licensing costs by over 90%
   - Lowers hardware expenditure by 70%
   - Cuts systems downtime by 60%
   - Reduces administration, engineering and supports costs up to 50% (MySQL , 2019)

Five clients (Small and medium organizations) using MySQL are:

- Adobe
- Plixer
- Quantel
- Danfoss
- Fujitsu (MySQL, 2019)

A.2 Database Normalization

Database Normalization is the process of organizing data in the database. It is the systematic process of eliminating data redundancy and undesirable characteristics like Update, Insertion and Deletion Anomalies. If the data is not normalized and have data repetition then it would occupy more space and makes difficult in handling operations like deletion and insertion. The added advantage of getting an organized data enhances the performance level. The main purpose of normalization is:

Increased consistency: Database normalization decreases the repetition of data placing data in one place and one place only, reducing the possibility of inconsistent data.

Easier object-to-data mapping: Database maintained under normalization rules in general are conceptually closer to object-oriented schemas as the object-oriented achievements of enhancing high cohesion and loose coupling between classes results in similar solutions.

Normalization rules are divided into following normal forms:

- First Normal Form
- Second Normal Form
- Third Normal Form
- BCNF (Boyce-Code Normal Form)

A.2.1 First Normal Form

First Normal Form (1NF) gives the fundamental rules for database normalization and connects to a single table within a relational database system.

Characteristics of INF

- Column in the table must be unique
- For each set of related data there should be separate table
- Each table must be denoted or named with a unique column or concatenated columns called the primary key
- No rows in the table may be duplicated
- No columns in the table may be duplicated
- No row/column intersections contain a null value
- No row/column intersections contain multivalued fields

Three steps for making 1NF tables

1st Step:

Multivalued columns of the table must be modified to make sure that each column does not take more than one entry.

2nd Step:

Removing repeated/duplicated values from a table is the next step toward first normalized form.

3rd Step:

The last step for implementing 1NF is having atomicity of data. (Techopedia , 2017)

Database Atomicity:

Database Atomicity states that a domain should have values in the relation which are impossible to be broken down into smaller contents of data, with respect to DBMS. An atomic value cannot be decomposed into smaller pieces. (Geeks For Geeks, 2018)

1NF example

Suppose a company wants to store the information of its employees. It creates a table that looks like below:

Emp_ID	Emp_name	Emp_address	Emp_mobile
101	Bijay Risal	Kathmandu	9869835740
102	Jebeen Rai	Biratnagar	9844293600
103	Shyam Magar	Palpa	9844435834 9844435832
104	Jalan Limbu	Kotang	9854035740

Employee Shyam Magar is having two mobile numbers so the company as stored two mobile numbers in the same column.

This table does not follow 1NF rules. As the rules says "each attribute of a table must have atomic or single values", the emp_mobile values for Shyam Magar violates 1NF rule.

To make the table following 1NF rules the company should have data like this:

Emp_ID	Emp_name	Emp_address	Emp_mobile
101	Bijay Risal	Kathmandu	9869835740
102	Jebeen Rai	Biratnagar	9844293600
103	Shyam Magar	Palpa	9844435834
104	Shyam Magar	Palpa	9844435832
105	Jalan Limbu	Kotang	9854035740

(SINGH, 2015)

A.2.2 Second Normal Form

An entity is said to be in the second normal form when it is already in first normal form and all the attributes contained within it are dependent solely on the unique identifier of the entity. In other words, the table is in the 2NF (Second Normal Form) if it satisfies the following two conditions:

- It is in 1NF (first normal form)
- All non-key attributes are fully functional dependent on the primary key

2NF Example

Suppose a school wanted to store data of teacher and the subject taught by them. It creates a table that looks like below:

Teacher_Id	Subject	Teacher_age
101	Computer architecture	30
101	Web development	30
102	Programming	30
103	Data structure	27
103	Introduction to database	27

Candidate Keys: {Teacher_Id, Subject}

Non-prime attribute: Teacher_age

The table has followed 1NF rules as each attribute has atomic values. However, it has not followed 2NF rules as non-prime attribute Teacher_age is dependent on Teacher_id alone which is a proper subset of candidate key. This violates the rule of 2NF as the rule says "no non-prime attribute is dependent on the proper subset of any candidate key of the table".

To make the table following 2NF rules the school should have break table like this:

Teacher_details table

Teacher_Id	Teacher_age
101	30
102	30
103	27

Teacher_Subject table:

Teacher_Id	Subject
101	Computer architecture

101	Web development
102	Programming
103	Data structure
103	Introduction to database

Now these two tables complies with Second Normal Form (2NF)

A.2.3 Third Normal Form

This Third normal form is the third step in normalizing a database and it builds on the first and second normal forms, 1NF and 2NF.In other words the table is in the 3NF if it satisfies the following criteria:

- Table in 2NF
- Table has no transitive functional dependencies.

Third Normal Form states that the column reference data which are not dependent on the primary key should be vanished.

3NF Example

Suppose a Project wanted to store the information of employee Address. It creates a table that looks like below:

Prj_id	Prj_name	Prj_zip	Prj_State_No	Prj_city	Prj_district
1001	Bijay	181005	First	Dharan	Sunsari
1002	Prabesh	111008	Second	Malangwa	Sarlahi
1003	Sadish	181007	Third	Kathmandu	Ktm

1001	Sangam	191008	Fourth	Pokhara	Kaski
1200	Aman	111999	Fifth	Dhangadi	Kailali

Super keys: {Prj_id}, { Prj _id, Prj _name}, { Prj _id, Prj _name, Prj _zip}…so on

Candidate Keys: {Prj_id}

Non-prime attributes: All attributes except Prj_id are non-primes they are not part of any candidate keys.

Here, Prj_state, Prj_city & Prj_district dependent on Prj_zip. And Prj_zip is dependent on Prj_id that makes non-prime attributes (Prj_state, Prj_city & Prj_district) transitively dependent on super key (Prj_id). This violates the rule of 3NF.

To make this table following with 3NF rules we have to break the table into two tables to remove the transitive dependency:

Project Employee table

Prj_id	Prj_name	Prj_zip
1001	Bijay	181005
1002	Prabesh	111008
1003	Sadish	181007
1001	Sangam	191008
1200	Aman	111999

Project Employee Table

Prj_zip	Prj_State_No	Prj_city	Prj_district
181005	First	Dharan	Sunsari
111008	Second	Malangwa	Sarlahi
181007	Third	Kathmandu	Ktm
191008	Fourth	Pokhara	Kaski
111999	Fifth	Dhangadi	Kailali

Now these two tables comply with 3NF. (SINGH, 2015)

B. Relational Model

B.1 Entities and Attributes

Entity Student

S.N	Attributes	Data Type	Data size	Keys
1	Student_Id	Varchar	15	Primary(not null)
2	Student_Name	Varchar	20	- (not null)
3	Student_Stream	Varchar	20	-
4	Student_Age	Integer	2	-
5	Student_Address	Varchar	20	-
6	Student_Phone	Integer	10	-
7	Student_Email	Varchar	20	-
8	Student_DOB	Date	10	-
9	Guardian_Name	Varchar	20	-
10	Guardian_Phone	Integer	10	-

Entity Employee

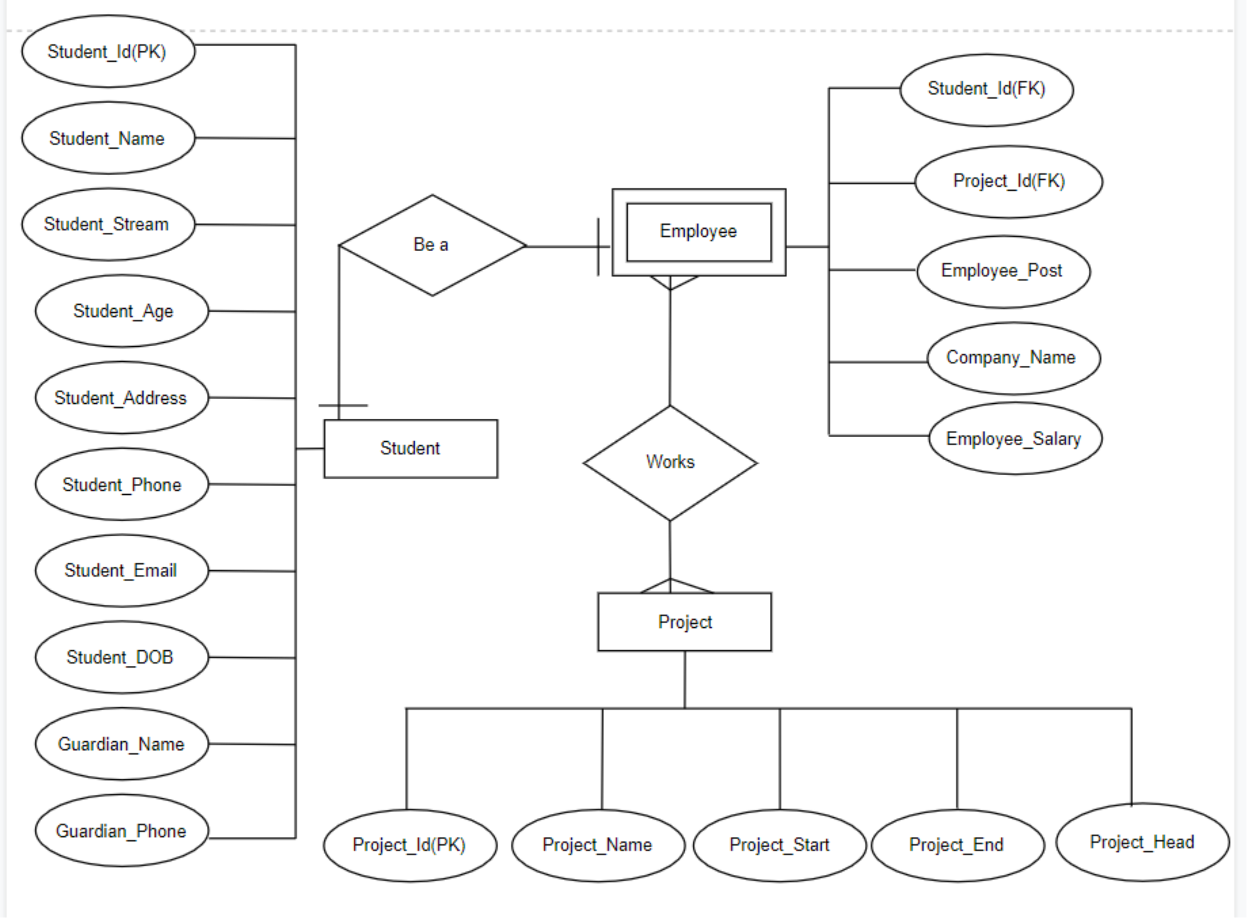
S.N	Attribute	Data Type	Data size	Keys
1	Student _Id	Varchar	15	Foreign(Not null)
2	Project_Id	Varchar	20	Foreign(Not null)
3	Company_name	Varchar	20	-
4	Employee_Post	Varchar	15	_
5	Employee_salary	Integer	10	_

Entity Project

S.N	Attribute	Data type	Data size	Keys
1	Project_Id	Varchar	10	Primary(Not null)
2	Project_Name	Varchar	15	-
3	Project_Start	Date	10	-
4	Project_End	Date	10	-
5	Project_Head	Varchar	20	_

C. DBMS

C.1 ER Diagram

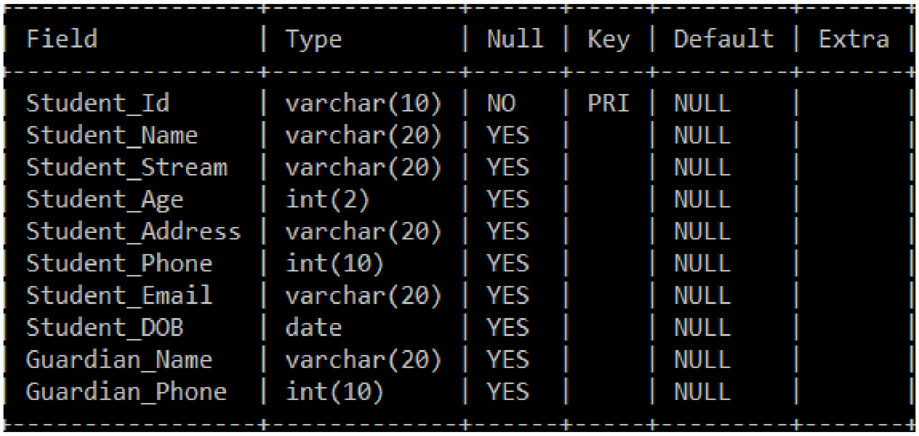


D. SQL

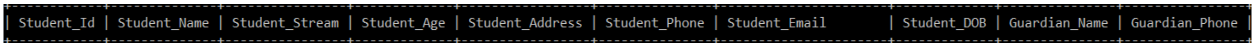
D.1) Create table and Insert at least 10 records for each

I) creating Student table

SQL: CREATE TABLE Student(Student_Id Varchar(10) primary key not null,Student_Name Varchar(20),Student_Stream Varchar(20), Student_Age Integer(2), Student_Address Varchar(20), Student_Phone Integer(10), Student_Email Varchar(20), Student_DOB date, Guardian_Name Varchar(20), Guardian_Phone Integer(10));



```
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| Student_Id      | varchar(10) | NO   | PRI | NULL    |       |
| Student_Name    | varchar(20) | YES  |     | NULL    |       |
| Student_Stream  | varchar(20) | YES  |     | NULL    |       |
| Student_Age     | int(2)      | YES  |     | NULL    |       |
| Student_Address | varchar(20) | YES  |     | NULL    |       |
| Student_Phone   | int(10)     | YES  |     | NULL    |       |
| Student_Email   | varchar(20) | YES  |     | NULL    |       |
| Student_DOB     | date        | YES  |     | NULL    |       |
| Guardian_Name   | varchar(20) | YES  |     | NULL    |       |
| Guardian_Phone  | int(10)     | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
```



```
+------------+--------------+----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| Student_Id | Student_Name | Student_Stream | Student_Age | Student_Address | Student_Phone | Student_Email        | Student_DOB | Guardian_Name | Guardian_Phone |
+------------+--------------+----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
```

II) Adding values into Student

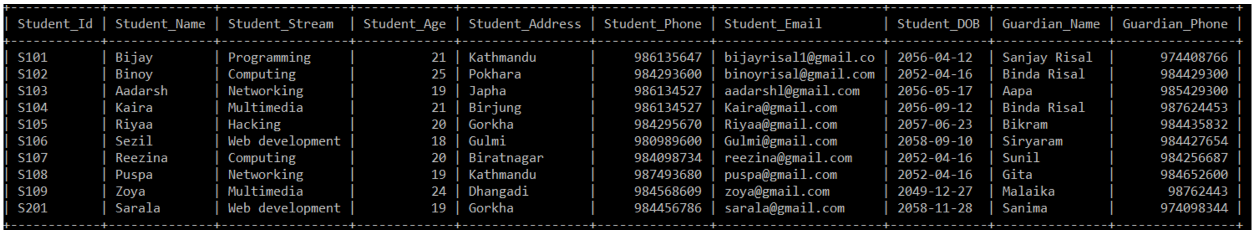
SQL:

INSERT INTO Student VALUES

("S101","Bijay","Programming", 21,"Kathmandu",'986135647',"bijayrisal1@gmail.com","2056-04-12","Sanjay Risal",'974408766');

```
INSERT INTO Student VALUES

 (“S102”,”Binoy”,”Computing”,25,”Pokhara”,’984293600’,”binoyrisal@gmail.com”,”2052-04-16”,”Binda Risal”,’984429300’);

INSERT INTO Student VALUES

 (“S103”,”Aadarsh”,”Networking”,19,”Japha”,’9844293450’,”aadarshl@gmail.com”,”2056-05-17”,”Aapa”,’985429300’);

INSERT INTO Student VALUES

 (“S104”,”Kaira”,”Multimedia”,21,”Birjung”,’9843493600’,”Kaira@gmail.com”,”2053-06-31”,”Binda Risal”,’9844345668’);

INSERT INTO Student VALUES

 (“S105”,”Riyaa”,”Hacking”,20,”Gorkha”,’984295670’,”Riyaa@gmail.com”,”2057-06-23”,”Bikram”,’984435832’);

INSERT INTO Student VALUES

 (“S106”,”Sezil”,”Web  development”,18,”Gulmi”,’980989600’,”Gulmi@gmail.com”,”2058-09-10”,”Siryaram”,’984427654’);

INSERT INTO Student VALUES

 (“S107”,”Reezina”,”Computing”,20,”Biratnagar”,’984098734’,”reezina@gmail.com”,”2052-04-16”,”Sunil”,’984256687’);

INSERT INTO Student VALUES

 (“S108”,”Puspa”,”Networking”,19,”Kathmandu”,’987493680’,”puspa@gmail.com”,”2052-04-16”,”Gita”,’984652600’);

INSERT INTO Student VALUES

(“S109”,”Zoya”,”Multimedia”,24,”Dhangadi”,’984568609’,”zoya@gmail.com”,”2049-12-27”,”Malaika”,’98762443’);

INSERT INTO Student VALUES
```

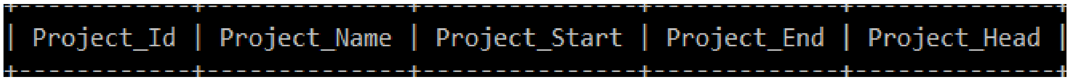
("S201","Sarala","Web development",19,"Gorkha",'984456786',"sarala@gmail.com","2058-11-28","Sanima",'974098344');



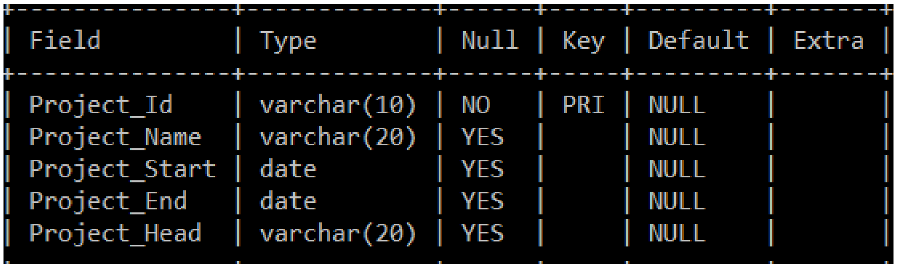
```
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| Student_Id | Student_Name | Student_Stream  | Student_Age | Student_Address | Student_Phone | Student_Email        | Student_DOB | Guardian_Name | Guardian_Phone |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| S101       | Bijay        | Programming     |          21 | Kathmandu       |     986135647 | bijayrisal1@gmail.co | 2056-04-12  | Sanjay Risal  |      974408766 |
| S102       | Binoy        | Computing       |          25 | Pokhara         |     984293600 | binoyrisal@gmail.com | 2052-04-16  | Binda Risal   |      984429300 |
| S103       | Aadarsh      | Networking      |          19 | Japha           |     986134527 | aadarshl@gmail.com   | 2056-05-17  | Aapa          |      985429300 |
| S104       | Kaira        | Multimedia      |          21 | Birjung         |     986134527 | Kaira@gmail.com      | 2056-09-12  | Binda Risal   |      987624453 |
| S105       | Riyaa        | Hacking         |          20 | Gorkha          |     984295670 | Riyaa@gmail.com      | 2057-06-23  | Bikram        |      984435832 |
| S106       | Sezil        | Web development |          18 | Gulmi           |     980989600 | Gulmi@gmail.com      | 2058-09-10  | Siryaram      |      984427654 |
| S107       | Reezina      | Computing       |          20 | Biratnagar      |     984098734 | reezina@gmail.com    | 2052-04-16  | Sunil         |      984256687 |
| S108       | Puspa        | Networking      |          19 | Kathmandu       |     987493680 | puspa@gmail.com      | 2052-04-16  | Gita          |      984652600 |
| S109       | Zoya         | Multimedia      |          24 | Dhangadi        |     984568609 | zoya@gmail.com       | 2049-12-27  | Malaika       |       98762443 |
| S201       | Sarala       | Web development |          19 | Gorkha          |     984456786 | sarala@gmail.com     | 2058-11-28  | Sanima        |      974098344 |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
```

I) Creating table Project

SQL: Create table Project (Project_Id Varchar(10) primary key not null, Project_Name Varchar(20), Project_Start date, Project_End date, Project_Head Varchar(20));



```
+------------+--------------+---------------+-------------+--------------+
| Project_Id | Project_Name | Project_Start | Project_End | Project_Head |
+------------+--------------+---------------+-------------+--------------+
```



```
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Project_Id    | varchar(10) | NO   | PRI | NULL    |       |
| Project_Name  | varchar(20) | YES  |     | NULL    |       |
| Project_Start | date        | YES  |     | NULL    |       |
| Project_End   | date        | YES  |     | NULL    |       |
| Project_Head  | varchar(20) | YES  |     | NULL    |       |
```

II) Adding values into Project

SQL:

INSERT INTO Project VALUES ("P101","Tamor","2015-09-10","2019-03-09","Reziina");

INSERT INTO Project VALUES ("P102","Rapti","2016-09-10","2020-03-09","Prabesh");

INSERT INTO Project VALUES ("P103","Karnali","2017-09-10","2020-03-09","Bhai");

INSERT INTO Project VALUES ("P104","RawaKhola","2021-09-10","2019-03-09","Sir EL");

INSERT INTO Project VALUES (“P105”,”Lalbandi”,”2015-02-10”,”2019-04-30”,”Sadish”);

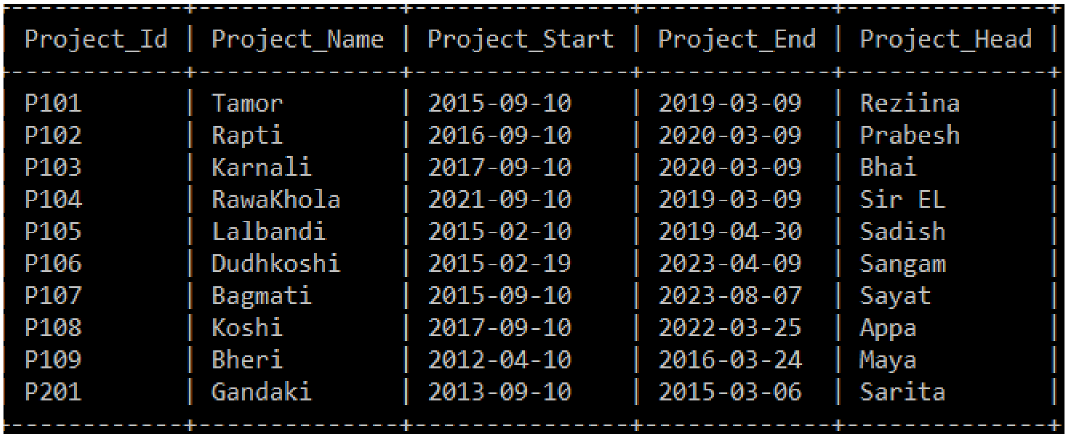
INSERT INTO Project VALUES (“P106”,”Dudhkoshi”,”2015-02-19”,”2023-04-09”,”Sangam”);

INSERT INTO Project VALUES (“P107”,”Bagmati”,”2015-09-10”,”2023-08-07”,”Sayat”);

INSERT INTO Project VALUES (“P108”,”Koshi”,”2017-09-10”,”2022-03-25”,”Appa”);

INSERT INTO Project VALUES (“P109”,”Bheri”,”2012-04-10”,”2016-03-24”,”Maya”);

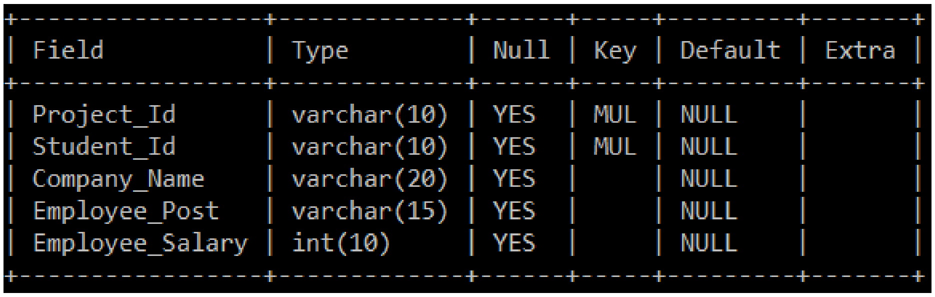
INSERT INTO Project VALUES (“P201”,”Gandaki”,”2013-09-10”,”2015-03-06”,”Sarita”);



```
------------+--------------+---------------+-------------+--------------+
 Project_Id | Project_Name | Project_Start | Project_End | Project_Head |
------------+--------------+---------------+-------------+--------------+
 P101       | Tamor        | 2015-09-10    | 2019-03-09  | Reziina      |
 P102       | Rapti        | 2016-09-10    | 2020-03-09  | Prabesh      |
 P103       | Karnali      | 2017-09-10    | 2020-03-09  | Bhai         |
 P104       | RawaKhola    | 2021-09-10    | 2019-03-09  | Sir EL       |
 P105       | Lalbandi     | 2015-02-10    | 2019-04-30  | Sadish       |
 P106       | Dudhkoshi    | 2015-02-19    | 2023-04-09  | Sangam       |
 P107       | Bagmati      | 2015-09-10    | 2023-08-07  | Sayat        |
 P108       | Koshi        | 2017-09-10    | 2022-03-25  | Appa         |
 P109       | Bheri        | 2012-04-10    | 2016-03-24  | Maya         |
 P201       | Gandaki      | 2013-09-10    | 2015-03-06  | Sarita       |
------------+--------------+---------------+-------------+--------------+
```

I) Creating Table Employee

SQL: Create table Employee (foreign key(Project_ID) references Project(Project_Id), foreign key(Student_Id) references Student(Student_Id),Company_Name Varchar(20),Employee_Post Varchar(15), Employee_Salary Integer (10));



```
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| Project_Id      | varchar(10) | YES  | MUL | NULL    |       |
| Student_Id      | varchar(10) | YES  | MUL | NULL    |       |
| Company_Name    | varchar(20) | YES  |     | NULL    |       |
| Employee_Post   | varchar(15) | YES  |     | NULL    |       |
| Employee_Salary | int(10)     | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
```

II) Adding values into the Employee

SQL: Insert into Employee Values(“P101”,”S102”,”3D Com”,”Manager”,650000);

Insert into Employee Values(“P102”,”S103”,”JavaTpoint”,”CEO”,500000);

Insert into Employee Values(“P103”,”S101”,”Mero Rojgar”,”Designer”,30000);

Insert into Employee Values(“P104”,”S105”,”Teach me”,”Security”,40000);

Insert into Employee Values(“P105”,”S106”,”Hamro Sansar”,”Reporter”,20000);

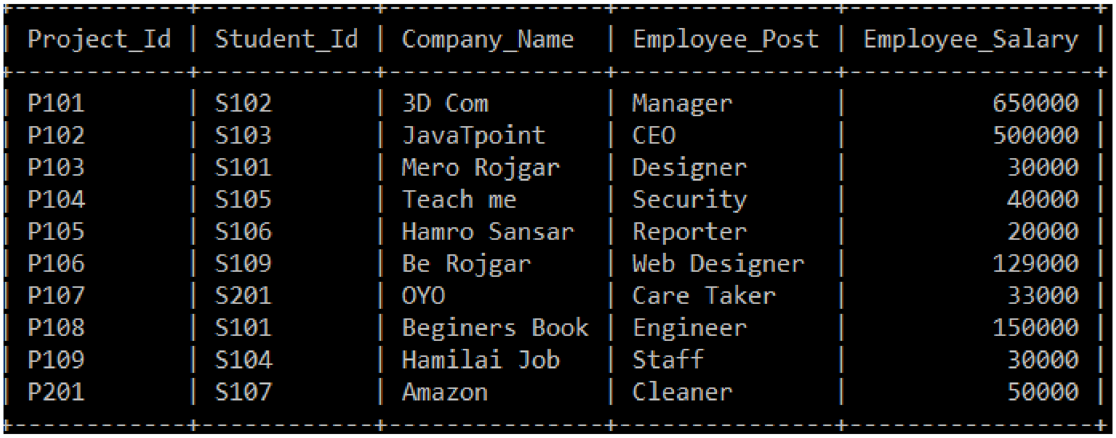
Insert into Employee Values(“P106”,”S109”,”Be Rojgar”,”Web Designer”,129000);

Insert into Employee Values(“P107”,”S201”,”OYO”,”Care Taker”,33000);

Insert into Employee Values(“P108”,”S101”,”Beginers Book”,”Engineer”,150000);

Insert into Employee Values(“P109”,”S104”,”Hamilai Job”,”Staff”,30000);

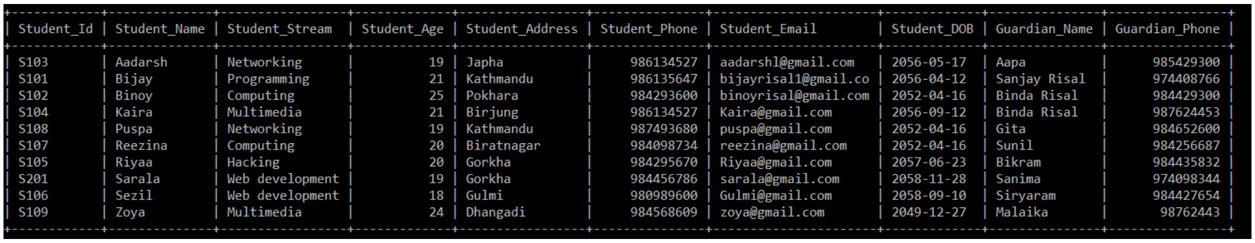
Insert into Employee Values(“P201”,”S107”,”Amazon”,”Cleaner”,50000);



```
+------------+------------+---------------+---------------+-----------------+
| Project_Id | Student_Id | Company_Name  | Employee_Post | Employee_Salary |
+------------+------------+---------------+---------------+-----------------+
| P101       | S102       | 3D Com        | Manager       |          650000 |
| P102       | S103       | JavaTpoint    | CEO           |          500000 |
| P103       | S101       | Mero Rojgar   | Designer      |           30000 |
| P104       | S105       | Teach me      | Security      |           40000 |
| P105       | S106       | Hamro Sansar  | Reporter      |           20000 |
| P106       | S109       | Be Rojgar     | Web Designer  |          129000 |
| P107       | S201       | OYO           | Care Taker    |           33000 |
| P108       | S101       | Beginers Book | Engineer      |          150000 |
| P109       | S104       | Hamilai Job   | Staff         |           30000 |
| P201       | S107       | Amazon        | Cleaner       |           50000 |
+------------+------------+---------------+---------------+-----------------+
```

D2) Display the name of Student in alphabetical order

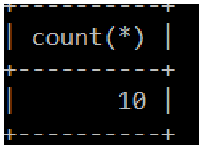
SQL: select * from student order by Student_Name asc;



```
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| Student_Id | Student_Name | Student_Stream  | Student_Age | Student_Address | Student_Phone | Student_Email        | Student_DOB | Guardian_Name | Guardian_Phone |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| S103       | Aadarsh      | Networking      |          19 | Japha           |     986134527 | aadarshl@gmail.com   | 2056-05-17  | Aapa          |      985429300 |
| S101       | Bijay        | Programming     |          21 | Kathmandu       |     986135647 | bijayrisal1@gmail.co | 2056-04-12  | Sanjay Risal  |      974408766 |
| S102       | Binoy        | Computing       |          25 | Pokhara         |     984293600 | binoyrisal@gmail.com | 2052-04-16  | Binda Risal   |      984429300 |
| S104       | Kaira        | Multimedia      |          21 | Birjung         |     986134527 | Kaira@gmail.com      | 2056-09-12  | Binda Risal   |      987624453 |
| S108       | Puspa        | Networking      |          19 | Kathmandu       |     987493680 | puspa@gmail.com      | 2052-04-16  | Gita          |      984652600 |
| S107       | Reezina      | Computing       |          20 | Biratnagar      |     984098734 | reezina@gmail.com    | 2052-04-16  | Sunil         |      984256687 |
| S105       | Riyaa        | Hacking         |          20 | Gorkha          |     984295670 | Riyaa@gmail.com      | 2057-06-23  | Bikram        |      984435832 |
| S201       | Sarala       | Web development |          19 | Gorkha          |     984456786 | sarala@gmail.com     | 2058-11-28  | Sanima        |      974098344 |
| S106       | Sezil        | Web development |          18 | Gulmi           |     980989600 | Gulmi@gmail.com      | 2058-09-10  | Siryaram      |      984427654 |
| S109       | Zoya         | Multimedia      |          24 | Dhangadi        |     984568609 | zoya@gmail.com       | 2049-12-27  | Malaika       |       98762443 |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
```

D3) Count the Total Number of employments.

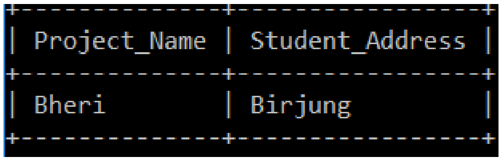
SQL: Select count(*) from Employee;



```
+----------+
| count(*) |
+----------+
|       10 |
+----------+
```

D4) Display the Project Name and Address of the student whose name is starting with 'K'.

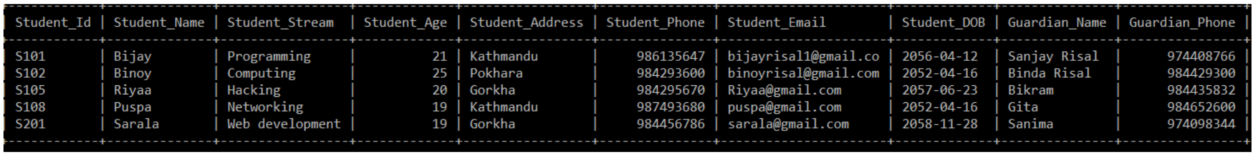
```
SQL:     select Project.Project_Name, Student_Address from Employee
         inner join Student on Employee.Student_Id=Student.Student_Id
         inner join project on Employee.Project_Id=Project.Project_Id
         where Student.Student_Name like 'K%';
```



```
+--------------+-----------------+
| Project_Name | Student_Address |
+--------------+-----------------+
| Bheri        | Birjung         |
+--------------+-----------------+
```

D5) Select all information about students who belongs to Pokhara, Kathmandu and Gorkha.

```
SQL:          select * from student where Student_Address in ('Gorkha','Pokhara','Kathmandu');
```



```
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| Student_Id | Student_Name | Student_Stream  | Student_Age | Student_Address | Student_Phone | Student_Email        | Student_DOB | Guardian_Name | Guardian_Phone |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
| S101       | Bijay        | Programming     |          21 | Kathmandu       |     986135647 | bijayrisal1@gmail.co | 2056-04-12  | Sanjay Risal  |      974408766 |
| S102       | Binoy        | Computing       |          25 | Pokhara         |     984293600 | binoyrisal@gmail.com | 2052-04-16  | Binda Risal   |      984429300 |
| S105       | Riyaa        | Hacking         |          20 | Gorkha          |     984295670 | Riyaa@gmail.com      | 2057-06-23  | Bikram        |      984435832 |
| S108       | Puspa        | Networking      |          19 | Kathmandu       |     987493680 | puspa@gmail.com      | 2052-04-16  | Gita          |      984652600 |
| S201       | Sarala       | Web development |          19 | Gorkha          |     984456786 | sarala@gmail.com     | 2058-11-28  | Sanima        |      974098344 |
+------------+--------------+-----------------+-------------+-----------------+---------------+----------------------+-------------+---------------+----------------+
```

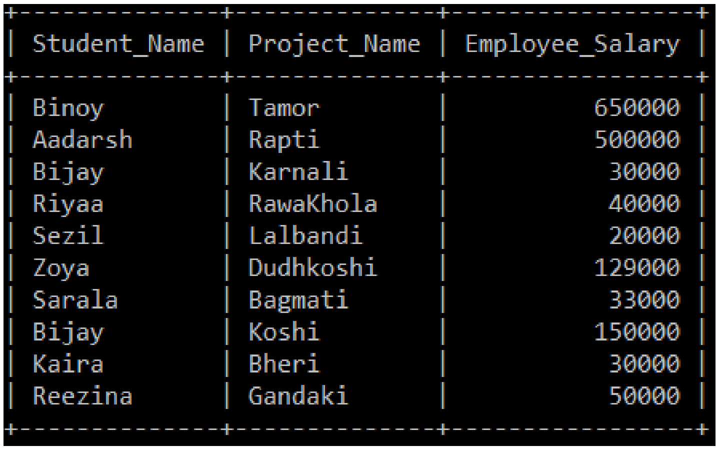
D6) Identify the total salaries paid to each students from each project.

SQL: select Student.Student_Name,Project.Project_Name,Employee.Employee_Salary

from Employee

inner join Student on Student.Student_Id=Employee.Student_Id

inner join Project on Project.Project_Id=Employee.Project_Id;



```
+--------------+--------------+-----------------+
| Student_Name | Project_Name | Employee_Salary |
+--------------+--------------+-----------------+
| Binoy        | Tamor        |          650000 |
| Aadarsh      | Rapti        |          500000 |
| Bijay        | Karnali      |           30000 |
| Riyaa        | RawaKhola    |           40000 |
| Sezil        | Lalbandi     |           20000 |
| Zoya         | Dudhkoshi    |          129000 |
| Sarala       | Bagmati      |           33000 |
| Bijay        | Koshi        |          150000 |
| Kaira        | Bheri        |           30000 |
| Reezina      | Gandaki      |           50000 |
+--------------+--------------+-----------------+
```

Conclusion

This project engages students in understanding database and most SQL queries. The report contents successfully accomplishes that achievement by allowing students to work with Xampp, and testing and learning SQL queries, which in turn helps them to teach database management and understand it further. This coursework has helped students to become more familiar with the concepts of database management and its entities, attributes, ER diagram and queries. Syntax and functions to create, update and insert attributes were also a new experience. Xampp is a complex but interesting application which kept the coursework less tedious to do.

The coursework was an interesting experience that helped me learn the ways of database management systems and how to operate them. Problems that were encountered throughout the coursework and its progress but they were quickly solved by the help of the module leader and the internet.

Bibliography

A MySQL Strategy Whitepaper. (2011). *Top 10 Reasons to choose MySQL for Web-based Applications.* US: oracle.

Kumar, R. (2009). *MySQL features.* Delhi: OSS Cube.

MySQL . (2019). *MySQL Server FAQ*. Retrieved 05 15, 2019, from MySQL: https://www.mysql.com/industry/faq/

MySQL. (2019). *Case Studies*. Retrieved 05 15, 2019, from MySQL: https://www.mysql.com/why-mysql/case-studies/?main=0&topic=12&type=5&lang=en

Rouse, M. (2018, 06). *MySQL*. Retrieved 05 13, 2019, from tech target: https://searchoracle.techtarget.com/definition/MySQL?fbclid=IwAR0mRYTXi5nkqq1lSJpagZxM7f4QFNPLfcktMIOVZNWYosNrs5x7fTLwqOw

Search It Channel. (2019). *What are the top MySQL features?* Retrieved 05 15, 2019, from techtarget: https://searchitchannel.techtarget.com/feature/What-are-the-top-MySQL-features-What-is-MySQL

SINGH, C. (2015, 05). *Normalization in DBMS:*. Retrieved 05 15, 2019, from Beginnners Book: https://www.google.com/search?q=Normalization+in+database&rlz=1C1CHBF_enNP833NP833&oq=No&aqs=chrome.0.69i59j69i57j69i60l4.1445j0j1&sourceid=chrome&ie=UTF-8

Techopedia . (2017). *First Normal Form(1NF)*. Retrieved 05 18, 2109, from techopedia: https://www.techopedia.com/definition/25955/first-normal-form-1nf

Whitehorn, M. (2007, 03 18). *MySQL - is this database fit for the Enterprise*. Retrieved 05 15, 2019, from The Register: https://www.theregister.co.uk/2007/03/18/mysql_enterprise_fit/