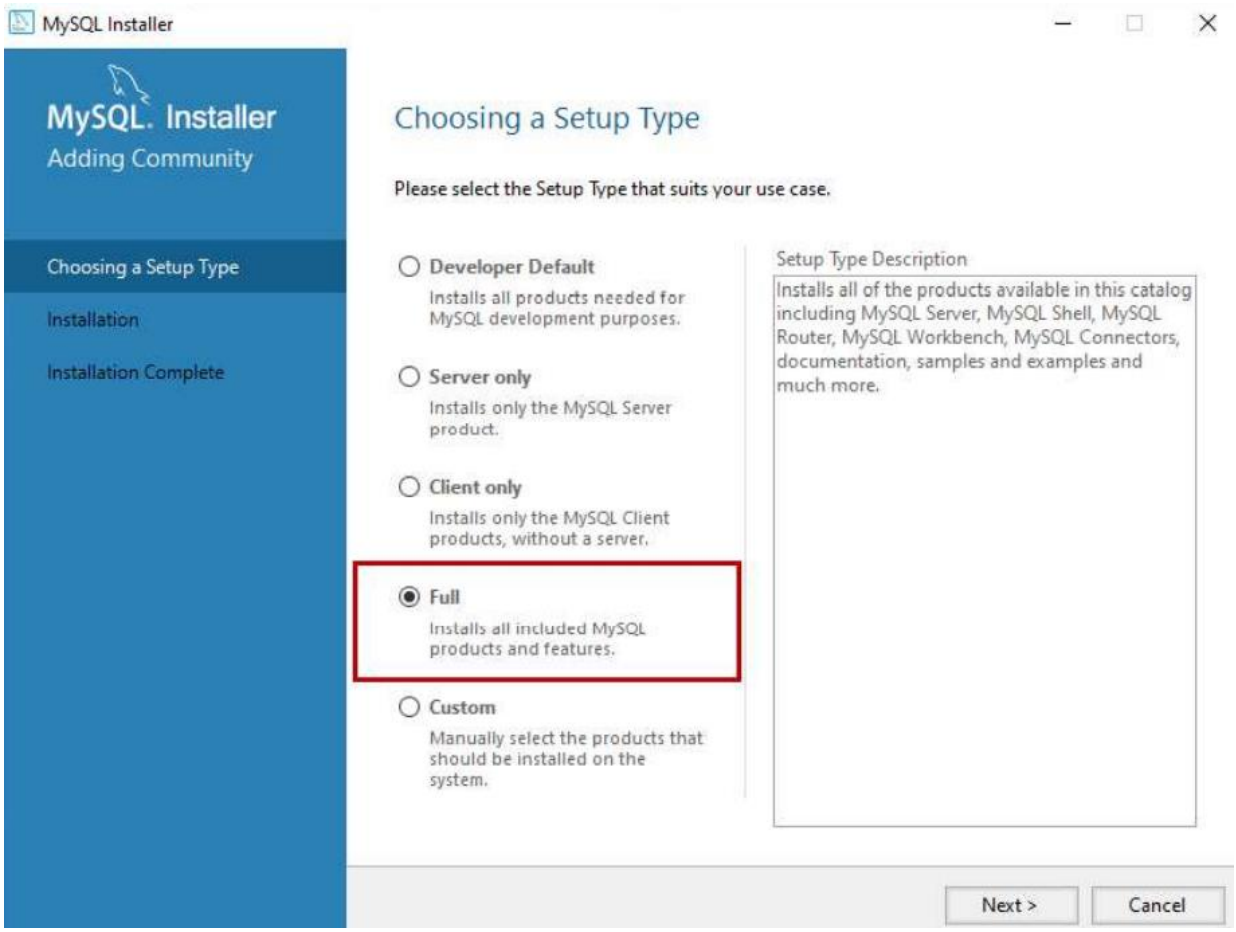
Name: Kele Waidehee

Roll no.: 21

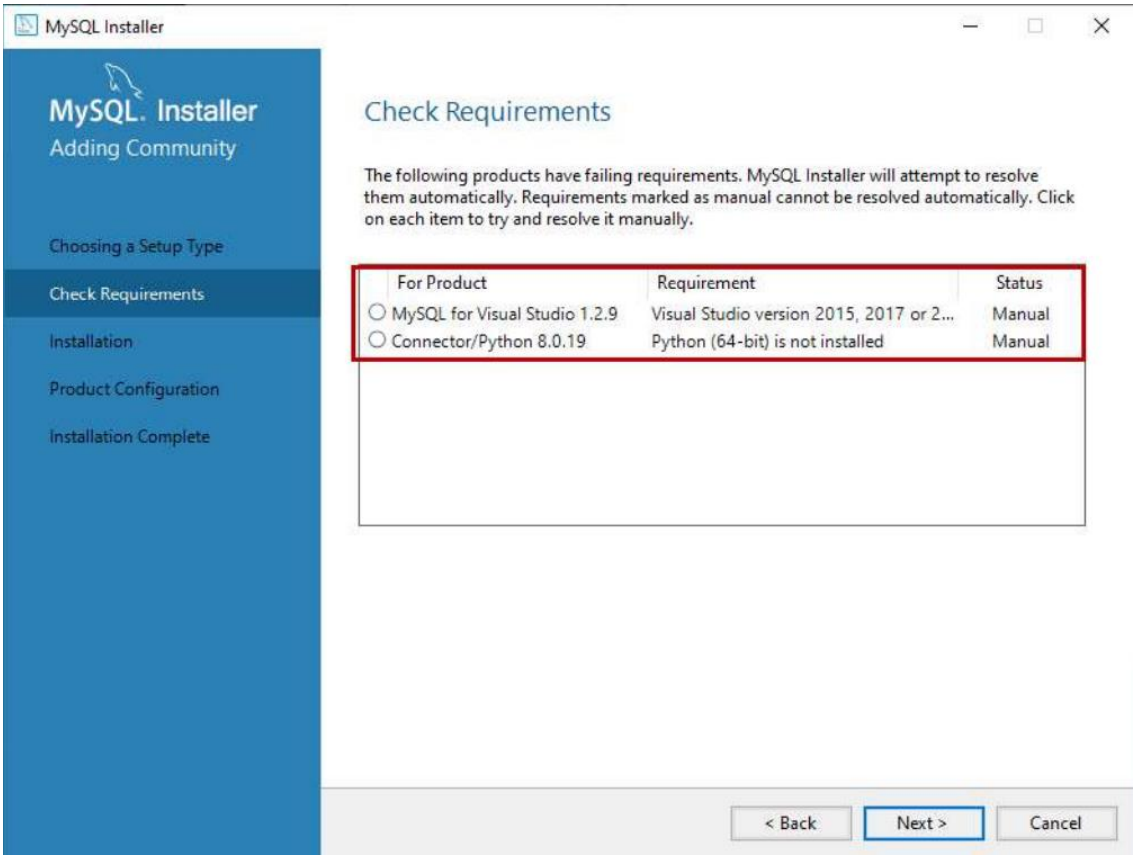
Batch: T2

Practical 1: Download and install MySQL database server

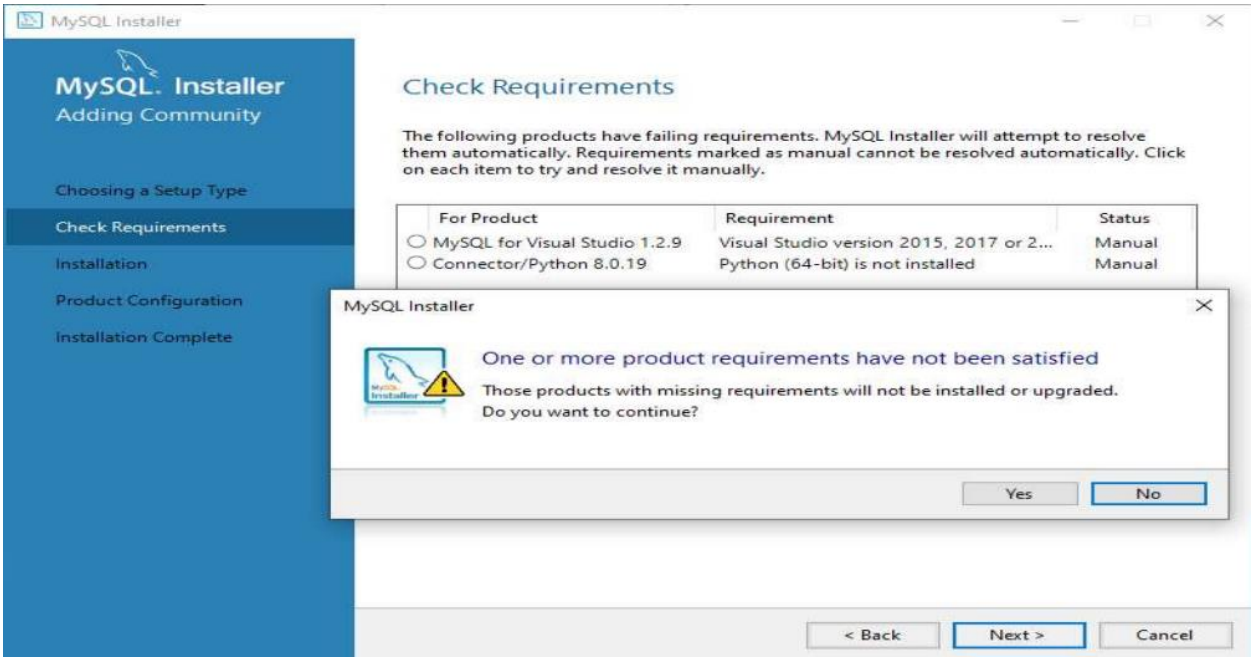
1. Choosing a Setup Type



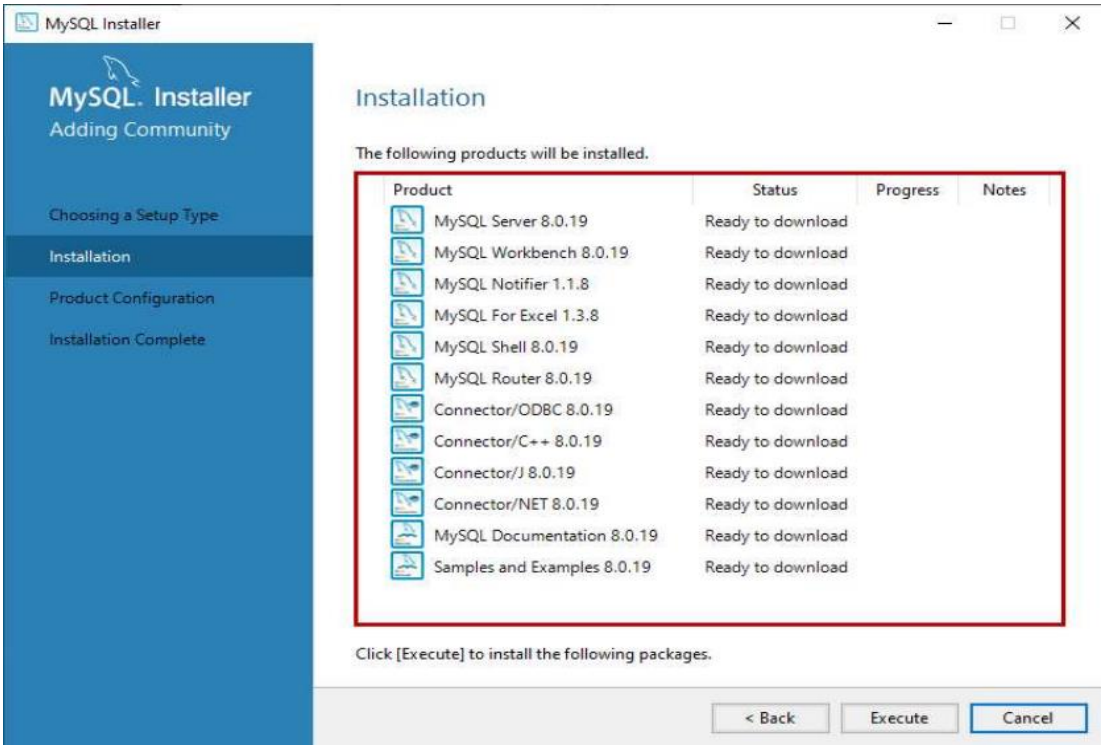
2. Check Requirements

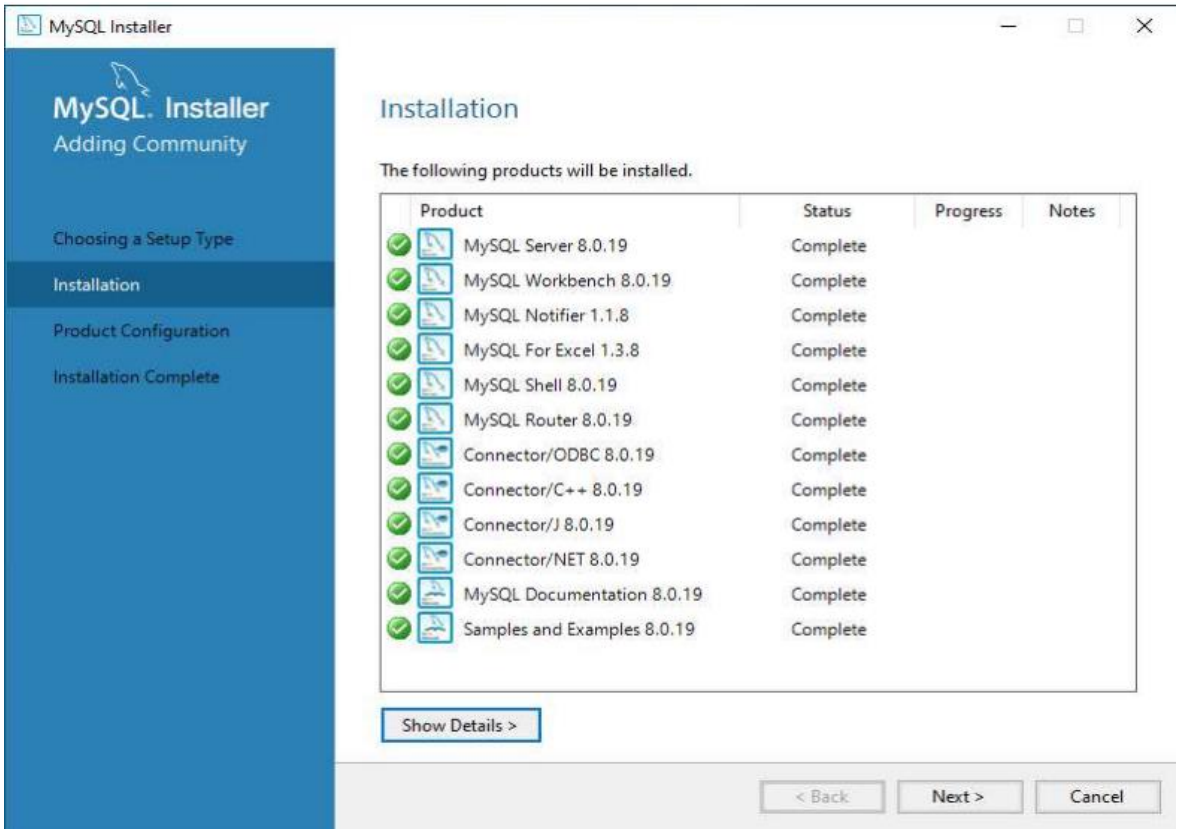


3. Warning

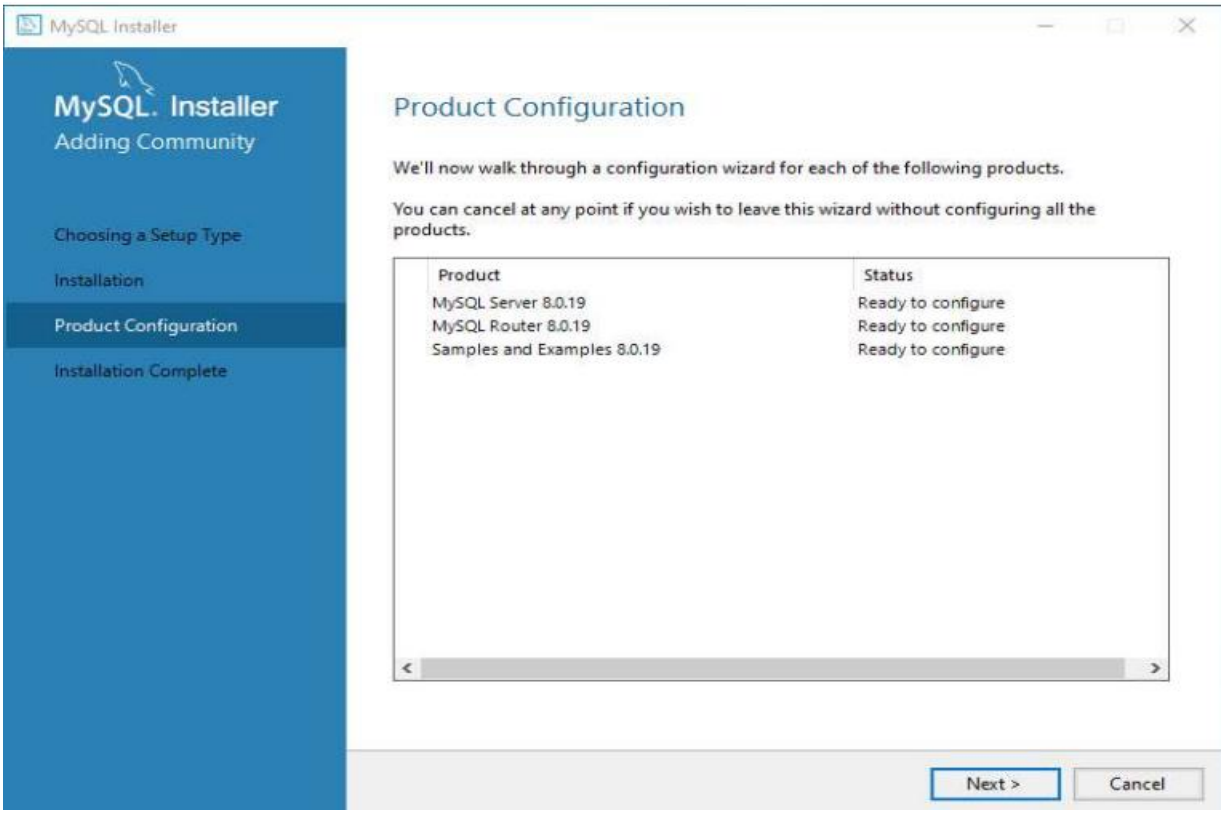


4. Installation

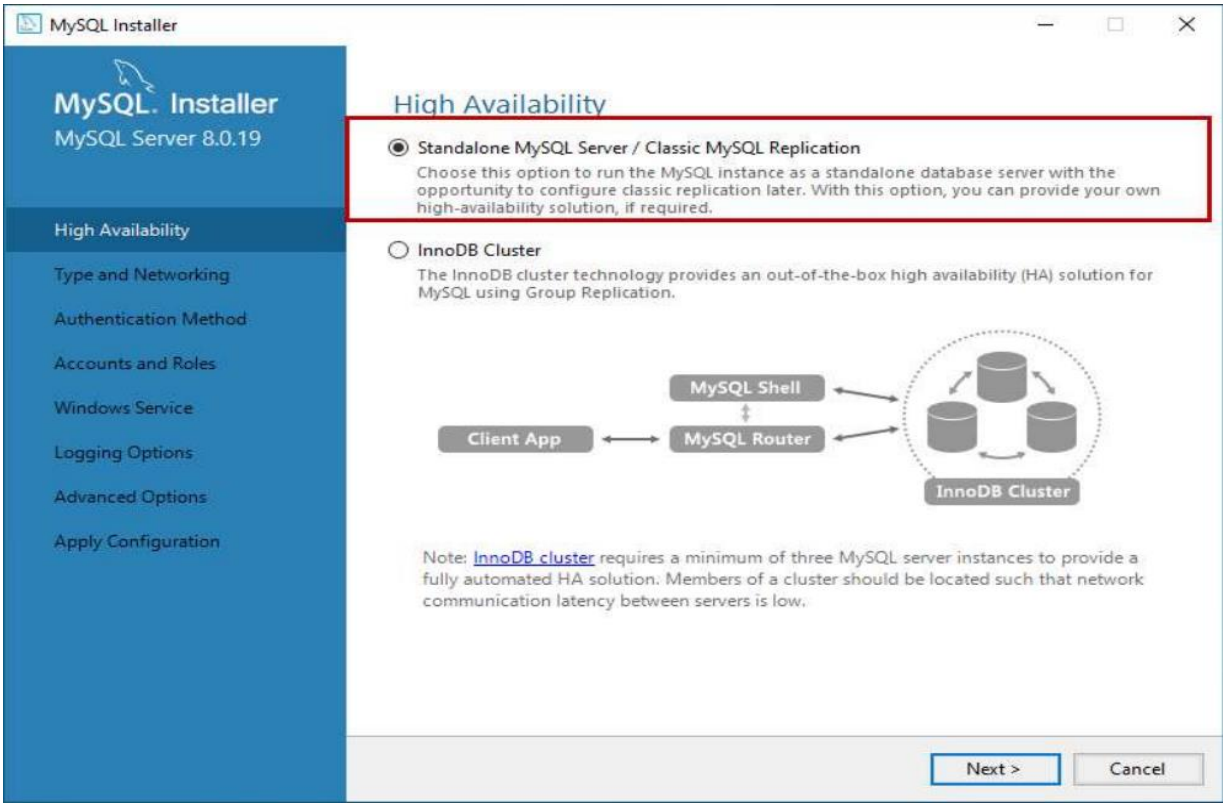




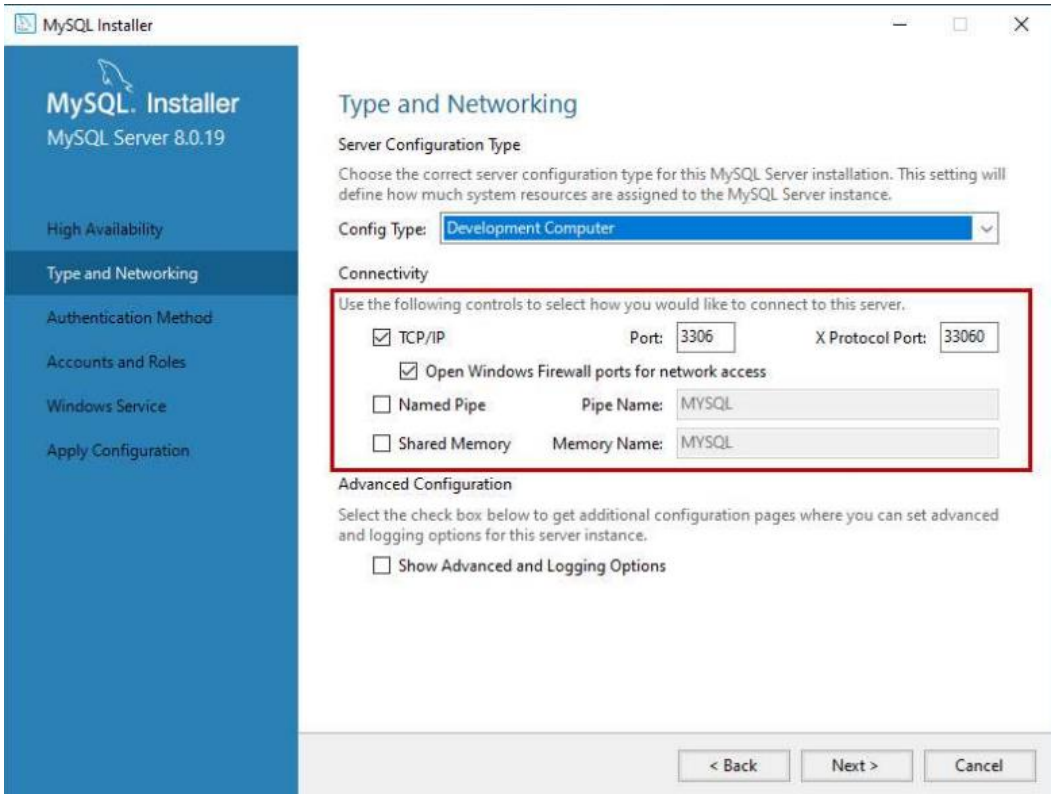
5. Product Configuration



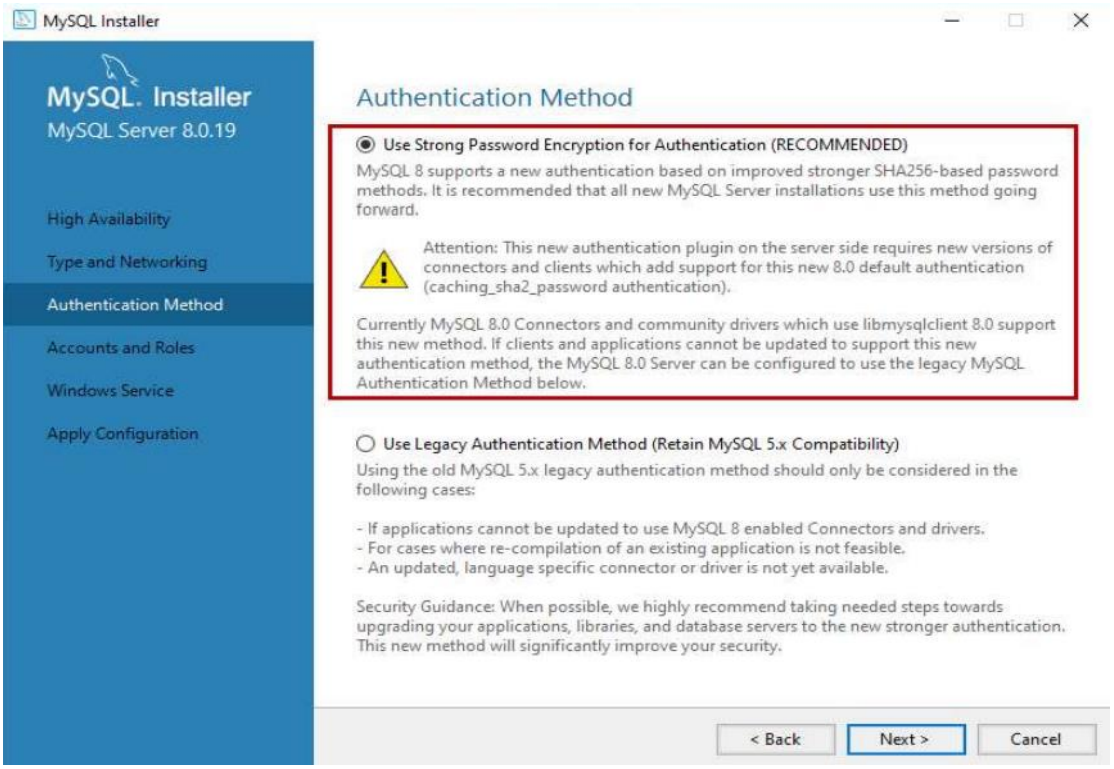
6. High Availability



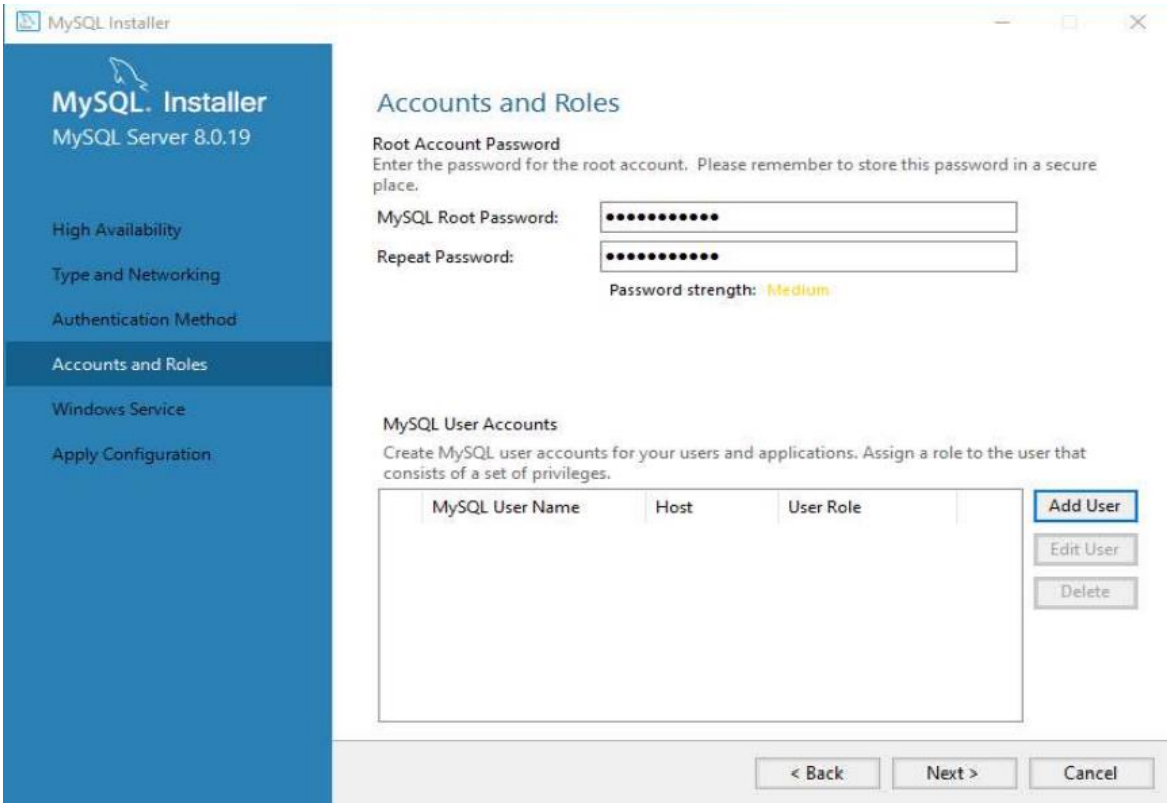
7. Type and Networking



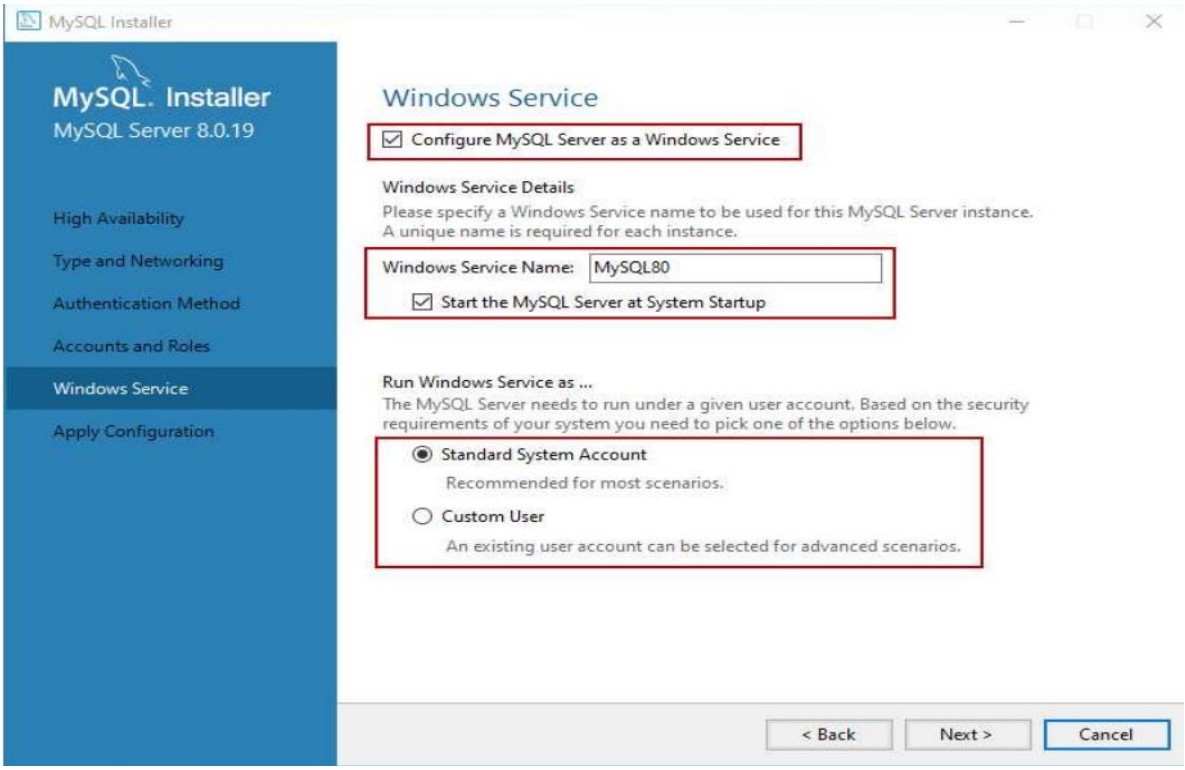
8. Authentication Method



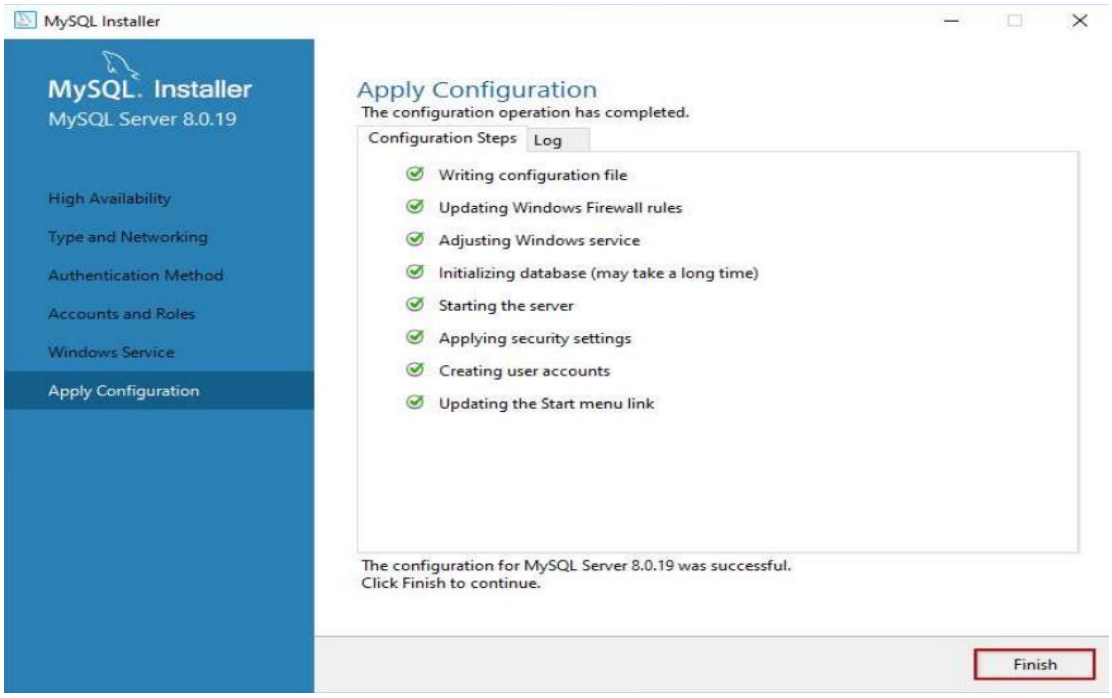
9. Accounts and Roles



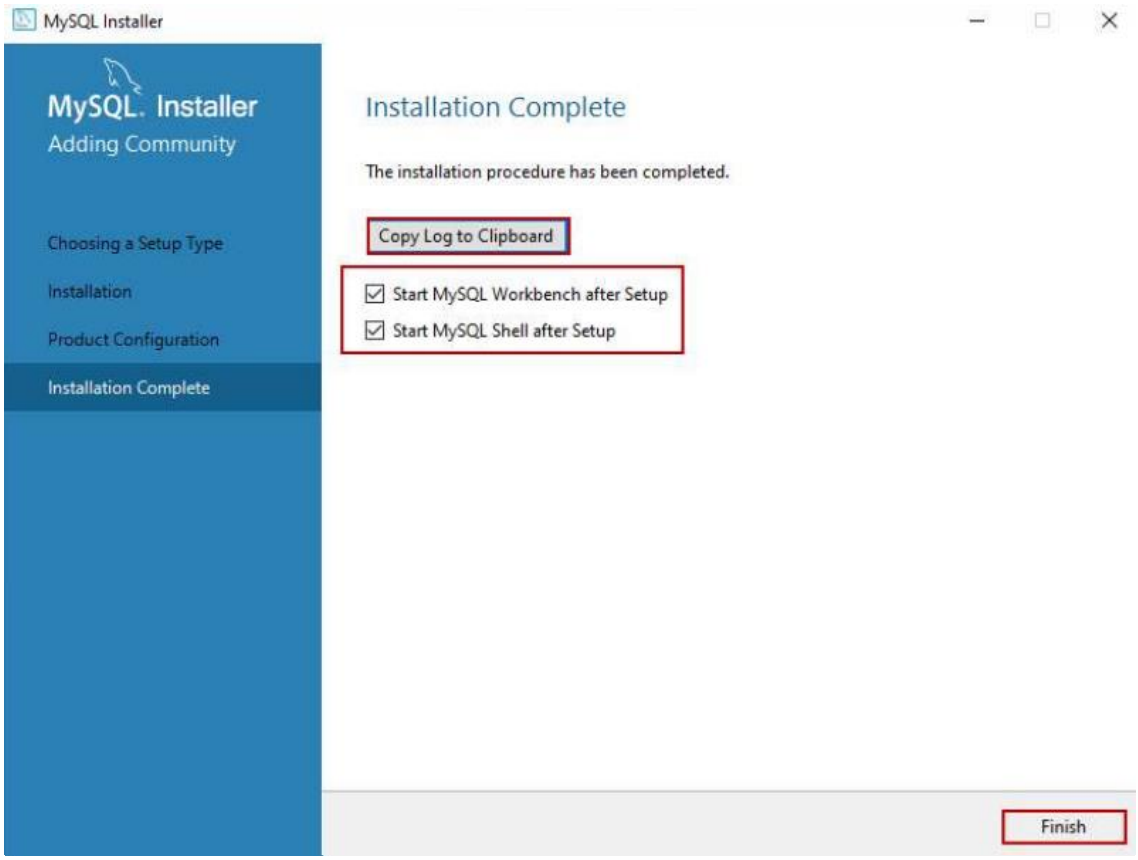
10. Windows Service



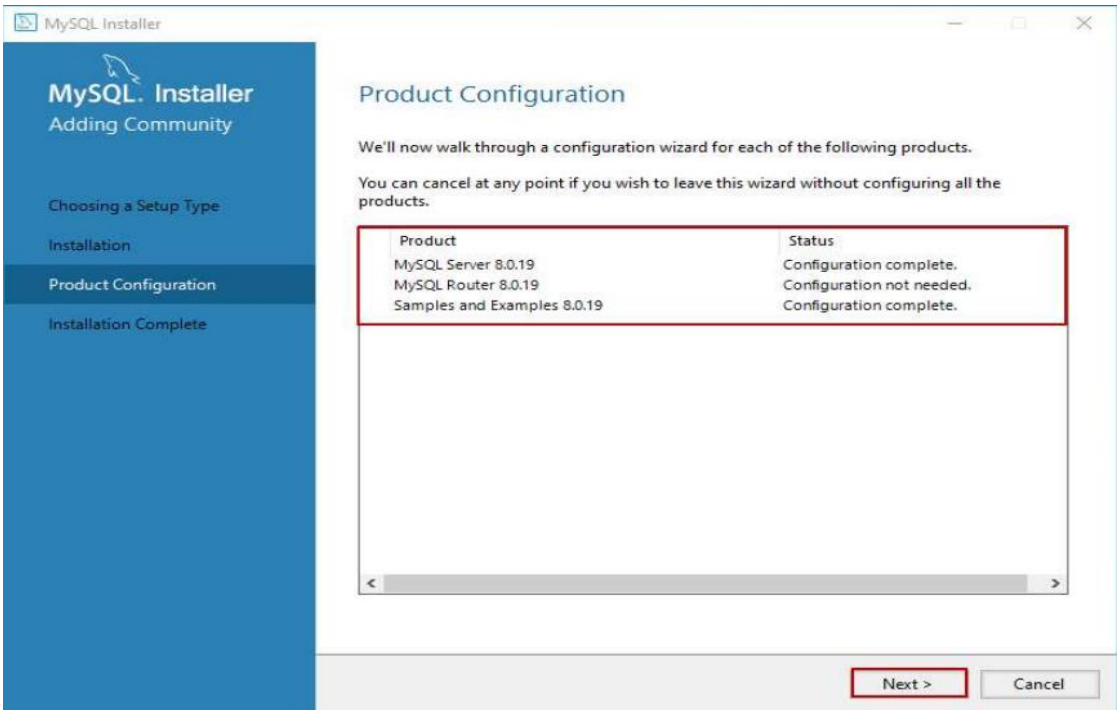
11. Apply Configuration



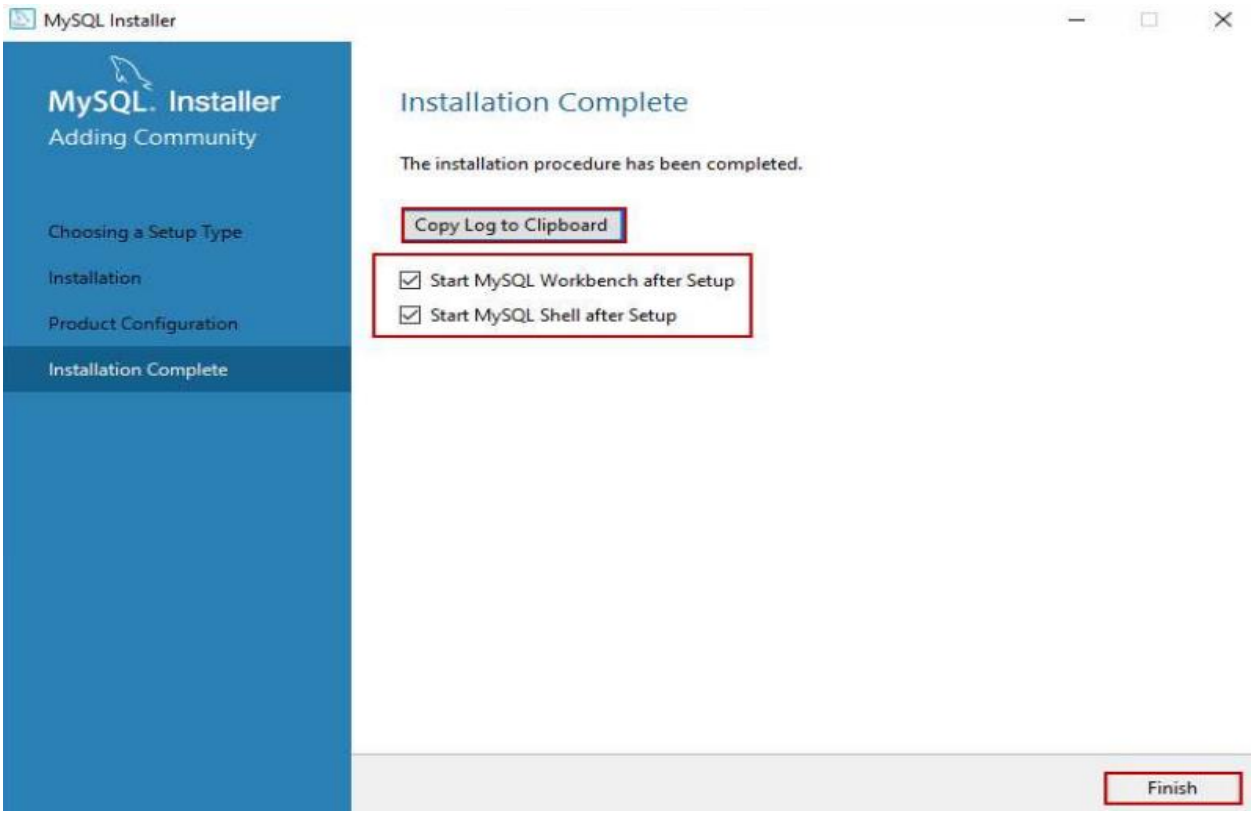
a. Connect to server



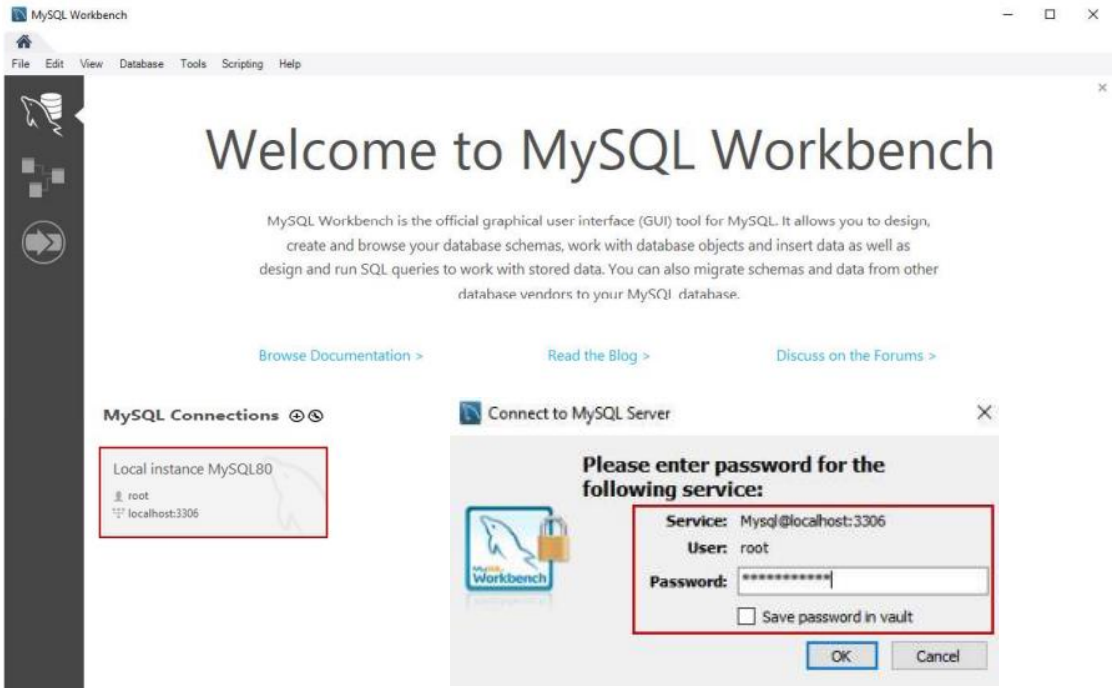
b. Product Configuration

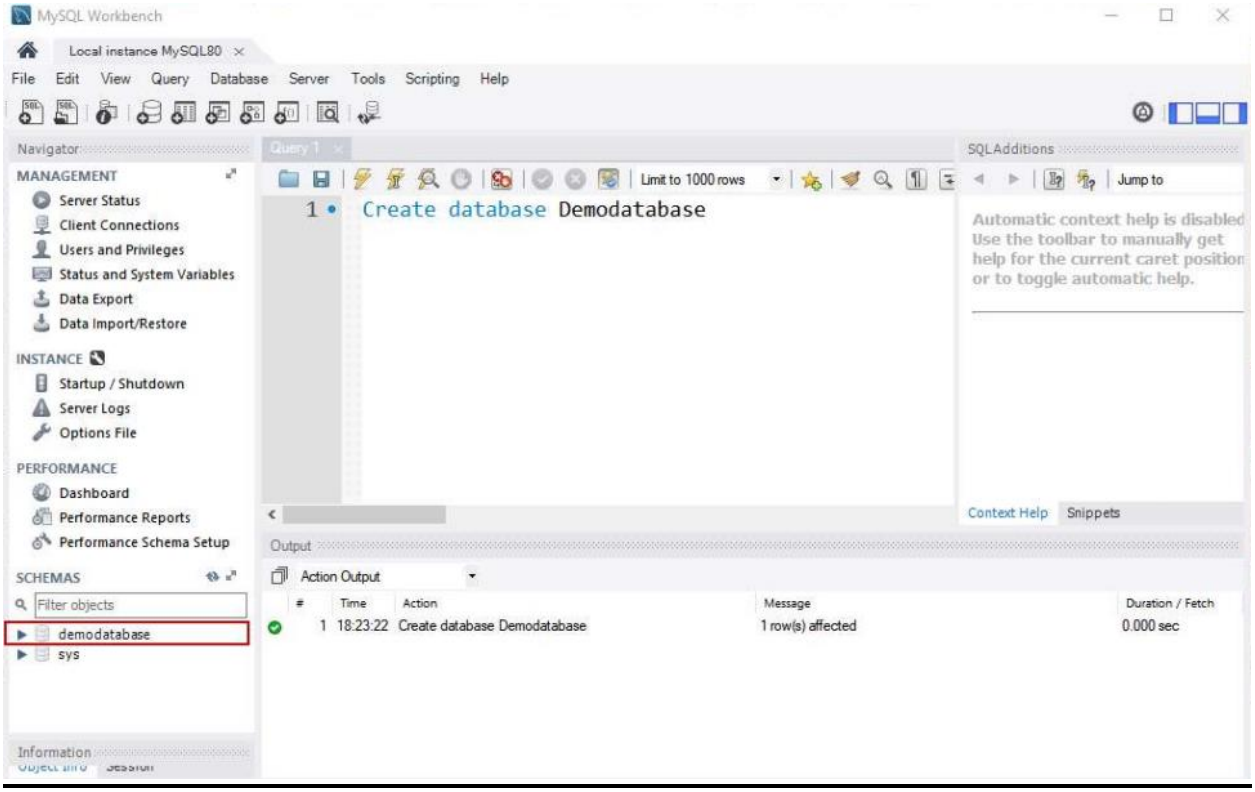


12. Installation Complete



- Connect to MySql Server:





Name: Kele Waidehee

Roll no.: 21

Batch: T2

Practical **9 :** Normalization in DBMS

Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly

Anomalies in DBMS

There are three types of **anomalies that occur when the database is not normalized**. These are: Insertion, update and deletion anomaly. Let's take an example to understand this.

Example: A manufacturing company stores the employee details in a table Employee that has four attributes: Emp_Id for storing employee's id, Emp_Name for storing employee's name, Emp_Address for storing employee's address and Emp_Dept for storing the department details in which the employee works. At some point of time the table looks like this:

Emp_Id	Emp_Name	Emp_Address	Emp_Dept
101	Rick	Delhi	D001
101	Rick	Delhi	D002
123	Maggie	Agra	D890
166	Glenn	Chennai	D900
166	Glenn	Chennai	D004

This table is not normalized. We will see the problems that we face when a table in database is not normalized.

Update anomaly: In the above table we have two rows for employee Rick as he belongs to two departments of the company. If we want to update the address of Rick then we have to update the same in two rows or the data will become inconsistent. If somehow, the correct address gets updated in one department but not in other then as per the database, Rick would be having two different addresses, which is not correct and would lead to inconsistent data.

Insert anomaly: Suppose a new employee joins the company, who is under training and currently not assigned to any department then we would not be able to insert the data into the table if Emp_Dept field doesn't allow null.

Delete anomaly: Let's say in future, company closes the department D890 then deleting the rows that are having Emp_Dept as D890 would also delete the information of employee Maggie since she is assigned only to this department.

To overcome these anomalies we need to normalize the data. In the next section we will discuss about normalization.

Normalization

Here are the most commonly used normal forms:

- First normal form(1NF)
- Second normal form(2NF)
- Third normal form(3NF)
- Boyce & Codd normal form (BCNF)

First normal form (1NF)

A relation is said to be in **1NF (first normal form)**, if it doesn't contain any multi-valued attribute. In other words you can say that a relation is in 1NF if each attribute contains only atomic(single) value only.

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.

Example: Let's say a company wants to store the names and contact details of its employees. It creates a table in the database that looks like this:

Emp_Id	Emp_Name	Emp_Address	Emp_Mobile
101	Herschel	New Delhi	8912312390
102	Jon	Kanpur	8812121212 , 9900012222
103	Ron	Chennai	7778881212
104	Lester	Bangalore	9990000123, 8123450987

Two employees (`Jon` & `Lester`) have two mobile numbers that caused the `Emp_Mobile` field to have multiple values for these two employees.

This table is **not in 1NF** as the rule says “each attribute of a table must have atomic (single) values”, the Emp_Mobile values for employees Jon & Lester violates that rule.

To make the table complies with 1NF we need to create separate rows for the each mobile number in such a way so that none of the attributes contains multiple values.

Emp_Id	Emp_Name	Emp_Address	Emp_Mobile
101	Herschel	New Delhi	8912312390
102	Jon	Kanpur	8812121212
102	Jon	Kanpur	9900012222
103	Ron	Chennai	7778881212
104	Lester	Bangalore	9990000123
104	Lester	Bangalore	8123450987

Second normal form (2NF)

A table is said to be in 2NF if both the following conditions hold:

- Table is in 1NF (First normal form)
- No non-prime attribute is dependent on the proper subset of any candidate key of table.

An attribute that is not part of any candidate key is known as non-prime attribute.

Example: Let's say a school wants to store the data of teachers and the subjects they teach. They create a table Teacher that looks like this: Since a teacher can teach more than one subjects, the table can have multiple rows for a same teacher.

Teacher_Id	Subject	Teacher_Age
111	Maths	38
111	Physics	38
222	Biology	38
333	Physics	40
333	Chemistry	40

Candidate Keys: {Teacher_Id, Subject}

Non prime attribute: Teacher_Age

This table is in 1 NF because each attribute has atomic values. However, it is not in 2NF because non prime attribute Teacher_Age is dependent on Teacher_Id alone which is a proper subset of candidate key. This violates the rule for 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 complies with 2NF we can disintegrate it in two tables like this:

0

Teacher_Id	Teacher_Age
111	38
222	38
333	40

Teacher_Subject table:

Teacher_Id	Subject
111	Maths
111	Physics
222	Biology
333	Physics
333	Chemistry

Third Normal form (3NF)

A table design is said to be in 3NF if both the following conditions hold:

- Table must be in 2NF
- Transitive functional dependency of non-prime attribute on any super key should be removed.

An attribute that is not part of any candidate key is known as non-prime attribute.

In other words 3NF can be explained like this: A table is in 3NF if it is in 2NF and for each functional dependency X-> Y at least one of the following conditions hold:

- X is a super key of table
- Y is a prime attribute of table

An attribute that is a part of one of the candidate keys is known as prime attribute.

Example: Let's say a company wants to store the complete address of each employee, they create a table named Employee_Details that looks like this:

Emp_Id	Emp_Name	Emp_Zip	Emp_State	Emp_City	Emp_District
1001	John	282005	UP	Agra	Dayal Bagh
1002	Ajeet	222008	TN	Chennai	M-City
1006	Lora	282007	TN	Chennai	Urrapakkam
1101	Lilly	292008	UK	Pauri	Bhagwan
1201	Steve	222999	MP	Gwalior	Ratan

Super keys: {Emp_Id}, {Emp_Id, Emp_Name}, {Emp_Id, Emp_Name, Emp_Zip}…so on

Candidate Keys: {Emp_Id}

Non-prime attributes: all attributes except Emp_Id are non-prime as they are not part of any candidate keys.

Here, Emp_State, Emp_City & Emp_District dependent on Emp_Zip. Further Emp_zip is dependent on Emp_Id that makes non-prime attributes (Emp_State, Emp_City & Emp_District) transitively dependent on super key (Emp_Id). This violates the rule of 3NF.

To make this table complies with 3NF we have to disintegrate the table into two tables to remove the transitive dependency:

Employee Table:

Emp_Id	Emp_Name	Emp_Zip
1001	John	282005
1002	Ajeet	222008
1006	Lora	282007
1101	Lilly	292008
1201	Steve	222999

Employee_Zip table:

Emp_Zip	Emp_State	Emp_City	Emp_District
282005	UP	Agra	Dayal Bagh
222008	TN	Chennai	M-City
282007	TN	Chennai	Urrapakkam
292008	UK	Pauri	Bhagwan
222999	MP	Gwalior	Ratan

Boyce Codd normal form (BCNF)

It is an advance version of 3NF that's why it is also referred as 3.5NF. BCNF is stricter than 3NF. A table complies with BCNF if it is in 3NF and for every functional dependency X->Y, X should be the super key of the table.

Example: Suppose there is a company wherein employees work in more than one department. They store the data like this:

Emp_Id	Emp_Nationality	Emp_Dept	Dept_Type	Dept_No_Of_Emp
1001	Austrian	Production and planning	D001	200
1001	Austrian	stores	D001	250
1002	American	design and technical support	D134	100
1002	American	Purchasing department	D134	600

Functional dependencies in the table above:

Emp_Id -> Emp_Nationality

Emp_Dept -> {Dept_Type, Dept_No_Of_Emp}

Candidate key: {Emp_Id, Emp_Dept}

The table is not in BCNF as neither Emp_Id nor Emp_Dept alone are keys.

To make the table comply with BCNF we can break the table in three tables like this:

Emp_Nationality table:

Emp_Id	Emp_Nationality
1001	Austrian
1002	American

Emp_Dept table:

Emp_Dept	Dept_Type	Dept_No_Of_Emp
Production and planning	D001	200
stores	D001	250
design and technical support	D134	100
Purchasing department	D134	600

Emp_Dept_Mapping table:

Emp_Id	Emp_Dept
1001	Production and planning
1001	stores
1002	design and technical support
1002	Purchasing department

Functional dependencies:

`Emp_Id` -> `Emp_Nationality`

`Emp_Dept` -> {`Dept_Type`, `Dept_No_Of_Emp`}

Candidate keys:

For first table: `Emp_Id`

For second table: `Emp_Dept`

For third table: {`Emp_Id`, `Emp_Dept`}

This table is now in BCNF as in both the functional dependencies left side part is a key.

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 2: Using SQL prompt create database and use of SQL commands (DDL, DML and DCL).

- **DDL : Data Definition Language**

1. Create Table :

```
mysql> create table Emp(Emp_Id int(10),Emp_Name varchar(20),Salary int(10),City varchar(20));
Query OK, 0 rows affected, 2 warnings (0.03 sec)
```

2. Alter

   a) ADD b) Modify

```
mysql> alter table Emp ADD(Mobile_No int(10));
Query OK, 0 rows affected, 1 warning (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> alter table Emp modify(Emp_Id int(50));
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for
the right syntax to use near '(Emp_Id int(50))' at line 1
mysql> alter table Emp modify Emp_Id int(50);
Query OK, 0 rows affected, 1 warning (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> desc Emp;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Emp_Id    | int         | YES  |     | NULL    |       |
| Emp_Name  | varchar(20) | YES  |     | NULL    |       |
| Salary    | int         | YES  |     | NULL    |       |
| City      | varchar(20) | YES  |     | NULL    |       |
| Mobile_No | int         | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.02 sec)
```

3. Drop

```
mysql> Drop table Emp;
Query OK, 0 rows affected (0.03 sec)
```

4. Rename

```
mysql> rename table dept to Department;
Query OK, 0 rows affected (0.07 sec)
```

- **DML : Data Manipulation Language**

1. Insert

```
mysql> insert into Emp values(101,'Samruddhi',700000,'Dhule');
Query OK, 1 row affected (0.02 sec)
```

2.Select

3.Update

```
mysql> select * from Emp;
+--------+-----------+--------+---------+
| Emp_Id | Emp_Name  | Salary | City    |
+--------+-----------+--------+---------+
|    101 | Samruddhi | 700000 | Dhule   |
|    102 | Janhavi   | 800000 | Pune    |
|    103 | Purti     | 900000 | Jalgoan |
|    104 | Waidehee  | 500000 | Nashik  |
|    105 | Jayeshri  | 400000 | Mumbai  |
+--------+-----------+--------+---------+
5 rows in set (0.00 sec)

mysql> update Emp set City ='Delhi';
Query OK, 5 rows affected (0.01 sec)
Rows matched: 5  Changed: 5  Warnings: 0

mysql> select * from Emp;
+--------+-----------+--------+-------+
| Emp_Id | Emp_Name  | Salary | City  |
+--------+-----------+--------+-------+
|    101 | Samruddhi | 700000 | Delhi |
|    102 | Janhavi   | 800000 | Delhi |
|    103 | Purti     | 900000 | Delhi |
|    104 | Waidehee  | 500000 | Delhi |
|    105 | Jayeshri  | 400000 | Delhi |
+--------+-----------+--------+-------+
5 rows in set (0.00 sec)
```

4.Delete

```
mysql> select * from Emp;
+--------+-----------+--------+-------+
| Emp_Id | Emp_Name  | Salary | City  |
+--------+-----------+--------+-------+
|    101 | Samruddhi | 700000 | Delhi |
|    102 | Janhavi   | 800000 | Delhi |
|    103 | Purti     | 900000 | Delhi |
|    104 | Waidehee  | 500000 | Delhi |
|    105 | Jayeshri  | 400000 | Delhi |
+--------+-----------+--------+-------+
5 rows in set (0.00 sec)

mysql> delete from Emp where Emp_Id = 105;
Query OK, 1 row affected (0.02 sec)

mysql> select * from Emp;
+--------+-----------+--------+-------+
| Emp_Id | Emp_Name  | Salary | City  |
+--------+-----------+--------+-------+
|    101 | Samruddhi | 700000 | Delhi |
|    102 | Janhavi   | 800000 | Delhi |
|    103 | Purti     | 900000 | Delhi |
|    104 | Waidehee  | 500000 | Delhi |
+--------+-----------+--------+-------+
4 rows in set (0.00 sec)
```

- **DCL : Data Control Language**

1) Grant :

```
mysql> create user 'Samruddhi'@'127.0.0.1' identified by 'Test@12';
Query OK, 0 rows affected (0.04 sec)

mysql> grant select on Emp to 'Samruddhi@127.0.0.1';
ERROR 1410 (42000): You are not allowed to create a user with GRANT
mysql> grant select,insert,update,delete on Emp to Samruddhi@127.0.0.1;
Query OK, 0 rows affected (0.01 sec)

mysql> grant all on Emp to Samruddhi@127.0.0.1;
Query OK, 0 rows affected (0.01 sec)
```

2) Revoke:

```
mysql> revoke all on Emp from Samruddhi@127.0.0.1;
Query OK, 0 rows affected (0.01 sec)
```

3) Drop User :

```
mysql> drop user Samruddhi@127.0.0.1;
Query OK, 0 rows affected (0.02 sec)
```

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 3: Create tables for suitable database schema with constraint like primary key, foreign key and NOT Null. Then design at least ten SQL queries SQL DML statements: Insert, Select, Update, Delete using distinct and count clause.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| pract              |
| sakila             |
| sys                |
| world              |
+--------------------+
7 rows in set (0.04 sec)

mysql> use pract;
Database changed
mysql> show tables;
```

Required Tables :

```
mysql> show tables;
+-----------------+
| Tables_in_pract |
+-----------------+
| dept            |
| dept_location   |
| employee        |
| project         |
| works_on        |
+-----------------+
5 rows in set (0.02 sec)

mysql> select * from dept;
+---------+------------+--------+----------------+
| Dept_No | Dept_Name  | Mgr_ID | Mgr_Start_Date |
+---------+------------+--------+----------------+
|     100 | Computer   |    111 | 2010-06-23     |
|     101 | Civil      |    222 | 2011-08-17     |
|     102 | Mechanical |    333 | 2013-09-13     |
|     103 | Electrical |    444 | 2012-10-29     |
|     104 | IT         |    555 | 2014-07-15     |
|     105 | Metallurgy |      0 | 2015-10-27     |
+---------+------------+--------+----------------+
6 rows in set (0.01 sec)

mysql> select * from dept_location;
+---------+----------+
| Dept_No | Dept_Loc |
+---------+----------+
|     100 | Pune     |
|     101 | Mumbai   |
|     102 | Delhi    |
|     103 | Kolkata  |
|     104 | Surat    |
|     105 | Nashik   |
+---------+----------+
```

```
6 rows in set (0.00 sec)

mysql> select * from employee;
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
| Emp_Id | F_Name   | L_Name   | DOB        | Join_Date  | Address | Gender | Dept_Id | Salary |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
|      0 | Madhvika | Jadhav   | 1988-02-22 | 2011-03-22 | Indore  | F      |     105 |  37000 |
|    111 | Pranil   | Patil    | 1985-04-08 | 2008-05-05 | Pune    | M      |     100 |  95000 |
|    222 | Mohan    | Sharma   | 1982-07-23 | 2009-08-11 | Delhi   | M      |     101 |  60000 |
|    333 | Raghuvir | Prasad   | 1983-08-11 | 2009-06-21 | Delhi   | M      |     102 |  65000 |
|    444 | Ram      | Kulkarni | 1979-09-11 | 2008-04-23 | Mumbai  | M      |     103 |  77000 |
|    555 | Rohit    | Jadhav   | 1987-03-10 | 2009-03-03 | Nashik  | M      |     104 |  55000 |
|    666 | Sakshi   | Patil    | 1989-07-21 | 2011-02-22 | Pune    | F      |     101 |  30000 |
|    777 | Samiksha | More     | 1987-01-17 | 2012-04-21 | Surat   | F      |     102 |  55000 |
|    888 | Arti     | Vaidya   | 1985-07-22 | 2011-07-22 | Noida   | F      |     103 |  56000 |
|    999 | Sam      | Wadekar  | 1985-09-21 | 2012-12-17 | Pune    | F      |     104 |  83000 |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
10 rows in set (0.00 sec)

mysql> select * from project;
+-------------+--------+--------------+----------+
| Pro_Name    | Pro_No | Pro_Location | Dept_Num |
+-------------+--------+--------------+----------+
| CodePro     |   4141 | Pune         |      100 |
| NH47        |   4142 | Mumbai       |      101 |
| HAL         |   4143 | Nagpur       |      102 |
| BHEL        |   4144 | Nashik       |      103 |
| CodeChef    |   4145 | Surat        |      104 |
| CodingNinja |   4146 | Jalgaon      |      105 |
+-------------+--------+--------------+----------+
6 rows in set (0.00 sec)

mysql> select * from works_on;
+----------+-----------+---------+----------+-----+
| W_Emp_ID | W_Pro_Num | Dept_No | Relation | Hrs |
+----------+-----------+---------+----------+-----+
|      111 |      4141 |     100 | Manager  | 310 |
|      222 |      4142 |     101 | Manager  | 320 |
|      333 |      4143 |     102 | Manager  | 330 |
|      444 |      4144 |     103 | Jr.Engg  | 340 |
|      555 |      4145 |     104 | Sr.Engg  | 350 |
|      666 |      4146 |     105 | Sr.Engg  | 360 |
|      777 |      4141 |     101 | Clerk    | 370 |
|      888 |      4143 |     103 | Clerk    | 380 |
|      999 |      4142 |     102 | Jr.Engg  | 390 |
|        0 |      4144 |     105 | Sr.Engg  | 400 |
+----------+-----------+---------+----------+-----+
10 rows in set (0.00 sec)
```

- **Queries**

1. List the employee who join the before 2012.

```
mysql> select F_Name,L_Name from Employee where Join_Date < '2012-01-01';
+----------+----------+
| F_Name   | L_Name   |
+----------+----------+
| Madhvika | Jadhav   |
| Pranil   | Patil    |
| Mohan    | Sharma   |
| Raghuvir | Prasad   |
| Ram      | Kulkarni |
| Rohit    | Jadhav   |
| Sakshi   | Patil    |
| Arti     | Vaidya   |
+----------+----------+
8 rows in set (0.02 sec)
```

2. List the emps in the ascending order of their Salaries?

```
mysql> select Emp_Id,F_Name,L_Name,Salary
    -> From Employee order by Salary asc;
+--------+----------+----------+--------+
| Emp_Id | F_Name   | L_Name   | Salary |
+--------+----------+----------+--------+
|    666 | Sakshi   | Patil    |  30000 |
|      0 | Madhvika | Jadhav   |  37000 |
|    555 | Rohit    | Jadhav   |  55000 |
|    777 | Samiksha | More     |  55000 |
|    888 | Arti     | Vaidya   |  56000 |
|    222 | Mohan    | Sharma   |  60000 |
|    333 | Raghuvir | Prasad   |  65000 |
|    444 | Ram      | Kulkarni |  77000 |
|    999 | Sam      | Wadekar  |  83000 |
|    111 | Pranil   | Patil    |  95000 |
+--------+----------+----------+--------+
10 rows in set (0.00 sec)
```

3. List the all emps in the asc order of Annual salary.

```
mysql> select Emp_Id,F_Name,L_Name,Salary*12 as Annual_Sal
    -> from Employee
    -> order by Salary*12 asc;
+--------+----------+----------+------------+
| Emp_Id | F_Name   | L_Name   | Annual_Sal |
+--------+----------+----------+------------+
|    666 | Sakshi   | Patil    |     360000 |
|      0 | Madhvika | Jadhav   |     444000 |
|    555 | Rohit    | Jadhav   |     660000 |
|    777 | Samiksha | More     |     660000 |
|    888 | Arti     | Vaidya   |     672000 |
|    222 | Mohan    | Sharma   |     720000 |
|    333 | Raghuvir | Prasad   |     780000 |
|    444 | Ram      | Kulkarni |     924000 |
|    999 | Sam      | Wadekar  |     996000 |
|    111 | Pranil   | Patil    |    1140000 |
+--------+----------+----------+------------+
10 rows in set (0.02 sec)
```

4. Find the all details of employee who work for HAL project.

```
mysql> Select * from Employee Left Join project on employee.dept_id = project.dept_num where Project.pro_name = 'HAL';
+--------+----------+-------+------------+------------+---------+--------+---------+--------+----------+--------+-------------+----------+
| Emp_Id | F_Name   | L_Name | DOB        | Join_Date  | Address | Gender | Dept_Id | Salary | Pro_Name | Pro_No | Pro_Location | Dept_Num |
+--------+----------+-------+------------+------------+---------+--------+---------+--------+----------+--------+-------------+----------+
|    333 | Raghuvir | Prasad | 1983-08-11 | 2009-06-21 | Delhi   | M      |     102 |  65000 | HAL      |   4143 | Nagpur       |      102 |
|    777 | Samiksha | More   | 1987-01-17 | 2012-04-21 | Surat   | F      |     102 |  55000 | HAL      |   4143 | Nagpur       |      102 |
+--------+----------+-------+------------+------------+---------+--------+---------+--------+----------+--------+-------------+----------+
2 rows in set (0.03 sec)
```

5. List the emps Who's Annual salary ranging from 500000 and 800000.

```
mysql> select Emp_Id,F_Name,L_Name,Salary*12 as Annual_Sal
    -> from Employee where Salary*12 between 500000 and 700000;
+--------+----------+--------+------------+
| Emp_Id | F_Name   | L_Name | Annual_Sal |
+--------+----------+--------+------------+
|    555 | Rohit    | Jadhav |     660000 |
|    777 | Samiksha | More   |     660000 |
|    888 | Arti     | Vaidya |     672000 |
+--------+----------+--------+------------+
3 rows in set (0.02 sec)
```

6. a) List the emps who joined in January

```
mysql> select Emp_Id,F_Name,L_Name,Join_Date
    -> from Employee where  month(Join_Date) = 01;
Empty set (0.02 sec)
```

b) List the emps who joined in March

```
mysql> select Emp_Id,F_Name,L_Name,Join_Date
    -> from Employee where  month(Join_Date) = 03;
+--------+----------+--------+------------+
| Emp_Id | F_Name   | L_Name | Join_Date  |
+--------+----------+--------+------------+
|      0 | Madhvika | Jadhav | 2011-03-22 |
|    555 | Rohit    | Jadhav | 2009-03-03 |
+--------+----------+--------+------------+
2 rows in set (0.00 sec)
```

7. List the emps whose Sal is five-digit number and not starting with digit 3.

```
mysql> Select F_Name ,L_Name,Salary from Employee
    -> where length(Salary) = 5 and Salary not like '3%';
+----------+----------+--------+
| F_Name   | L_Name   | Salary |
+----------+----------+--------+
| Pranil   | Patil    |  95000 |
| Mohan    | Sharma   |  60000 |
| Raghuvir | Prasad   |  65000 |
| Ram      | Kulkarni |  77000 |
| Rohit    | Jadhav   |  55000 |
| Samiksha | More     |  55000 |
| Arti     | Vaidya   |  56000 |
| Sam      | Wadekar  |  83000 |
+----------+----------+--------+
8 rows in set (0.02 sec)
```

8. Find the project location of 4141 and 5151.

```
mysql> Select Pro_No ,Pro_Location From Project
    -> where Pro_No = 4141 or Pro_No = 4145;
+--------+--------------+
| Pro_No | Pro_Location |
+--------+--------------+
|   4141 | Pune         |
|   4145 | Surat        |
+--------+--------------+
2 rows in set (0.00 sec)
```

9. List the department, details where at least two emps are working.

```
mysql> Select Dept_Id,count(*) from Employee
    -> group by Dept_Id having count(*) > 1;
+---------+----------+
| Dept_Id | count(*) |
+---------+----------+
|     101 |        2 |
|     102 |        2 |
|     103 |        2 |
|     104 |        2 |
+---------+----------+
4 rows in set (0.02 sec)
```

10. Update the salary of employee 444

```
mysql> select Emp_Id,F_Name,Salary from Employee
    -> where Emp_Id = 444;
+--------+--------+--------+
| Emp_Id | F_Name | Salary |
+--------+--------+--------+
|    444 | Ram    |  77000 |
+--------+--------+--------+
1 row in set (0.00 sec)

mysql> update Employee Set Salary = 80000
    -> where Emp_Id = 444;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select Emp_Id,F_Name,Salary from Employee
    -> where Emp_Id = 444;
+--------+--------+--------+
| Emp_Id | F_Name | Salary |
+--------+--------+--------+
|    444 | Ram    |  80000 |
+--------+--------+--------+
1 row in set (0.00 sec)
```

11. Find the maximum salary of each department.

```
mysql> select Dept_Id,Dept_Name,max(Salary)
    -> from Employee
    -> join Department on Employee.Dept_Id = Department.Dept_No group by Dept_Id;
+---------+------------+-------------+
| Dept_Id | Dept_Name  | max(Salary) |
+---------+------------+-------------+
|     100 | Computer   |       95000 |
|     101 | Civil      |       60000 |
|     102 | Mechanical |       65000 |
|     103 | Electrical |       80000 |
|     104 | IT         |       83000 |
|     105 | Metallurgy |       37000 |
+---------+------------+-------------+
6 rows in set (0.03 sec)
```

12. List the First Name of employees F_names contains ‘A’.

```
mysql> select F_Name from Employee
    -> where F_name like '%a%';
+----------+
| F_Name   |
+----------+
| Madhvika |
| Pranil   |
| Mohan    |
| Raghuvir |
| Ram      |
| Sakshi   |
| Samiksha |
| Arti     |
| Sam      |
+----------+
9 rows in set (0.01 sec)
```

13. List the employe whose Emp_Id not starting with digit 3

```
mysql> select Emp_Id,F_Name,L_Name from Employee
    -> where Emp_Id not like '3%';
+--------+----------+----------+
| Emp_Id | F_Name   | L_Name   |
+--------+----------+----------+
|      0 | Madhvika | Jadhav   |
|    111 | Pranil   | Patil    |
|    222 | Mohan    | Sharma   |
|    444 | Ram      | Kulkarni |
|    555 | Rohit    | Jadhav   |
|    666 | Sakshi   | Patil    |
|    777 | Samiksha | More     |
|    888 | Arti     | Vaidya   |
|    999 | Sam      | Wadekar  |
+--------+----------+----------+
9 rows in set (0.00 sec)
```

14. Display unique Jobs from Works_on table.

```
mysql> Select distinct relation as jobs
    -> from works_on;
+---------+
| jobs    |
+---------+
| Manager |
| Jr.Engg |
| Sr.Engg |
| Clerk   |
+---------+
4 rows in set (0.01 sec)
```

15. Add new department 106 ‘Chemical’ in department table and then delete entry from department table.

```
mysql> insert into Department values(106,'Chemical',666,'2010-10-12');
Query OK, 1 row affected (0.02 sec)

mysql> Select * From Department;
+---------+------------+--------+----------------+
| Dept_No | Dept_Name  | Mgr_ID | Mgr_Start_Date |
+---------+------------+--------+----------------+
|     100 | Computer   |    111 | 2010-06-23     |
|     101 | Civil      |    222 | 2011-08-17     |
|     102 | Mechanical |    333 | 2013-09-13     |
|     103 | Electrical |    444 | 2012-10-29     |
|     104 | IT         |    555 | 2014-07-15     |
|     105 | Metallurgy |      0 | 2015-10-27     |
|     106 | Chemical   |    666 | 2010-10-12     |
+---------+------------+--------+----------------+
7 rows in set (0.00 sec)

mysql> delete from Department where Dept_No = 106;
Query OK, 1 row affected (0.02 sec)

mysql> Select * From Department;
+---------+------------+--------+----------------+
| Dept_No | Dept_Name  | Mgr_ID | Mgr_Start_Date |
+---------+------------+--------+----------------+
|     100 | Computer   |    111 | 2010-06-23     |
|     101 | Civil      |    222 | 2011-08-17     |
|     102 | Mechanical |    333 | 2013-09-13     |
|     103 | Electrical |    444 | 2012-10-29     |
|     104 | IT         |    555 | 2014-07-15     |
|     105 | Metallurgy |      0 | 2015-10-27     |
+---------+------------+--------+----------------+
6 rows in set (0.00 sec)
```

16. Check whether all the employee number are indeed unique.

```
mysql> select Emp_Id,count(Emp_Id)
    -> from Employee group by Emp_Id;
+--------+---------------+
| Emp_Id | count(Emp_Id) |
+--------+---------------+
|      0 |             1 |
|    111 |             1 |
|    222 |             1 |
|    333 |             1 |
|    444 |             1 |
|    555 |             1 |
|    666 |             1 |
|    777 |             1 |
|    888 |             1 |
|    999 |             1 |
+--------+---------------+
10 rows in set (0.00 sec)
```

17.Select the maximum average salary drawn for each dept.

```
mysql> select Dept_Id, avg(Salary) From Employee
    -> group by Dept_Id;
+---------+-------------+
| Dept_Id | avg(Salary) |
+---------+-------------+
|     100 |  95000.0000 |
|     101 |  45000.0000 |
|     102 |  60000.0000 |
|     103 |  68000.0000 |
|     104 |  69000.0000 |
|     105 |  37000.0000 |
+---------+-------------+
6 rows in set (0.00 sec)
```

18. List the unique jobs of dept 101 and 102 in desc order.

```
mysql> select Dept_No,Relation as Jobs
    -> from Works_On
    -> where Dept_No = 101 or Dept_No = 102
    -> order by relation desc;
+---------+---------+
| Dept_No | Jobs    |
+---------+---------+
|     101 | Manager |
|     102 | Manager |
|     102 | Jr.Engg |
|     101 | Clerk   |
+---------+---------+
4 rows in set (0.01 sec)
```

19. List the highest paid employee.

```
mysql> select Emp_Id,F_Name,L_Name,max(Salary) from Employee;
+--------+----------+--------+-------------+
| Emp_Id | F_Name   | L_Name | max(Salary) |
+--------+----------+--------+-------------+
|      0 | Madhvika | Jadhav |       95000 |
+--------+----------+--------+-------------+
1 row in set (0.00 sec)
```

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 4 : Consider suitable database schema and design five SQL Nested Queries with/without where clause.

Database Schema into consideration:

```
mysql> show tables;
+------------------+
| Tables_in_pract  |
+------------------+
| dept             |
| dept_location    |
| employee         |
| project          |
| works_on         |
+------------------+
5 rows in set (0.02 sec)

mysql> select * from dept;
+---------+-------------+---------+-----------------+
| Dept_No | Dept_Name   | Mgr_ID  | Mgr_Start_Date  |
+---------+-------------+---------+-----------------+
|     100 | Computer    |     111 | 2010-06-23      |
|     101 | Civil       |     222 | 2011-08-17      |
|     102 | Mechanical  |     333 | 2013-09-13      |
|     103 | Electrical  |     444 | 2012-10-29      |
|     104 | IT          |     555 | 2014-07-15      |
|     105 | Metallurgy  |       0 | 2015-10-27      |
+---------+-------------+---------+-----------------+
6 rows in set (0.01 sec)

mysql> select * from dept_location;
+---------+-----------+
| Dept_No | Dept_Loc  |
+---------+-----------+
|     100 | Pune      |
|     101 | Mumbai    |
|     102 | Delhi     |
|     103 | Kolkata   |
|     104 | Surat     |
|     105 | Nashik    |
+---------+-----------+
```

```
6 rows in set (0.00 sec)

mysql> select * from employee;
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
| Emp_Id | F_Name   | L_Name   | DOB        | Join_Date  | Address | Gender | Dept_Id | Salary |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
|      0 | Madhvika | Jadhav   | 1988-02-22 | 2011-03-22 | Indore  | F      |     105 |  37000 |
|    111 | Pranil   | Patil    | 1985-04-08 | 2008-05-05 | Pune    | M      |     100 |  95000 |
|    222 | Mohan    | Sharma   | 1982-07-23 | 2009-08-11 | Delhi   | M      |     101 |  60000 |
|    333 | Raghuvir | Prasad   | 1983-08-11 | 2009-06-21 | Delhi   | M      |     102 |  65000 |
|    444 | Ram      | Kulkarni | 1979-09-11 | 2008-04-23 | Mumbai  | M      |     103 |  77000 |
|    555 | Rohit    | Jadhav   | 1987-03-10 | 2009-03-03 | Nashik  | M      |     104 |  55000 |
|    666 | Sakshi   | Patil    | 1989-07-21 | 2011-02-22 | Pune    | F      |     101 |  30000 |
|    777 | Samiksha | More     | 1987-01-17 | 2012-04-21 | Surat   | F      |     102 |  55000 |
|    888 | Arti     | Vaidya   | 1985-07-22 | 2011-07-22 | Noida   | F      |     103 |  56000 |
|    999 | Sam      | Wadekar  | 1985-09-21 | 2012-12-17 | Pune    | F      |     104 |  83000 |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
10 rows in set (0.00 sec)

mysql> select * from project;
+-------------+--------+--------------+----------+
| Pro_Name    | Pro_No | Pro_Location | Dept_Num |
+-------------+--------+--------------+----------+
| CodePro     |   4141 | Pune         |      100 |
| NH47        |   4142 | Mumbai       |      101 |
| HAL         |   4143 | Nagpur       |      102 |
| BHEL        |   4144 | Nashik       |      103 |
| CodeChef    |   4145 | Surat        |      104 |
| CodingNinja |   4146 | Jalgaon      |      105 |
+-------------+--------+--------------+----------+
6 rows in set (0.00 sec)

mysql> select * from works_on;
+----------+-----------+---------+----------+-----+
| W_Emp_ID | W_Pro_Num | Dept_No | Relation | Hrs |
+----------+-----------+---------+----------+-----+
|      111 |      4141 |     100 | Manager  | 310 |
|      222 |      4142 |     101 | Manager  | 320 |
|      333 |      4143 |     102 | Manager  | 330 |
|      444 |      4144 |     103 | Jr.Engg  | 340 |
|      555 |      4145 |     104 | Sr.Engg  | 350 |
|      666 |      4146 |     105 | Sr.Engg  | 360 |
|      777 |      4141 |     101 | Clerk    | 370 |
|      888 |      4143 |     103 | Clerk    | 380 |
|      999 |      4142 |     102 | Jr.Engg  | 390 |
|        0 |      4144 |     105 | Sr.Engg  | 400 |
+----------+-----------+---------+----------+-----+
10 rows in set (0.00 sec)
```

- **Queries**

1. Find Employee ID of who works for CodeChef project

```
mysql> select Emp_Id,F_Name from employee where Dept_Id In
    -> (select Dept_Num from Project where Pro_Name = 'CodeChef');
+--------+--------+
| Emp_Id | F_Name |
+--------+--------+
|    555 | Rohit  |
|    999 | Sam    |
+--------+--------+
2 rows in set (0.01 sec)
```

2. Find the First Name and Last Name of Employee who worked more than 350 hrs.

```
mysql> select F_Name,L_Name from Employee
    -> where Emp_Id In
    -> (select W_Emp_Id from Works_On where Hrs > 350);
+----------+---------+
| F_Name   | L_Name  |
+----------+---------+
| Sakshi   | Patil   |
| Samiksha | More    |
| Arti     | Vaidya  |
| Sam      | Wadekar |
| Madhvika | Jadhav  |
+----------+---------+
5 rows in set (0.00 sec)
```

3. Find the Dept Name of project ‘Mumbai’

```
mysql> select Dept_Name from Department
    -> where Dept_No In
    -> (select Dept_Num from Project where Pro_Location ='Mumbai');
+-----------+
| Dept_Name |
+-----------+
| Civil     |
+-----------+
1 row in set (0.01 sec)
```

4. Select All data of employee who works as 'Jr.Eng.' and 'Sr.Eng.' and working Hrs are greater than 320

```
mysql> select * from Employee
    -> where Emp_Id In
    -> (select W_Emp_Id from Works_On
    -> where Relation = 'Jr.Engg' or Relation = 'Sr.Engg'
    -> and Hrs > 320);
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
| Emp_Id | F_Name   | L_Name   | DOB        | Join_Date  | Address | Gender | Dept_Id | Salary |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
|    444 | Ram      | Kulkarni | 1979-09-11 | 2008-04-23 | Mumbai  | M      |     103 |  80000 |
|    555 | Rohit    | Jadhav   | 1987-03-10 | 2009-03-03 | Nashik  | M      |     104 |  55000 |
|    666 | Sakshi   | Patil    | 1989-07-21 | 2011-02-22 | Pune    | F      |     101 |  30000 |
|    999 | Sam      | Wadekar  | 1985-09-21 | 2012-12-17 | Pune    | F      |     104 |  83000 |
|      0 | Madhvika | Jadhav   | 1988-02-22 | 2011-03-22 | Indore  | F      |     105 |  37000 |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
5 rows in set (0.01 sec)
```

5. Create Backup Table EMP_Backup Contains Only Emp_ID, F_Name,L_Name,Join_Date, Salary Columns. Insert the values into EMP_Backup table with the help of sub query from Employee table

```
mysql> create table Emp_Backup(Emp_Id int(10),F_Name varchar(20),L_Name varchar(20),Join_Date date,Salary int(10));
Query OK, 0 rows affected, 2 warnings (0.20 sec)

mysql> Insert into Emp_Backup
    -> (select Emp_Id,F_Name,L_Name,Join_Date,Salary
    -> from Employee where Emp_Id is Not Null);
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> Select * from Emp_Backup;
+--------+----------+----------+------------+--------+
| Emp_Id | F_Name   | L_Name   | Join_Date  | Salary |
+--------+----------+----------+------------+--------+
|      0 | Madhvika | Jadhav   | 2011-03-22 |  37000 |
|    111 | Pranil   | Patil    | 2008-05-05 |  95000 |
|    222 | Mohan    | Sharma   | 2009-08-11 |  60000 |
|    333 | Raghuvir | Prasad   | 2009-06-21 |  65000 |
|    444 | Ram      | Kulkarni | 2008-04-23 |  80000 |
|    555 | Rohit    | Jadhav   | 2009-03-03 |  55000 |
|    666 | Sakshi   | Patil    | 2011-02-22 |  30000 |
|    777 | Samiksha | More     | 2012-04-21 |  55000 |
|    888 | Arti     | Vaidya   | 2011-07-22 |  56000 |
|    999 | Sam      | Wadekar  | 2012-12-17 |  83000 |
+--------+----------+----------+------------+--------+
10 rows in set (0.00 sec)
```

6. Update the salary in EMP_Backup table by 30% who works for project ‘BHEL’

```
mysql> update Emp_Backup set Salary = Salary+Salary*0.30
    -> where Emp_Id In
    -> (select Emp_Id from Employee where Dept_Id = 103);
Query OK, 2 rows affected (0.02 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> Select * from Emp_Backup;
+--------+----------+----------+------------+--------+
| Emp_Id | F_Name   | L_Name   | Join_Date  | Salary |
+--------+----------+----------+------------+--------+
|      0 | Madhvika | Jadhav   | 2011-03-22 |  37000 |
|    111 | Pranil   | Patil    | 2008-05-05 |  95000 |
|    222 | Mohan    | Sharma   | 2009-08-11 |  60000 |
|    333 | Raghuvir | Prasad   | 2009-06-21 |  65000 |
|    444 | Ram      | Kulkarni | 2008-04-23 | 104000 |
|    555 | Rohit    | Jadhav   | 2009-03-03 |  55000 |
|    666 | Sakshi   | Patil    | 2011-02-22 |  30000 |
|    777 | Samiksha | More     | 2012-04-21 |  55000 |
|    888 | Arti     | Vaidya   | 2011-07-22 |  72800 |
|    999 | Sam      | Wadekar  | 2012-12-17 |  83000 |
+--------+----------+----------+------------+--------+
10 rows in set (0.00 sec)
```

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 5: Write a stored procedure to insert and retrieve data from database table.

Consider the following schema and creates tables.

1. Borrower (Roll_No, Name, Date_of_Issue, Book_Name, Status)

```
mysql> use pract;
Database changed
mysql> Create Table Borrower (Roll_No int Primary Key, Name Varchar(15),
    ->  Date_of_Issue Date, Book_Name Varchar(15), Status Varchar(10) );
Query OK, 0 rows affected (0.10 sec)

mysql> Insert Into Borrower Values(1,'Samruddhi','2019-08-10','TOC','Issue');
Query OK, 1 row affected (0.03 sec)

mysql> Insert Into Borrower Values(2,'Waidehee','2019-08-10','DBS','Issue');
Query OK, 1 row affected (0.01 sec)

mysql> Insert Into Borrower Values(3,'Sakshi','2019-07-14','SE','Issue');
Query OK, 1 row affected (0.00 sec)

mysql> Insert Into Borrower Values(4,'Ananya','2019-07-20','HCI','Issue');
Query OK, 1 row affected (0.00 sec)

mysql> Insert Into Borrower Values(5,'Arya','2019-06-15','SQL','Issue');
Query OK, 1 row affected (0.01 sec)

mysql> update Borrower set Date_of_Issue = '2019-07-13'
    -> where Roll_No = 2
    -> ;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Borrower;
+---------+-----------+---------------+-----------+--------+
| Roll_No | Name      | Date_of_Issue | Book_Name | Status |
+---------+-----------+---------------+-----------+--------+
|       1 | Samruddhi | 2019-08-10    | TOC       | Issue  |
|       2 | Waidehee  | 2019-07-13    | DBS       | Issue  |
|       3 | Sakshi    | 2019-07-14    | SE        | Issue  |
|       4 | Ananya    | 2019-07-20    | HCI       | Issue  |
|       5 | Arya      | 2019-06-15    | SQL       | Issue  |
+---------+-----------+---------------+-----------+--------+
5 rows in set (0.01 sec)
```

2. Fine(Roll_no,Fine_Date,Amount)

```
mysql> create table Fine(Roll_no int References Borrower(Roll_No), Fine_Date Date, Amount int);
Query OK, 0 rows affected (0.10 sec)
```

3. Before writing the procedure change the Delimiter //

a. Create Procedure:

```
mysql> Delimiter //
mysql>
mysql> create Procedure Sam_GetData()
    -> Begin
    -> select * from Borrower;
    -> End //
Query OK, 0 rows affected (0.06 sec)

mysql> call Sam_GetData()//
+---------+-----------+---------------+-----------+--------+
| Roll_No | Name      | Date_of_Issue | Book_Name | Status |
+---------+-----------+---------------+-----------+--------+
|       1 | Samruddhi | 2019-08-10    | TOC       | Issue  |
|       2 | Waidehee  | 2019-07-13    | DBS       | Issue  |
|       3 | Sakshi    | 2019-07-14    | SE        | Issue  |
|       4 | Ananya    | 2019-07-20    | HCI       | Issue  |
|       5 | Arya      | 2019-06-15    | SQL       | Issue  |
+---------+-----------+---------------+-----------+--------+
5 rows in set (0.02 sec)

Query OK, 0 rows affected (0.02 sec)
```

b. With IN Parameter :

```
mysql> Create Procedure Sam_GateData1( IN ID Int)
    ->  Begin
    ->  Select * from Borrower
    ->  Where Roll_NO= ID;
    ->  End //
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> call Sam_GateData1(5)//
+---------+------+---------------+-----------+--------+
| Roll_No | Name | Date_of_Issue | Book_Name | Status |
+---------+------+---------------+-----------+--------+
|       5 | Arya | 2019-06-15    | SQL       | Issue  |
+---------+------+---------------+-----------+--------+
1 row in set (0.01 sec)
```

c. With Out Parameter:

```
mysql> Create Procedure Sam_Data_out(OUT R1 Int)
    ->  Begin
    ->  Set R1=10;
    ->  End //
Query OK, 0 rows affected (0.02 sec)

mysql> call Sam_Data_out(@ S1)//
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL s
1)' at line 1
mysql> call Sam_Data_out(@S1)//
Query OK, 0 rows affected (0.01 sec)

mysql> select @S1//
+------+
| @S1  |
+------+
|   10 |
+------+
```

d. Variables in Procedures:

Query- Insert new row into table Borrower

Solution: As per the Borrower table we required five variables to insert data into borrower table

```
mysql> Create Procedure Sam_Variables( )
    ->  Begin
    ->  Declare Var1 int;
    ->  Declare Var2, Var3,Var4 varchar (20);
    ->  Declare Var5 Date;
    ->  SET Var1=6;
    ->  SET Var2= 'Dev';
    ->  SET Var3= 'BC';
    ->  SET Var4= 'Issue';
    ->  SET Var5= '2019-08-22';
    ->  Insert into Borrower values (Var1,Var2,Var5,Var3,Var4);
    ->  Select * from Borrower;
    ->  End//
Query OK, 0 rows affected (0.02 sec)

mysql> call Sam_Variables()//
+---------+-----------+---------------+-----------+--------+
| Roll_No | Name      | Date_of_Issue | Book_Name | Status |
+---------+-----------+---------------+-----------+--------+
|       1 | Samruddhi | 2019-08-10    | TOC       | Issue  |
|       2 | Waidehee  | 2019-07-13    | DBS       | Issue  |
|       3 | Sakshi    | 2019-07-14    | SE        | Issue  |
|       4 | Ananya    | 2019-07-20    | HCI       | Issue  |
|       5 | Arya      | 2019-06-15    | SQL       | Issue  |
|       6 | Dev       | 2019-08-22    | BC        | Issue  |
+---------+-----------+---------------+-----------+--------+
6 rows in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)
```

4. IF-Then-Else

```
mysql> Create Procedure Sam_IF_Then_Else(IN T1 Int)
    ->  Begin
    ->  IF(T1>1&&T1<7)
    ->  Then
    ->  select * from Borrower
    ->  where Roll_NO=T1;
    ->  Else
    ->  Select 'Not Found';
    ->  End If;
    ->  End//
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> call Sam_IF_Then_Else(3)//
+---------+--------+---------------+-----------+--------+
| Roll_No | Name   | Date_of_Issue | Book_Name | Status |
+---------+--------+---------------+-----------+--------+
|       3 | Sakshi | 2019-07-14    | SE        | Issue  |
+---------+--------+---------------+-----------+--------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> call Sam_IF_Then_Else(8)//
+------------+
| Not Found |
+------------+
| Not Found |
+------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

5. Loops:

a. WHILE ... END WHILE

```
mysql> Create table Temp (val int)//
Query OK, 0 rows affected (0.04 sec)

mysql> Create Procedure Sam_While_Loop()
    ->  Begin
    ->  DECLARE v INT;
    ->  SET v = 0;
    ->  WHILE v < 5 DO
    ->  Insert Into Temp Values (v);
    ->  SET v = v + 1;
    ->  END WHILE;
    ->  END//
Query OK, 0 rows affected (0.01 sec)

mysql> call Sam_While_Loop()//
Query OK, 1 row affected (0.03 sec)

mysql> select * from Temp//
+------+
| val  |
+------+
|    0 |
|    1 |
|    2 |
|    3 |
|    4 |
+------+
5 rows in set (0.00 sec)
```

b. REPEAT ... END REPEAT

```
mysql> delimiter //
mysql> Create table Temp1(val int) //
Query OK, 0 rows affected (0.12 sec)
```

```
mysql> Create Procedure SW_Repeat_Loop ()
    -> Begin
    -> Declare v Int;
    -> SET v = 0;
    -> REPEAT
    -> Insert into Temp1 values (v);
    -> SET v = v + 1;
    -> UNTIL v >= 5
    -> END REPEAT;
    -> End; //
Query OK, 0 rows affected (0.01 sec)

mysql> call SW_Repeat_Loop()//
Query OK, 1 row affected (0.05 sec)

mysql> Select * from Temp 1//
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corre
' at line 1
mysql> Select * from Temp1 //
+------+
| val  |
+------+
|    0 |
|    1 |
|    2 |
|    3 |
|    4 |
+------+
5 rows in set (0.00 sec)
```

Query: Check the number of days (from date of issue), if days are between 15 to 30 then fine amount will be Rs 5per day. If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day. After submitting the book, status will change from Issue to Return. If condition of fine is true, then details will be stored into fine table. Delimiter //

/**********

********** Description: Accpet Roll Number and Book Name and calculate Fine amount as per Number of days. Handle Exception if Roll Number not found. Create Date: 03-05-2022

****

****

```
mysql> Create Procedure LIB( Roll_New int,Book_Name Varchar(15))
    -> BEGIN
    ->  Declare X1 integer;
    ->  Declare Continue Handler for NOT FOUND
    ->  BEGIN
    ->  Select 'Not Found';
    ->  END;
    ->  Select DATEDIFF ( curdate(), Date_of_Issue) into X1 from Borrower where
    -> Roll_NO=Roll_New;
    ->  IF (X1>15&&X1<30)
    ->  Then
    ->  Insert Into Fine Values( Roll_New, curdate(),(X1*5));
    ->  UPDATE Borrower SET Status='Return' WHERE Roll_NO=Roll_New;
    ->  End IF;
    ->  IF (X1>30)
    ->  Then
    ->  Insert into Fine values( Roll_New, curdate(),(X1*50));
    ->  UPDATE Borrower SET Status='Return' WHERE Roll_NO=Roll_New;
    ->  End IF ;
    -> END;
    ->  //
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> select * from Fine//
Empty set (0.05 sec)
```

```
mysql> call LIB(1,'TOC');//
Query OK, 1 row affected (0.04 sec)

mysql> call LIB(2,'BC');//
Query OK, 1 row affected (0.02 sec)

mysql> call LIB(3,'DBS');//
Query OK, 1 row affected (0.02 sec)

mysql> call LIB(4,'HCI');//
Query OK, 1 row affected (0.01 sec)

mysql> call LIB(5,'SE');//
Query OK, 1 row affected (0.01 sec)

mysql> select * from Fine//
+---------+------------+--------+
| Roll_no | Fine_Date  | Amount |
+---------+------------+--------+
|       1 | 2022-12-04 |  60600 |
|       2 | 2022-12-04 |  62000 |
|       3 | 2022-12-04 |  61950 |
|       4 | 2022-12-04 |  61650 |
|       5 | 2022-12-04 |  63400 |
+---------+------------+--------+
5 rows in set (0.01 sec)
```

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 6: Write a SQL procedure for cursor and trigger.

Procedures :

Query:

Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and900 category is first class, if marks 899 and 825 category is Higher Second Class. Write a PL/SQL block for using procedure created with above requirement.

create table Stud_marks

```
mysql> create table Stud_Marks(Roll_No int,Name varchar(20),Total_Marks int(10));
Query OK, 0 rows affected, 1 warning (0.16 sec)
```

```
mysql> select * from Stud_Marks//
+---------+-------+-------------+
| Roll_No | Name  | Total_Marks |
+---------+-------+-------------+
|       1 | Jayu  |         500 |
|       2 | Gayu  |         400 |
|       3 | Aashu |         450 |
|       4 | Samu  |         350 |
|       5 | Nanu  |         300 |
+---------+-------+-------------+
5 rows in set (0.00 sec)
```

Create table result

```
mysql> create table Result(Roll_No int,Class varchar(20))//
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> desc Result//
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| Roll_No | int         | YES  |     | NULL    |       |
| Class   | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
2 rows in set (0.06 sec)
```

Create Procedure:

```
mysql> create procedure pro_grade()
    -> begin
    -> declare v1 int;
    -> declare v2 int;
    -> declare nomorerows int;
    -> declare grade_check cursor for select roll_no,total_marks from stud_marks ;
    -> declare continue handler for not found set nomorerows = 1;
    -> set nomorerows = 0;
    -> open grade_check;
    -> loop_name:loop
    -> fetch grade_check into v1,v2;
    -> if (v2>=450&&v2<=500)
    -> then
    -> insert into result values(v1,'distinction');
    -> end if;
    -> if(v2>=350&&v2<=449)
    -> then
    -> insert into result values(v1,'first class');
    -> end if;
    -> if(v2>=250&&v2<=349)
    -> then
    -> insert into result values(v1,'higher sec.class');
    -> end if;
    -> if nomorerows
    -> then
    -> leave loop_name;
    -> end if;
    -> end loop loop_name;
    -> close grade_check;
    -> end//
Query OK, 0 rows affected, 3 warnings (0.03 sec)

mysql> call pro_grade()//
Query OK, 1 row affected (0.03 sec)

mysql> select * from result//
+---------+------------------+
| Roll_No | Class            |
+---------+------------------+
|       1 | distinction      |
|       2 | first class      |
|       3 | distinction      |
|       4 | first class      |
|       5 | higher sec.class |
|       5 | higher sec.class |
+---------+------------------+
6 rows in set (0.00 sec)
```

Triggers:

Query: Write a database trigger on Library table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in Library_Audit table.

Create table Library:

```
mysql> Delimiter //
mysql> Create table Library
    -> (Book_ID int, Title varchar(20), Author_Name varchar(20), Dept varchar(20)); //
Query OK, 0 rows affected (0.09 sec)

mysql> Insert into Library Values(101,'OS','Galvin','Computer') //
Query OK, 1 row affected (0.03 sec)

mysql> Insert into Library Values(102,'SM','Albert','Civil') //
Query OK, 1 row affected (0.01 sec)

mysql> Insert into Library Values(103,'DE','M Mano','Electronics') //
Query OK, 1 row affected (0.01 sec)

mysql> Select * from Library //
+---------+-------+-------------+-------------+
| Book_ID | Title | Author_Name | Dept        |
+---------+-------+-------------+-------------+
|     101 | OS    | Galvin      | Computer    |
|     102 | SM    | Albert      | Civil       |
|     103 | DE    | M Mano      | Electronics |
+---------+-------+-------------+-------------+
3 rows in set (0.01 sec)
```

Create table Library_Audit

```
mysql> Create table Lib_Audit
    -> ( Book_ID int, Old_Book_ID_Before_Update Int, Up_Del_Date date, Status varchar(20)); //
Query OK, 0 rows affected (0.04 sec)
```

Create Trigger

1. Trigger for Update data of library table:

```
mysql> Create TRIGGER Lib_Trig_Update AFTER update ON Library
    -> FOR EACH ROW
    -> BEGIN
    -> INSERT INTO LIB_AUDIT (Book_ID, Old_Book_ID_Before_Update, Up_Del_Date, Status)
    -> Values (New.Book_ID, Old.Book_ID, Now(), 'Update');
    -> END; //
Query OK, 0 rows affected (0.02 sec)

mysql> Update Library set Book_ID = 501 where Book_ID = 102 //
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> Select * from Lib_Audit //
+---------+---------------------------+-------------+--------+
| Book_ID | Old_Book_ID_Before_Update | Up_Del_Date | Status |
+---------+---------------------------+-------------+--------+
|     501 |                       102 | 2022-12-05  | Update |
+---------+---------------------------+-------------+--------+
1 row in set (0.00 sec)

mysql> Update Library set Book_ID = 400 where Book_ID = 103 //
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> Select * from Lib_Audit //
+---------+---------------------------+-------------+--------+
| Book_ID | Old_Book_ID_Before_Update | Up_Del_Date | Status |
+---------+---------------------------+-------------+--------+
|     501 |                       102 | 2022-12-05  | Update |
|     400 |                       103 | 2022-12-05  | Update |
+---------+---------------------------+-------------+--------+
2 rows in set (0.00 sec)

mysql> Select * from Library //
+---------+-------+-------------+-------------+
| Book_ID | Title | Author_Name | Dept        |
+---------+-------+-------------+-------------+
|     101 | OS    | Galvin      | Computer    |
|     501 | SM    | Albert      | Civil       |
|     400 | DE    | M Mano      | Electronics |
+---------+-------+-------------+-------------+
3 rows in set (0.00 sec)
```

Name: Samruddhi Wadekar

Roll no : 54

Batch: T4

Practical 7: Write SQL to apply Aggregating Data using Group functions

Table Employee:

```
mysql> desc Employee;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Emp_Id    | int         | NO   | PRI | NULL    |       |
| F_Name    | varchar(20) | NO   |     | NULL    |       |
| L_Name    | varchar(20) | NO   |     | NULL    |       |
| DOB       | date        | NO   |     | NULL    |       |
| Join_Date | date        | NO   |     | NULL    |       |
| Address   | varchar(20) | NO   |     | NULL    |       |
| Gender    | varchar(20) | NO   |     | NULL    |       |
| Dept_Id   | int         | NO   | MUL | NULL    |       |
| Salary    | int         | NO   |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
9 rows in set (0.04 sec)

mysql> Select * From Employee;
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
| Emp_Id | F_Name   | L_Name   | DOB        | Join_Date  | Address | Gender | Dept_Id | Salary |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
|      0 | Madhvika | Jadhav   | 1988-02-22 | 2011-03-22 | Indore  | F      |     105 |  37000 |
|    111 | Pranil   | Patil    | 1985-04-08 | 2008-05-05 | Pune    | M      |     100 |  95000 |
|    222 | Mohan    | Sharma   | 1982-07-23 | 2009-08-11 | Delhi   | M      |     101 |  60000 |
|    333 | Raghuvir | Prasad   | 1983-08-11 | 2009-06-21 | Delhi   | M      |     102 |  65000 |
|    444 | Ram      | Kulkarni | 1979-09-11 | 2008-04-23 | Mumbai  | M      |     103 |  80000 |
|    555 | Rohit    | Jadhav   | 1987-03-10 | 2009-03-03 | Nashik  | M      |     104 |  55000 |
|    666 | Sakshi   | Patil    | 1989-07-21 | 2011-02-22 | Pune    | F      |     101 |  30000 |
|    777 | Samiksha | More     | 1987-01-17 | 2012-04-21 | Surat   | F      |     102 |  55000 |
|    888 | Arti     | Vaidya   | 1985-07-22 | 2011-07-22 | Noida   | F      |     103 |  56000 |
|    999 | Sam      | Wadekar  | 1985-09-21 | 2012-12-17 | Pune    | F      |     104 |  83000 |
+--------+----------+----------+------------+------------+---------+--------+---------+--------+
10 rows in set (0.03 sec)
```

Table Department:

```
mysql> desc Department;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Dept_No       | int         | NO   | PRI | NULL    |       |
| Dept_Name     | varchar(20) | NO   |     | NULL    |       |
| Mgr_ID        | int         | NO   |     | NULL    |       |
| Mgr_Start_Date | date       | NO   |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> Select * from Department;
+---------+------------+--------+----------------+
| Dept_No | Dept_Name  | Mgr_ID | Mgr_Start_Date |
+---------+------------+--------+----------------+
|     100 | Computer   |    111 | 2010-06-23     |
|     101 | Civil      |    222 | 2011-08-17     |
|     102 | Mechanical |    333 | 2013-09-13     |
|     103 | Electrical |    444 | 2012-10-29     |
|     104 | IT         |    555 | 2014-07-15     |
|     105 | Metallurgy |      0 | 2015-10-27     |
+---------+------------+--------+----------------+
6 rows in set (0.02 sec)
```

Table Project:

```
mysql> desc Project;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| Pro_Name     | varchar(20) | NO   |     | NULL    |       |
| Pro_No       | int         | NO   | PRI | NULL    |       |
| Pro_Location | varchar(20) | NO   |     | NULL    |       |
| Dept_Num     | int         | NO   | MUL | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> Select * from Project;
+-------------+--------+--------------+----------+
| Pro_Name    | Pro_No | Pro_Location | Dept_Num |
+-------------+--------+--------------+----------+
| CodePro     |   4141 | Pune         |      100 |
| NH47        |   4142 | Mumbai       |      101 |
| HAL         |   4143 | Nagpur       |      102 |
| BHEL        |   4144 | Nashik       |      103 |
| CodeChef    |   4145 | Surat        |      104 |
| CodingNinja |   4146 | Jalgaon      |      105 |
+-------------+--------+--------------+----------+
6 rows in set (0.02 sec)
```

Aggregate Functions

1.maximum salary, the minimum salary, and the average salary among all employees

```
mysql> select count(*) from Employee
    -> ,Department
    -> where Dept_Id = Dept_No and Dept_Name = 'Computer';
+----------+
| count(*) |
+----------+
|        1 |
+----------+
1 row in set (0.01 sec)
```

2.Total number of employees in the company:

```
mysql> select count(*) as Total_Employee
    -> from Employee;
+----------------+
| Total_Employee |
+----------------+
|             10 |
+----------------+
1 row in set (0.03 sec)
```

3. No. of employees in the 'Computer’ department

```
mysql> select max(salary) as Max_Salary,min(salary) as Min_Salary,avg(salary) as Avg_Salary from Employee;
+------------+------------+------------+
| Max_Salary | Min_Salary | Avg_Salary |
+------------+------------+------------+
|      95000 |      30000 | 61600.0000 |
+------------+------------+------------+
1 row in set (0.01 sec)
```

Grouping :

1. Department number, the number of employees in the department, and their average salary from each department

```
mysql> select Dept_Id, count(*) as No_Of_Employee,avg(Salary) as Avg_Salary
    -> from Employee
    -> group by Dept_Id;
+---------+----------------+------------+
| Dept_Id | No_Of_Employee | Avg_Salary |
+---------+----------------+------------+
|     100 |              1 | 95000.0000 |
|     101 |              2 | 45000.0000 |
|     102 |              2 | 60000.0000 |
|     103 |              2 | 68000.0000 |
|     104 |              2 | 69000.0000 |
|     105 |              1 | 37000.0000 |
+---------+----------------+------------+
6 rows in set (0.01 sec)
```

2. For each project on which more than two employees work, retrieve the project number, project name, and the number of employees who work on that project

```
mysql> select Pro_No,Pro_Name,count(*) as No_Of_Employee
    -> from Project,Employee
    -> where Dept_Id = Dept_Num
    -> group by Pro_No having count(*) > 2;
Empty set (0.00 sec)

mysql> select Pro_No,Pro_Name,count(*) as No_Of_Employee
    -> from Project,Employee
    -> where Dept_Id = Dept_Num
    -> group by Pro_No having count(*) >= 1;
+--------+-------------+----------------+
| Pro_No | Pro_Name    | No_Of_Employee |
+--------+-------------+----------------+
|   4141 | CodePro     |              1 |
|   4142 | NH47        |              2 |
|   4143 | HAL         |              2 |
|   4144 | BHEL        |              2 |
|   4145 | CodeChef    |              2 |
|   4146 | CodingNinja |              1 |
+--------+-------------+----------------+
6 rows in set (0.00 sec)
```

Name: Samruddhi Wadekar

Roll no.: 54

Batch: T4

Practical 8: Design a queries for suitable database application using SQL DML statements: all types of Join,Views.

Table Supplier

```
mysql> desc Supplier;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Sup_Id   | int         | YES  |     | NULL    |       |
| Sup_Name | varchar(20) | YES  |     | NULL    |       |
| Address  | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.03 sec)

mysql> select * from Supplier;
+--------+---------------+-----------+
| Sup_Id | Sup_Name      | Address   |
+--------+---------------+-----------+
|    101 | Akash Auto    | Chennai   |
|    102 | Hitachi       | Mumbai    |
|    103 | OM Breaks     | Pune      |
|    104 | JK Tires      | Ahemdabad |
|    105 | Telco Glass   | Hyderabad |
|    106 | Ashok Leyland | Bangluru  |
|    107 | Bharat Benz   | Kolkata   |
+--------+---------------+-----------+
7 rows in set (0.00 sec)
```

Table Parts

```
mysql> desc Parts
    -> ;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Part_Id   | int         | YES  |     | NULL    |       |
| Part_Name | varchar(20) | YES  |     | NULL    |       |
| Part_Type | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql> select * from Parts;
+---------+--------------+--------------+
| Part_Id | Part_Name    | Part_Type    |
+---------+--------------+--------------+
|     501 | Fuel Pump    | Engine       |
|     502 | Break Liners | Break System |
|     503 | Head Lamps   | Car Body     |
|     504 | Gear Box     | Engine       |
|     505 | AC Button    | Dashboard    |
|     506 | Music System | Dashboard    |
|     507 | Battery      | Electric     |
+---------+--------------+--------------+
7 rows in set (0.00 sec)
```

Table Orders

```
mysql> desc Orders;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| Order_No   | int  | YES  |     | NULL    |       |
| Part_ID    | int  | YES  |     | NULL    |       |
| Sup_ID     | int  | YES  |     | NULL    |       |
| Order_Date | date | YES  |     | NULL    |       |
+------------+------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> select * from Orders;
+----------+---------+--------+------------+
| Order_No | Part_ID | Sup_ID | Order_Date |
+----------+---------+--------+------------+
|     1001 |     501 |    101 | 2017-02-03 |
|     1002 |     502 |    102 | 2017-04-10 |
|     1003 |     503 |    104 | 2017-05-11 |
|     1004 |     504 |    102 | 2017-05-11 |
|     1005 |     505 |    101 | 2017-06-11 |
|     1006 |     506 |    103 | 2017-07-11 |
+----------+---------+--------+------------+
6 rows in set (0.00 sec)
```

Table Catalog

```
mysql> desc Catalog;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| P_ID            | int         | YES  |     | NULL    |       |
| Part_Name       | varchar(20) | YES  |     | NULL    |       |
| Cost            | int         | YES  |     | NULL    |       |
| Available_Stock | int         | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> select * from Catalog;
+------+--------------+-------+-----------------+
| P_ID | Part_Name    | Cost  | Available_Stock |
+------+--------------+-------+-----------------+
|  501 | Fuel Pump    |  3000 |              12 |
|  502 | Break Liners |  1600 |              10 |
|  503 | Head Lamps   |  7000 |               6 |
|  504 | Gear Box     | 12000 |               2 |
|  505 | AC Button    |   700 |              25 |
|  506 | Music System | 22000 |               2 |
+------+--------------+-------+-----------------+
6 rows in set (0.00 sec)
```

1. Find the Supplier ID's and date of supply of supplier from the list of suppliers (Solve using Join)

```
mysql> select supplier.sup_id,order_date
    -> from supplier join orders
    -> on supplier.sup_id = orders.sup_id;
+--------+------------+
| sup_id | order_date |
+--------+------------+
|    101 | 2017-02-03 |
|    101 | 2017-06-11 |
|    102 | 2017-04-10 |
|    102 | 2017-05-11 |
|    103 | 2017-07-11 |
|    104 | 2017-05-11 |
+--------+------------+
6 rows in set (0.01 sec)
```

2. Find All Suppliers who supply parts as well as not

```
mysql> select supplier.sup_id,order_date
    -> from supplier left outer join
    -> orders
    -> on supplier.sup_id = orders.sup_id;
+--------+------------+
| sup_id | order_date |
+--------+------------+
|    101 | 2017-02-03 |
|    102 | 2017-04-10 |
|    104 | 2017-05-11 |
|    102 | 2017-05-11 |
|    101 | 2017-06-11 |
|    103 | 2017-07-11 |
|    105 | NULL       |
|    106 | NULL       |
|    107 | NULL       |
+--------+------------+
9 rows in set (0.01 sec)
```

3. Find all suppliers who supply parts.(Using Join)

```
mysql> select supplier.sup_id,order_date
    -> from supplier right outer join
    -> orders
    -> on supplier.sup_id = orders.sup_id;
+--------+------------+
| sup_id | order_date |
+--------+------------+
|    101 | 2017-02-03 |
|    101 | 2017-06-11 |
|    102 | 2017-04-10 |
|    102 | 2017-05-11 |
|    103 | 2017-07-11 |
|    104 | 2017-05-11 |
+--------+------------+
6 rows in set (0.00 sec)
```

4. Create View for total cost of every part which is ordered

```
mysql> create view total_part_cost as
    -> select part_name,cost * available_stock as total_cost
    -> from catalog;
Query OK, 0 rows affected (0.03 sec)

mysql> select * from Total_Part_Cost;
+--------------+------------+
| part_name    | total_cost |
+--------------+------------+
| Fuel Pump    |      36000 |
| Break Liners |      16000 |
| Head Lamps   |      42000 |
| Gear Box     |      24000 |
| AC Button    |      17500 |
| Music System |      44000 |
+--------------+------------+
6 rows in set (0.02 sec)
```

5. Find the Details of supplier who supply a ‘Fuel Pumps’

```
mysql> select * from Supplier
    -> where sup_id =
    -> (select sup_id from orders
    -> where part_id =
    -> (select part_id from parts
    -> where part_name = 'Fuel Pump'));
+--------+------------+---------+
| Sup_Id | Sup_Name   | Address |
+--------+------------+---------+
|    101 | Akash Auto | Chennai |
+--------+------------+---------+
1 row in set (0.01 sec)
```