

Lab # 04: Database Development using MySQL

Lab04



Fall 2024

Submitted by: **Mohsin Sajjad**

Registration No: **22pwsce2149**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

A handwritten signature in black ink, reading "Mohsin Sajjad". The signature is written in a cursive style with a large initial 'M'.

Student Signature: _____

Submitted to:

Engr. Summeya Salahudin

Month Day, Year (02 23, 2025)

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

OBJECTIVES OF THE LAB

This lab aims at the understanding of:

- *An Example SQL Database Management System i.e. MySQL*
- *Using MySQL Command Line in Windows*
- *MySQL Command Categories*
- *Use of MySQL Commands for a Real-World Problem*

TASKS:

1. **Theory:** What is DDL, DML, TCL, and DCL? Explain in your own words. Also, list few commands in each language. (2 Marks)

Data Definition Language (DDL)

The overall design of a database is called the database schema. A database schema is specified by a set of definitions that are expressed using a data definition language. It is used for structuring the database. We use the data definition language statements **CREATE**, **ALTER**, **DROP**, **TRUNCATE** to create new objects, alter the structure of existing objects, completely remove objects from system, or delete all rows permanently from the table leaving the structure of the table.

Data Manipulation Language (DML)

These commands are the most frequently used SQL commands. They are used to query and manipulate existing objects like tables. DML commands are: **SELECT**, **UPDATE**, **INSERT**, and **DELETE**.

Transaction Control Language (TCL)

The changes made to the database are known as transaction. Transactions can be made permanent to a database by commit. The various commands in TCL are: **COMMIT**, **SAVEPOINT**, and **ROLLBACK**.

Data Control Language (DCL)

DCL statements are used for securing the database. DCL statements such as **GRANT** and **REVOKE** control access to database and affirm or revoke database transactions. It provides the user with privilege commands. The owner of the database can grant privileges or withdraw (revoke) privilege to other database users so that they can perform the operations accordingly on the tables.

Practical 1: Perform all the commands covered in this lab. In your lab report, write each command and show its output taken as a screenshot. Write comment about each command -----

Command: `create database sam_db;`

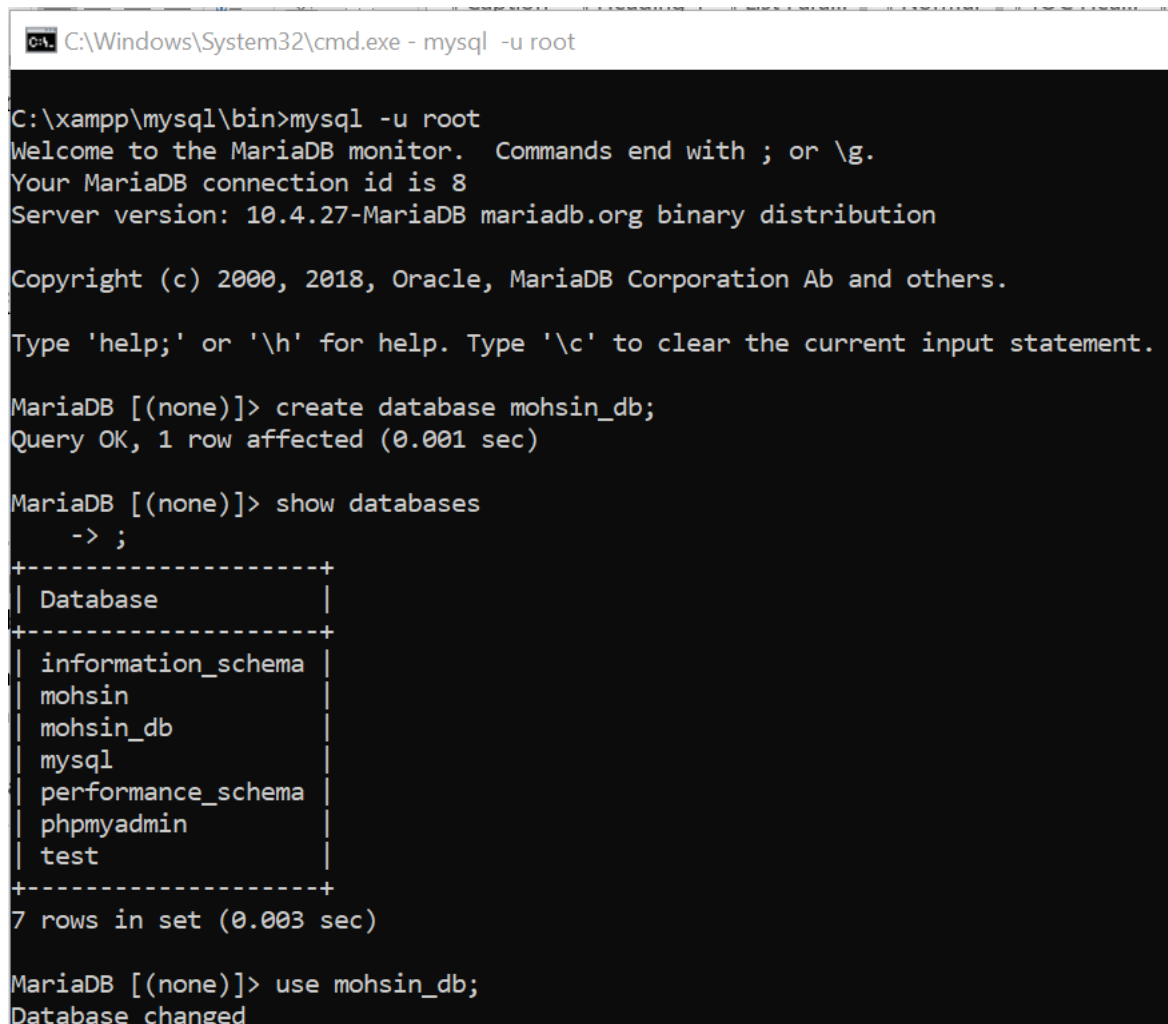
Detail: This command creates a new database sam_db.

Command: `show databases;`

Detail: This command lists all the existing databases in mysql environment.

Command: `use sam_db;`

Detail: This command switches over from default mysql database to specified database. In current case, switched database is sam_db.



```
C:\Windows\System32\cmd.exe - mysql -u root

C:\xampp\mysql\bin>mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 10.4.27-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database mohsin_db;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> show databases
-> ;
+-----+
| Database |
+-----+
| information_schema |
| mohsin |
| mohsin_db |
| mysql |
| performance_schema |
| phpmyadmin |
| test |
+-----+
7 rows in set (0.003 sec)

MariaDB [(none)]> use mohsin_db;
Database changed
```

Command: `create table department(`

`departmentID int not null auto_increment,`

```
name varchar(30),  
primary key(departmentID));
```

Detail: This command creates a new table department in sam_db database. It consists of two columns departmentID and name. The datatype of each column is specified along with its size. The primary key of table has been set to departmentID attribute. Note that auto_increment option automatically increments the primary key if it's not specified during record entries in table.

Command:

```
create table employee(  
    employeeID int not null auto_increment,  
    name varchar(80),  
    job varchar(30),  
    departmentID int default 0,  
    primary key(employeeID),  
    foreign key(departmentID) references department(departmentID));
```

Detail: This command creates a new table employee in sam_db database. It consists of four columns employeeID, name, job, & departmentID where employeeID has been set as primary key and departmentID as foreign key.

Command:

```
show tables;
```

Detail: This command shows all the existing tables in the sam_db database. Since two tables i.e. department and employee are created in above commands, so these two are listed.

Command:

```
describe department;
```

Detail: This command describes the table structure of department table. Description includes field or attribute names, their respective data types, primary key information, default value of any field and extra options (such as auto_increment).

C:\Windows\System32\cmd.exe - mysql -u root

```
MariaDB [mohsin_db]> create table department(
  -> departmentID int not null auto_increment,
  -> name varchar(30),
  -> primary key(departmentID));
Query OK, 0 rows affected (0.030 sec)

MariaDB [mohsin_db]> create table employee(
  -> employeeID int not null auto_increment,
  -> name varchar(80),
  -> job varchar(30),
  -> departmentID int default 0,
  -> primary key(employeeID),
  -> foreign key(departmentID) references department(departmentID));
Query OK, 0 rows affected (0.036 sec)

MariaDB [mohsin_db]> show tables;
+-----+
| Tables_in_mohsin_db |
+-----+
| department          |
| employee            |
+-----+
2 rows in set (0.001 sec)

MariaDB [mohsin_db]> describe department;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| departmentID | int(11)   | NO   | PRI | NULL    | auto_increment |
| name        | varchar(30) | YES  |     | NULL    |              |
```

Command: `insert into department values`

(10, 'Computer Systems Engineering'),
(15, 'Electrical Engineering'),
(20, 'Chemical Engineering'),
(25, 'Mining Engineering');

Detail: This command populates multiple records in department table.

Command: `insert into employee values`

(1, 'ABC', 'Lecturer', 10),
(3, 'ACB', 'Lecturer', 10),
(4, 'XYZ', 'Assistant Professor', 10),

(5, 'CAB', 'Lecturer', 15);

Detail: This command populates multiple records in employee table.

Command: `select * from employee;`

Detail: This command selects records from employee table where * means to select and show values from all the columns.

```
MariaDB [mohsin_db]> insert into department values
-> (10, 'Computer Systems Engineering'),
-> (15, 'Electrical Engineering'),
-> (20, 'Chemical Engineering'),
-> (25, 'Mining Engineering');
Query OK, 4 rows affected (0.021 sec)
Records: 4 Duplicates: 0 Warnings: 0

MariaDB [mohsin_db]> insert into employee values
-> (1, 'ABC', 'Lecturer', 10),
-> (3, 'ACB', 'Lecturer', 10),
-> (4, 'XYZ', 'Assistant Professor', 10),
-> (5, 'CAB', 'Lecturer', 15);
Query OK, 4 rows affected (0.012 sec)
Records: 4 Duplicates: 0 Warnings: 0

MariaDB [mohsin_db]> select * from employee;
+-----+-----+-----+-----+
| employeeID | name | job | departmentID |
+-----+-----+-----+-----+
| 1 | ABC | Lecturer | 10 |
| 3 | ACB | Lecturer | 10 |
| 4 | XYZ | Assistant Professor | 10 |
| 5 | CAB | Lecturer | 15 |
+-----+-----+-----+-----+
4 rows in set (0.007 sec)
```

Command: `select * from employee where job = 'Lecturer';`

Detail: This command selects all the records from employee table whose job title is 'Lecturer'.

Command: `select * from department where departmentID > 15;`

Detail: This command selects those records from department table whose departmentID is greater than 15.

Command: `select * from department where departmentID = 10 or departmentID = 20;`

```

C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> select * from employee where job = 'Lecturer';
+-----+-----+-----+-----+
| employeeID | name | job      | departmentID |
+-----+-----+-----+-----+
|          1 | ABC  | Lecturer |          10  |
|          3 | ACB  | Lecturer |          10  |
|          5 | CAB  | Lecturer |          15  |
+-----+-----+-----+-----+
3 rows in set (0.003 sec)

MariaDB [mohsin_db]> select * from department where departmentID > 15;
+-----+-----+
| departmentID | name                |
+-----+-----+
|          20 | Chemical Engineering |
|          25 | Mining Engineering  |
+-----+-----+
2 rows in set (0.007 sec)

MariaDB [mohsin_db]> select * from department where departmentID = 10 or departmentID =20;
+-----+-----+
| departmentID | name                |
+-----+-----+
|          10 | Computer Systems Engineering |
|          20 | Chemical Engineering  |
+-----+-----+
2 rows in set (0.007 sec)

```

Command: `select` employee.name, department.name
`from` employee, department,
`where` employee.departmentID = department.departmentID;

Detail: There are many departments in an organization and a given department contains many employees. We are interested to find employee name and its respective department name, where an employee works. In employee table, department information is shown by departmentID that is a foreign key. Thus, to find accurate department name, only those records must be listed where departmentID in both table matches. This is done by condition specified after where keyword.

```
C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> select employee.name, department.name
-> from employee, department,
-> where employee.departmentID = department.departmentID;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'where employee.departmentID = department.departmentID' at line 3
MariaDB [mohsin_db]> SELECT employee.name, department.name
-> FROM employee, department
-> WHERE employee.departmentID = department.departmentID;
+-----+-----+
| name | name |
+-----+-----+
| ABC  | Computer Systems Engineering |
| ACB  | Computer Systems Engineering |
| XYZ  | Computer Systems Engineering |
| CAB  | Electrical Engineering       |
+-----+-----+
4 rows in set (0.008 sec)
```

Command: `select employee.name as empName, department.name as deptName`
`from employee, department,`
`where employee.departmentID = department.departmentID;`

Detail: This command shows the use of alias feature of MySQL. The given command is same as the above one except that column names are specified by alias names i.e. employee name is represented by empName and department name is represented by deptName.

```
MariaDB [mohsin_db]> select employee.name as empName, department.name as deptName
-> from employee, department,
-> where employee.departmentID = department.departmentID;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'where employee.departmentID = department.departmentID' at line 3
MariaDB [mohsin_db]> select employee.name as empName, department.name as deptName
-> from employee, department
-> where employee.departmentID = department.departmentID;
+-----+-----+
| empName | deptName |
+-----+-----+
| ABC     | Computer Systems Engineering |
| ACB     | Computer Systems Engineering |
| XYZ     | Computer Systems Engineering |
| CAB     | Electrical Engineering       |
+-----+-----+
4 rows in set (0.001 sec)
```

Command: `delete from department`
`where departmentID=25;`

Detail: This command deletes a record from department table, whose departmentID is 25.

Command: `select * from department;`

Detail: This command selects records from department table where * means to select and show values from all the columns. Note that the records shown don't contain department having 25 as it is deleted in above command.

Command: `select job from employee;`

Detail: This command selects and shows content of 'job' column in employee table.

```
C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> delete from department
-> where departmentID=25;
Query OK, 1 row affected (0.008 sec)

MariaDB [mohsin_db]> select * from department;
+-----+-----+
| departmentID | name                |
+-----+-----+
|          10 | Computer Systems Engineering |
|          15 | Electrical Engineering      |
|          20 | Chemical Engineering        |
+-----+-----+
3 rows in set (0.000 sec)

MariaDB [mohsin_db]> select job from employee;
+-----+
| job                |
+-----+
| Lecturer           |
| Lecturer           |
| Assistant Professor |
| Lecturer           |
+-----+
4 rows in set (0.000 sec)
```

Command: `select distinct job from employee;`

Detail: To select different jobs and not the repeated ones, given command is used. Distinct keyword selects only unique jobs present in job column.

Command: `select count(job) from employee;`

Detail: This command counts the total number of entries in job column of employee table.

Command: `select count(distinct job) from employee;`

Detail: To count the number of unique jobs only, distinct keyword is used with count function. The result of this command is count of unique jobs in job column.

C:\Windows\System32\cmd.exe - mysql -u root

```
MariaDB [mohsin_db]> select distinct job from employee;
+-----+
| job          |
+-----+
| Lecturer    |
| Assistant Professor |
+-----+
2 rows in set (0.007 sec)

MariaDB [mohsin_db]> select count(job) from employee;
+-----+
| count(job) |
+-----+
|          4 |
+-----+
1 row in set (0.005 sec)

MariaDB [mohsin_db]> select count(distinct job) from employee;
+-----+
| count(distinct job) |
+-----+
|                2 |
+-----+
1 row in set (0.000 sec)
```

Command: `select count(*) job,`
`from employee`
`group by job;`

Detail: To determine how many employees are hired for a specific job title, following command is used. It finds the count for all the unique jobs.

Command: `select count(*) job,`
`from employee`
`group by job`
`having count(*)=1;`

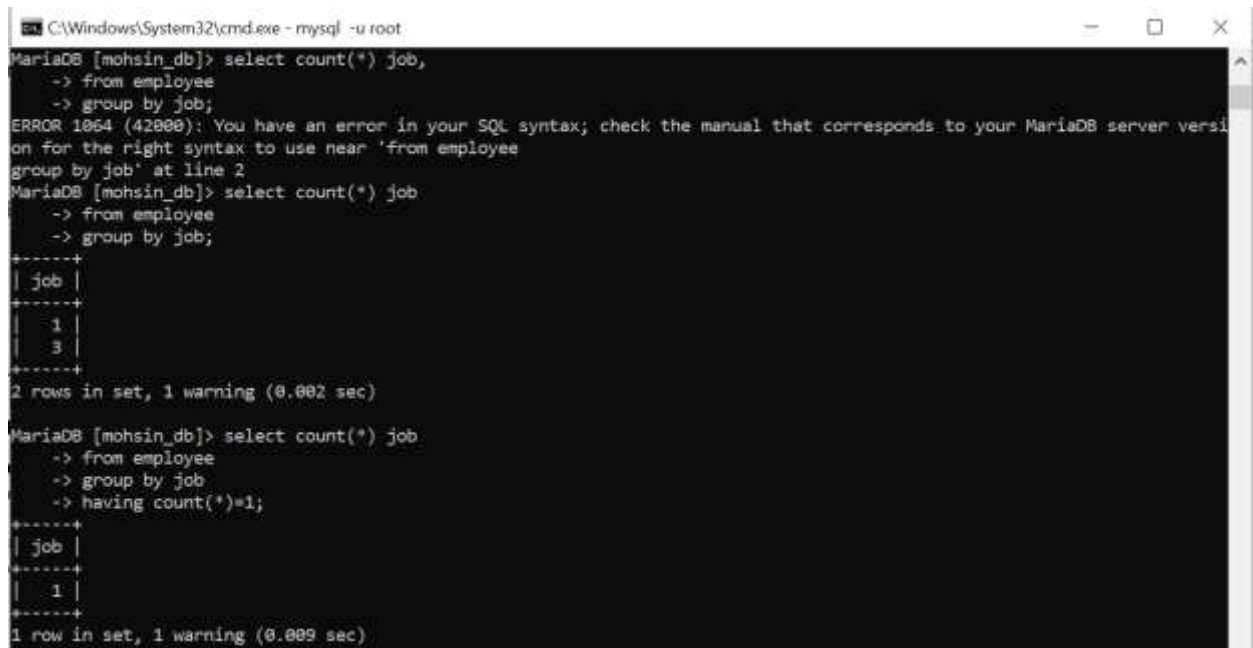
Detail: This command is the same as above one except the 'having' keyword. Here we are interested to find that *job title*, whose count equals 1 (i.e. find a job for which only one employee is hired?). If such job exists whose employee count is one, then it is returning the count; otherwise an empty set is returned.

Command: `select *`

`from employee`

`order by job asc;`

Detail: By default, records are sorted according to primary key in ascending order. But it can be changed. Given command lists all the entries in employee table, such that sorting order of the records is specified by 'job' field.



```
C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> select count(*) job,
-> from employee
-> group by job;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'from employee group by job' at line 2
MariaDB [mohsin_db]> select count(*) job
-> from employee
-> group by job;
+-----+
| job |
+-----+
| 1 |
| 3 |
+-----+
2 rows in set, 1 warning (0.002 sec)

MariaDB [mohsin_db]> select count(*) job
-> from employee
-> group by job
-> having count(*)=1;
+-----+
| job |
+-----+
| 1 |
+-----+
1 row in set, 1 warning (0.009 sec)
```

Command: `select *`
`from employee`
`limit 3;`

Detail: To select specific number of records from a given table, limit keyword can be used. The given command selects and displays three records from employee table

Command: `update employee`
`set job='Lab Engineer'`
`where employeeID=3;`

Detail: This command updates a record from employee table, sets job to 'Lab Engineer' from 'Lecturer' where employeeID is 3.

C:\Windows\System32\cmd.exe - mysql -u root

```
MariaDB [mohsin_db]> select *
-> from employee
-> order by job asc;
```

employeeID	name	job	departmentID
4	XYZ	Assistant Professor	10
1	ABC	Lecturer	10
3	ACB	Lecturer	10
5	CAB	Lecturer	15

```
4 rows in set (0.001 sec)
```



```
MariaDB [mohsin_db]> select *
-> from employee
-> limit 3;
```

employeeID	name	job	departmentID
1	ABC	Lecturer	10
3	ACB	Lecturer	10
4	XYZ	Assistant Professor	10

```
3 rows in set (0.001 sec)
```

Step 1: `alter table employee`
 `drop foreign key employee_ibfk_1;`

Detail: This command drops the foreign key from the employee table.

Step 2: `alter table employee`
 `add foreign key (departmentID) references department(departmentID)`
 `on update cascade;`

Detail: This command applies the foreign key on the employee table with integrity constraint update cascade. This means that if the foreign key is updated in the parent table i.e. department then it'll be reflected in child table i.e. employee. Similarly, other integrity constraints i.e. on update restrict, on update set null, and on update set default can be applied.

Step 3: `update department`
 `set departmentID=12`
 `where departmentID=10;`

Detail: This command updates a record from department table, sets departmentID to 12 from 10 where departmentID is 10.

```
C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> update employee
  -> set job='Lab Engineer'
  -> where employeeID=3;
Query OK, 1 row affected (0.012 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [mohsin_db]> alter table employee
  -> drop foreign key employee_ibfk_1;
Query OK, 0 rows affected (0.010 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [mohsin_db]> alter table employee
  -> add foreign key (departmentID) references department(departmentID)
  -> on update cascade;
Query OK, 4 rows affected (0.065 sec)
Records: 4  Duplicates: 0  Warnings: 0

MariaDB [mohsin_db]> update department
  -> set departmentID=12
  -> where departmentID=10;
Query OK, 1 row affected (0.053 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Step 4: `select * from department;`

Detail: This command lists all the entries in the department table. Note that now the departmentID starts from 12 instead of 10 due to previous command.

Step 5: `select * from employee;`

Detail: This command lists all the entries in the employee table. Note that all the employees of Computer Systems Engineering have foreign key value of 12 instead of 10 due to update cascade integrity constraint.

Command: `alter table employee`
`add foreign key (departmentID) references department(departmentID)`
`on update set null;`

Detail: This command applies the foreign key on the employee table with integrity constraint update set null. This means that if the foreign key is updated in the parent table i.e. department then it'll be set to null in the child table i.e. employee. Step 1, 3, 4, and 5 are the same as above. Only step has been changed.

```
C:\Windows\System32\cmd.exe - mysql -u root
MariaDB [mohsin_db]> select * from department;
+-----+-----+
| departmentID | name                |
+-----+-----+
|          12 | Computer Systems Engineering |
|          15 | Electrical Engineering      |
|          20 | Chemical Engineering        |
+-----+-----+
3 rows in set (0.001 sec)

MariaDB [mohsin_db]> select * from employee;
+-----+-----+-----+-----+
| employeeID | name | job                | departmentID |
+-----+-----+-----+-----+
|          1 | ABC  | Lecturer          |          12 |
|          3 | ACB  | Lab Engineer       |          12 |
|          4 | XYZ  | Assistant Professor |          12 |
|          5 | CAB  | Lecturer           |          15 |
+-----+-----+-----+-----+
4 rows in set (0.023 sec)

MariaDB [mohsin_db]> alter table employee
  -> add foreign key (departmentID) references department(departmentID)
  -> on update set null;
Query OK, 4 rows affected (0.060 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

Home work 1: What is difference between SQL and MySQL? Why is MySQL used? What are its features?

Answer:

Difference Between SQL and MySQL

- **SQL** is a language used to manage databases.
- **MySQL** is a database management system (RDBMS) that uses SQL.

Why is MySQL Used?

- It is fast, reliable, and easy to use.
- It is free and works on different platforms.
- Used for websites, applications, and data storage.

Features of MySQL

1. **Open-source** – Free to use.
2. **Fast & scalable** – Handles large data efficiently.
3. **Secure** – Provides authentication and encryption.
4. **Cross-platform** – Runs on Windows, Linux, and macOS.
5. **Supports multiple users** – Allows many users at once.
6. **Data backup & recovery** – Prevents data loss.

Home work 2: What is database engine? What purpose does it serve? How many types of engines are supported by MySQL? Which database engine is most commonly used and why?

Answer:

Database Engine & Its Purpose

A **database engine** manages data storage, retrieval, and processing in a database.

Types of MySQL Database Engines

1. **InnoDB** (Default)
2. **MyISAM**
3. **Memory**
4. **CSV**
5. **Archive**
6. **Federated**
7. **NDB (Cluster Engine)**

Most Used Engine & Why?

InnoDB is the most used because it supports **transactions, row-level locking, and foreign keys**, ensuring data integrity and performance.

Home work 3: Specify at least fifteen (15) or more different data types supported by MySQL. Provide the description with at least one example.

Answer:

MySQL Data Types with Examples

1. Integer Types

1. TINYINT – Small integer (-128 to 127) → age TINYINT
2. SMALLINT – Medium integer (-32,768 to 32,767) → year SMALLINT
3. MEDIUMINT – Larger integer (-8M to 8M) → views MEDIUMINT
4. INT – Standard integer → salary INT
5. BIGINT – Large integer → population BIGINT

2. Floating & Decimal Types

6. FLOAT – Single-precision float → temperature FLOAT(5,2)
7. DOUBLE – Double-precision float → price DOUBLE(10,2)
8. DECIMAL – Fixed precision → balance DECIMAL(10,2)

3. String Types

9. CHAR – Fixed-length string → gender CHAR(1)
10. VARCHAR – Variable-length string → name VARCHAR(50)
11. TEXT – Large text → description TEXT

4. Date & Time Types

12. DATE – Date (YYYY-MM-DD) → birth_date DATE
13. DATETIME – Date & time → created_at DATETIME
14. TIMESTAMP – Auto-updating timestamp → updated_at TIMESTAMP
15. TIME – Time (HH:MM:SS) → meeting_time TIME