MYSQL

Q) how to create and drop databases in MYSQL?
Ans)To create a new database, you can use the `CREATE DATABASE` statement.
Syntax:- CREATE DATABASE database_name;
Example:- **CREATE DATABASE imad;**

Q)how to drop databases in MYSQL?
Ans)To drop an existing database, you can use the `DROP DATABASE` statement.
Syntax:- DROP DATABASE database_name;
Example:-**DROP DATABASE imad**;

Q)Can we provide heading to database?
Ans)In MySQL, there is no specific command to give a heading to a database. The database name itself serves as an identifier or label for the database. However, you can add comments to provide additional information or a description about the database.

To add a comment to a database, you can use the `COMMENT` clause when creating the database. Here's an example:

```
CREATE DATABASE mydatabase COMMENT 'This is a sample database.';
```

CREATE TABLE

Q)how **create tables** in MYSQL?
Ans)In MySQL, you can provide column headings or column names to tables by defining them when creating the table. Each column in the table is defined with a name and a data type.

Here's an Example of creating a table with column headings:

```
CREATE TABLE table_name (
    column1_name datatype,
    column2_name datatype,
    column3_name datatype,
    ...
);
```

For instance, let's say we want to create a table called "employees" with three columns: "id", "name", and "salary". We can define it as follows:

CREATE TABLE employees (

id INT,

name VARCHAR(50),

salary DECIMAL(10,2));

In this example, the "employees" table has three columns:

"id" (integer), "name" (varchar), and "salary" (decimal).

Once the table is created, you can use these column names to insert, update, or query data from the table. For example:

INSERT INTO employees (id, name, salary) VALUES (1, 'John Doe', 5000.00);

Q)how to check the table employees?

Ans)To check the structure and contents of the table "employees" in MySQL, you can use the `DESCRIBE` statement or the `SHOW COLUMNS` statement. Both statements provide information about the columns in a table.

1>Using DESCRIBE:

The `DESCRIBE` statement provides a concise description of the table structure, including the column names, data types, and any additional attributes.
Syntax:- **DESCRIBE employees;**

mysql> DESCRIBE employees;

```
+--------+--------------+------+-----+---------+-------+
| Field  | Type         | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| id     | int          | YES  |     | NULL    |       |
| name   | varchar(50)  | YES  |     | NULL    |       |
| salary | decimal(10,2) | YES  |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

2>Using SHOW COLUMNS:

3>The `SHOW COLUMNS` statement provides a more detailed view of the table structure, including information like column names, data types, column order, default values, and more.

Syntax:- **SHOW COLUMNS FROM employees;**

R) how to check this [INSERT INTO employees (id, name, salary) VALUES (1, 'John Doe', 5000.00);] in table?

Ans)The SQL query you provided is an example of an `INSERT` statement that inserts a new row into the "employees" table with values for the "id", "name", and "salary" columns.

SELECT * FROM employees;

```
mysql> SELECT * FROM employees;
+------+-------------+-------------+
| id   | name        | salary      |
+------+-------------+-------------+
|  570 | imaduddin | 500000.00 |
|  571 | akbar       | 75000.00  |
+------+-------------+-------------+
2 rows in set (0.00 sec)
```

Q)What is **asterisk (*)** ?

Ans) The **asterisk (*)** in the `SELECT` statement represents **all columns**.

If you only want to see specific columns, you can replace the asterisk with the column names you want to retrieve. For example:

SELECT id, name, salary FROM employees;

Q) how to remove rows in employees table?

Ans)To remove rows from the "employees" table in MySQL, you can use the `DELETE` statement. The `DELETE` statement allows you to specify a condition that determines which rows to delete from the table.

Here's an example of how to use the `DELETE` statement to remove rows from the "employees" table:

DELETE FROM employees WHERE condition;

Example:-

DELETE FROM employees WHERE id = 1;

Output:-

```
mysql> select * from employees;
+------+-----------+-----------+
| id       | name       | salary      |
+------+-----------+-----------+
|    1     | John Doe    |   5000.00 |
|    1     | John Doe    |   5000.00 |
|    1     | John Doe    |   5000.00 |
|    1     | John Doe    |   5000.00 |
|  570   | imaduddin | 50000.00 |
|  571   | akbar     | 750000.00   |
+------+-----------+-----------+
6 rows in set (0.00 sec)
mysql> DELETE FROM employees WHERE id = 1;
Query OK, 4 rows affected (0.00 sec)
mysql> select * from employees;
+------+-----------+-----------+
| id   | name      | salary    |
+------+-----------+-----------+
|  570 | imaduddin | 500000.00 |
|  571 | akbar         | 750000.00 |
```

TABLE OPERATIONS

Q)Whats are the operations that can be performed in employees table?
Ans)**1>Inserting Rows:**
2>Updating Rows:
3>Deleting Rows:
4>Querying Rows:
5>Filtering Rows:
6>Sorting Rows:
7>Aggregating Data:
8>Joining Tables:

Q)How to **Insert Rows** in table?
Ans)To **insert rows** into the "employees" table in MySQL, you can use the `INSERT INTO` statement. Here's the general syntax:

Example:-
INSERT INTO employees (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

INSERT INTO employees (id, name, salary) VALUES (1, 'John Doe', 5000.00);

Q)How to Insert **MULTIPLE** Rows in table?
Ans)**INSERT INTO employees (id, name, salary) VALUES**
(1, 'John Doe', 5000.00),
(2, 'Jane Smith', 6000.00),
(3, 'Michael Johnson', 5500.00);

Q)**Updating Rows** in employees table?
Ans)Ans)To update rows in the "employees" table in MySQL, you can use the `UPDATE` statement.
syntax:
UPDATE employees SET column1 = value1, column2 = value2, ... WHERE condition;
Example:-
UPDATE employees SET name='mohammed',
salary=15000 where id=570;

```
mysql>  UPDATE employees SET name='mohammed',salary=15000
where id=570;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from employees;
+------+----------------+-----------+
| id   | name           | salary    |
+------+----------------+-----------+
|  570 | mohammed       |  15000.00 |
|  571 | akbar          | 750000.00 |
|    1 | John Doe       |   5000.00 |
|    2 | Jane Smith     |   6000.00 |
|    3 | Michael Johnson |   5500.00 |
+------+----------------+-----------+
5 rows in set (0.00 sec)
```

Q)How to update multiple rows simultaneously using a comma-separated list of conditions in the WHERE clause. Each row needs its own separate UPDATE statement.
Ans)1> We can use the `CASE` statement in the combination with the `UPDATE` statement.
2>In this example, the `CASE` statement is used to check the value of the "name" column for each row.
3>Based on the name, the corresponding salary value is assigned.
4>The `UPDATE` statement then updates the "salary" column for all the matching rows simultaneously.
5>Make sure to adjust the column names, values, and conditions based on your specific table structure and data requirements.
6>Executing this statement will update the salary for all the employees whose names match the specified conditions in a single query.
Here's an **example**:

```
mysql> update employees
    -> set salary=case
    -> when name='mohammed' then 789652
    -> when name='akbar' then 60000
    -> when name='John Doe' then 7000
    -> when name='Jane Smith' then 120000
    -> else salary end;
```

```
mysql> select * from employees;
+------+-----------------+-----------+
| id   | name            | salary    |
+------+-----------------+-----------+
|  570 | mohammed        | 789652.00 |
|  571 | akbar           |  60000.00 |
|  572 | John Doe        |   7000.00 |
|  573 | Jane Smith      | 120000.00 |
|  574 | Michael Johnson |   5500.00 |
+------+-----------------+-----------+
5 rows in set (0.00 sec)
```