----------LECTURE- 5----------

1 DML – REPLACE (Overwrite)

✅ What is REPLACE?

REPLACE is a DML command in MySQL that **inserts a new row**,
BUT **if the same PRIMARY KEY or UNIQUE value already exists**,
then MySQL will:

1. **Delete the old row**
2. **Insert the new row**
   So it behaves like **INSERT + DELETE + INSERT**.

☆ Syntax

```
REPLACE INTO table_name (col1, col2, col3)
VALUES (value1, value2, value3);
```

☆ Example

Suppose you have a table:

```
CREATE TABLE users (
   id INT PRIMARY KEY,
   email VARCHAR(50)
);
```

Table contains:

id	email
1	a@gmail.com

Now:

```
REPLACE INTO users VALUES (1, 'new@gmail.com');
```

What happens?

- MySQL deletes row id = 1
- Inserts a new row id = 1 with updated email

---

2 CALL → Execute Stored Procedure (Inside DML)

✓ What is CALL?

CALL is used to **run a stored procedure**.
A stored procedure is like a **saved mini-program** inside MySQL.

☆ **Example Syntax**

```
CALL procedure_name(parameters);
```

☆ **Example**

```
CALL getTotalSalary();
```

This will execute the procedure stored in the DB.

---

3 LOAD DATA → Import Large Dataset

This is used to **import bulk data** (very large files) very fast.

☆ **Syntax**

```
LOAD DATA INFILE '/path/yourfile.csv'
INTO TABLE table_name
```

Where do we use it?

- Importing Lakhs / Crores rows
- Faster than INSERT

---

4 BLOB Types (Binary Large Objects)

Used to store **images, videos, PDFs, audio**, any binary data.

Types of BLOB:

Type	Size
TINYBLOB	Up to 255 bytes
MEDIUMBLOB	Up to 16 MB
BLOB	Up to 64 KB
LONGBLOB	Up to 4 GB

Use-case examples:

- Profile pictures

- PDF documents
- Audio/video files

---

5 Data Export / Import from UI (MySQL Workbench)

This means:

- Export → download your table or DB as .sql file
- Import → upload .sql file into MySQL
-

You do this from:

Server → Data Export / Data Import in Workbench.
(No SQL syntax here, it's UI based.)

---

6 CREATE TABLE … LIKE (Copy Table Structure)

✓ What it does:
Copies **ONLY THE STRUCTURE**, not the data.

☆ Syntax

```
CREATE TABLE new_table LIKE old_table;
```

Example:

```
CREATE TABLE emp_log LIKE employee;
```

Creates emp_log with same columns, datatypes, constraints.

---

7 CREATE TABLE … AS SELECT (Copy Structure + Data)

✓ What it does:

Copies **structure + data** into a new table.

☆ Syntax

```
CREATE TABLE new_table AS
SELECT * FROM old_table;
```

Example:

```
CREATE TABLE emp_copy AS
SELECT * FROM employee;
```

You can also filter with WHERE:

```
CREATE TABLE emp_gt_20000 AS
SELECT * FROM employee WHERE salary > 20000;
```

---

8 INSERT INTO table SELECT * FROM another_table

Used to **copy data only** from one table into existing table.

☆ Syntax

```
INSERT INTO table1
SELECT * FROM table2;
```

Example:

```
INSERT INTO emp_log
SELECT * FROM employee;
```

- emp_log must already exist
- Columns must match

---

9 INSERT IGNORE (Avoid Errors)

✓ What it does:

INSERT IGNORE prevents errors when inserting:

- duplicate primary key
- duplicate unique value
- invalid values
-

Instead of error → **row is skipped**.

☆ Syntax

```
INSERT IGNORE INTO users VALUES (1, 'Rahul Kumar', 'Noida');
```

If id=1 already exists → it ignores the row and continues.

---

10 TRIGGERS

Triggers = automatic actions performed by MySQL **when something happens** to a table.

Examples:

- When you INSERT → log table updates
- When you DELETE → archive data
- When you UPDATE → track old values

☆ SHOW TRIGGERS

```
SHOW TRIGGERS;
```

☆ DROP TRIGGER

```
DROP TRIGGER trigger_name;
```

☆ Why do we drop triggers?

- If trigger causes problems
- If it inserts wrong logs
- If you change table structure
- If performance is slow
- If trigger is no longer needed

---

11 Updating Multiple Values at Once

Syntax

```
UPDATE employee
SET salary = salary + 5000,
   city = 'Delhi'
WHERE id = 3;
```

---

12 UPDATE With JOIN

Used for updating one table using another table.

☆ **Example**

```
UPDATE orders o
JOIN customers c ON o.cid = c.cid
SET o.city = c.city;
```

---

13 DELETE With JOIN

☆ **Example**

```
DELETE s
FROM student s
JOIN fees f ON s.id = f.sid
WHERE f.amount = 0;
```

---

14 Copy table from one DB to another DB

Syntax

```
CREATE TABLE new_db.table1 LIKE old_db.table1;
Then copy data:
INSERT INTO new_db.table1
SELECT * FROM old_db.table1;
```