SQL DML and Transactions – Simple Guide

# 1) What is UPDATE in SQL?

UPDATE is used when you want to change existing data in your database.

How to use UPDATE?

- You choose the table where data needs to be changed.

- Specify which column you want to update.

- Set the new value.

- Add a condition to update only the rows you want.

Example:

UPDATE EMP  
SET SAL = SAL \* 110 / 100  
WHERE DEPTNO = 10;

What does this mean?

- EMP is the employee table.

- SAL is the salary column.

- You multiply salary by 110%, which is a 10% increase.

- You only update employees in department 10 (DEPTNO = 10).

## Some Useful Examples

Q1: How to update commission to 1000 if it is currently empty (NULL)?

UPDATE EMP  
SET COMM = 1000  
WHERE COMM IS NULL;

Q2: How to increase commission by 200 for employees working in New York?

UPDATE EMP  
SET COMM = COMM + 200  
WHERE DEPTNO IN (  
 SELECT DEPTNO FROM DEPT WHERE LOC = 'NEW YORK'  
);

# 2) What is DELETE in SQL?

DELETE removes data from a table.

How to use DELETE?

- Choose the table.

- Give a condition to delete only specific rows.

Example: Delete all employees working in Chicago:

DELETE FROM EMP  
WHERE DEPTNO = (  
 SELECT DEPTNO FROM DEPT WHERE LOC = 'CHICAGO'  
);

# 3) Transactions in SQL

Think of transactions as a group of SQL commands you want to run together.

Why?

- To make sure either all changes happen, or none happen.

- Helps keep your data safe and consistent.

## Important commands:

- COMMIT: Saves all changes made in the transaction permanently.

- ROLLBACK: Undo changes made since last commit.

- SAVEPOINT: Set a checkpoint inside a transaction. You can rollback to this point if needed, instead of undoing everything.

Example of SAVEPOINT and ROLLBACK:

SAVEPOINT s1;  
-- Do some updates here  
ROLLBACK TO s1; -- Undo changes after savepoint s1

# 4) CASE Statement (Like IF in SQL)

CASE helps you choose what value to show based on some conditions.

## Example 1:

SELECT ENAME, SAL,  
 CASE   
 WHEN TO\_CHAR(HIREDATE, 'D') IN ('2','4','6') THEN  
 '\*' || SUBSTR(ENAME, 2, LENGTH(ENAME) - 1) || '\*'  
 WHEN TO\_CHAR(HIREDATE, 'D') IN ('3','5','7') THEN  
 SUBSTR(ENAME, 1, 1) || '\*' ||  
 SUBSTR(ENAME, 3, LENGTH(ENAME) - 2) || '\*' ||  
 SUBSTR(ENAME, LENGTH(ENAME), 1)  
 ELSE 'NULL'  
 END AS MODIFIED\_NAME  
FROM EMP;

Explanation:

- It checks the day of the week employee was hired.

- For some days, it adds \* before and after their name (with some changes).

- For other days, it hides parts of the name with \*.

## Example 2: Mark salary status

SELECT ENAME, SAL,  
 CASE  
 WHEN SAL < 2000 THEN 'LOW SALARY'  
 WHEN SAL >= 2000 THEN 'HIGH SALARY'  
 ELSE 'NULL'  
 END AS SALARY\_STATUS  
FROM EMP;

## Example 3: Replace first or last letter of name with \* based on name length

SELECT ENAME,  
 CASE  
 WHEN MOD(LENGTH(ENAME), 2) = 0 THEN REPLACE(ENAME, SUBSTR(ENAME, 1, 1), '\*')  
 WHEN MOD(LENGTH(ENAME), 2) = 1 THEN REPLACE(ENAME, SUBSTR(ENAME, -1), '\*')  
 ELSE 'NULL'  
 END AS MODIFIED\_NAME  
FROM EMP;

Explanation:

- If the name length is even, replace first letter with \*.

- If odd, replace last letter with \*.

# Summary

| Command | What it does |  
|--------------|-------------------------------------------|  
| UPDATE | Change existing data |  
| DELETE | Remove data |  
| COMMIT | Save changes permanently |  
| ROLLBACK | Undo changes before commit |  
| SAVEPOINT | Set a point to rollback to during a transaction |  
| CASE | Perform conditional logic in queries |