

- USER → needs a platform to interact with DATABASE → For **ORACLE** that **SQL** or **PL/SQL** can be used for that and FOR **MS SQL SERVER** **TSQL** is needed
Example, USER and LAPTOP → OS

Type of SQL Statement:

1. TCL → Transaction Control → COMMIT, ROLLBACK, SAVE POINT → Used to control the transaction process
2. DML → INSERT, UPDATE, DELETE
3. DCL → GRANT, REVOKE → Used for giving authorization (Assigning or revoking privileges)
4. DDL → CREATE, ALTER, DROP

After Using **Commit**, you won't be able to go back.

Queries must return only one row and INTO clause is required.

Naming Ambiguities:

Happens when DB Column name takes precedence over local variable name.

Naming Conventions

- Use a naming convention to avoid ambiguity in the WHERE clause.
- Avoid using database column names as identifiers.
- Syntax errors can arise because PL/SQL checks the database first for a column in the table.
- The names of local variables and formal parameters take precedence over the names of database *tables*.
- The names of database table *columns* take precedence over the names of local variables.

IMPORT and EXPORT → DMP (Data Pump) and SQL * LOADER is used for that.

Inserting Data:

Updating Data:

Deleting Data:

Merging Data: → aka upsert (UPDATE + INSERT)

SQL Cursor:

It's a pointer to a private memory area allocated by the Oracle Server. It's a Context Area in a Memory. Required for processing of SQL statements.

1. Implicit: Created and managed internally by the Oracle Server to process SQL statements. We don't have to access it.
2. Explicit: Declared explicitly by the programmers to retrieve multiple rows from a database table
 - a. Phases of Explicit Cursor:
 - i. Declare:
 - ii. OPEN → Allocating the memory
 - iii. FETCH:
 - iv. CLOSE → Deallocation of Memory

SQL Cursor Attributes for Implicit Cursors

Using SQL cursor attributes, you can test the outcome of your SQL statements.

SQL%FOUND	Boolean attribute that evaluates to TRUE if the most recent SQL statement affected at least one row
SQL%NOTFOUND	Boolean attribute that evaluates to TRUE if the most recent SQL statement did not affect even one row
SQL%ROWCOUNT	An integer value that represents the number of rows affected by the most recent SQL statement

FOUND and NOTFOUND → Boolean attributes

ROWCOUNT → Integer value

```

BEGIN
  INSERT INTO employees
    (employee_id, first_name, last_name, email,
     hire_date, job_id, salary)
    VALUES (employees_seq.NEXTVAL, 'Ruth', 'Cores',
            'RCORES', CURRENT_DATE, 'AD_ASST', 4000);
END;
/

```

```

DECLARE
  sal_increase    employees.salary%TYPE := 800;
BEGIN
  UPDATE          employees
  SET              salary = salary + sal_increase
  WHERE            job_id = 'ST_CLERK';
END;
/

```

```

DECLARE
  deptno    employees.department_id%TYPE := 10;
BEGIN
  DELETE FROM employees
  WHERE department_id = deptno;
END;
/

```

If the WHERE clause is not used all the rows in table can be removed if there are no integrity constraints.

```

DECLARE
  v_rows_deleted VARCHAR2(30)
  v_empno employees.employee_id%TYPE := 176;
BEGIN
  DELETE FROM employees
  WHERE employee_id = v_empno;
  v_rows_deleted := (SQL%ROWCOUNT ||
                    ' row deleted. ');
  DBMS_OUTPUT.PUT_LINE (v_rows_deleted);
END;

```