DDL (Data Definition Language)

- **CREATE** – Used to create a new database object (like a table, view, or database).
- **ALTER** – Modifies an existing database object, such as adding or deleting columns in a table.
- **DROP** – Deletes an existing database object permanently.
- **TRUNCATE** – Removes all rows from a table, but keeps the table structure intact.
- **RENAME** – Changes the name of an existing database object.

CREATE TABLE SYNTAX

Create table <table_name> (col1 datatype1, col2 datatype2 …coln datatypen);

Ex:

SQL> create table student (no number (2), name varchar (10), marks number (3));

CREATE WITH SELECT

We can create a table using existing table [along with data].

Syntax:

Create table <new_table_name> [col1, col2, col3 ... coln] as select * from <old_table_name>;

CONSTRAINTS

Constraints are categorized as follows.

Domain integrity constraints

- Not null
- Check

Entity integrity constraints

- Unique
- Primary key

Referential integrity constraints

- Foreign key

NOT NULL AND CHECK

Not Null

SQL> create table student(no number(2) not null, name varchar(10), marks number(3));

CHECK

This is used to insert the values based on specified condition.

We can add this co nstraint in all three levels.

SQL> create table student(no number(2) , name varchar(10), marks number(3) check

(marks > 300));

We can add constraints in three ways.

Constraints are always attached to a column not a table.

Column level -- along with the column definition

Table level -- after the table definition

Alter level -- using alter command

All three Levels

COLUMN LEVEL

SQL> create table student(no number(2) , name varchar(10), marks number(3) check

(marks > 300));

TABLE LEVEL

SQL> create table student(no number(2) , name varchar(10), marks number(3), check

(marks > 300));

ALTER LEVEL

SQL> alter table student add check(marks>300);

UNIQUE

This is used to avoid duplicates but it allow nulls.

We can add this constraint in all three levels.

COLUMN LEVEL

SQL> create table student(no number(2) unique, name varchar(10), marks number(3));

TABLE LEVEL

SQL> create table student(no number(2) , name varchar(10), marks number(3),

unique(no));

ALTER LEVEL

SQL> alter table student add unique(no);

PRIMARY KEY

This is used to avoid duplicates and nulls. This will work as combination of unique and not null.

Primary key always attached to the parent table.

We can add this constraint in all three levels.

PRIMARY KEY

COLUMN LEVEL

SQL> create table student(no number(2) primary key, name varchar(10), marks number(3));

marks number(3));

TABLE LEVEL

SQL> create table student(no number(2) , name varchar(10), marks number(3),

primary key(no));

ALTER LEVEL

SQL> alter table student add primary key(no);

FOREIGN KEY

This is used to reference the parent table primary key column which allows duplicates.

Foreign key always attached to the child table.

We can add this constraint in table and alter levels only.

TABLE LEVEL

SQL> create table emp(empno number(2), ename varchar(10), deptno number(2),

primary key(empno), foreign key(deptno) references dept(deptno));

TABLE LEVEL

SQL> create table emp(empno number(2), ename varchar(10), deptno number(2),

primary key(empno), foreign key(deptno) references dept(deptno));

USING ON DELTE CASCADE

By using this clause you can remove the parent record even it childs exists.

Because when ever you remove parent record oracle automatically removes all its dependent records from child table, if this clause is present while creating foreign key constraint.

Ex:

TABLE LEVEL

SQL> create table emp(empno number(2), ename varchar(10), deptno number(2),

primary key(empno), foreign key(deptno) references dept(deptno) on delete cascade);

USING ALTER

- **ADDING COLUMN** :alter table <table_name> add <col datatype>;
- **REMOVING COLUMN** :alter table <table_name> drop <col datatype>;
- **INCREASING OR DECREASING PRECISION OF A COLUMN:**

  alter table <table_name> modify <col datatype>;
- **DROPPING UNUSED COLUMNS**: alter table <table_name> drop unused columns;
- **RENAMING COLUMN** : alter table <table_name> rename column <old_col_name> to <new_col_name>;

USING TRUNCATE

This can be used to delete the entire table data permanently.

truncate table <table_name>;

USING DROP

This will be used to drop the database object.

Drop table <table_name>;

USING RENAME

This will be used to rename the database object.

rename <old_table_name> to <new_table_name>;

DML (Data Manipulation Language)

- **INSERT** – Adds new records (rows) into a table.
- **UPDATE** – Modifies existing records in a table.
- **DELETE** – Removes existing records from a table.

USING INSERT

INSERT WITH SELECT

Insert into <table1> select * from <table2>;

COLUMN ALIASES

Select <orginal_col> <alias_name> from <table_name>;

By value

INSERT INTO **student** (Name, **age**, **marks**)

VALUES ('Rahul', 30, 90);

USING INSERT on Date Column

```
CREATE TABLE student (
    student_id   NUMBER PRIMARY KEY,
    name         VARCHAR2(50),
    dob          DATE,
    gender       VARCHAR2(10),
    class        VARCHAR2(20)
);

INSERT INTO student (student_id, name, dob, gender, class)
VALUES (1, 'Alice Johnson', TO_DATE('2005-03-15', 'YYYY-MM-DD'), 'Female', '10-A');
```

UPDATE

SQL> update student s set s.address.city = 'bombay' where s.address.hno = 333;

DELETE

```
delete student s where s.address.hno = 111;
```