Create Table

```
CREATE TABLE table_name (
        colown_name1  datatype(size), constraint,
        coloumn_name2     "              "  ,
        );
```

datatype → (number, text, date)

constraint → jo data store karna chahte ho uspe koi constraint hona chahiye ya nhi

eg:-

```
CREATE TABLE student (
        student_id  NUMBER  PRIMARY KEY,
        age         NUMBER  NOT NULL,
        name        VARCHAR2(30)
```

running it :- (Caisa kuch basically banega)

student_id	age	name

SQL Data Types

Data type	Description	Example
(a) NUMBER(p)	Numeric value (+/-)	NUMBER(5) → upto 5 digit
(b) VARCHAR2(n)	Variable-length string	VARCHAR2(30)
(c) CHAR(n)	Fixed-length string	CHAR(10)
(d) DATE	Date values	'2025-10-29'
(e) FLOAT	Floating-Point	FLOAT(5)

(f) Difference b/w CHAR & VARCHAR2 is that if both are declared w/ a size say 10. but only 8 is used then 2 spaces are wasted in CHAR but VARCHAR resizes itself

Type of SQL commands

(i) DDL (Data Definition Language)

used to create, alter, delete, rename

(ii) DML (Data Manipulation Language)

used to insert, modify or delete actual data in tables

(iii) DQL (Data Query Language)

used to fetch info from one or more tables

(iv) DCL (Data control language)

used to give or restrict access to users

(v) TCL (Transcation control language)

used to control changes made by DML command

Table Related Queries

1) CREATE TABLE - discussed.

2) SELECT * FROM table_name :- table ka data dekne ke liye

3) INSERT INTO Table_name :- Data insert karne ke liye
(colname1, colname2)
VALUES
(col1_val1, col2_val1)
(col1_val2, col2_val2)

4) DELETE FROM table_name : To delete a entry from a table.
WHERE condition

KEYS

Primary Key

It is coloumn or set of coloum that uniquely identifies each row (a unique id).
There is only 1 PK & it should be unique and can never be null .

Foriengn key

A coloumn in a table that refers to primary key in & of another table.
(table can multiple FK , It can have duplicate & null values.

CONSTRAINTS

constrains means rules for data in table.

(a) NOT NULL :- coloums w/ NOT NULL cont have null value

(b) UNIQUE :- all values in this coloumn most be unique.

(c) PRIMARY KEY :- make colown unique .

(d) FOREIGN KEY : prevent actions that would destroy links b/w tables.

ex:-

```
CREATE TABLE temp(
    cust_id NUMBER,
    FOREIGN KEY (cust_id) references customer(id)
```

(e) DEFAULT : sets the default value of colown

```
salary NUMBER DEFAULT 25000
```

(Har employee ka salary 25000 hee rahegaa agar kuch alag se enter na karo toh.

(f) CHECK : to limit value allowed in colown

```
CONSTRAINT CHECK (age>=18 AND
                  city="Delhi")
```

OR

```
age NUMBER(2) CHECK(age >= 18)
```

SELECT

- Select → show me this info

(BB) • Basic syntax: (specific colowns)

SELECT coloum1, coloum2 ...

FROM table_name;

- TO get everything from table :-

SELECT * FROM table_name

- TO get only non-duplicate from a colown

SELECT DISTINCT colo_name FROM table_nam

WHERE

TO define some condition, used w/ SELECT

- SELECT col1, col2 FROM table_name

WHERE condition

- In the condition we use different operator to define the condition

(i) Arthemetic :- +, −, *, /, %

(ii) comparison : =, !=, >, <=, <, >=

(iii) logical :- AND, OR, NOT, IN, BETWEEN, ALL

BROWSE OPERATORS

LIMIT

set a upper limit on a number of rows to be returned.

eg) SELECT * FROM student LIMIT 3;

eg) SELECT col1, col2 FROM table_name
LIMIT number;

ORDER BY

TO sort in ascending order (ASC) or decending (DESC) order

eg) SELECT * FROM student.
ORDER BY city ASC;

eg) SELECT * FROM student
ORDER BY marks DESC
LIMIT 3;
(gives top 3 scores)

GROUP BY

group rows that have same value into summary rows.

used with functions (MAX(), MIN(), AVG(), COUNT())

ex:- Count no. of student in each city

```
SELECT city, count(name)
FROM STUDENT GROUP BY city;
```

HAVING CLAUSE

similar to where i.e applies condition on rows

generally used when applying condition after grouping

eg:- no. of student in each city where max marks cross 90.

```
SELECT city, count(name) city
FROM student
max(marks) > 90;
HAVING
```

General order of commands

SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY

UPDATE

```
UPDATE table_name
SET col1 = val1, col2 = val2
WHERE condition;
```

```
UPDATE Student
SET grade = 'A+'
WHERE grade = 'A'
```

Now jiska jiska A hai wo A+ hojaega

ALTER

(a) add a new column

TABLE

ALTER tablename

ADD colname datatype constrain

(b) Rename a ~~table~~ column

ALTE TABLE table_name

RENAME COLUMN old_name TO new_name

(c) change datatype / size

ALTER TABLE table_name

MODIFY (col_name, new_datatype);

TRUNCATE

used to remove all rows from a table

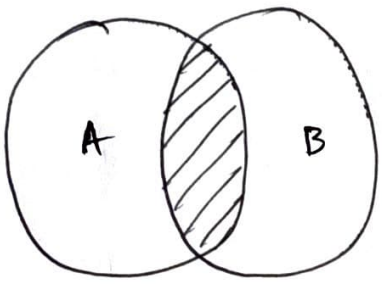
quickly

TRUNCATE TABLE table_name.

JOIN

- Join is used to combine rows from two or more table based on related coloumn, like emp_id in salary table & attendendance table.
- Foreign key usage not compulsory
- Types of Join

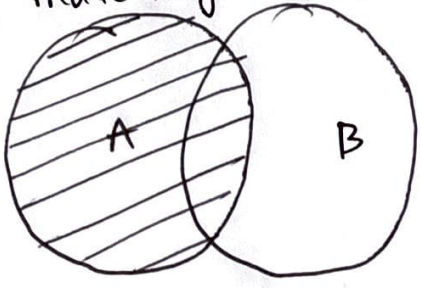
(i) Inner Join : Shows rows common in both A & B table.



(ii) Outer Join

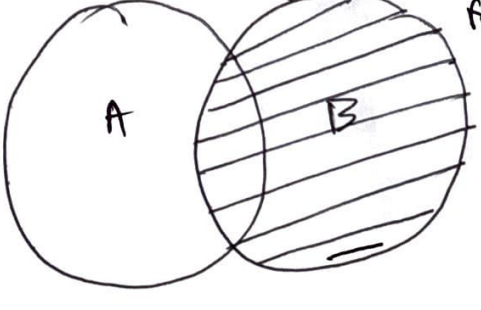
(a) Left Join

All rows from left but only matching ones from right

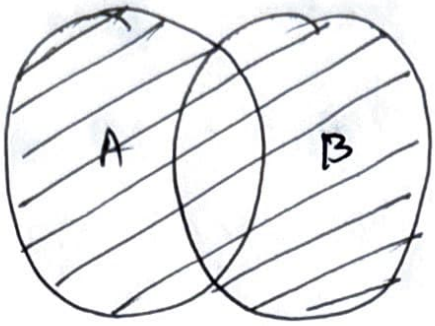


(b) Right Join

All rows from right but only matching from left



(C) Full Join : All rows from Both tables



If we 2 have tables emp & salary table.

1- emp-table has emp id & emp-name

2- ~~emp~~ salary table has emp-id & salary ~~empname~~

Inner Join : ~~All~~ emp id present in both tables

Left Join :- All emp table rows even though not present in salary table; there salary will shown NULL them

Right Join :- All ~~emp~~ salary table rows ~~even~~ if emp is in salary but not in emp-table will have name NULL

Full Join :- All rows from both table

Empty spaces filled w/ NULL

SELF JOIN

- A table ~~that~~ is join w/ itself.
- We creat one copy as table 1 (A) & other table (B) - even though both are same.
- Use when you want to find relationship b/w rows of same table.
- eg:-

emp_id	emp_name	manager_id
1	Alice	NULL
2	Bob	1
3	Charlie	1
4	David	2

we want to show each employee name with their manager name

```
SELECT a.emp_name, ~~Employee~~
       b.emp_name As Manager
FROM EMPLOYEE as a
JOIN EMPLOYEE as b
ON a.~~id~~ empid = b.manager_id
```

UNION

Used to combine 2 or more select statement into single result.

JOIN combines colooms (side by side) but UNION combine rows (top to bottom)

eg:-

emp - A

id	name
1	Alice
2	Bob
3	Charlie

emp - B

id	name
4	David
5	Eve
2	Bob

```
SELECT id, name
FROM emp-A
UNION
SELECT id, name
FROM emp_B
```

id	name
1	Alice
2	Bob
3	Charlie
4	David
5	Eve

Duplicates are removed.
(if duplicates needed use UNION ALL)

for UNION

both ~~table~~ ~~must~~ selected must equal columns
~~Column~~
table

same data type for ~~@~~ select columns