5. JOINS

5.1 AS Statement

AS allows us to rename columns or table selections with an alias

Eg:-

```
SELECT payment-id AS my_payment_coltumn.
FROM payment;
```

It return paymat-id coltumn as my-payment-column.

```
SELECT cust_id, SUM(amount) AS total_spent
FROM payment
GROUP BY cust_id;
```

Convert this column name as

this one.

5.2

JOINS

INNER JOINS

There are several kinds of joins including INNER JOIN, OUTER JOIN and self-join.

- Suppose you want to get data from two tables named A & B.

- The B table has the fka field that relates to the Primary key of A table

→ To get data from both tables, you ~~$~~ use the INNER JOIN Clause in the SELECT statement as follows:

```
SELECT A.pka, A.c1, B.pkb, B.c2.
FROM A
INNER JOIN B ON A.pka = B.fka;
```

→ First, you specify the column in both tables from which you want to select data in the SELECT clause.

→ Second, you specify the main table i.e., A in the FROM clause.

→ Third, you specify the table that the main table joins to i.e., B in the INNER JOIN clause. In addition, you put a join condition after the ON keyword i.e., A.pka = B.fka

```
SELECT A.pka, A.C1, B.pkb, B.C2
FROM A
INNER JOIN B ON A.pka = B.fka;
```

→ In case of name of the table is long, you can use a table alias e.g., tbl and refer to the column as tbl.column_name

Eg:-
```
SELECT customer.cust_id, first_name, last_name, email,
       amount, payment_data
FROM customer.
INNER JOIN payment ON payment.cust_id
                      = customer.cust_id
```

```
ORDER BY Customer.Cust_id ;

WHERE Customer.Cust_id = 2; → only
```

Eg :- 15.3 :- More Eg

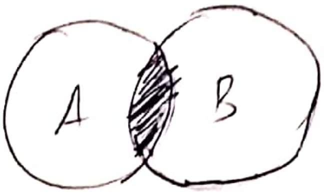
```
SELECT payment_id , amount, first_name, last_name
FROM payment
INNER JOIN Staff ON payment.staff_id = staff.staff_id;
```

```
                  COUNT (Title)
→ SELECT title, FROM inventory
INNER JOIN film ON inventory.film_id = film.film_id ,
WHERE store_id = 1.
GROUP By title;
ORDER By tittle ;
```

```
SELECT film.title, language.name AS movie_language.
FROM film
INNER JOIN language.ON language.language_id = film.language_ic
```

5.4 - overview of join types

5.4.1 INNER JOIN :- Produces only the set of Records that match in both Table A & table B.



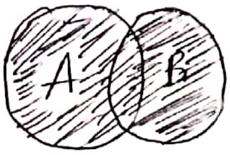
Original tables

id	Name	id	Name
1	Pirate	1	Rutabaga
2	Monkey	2	Pirate
3	NINJA	3	Darth Vader
4	Spaghetti	4	Ninja

```
SELECT * FROM table A
INNER JOIN Table B
ON Table A.name = Table B.name
```

id	name	id	name
1	Pirate	2	Pirate
3	Ninja	4	Ninja

5.4.2 Full OUTER JOIN :- Full outer join produces the set of all records in Table A & table B, with matching records from both sides where available. If there is no match, the missing side will Contain null.



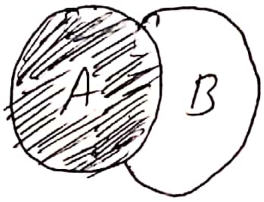
Original Tables

id	name	id	name
1	Pirate	1	Rutabaga
2	Monkey.	2	Pirate
3	NiNja	3	Darth Vader
4	Spaghetti	4	NiNja

```
SELECT * FROM Table A.
FULL OUTER JOIN Table B.
ON Table A.name = TableB.name.
```

id	name	id	name
1	Pirate	2	Pirate
2	Monkey	null	null
3	NiNja	4	NiNja
4	Spaghetti	null	null
null	null	1	Rutabaga
null	null	3	Darth Vader.

5.4.3 LEFT OUTER JOIN :- Produces a Complete set of records from Table A, with the matching records (where available) in table B. If there is no match, the right side will contain null.



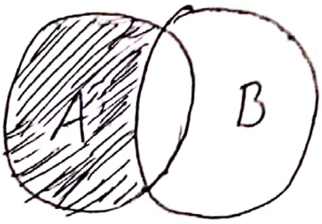
original tables

id	name	id	name
1	Pirate	1	Rutabaga
2	Monkey	2	Pirate
3	Ninja	3	Darth Vader.
4	Spaghetti	4	Ninja

```
SELECT * FROM Table A
LEFT OUTER JOIN Table B.
ON TableA.name = TableB.name
```

id	name	id	name
1	Pirate	2	Pirate
2	Monkay	null	null
3	Ninja	4	Ninja
4	Spaghetti	null	null

5.4.4 LEFT OUTER JOIN :- produces the set of records only in table A, but not in table B, we perform the same left outer join, then exclude the records we don't want from the right side via a WHERE clause.



Original Tables

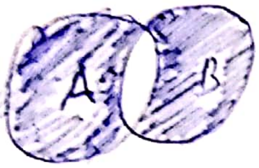
id	name	id	name
1	Pirate	1	Rutabaga
2	monkey.	2	Pirate
3	Ninja	3	Darth Vader
4	Spaghetti	4	Ninja

```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableB.id IS NULL
```

id	name	id	name
2	Monkey	null	null
4	Spaghetti	null	null

5.4.5 Full OUTER JOIN with WHERE

Produces the set of records unique to Table A & Table B, we perform the same full outer join, then exclude the records we don't want from both sides via a where clause.



Original tables

id	name	id	name
1	Pirate	1	Rutabaga
2	Monkey	2	Pirate
3	NiNja	3	Darth Vader
4	Spaghetti	4	NiNja

```
SELECT * FROM TableA
Full OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableA.id IS null
 OR TableB.id IS null
```

id	name	id	name
2	Monkey	null	null
4	Spaghetti	null	null
null	null	1	Rutabega
null	null	3	Darth vador

5.4.6 LEFT OUTER JOIN = Egs

```
SELECT film.film_id., film.title, inventory_id
FROM film.
LEFT OUTER JOIN inventory ON inventory.film_id=film.film_id;
WHERE inventory.film_id IS NULL;
ORDER BY film.film_id;
```

5.5 UNION

The Union operator Combines result sets of two or more SELECT statements into a single result set.

→ Syntax

```
SELECT Column_1, Column_2
FROM tbl_name_1
UNION
SELECT Column_1, Column_2
FROM Tbl_name_2
```

Rules to be applied to of them

① Both queries must return the same number of columns.

② The corresponding columns in the queries must have compatible datatypes.

Eg :-

Sales 2007 q1

~~name~~	~~amount~~
mike	— — — —
Jon	— — — —
mary	— — — —

Sales 2007 q2

name	amount
mike	— — —
JON	— — —
mary	— — —

```
SELECT * FROM Sales2007q1
Union
SELECT * FROM Sales2007q2;
```

Union

name	amount
JON	— — —
JON	— — —
mary	— — —
mike	— — —
Jon	— — —
mary	— — —

```
SELECT * FROM Sales2007q1
UNION ALL
SELECT * FROM Sales2007q2
```

name	amount
mike	— —
Jon	— —
mary	— —
mike	— —
JON	— —
May	— —

it deletes the duplicate table