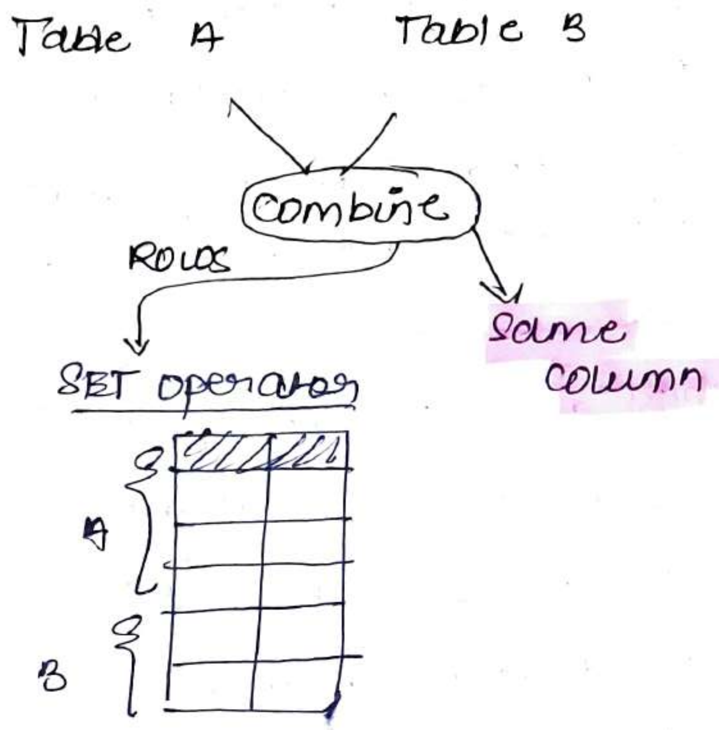
SET OPERATOR

Set operator in SQL combine the results of Multiple Queries into a single Result Set.



Syntax & Rules

```
                    SELECT
1st Select      ←         Firstname
   Statement              Lastname
                    FROM  customers

                 [UNION] → SET OPERATOR

                   SELECT
2nd Select               Firstname
   statement     ←       Lastname
                   FROM  Employee
```

Rule

1) SQL clauses

- SET operator can be used almost in all SQL clauses.
  - WHERE | JOIN | GROUP BY | HAVING.
- ORDER BY is allowed only once at the end of Query.
  ↓ can be used only at the end to sort the final Result.

2) RULE | Number of columns.

- Number of columns in each query must be the same.

3) DATA TYPES

- Data types in each Query Must be compatible. Means Data type should also not change.

4) ORDER OF COLUMNS

- Order of column in each query must be the same.

5) Column Alias

- Column names in the Result set are determined by the column names specified in the first Query.

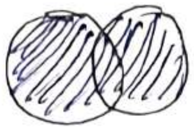
```
SELECT
  customer ID,  ]
  Lastname      ]  → 1st query controls column Name.
  FROM customer

UNION
    SELECT
        Employee ID,
        Lastname
      FROM employee.
```

6) CORRECT COLUMNS

- Even if all Rules are met and SQL shows no errors - the result may be incorrect.

- Incorrect column selection leads to inaccurate Result.

1) UNION



- Return All distric Row from both Queries.

- Removes Duplicates Rows from the List.

Ex:- combine the data from employees and customers into one Table.

- ~~Customers~~
- Firstname
- Lastname.

```
SELECT
    firstname,
    Lastname
FROM customers
UNION
    SELECT
      Firstname,
      Lastname
    FROM Employee.
```

Order of Queries in UNION doesn't affect the Result.

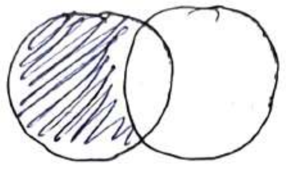
2) UNION ALL

- Return all Rows From both Queries, Including duplicates.

- UNION All is generally Faster than UNION.
- If you're confident there are no duplicates, use UNION All.

• use union all to find duplicates and Quality issues.

2) EXCEPT (MINUS)

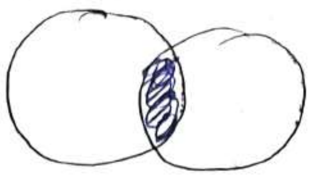


- Removes All distinct rows from the first Query.
- They are not found in the second Query.
- It is the only one where the order of Queries affects the Result.

Ex:- find employees who are not customers at same time.

```
SELECT
    firstname,
    Lastname
FROM employee

 EXCEPT
      SELECT
          firstname,
         Lastname
      FROM customer
```

3) INTERSECT



- Return only the Rows that are comman in both queries.
- Duplicates are not Allowed.

Ex: Employees who are also customers.

```
SELECT
      Firstname,
      Lastname
FROM ~~Sales~~. Employee

INTERSECT
      SELECT
            Firstname,
            Lastname
      FROM customers
```

4) UNION use cases.

- combine information (similar) before analyzing the data.

Eg: orders are stored in separate Tables (order and orders Archive).

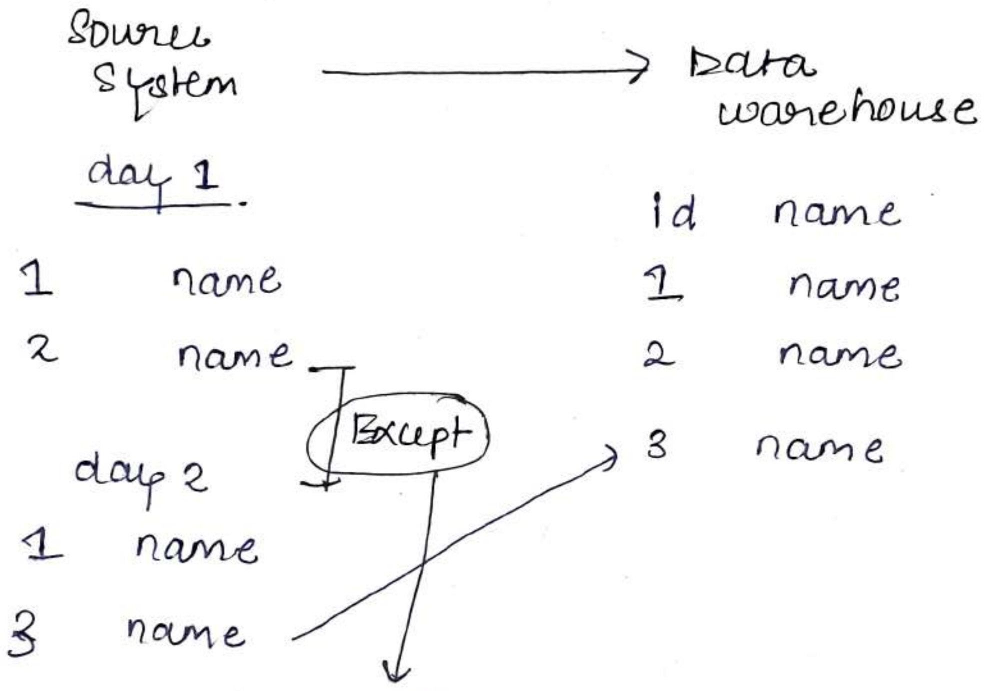
- Combine all orders into one Report without duplicates.

```
SELECT
 'Orders' As source Table,
      orderID,
      product ID,
      customerID,
      salesperson ID,
      OrderDate
   FROM sales.orders
SELECT
 'Orders Archieve' As source Table,
         OrderID,
         productID,
         customerID,
         salesperson ID,
         OrderDate
      FROM sales.orders Archieve.
```

• Except use cases

DELTA EXECUTION

identifying the difference or changes (delta) b/w two batches of data.



day 1

1	name
2	name

day 2

1	name
3	name

id	name
1	name
2	name
3	name

we use except to see new data/Record on this by remove similiar data from previous Table.

Data compltness check.

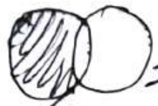
except operator can be used to compare Tables to detect discrepancies b/w databases.

Database A

Database B

Table original $\xrightarrow{\text{Transper}}$ Table copy

use Bxcept



⇒ Bmpty

To check that all data is Transper by SET operator (Bxcept) in DB A (Table) to DB B (Table).

Empty = Means all data Transper.

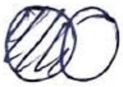
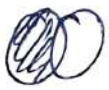
Table D  Table C ⇒ Empty

Table C  Table D ⇒ Empty.