Set operations

create table emp1(empid int primary key,name varchar (10),country varchar(20));
insert into emp1 values(1, 'shubham', 'india');
insert into emp1 values(2, 'aman', 'australia');
insert into emp1 values(3, 'naveen', 'sri lanka');
insert into emp1 values(4, 'aditya', 'austria');
insert into emp1 values(5, 'nishant', 'spain');

create table emp2(empid int primary key,name varchar(10),country varchar(20));
insert into emp2 values(1, 'tommy', 'england');
insert into emp2 values(2, 'alan', 'france');
insert into emp2 values(3, 'nancy', 'india');
insert into emp2 values(4, 'adi', 'ireland');
insert into emp2 values(5, 'sandy', 'spain');

empid	name	country
1	shubham	india
2	aman	australia
3	naveen	sri lanka
4	aditya	austria
5	nishant	spain

empid	name	country
1	tommy	england
2	alan	france
3	nancy	india
4	adi	ireland
5	sandy	spain

select country from emp1 union/union all/intersect/except select country from emp2 order by country;

country ▲ 1
australia
austria
england
france
india
ireland
spain
sri lanka

country ▲ 1
australia
austria
england
france
india
india
ireland
spain
spain
sri lanka

country ▲ 1
india
spain

country ▲ 1
australia
austria
sri lanka

SQL Joins:

create table customerz(cust_id int primary key, first_name varchar(10), last_name varchar(10), add_id int(10));
insert into customerz values(1, 'mary', 'jonas', 5);
insert into customerz values(2, 'linda', 'smith', 7);
insert into customerz values(3, 'madan', 'mohan', 8);
insert into customerz values(4, 'bary', 'williams', 6);

create table transactionz(cust_id int primary key, amount int (10), mode varchar(20), datez date, foreign key (cust_id) references customerz(cust_id));
insert into transactionz values(1, 60, 'cash', '3/4/2024');
insert into transactionz values(2, 30, 'credit card', '5/4/2024');
insert into transactionz values(8, 110,'cash', '15/4/2024');
insert into transactionz values(10, 70,'mobile payment', '20/4/2024');
insert into transactionz values(11, 80,'cash', '29/4/2024');

select * from customerz as c inner join transactionz as p on c.cust_id=p.cust_id;

cust_id	first_name	last_name	add_id	cust_id	amount	mode	datez
1	mary	jonas	5	1	60	cash	3/4/2024
2	linda	smith	7	2	30	credit card	5/4/2024

select * from customerz as c left join transactionz as p on c.cust_id=p.cust_id;

cust_id	first_name	last_name	add_id	cust_id	amount	mode	datez
1	mary	jonas	5	1	60	cash	3/4/2024
2	linda	smith	7	2	30	credit card	5/4/2024
3	madan	mohan	8				
4	bary	williams	6				

select * from customerz as c right join transactionz as p on c.cust_id=p.cust_id;

cust_id	first_name	last_name	add_id	cust_id	amount	mode
1	mary	jonas	5	1	60	cash
2	linda	smith	7	2	30	credit card

select * from customerz as c full join transactionz as p on c.cust_id=p.cust_id;

Window function:

create table new(new_id int, new_cat varchar);
insert into new values(100, 'agni');
insert into new values(200, 'agni');
insert into new values(500, 'dharti');
insert into new values(700, 'dharti');
insert into new values(200, 'vayu');

insert into new values(300, 'vayu');
insert into new values(500, 'vayu');

New

new_id	new_cat
100	agni
200	agni
500	dharti
700	dharti
200	vayu
300	vayu
500	vayu

1. Aggregate

```
select new_id, new_cat,
SUM(new_id)OVER(PARTITION BY new_cat ORDER BY new_id) AS "Total",
AVG(new_id)OVER(PARTITION BY new_cat ORDER BY new_id) AS "Average",
COUNT(new_id)OVER(PARTITION BY new_cat ORDER BY new_id) AS "Count",
MIN(new_id)OVER(PARTITION BY new_cat ORDER BY new_id) AS "Min",
MAX(new_id)OVER(PARTITION BY new_cat ORDER BY new_id) AS "Max"
FROM new;
```

Output

new_id	new_cat	Total	Average	Count	Min	Max
100	agni	100	100	1	100	100
200	agni	300	150	2	100	200
500	dharti	500	500	1	500	500
700	dharti	1200	600	2	500	700
200	vayu	200	200	1	200	200
300	vayu	500	250	2	200	300
500	vayu	1000	333.3333333333333	3	200	500

```
select new_id, new_cat,
SUM(new_id)OVER(ORDER BY new_id ROWS BETWEEN UNBOUNDED PRECEDING AND
UNBOUNDED FOLLOWING) AS "Total",
AVG(new_id)OVER(ORDER BY new_id ROWS BETWEEN UNBOUNDED PRECEDING AND
UNBOUNDED FOLLOWING) AS "Average",
COUNT(new_id)OVER(ORDER BY new_id ROWS BETWEEN UNBOUNDED PRECEDING
AND UNBOUNDED FOLLOWING) AS "Count",
MIN(new_id)OVER(ORDER BY new_id ROWS BETWEEN UNBOUNDED PRECEDING AND
UNBOUNDED FOLLOWING) AS "Min",
MAX(new_id)OVER(ORDER BY new_id ROWS BETWEEN UNBOUNDED PRECEDING AND
UNBOUNDED FOLLOWING) AS "Max"
FROM new;
```

Output

new_id	new_cat	Total	Average	Count	Min	Max
100	agni	2500	357.14285714285717	7	100	700
200	agni	2500	357.14285714285717	7	100	700
200	vayu	2500	357.14285714285717	7	100	700
300	vayu	2500	357.14285714285717	7	100	700
500	dharti	2500	357.14285714285717	7	100	700
500	vayu	2500	357.14285714285717	7	100	700
700	dharti	2500	357.14285714285717	7	100	700

2.Ranking

```
select new_id,
row_number() over (order by new_id) as "row_number",
rank() over (order by new_id) as "rank",
dense_rank() over (order by new_id) as "dense_rank",
percent_rank() over (order by new_id) as "percent_rank"
FROM new;
```

Output

new_id	row_number	rank	dense_rank	percent_rank
100	1	1	1	0
200	2	2	2	0.16666666666666666
200	3	2	2	0.16666666666666666
300	4	4	3	0.5
500	5	5	4	0.6666666666666666
500	6	5	4	0.6666666666666666
700	7	7	5	1

3.Analytic

```
select new_id,
first_value(new_id) over (order by new_id) as "first_value",
last_value(new_id) over (order by new_id) as "last_value",
lead(new_id) over (order by new_id) as "lead",
lag(new_id) over (order by new_id) as "lag"
FROM new;
```

Output

new_id	first_value	last_value	lead	lag
100	100	100	200	
200	100	200	200	100
200	100	200	300	200
300	100	300	500	200
500	100	500	500	300
500	100	500	700	500
700	100	700		500

```
select new_id,
lead(new_id,2) over (order by new_id) as "lead_by2",
lag(new_id,2) over (order by new_id) as "lag_by2"
FROM new;
```

Output

new_id	lead_by2	lag_by2
100	200	
200	300	
200	500	100
300	500	200
500	700	200
500		300
700		500