Sub_Query

1. Here in out orders table…if we want to find the avg of all sales…we just do **avg(sales) from orders**
2. if we have table like this

order_id	product_id	sales
1	100	500
1	200	700
2	300	600

..then if calculate avg of sales..we get 600..but in reality..it shud be

order_id	sales
1	1200
2	600

900

3. So here first we have to calculate the total sales of each order_id..and then calculate avg of sales…
4. Here come the subqueries
5. Lets see this in SQL now…to find avg order sales from orders

```
--sub query

select avg(order_sales) from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated

--230
--458
```

6.
7. Here first we will compute..sum of sales for each order_id and we will name it as orders_aggregated(this is our inner query)
8. Next from our orders_aggregated table..we compute avg(order_sales) from this table..see code and get intuition
9. Now we want to find order_id with sales greater than avg_sales

```
select order_id
from orders
group by order_id
having sum(sales) > (select avg(order_sales) as avg_order_value from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated)
```

10.
11. Here for our second subquery ..we didn't give any alias name..because in our 3rd query..we are just comparing the result with the second query..
12. And we have to give alias only if we using **from**…and here if we can see..there's no from keyword in our 3rd subquery

```
select * from employee
where dept_id not in (select dep_id from dept)
```

13. choosing an employee dept id..where the dep_id is not in dept table

14. Here in this query ..we used inner query as condition..so there's no need of aliasing the inner query
15.
```
select * from employee
where dept_id not in (select dep_id from dept)
```
try this query ..giving two columns in outer query and inner query(task for myself)
16.
```
select dept_id from employee
except
select dep_id from dept
```
try this..using * instead of dept_id & dep_id(task for myself)
17. Subqueries with joins
18.
```
select A.*,B.*
from
(select order_id , sum(sales) as order_sales
from orders
group by order_id ) A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated) B
on 1=1;
```
here you can compare this query with this
```
select a.*,b.
from
table a
inner join table b on col=col
```
19.
```
select A.*,B.*
from
(select order_id , sum(sales) as order_sales
from orders
group by order_id ) A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated) B
on 1=1;

select a.*,b.*
from
```
100 %

Results | Messages

	order_id	order_sales	avg_order_value
1	CA-2018-100006	377.97	458.614865661807
2	CA-2018-100090	699.192	458.614865661807
3	CA-2018-100293	91.056	458.614865661807
4	CA-2018-100328	3.928	458.614865661807
5	CA-2018-100363	21.376	458.614865661807
6	CA-2018-100391	14.62	458.614865661807
7	CA-2018-100678	697.074	458.614865661807
8	CA-2018-100706	129.44	458.614865661807

the output of subqueries using inner join query

20. Order id with order sales greater than avg_order_sales

```
select A.*,B.*
from
(select order_id , sum(sales) as order_sales
from orders
group by order_id ) A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated) B
on 1=1
where order_sales > avg_order_value;

select a.*,b.*
```

100 %

Results | Messages

	order_id	order_sales	avg_order_value
1	CA-2018-100090	699.192	458.614665681807
2	CA-2018-100678	697.074	458.614665681807
3	CA-2018-100762	508.62	458.614665681807
4	CA-2018-100895	605.47	458.614665681807
5	CA-2018-100916	788.86	458.614665681807
6	CA-2018-101560	542.34	458.614665681807
7	CA-2018-101602	803.96	458.614665681807
8	CA-2018-101931	1252.602	458.614665681807

21. Here we have our employee table

```
select * from employee;
```

100 %

Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob
1	1	Ankit	100	10000	4	39	1983-12-02
2	2	Mohit	100	15000	5	48	1974-12-02
3	3	Vikas	100	10000	4	37	1985-12-02
4	4	Rohit	100	5000	2	16	2006-12-02
5	5	Mudit	200	12000	6	55	1967-12-02
6	6	Agam	200	12000	2	14	2008-12-02
7	7	Sanjay	200	9000	2	13	2009-12-02
8	8	Ashish	200	5000	2	12	2010-12-02
9	9	Mukesh	300	6000	6	51	1971-12-02
10	10	Rakesh	700	7000	6	50	1972-12-02
11	11	Ramesh	300	8000	6	52	1970-12-02

22.

23. Here we are trying to calculate the avg salaries for each department in a separate table..next we will join avg salaries table..with the employee table

24. Now here we have calculated avg dept salary..see pic..2nd query is avg dep sal and first query is employee table

```
select * from employee;

select dept_id,avg(salary) as avg_dep_salary
from employee
group by dept_id
```

100 %

Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob
1	1	Ankit	100	10000	4	39	1983-12-02
2	2	Mohit	100	15000	5	48	1974-12-02
3	3	Vikas	100	10000	4	37	1985-12-02
4	4	Rohit	100	5000	2	16	2006-12-02
5	5	Mudit	200	12000	6	55	1967-12-02
6	6	Agam	200	12000	2	14	2008-12-02
7	7	Sanjay	200	9000	2	13	2009-12-02
8	8	Ashish	200	5000	2	12	2010-12-02

	dept_id	avg_dep_salary
1	100	10000
2	200	9500
3	300	7000

25. Now we will join this 2 queries using inner join

```
select e.*,d.avg_dep_salary from
employee e
inner join
(select dept_id,avg(salary) as avg_dep_salary
from employee
group by dept_id) d
on e.dept_id=d.dept_id
```

100 %

Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	avg_dep_salary
1	1	Ankit	100	10000	4	39	1983-12-02	10000
2	2	Mohit	100	15000	5	48	1974-12-02	10000
3	3	Vikas	100	10000	4	37	1985-12-02	10000
4	4	Rohit	100	5000	2	16	2006-12-02	10000
5	5	Mudit	200	12000	6	55	1967-12-02	9500
6	6	Agam	200	12000	2	14	2008-12-02	9500
7	7	Sanjay	200	9000	2	13	2009-12-02	9500
8	8	Ashish	200	5000	2	12	2010-12-02	9500
9	9	Mukesh	300	8000	6	51	1971-12-02	7000
10	11	Ramesh	300	8000	6	52	1970-12-02	7000
11	10	Rakesh	700	7000	6	50	1972-12-02	7000

26. Basically imaging this subquery results as a table…here calculating avg salaries output is named as table d and employee table as table e…and we performed inner join on this two tables

27. For icc_world_cup table question KSQL7_Assignment ..the answer is

```
select team_name,count(1) as matches_played,sum(win_flag) matches_won,count(1)-sum(win_flag) as lost_matche
from
(select team_1 as team_name,case when team_1=winner then 1 else 0 end as win_flag
from icc_world_cup
union all
select team_2 as team_name,case when team_2=winner then 1 else 0 end as win_flag
from icc_world_cup) A
group by team_name
;
team_nae,matches_palyes,no of wins,lost
```

100 %

Results | Messages

	team_name	matches_played	matches_won	lost_matches
1	Aus	2	1	1
2	Eng	2	1	1
3	India	2	2	0
4	NZ	1	1	0
5	SA	1	0	1
6	SL	2	0	2

28. In the above query …first we have performed union all..with using two queries

```
select team_1 as team_name,case when team_1=winner then 1 else 0 end as win_flag
from icc_world_cup
union all
select team_2 as team_name,case when team_2=winner then 1 else 0 end as win_flag
from icc_world_cup

team_nae,matches_palyes,won,lost
```

100 %

Results | Messages

	team_name	win_flag
1	India	1
2	SL	0
3	SA	0
4	Eng	0
5	Aus	0
6	SL	0
7	Aus	1
8	Eng	1
9	NZ	1
10	India	1

Query executed successfully.

29. Now we will name this output table as A(sub query concept)..and from this table..we generate our required output…which is “team_name,matches_played,won,lost”

CTE - common table expression

1. The CTE’s are similar to subqueries…but in CTE..we don't alias ..instead ..we use “**with table_name as”** and perform the query…and store this result output table in **table_name**

2.
```
--cte common table expression
with A as
(select team_1 as team_name,case when team_1=winner then 1 else 0 end as win_flag
from icc_world_cup
union all
select team_2 as team_name,case when team_2=winner then 1 else 0 end as win_flag
from icc_world_cup)
select team_name,count(*) as matches_played,sum(win_flag) matches_won,count(1)-sum(win_flag) as lost_matche
from A
group by team_name
```

Results | Messages

team_name	matches_played	matches_won	lost_matches
Aus	2	1	1
Eng	2	1	1
India	2	2	0
NZ	1	1	0
SA	1	0	1

3. After that we performed ..the last 3 lines of the SQL queries…using **table_name** output
4. Another example of CTE

```
with dep as (select dept_id,avg(salary) as avg_dep_salary
from employee
group by dept_id)
select e.*,d.* from
employee e
inner join dep d
on e.dept_id=d.dept_id;
```

100 % | Results | Messages

	emp_id	emp_name	dept_id	salary	manager_id	emp_age	dob	dept_id	avg_dep_salary
1	1	Ankit	100	10000	4	39	1983-12-02	100	10000
2	2	Mohit	100	15000	5	48	1974-12-02	100	10000
3	3	Vikas	100	10000	4	37	1985-12-02	100	10000
4	4	Rohit	100	5000	2	16	2006-12-02	100	10000
5	5	Mudit	200	12000	6	55	1967-12-02	200	9500
6	6	Agam	200	12000	2	14	2008-12-02	200	9500

5. Refer the previous example…and understand this
6. Structure wise..both the CTE and subqueries are same..but the way of writing is diff
7. Nested CTE example

```
with dep as (select dept_id,avg(salary) as avg_dep_salary
from employee
group by dept_id)
,total_salary as (select sum(salary) as ts from employee)
select e.*,d.* from
employee e
inner join dep d
on e.dept_id=d.dept_id;
```

8. Readability will be good in CTE compared to subqueries

```sql
select A.*,B.*
from
(select order_id , sum(sales) as order_sales
from orders
group by order_id ) A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
(select order_id,sum(sales) as order_sales
from orders
group by order_id) as orders_aggregated) B
on 1=1
where order_sales > avg_order_value;
```

9.

10. This query with CTE would be

```sql
with order_wise_sales as (select order_id , sum(sales) as order_sales
from orders
group by order_id)
select A.*,B.*
from
order_wise_sales A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
order_wise_sales as orders_aggregated) B
on 1=1
where order_sales > avg_order_value;
```

11. Now in below highlighted query…we can use CTE there

```sql
with order_wise_sales as (select order_id , sum(sales) as order_sales
from orders
group by order_id)
,
select A.*,B.*
from
order_wise_sales A
inner join
(select avg(orders_aggregated.order_sales) as avg_order_value from
order_wise_sales as orders_aggregated) B
on 1=1
where order_sales > avg_order_value;
```

12.

13. It would look like this

```sql
with order_wise_sales as (select order_id , sum(sales) as order_sales
from orders
group by order_id)
,B as (select avg(orders_aggregated.order_sales) as avg_order_value from
order_wise_sales as orders_aggregated)
select A.*,B.*
from
order_wise_sales A
inner join
 B
on 1=1
where order_sales > avg_order_value;
```

14. It uses nested CTE’s
15. Whatever can be done by subquery..can be done by CTE’s as well
16. We have this query

```
select *,(select avg(salary) from employee) as avg_sal from employee
where dept_id  in (select dep_id from dept);
```

17.
18. Now this can also be