CODTECH IT SLOUTIONS INTERNSHIP TASK 2 [SQL]

DATA ANALYSIS WITH COMPLEX QUERIES

- Window_ function_ name: The name of the window function (e.g., ROW_NUMBER(), RANK(), SUM()).
- expression: The target column or expression on which the window function operates.
- OVER: Defines the window or set of rows the function operates on.
- PARTITION BY: Divides the result set into partitions.
- ORDER BY: Specifies the order of rows within each partition.

CTE(COMMON TABLE EXPRESSIONS)

In SQL, CTE stands for Common Table Expression. It is a temporary result set that you can reference within a SELECT, INSERT, UPDATE, or DELETE statement. CTEs make complex queries easier to read and maintain by breaking them into smaller, logical parts.

Key Features:

1. Improves Readability: Simplifies complex queries by dividing them into manageable parts.
2. Reusable: You can reference the CTE multiple times within the same query.
3. Temporary: Exists only during the execution of the query.

What are Subqueries?

- A subquery is a query within a query.
- It is enclosed in parentheses () and can be used in SELECT, FROM, WHERE, or other clauses.
- Subqueries can return a single value (scalar), a list of values, or a table.

Types of Subqueries

1. Single-row Subquery: Returns one row with one column.
2. Multi-row Subquery: Returns multiple rows (used with operators like IN, ANY, ALL).
3. Correlated Subquery: Depends on the outer query for its values and is executed repeatedly for each row of the outer query.

TABLES CREATION SCRIPT

```sql
create database karthiksv676

create table employees110(
empID int,
ename varchar (50),
deptid int,
salary int,
join_date date
);

insert into employees110 values
(1,'KARTHIK',1,60000,'2021-01-15'),

(2,'KIRAN',1,55000,'2023-03-22'),

(3,'MADHAN',2,70000,'2021-05-10'),

(4,'GOPICHAND',2,72000,'2023-02-01'),

(5,'GURURAJ',3,50000,'2021-08-12');

select * from employees110

create table Department110(
Dept_id int primary key,
Dept_Name varchar(50)
);

insert into Department110 values
(1,'IT'),

(2,'Finace'),

(3,'HR'),

(4,'Marketing'),

(5,'Analytics');
```

select * from Department110

select * from employees110

	Dept_id	Dept_Name
1	1	IT
2	2	Finace
3	3	HR
4	4	Marketing
5	5	Analytics

	empID	ename	deptid	salary	join_date
1	1	KARTHIK	1	60000	2021-01-15
2	2	KIRAN	1	55000	2023-03-22
3	3	MADHAN	2	70000	2021-05-10
4	4	GOPIC...	2	72000	2023-02-01
5	5	GURU...	3	50000	2021-08-12

FIND THE SECOND HIGHEST SALARY

```
select ename , deptid ,   MAX(salary) as secondhightestsalary
from employees110
where salary<(select MAX(salary) from employees110)
group by ename, deptid
```

	ename	deptid	secondhightestsalary
1	KARTHIK	1	60000
2	KIRAN	1	55000
3	MADHAN	2	70000
4	GURURAJ	3	50000

OR

```
select MAX(salary) as secondhightestsalary
from employees110
where salary<(select MAX(salary) from employees110)
```

	secondhightestsalary
1	70000

USE OF WINDOWS FUNCTION – RANKED EMPLOYEES SALARY

select ename,deptid,salary,

RANK() over(partition by deptid order by salary desc) as SalaryRnk from employees110;

	ename	deptid	salary	SalaryRnk
1	KARTHIK	1	60000	1
2	KIRAN	1	55000	2
3	GOPICHAND	2	72000	1
4	MADHAN	2	70000	2
5	GURURAJ	3	50000	1

RANK()

ROW_NUMBER()

DENSE_RANK()

```
select *,
RANK()over(order by salary desc) as rnk
,dense_rank() over(order by salary asc) as densrnk
,row_number() over(order by salary) as rn
from employees110
order by salary
```

	empID	ename	deptid	salary	join_date	rnk	densrnk	rn
1	5	GURURAJ	3	50000	2021-08-12	5	1	1
2	2	KIRAN	1	55000	2023-03-22	4	2	2
3	1	KARTHIK	1	60000	2021-01-15	3	3	3
4	3	MADHAN	2	70000	2021-05-10	2	4	4
5	4	GOPICHAND	2	72000	2023-02-01	1	5	5

USE OF CTE FIND THE RECENT JOINERS

```
with Recentjoiners as(
select ename,join_date     from employees110
where join_date>='2022-01-01'
)
select * from Recentjoiners
```

	ename	join_date
1	KIRAN	2023-03-22
2	GOPICHAND	2023-02-01

AVERAGE SALARY OF EACH DEPARTMENT

```
select  deptid, Department110.Dept_Name  ,AVG( salary)as avgsalary
from employees110
join Department110 on employees110.deptid=Department110.Dept_id
group by deptid,   Department110.Dept_Name
```

	deptid	Dept_Name	avgsalary
1	1	IT	57500
2	2	Finace	71000
3	3	HR	50000

USE OF SUBQUERY FIND EMPLOYEES EARNING ABOVE DEPARTMENT AVERAGE

```
select  ename,salary,deptid  from employees110
where salary>(select AVG(salary) from employees110 as e2
where e2.deptid=employees110.deptid
);
```

	ename	salary	deptid
1	KARTHIK	60000	1
2	GOPICHAND	72000	2