Section A – Leet Code Questions

1. **Replace Employee ID With The Unique Identifier**

```
Select eu.unique_id, e.name from employees e
left join employeeuni eu
on e.id = eu.id;
```

2. **Product Sales Analysis I**

```
Select p.product_name, s.year, s.price
from Sales s
left join product p
on s.product_id = p.product_id;
```

3. **Customer Who Visited but Did Not Make Any Transactions**

```
Select v.customer_id, count(v.visit_id) as count_no_trans
from visits v
left join transactions t
on v.visit_id = t.visit_id
where t.transaction_id is null
group by v.customer_id
order by v.customer_id;
```

4. **Employee Bonus**

```
Select e.name,b.bonus
from Employee e
left join bonus b
on e.empid = b.empid
where b.bonus < 1000 or b.bonus is null
order by b.bonus;
```

Section B – Adventure Works Table

5. **SalesLT. Sales Order Header**

- **Find the top 5 customers by total number of orders placed.**

```
Select top 5 customerid, count(salesorderid) as total_order_placed from [SalesLT].[SalesOrderHeader]
group by CustomerID
```

order by total_order_placed desc ;

- **Find customers who placed at least 3 orders and have spent more than 2000 in total.**

```
Select CustomerID
from [SalesLT].[SalesOrderHeader]
group by CustomerID
having count(SalesOrderID) >= 3 and sum(subtotal) > 2000;
```

6. **SalesLT. Product**

- **List all products that have a List Price greater than the average List Price.**

```
Select distinct Name
from [SalesLT].[Product]
where ListPrice > (select avg(ListPrice) from [SalesLT].[Product])
order by name;
```

- **Find the product categories (Product Category ID) that have more than 5 products with a ListPrice greater than 1000.**

```
Select   productcategoryid
from [SalesLT].[Product]
where ListPrice > 1000
group by productcategoryid
having count(ProductID) > 5;
```

Section C SQL Hands on

In this part, you will design and populate your own table called Employees. You will:

Q1. Create a table named Employees with the following columns:

-- Emp ID (INT, Primary Key)

-- Emp Name (VARCHAR(50))

-- Department (VARCHAR(50))

-- Salary (DECIMAL(10,2))

-- Years Of Service (INT)

```
Create table Employees (
EmpID INT primary key,
EmpName VARCHAR(50),
Department Varchar(50),
Salary DECIMAL(10,2),
YearsofService INT
);
```

Q2. Insert at least 15 records of your own choice into the Employees table.

- **Make sure you include employees from at least 3 different departments (e.g., IT, HR, Finance, Sales, etc.).**

```
INSERT INTO Employees values
(1,'Ram','IT', 12000,5),
(2,'Shyam','HR', 11009.5,3),
(3,'Tim','IT', 80000,2),
(4,'Gina','Finance', 10009.7,5),
(5,'Hina','IT', 90000,2),
(6,'Pratik','Sales', 150000,7),
(7,'Shweta','Sales', 115000,4),
(8,'Rohit','IT', 125000,5),
(9,'Sparsh','Finance', 92000,3),
(10,'Shristi','Finance', 150000,7),
(11,'Tom','HR', 250000,14),
(12,'Jim','IT', 60000,2),
(13,'Sim','IT', 45000,1),
(14,'Ryan','Finance', 52000,1),
(15,'Dave','Finance', 68000,2),
(16,'Shubham','Sales', 12000,3),
(17,'Ram','HR', 97500.5,4);
```

Q3. From the Employees table, find the average salary in each department.

```
Select department , cast(round(avg(salary),2) as decimal(10,2)) as average_salary from Employees
group by Department
order by department;
```

Q4. From the Employees table, find the total salary expenditure by each department.

```
select department , cast(sum(salary) as decimal(10,1)) as total_salary from Employees
group by department
order by department;
```

Q5. List the departments that have more than 3 employees.

```
Select department
from employees
group by department
having count(empid) > 3;
```

Q6. Find the employee(s) with the maximum years of service in each department.

```
select empid, empname,department, yearsofservice from (
select e.*, row_number() over(partition by department order by yearsofService desc) rn
from employees e )a
where rn =1;
```