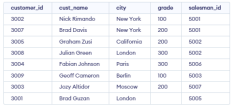
Question 1:

From the following tables, write a SQL query to find all salespeople and customers located in the city of London.

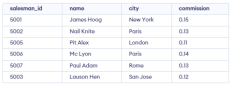
Table Structure:

- salesman_id INT , name VARCHAR, city VARCHAR, commission DECIMAL
- customer_id INT , cust_name VARCHAR, city VARCHAR(50), grade INT, salesman_id INT

Input Tables:



customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3007	Brad Davis	New York	200	5001
3005	Graham Zusi	California	200	5002
3008	Julian Green	London	300	5002
3004	Fabian Johnson	Paris	300	5006
3009	Geoff Cameron	Berlin	100	5003
3003	Jozy Altidor	Moscow	200	5007
3001	Brad Guzan	London		5005



salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
5003	Lauson Hen	San Jose	0.12

Title for Question 1: Union function Salesman Question

Solution:

```
SELECT salesman_id "ID", name, 'Salesman'
FROM salesman
WHERE city='London'
UNION
(SELECT customer_id "ID", cust_name, 'Customer'
FROM customer
WHERE city='London')
```

TestCases:

S.No	Inputs	Outputs
1	na	ID | name | Salesman ---- | ---------- | -------- 5005 | Pit Alex | Salesman 3001 | Brad Guzan | Customer 3008 | Julian Green | Customer
2		
3		
4		
5		
6		

White List:

Black List:

Question 2:

Imagine you are an HR manager at a company, and you need to identify employees who are earning above the company-wide average salary.

Requirements:

- Using Sub-query
- Using Join

Output Format :

The result of this query will provide you with a list of employees (their names and departments) who earn salaries above the average salary of all employees in the company.

Table Structure:

EmployeeID INTEGER PRIMARY KEY, Name TEXT, Department TEXT, Salary REAL

Input Table:??????

EmployeeID	Name	Department	Salary
1	Alice	HR	65,000.00
2	Bob	Engineering	75,000.00
3	Charlie	HR	62,000.00
4	David	Sales	58,000.00
5	Eve	Engineering	72,000.00

Title for Question 2: Query Equalization (Sub-query and Join)

Solution:

```
SELECT E.Name, E.Department
FROM Employees E
WHERE E.Salary > (SELECT AVG(Salary) FROM Employees);
SELECT E.Name, E.Department
FROM Employees E
JOIN (SELECT AVG(Salary) AS AvgSal FROM Employees) AS Subquery
ON E.Salary > Subquery.AvgSal;
```

TestCases:

S.No	Inputs	Outputs
1	na	Name | Department ---- | ---------- Bob | Engineering Eve | Engineering Name | Department ---- | ---------- Bob | Engineering Eve | Engineering
2		

S.No	Inputs	Outputs
3		
4		
5		
6		

White List:

Black List:

Question 3:

Imagine you have a database storing information about authors and the books they've written.

- The "Authors" table contains AuthorID and AuthorName, while the "Books" table includes ISBN, Title, AuthorID, and PublishedYear.
- Your task is to find all the books with a publication year earlier than 2010 and display the book title along with the author's name.
- Please write the SQL query to achieve the desired result.

Table structure:

Author Table:

- AuthorID, AuthorName

Books Table:

- ISBN, Title, AuthorID, PublishedYear

Input Table:

Authors

AuthorID	AuthorName
1	Author 1
2	Author 2
3	Author 3

Books

ISBN	Title	AuthorID	PublishedYear
978-1234567890	Book 1	1	2000
978-0987654321	Book 2	2	2010
978-5555555555	Book 3	3	2005

Title for Question 3: Author and books

Solution:

```
SELECT B.Title, A.AuthorName
FROM Books B
JOIN Authors A ON B.AuthorID = A.AuthorID
WHERE B.PublishedYear < 2010;
```

TestCases:

S.No	Inputs	Outputs
1		Title | AuthorName ------ | ---------- Book 1 | Author 1 Book 3 | Author 3
2		
3		
4		
5		
6		

White List:

Black List:

Question 4:

You are tasked with designing a database to store historical product prices.

- The "ProductPrices" table consists of ProductID, ValidFrom, ValidTo, and Price columns.
- It allows you to keep track of product prices over time.
- Can you provide an SQL query to retrieve the price of a specific product on a particular date, for example, on '2023-02-10' for ProductID 1?
- Please write the SQL query that accomplishes this task.

Table structure:

- ProductID, ValidFrom, ValidTo, Price

Input Table:

ProductPrices

ProductID	ValidFrom	ValidTo	Price
1	2023-01-01	2023-01-15	19.99
1	2023-01-16	2023-02-28	21.99
1	2023-03-01	2023-03-15	24.99
2	2023-01-01	2023-01-31	29.99
2	2023-02-01	2023-03-15	31.99

Title for Question 4: Price of the product

Solution:

```
-- Query to get the price of a product on a specific date
SELECT Price
FROM ProductPrices
WHERE ProductID = 1
    AND '2023-02-10' BETWEEN ValidFrom AND ValidTo;
```

TestCases:

S.No	Inputs	Outputs
1		Price ----- 21.99
2		
3		
4		
5		
6		

White List:

Black List:

Question 5:

Imagine you work at a medium-sized company, and the HR department recently assigned employees to different departments. You want to retrieve a list of employees along with their corresponding department names. This information will help you understand which employees belong to which departments. Note: Department, Employees are keep in two different table.

Requirements:

- Must use inner join.

Table Structure:

- ?????EmployeeID, FirstName, LastName, DeptID

Input Table 1:

EmployeeID	FirstName	LastName	DeptID
1	John	Doe	101
2	Jane	Smith	102
5	Robert	Brown	103

Input Table 2:

DeptID	DepartmentName
101	Sales
102	Marketing
103	HR

Title for Question 5: inner join without where clause

Solution:

```
SELECT Employees.FirstName, Department.DepartmentName
FROM Employees
INNER JOIN Department ON Employees.DeptID = Department.DeptID;
```

TestCases:

S.No	Inputs	Outputs
1		FirstName | DepartmentName --------- | ---------- John | Sales Jane | Marketing Robert | HR
2		
3		
4		
5		
6		

White List:

Black List:

Question 6:

Imagine you are managing a retail business and want to create a report that lists all customer names and their associated order IDs. You are interested in seeing all customers, even if they haven't placed any orders yet.

Requirements:

- Must use left join.

Table Structure:

- CustomerID, CustomerName

Input Table Customer:

CustomerID	CustomerName
1	Customer 1
2	Customer 2
3	Customer 3
4	Customer 4

Input Table Orde:

OrderID	CustomerID	OrderDate
1	4	2023-01-16
2	3	2023-02-20
3	8	2023-03-25
4	1	2023-04-10

Title for Question 6: Left join without where clause

Solution:

```
SELECT * FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

TestCases:

S.No	Inputs	Outputs
1		CustomerID | CustomerName | OrderID | OrderDate ---------- | ---------- | ------- | ---------- 1 | Customer 1 | 4 | 2023-04-10 null | Customer 2 | null | null 3 | Customer 3 | 2 | 2023-02-20 null | Customer 4 | null | null
2		
3		
4		
5		

S.No	Inputs	Outputs
6		

White List:

Black List:
