**Table: Customers**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **customer\_id** | **first\_name** | **last\_name** | **email** | **address** | **city** | **state** | **zip** |
| 1 | George | Washington | gwashington@usa.gov | 3200 Mt Vernon Hwy | Mount Vernon | VA | 22121 |
| 2 | John | Adams | jadams@usa.gov | 1250 Hancock St | Quincy | MA | 02169 |
| 3 | Thomas | Jefferson | tjefferson@usa.gov | 931 Thomas Jefferson Pkwy | Charlottesville | VA | 22902 |
| 4 | James | Madison | jmadison@usa.gov | 11350 Constitution Hwy | Orange | VA | 22960 |
| 5 | James | Monroe | jmonroe@usa.gov | 2050 James Monroe Parkway | Charlottesville | VA | 22902 |

**Table: Orders**

|  |  |  |  |
| --- | --- | --- | --- |
| **order\_id** | **order\_date** | **amount** | **customer\_id** |
| 1 | 07/04/1776 | $234.56 | 1 |
| 2 | 03/14/1760 | $78.50 | 3 |
| 3 | 05/23/1784 | $124.00 | 2 |
| 4 | 09/03/1790 | $65.50 | 3 |
| 5 | 07/21/1795 | $25.50 | 10 |
| 6 | 11/27/1787 | $14.40 | 9 |

Join

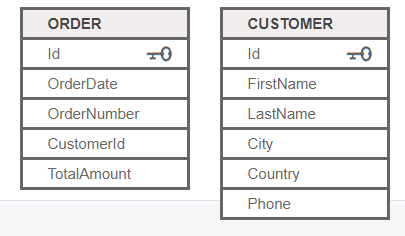
## Q. Inner Join

**Problem:** Let’s say we wanted to get a list of those customers who placed an order and the details of the order they placed. This would be a perfect fit for an inner join, since an inner join returns records at the intersection of the two tables.

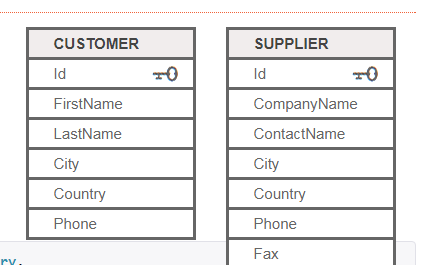
## Q. Left Join

**Problem:** If we wanted to simply append information about orders to our customers table, regardless of whether a customer placed an order or not, we would use a left join. A left join returns all records from table A and any matching records from table B.

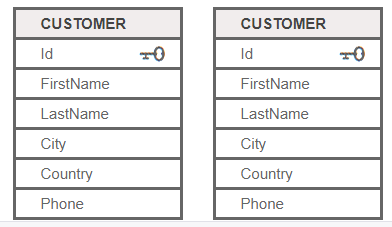
**Q. SQL RIGHT**

  
  
**Problem:** List customers that have not placed orders

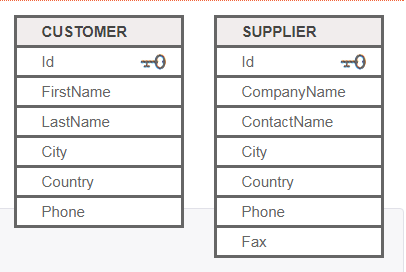
**Q. SQL FULL JOIN**

  
  
**Problem:** Match all customers and suppliers by country

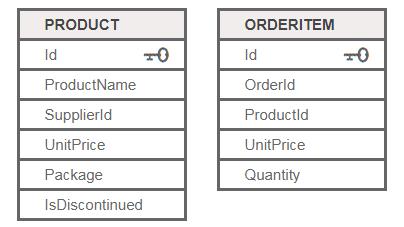
**Q. SQL Self JOIN**

  
  
**Problem:** Match customers that are from the same city and country

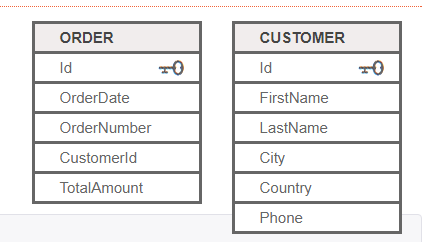
**Q. SQL UNION**

  
**Problem:** List all contacts, i.e., suppliers and customers.

**Q. SQL Subquery**

  
  
**Problem:** List products with order quantities greater than 100.

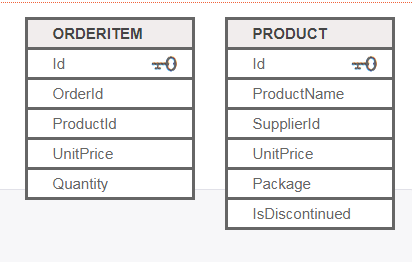
**Q. SQL Subquery**

  
**Problem:** List all customers with their total number of orders

**Q. SQL where**

**Problem**: Which products were sold by the unit (i.e. quantity = 1)

**Q. SQL where**

  
**Problem**: List customers who placed orders that are   
larger than the average of each customer order

**Q. SQL where**

**Problem:** Find suppliers with products over $100.

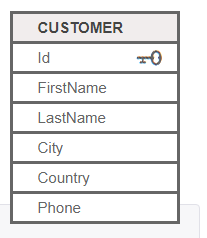
**Q. SQL SELECT INTO**

**Problem:** Copy all suppliers from USA to a new SupplierUSA table.

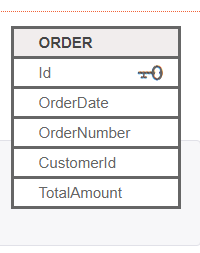
**Q. SQL INSERT SELECT INTO**

**Problem:** Copy all Canadian suppliers into the Customer table

**Q. SQL GROUP BY**

  
  
**Problem:** List the number of customers in each country. Only include countries with more than 10 customers.

**Problem:** List the number of customers in each country, except the USA, sorted high to low.   
Only include countries with 9 or more customers.



**Problem:** List all customer with average orders between $1000 and $1200.

### Q. SQL CASE statement

**Problem: For a given product and category abbreviation, show the full category name using a CASE expression.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| PRODUCTNUMBER | CATEGORY | NAME | PRICE\_RNAGE |  |  |
|  | R |  |  |  |  |
|  | M |  |  |  |  |
|  | T |  |  |  |  |

R - 'Road'

M - 'Mountain'

T- 'Touring'

**Problem: Based on an item list price, show the price range for the item.**The example below shows a searched CASE expression which evaluates a set of Boolean expressions to determine the result.

WHEN ListPrice = 0 THEN 'Mfg item - not for resale'  
 WHEN ListPrice < 50 THEN 'Under $50'  
 WHEN ListPrice >= 50 and ListPrice < 250 THEN 'Under $250'  
 WHEN ListPrice >= 250 and ListPrice < 1000 THEN 'Under $1000'  
 ELSE 'Over $1000'

**Mix – Questions**

**Table Name : Employee**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Employee\_id** | **First\_name** | **Last\_name** | **Salary** | **Joining\_date** | **Department** |
| 1 | John | Abraham | 1000000 | 01-JAN-13 12.00.00 AM | Banking |
| 2 | Michael | Clarke | 800000 | 01-JAN-13 12.00.00 AM | Insurance |
| 3 | Roy | Thomas | 700000 | 01-FEB-13 12.00.00 AM | Banking |
| 4 | Tom | Jose | 600000 | 01-FEB-13 12.00.00 AM | Insurance |
| 5 | Jerry | Pinto | 650000 | 01-FEB-13 12.00.00 AM | Insurance |
| 6 | Philip | Mathew | 750000 | 01-JAN-13 12.00.00 AM | Services |
| 7 | TestName1 | 123 | 650000 | 01-JAN-13 12.00.00 AM | Services |
| 8 | TestName2 | Lname% | 600000 | 01-FEB-13 12.00.00 AM | Insurance |

**Table Name : Incentives**

|  |  |  |
| --- | --- | --- |
| **Employee\_ref\_id** | **Incentive\_date** | **Incentive\_amount** |
| 1 | 01-FEB-13 | 5000 |
| 2 | 01-FEB-13 | 3000 |
| 3 | 01-FEB-13 | 4000 |
| 1 | 01-JAN-13 | 4500 |
| 2 | 01-JAN-13 | 3500 |

1. Get all employee details from the employee table

2. Get First\_Name,Last\_Name from employee table

3. Get First\_Name from employee table using alias name “Employee Name”

4. Get First\_Name from employee table in upper case

5. Get First\_Name from employee table in lower case

6. Get unique DEPARTMENT from employee table

7. Select first 3 characters of FIRST\_NAME from EMPLOYEE

8. Get position of 'o' in name 'John' from employee table

9. Get FIRST\_NAME from employee table after removing white spaces from right side

10. Get FIRST\_NAME from employee table after removing white spaces from left side

11. Get length of FIRST\_NAME from employee table

12. Get First\_Name from employee table after replacing 'o' with '$'

13. Get First\_Name and Last\_Name as single column from employee table separated by a '\_'

14. Get FIRST\_NAME ,Joining year,Joining Month and Joining Date from employee table