SQL Subqueries - Lab Assignment #2

Introduction

Now that you've seen how subqueries work, it's time to get some practice writing them! Not all of the queries will require subqueries, but all will be a bit more complex and require some thought and review about aggregates, grouping, ordering, filtering, joins and subqueries. Good luck!

Objectives

You will be able to:

- Write subqueries to decompose complex queries

CRM Database ERD

Once again, here's the schema for the CRM database you'll continue to practice with.

Connect to the Database

As usual, start by importing the necessary packages and connecting to the database `data2.sqlite` in the data folder.

In [9]:
```
# Your code here; import the necessary packages
import sqlite3
import pandas as pd
```

In [10]:
```
# Your code here; create the connection
conn = sqlite3.Connection("data/data2.sqlite")
```

Write an Equivalent Query using a Subquery

The following query works using a `JOIN`. Rewrite it so that it uses a subquery instead.

```
SELECT
    customerNumber,
    contactLastName,
    contactFirstName
FROM customers
JOIN orders
```

```
    USING(customerNumber)
WHERE orderDate = '2003-01-31'
;
```

In [51]:
```
q0 = """
SELECT
    customerNumber,
    contactLastName,
    contactFirstName
FROM customers
WHERE customerNumber IN (
    SELECT customerNumber FROM orders
    WHERE orderDate = '2003-01-31'
)
;
"""

pd.read_sql(q0, conn)
```

Out[51]:

	customerNumber	contactLastName	contactFirstName
0	141	Freyre	Diego

Select the Total Number of Orders for Each Product Name

Sort the results by the total number of items sold for that product.

In [40]:
```
# Your code here

q2 = """
SELECT p.productName, COUNT(o.orderNumber) as total_number_of_orders
FROM orderdetails o, products p
WHERE o.productCode = p.productCode
GROUP BY
    p.productName

ORDER BY
    total_number_of_orders
    DESC

;
"""

pd.read_sql(q2, conn)
```

Out[40]:

	productName	total_number_of_orders
0	1992 Ferrari 360 Spider red	53
1	P-51-D Mustang	28
2	HMS Bounty	28
3	F/A 18 Hornet 1/72	28
4	Diamond T620 Semi-Skirted Tanker	28
...	...	...
104	1932 Alfa Romeo 8C2300 Spider Sport	25
105	1917 Grand Touring Sedan	25
106	1911 Ford Town Car	25
107	1957 Ford Thunderbird	24
108	1952 Citroen-15CV	24

109 rows × 2 columns

Select the Product Name and the Total Number of People Who Have Ordered Each Product

Sort the results in descending order.

A quick note on the SQL `SELECT DISTINCT` statement:

The `SELECT DISTINCT` statement is used to return only distinct values in the specified column. In other words, it removes the duplicate values in the column from the result set.

Inside a table, a column often contains many duplicate values; and sometimes you only want to list the unique values. If you apply the `DISTINCT` clause to a column that has `NULL`, the `DISTINCT` clause will keep only one NULL and eliminates the other. In other words, the DISTINCT clause treats all `NULL` "values" as the same value.

In [36]:

```
# Your code here
# Hint: because one of the tables we'll be joining has duplicate customer numbe
q3 = """
SELECT p.productName, COUNT(DISTINCT o.quantityOrdered) as total_number_of_peop
FROM products p, orderdetails o
WHERE p.productCode = o.productCode

GROUP BY
    p.productName
ORDER BY total_number_of_people_who_have_ordered DESC

;"""

pd.read_sql(q3, conn)
```

Out[36]:

	productName	total_number_of_people_who_have_ordered
0	1992 Ferrari 360 Spider red	28
1	The Titanic	22
2	2002 Suzuki XREO	22
3	1956 Porsche 356A Coupe	22
4	1937 Lincoln Berline	22
...	...	...
104	2001 Ferrari Enzo	15
105	1969 Corvair Monza	15
106	2002 Chevy Corvette	14
107	1958 Setra Bus	14
108	1957 Ford Thunderbird	14

109 rows × 2 columns

Select the Employee Number, First Name, Last Name, City (of the office), and Office Code of the Employees Who Sold Products That Have Been Ordered by Fewer Than 20 people.

This problem is a bit tougher. To start, think about how you might break the problem up. Be sure that your results only list each employee once.

In [61]:
```
# Your code here
q4 = """
SELECT
    e.employeeNumber, e.firstName, e.lastName, e.officeCode, o.city
FROM employees e, offices o, customers c
WHERE
    e.officeCode = o.officeCode AND
    c.salesRepEmployeeNumber = e.employeeNumber
GROUP BY
    employeeNumber, firstName, lastName, o.city, e.officeCode

HAVING COUNT(customerNumber) < 20

;"""

pd.read_sql(q4, conn)
```

Out[61]:

	employeeNumber	firstName	lastName	officeCode	city
0	1165	Leslie	Jennings	1	San Francisco
1	1166	Leslie	Thompson	1	San Francisco
2	1188	Julie	Firrelli	2	Boston
3	1216	Steve	Patterson	2	Boston
4	1286	Foon Yue	Tseng	3	NYC
5	1323	George	Vanauf	3	NYC
6	1337	Loui	Bondur	4	Paris
7	1370	Gerard	Hernandez	4	Paris
8	1401	Pamela	Castillo	4	Paris
9	1501	Larry	Bott	7	London
10	1504	Barry	Jones	7	London
11	1611	Andy	Fixter	6	Sydney
12	1612	Peter	Marsh	6	Sydney
13	1621	Mami	Nishi	5	Tokyo
14	1702	Martin	Gerard	4	Paris

Select the Employee Number, First Name, Last Name, and Number of Customers for Employees Whose Customers Have an Average Credit Limit Over 15K

In [65]:

```
# Your code here
q5 = """
SELECT e.employeeNumber, e.firstName, e.lastName, COUNT(c.customerNumber) AS Nu
FROM employees e, customers c
WHERE
    e.employeeNumber = c.salesRepEmployeeNumber
GROUP BY
    employeeNumber, firstName, lastName
HAVING
    AVG(creditLimit) > 15000

;"""

pd.read_sql(q5, conn)
```

Out[65]:

	employeeNumber	firstName	lastName	Number_of_customers
0	1165	Leslie	Jennings	6
1	1166	Leslie	Thompson	6
2	1188	Julie	Firrelli	6
3	1216	Steve	Patterson	6
4	1286	Foon Yue	Tseng	7
5	1323	George	Vanauf	8
6	1337	Loui	Bondur	6
7	1370	Gerard	Hernandez	7
8	1401	Pamela	Castillo	10
9	1501	Larry	Bott	8
10	1504	Barry	Jones	9
11	1611	Andy	Fixter	5
12	1612	Peter	Marsh	5
13	1621	Mami	Nishi	5
14	1702	Martin	Gerard	6

Summary

In this lesson, you got to practice some more complex SQL queries, some of which required subqueries. There's still plenty more SQL to be had though; hope you've been enjoying some of these puzzles!