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!

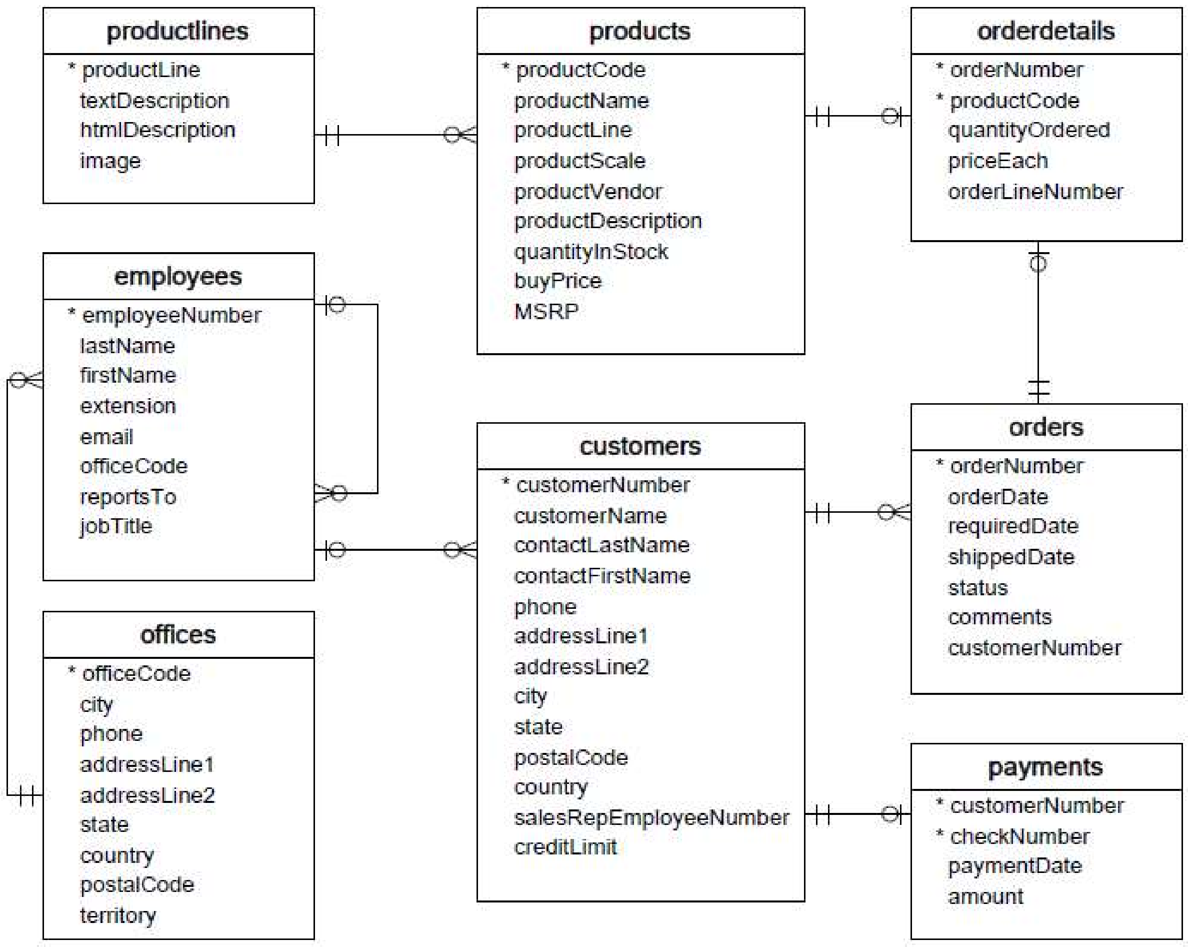
Objectives

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 [152]:
```
# Your code here; import the necessary packages
import pandas as pd
```

In [153]:
```
%%capture
!pip install ipython-sql sqlalchemy
import sqlalchemy
engine=sqlalchemy.create_engine("sqlite:///data2.sqlite")
%load_ext sql
%sql sqlite:///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 [154]:
```
##using join statement
%%sql
SELECT
    customerNumber,
    contactLastName,
    contactFirstName
FROM customers
JOIN orders
    USING(customerNumber)
WHERE orderDate = '2003-01-31'
;
```

```
   sqlite:///data.sqlite
 * sqlite:///data2.sqlite
Done.
```

Out[154]:

customerNumber	contactLastName	contactFirstName
141	Freyre	Diego

In [155]:
```
# Your code here using subquery
pd.read_sql('''
SELECT customerNumber, contactLastName,contactFirstname
FROM customers
WHERE customerNumber=(SELECT o.customerNumber
    FROM orders as o
        WHERE o.orderDate='2003-01-31')
''', engine)
```

Out[155]:

	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 [156]:
```
# Your code here using join
pd.read_sql('''
SELECT p.productName,COUNT(od.quantityOrdered) as total_number_orders
FROM products as p
INNER JOIN orderdetails as od on p.productCode=od.productCode
GROUP BY p.productName
ORDER BY total_number_orders DESC

''', engine)
```

Out[156]:

	productName	total_number_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 [157]:

```
q2 = """
WITH unique_customer AS (
    SELECT DISTINCT
        o.customerNumber,o.orderNumber
    FROM
        orders as o
)
SELECT
    p.productName,COUNT(uc.customerNumber) as total_orders
FROM
    orderdetails AS od
    INNER JOIN unique_customer AS uc
        ON uc.orderNumber=od.orderNumber
    INNER JOIN products as p
        on p.productCode=od.productCode
    GROUP BY p.productName
    ORDER BY total_orders DESC

;
"""
q2_result = pd.read_sql(q2, engine)
q2_result
```

Out[157]:

	productName	total_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 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 [172]:

```
pd.read_sql('''
WITH unique_customer AS (
    SELECT DISTINCT
        e.employeeNumber,e.firstName,e.lastName,o.officeCode,o.city,c.custome
    FROM
        customers as c
    INNER JOIN employees as e on e.employeeNumber=c.salesRepEmployeeNumber
    INNER JOIN offices as o on o.officeCode=e.officeCode
    GROUP BY e.employeeNumber
)
SELECT employeeNumber,firstname,lastName,city,officeCode
FROM unique_customer
;
''', engine)
```

Out[172]:

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

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

In [174]:

```
# Your code here
q4 = """
WITH average_credit_limit AS (
SELECT DISTINCT
    e.employeeNumber,e.lastName,e.firstName,avg(c.creditLimit) as average_cus
FROM
   employees as e
INNER JOIN customers as c
ON e.employeeNumber=c.salesRepEmployeeNumber
GROUP BY e.lastName
)

SELECT employeeNumber,lastName,firstName
FROM average_credit_limit
WHERE average_cust_credit_limit>15000
;
"""
q4_result = pd.read_sql(q4, engine)
q4_result
```

Out[174]:

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

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!