SQL_Lab_Assignment_2

October 3, 2022

1 SQL Subqueries - Lab Assignment #2

1.1 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!

1.2 Objectives

You will be able to:

- Write subqueries to decompose complex queries

1.3 CRM Database ERD

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

1.4 Connect to the Database

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

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

```python
[4]: # Your code here; create the connection
     conn = sqlite3.Connection("C:/Users/maxil/OneDrive - informatik.hs-fulda.de/
      ↪Studium Hochschule Fulda/Master/3. Semester (USA)/Technologies in Data␣
      ↪Analytics/DS311-Technologies-in-Data-Analytic/Week_4_SQL_Queries/
      ↪Lab_Assignment/data/data2.sqlite")
```

1.5 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'
;
```

```
[5]: # Your code here
     q0 = """
     SELECT customerNumber, contactLastName, contactFirstName
     FROM customers
     WHERE customerNumber
     IN (SELECT customerNumber FROM orders WHERE orderDate = '2003-01-31')
     ;
     """

     pd.read_sql(q0, conn)
```

```
[5]:    customerNumber contactLastName contactFirstName
     0             141          Freyre            Diego
```

1.6 Select the Total Number of Orders for Each Product Name

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

```
[6]: # Your code here
     q1 = """
     SELECT productName, SUM(quantityOrdered) AS NumberOfOrders
     FROM products
     INNER JOIN orderdetails USING(productCode)
     GROUP BY productName
     ORDER BY NumberOfOrders DESC
     ;
     """

     pd.read_sql(q1, conn)
```

```
[6]:                             productName  NumberOfOrders
     0              1992 Ferrari 360 Spider red            1808
     1                     1937 Lincoln Berline            1111
     2                American Airlines: MD-11S            1085
     3    1941 Chevrolet Special Deluxe Cabriolet           1076
     4              1930 Buick Marquette Phaeton           1074
     ..                                    ...             ...
     104            1999 Indy 500 Monte Carlo SS            855
     105                      1911 Ford Town Car            832
     106        1936 Mercedes Benz 500k Roadster            824
     107             1970 Chevy Chevelle SS 454            803
     108                   1957 Ford Thunderbird            767
```

```
[109 rows x 2 columns]
```

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

Sort the results in descending order.

1.7.1   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.

```python
[7]:  # Your code here
      # Hint: because one of the tables we'll be joining has duplicate customer␣
       ↪numbers, you should use DISTINCT
      q2 = """
      SELECT productName, COUNT(DISTINCT customerNumber) as NumberPeopleOrdered FROM␣
       ↪products
      INNER JOIN orderdetails USING(productCode)
      INNER JOIN orders USING(orderNumber)
      GROUP BY productName
      ORDER BY NumberPeopleOrdered DESC
      ;
      """

      pd.read_sql(q2, conn)
```

```
[7]:                             productName  NumberPeopleOrdered
      0                 1992 Ferrari 360 Spider red                   40
      1                          Boeing X-32A JSF                   27
      2                        1972 Alfa Romeo GTA                   27
      3                    1952 Alpine Renault 1300                   27
      4                          1934 Ford V8 Coupe                   27
      ..                                     ...                  ...
      104   1958 Chevy Corvette Limited Edition                   19
      105                     2002 Chevy Corvette                   18
      106                 1969 Chevrolet Camaro Z28                   18
      107                        1952 Citroen-15CV                   18
      108                        1949 Jaguar XK 120                   18

      [109 rows x 2 columns]
```

1.8 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.

```
[8]: # Your code here
     q3 = """
     SELECT employeeNumber, firstName, lastName, offices.city, officeCode from␣
       ↪employees
     INNER JOIN offices USING(officeCode)
     INNER JOIN customers on employeeNumber = salesRepEmployeeNumber
     INNER JOIN orders USING(customerNumber)
     GROUP BY employeeNumber
     HAVING COUNT(orders.customerNumber) < 20
     ;
     """

     pd.read_sql(q3, conn)
```

[8]:		employeeNumber	firstName	lastName	city	officeCode
	0	1166	Leslie	Thompson	San Francisco	1
	1	1188	Julie	Firrelli	Boston	2
	2	1216	Steve	Patterson	Boston	2
	3	1286	Foon Yue	Tseng	NYC	3
	4	1611	Andy	Fixter	Sydney	6
	5	1612	Peter	Marsh	Sydney	6
	6	1621	Mami	Nishi	Tokyo	5
	7	1702	Martin	Gerard	Paris	4

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

```
[9]: # Your code here
     q4 = """
     SELECT employeeNumber, firstName, lastName, COUNT(customerNumber) AS␣
       ↪NumberCustomers from employees
     INNER JOIN offices USING(officeCode)
     INNER JOIN customers on employeeNumber = salesRepEmployeeNumber
     GROUP BY employeeNumber
     HAVING AVG(creditLimit) > 15
     ORDER BY NumberCustomers DESC
     ;
     """

     pd.read_sql(q4, conn)
```

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

1.10 Summary

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