index

February 25, 2022

1 Join Statements - Lab

1.1 Introduction

In this lab, you'll practice your knowledge of `JOIN` statements, using various types of joins and various methods for specifying the links between them.

1.2 Objectives

You will be able to: * Write SQL queries that make use of various types of joins * Compare and contrast the various types of joins * Discuss how primary and foreign keys are used in SQL * Decide and perform whichever type of join is best for retrieving desired data

1.3 CRM ERD

In this lab, you'll use the same customer relationship management (CRM) database that you saw from the previous lesson.

1.4 Connecting to the Database

Import the necessary packages and connect to the database `'data.sqlite'`.

[2]:
```
# Your code here
import pandas as pd
import sqlite3
conn = sqlite3.connect("data.sqlite")
```

1.5 Select the names of all employees in Boston

Hint: join the employees and offices tables. Select the first and last name.

[3]:
```
# Your code here
q = """
SELECT firstname, lastname
FROM employees AS e
LEFT JOIN offices AS o
ON e.officeCode = o.officeCode
WHERE city = "Boston"
"""
pd.read_sql(q, conn)
```

[3]:
```
  firstName   lastName
0     Julie   Firrelli
1     Steve  Patterson
```

1.6 Are there any offices that have zero employees?

Hint: Combine the employees and offices tables and use a group by. Select the office code, city, and number of employees.

[4]:
```
# Your code here
q = """
SELECT *
FROM offices AS o
LEFT JOIN employees AS e
ON e.officeCode = o.officeCode
WHERE firstName is null
"""
pd.read_sql(q, conn)
```

[4]:
```
  officeCode    city            phone          addressLine1 addressLine2  \
0         27  Boston  +1 977 299 8345  105 Cambridge Street

  state country postalCode territory employeeNumber lastName firstName  \
0    MA     USA      02331        NA           None     None      None

  extension email officeCode reportsTo jobTitle
0      None  None       None      None     None
```

[5]:
```
### FROM GitHub

q = """
SELECT
    o.officeCode,
    o.city,
    COUNT(e.employeeNumber) AS n_employees
FROM offices AS o
LEFT JOIN employees AS e
    USING(officeCode)
GROUP BY officeCode
HAVING n_employees = 0
;
"""
pd.read_sql(q, conn)
```

[5]:
```
   officeCode    city  n_employees
0          27  Boston            0
```

1.7 Write 3 questions of your own and answer them

[16]:
```
# Answers will vary

# Example question:
"""
How many customers are there per office?
"""

q = """
SELECT o.officeCode, o.city, COUNT(c.customerNumber) as ctm_num
FROM offices as o
JOIN employees as e
on e.officeCode = o.officeCode
JOIN customers as c
on e.employeeNumber = c.salesRepEmployeeNumber
GROUP BY o.city
order by ctm_num DESC
"""

pd.read_sql(q,conn)
```

[16]:

	officeCode	city	ctm_num
0	4	Paris	29
1	7	London	17
2	3	NYC	15
3	1	San Francisco	12
4	2	Boston	12
5	6	Sydney	10
6	5	Tokyo	5

[19]:
```
"""
How much is average payment per city?
"""

# Your code here
q = """
SELECT of.city, AVG(p.amount) AS avg_payment
FROM offices AS of
JOIN employees AS e
USING(officeCode)
JOIN customers AS c
ON e.employeeNumber = c.salesRepEmployeeNumber
JOIN payments AS p
USING(customerNumber)
GROUP BY of.city
ORDER by avg_payment DESC
```

```
"""
pd.read_sql(q,conn)
```

[19]:

	city	avg_payment
0	San Francisco	39336.458235
1	Sydney	33576.432667
2	Tokyo	32650.719286
3	Paris	32404.240230
4	London	31531.569048
5	NYC	28989.715405
6	Boston	28823.528621

[ ]:
```
"""
Question 2
"""

# Your code here
```

[ ]:
```
"""
Question 3
"""

# Your code here
```

1.8 Level Up 1: Display the names of every individual product that each employee has sold

Hint: You will need to use multiple `JOIN` clauses to connect all the way from employee names to product names.

[33]:
```
# Your code here

q = """
SELECT e.firstName, COUNT(p.productName) AS num_products_sold

FROM products AS p

JOIN orderdetails AS od

    USING(productCode)

JOIN orders AS ord

    USING(orderNumber)

JOIN customers AS c
```

```
    USING(customerNumber)

JOIN employees AS e

    ON e.employeeNumber = c.salesRepEmployeeNumber

GROUP BY e.firstName

"""



# q = """

# SELECT firstName, lastName, employeeNumber, p.productName


# FROM employees as e

# JOIN customers as c

#     ON e.employeeNumber = c.salesRepEmployeeNumber

# JOIN orders as ord

#     USING(customerNumber)

# JOIN orderdetails AS od

#     USING(orderNumber)

# JOIN products AS p

#      USING(productCode)
# """

pd.read_sql(q, conn)
```

[33]:

	firstName	num_products_sold
0	Andy	185
1	Barry	220
2	Foon Yue	142
3	George	211
4	Gerard	396
5	Julie	124
6	Larry	236
7	Leslie	445

```
8       Loui                177
9       Mami                137
10    Martin                114
11    Pamela                272
12     Peter                185
13     Steve                152
```

1.9 Level Up 2: Display the number of products each employee has sold

Alphabetize the results by employee last name.

Hint: Use the `quantityOrdered` column from `orderDetails`. Also, think about how to group the data when some employees might have the same first or last name.

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

1.10 Level Up 3: Display the names employees who have sold more than 200 different products

Hint: this is different from the previous question because the quantity sold doesn't matter, only the number of different products

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

1.11 Summary

Congrats! You practiced using join statements and leveraged your foreign keys knowledge!

[ ]:
```

```