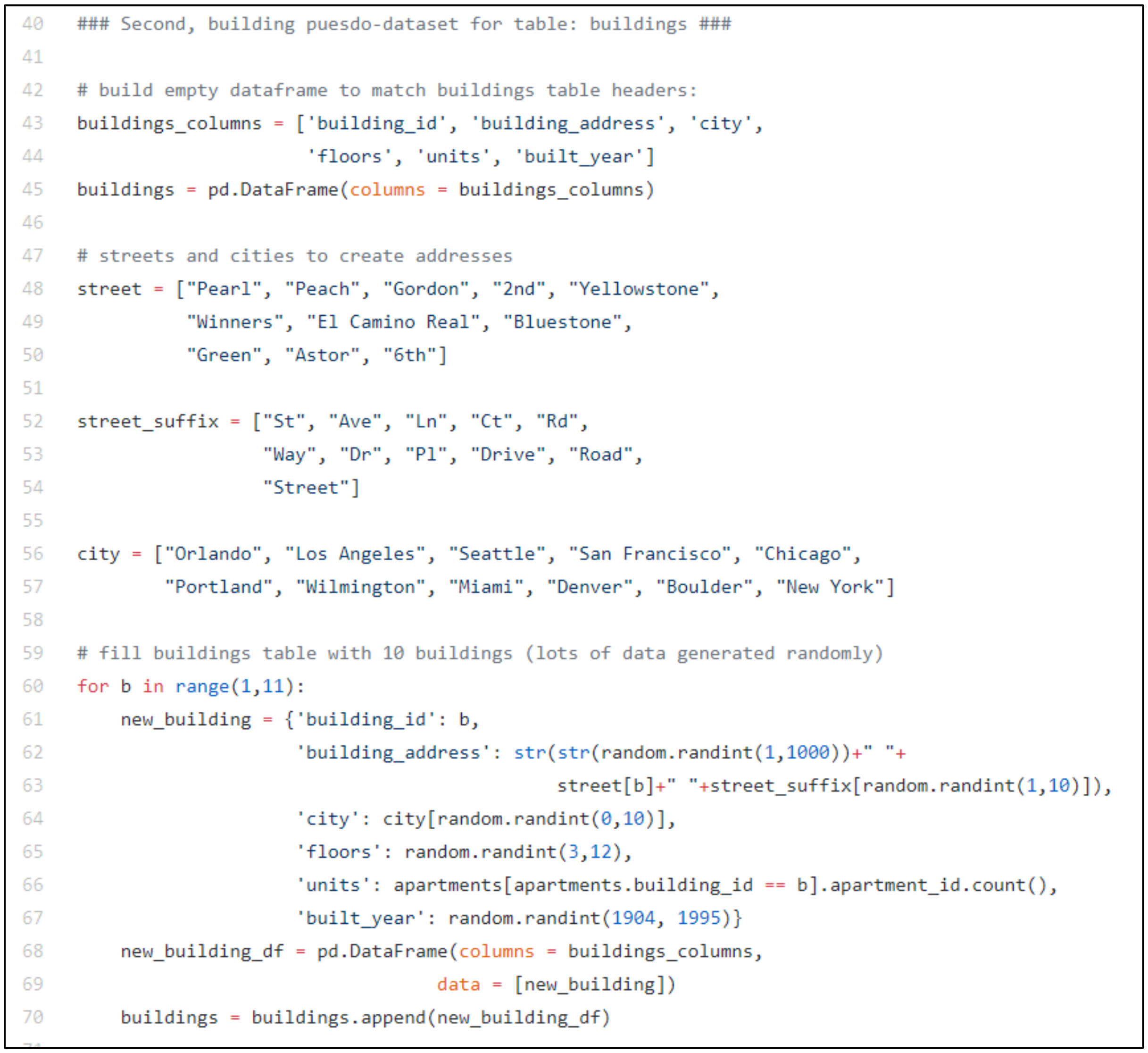
Author: Michael O’Donnell
Date: 10/23/19
Goal: Create pseudo-data in python, load into mySQL database as 3 tables, run practice join queries in mySQL

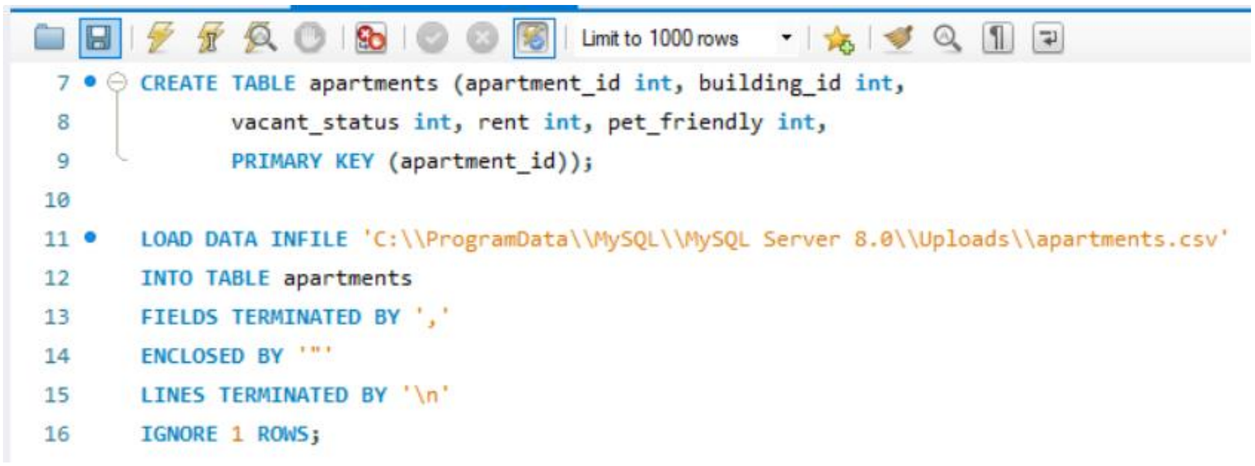
1. **Created pseudo-data in python with the following script:**
   https://github.com/odonnell31/python_data_science_basics/blob/master/SQL/generate_puesdo_data.py

   In the above script, 3 dataframes are created and exported as CSV files. Small sample snip below:

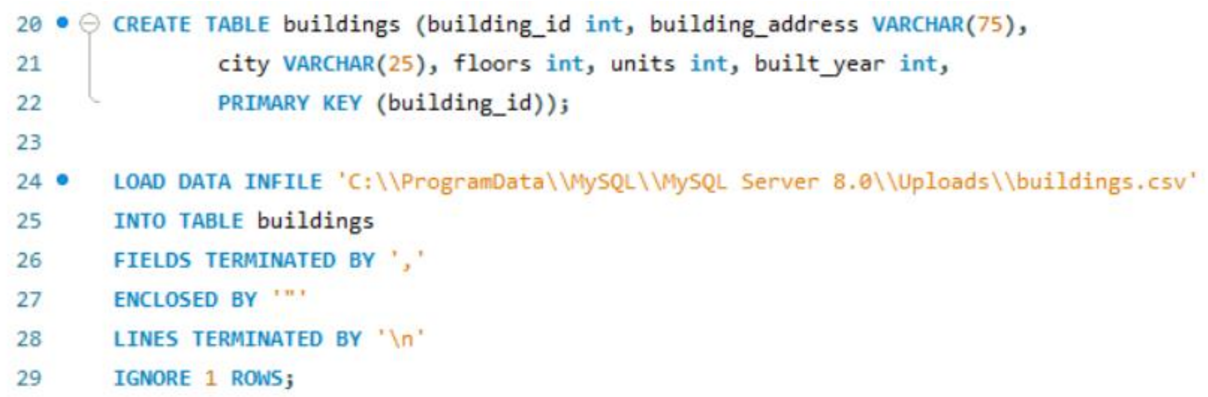


```
### Second, building puesdo-dataset for table: buildings ###

# build empty dataframe to match buildings table headers:
buildings_columns = ['building_id', 'building_address', 'city',
                     'floors', 'units', 'built_year']
buildings = pd.DataFrame(columns = buildings_columns)

# streets and cities to create addresses
street = ["Pearl", "Peach", "Gordon", "2nd", "Yellowstone",
          "Winners", "El Camino Real", "Bluestone",
          "Green", "Astor", "6th"]

street_suffix = ["St", "Ave", "Ln", "Ct", "Rd",
                 "Way", "Dr", "Pl", "Drive", "Road",
                 "Street"]

city = ["Orlando", "Los Angeles", "Seattle", "San Francisco", "Chicago",
        "Portland", "Wilmington", "Miami", "Denver", "Boulder", "New York"]

# fill buildings table with 10 buildings (lots of data generated randomly)
for b in range(1,11):
    new_building = {'building_id': b,
                    'building_address': str(str(random.randint(1,1000))+" "+
                                            street[b]+" "+street_suffix[random.randint(1,10)]),
                    'city': city[random.randint(0,10)],
                    'floors': random.randint(3,12),
                    'units': apartments[apartments.building_id == b].apartment_id.count(),
                    'built_year': random.randint(1904, 1995)}
    new_building_df = pd.DataFrame(columns = buildings_columns,
                                   data = [new_building])
    buildings = buildings.append(new_building_df)
```

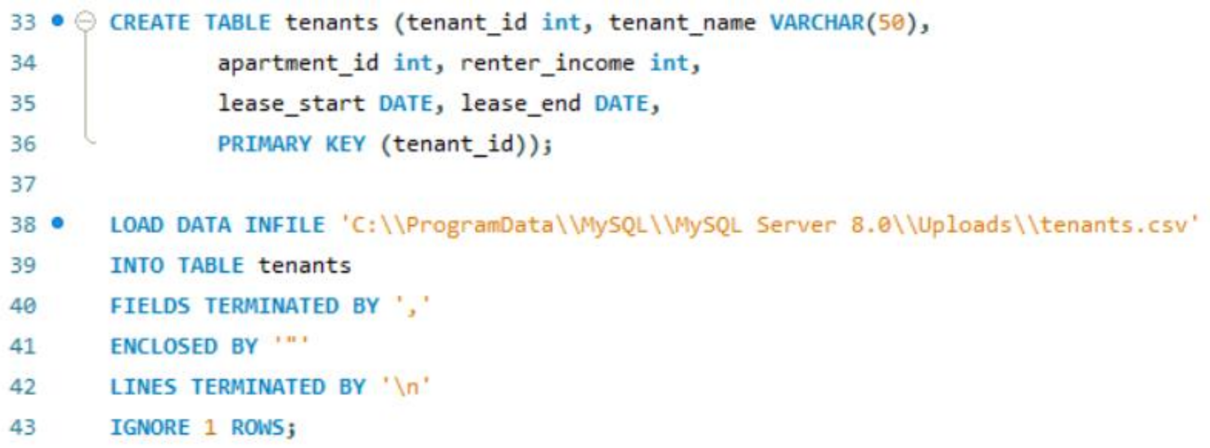
2. **Created 3 tables in mySQL and loaded them with the CSV’s created in step (1)**:
   https://github.com/odonnell31/python_data_science_basics/blob/master/SQL/apartment_create_tables.sql



```sql
CREATE TABLE apartments (apartment_id int, building_id int,
        vacant_status int, rent int, pet_friendly int,
        PRIMARY KEY (apartment_id));

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\apartments.csv'
INTO TABLE apartments
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

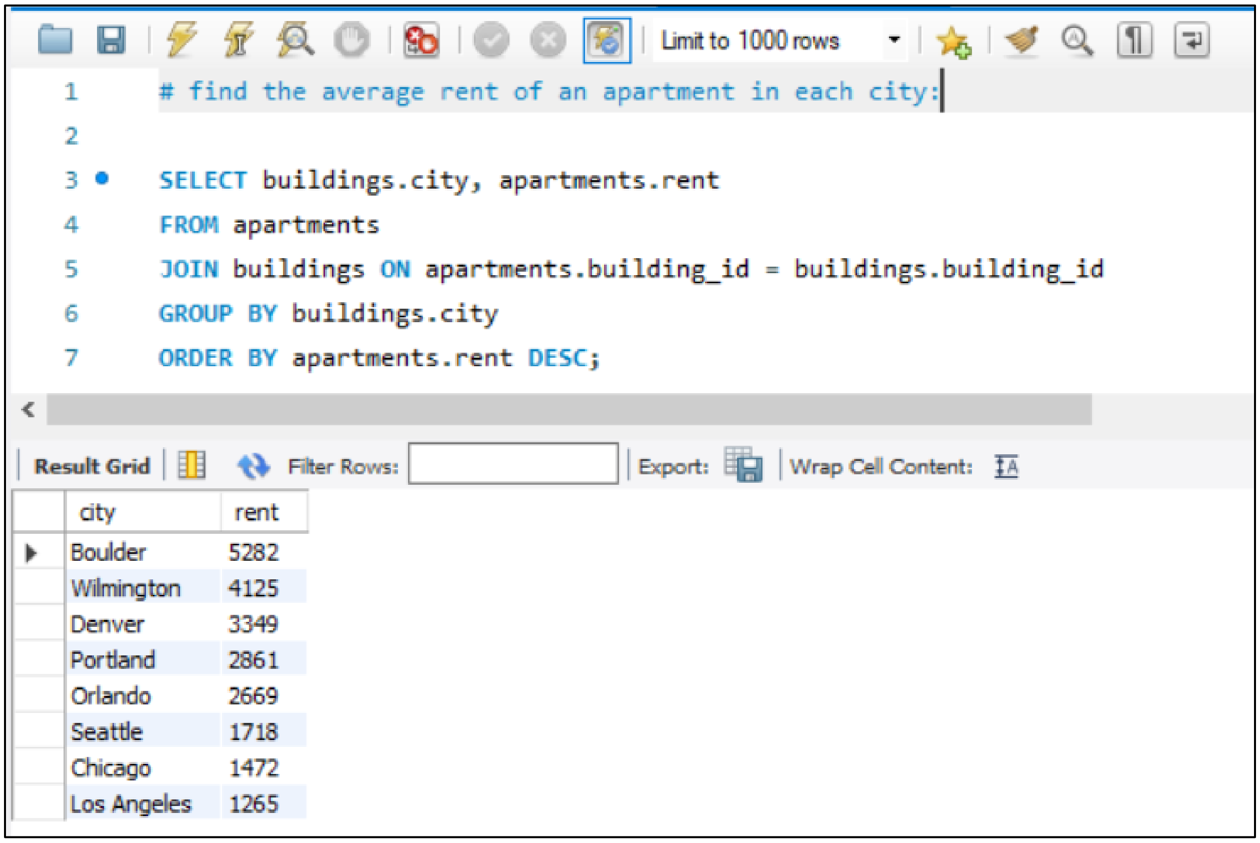


```sql
CREATE TABLE buildings (building_id int, building_address VARCHAR(75),
        city VARCHAR(25), floors int, units int, built_year int,
        PRIMARY KEY (building_id));

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\buildings.csv'
INTO TABLE buildings
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```



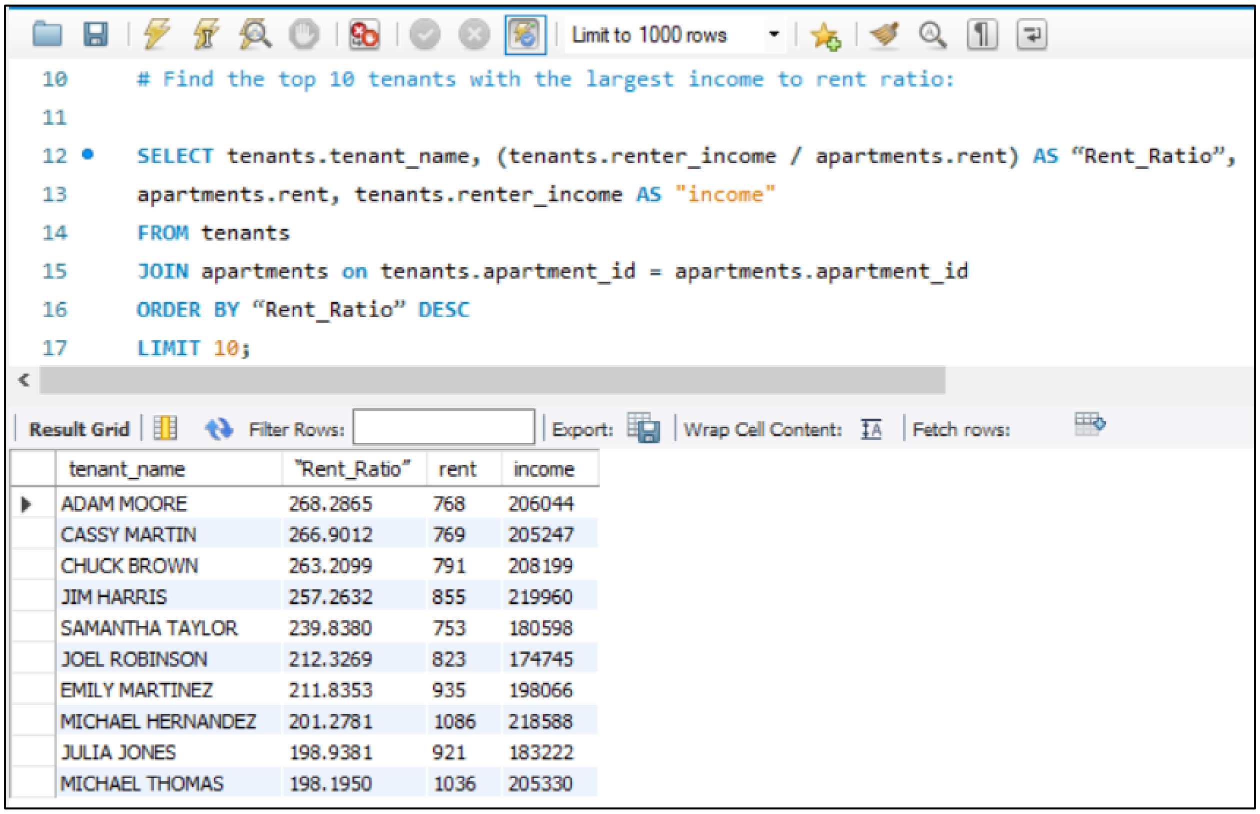
```sql
CREATE TABLE tenants (tenant_id int, tenant_name VARCHAR(50),
        apartment_id int, renter_income int,
        lease_start DATE, lease_end DATE,
        PRIMARY KEY (tenant_id));

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\tenants.csv'
INTO TABLE tenants
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

3. **Ran practice queries on the newly created mySQL database:**
   https://github.com/odonnell31/python_data_science_basics/blob/master/SQL/query_apartment_db.sql



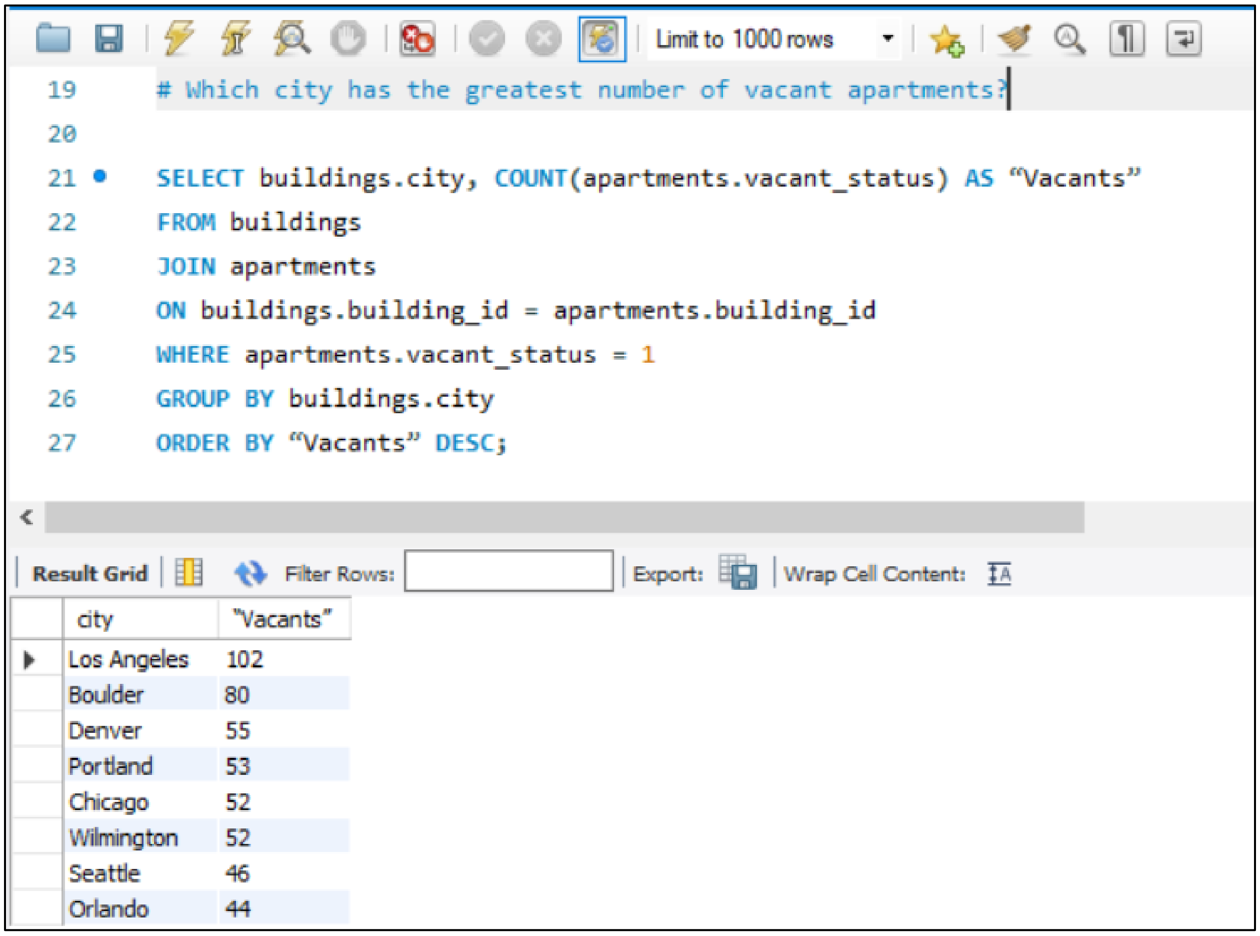
```
# find the average rent of an apartment in each city:

SELECT buildings.city, apartments.rent
FROM apartments
JOIN buildings ON apartments.building_id = buildings.building_id
GROUP BY buildings.city
ORDER BY apartments.rent DESC;
```

city	rent
Boulder	5282
Wilmington	4125
Denver	3349
Portland	2861
Orlando	2669
Seattle	1718
Chicago	1472
Los Angeles	1265



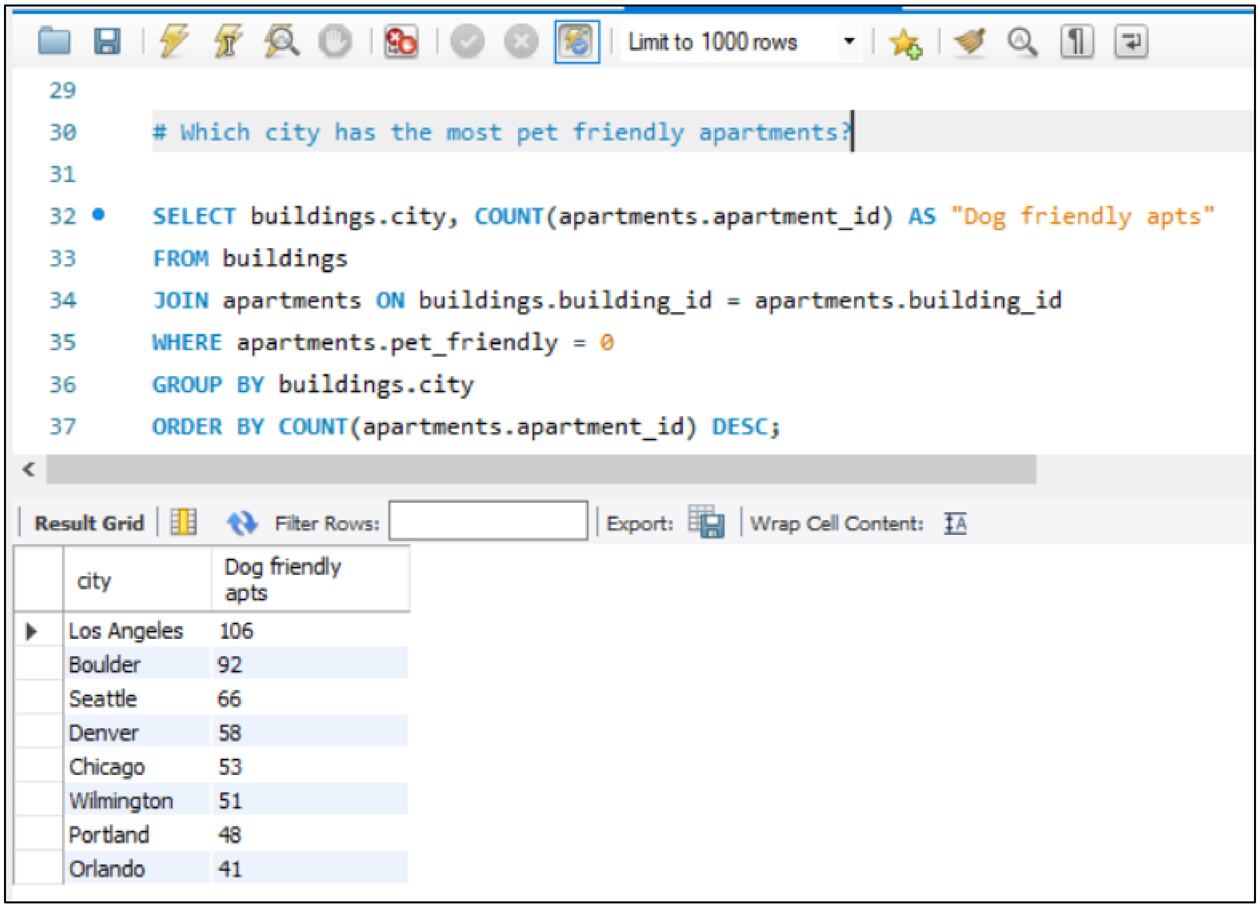
```
# Find the top 10 tenants with the largest income to rent ratio:

SELECT tenants.tenant_name, (tenants.renter_income / apartments.rent) AS “Rent_Ratio”,
apartments.rent, tenants.renter_income AS "income"
FROM tenants
JOIN apartments on tenants.apartment_id = apartments.apartment_id
ORDER BY “Rent_Ratio” DESC
LIMIT 10;
```

tenant_name	“Rent_Ratio”	rent	income
ADAM MOORE	268.2865	768	206044
CASSY MARTIN	266.9012	769	205247
CHUCK BROWN	263.2099	791	208199
JIM HARRIS	257.2632	855	219960
SAMANTHA TAYLOR	239.8380	753	180598
JOEL ROBINSON	212.3269	823	174745
EMILY MARTINEZ	211.8353	935	198066
MICHAEL HERNANDEZ	201.2781	1086	218588
JULIA JONES	198.9381	921	183222
MICHAEL THOMAS	198.1950	1036	205330



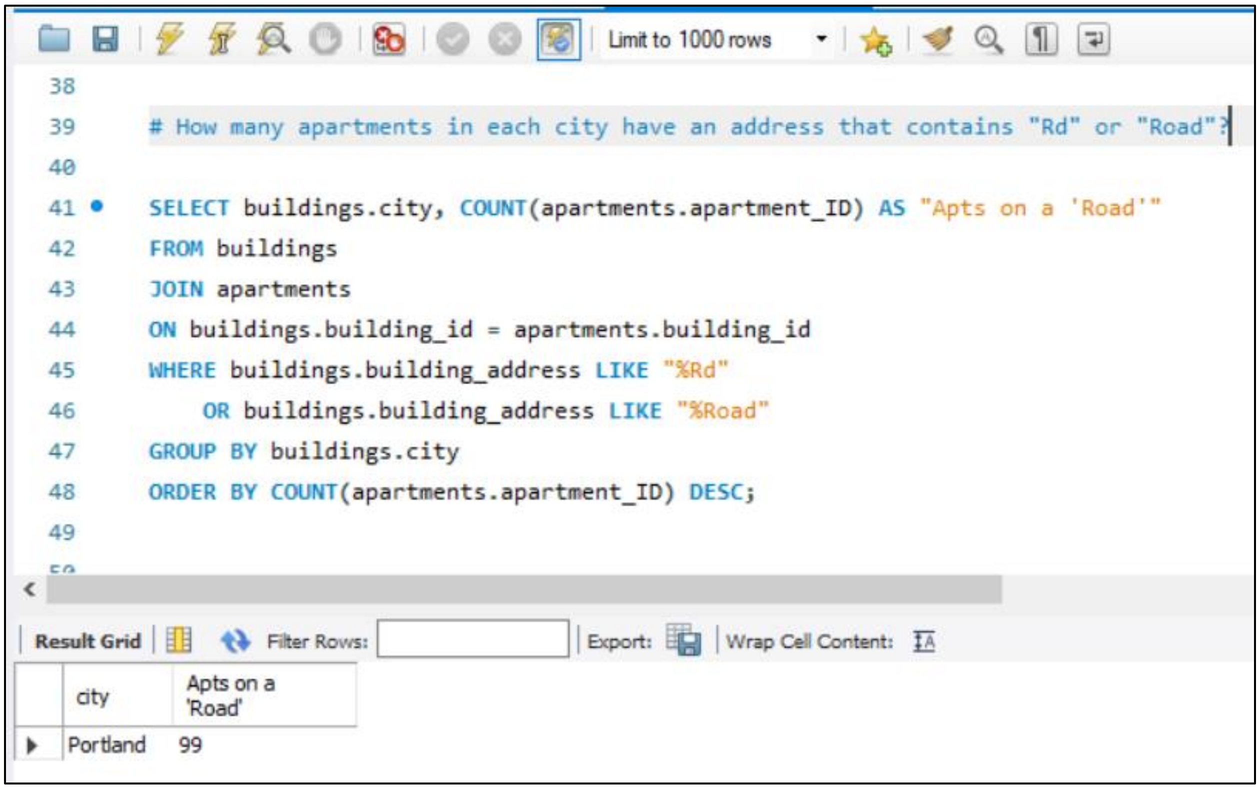
```
# Which city has the greatest number of vacant apartments?

SELECT buildings.city, COUNT(apartments.vacant_status) AS “Vacants”
FROM buildings
JOIN apartments
ON buildings.building_id = apartments.building_id
WHERE apartments.vacant_status = 1
GROUP BY buildings.city
ORDER BY “Vacants” DESC;
```

city	"Vacants"
Los Angeles	102
Boulder	80
Denver	55
Portland	53
Chicago	52
Wilmington	52
Seattle	46
Orlando	44



```sql
# Which city has the most pet friendly apartments?

SELECT buildings.city, COUNT(apartments.apartment_id) AS "Dog friendly apts"
FROM buildings
JOIN apartments ON buildings.building_id = apartments.building_id
WHERE apartments.pet_friendly = 0
GROUP BY buildings.city
ORDER BY COUNT(apartments.apartment_id) DESC;
```

city	Dog friendly apts
Los Angeles	106
Boulder	92
Seattle	66
Denver	58
Chicago	53
Wilmington	51
Portland	48
Orlando	41



```sql
# How many apartments in each city have an address that contains "Rd" or "Road"?

SELECT buildings.city, COUNT(apartments.apartment_ID) AS "Apts on a 'Road'"
FROM buildings
JOIN apartments
ON buildings.building_id = apartments.building_id
WHERE buildings.building_address LIKE "%Rd"
    OR buildings.building_address LIKE "%Road"
GROUP BY buildings.city
ORDER BY COUNT(apartments.apartment_ID) DESC;
```

city	Apts on a 'Road'
Portland	99