jupyter-labs-eda-sql-coursera_sqllite

February 27, 2025

Assignment: SQL Notebook for Peer Assignment

Estimated time needed: **60** minutes.

0.1 Introduction

Using this Python notebook you will:

1. Understand the Spacex DataSet
2. Load the dataset into the corresponding table in a Db2 database
3. Execute SQL queries to answer assignment questions

0.2 Overview of the DataSet

SpaceX has gained worldwide attention for a series of historic milestones.

It is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars wheras other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage.

Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

This dataset includes a record for each payload carried during a SpaceX mission into outer space.

0.2.1 Download the datasets

This assignment requires you to load the spacex dataset.

In many cases the dataset to be analyzed is available as a .CSV (comma separated values) file, perhaps on the internet. Click on the link below to download and save the dataset (.CSV file):

Spacex DataSet

[2]:
```
!pip install sqlalchemy==1.3.9
```

```
Requirement already satisfied: sqlalchemy==1.3.9 in
/opt/conda/lib/python3.12/site-packages (1.3.9)
```

0.2.2 Connect to the database

Let us first load the SQL extension and establish a connection with the database

[3]:
```
!pip install ipython-sql
!pip install ipython-sql prettytable
```

```
Collecting ipython-sql
  Downloading ipython_sql-0.5.0-py3-none-any.whl.metadata (17 kB)
Collecting prettytable (from ipython-sql)
  Downloading prettytable-3.15.0-py3-none-any.whl.metadata (33 kB)
Requirement already satisfied: ipython in /opt/conda/lib/python3.12/site-
packages (from ipython-sql) (8.31.0)
Collecting sqlalchemy>=2.0 (from ipython-sql)
  Downloading SQLAlchemy-2.0.38-cp312-cp312-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.6 kB)
Collecting sqlparse (from ipython-sql)
  Downloading sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Requirement already satisfied: six in /opt/conda/lib/python3.12/site-packages
(from ipython-sql) (1.17.0)
Requirement already satisfied: ipython-genutils in
/opt/conda/lib/python3.12/site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: greenlet!=0.4.17 in
/opt/conda/lib/python3.12/site-packages (from sqlalchemy>=2.0->ipython-sql)
(3.1.1)
Requirement already satisfied: typing-extensions>=4.6.0 in
/opt/conda/lib/python3.12/site-packages (from sqlalchemy>=2.0->ipython-sql)
(4.12.2)
Requirement already satisfied: decorator in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (0.19.2)
Requirement already satisfied: matplotlib-inline in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (4.9.0)
Requirement already satisfied: prompt_toolkit<3.1.0,>=3.0.41 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (3.0.50)
Requirement already satisfied: pygments>=2.4.0 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (2.19.1)
Requirement already satisfied: stack_data in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (0.6.3)
Requirement already satisfied: traitlets>=5.13.0 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (5.14.3)
Requirement already satisfied: wcwidth in /opt/conda/lib/python3.12/site-
packages (from prettytable->ipython-sql) (0.2.13)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in
/opt/conda/lib/python3.12/site-packages (from jedi>=0.16->ipython->ipython-sql)
(0.8.4)
```

```
Requirement already satisfied: ptyprocess>=0.5 in
/opt/conda/lib/python3.12/site-packages (from pexpect>4.3->ipython->ipython-sql)
(0.7.0)
Requirement already satisfied: executing>=1.2.0 in
/opt/conda/lib/python3.12/site-packages (from stack_data->ipython->ipython-sql)
(2.1.0)
Requirement already satisfied: asttokens>=2.1.0 in
/opt/conda/lib/python3.12/site-packages (from stack_data->ipython->ipython-sql)
(3.0.0)
Requirement already satisfied: pure_eval in /opt/conda/lib/python3.12/site-
packages (from stack_data->ipython->ipython-sql) (0.2.3)
Downloading ipython_sql-0.5.0-py3-none-any.whl (20 kB)
Downloading
SQLAlchemy-2.0.38-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(3.3 MB)
                         3.3/3.3 MB
127.5 MB/s eta 0:00:00
Downloading prettytable-3.15.0-py3-none-any.whl (33 kB)
Downloading sqlparse-0.5.3-py3-none-any.whl (44 kB)
Installing collected packages: sqlparse, sqlalchemy, prettytable, ipython-sql
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 1.3.9
    Uninstalling SQLAlchemy-1.3.9:
      Successfully uninstalled SQLAlchemy-1.3.9
Successfully installed ipython-sql-0.5.0 prettytable-3.15.0 sqlalchemy-2.0.38
sqlparse-0.5.3
Requirement already satisfied: ipython-sql in /opt/conda/lib/python3.12/site-
packages (0.5.0)
Requirement already satisfied: prettytable in /opt/conda/lib/python3.12/site-
packages (3.15.0)
Requirement already satisfied: ipython in /opt/conda/lib/python3.12/site-
packages (from ipython-sql) (8.31.0)
Requirement already satisfied: sqlalchemy>=2.0 in
/opt/conda/lib/python3.12/site-packages (from ipython-sql) (2.0.38)
Requirement already satisfied: sqlparse in /opt/conda/lib/python3.12/site-
packages (from ipython-sql) (0.5.3)
Requirement already satisfied: six in /opt/conda/lib/python3.12/site-packages
(from ipython-sql) (1.17.0)
Requirement already satisfied: ipython-genutils in
/opt/conda/lib/python3.12/site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: wcwidth in /opt/conda/lib/python3.12/site-
packages (from prettytable) (0.2.13)
Requirement already satisfied: greenlet!=0.4.17 in
/opt/conda/lib/python3.12/site-packages (from sqlalchemy>=2.0->ipython-sql)
(3.1.1)
Requirement already satisfied: typing-extensions>=4.6.0 in
/opt/conda/lib/python3.12/site-packages (from sqlalchemy>=2.0->ipython-sql)
(4.12.2)
```

```
Requirement already satisfied: decorator in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (0.19.2)
Requirement already satisfied: matplotlib-inline in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (4.9.0)
Requirement already satisfied: prompt_toolkit<3.1.0,>=3.0.41 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (3.0.50)
Requirement already satisfied: pygments>=2.4.0 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (2.19.1)
Requirement already satisfied: stack_data in /opt/conda/lib/python3.12/site-
packages (from ipython->ipython-sql) (0.6.3)
Requirement already satisfied: traitlets>=5.13.0 in
/opt/conda/lib/python3.12/site-packages (from ipython->ipython-sql) (5.14.3)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in
/opt/conda/lib/python3.12/site-packages (from jedi>=0.16->ipython->ipython-sql)
(0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in
/opt/conda/lib/python3.12/site-packages (from pexpect>4.3->ipython->ipython-sql)
(0.7.0)
Requirement already satisfied: executing>=1.2.0 in
/opt/conda/lib/python3.12/site-packages (from stack_data->ipython->ipython-sql)
(2.1.0)
Requirement already satisfied: asttokens>=2.1.0 in
/opt/conda/lib/python3.12/site-packages (from stack_data->ipython->ipython-sql)
(3.0.0)
Requirement already satisfied: pure_eval in /opt/conda/lib/python3.12/site-
packages (from stack_data->ipython->ipython-sql) (0.2.3)
```

[4]:
```
%load_ext sql
```

[5]:
```
import csv, sqlite3
import prettytable
prettytable.DEFAULT = 'DEFAULT'

con = sqlite3.connect("my_data1.db")
cur = con.cursor()
```

[6]:
```
!pip install -q pandas
```

[7]:
```
%sql sqlite:///my_data1.db
```

[8]:
```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.
↪cloud/IBM-DS0321EN-SkillsNetwork/labs/module_2/data/Spacex.csv")
```

```
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```

[8]: 101

Note:This below code is added to remove blank rows from table

[9]:
```
#DROP THE TABLE IF EXISTS

%sql DROP TABLE IF EXISTS SPACEXTABLE;
```

```
 * sqlite:///my_data1.db
Done.
```

[9]: []

[10]:
```
%sql create table SPACEXTABLE as select * from SPACEXTBL where Date is not null
```

```
 * sqlite:///my_data1.db
Done.
```

[10]: []

0.3 Tasks

Now write and execute SQL queries to solve the assignment tasks.

Note: If the column names are in mixed case enclose it in double quotes For Example "Landing_Outcome"

0.3.1 Task 1

Display the names of the unique launch sites in the space mission

[11]:
```
# Task 1: Unique launch sites
print("Task 1:")
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

```
Task 1:
 * sqlite:///my_data1.db
Done.
```

[11]: [('CCAFS LC-40',), ('VAFB SLC-4E',), ('KSC LC-39A',), ('CCAFS SLC-40',)]

0.3.2 Task 2

Display 5 records where launch sites begin with the string 'CCA'

[12]:
```
print("\nTask 2:")
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

```
Task 2:
 * sqlite:///my_data1.db
Done.
```

[12]:
```
[('2010-06-04', '18:45:00', 'F9 v1.0  B0003', 'CCAFS LC-40', 'Dragon Spacecraft
Qualification Unit', 0, 'LEO', 'SpaceX', 'Success', 'Failure (parachute)'),
 ('2010-12-08', '15:43:00', 'F9 v1.0  B0004', 'CCAFS LC-40', 'Dragon demo flight
C1, two CubeSats, barrel of Brouere cheese', 0, 'LEO (ISS)', 'NASA (COTS) NRO',
'Success', 'Failure (parachute)'),
 ('2012-05-22', '7:44:00', 'F9 v1.0  B0005', 'CCAFS LC-40', 'Dragon demo flight
C2', 525, 'LEO (ISS)', 'NASA (COTS)', 'Success', 'No attempt'),
 ('2012-10-08', '0:35:00', 'F9 v1.0  B0006', 'CCAFS LC-40', 'SpaceX CRS-1', 500,
'LEO (ISS)', 'NASA (CRS)', 'Success', 'No attempt'),
 ('2013-03-01', '15:10:00', 'F9 v1.0  B0007', 'CCAFS LC-40', 'SpaceX CRS-2',
677, 'LEO (ISS)', 'NASA (CRS)', 'Success', 'No attempt')]
```

0.3.3 Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

[13]:
```
print("\nTask 3:")
%sql SELECT SUM("PAYLOAD_MASS") FROM SPACEXTABLE WHERE "Customer" = 'NASA␣
 ↪(CRS)';
```

```
Task 3:
 * sqlite:///my_data1.db
Done.
```

[13]:
```
[(0.0,)]
```

0.3.4 Task 4

Display average payload mass carried by booster version F9 v1.1

[14]:
```
print("\nTask 4:")
%sql SELECT AVG("PAYLOAD_MASS") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9␣
 ↪v1.1';
```

```
Task 4:
 * sqlite:///my_data1.db
Done.
```

[14]:
```
[(0.0,)]
```

0.3.5 Task 5

List the date when the first succesful landing outcome in ground pad was acheived.
Hint:Use min function

[15]:
```
print("\nTask 5:")
%sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success␣
 ↪(ground pad)';
```

```
Task 5:
 * sqlite:///my_data1.db
Done.
```

[15]:
```
[('2015-12-22',)]
```

0.3.6 Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

[16]:
```
print("\nTask 6:")
%sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE \
WHERE "Landing_Outcome" = 'Success (drone ship)' \
AND "PAYLOAD_MASS" > 4000 AND "PAYLOAD_MASS" < 6000;
```

```
Task 6:
 * sqlite:///my_data1.db
Done.
```

[16]:
```
[]
```

0.3.7 Task 7

List the total number of successful and failure mission outcomes

[17]:
```
print("\nTask 7:")
%sql SELECT "Mission_Outcome", COUNT(*) AS Count \
FROM SPACEXTABLE GROUP BY "Mission_Outcome";
```

```
Task 7:
 * sqlite:///my_data1.db
Done.
```

[17]:
```
[('Failure (in flight)', 1),
 ('Success', 98),
 ('Success ', 1),
 ('Success (payload status unclear)', 1)]
```

0.3.8 Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

[18]:
```
print("\nTask 8:")
%sql SELECT "Booster_Version" FROM SPACEXTABLE \
WHERE "PAYLOAD_MASS" = (SELECT MAX("PAYLOAD_MASS") FROM SPACEXTABLE);
```

```
Task 8:
```

```
 * sqlite:///my_data1.db
Done.
```

[18]:
```
[('F9 v1.0  B0003',),
 ('F9 v1.0  B0004',),
 ('F9 v1.0  B0005',),
 ('F9 v1.0  B0006',),
 ('F9 v1.0  B0007',),
 ('F9 v1.1  B1003',),
 ('F9 v1.1',),
 ('F9 v1.1',),
 ('F9 v1.1',),
 ('F9 v1.1',),
 ('F9 v1.1',),
 ('F9 v1.1 B1011',),
 ('F9 v1.1 B1010',),
 ('F9 v1.1 B1012',),
 ('F9 v1.1 B1013',),
 ('F9 v1.1 B1014',),
 ('F9 v1.1 B1015',),
 ('F9 v1.1 B1016',),
 ('F9 v1.1 B1018',),
 ('F9 FT B1019',),
 ('F9 v1.1 B1017',),
 ('F9 FT B1020',),
 ('F9 FT B1021.1',),
 ('F9 FT B1022',),
 ('F9 FT B1023.1',),
 ('F9 FT B1024',),
 ('F9 FT B1025.1',),
 ('F9 FT B1026',),
 ('F9 FT B1029.1',),
 ('F9 FT B1031.1',),
 ('F9 FT B1030',),
 ('F9 FT  B1021.2',),
 ('F9 FT B1032.1',),
 ('F9 FT B1034',),
 ('F9 FT B1035.1',),
 ('F9 FT  B1029.2',),
 ('F9 FT B1036.1',),
 ('F9 FT B1037',),
 ('F9 B4 B1039.1',),
 ('F9 FT B1038.1',),
 ('F9 B4 B1040.1',),
 ('F9 B4 B1041.1',),
 ('F9 FT  B1031.2',),
 ('F9 B4 B1042.1',),
```

```
('F9 FT  B1035.2',),
('F9 FT  B1036.2',),
('F9 B4 B1043.1',),
('F9 FT  B1032.2',),
('F9 FT  B1038.2',),
('F9 B4 B1044',),
('F9 B4  B1041.2',),
('F9 B4  B1039.2',),
('F9 B4 B1045.1',),
('F9 B5  B1046.1',),
('F9 B4  B1043.2',),
('F9 B4  B1040.2',),
('F9 B4 B1045.2',),
('F9 B5B1047.1',),
('F9 B5B1048.1',),
('F9 B5 B1046.2',),
('F9 B5B1049.1',),
('F9 B5 B1048.2',),
('F9 B5 B1047.2',),
('F9 B5 B1046.3',),
('F9 B5B1050',),
('F9 B5B1054',),
('F9 B5 B1049.2',),
('F9 B5 B1048.3',),
('F9 B5B1051.1',),
('F9 B5B1056.1 ',),
('F9 B5 B1049.3',),
('F9 B5 B1051.2 ',),
('F9 B5 B1056.2 ',),
('F9 B5 B1047.3 ',),
('F9 B5 B1048.4',),
('F9 B5B1059.1',),
('F9 B5 B1056.3 ',),
('F9 B5 B1049.4',),
('F9 B5 B1046.4',),
('F9 B5 B1051.3',),
('F9 B5 B1056.4',),
('F9 B5 B1059.2',),
('F9 B5 B1048.5',),
('F9 B5 B1051.4',),
('F9 B5B1058.1 ',),
('F9 B5 B1049.5',),
('F9 B5 B1059.3',),
('F9 B5B1060.1',),
('F9 B5 B1058.2 ',),
('F9 B5 B1051.5',),
('F9 B5 B1049.6',),
```

```
 ('F9 B5 B1059.4',),
 ('F9 B5 B1060.2 ',),
 ('F9 B5 B1058.3 ',),
 ('F9 B5 B1051.6',),
 ('F9 B5 B1060.3',),
 ('F9 B5B1062.1',),
 ('F9 B5B1061.1 ',),
 ('F9 B5B1063.1',),
 ('F9 B5 B1049.7 ',),
 ('F9 B5 B1058.4 ',)]
```

0.3.9 Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015. Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

[19]:
```
%sql SELECT substr(Date, 6,2) AS Month, "Booster_Version", "Launch_Site" \
FROM SPACEXTABLE \
WHERE substr(Date,0,5) = '2015' \
AND "Landing_Outcome" = 'Failure (drone ship)';
```

```
 * sqlite:///my_data1.db
Done.
```

[19]:
```
[('01', 'F9 v1.1 B1012', 'CCAFS LC-40'),
 ('04', 'F9 v1.1 B1015', 'CCAFS LC-40')]
```

0.3.10 Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

[20]:
```
print("\nTask 10:")
%sql SELECT "Landing_Outcome", COUNT(*) AS Count, \
RANK() OVER (ORDER BY COUNT(*) DESC) AS Outcome_Rank \
FROM SPACEXTABLE \
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY "Landing_Outcome";
```

```
Task 10:
 * sqlite:///my_data1.db
Done.
```

[20]:
```
[('No attempt', 10, 1),
 ('Success (drone ship)', 5, 2),
 ('Failure (drone ship)', 5, 2),
 ('Success (ground pad)', 3, 4),
```

```
('Controlled (ocean)', 3, 4),
('Uncontrolled (ocean)', 2, 6),
('Failure (parachute)', 2, 6),
('Precluded (drone ship)', 1, 8)]
```

0.3.11 Reference Links

- Hands-on Lab : String Patterns, Sorting and Grouping
- Hands-on Lab: Built-in functions
- Hands-on Lab : Sub-queries and Nested SELECT Statements
- Hands-on Tutorial: Accessing Databases with SQL magic
- Hands-on Lab: Analyzing a real World Data Set

0.4 Author(s)

Lakshmi Holla

0.5 Other Contributors

Rav Ahuja

##