jupyter-labs-eda-sql-coursera_sqllite

June 24, 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

[1]:
```
!pip install sqlalchemy==1.3.9
```

```
Collecting sqlalchemy==1.3.9
  Downloading SQLAlchemy-1.3.9.tar.gz (6.0 MB)
                             6.0/6.0 MB
104.5 MB/s eta 0:00:00
  Preparing metadata (setup.py) … one
```

```
Building wheels for collected packages: sqlalchemy
  Building wheel for sqlalchemy (setup.py) …done
  Created wheel for sqlalchemy:
filename=SQLAlchemy-1.3.9-cp312-cp312-linux_x86_64.whl size=1160111
sha256=2d0ea2b098f93e9d4339c91b7b5edbf5b27261fe8e29ad7adb5c3c10ceee19d4
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/b3/1c/42/0e26b8d512adc
6bce10ff71a05229366b4ccec641cd3b42111
Successfully built sqlalchemy
Installing collected packages: sqlalchemy
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 2.0.37
    Uninstalling SQLAlchemy-2.0.37:
      Successfully uninstalled SQLAlchemy-2.0.37
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
jupyterhub 5.2.1 requires SQLAlchemy>=1.4.1, but you have sqlalchemy 1.3.9 which
is incompatible.
Successfully installed sqlalchemy-1.3.9
```

0.2.2 Connect to the database

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

[2]:
```
!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-3.16.0-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-2.0.41-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>=1 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-2.0.41-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(3.3 MB)
                         3.3/3.3 MB
68.1 MB/s eta 0:00:00
Downloading prettytable-3.16.0-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-3.16.0 sqlalchemy-2.0.41
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 (3.16.0)
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) (2.0.41)
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>=1 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
```

```
The sql extension is already loaded. To reload it, use:
  %reload_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

[13]:
```
%sql select Distinct LAUNCH_SITE from SPACEXTBL;
```

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

[13]:
```
[('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'

[15]:
```
%sql select LAUNCH_SITE from SPACEXTBL where LAUNCH_SITE like "CCA%" limit 5
```

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

[15]:
```
[('CCAFS LC-40',),
 ('CCAFS LC-40',),
 ('CCAFS LC-40',),
 ('CCAFS LC-40',),
 ('CCAFS LC-40',)]
```

0.3.3 Task 3

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

[21]:
```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTBL where "Customer" like "NASA␣
↪(CRS)%"
```

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

[21]:
```
[(48213,)]
```

0.3.4 Task 4

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

[22]:
```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTBL where "Booster_Version" like␣
↪"F9 v1.1%"
```

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

[22]:
```
[(2534.6666666666665,)]
```

0.3.5 Task 5

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

[26]:
```
%sql select min("Date") from SPACEXTBL where "Landing_Outcome" = "Success␣
↪(ground pad)"
```

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

[26]:
```
[('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

[28]:
```
%sql Select "Booster_Version" from SPACEXTBL where "Landing_Outcome"="Success␣
↪(drone ship)" and "PAYLOAD_MASS__KG_" between 4000 and 6000
```

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

[28]:
```
[('F9 FT B1022',), ('F9 FT B1026',), ('F9 FT  B1021.2',), ('F9 FT  B1031.2',)]
```

0.3.7 Task 7

List the total number of successful and failure mission outcomes

[29]:
```
%sql select count (Mission_Outcome) as missionoutcome from SPACEXTBL Group by␣
↪Mission_Outcome
```

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

[29]:
```
[(1,), (98,), (1,), (1,)]
```

0.3.8 Task 8

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

[36]:
```
%sql select Booster_version as BoosterVersion, PAYLOAD_MASS__KG_ as payload␣
↪from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from␣
↪SPACEXTBL)
```

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

[36]:
```
[('F9 B5 B1048.4', 15600),
 ('F9 B5 B1049.4', 15600),
 ('F9 B5 B1051.3', 15600),
 ('F9 B5 B1056.4', 15600),
 ('F9 B5 B1048.5', 15600),
 ('F9 B5 B1051.4', 15600),
 ('F9 B5 B1049.5', 15600),
 ('F9 B5 B1060.2 ', 15600),
```

```
 ('F9 B5 B1058.3 ', 15600),
 ('F9 B5 B1051.6', 15600),
 ('F9 B5 B1060.3', 15600),
 ('F9 B5 B1049.7 ', 15600)]
```

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.

[37]:
```
%sql Select CASE SUBSTR("Date", 6, 2) when "01" then "January" when "02" then␣
 ↪"February" when "03" then "MArch" when "04" then "April" when "05" then␣
 ↪"May" when "06" then "June" when "07" then "July" when "08" then "August"␣
 ↪when "09" then "September" when "10" then "October" when "11" then␣
 ↪"November" when "12" then "December" ELSE "Unknown" END AS month,␣
 ↪"Landing_Outcome"= "Failure (drone ship)" , "Booster_Version", "Launch_Site"␣
 ↪from SPACEXTBL Where substr ("Date", 0, 5) = "2015"
```

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

[37]:
```
[('January', 1, 'F9 v1.1 B1012', 'CCAFS LC-40'),
 ('February', 0, 'F9 v1.1 B1013', 'CCAFS LC-40'),
 ('MArch', 0, 'F9 v1.1 B1014', 'CCAFS LC-40'),
 ('April', 1, 'F9 v1.1 B1015', 'CCAFS LC-40'),
 ('April', 0, 'F9 v1.1 B1016', 'CCAFS LC-40'),
 ('June', 0, 'F9 v1.1 B1018', 'CCAFS LC-40'),
 ('December', 0, 'F9 FT B1019', '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.

[47]:
```
%sql select "Landing_Outcome", count("Landing_Outcome") from SPACEXTBL where␣
 ↪"Date" between 20100604 and 20170320 group by "Landing_Outcome" order by 2␣
 ↪desc
```

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

[47]:
```
[('Success (drone ship)', 12),
 ('No attempt', 12),
 ('Success (ground pad)', 8),
 ('Failure (drone ship)', 5),
 ('Controlled (ocean)', 4),
 ('Uncontrolled (ocean)', 2),
 ('Precluded (drone ship)', 1)]
```

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

##