Assignment: SQL Notebook for Peer Assignment

Estimated time needed: **60** minutes.

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

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.

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

In [39]:
```
!pip install sqlalchemy==1.3.9
```

```
Collecting sqlalchemy==1.3.9
  Using cached SQLAlchemy-1.3.9-cp311-cp311-linux_x86_64.whl
Installing collected packages: sqlalchemy
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 2.0.36
    Uninstalling SQLAlchemy-2.0.36:
      Successfully uninstalled SQLAlchemy-2.0.36
ERROR: pip's dependency resolver does not currently take into account all the packag
es that are installed. This behaviour is the source of the following dependency conf
licts.
ipython-sql 0.5.0 requires sqlalchemy>=2.0, but you have sqlalchemy 1.3.9 which is i
ncompatible.
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
```

Connect to the database

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

In [40]:
```
!pip install ipython-sql
!pip install ipython-sql prettytable
```

```
Requirement already satisfied: ipython-sql in /opt/conda/lib/python3.11/site-package
s (0.5.0)
Requirement already satisfied: prettytable in /opt/conda/lib/python3.11/site-package
s (from ipython-sql) (3.12.0)
Requirement already satisfied: ipython in /opt/conda/lib/python3.11/site-packages (f
rom ipython-sql) (8.22.2)
Collecting sqlalchemy>=2.0 (from ipython-sql)
  Using cached SQLAlchemy-2.0.36-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86
_64.whl.metadata (9.7 kB)
Requirement already satisfied: sqlparse in /opt/conda/lib/python3.11/site-packages
(from ipython-sql) (0.5.3)
Requirement already satisfied: six in /opt/conda/lib/python3.11/site-packages (from
ipython-sql) (1.16.0)
Requirement already satisfied: ipython-genutils in /opt/conda/lib/python3.11/site-pa
ckages (from ipython-sql) (0.2.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /opt/conda/lib/python3.1
1/site-packages (from sqlalchemy>=2.0->ipython-sql) (4.12.2)
Requirement already satisfied: greenlet!=0.4.17 in /opt/conda/lib/python3.11/site-pa
ckages (from sqlalchemy>=2.0->ipython-sql) (3.0.3)
Requirement already satisfied: decorator in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (0.19.1)
Requirement already satisfied: matplotlib-inline in /opt/conda/lib/python3.11/site-p
ackages (from ipython->ipython-sql) (0.1.7)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.41 in /opt/conda/lib/pytho
n3.11/site-packages (from ipython->ipython-sql) (3.0.42)
Requirement already satisfied: pygments>=2.4.0 in /opt/conda/lib/python3.11/site-pac
kages (from ipython->ipython-sql) (2.18.0)
Requirement already satisfied: stack-data in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (0.6.2)
Requirement already satisfied: traitlets>=5.13.0 in /opt/conda/lib/python3.11/site-p
ackages (from ipython->ipython-sql) (5.14.3)
Requirement already satisfied: pexpect>4.3 in /opt/conda/lib/python3.11/site-package
s (from ipython->ipython-sql) (4.9.0)
Requirement already satisfied: wcwidth in /opt/conda/lib/python3.11/site-packages (f
rom prettytable->ipython-sql) (0.2.13)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /opt/conda/lib/python3.11/site
-packages (from jedi>=0.16->ipython->ipython-sql) (0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in /opt/conda/lib/python3.11/site-pac
kages (from pexpect>4.3->ipython->ipython-sql) (0.7.0)
Requirement already satisfied: executing>=1.2.0 in /opt/conda/lib/python3.11/site-pa
ckages (from stack-data->ipython->ipython-sql) (2.0.1)
Requirement already satisfied: asttokens>=2.1.0 in /opt/conda/lib/python3.11/site-pa
ckages (from stack-data->ipython->ipython-sql) (2.4.1)
Requirement already satisfied: pure-eval in /opt/conda/lib/python3.11/site-packages
(from stack-data->ipython->ipython-sql) (0.2.2)
Using cached SQLAlchemy-2.0.36-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_6
4.whl (3.2 MB)
Installing collected packages: sqlalchemy
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 1.3.9
    Uninstalling SQLAlchemy-1.3.9:
      Successfully uninstalled SQLAlchemy-1.3.9
Successfully installed sqlalchemy-2.0.36
Requirement already satisfied: ipython-sql in /opt/conda/lib/python3.11/site-package
```

```
s (0.5.0)
Requirement already satisfied: prettytable in /opt/conda/lib/python3.11/site-package
s (3.12.0)
Requirement already satisfied: ipython in /opt/conda/lib/python3.11/site-packages (f
rom ipython-sql) (8.22.2)
Requirement already satisfied: sqlalchemy>=2.0 in /opt/conda/lib/python3.11/site-pac
kages (from ipython-sql) (2.0.36)
Requirement already satisfied: sqlparse in /opt/conda/lib/python3.11/site-packages
(from ipython-sql) (0.5.3)
Requirement already satisfied: six in /opt/conda/lib/python3.11/site-packages (from
ipython-sql) (1.16.0)
Requirement already satisfied: ipython-genutils in /opt/conda/lib/python3.11/site-pa
ckages (from ipython-sql) (0.2.0)
Requirement already satisfied: wcwidth in /opt/conda/lib/python3.11/site-packages (f
rom prettytable) (0.2.13)
Requirement already satisfied: typing-extensions>=4.6.0 in /opt/conda/lib/python3.1
1/site-packages (from sqlalchemy>=2.0->ipython-sql) (4.12.2)
Requirement already satisfied: greenlet!=0.4.17 in /opt/conda/lib/python3.11/site-pa
ckages (from sqlalchemy>=2.0->ipython-sql) (3.0.3)
Requirement already satisfied: decorator in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (0.19.1)
Requirement already satisfied: matplotlib-inline in /opt/conda/lib/python3.11/site-p
ackages (from ipython->ipython-sql) (0.1.7)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.41 in /opt/conda/lib/pytho
n3.11/site-packages (from ipython->ipython-sql) (3.0.42)
Requirement already satisfied: pygments>=2.4.0 in /opt/conda/lib/python3.11/site-pac
kages (from ipython->ipython-sql) (2.18.0)
Requirement already satisfied: stack-data in /opt/conda/lib/python3.11/site-packages
(from ipython->ipython-sql) (0.6.2)
Requirement already satisfied: traitlets>=5.13.0 in /opt/conda/lib/python3.11/site-p
ackages (from ipython->ipython-sql) (5.14.3)
Requirement already satisfied: pexpect>4.3 in /opt/conda/lib/python3.11/site-package
s (from ipython->ipython-sql) (4.9.0)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /opt/conda/lib/python3.11/site
-packages (from jedi>=0.16->ipython->ipython-sql) (0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in /opt/conda/lib/python3.11/site-pac
kages (from pexpect>4.3->ipython->ipython-sql) (0.7.0)
Requirement already satisfied: executing>=1.2.0 in /opt/conda/lib/python3.11/site-pa
ckages (from stack-data->ipython->ipython-sql) (2.0.1)
Requirement already satisfied: asttokens>=2.1.0 in /opt/conda/lib/python3.11/site-pa
ckages (from stack-data->ipython->ipython-sql) (2.4.1)
Requirement already satisfied: pure-eval in /opt/conda/lib/python3.11/site-packages
(from stack-data->ipython->ipython-sql) (0.2.2)
```

In [41]:
```
%load_ext sql
```

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

In [42]:
```
import csv, sqlite3
import prettytable
prettytable.DEFAULT = 'DEFAULT'
```

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

In [43]:
```
!pip install -q pandas
```

In [44]:
```
%sql sqlite:///my_data1.db
```

In [45]:
```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clou
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```

Out[45]: 101

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

In [46]:
```
#DROP THE TABLE IF EXISTS

%sql DROP TABLE IF EXISTS SPACEXTABLE;
```

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

Out[46]: []

In [47]:
```
%sql create table SPACEXTABLE as select * from SPACEXTBL where Date is not null
```

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

Out[47]: []

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"

Task 1

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

In [48]:
```
%%sql
    select distinct "Launch_Site"
    from SPACEXTABLE
```

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

Out[48]:

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Task 2

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

In [49]:
```
%%sql
    select Launch_Site
    from SPACEXTABLE
    where Launch_Site like 'CCA%'
    limit 5
```

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

Out[49]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Task 3

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

In [55]:
```
%%sql
SELECT SUM("PAYLOAD_MASS__KG_")
FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)'
```

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

Out[55]:

SUM("PAYLOAD_MASS__KG_")
45596

Task 4

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

In [57]:
```
%%sql
SELECT AVG("PAYLOAD_MASS__KG_")
FROM SPACEXTABLE
WHERE "Booster_Version" like 'F9 v1.1%'
```

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

Out[57]:

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [60]:
```
%%sql
SELECT Date
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)'
```

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

Out[60]:

Date
2015-12-22
2016-07-18
2017-02-19
2017-05-01
2017-06-03
2017-08-14
2017-09-07
2017-12-15
2018-01-08

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

In [66]:
```
%%sql
SELECT "Booster_Version"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)'
    and "PAYLOAD_MASS_KG_" > 4000
    and "PAYLOAD_MASS_KG_" < 6000
```

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

Out[66]:

Booster_Version

Task 7

List the total number of successful and failure mission outcomes

In [67]:
```
%%sql
SELECT Landing_Outcome, COUNT(*) AS Total
FROM SPACEXTABLE
GROUP BY Landing_Outcome;
```

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

Out[67]:

Landing_Outcome	Total
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

Task 8

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

In [74]:
```
%%sql
select "Booster_Version"
from SPACEXTABLE
where "PAYLOAD_MASS__KG_" == (select MAX("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

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

Out[74]:

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

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.

In [75]:

```
%%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'
    END AS Month_Name,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Failure (drone ship)'
  AND SUBSTR(Date, 1, 4) = '2015';
```

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

Out[75]:

Month_Name	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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.

In [77]:

```
%%sql
SELECT Landing_Outcome, COUNT(*) AS Outcome_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
ORDER BY Outcome_Count DESC;
```

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

Out[77]:

Landing_Outcome	Outcome_Count	Outcome_Rank
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

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

Author(s)

Lakshmi Holla

Other Contributors

Rav Ahuja