# 1 Assignment: SQL Notebook for Peer Assignment

Estimated time needed: **60** minutes.

## 1.1 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

## 1.2 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 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.

### 1.2.1 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

In [2]:
```
!pip install sqlalchemy==1.3.9
```
```
Collecting sqlalchemy==1.3.9
  Using cached SQLAlchemy-1.3.9-cp39-cp39-macosx_10_9_x86_64.whl
Installing collected packages: sqlalchemy
Attempting uninstall: sqlalchemy
  Found existing installation: SQLAlchemy 2.0.29
  Uninstalling SQLAlchemy-2.0.29:
    Successfully uninstalled SQLAlchemy-2.0.29
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behav
iour is the source of the following dependency conflicts.
ipython-sql 0.5.0 requires sqlalchemy>=2.0, but you have sqlalchemy 1.3.9 which is incompatible.
Successfully installed sqlalchemy-1.3.9
```

### 1.2.2 Connect to the database¶

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

In [3]:
```
#Please uncomment and execute the code below if you are working locally.

!pip install ipython-sql
```
```
Requirement already satisfied: ipython-sql in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (0.5.0)
Requirement already satisfied: ipython in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-sq
l) (7.29.0)
Requirement already satisfied: six in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-sql)
(1.16.0)
Requirement already satisfied: ipython-genutils in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from
ipython-sql) (0.2.0)
Requirement already satisfied: prettytable in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipyth
on-sql) (3.10.0)
Collecting sqlalchemy==2.0
  Using cached SQLAlchemy-2.0.29-cp39-cp39-macosx_10_9_x86_64.whl (2.1 MB)
Requirement already satisfied: sqlparse in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (0.4.4)
Requirement already satisfied: greenlet!=0.4.17 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from s
qlalchemy==2.0->ipython-sql) (1.1.1)
Requirement already satisfied: typing-extensions>=4.6.0 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages
 (from sqlalchemy==2.0->ipython-sql) (4.11.0)
Requirement already satisfied: backcall in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (0.2.0)
Requirement already satisfied: matplotlib-inline in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from
ipython-sql) (0.1.2)
Requirement already satisfied: pickleshare in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipyth
on->ipython-sql) (0.7.5)
Requirement already satisfied: decorator in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (5.1.0)
Requirement already satisfied: jedi>=0.16 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython
->ipython-sql) (0.18.0)
Requirement already satisfied: pexpect>4.3 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipyth
on->ipython-sql) (4.8.0)
Requirement already satisfied: setuptools>18.5 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ip
ython->ipython-sql) (58.0.4)
Requirement already satisfied: pygments in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (2.10.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /users/kexuanren/opt/anaconda3/lib/py
thon3.9/site-packages (from ipython->ipython-sql) (3.0.20)
Requirement already satisfied: traitlets>=4.2 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ip
ython->ipython-sql) (5.1.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages
 (from sqlalchemy==2.0->ipython-sql) (4.11.0)
Requirement already satisfied: greenlet!=0.4.17 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from s
qlalchemy==2.0->ipython-sql) (1.1.1)
Requirement already satisfied: backcall in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (0.2.0)
Requirement already satisfied: matplotlib-inline in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from
ipython-sql) (0.1.2)
Requirement already satisfied: pickleshare in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipyth
on->ipython-sql) (0.7.5)
Requirement already satisfied: decorator in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (5.1.0)
Requirement already satisfied: jedi>=0.16 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython
->ipython-sql) (0.18.0)
Requirement already satisfied: pexpect>4.3 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipyth
on->ipython-sql) (4.8.0)
Requirement already satisfied: setuptools>18.5 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ip
ython->ipython-sql) (58.0.4)
Requirement already satisfied: pygments in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ipython-
sql) (2.10.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /users/kexuanren/opt/anaconda3/lib/py
thon3.9/site-packages (from ipython->ipython-sql) (3.0.20)
Requirement already satisfied: traitlets>=4.2 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from ip
ython->ipython-sql) (5.1.0)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (fro
m jedi>=0.16->ipython->ipython-sql) (0.8.2)
Requirement already satisfied: ptyprocess>=0.5 in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from pe
xpect>4.3->ipython->ipython-sql) (0.7.0)
Requirement already satisfied: wcwidth in /users/kexuanren/opt/anaconda3/lib/python3.9/site-packages (from prompt-too
lkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython->ipython-sql) (0.2.5)
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-2.0.29
```

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

In [5]:
```
import csv, sqlite3

con = sqlite3.connect("my_data1.db")
#This line establishes a connection to an SQLite database named "my_data1.db". If the database file does not exist,
#it will be created. The connect() function returns a connection object (con).
cur = con.cursor()
# A cursor is used to execute SQL commands and navigate the database. i.e. cur.execute("SELECT * FROM my_table")
```

In [6]:
```
!pip install -q pandas==1.1.5
```
```
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behav
iour is the source of the following dependency conflicts.
sensors 0.13.0 requires pandas>=1.2, but you have pandas 1.1.5 which is incompatible.
pybbn 3.1.9 requires pandas<=1.4.4, but you have pandas 1.1.5 which is incompatible.
```

In [10]:
```
%load sqlite:///my_data1.db
#This line tells the %sql magic command to connect to the SQLite database file named my_data1.db.
#After running this command, you can execute SQL statements directly in subsequent cells using %sql followed by your
 SQL query.
```

In [7]:
```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/lab
s/module_2/data/spacex.csv")
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```
```
/Users/kexuanren/opt/anaconda3/lib/python3.9/site-packages/pandas/core/generic.py:2605: UserWarning: The spaces in th
ese column names will not be changed. In pandas versions < 0.14, spaces were converted to underscores.
  sql.to_sql(
```

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

In [9]:
```
%sql create table SPACEXTABLE as select * from SPACEXTBL where Date is not null
```
```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) table SPACEXTABLE already exists
[SQL: create table SPACEXTABLE as select * from SPACEXTBL where Date is not null]
(Background on this error at: https://sqlalche.me/e/20/e3q8)
```

## 1.3 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"**

### 1.3.1 Task 1¶

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

In [13]:
```
%sql SELECT * FROM SPACEXTABLE LIMIT 10
```
```
 * sqlite:///my_data1.db
Done.
```

Out[13]:

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 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 |
| 2013-09-29 | 16:00:00 | F9 v1.1 B1003 | VAFB SLC-4E | CASSIOPE | 500 | Polar LEO | MDA | Success | Uncontrolled (ocean) |
| 2013-12-03 | 22:41:00 | F9 v1.1 | CCAFS LC-40 | SES-8 | 3170 | GTO | SES | Success | No attempt |
| 2014-01-06 | 22:06:00 | F9 v1.1 | CCAFS LC-40 | Thaicom 6 | 3325 | GTO | Thaicom | Success | No attempt |
| 2014-04-18 | 19:25:00 | F9 v1.1 | CCAFS LC-40 | SpaceX CRS-3 | 2296 | LEO (ISS) | NASA (CRS) | Success | Controlled (ocean) |
| 2014-07-14 | 15:15:00 | F9 v1.1 | CCAFS LC-40 | OG2 Mission 1 6 Orbcomm-OG2 satellites | 1316 | LEO | Orbcomm | Success | Controlled (ocean) |

In [15]:
```
%sql select DISTINCT(Launch_Site) FROM SPACEXTABLE
```
```
 * sqlite:///my_data1.db
Done.
```

Out[15]:

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

### 1.3.2 Task 2¶

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

In [16]:
```
%sql SELECT * FROM spacextable WHERE Launch_Site LIKE 'CCA%' LIMIT 5
```
```
 * sqlite:///my_data1.db
Done.
```

Out[16]:

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 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 |

### 1.3.3 Task 3¶

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

In [18]:
```
%sql SELECT customer, SUM(payload_mass__kg_) AS total_payload_mass_kgsa FROM spacextable WHERE customer = 'NASA (CR
S)'
```
```
 * sqlite:///my_data1.db
Done.
```

Out[18]:

| Customer | total_payload_mass_kgsa |
|---|---|
| NASA (CRS) | 45596 |

### 1.3.4 Task 4¶

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

In [19]:
```
%sql SELECT booster_version, AVG(payload_mass__kg_) AS avg FROM spacextable WHERE booster_version = 'F9 v1.1'
```
```
 * sqlite:///my_data1.db
Done.
```

Out[19]:

| Booster_Version | avg |
|---|---|
| F9 v1.1 | 3928.4 |

### 1.3.5 Task 5¶

*List the date when the first successful landing outcome in ground pad was achieved.*

*Hint:Use min function*

In [22]:
```
%sql SELECT DISTINCT(landing_outcome) FROM spacextable
```
```
 * sqlite:///my_data1.db
Done.
```

Out[22]:

| Landing_Outcome |
|---|
| Failure (parachute) |
| No attempt |
| Uncontrolled (ocean) |
| Controlled (ocean) |
| Failure (drone ship) |
| Precluded (drone ship) |
| Success (ground pad) |
| Success (drone ship) |
| Success |
| Failure |
| No attempt |

In [23]:
```
%sql SELECT MIN(date) FROM spacextable WHERE landing_Outcome = 'Success (ground pad)'
```
```
 * sqlite:///my_data1.db
Done.
```

Out[23]:

| MIN(date) |
|---|
| 2015-12-22 |

### 1.3.6 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*

In [25]:
```
%sql SELECT booster_version, landing_outcome, payload_mass__kg_ FROM spacextable WHERE (landing_Outcome = 'Success
(drone ship)') AND (payload_mass__kg_ Between 4000 and 6000)
```
```
 * sqlite:///my_data1.db
Done.
```

Out[25]:

| Booster_Version | Landing_Outcome | PAYLOAD_MASS__KG_ |
|---|---|---|
| F9 FT B1022 | Success (drone ship) | 4696 |
| F9 FT B1026 | Success (drone ship) | 4600 |
| F9 FT B1021.2 | Success (drone ship) | 5300 |
| F9 FT B1031.2 | Success (drone ship) | 5200 |

### 1.3.7 Task 7¶

*List the total number of successful and failure mission outcomes*

In [27]:
```
%sql SELECT mission_outcome, COUNT(*) AS NUM_mission_outcomes FROM spacextable GROUP BY mission_outcome
```
```
 * sqlite:///my_data1.db
Done.
```

Out[27]:

| Mission_Outcome | NUM_mission_outcomes |
|---|---|
| Failure (in flight) | 1 |
| Success | 98 |
| Success | 1 |
| Success (payload status unclear) | 1 |

### 1.3.8 Task 8¶

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

In [29]:
```
%sql SELECT booster_version, payload_mass_kg_ FROM spacextable WHERE payload_mass__kg_ == (SELECT MAX(payload_mass_
_kg_) FROM spacextable)
```
```
 * sqlite:///my_data1.db
Done.
```

Out[29]:

| Booster_Version | PAYLOAD_MASS__KG_ |
|---|---|
| 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 |

### 1.3.9 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: SQLite 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 [32]:
```
%sql SELECT landing_outcome, substr(Date, 6,2) as month, booster_version, launch_site FROM spacextable WHERE landing
_outcome = 'Failure (drone ship)' AND substr(Date,0,5)='2015'
```
```
 * sqlite:///my_data1.db
Done.
```

Out[32]:

| Landing_Outcome | month | Booster_Version | Launch_Site |
|---|---|---|---|
| Failure (drone ship) | 01 | F9 v1.1 B1012 | CCAFS LC-40 |
| Failure (drone ship) | 04 | F9 v1.1 B1015 | CCAFS LC-40 |

### 1.3.10 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.*

In [33]:
```
%sql SELECT landing_outcome, COUNT(*) AS count FROM spacextable WHERE Date BETWEEN '20100604' AND '20170320' GROUP BY la
nding_outcome ORDER BY count DESC;
```
```
 * sqlite:///my_data1.db
Done.
```

Out[33]:

| Landing_Outcome | count |
|---|---|
| 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 |

### 1.3.11 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

## 1.4 Author(s)¶

**Lakshmi Holla**

## 1.5 Other Contributors¶

**Rav Ahuja**

## 1.6 Change log¶

| Date | Version | Changed by | Change Description |
|---|---|---|---|
| 2021-07-09 | 0.2 | Lakshmi Holla | Changes made in magic sql |
| 2021-05-20 | 0.1 | Lakshmi Holla | Created Initial Version |

## 1.7



### 1.7.2