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

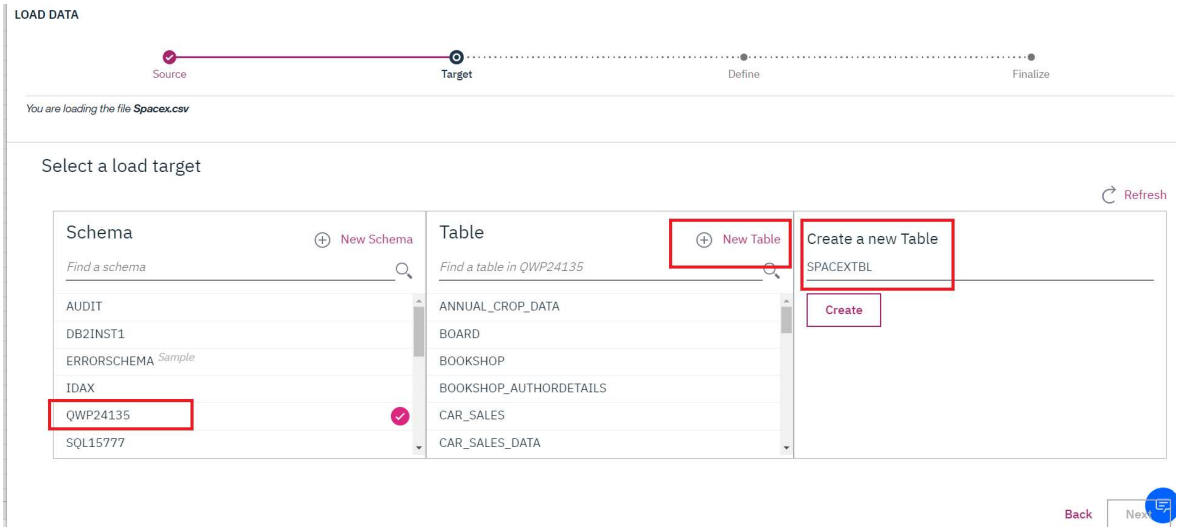
This assignment requires you to load the spacex dataset.

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

Store the dataset in database table

it is highly recommended to manually load the table using the database console LOAD tool in DB2.



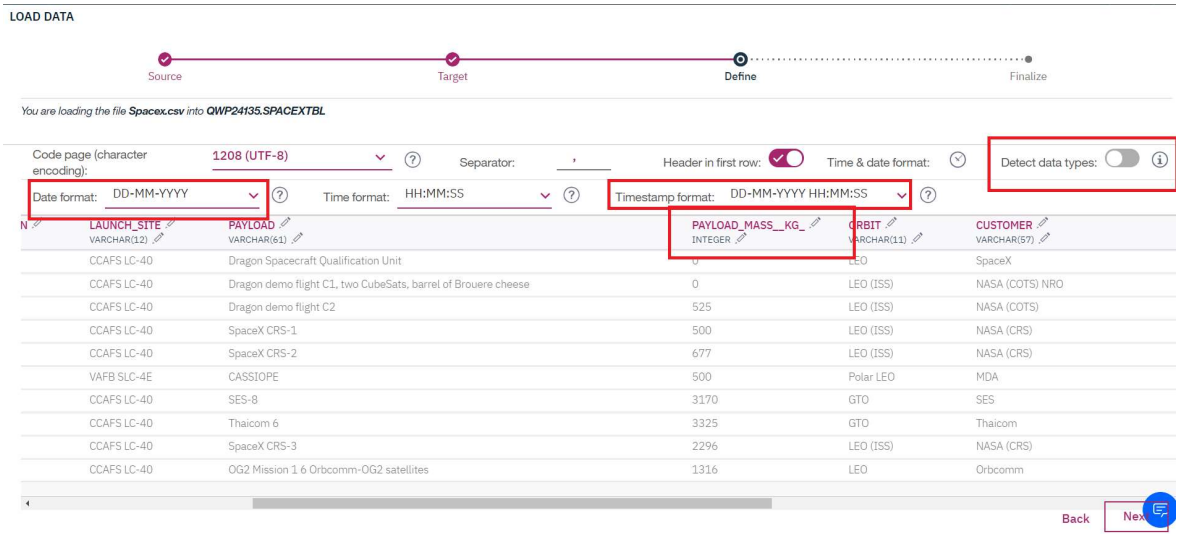
Now open the Db2 console, open the LOAD tool, Select / Drag the .CSV file for the dataset, Next create a New Table, and then follow the steps on-screen instructions to load the data. Name the new table as follows:

SPACEXDATASET

Follow these steps while using old DB2 UI which is having Open Console Screen

Note:While loading Spacex dataset, ensure that detect datatypes is disabled. Later click on the pencil icon(edit option).

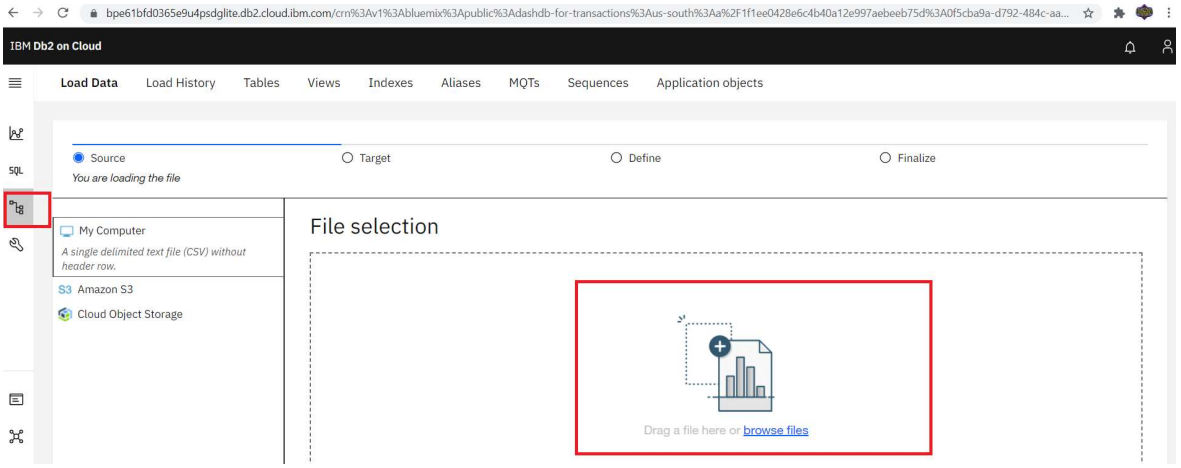
1. Change the Date Format by manually typing DD-MM-YYYY and timestamp format as DD-MM-YYYY HH:MM:SS
2. Change the PAYLOAD_MASS__KG_ datatype to INTEGER.



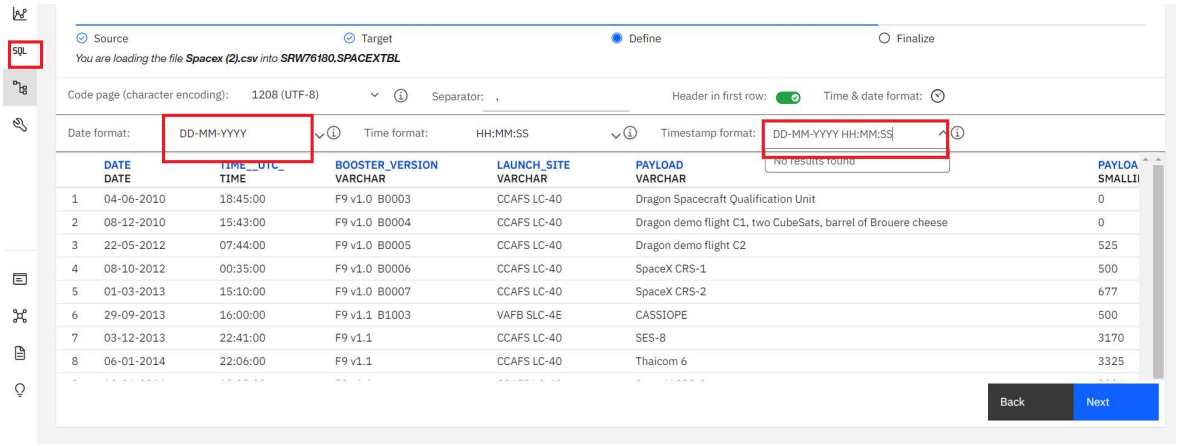
Changes to be considered when having DB2 instance with the new UI having Go to UI screen

- Refer to this insruction in this link for viewing the new Go to UI screen.

- Later click on **Data link(below SQL)** in the Go to UI screen and click on **Load Data** tab.
- Later browse for the downloaded spacex file.



- Once done select the schema andload the file.



In [1]:
```
!pip install sqlalchemy==1.3.9
!pip install ipython-sql
```

```
Collecting sqlalchemy==1.3.9
  Downloading SQLAlchemy-1.3.9.tar.gz (6.0 MB)
     |████████████████████████████████| 6.0 MB 10.9 MB/s eta 0:00:01
Building wheels for collected packages: sqlalchemy
  Building wheel for sqlalchemy (setup.py) ... done
  Created wheel for sqlalchemy: filename=SQLAlchemy-1.3.9-cp38-cp38-linux_x86_64.whl
size=1209506 sha256=a4f66e96cdfed89eedd2a2d548ae0a41c6d6a5e035ecf5afb087f3fc548e905e
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/cb/43/46/fa638f2422554332b7865d
600275b24568bf60e76104a94bb4
Successfully built sqlalchemy
Installing collected packages: sqlalchemy
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 1.4.22
    Uninstalling SQLAlchemy-1.4.22:
      Successfully uninstalled SQLAlchemy-1.4.22
Successfully installed sqlalchemy-1.3.9
Collecting ipython-sql
  Downloading ipython_sql-0.4.0-py3-none-any.whl (19 kB)
Requirement already satisfied: six in /opt/conda/envs/Python-3.8-main/lib/python3.8/
site-packages (from ipython-sql) (1.15.0)
```

```
Collecting sqlparse
  Downloading sqlparse-0.4.2-py3-none-any.whl (42 kB)
     |████████████████████████████████| 42 kB 2.3 MB/s  eta 0:00:01
Requirement already satisfied: ipython>=1.0 in /opt/conda/envs/Python-3.8-main/lib/p
ython3.8/site-packages (from ipython-sql) (7.27.0)
Requirement already satisfied: ipython-genutils>=0.1.0 in /opt/conda/envs/Python-3.8
-main/lib/python3.8/site-packages (from ipython-sql) (0.2.0)
Collecting prettytable<1
  Downloading prettytable-0.7.2.zip (28 kB)
Requirement already satisfied: sqlalchemy>=0.6.7 in /opt/conda/envs/Python-3.8-main/
lib/python3.8/site-packages (from ipython-sql) (1.3.9)
Requirement already satisfied: matplotlib-inline in /opt/conda/envs/Python-3.8-main/
lib/python3.8/site-packages (from ipython>=1.0->ipython-sql) (0.1.2)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /opt/
conda/envs/Python-3.8-main/lib/python3.8/site-packages (from ipython>=1.0->ipython-s
ql) (3.0.20)
Requirement already satisfied: traitlets>=4.2 in /opt/conda/envs/Python-3.8-main/li
b/python3.8/site-packages (from ipython>=1.0->ipython-sql) (5.0.5)
Requirement already satisfied: pygments in /opt/conda/envs/Python-3.8-main/lib/pytho
n3.8/site-packages (from ipython>=1.0->ipython-sql) (2.9.0)
Requirement already satisfied: pexpect>4.3 in /opt/conda/envs/Python-3.8-main/lib/py
thon3.8/site-packages (from ipython>=1.0->ipython-sql) (4.8.0)
Requirement already satisfied: decorator in /opt/conda/envs/Python-3.8-main/lib/pyth
on3.8/site-packages (from ipython>=1.0->ipython-sql) (5.0.9)
Requirement already satisfied: setuptools>=18.5 in /opt/conda/envs/Python-3.8-main/l
ib/python3.8/site-packages (from ipython>=1.0->ipython-sql) (52.0.0.post20211006)
Requirement already satisfied: backcall in /opt/conda/envs/Python-3.8-main/lib/pytho
n3.8/site-packages (from ipython>=1.0->ipython-sql) (0.2.0)
Requirement already satisfied: jedi>=0.16 in /opt/conda/envs/Python-3.8-main/lib/pyt
hon3.8/site-packages (from ipython>=1.0->ipython-sql) (0.17.2)
Requirement already satisfied: pickleshare in /opt/conda/envs/Python-3.8-main/lib/py
thon3.8/site-packages (from ipython>=1.0->ipython-sql) (0.7.5)
Requirement already satisfied: parso<0.8.0,>=0.7.0 in /opt/conda/envs/Python-3.8-mai
n/lib/python3.8/site-packages (from jedi>=0.16->ipython>=1.0->ipython-sql) (0.7.0)
Requirement already satisfied: ptyprocess>=0.5 in /opt/conda/envs/Python-3.8-main/li
b/python3.8/site-packages (from pexpect>4.3->ipython>=1.0->ipython-sql) (0.7.0)
Requirement already satisfied: wcwidth in /opt/conda/envs/Python-3.8-main/lib/python
3.8/site-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=1.0->
ipython-sql) (0.2.5)
Building wheels for collected packages: prettytable
  Building wheel for prettytable (setup.py) ... done
  Created wheel for prettytable: filename=prettytable-0.7.2-py3-none-any.whl size=13
700 sha256=2fe2902661c7c57d1d1c51ed39c609e22267efe5f1be0e785623028c5732097b
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/48/6d/77/9517cb933af254f51a446f
1a5ec9c2be3e45f17384940bce68
Successfully built prettytable
Installing collected packages: sqlparse, prettytable, ipython-sql
Successfully installed ipython-sql-0.4.0 prettytable-0.7.2 sqlparse-0.4.2
```

In [2]:

```
import sqlalchemy
```

Connect to the database

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

In [3]:
```
%load_ext sql
```

In [4]:
```
import csv, sqlite3

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

In [5]:
```
!pip install -q pandas==1.1.5
```

```
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.
sparkmagic 0.18.0 requires nose, which is not installed.
hdijupyterutils 0.18.0 requires jupyter>=1, which is not installed.
hdijupyterutils 0.18.0 requires nose, which is not installed.
```

In [6]:
```
%sql sqlite:///my_data1.db
```

In [7]:
```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```

```
/opt/conda/envs/Python-3.8-main/lib/python3.8/site-packages/pandas/core/generic.py:2
779: UserWarning: The spaces in these column names will not be changed. In pandas ve
rsions < 0.14, spaces were converted to underscores.
  name       (index) object 'falcon' 'parrot' 'lion' 'monkey'
```

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 [8]:
```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

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

Out[8]:

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 [9]:
```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

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

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	M
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	
08-12-2010	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	
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	

Task 3

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

In [10]:
```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

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

Out[10]:

SUM("PAYLOAD_MASS__KG_")
45596

Task 4

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

In [11]:
```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F
```

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

Out[11]:

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 [12]:
```
%sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" LIKE '%Success%'
```

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

Out[12]:

MIN("DATE")
01-05-2017

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 [13]:
```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (dr
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

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

Out[13]:

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Task 7

List the total number of successful and failure mission outcomes

In [14]:
```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME"
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failu
```

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

Out[14]:

SUCCESS	FAILURE

100	1

Task 8

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

In [15]:
```
%sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL \
WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

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

Out[15]:

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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [16]:
```
%sql SELECT substr("DATE", 4, 2) AS MONTH, "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPA
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

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

Out[16]:

MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

In [17]:
```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

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

Out[17]:

Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

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

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