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

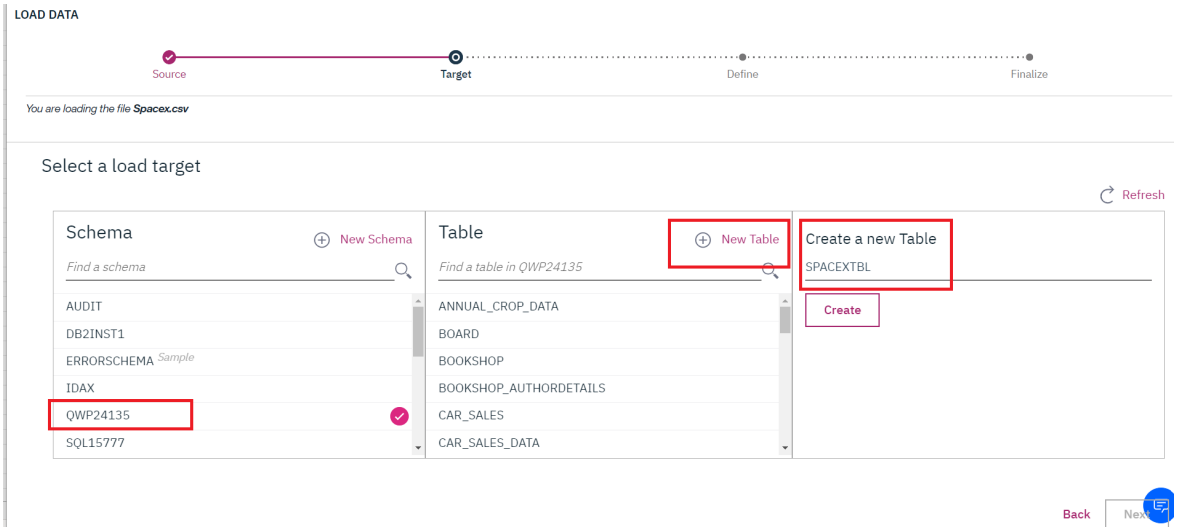
This assignment requires you to load the spacex dataset.

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

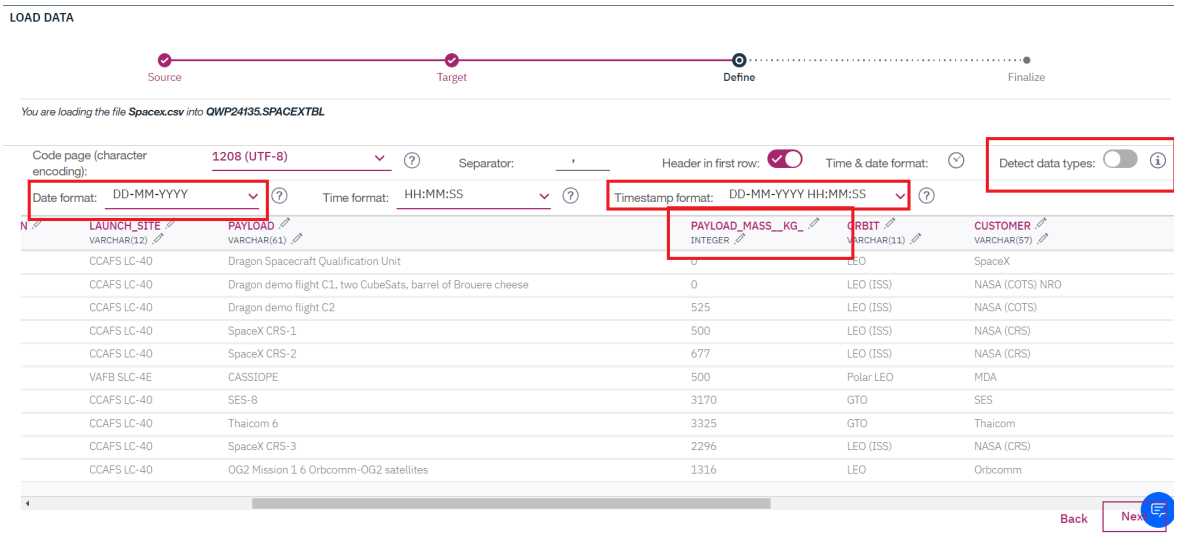
Follow these steps while using old DB2 UI which is having Open Console Screen

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.

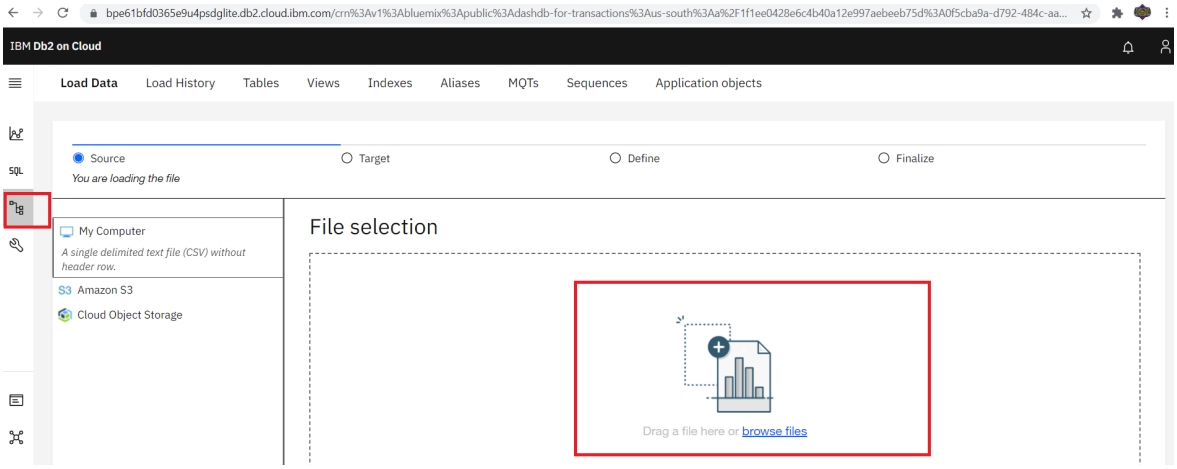
   Here you should place the cursor at Date field and manually type as DD-MM-YYYY.

2. Change the PAYLOAD*MASS*_KG_ datatype to INTEGER.

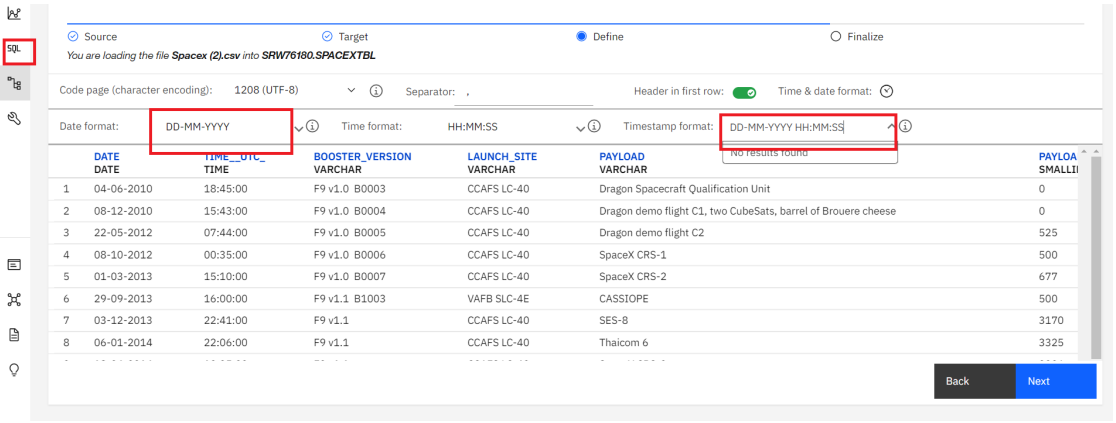


Changes to be considered when having DB2 instance with the new UI having Go to UI screen

- 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 ibm_db_sa
!pip install ipython-sql
```

```
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/dhcrypt
o.py:16: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_
bytes instead
  from cryptography.utils import int_from_bytes
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/util.py:
25: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_bytes
instead
  from cryptography.utils import int_from_bytes
Collecting sqlalchemy==1.3.9
  Downloading SQLAlchemy-1.3.9.tar.gz (6.0 MB)
     |████████████████████████████████| 6.0 MB 14.3 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-cp37-cp37m-linux_x86_64.wh
l size=1207789 sha256=55ab3f3eb7a89133b4c3e948b7cf7c7c566c8d26392f6d9f0e33e66b8a78e5
e2
```

```
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/03/71/13/010faf12246f72dc76b415
0e6e599d13a85b4435e06fb9e51f
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
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/dhcrypt
o.py:16: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_
bytes instead
  from cryptography.utils import int_from_bytes
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/util.py:
25: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_bytes
instead
  from cryptography.utils import int_from_bytes
Requirement already satisfied: ibm_db_sa in /opt/conda/envs/Python-3.7-OpenCE/lib/py
thon3.7/site-packages (0.3.7)
Requirement already satisfied: ibm-db>=2.0.0 in /opt/conda/envs/Python-3.7-OpenCE/li
b/python3.7/site-packages (from ibm_db_sa) (3.0.4)
Requirement already satisfied: sqlalchemy>=0.7.3 in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ibm_db_sa) (1.3.9)
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/dhcrypt
o.py:16: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_
bytes instead
  from cryptography.utils import int_from_bytes
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorage/util.py:
25: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_bytes
instead
  from cryptography.utils import int_from_bytes
Collecting ipython-sql
  Downloading ipython_sql-0.4.0-py3-none-any.whl (19 kB)
Requirement already satisfied: six in /opt/conda/envs/Python-3.7-OpenCE/lib/python3.
7/site-packages (from ipython-sql) (1.15.0)
Collecting sqlparse
  Downloading sqlparse-0.4.2-py3-none-any.whl (42 kB)
     |████████████████████████████████| 42 kB 2.9 MB/s  eta 0:00:01
Requirement already satisfied: ipython-genutils>=0.1.0 in /opt/conda/envs/Python-3.7
-OpenCE/lib/python3.7/site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: sqlalchemy>=0.6.7 in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ipython-sql) (1.3.9)
Collecting prettytable<1
  Downloading prettytable-0.7.2.tar.bz2 (21 kB)
Requirement already satisfied: ipython>=1.0 in /opt/conda/envs/Python-3.7-OpenCE/li
b/python3.7/site-packages (from ipython-sql) (7.15.0)
Requirement already satisfied: traitlets>=4.2 in /opt/conda/envs/Python-3.7-OpenCE/l
ib/python3.7/site-packages (from ipython>=1.0->ipython-sql) (4.3.3)
Requirement already satisfied: pickleshare in /opt/conda/envs/Python-3.7-OpenCE/lib/
python3.7/site-packages (from ipython>=1.0->ipython-sql) (0.7.5)
Requirement already satisfied: jedi>=0.10 in /opt/conda/envs/Python-3.7-OpenCE/lib/p
ython3.7/site-packages (from ipython>=1.0->ipython-sql) (0.17.1)
Requirement already satisfied: backcall in /opt/conda/envs/Python-3.7-OpenCE/lib/pyt
hon3.7/site-packages (from ipython>=1.0->ipython-sql) (0.2.0)
Requirement already satisfied: pygments in /opt/conda/envs/Python-3.7-OpenCE/lib/pyt
hon3.7/site-packages (from ipython>=1.0->ipython-sql) (2.6.1)
Requirement already satisfied: setuptools>=18.5 in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ipython>=1.0->ipython-sql) (47.3.1.post20200622)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /opt/
conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages (from ipython>=1.0->ipython
```

```
-sql) (3.0.5)
Requirement already satisfied: pexpect; sys_platform != "win32" in /opt/conda/envs/P
ython-3.7-OpenCE/lib/python3.7/site-packages (from ipython>=1.0->ipython-sql) (4.8.
0)
Requirement already satisfied: decorator in /opt/conda/envs/Python-3.7-OpenCE/lib/py
thon3.7/site-packages (from ipython>=1.0->ipython-sql) (4.4.2)
Requirement already satisfied: parso<0.8.0,>=0.7.0 in /opt/conda/envs/Python-3.7-Ope
nCE/lib/python3.7/site-packages (from jedi>=0.10->ipython>=1.0->ipython-sql) (0.7.0)
Requirement already satisfied: wcwidth in /opt/conda/envs/Python-3.7-OpenCE/lib/pyth
on3.7/site-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=1.0
->ipython-sql) (0.2.4)
Requirement already satisfied: ptyprocess>=0.5 in /opt/conda/envs/Python-3.7-OpenCE/
lib/python3.7/site-packages (from pexpect; sys_platform != "win32"->ipython>=1.0->ip
ython-sql) (0.6.0)
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=61e5724e343ffbfd8afd47d17ad413c2745c3c72886a3bad59df641ca6583fbe
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/8c/76/0b/eb9eb3da7e2335e3577e3f
96a0ae9f74f206e26457bd1a2bc8
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
```

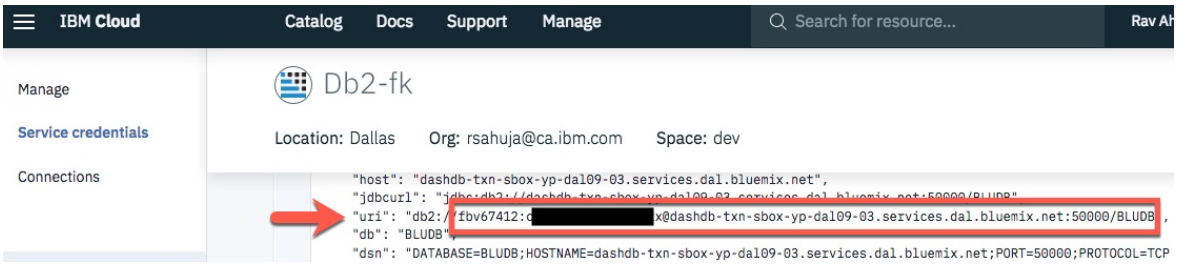
Connect to the database

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

In [2]:
```
%load_ext sql
```

DB2 magic in case of old UI service credentials.

In the next cell enter your db2 connection string. Recall you created Service Credentials for your Db2 instance before. From the **uri** field of your Db2 service credentials copy everything after db2:// (except the double quote at the end) and paste it in the cell below after ibm_db_sa://



in the following format

%sql ibm_db_sa://my-username:my-password\@my-hostname:my-port/my-db-name

DB2 magic in case of new UI service credentials.

- Use the following format.
- Add security=SSL at the end

%sql ibm_db_sa://my-username:my-password\@my-hostname:my-port/my-db-name?security=SSL

In [4]:
```
%sql ibm_db_sa://cky43798:B3o4ThmQsD6Cuzq9@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1og
```

Tasks

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

Task 1

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

In [17]:
```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde0
0.databases.appdomain.cloud:32733/bludb
Done.
```

Out[17]:

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

Task 2

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

In [18]:
```
%%sql
SELECT * FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[18]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	m
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	
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	
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	
2013-03-01	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 [19]:
```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) AS total_payload_mass_kg
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[19]:

total_payload_mass_kg
45596

Task 4

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

In [20]:
```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS avg_payload_mass_kg
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1'
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde0
0.databases.appdomain.cloud:32733/bludb
Done.
```

Out[20]:

avg_payload_mass_kg
2928

Task 5

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

Hint:Use min function

In [21]:
```
%%sql
SELECT MIN(DATE) AS first_successful_landing_date
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (ground pad)'
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde0
0.databases.appdomain.cloud:32733/bludb
Done.
```

Out[21]:

first_successful_landing_date
2015-12-22

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 [22]:
```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde0
0.databases.appdomain.cloud:32733/bludb
Done.
```

Out[22]:

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 [23]:
```
%%sql
SELECT MISSION_OUTCOME, COUNT(*) AS total_number
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[23]:

mission_outcome	total_number
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Task 8

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

In [7]:
```
%%sql
SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[7]:

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600

F9 B5 B1060.3 15600

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [9]:
```
%%sql
SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = '2015'
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[9]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
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 [26]:
```
%%sql
SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS total_number
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY total_number DESC
```

```
 * ibm_db_sa://cky43798:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[26]:

landing__outcome	total_number
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

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
2021-10-12	0.4	Lakshmi Holla	Changed markdown
2021-08-24	0.3	Lakshmi Holla	Added library update
2021-07-09	0.2	Lakshmi Holla	Changes made in magic sql
2021-05-20	0.1	Lakshmi Holla	Created Initial Version