

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 whereas 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.

LOAD DATA

Source Target Define Finalize

You are loading the file *Spacex.csv*

Select a load target

Schema ⊕ New Schema Find a schema

Table ⊕ New Table Find a table in QWP24135

Create a new Table
SPACEXTBL
Create

AUDIT
DB2INST1
ERRORSCHEMA *Sample*
IDAX
QWP24135 ✓
SQL15777

ANNUAL_CROP_DATA
BOARD
BOOKSHOP
BOOKSHOP_AUTHORDetails
CAR_SALES
CAR_SALES_DATA

Back Next

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 PAYLOADMASS_KG_ datatype to INTEGER.

LOAD DATA

Source Target Define Finalize

You are loading the file *Spacex.csv* into **QWP24135.SPACEXTBL**

Code page (character encoding): **1208 (UTF-8)** ? Separator: **,** Header in first row: ☒ Time & date format: ? Detect data types: ☐ ?

Date format: **DD-MM-YYYY** ? Time format: **HH:MM:SS** ? Timestamp format: **DD-MM-YYYY HH:MM:SS** ?

LAUNCH_SITE	PAYLOAD	PAYLOAD_MASS_KG_	ORBIT	CUSTOMER
VARCHAR(12)	VARCHAR(61)	INTEGER	VARCHAR(11)	VARCHAR(57)
CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO
CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)
VAFB SLC-4E	CASSIOPE	500	Polar LEO	MDA
CCAFS LC-40	SES-8	3170	GTO	SES
CCAFS LC-40	Thaicom 6	3325	GTO	Thaicom
CCAFS LC-40	SpaceX CRS-3	2296	LEO (ISS)	NASA (CRS)
CCAFS LC-40	OG2 Mission 1.6 Orbcomm-OG2 satellites	1316	LEO	Orbcomm

Back Next

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

- The screenshot shows the IBM Db2 on Cloud console interface. At the top, there's a navigation bar with the IBM Db2 on Cloud logo and a user profile icon. Below this is a main navigation menu with options like 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Load Data' section is active, showing a workflow with four steps: 'Source' (selected), 'Target', 'Define', and 'Finalize'. Below the workflow, there's a 'File selection' area. This area has a dashed border and contains a file icon with a plus sign, indicating where to drag a file. Below the icon, it says 'Drag a file here or [browse files](#)'. On the left side of the 'File selection' area, there's a sidebar with options: 'My Computer' (with a description 'A single delimited text file (CSV) without header row.'), 'Amazon S3', and 'Cloud Object Storage'. The 'File selection' area is highlighted with a red box.

- SQL

Source

Target

Define

Finalize

You are loading the file **Spacex (2).csv** into **SRW76180.SPACEXTBL**

Code page (character encoding): 1208 (UTF-8)

Separator: ,

Header in first row: ☒

Time & date format:

Date format: DD-MM-YYYY

Time format: HH:MM:SS

Timestamp format: DD-MM-YYYY HH:MM:SS

	DATE DATE	TIME TIME	BOOSTER_VERSION VARCHAR	LAUNCH_SITE VARCHAR	PAYLOAD VARCHAR	PAYLOAD SMALLINT
1	04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0
2	08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0
3	22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525
4	08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500
5	01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677
6	29-09-2013	16:00:00	F9 v1.1 B1003	VAFB SLC-4E	CASSIOPE	500
7	03-12-2013	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170
8	06-01-2014	22:06:00	F9 v1.1	CCAFS LC-40	Thaicom 6	3325

Back

Next

```
!pip install sqlalchemy==1.3.9
!pip install ibm_db_sa
!pip install ipython-sql
```

```
Created wheel for sqlalchemy: filename=SQLAlchemy-1.3.9-cp37-cp37m-linux_x86_64.whl size=1207789 sha256=55ab3f3eb7a89133b4c3e948b7cf7c7c566c8d26392f6d9f0e33e66b8a78e5e2
```

```

    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/dhcrypto.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/python3.7/site-packages (0.3.7)
Requirement already satisfied: ibm-db>=2.0.0 in /opt/conda/envs/Python-3.7-OpenCE/lib/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-OpenCE/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/dhcrypto.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-OpenCE/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/lib/python3.7/site-packages (from ipython-sql) (7.15.0)
Requirement already satisfied: traitlets>=4.2 in /opt/conda/envs/Python-3.7-OpenCE/lib/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/python3.7/site-packages (from ipython>=1.0->ipython-sql) (0.17.1)
Requirement already satisfied: backcall in /opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages (from ipython>=1.0->ipython-sql) (0.2.0)
Requirement already satisfied: pygments in /opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages (from ipython>=1.0->ipython-sql) (2.6.1)
Requirement already satisfied: setuptools>=18.5 in /opt/conda/envs/Python-3.7-OpenCE/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/Python-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/python3.7/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-OpenCE/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/python3.7/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->ipython-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=13700 sha256=61e5724e343ffbfd8afd47d17ad413c2745c3c72886a3bad59df641ca6583fbc
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/8c/76/0b/eb9eb3da7e2335e3577e3f96a0ae9f74f206e26457bd1a2bc8
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

```

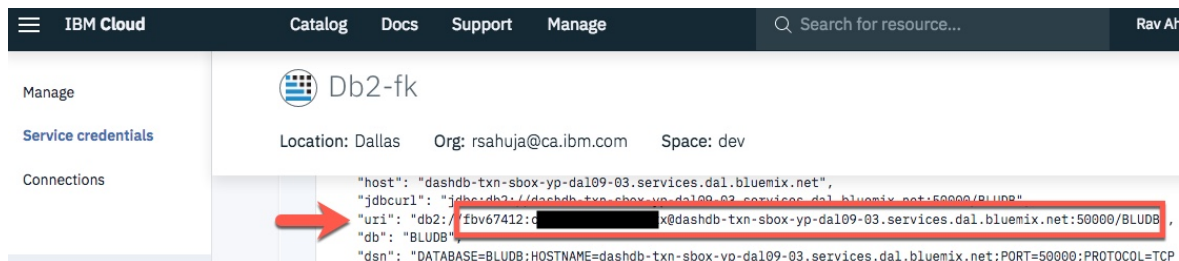
Connect to the database

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.

```

    "password": "qdg93144",
    "username": "qdg93144"
  },
  "certificate": {
    "certificate_base64": "LS0tLS1CRUdJTiBDRVJUSUJQ0FURS0tLS0tCk1JSURFakNDQWZxOz0F3SUJBZ0lKQVA1S0R3ZTNCTkxiTUEwR0NTcl
FFQkN3VUFNqjR4SERBYUJnTlYkQkFNTUuWbENUU0JEYkc5MVPDQkVZWfJowW1GelPjYTXdIaGN0TwpBd01qSTVNRRF5TVRBeVdoY05NekF3TWpJMgpNRRF5TVI
NUnd3R2dZRFZRUUREQk5KUWswZ1EYeHZkV1FnUkdGMF1XSmhjM1Z6TU1JQk1qQU5CZ2txCmhraUc5dzBCQVFRkFBT0NBUThtBTU1JQkNnS0NBuUVBdXUvbitj
NU8xSGpEa1psK25iYjE4UkR4ZGwKTzRUL3FoUGMxMTREY1FUK0p1RXdhG13aG1jTGxaQnF2QWFMb1hzbmhqSVFOMG01L0x5YzdBY291VXNmSGR0QwpDVGcr!
DMzTHM3d1dTakxqVE96N3M3M1ZUSU5yYmx3cnRIRUlvM1JWTKV6SkNHYW5LSXZdZWZVSUtrClNM1R0SD15cnFsSGN0Z2pIU1FmRkVTRmlYaHJiODhSQmd0ar
pCaTFBeEVadWobNobWZ2QVRmNEN0Y3EKY21QcHNqdDBPTnI0YnhJMVRyUWxEmNiN1hMSFBzWW91SUprdnVzMUZvaTEySmRNM1MkK3labFZPMUZmZkU3bwpKMjI
GOGtIU0NMNSKJvITTF5Z3FPZG90Vm5Q0C9E0WZhamNNN01wd2V4a01S0TKNR1FJREFRQUJvMU13ClVUQWRCZ05WSFE0RUZnUUVV1Q3JZanFJQzc1VUpxVmZEMDh:
UmN3SHdZRFZSMGpCQmd3Rm9BVWVdclKkanFJQzc1VUpxVmZEMDh1ZWdqeDZiUmN3RHdZRFZSMFRBUUgVQkFvd0F3RUIvekF0QmdrcWhraUc5dzBCQVFRzRgpBt
UkyRTBU0Ut3MlN3RjJ2MXBqaHV4M01kWWV2SGFVSkRmb0tPd0hSRnFS0HgxZ2dRcGVFcFBNMk5SCkx3R08yek85SWZUMmhLaWd1d2orWnJ5SGxxcHlxQ0pL0I
VPekIyWmE2S1YrQTVscEttMwdjV3VHYzMkK1UxVTFzTDd1Ujd3ZFVjU0TVU4aERvNi9sVHRMRVB2Mnc3VlNPS1FDK013ejgrTFJMdjVHSW5BN1JySWNhKw
4ZEtd1pLYThWcnBMXJ3QzRnY3d1YUhyMUNEWE42K0JIBzhvWGSYkWh6UG91c1dYS1BoaGdXZ2J5CkNDcUdIK0NWnQ1eFg3b05NS3VNSUqRVZndnNLWnR
NVZZbHQ0b1J3dTFlbGdzRDNjek1tbj1LREQKNHB1REFvYTZyMktZZE4xVksuN3F3VG1TbD1TU05RPT0KLS0tLS1FTkQgQ0VSVe1GSUNBEUtlS0tLQo=",
    "name": "1cbbb1b6-3a1a-4d49-9262-3102a8f7a7c8"
  },
  "composed": [
    "1cbbb1b6-3a1a-4d49-9262-3102a8f7a7c8-1cbbb1b6-3a1a-4d49-9262-3102a8f7a7c8-databases.appdomain.c
3/bludb?authSource=admin&replicaSet=replset"
  ],
  "database": "bludb",
  "host_ros": [
    "54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30592"
  ],
  "hosts": [
    {
      "hostname": "1cbbb1b6-3a1a-4d49-9262-3102a8f7a7c8",
      "port": 32733
    }
  ]
}

```

- 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.c1ogj3sd0tgtu0lqde00.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.c1ogj3sd0tgtu0lqde0
0.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.c1ogj3sd0tgtu0lqde0
0.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.c1ogj3sd0tgtu0lqde0
0.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.c1ogj3sd0tgtu0lqde0
0.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

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.c1ogj3sd0tgtu0lqde0
0.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.c1ogj3sd0tgtu0lqde0
0.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

© IBM Corporation 2021. All rights reserved.