Lab1 - Getting started on Watson Studio Hands-On

In this first Lab, we will start getting to grips with **IBM Watson Studio** projects, services, data assets, and run our first **Jupyter Notebook.**

Setup

Watson Studio is an IBM Cloud service, so in addition to the IBM Cloud account setup, you will need to create the Watson Studio instance. In addition, Watson Studio makes use of additional data and AI related services from the IBM Cloud platform, so we will create some artifacts for use within Watson Studio at runtime.

1. Create a Watson Studio service instance
2. Create a Watson Studio Project for the workshop.
3. Provision a set of additional services
4. Load data files into the project as Data Assets

Getting started with data exploration and notebooks

Once the Watson Studio project is completed, we can start our data related work

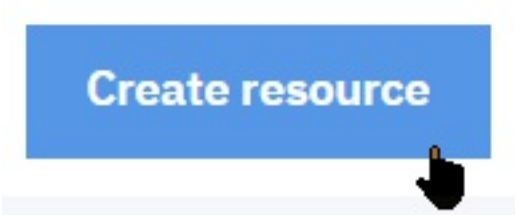
1. Quick assesment of the contents of a Data Asset
2. Work with Jupyter notebooks

The source material for the Workshop is held in a Box folder at URL
https://ibm.box.com/v/WatsonStudio-WS

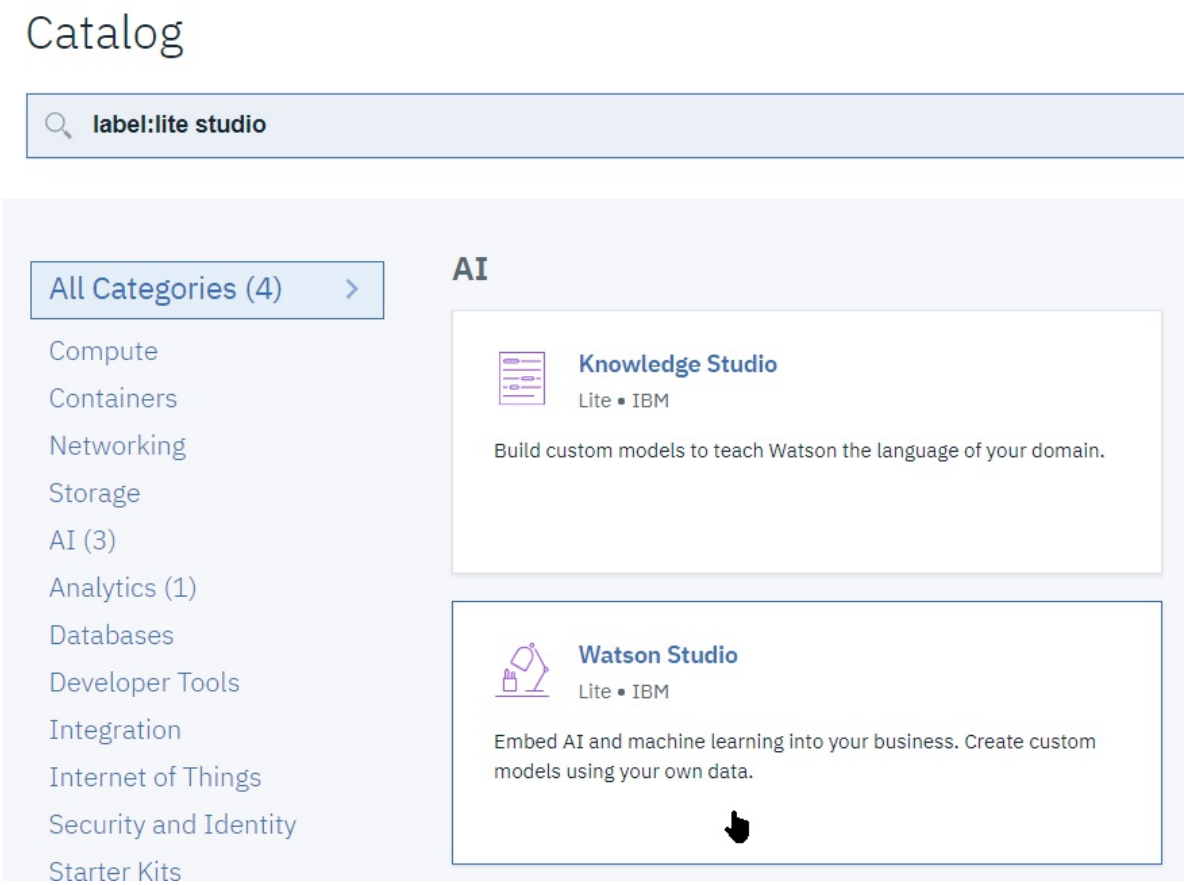
Creating a Watson Studio instance

From IBM Cloud, we will instanciate a Watson Studio service, as the anchor for the toolset within IBM Cloud. Note that this is a one-time setup, only one instance of Watson Studio per region needs to be created.

1. Log-in to you IBM Cloud account's dashboard (at https://console.bluemix.net/dashboard/apps)
2. Click the `[Create Resource]` button at the top right



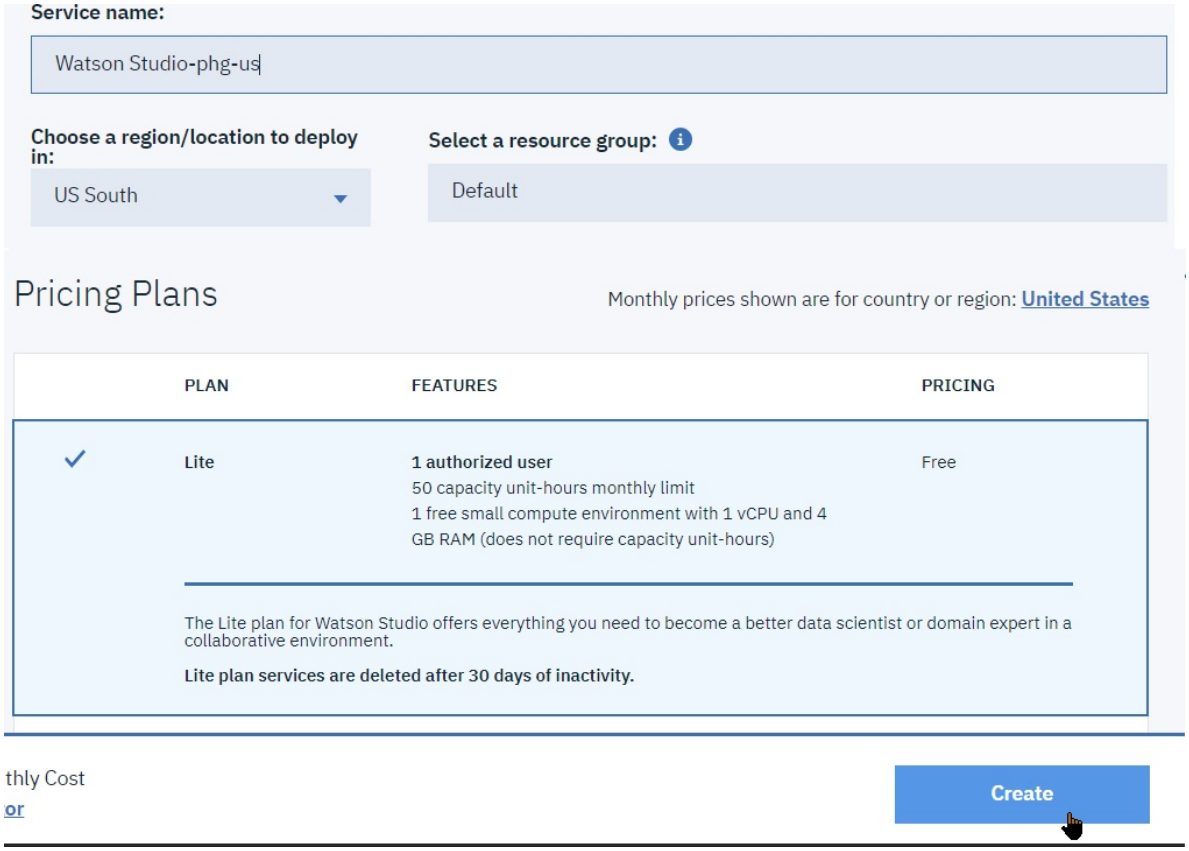
3. In the search filter field, add the single word `studio`. This should reveal the lite services having the `studio` word in their name.



and click the `Watson Studio` tile.

NOTE: Make sure to use `Watson Studio`, and *not* `Knowledge Studio`

4. You are taken to the service creation page. Although it is possible to create an instance of Watson Studio in either `US South` or `United Kingdom` regions, it is recommended to use `US South` because this is where services, including new beta ones are updated first. You can change the service name suffix or keep the suggested name. Keep the `Lite` service plan and click the `[Create]` button.

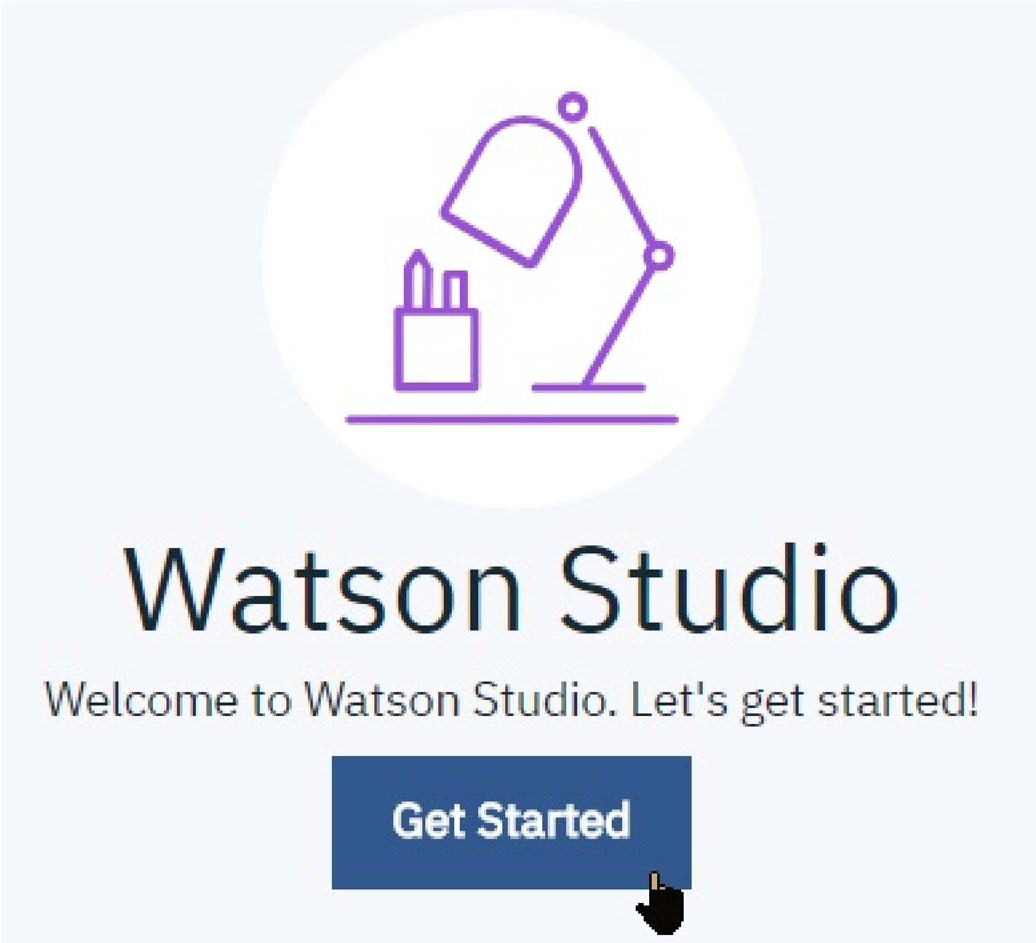


NOTE: In the rest of the labs, if you created your Waston Studio instance in the `US-South` region, you will need to use the plain URLs without prefix, e.g. `dataplatform.ibm.com`, but if you created in the `United Kingdom` region, you will need to use the `eu-gb` URLs, e.g. `eu-gb.dataplatform.ibm.com`.

Creating a Watson Studio project

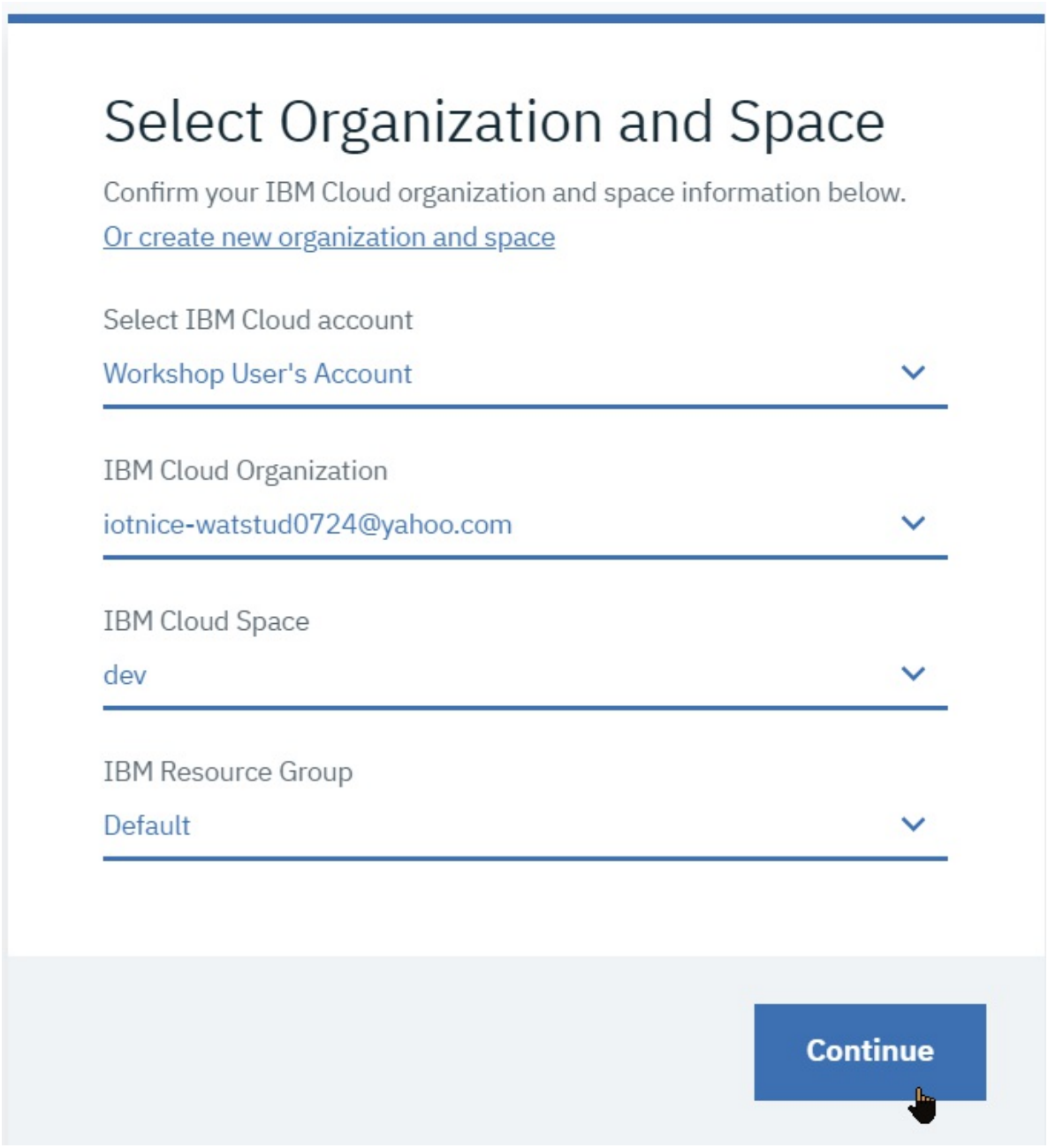
Now that we have put in place the infrastructure to work with Data & AI, we can start creating a project for a specific data handling project.

1. If not already signed-in, login to your Watson Studio environment within IBM Data Platform. For this, go back to the IBM Cloud dashboard, select the `Watson Studio` service instance, and click the '[Get Started]' button

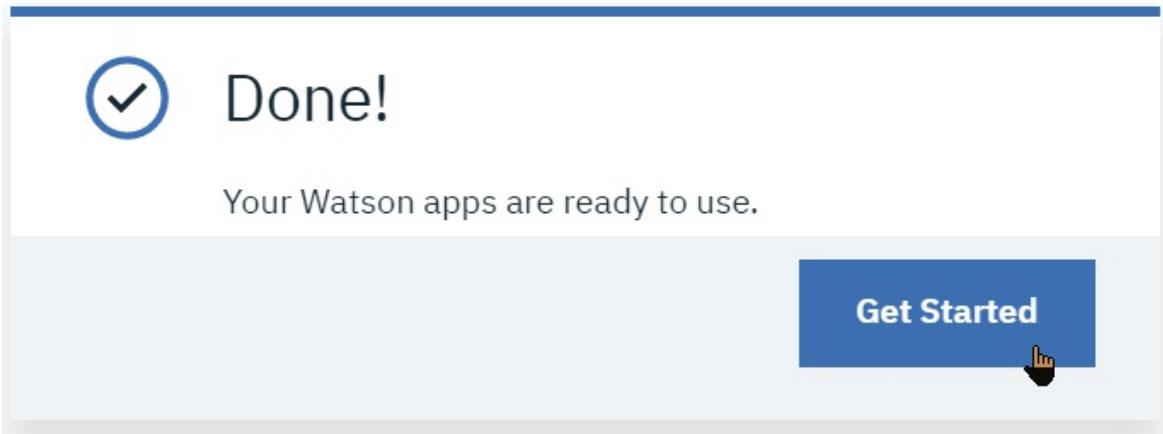


.

The first time you start the Watson Studio UI, you will be asked to confirm some details, click the `[Continue]` button:

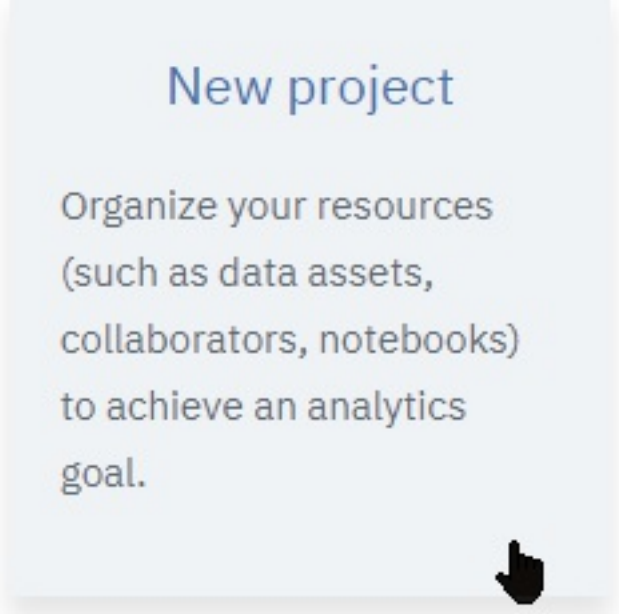


, and then validate

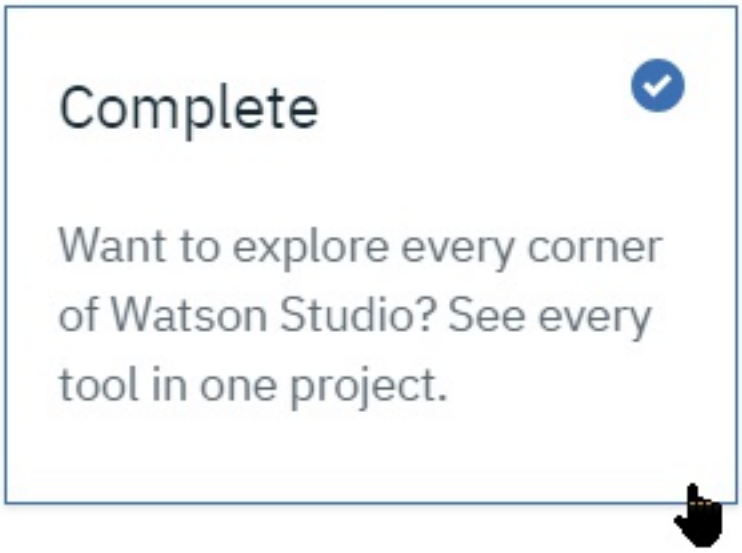


Note that you can also go directly to the service's Cloud Web UI using the URL for the region where the service has been created, either https://dataplatform.ibm.com/projects?context=analytics for 'US-South' or https://eu-gb.dataplatform.ibm.com/projects?

context=analytics for 'United Kingdom'
Create a new project using the `New Project` button tile

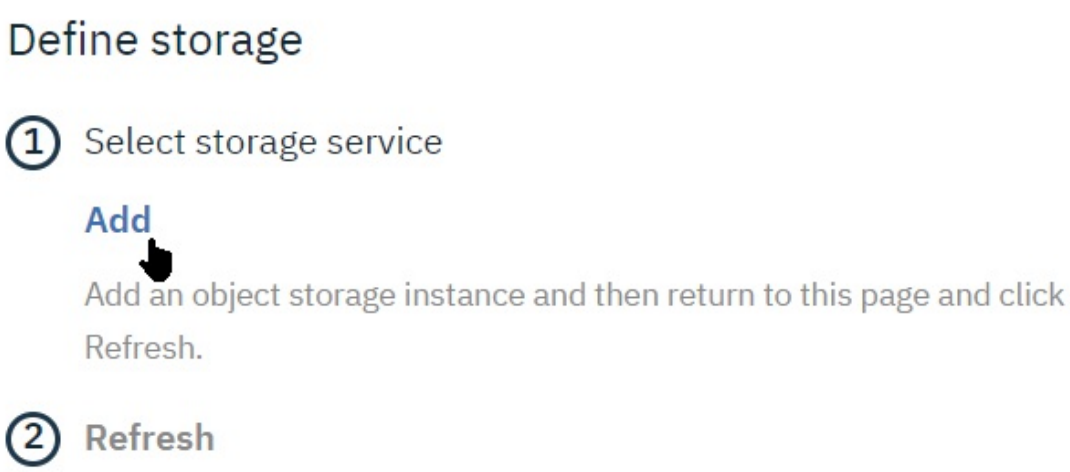


Then select a `Complete` configuration. This governs which tools are made available to the project, and can be altered later if need be



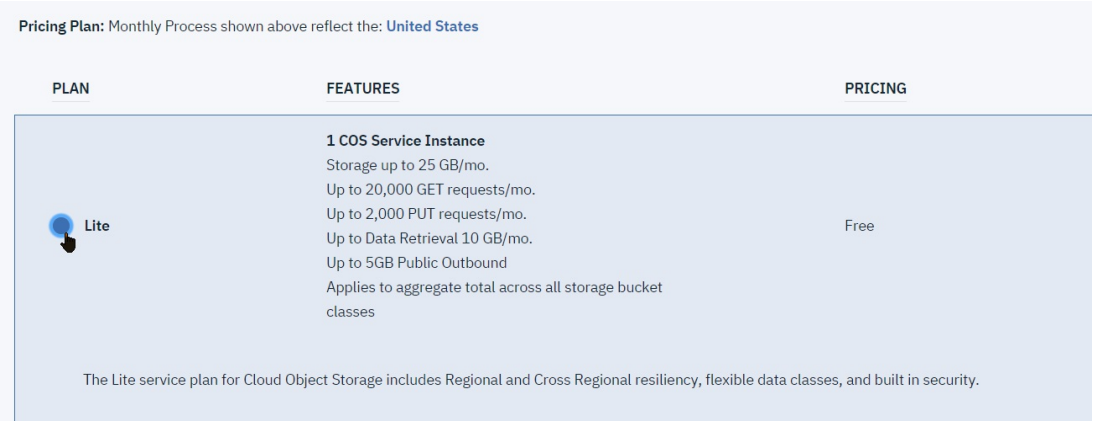
Validate with the `[OK]` button

2. Name this new project e.g. `WatStud_Workshop`.
   Note that you will want to leave the 'Restrict who can be a collaborator' unchecked, it will make sharing the project with another account more straightforward.
   Watson Studio stores its file-like artifacts into an instance of `Cloud Object Storage`, we will create a COS service instance at this stage.

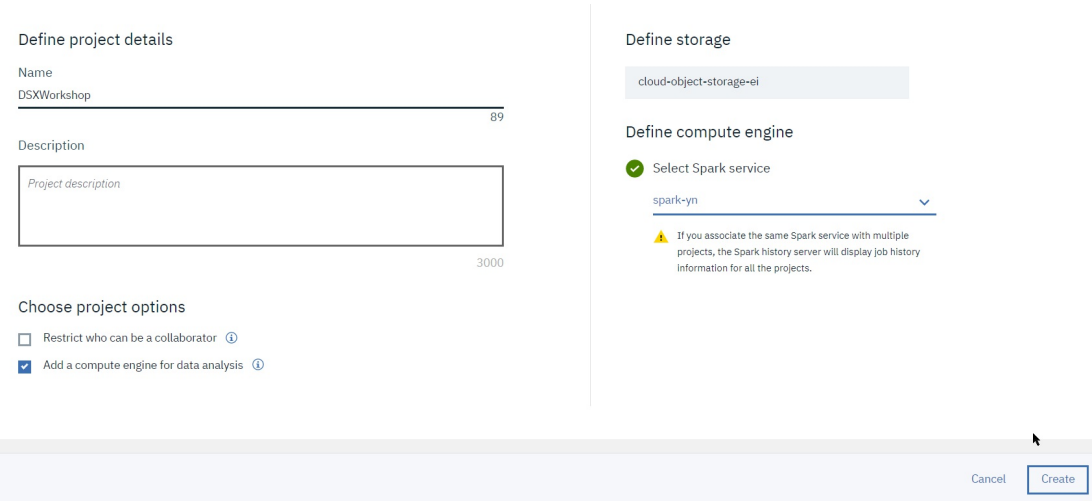


Currently, your only choice is **IBM Cloud Object Storage**. Information stored with **IBM Cloud Object Storage** is encrypted, resilient and dispersed across multiple geographic locations, and accessed over HTTP using a REST API.
Each project and catalog has its own dedicated bucket.

1. Select the Lite Plan



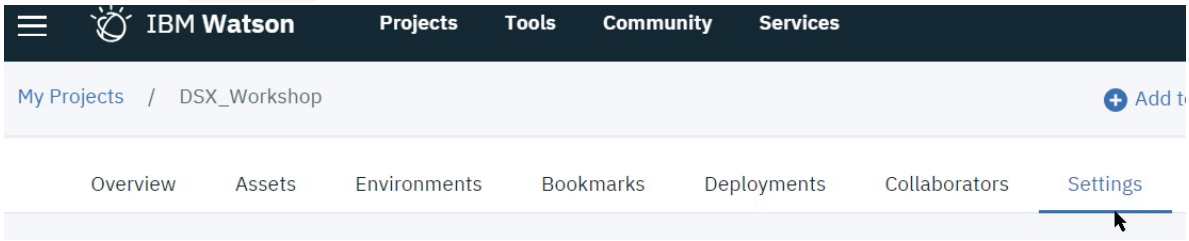
2. Accept the default names for resource group and Service name
3. Back to the Project creation page, select Refresh then the new Object Storage service instance
4. Finally, click **Create**.



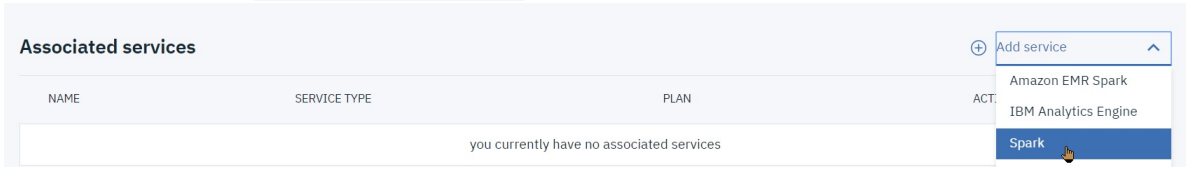
Spark Service engine setup

Some of the **IBM Watson Studio** operations uses Spark at the backend, so we will need to associate a Spark engine to our project.

1. Switch to the `Settings` tab

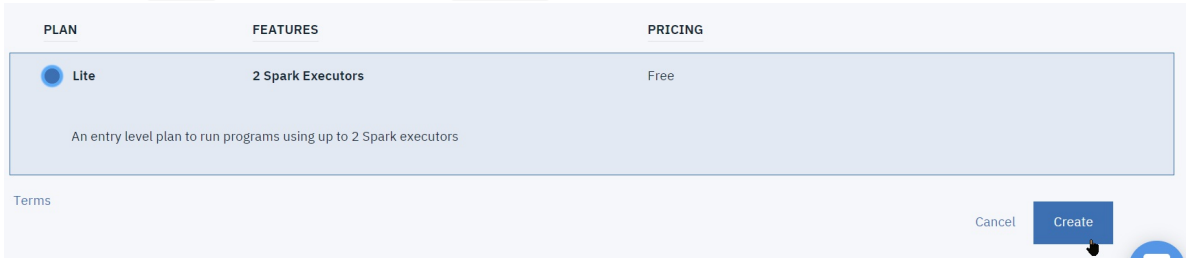


2. scroll-down to the `Associated Service` section

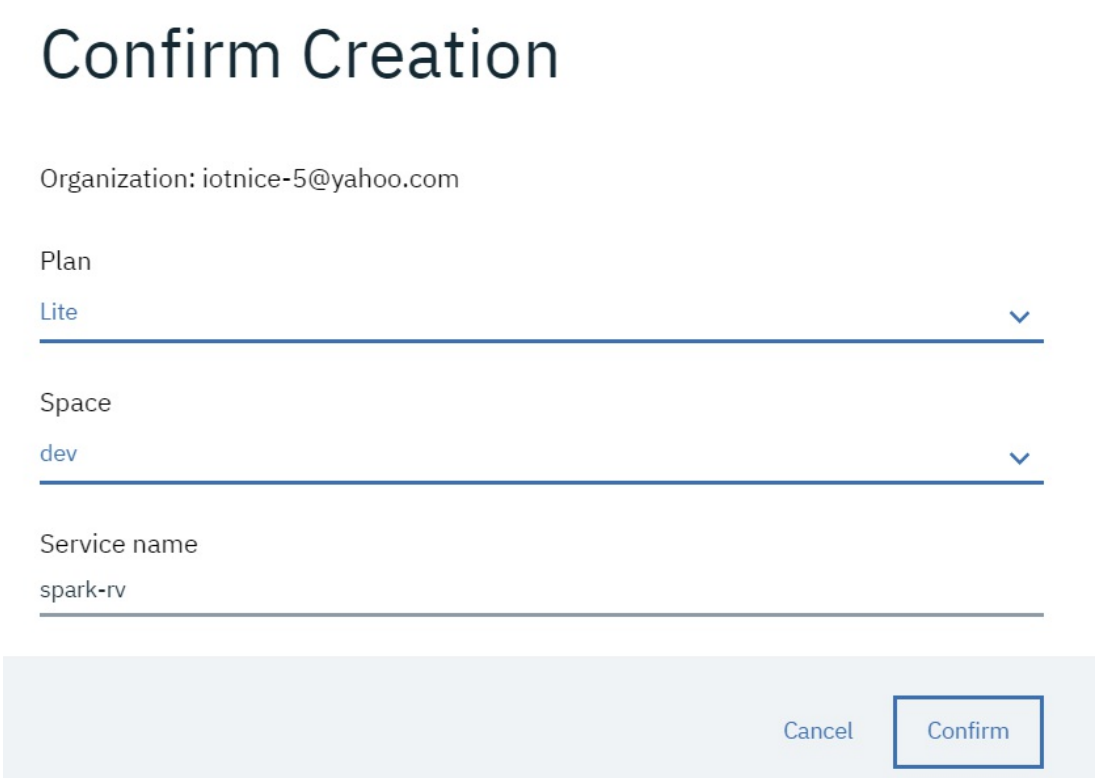


, select `Spark` from the `Add Service` pull-down menu.

3. Select the `Lite` pricing plan and `Create`



> Note that if you created your **IBM Watson Studio** in the eu-gb region, you will need to make sure that the Spark service creation is in https://eu-gb.dataplatform.ibm.com
>
> 1. Keep the defaults on the 'Confirm Creation' panel and select `Confirm`.

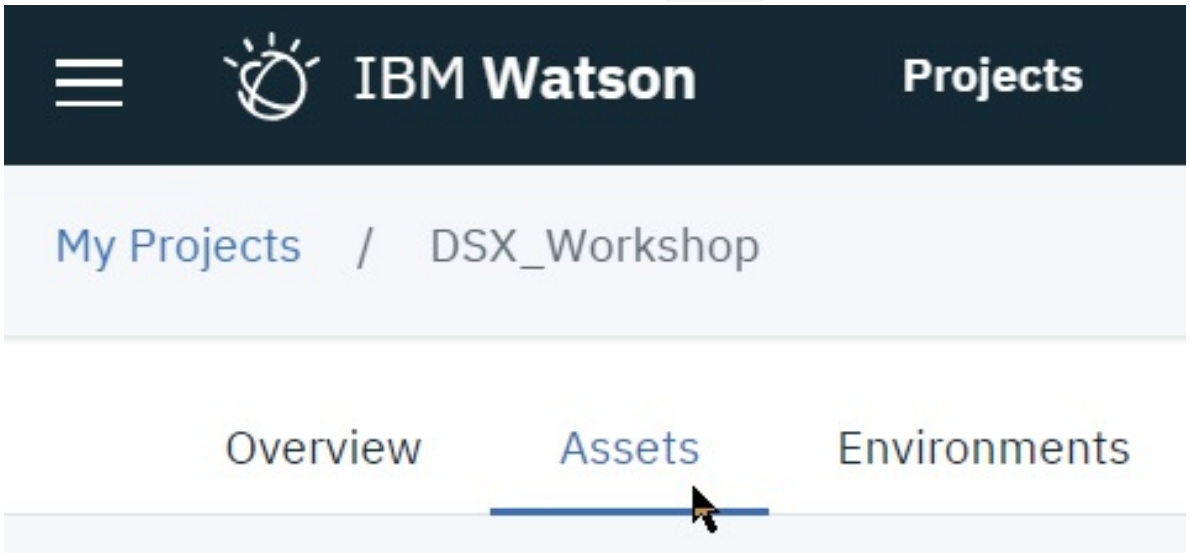


You've just associated a Spark service to your project.

Loading Data Assets for the project

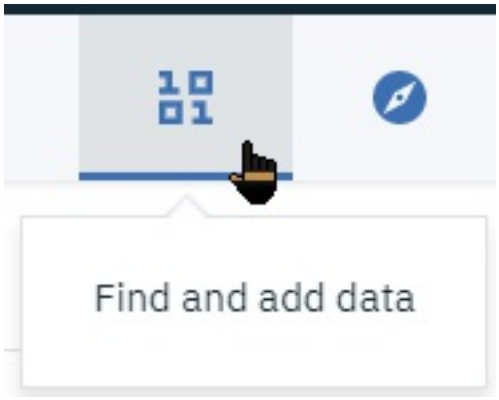
We will load some of the files used during the Hands-On lab as Data Assets available to your project.
The files are available in the Box folder.

1. In your **IBM Watson Studio** project, switch to the `Assets` tab:

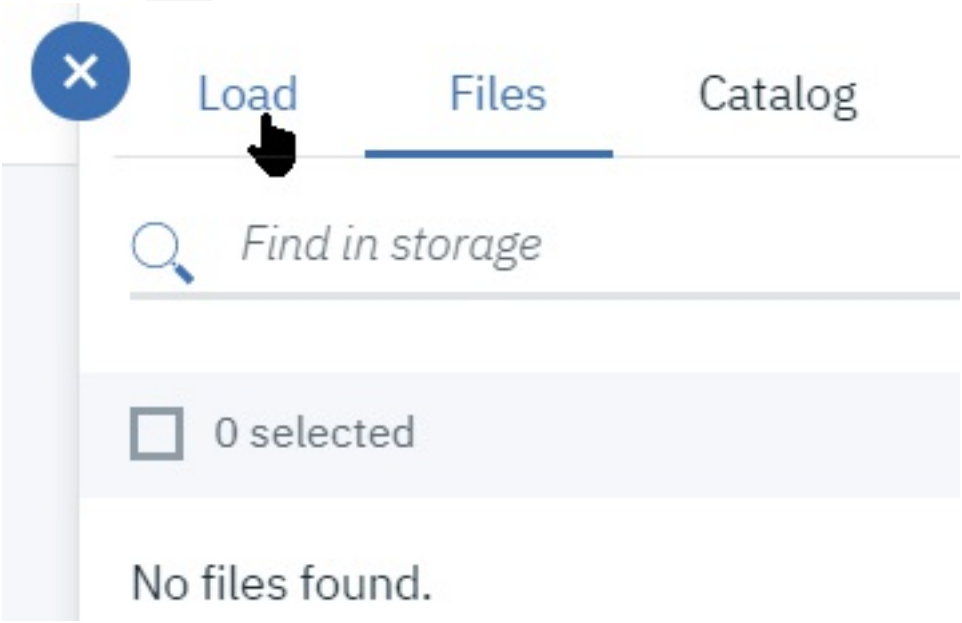


2. Initially the Data Assets list should be empty. If not opened yet, open the Data Pane by

selecting the `1001` icon at top right:



3. Select the `Load` tab



4. Click Browse to add files that you will have downloaded to your computer's disk from the Box folder.
   Among the files that we will need, you can start loading the following ones:
   The source data for these files can also be found at their original lolocation on the web.

File name	Original location
`GoSales_Tx.csv`	https://dataplatform.cloud.ibm.com/exchange/public/entry/view/ba9a
`cars.csv`	https://dataplatform.cloud.ibm.com/api/exchange/actions/download-dataset/c81e9be8daf6941023b9dc86f303053b
`201701-citibike-tripdata.csv`	https://www.citibikenyc.com/system-data

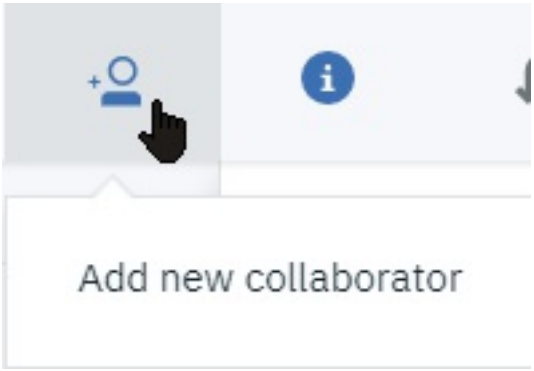
5. Once done, the files will show up in the `Data assets` list.

Project collaboration

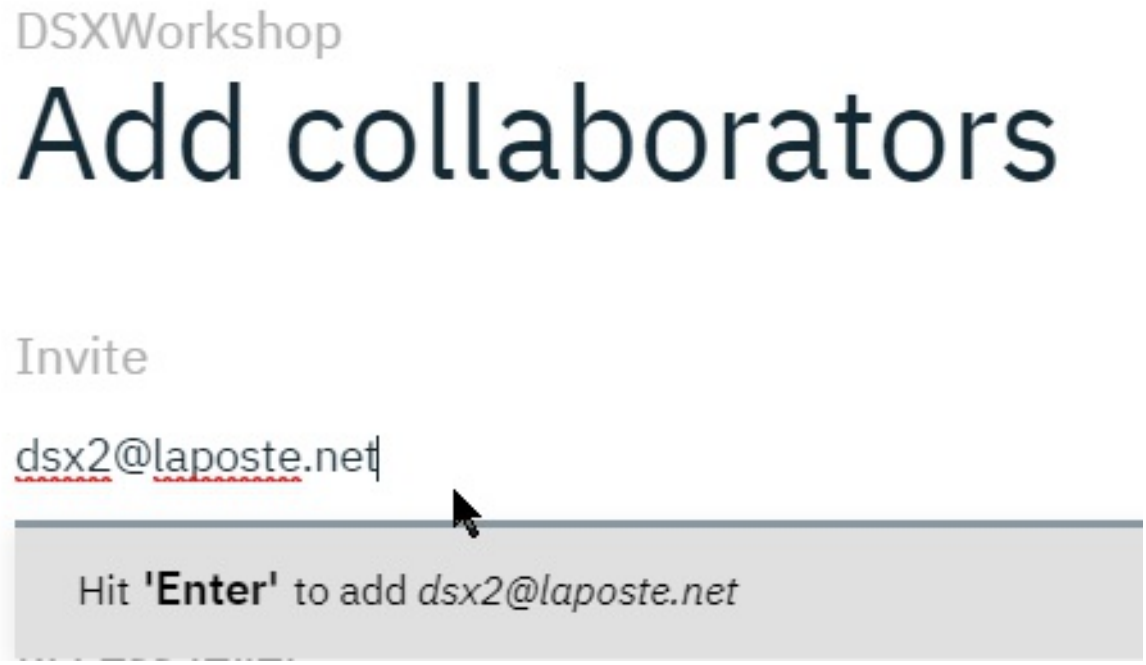
One of the strengths of **IBM Watson Studio** is to allow to easily collaborate on shared projects.

* Optional *

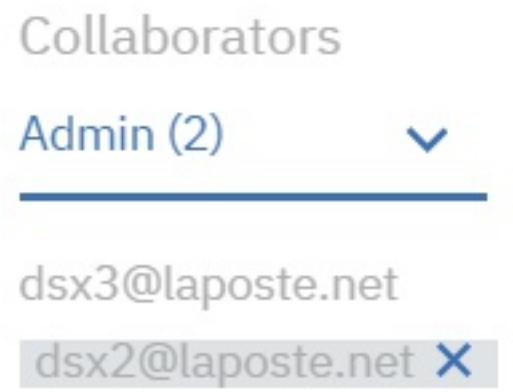
If you have for example another **IBM Cloud** account, you can add that other account as a collaborator on this `DSX_Workshop` project:
(Or you can share this with your class neighbour)



- Select the `Add new collaborator` button
- Enter the e-mail address of another account



- Select an access level, Admin will allow full control, the click `Add`



- The new collaborator shows up in the summary
- Finally click `Invite` to validate the change
- If you login with another account to DSX, you will be abe to access this project too.

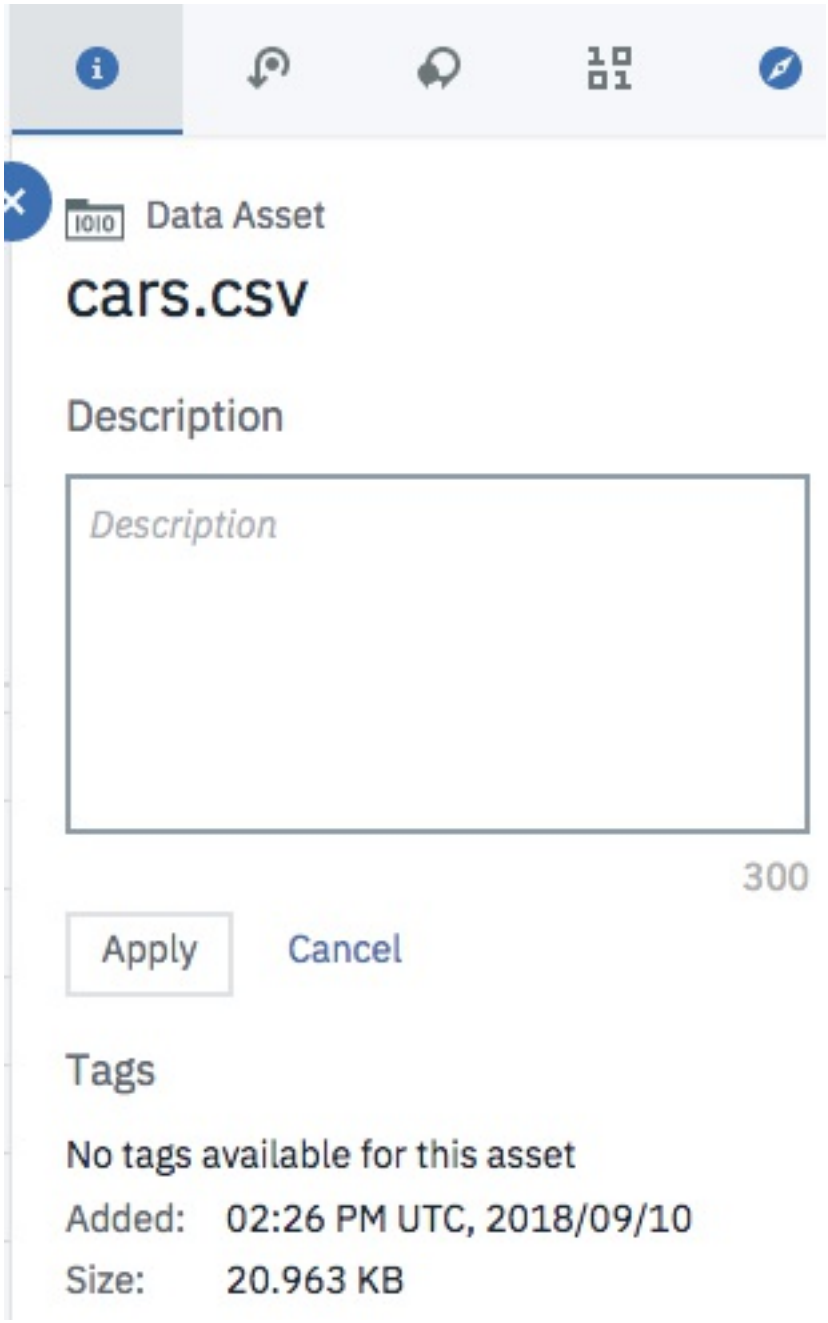
Quick assessment of the Data Asset

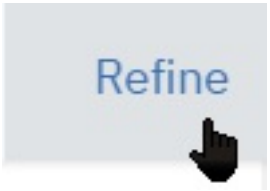
You can quickly browse through sample from one of the Data Assets, so as to get an idea of the data format.

For example:

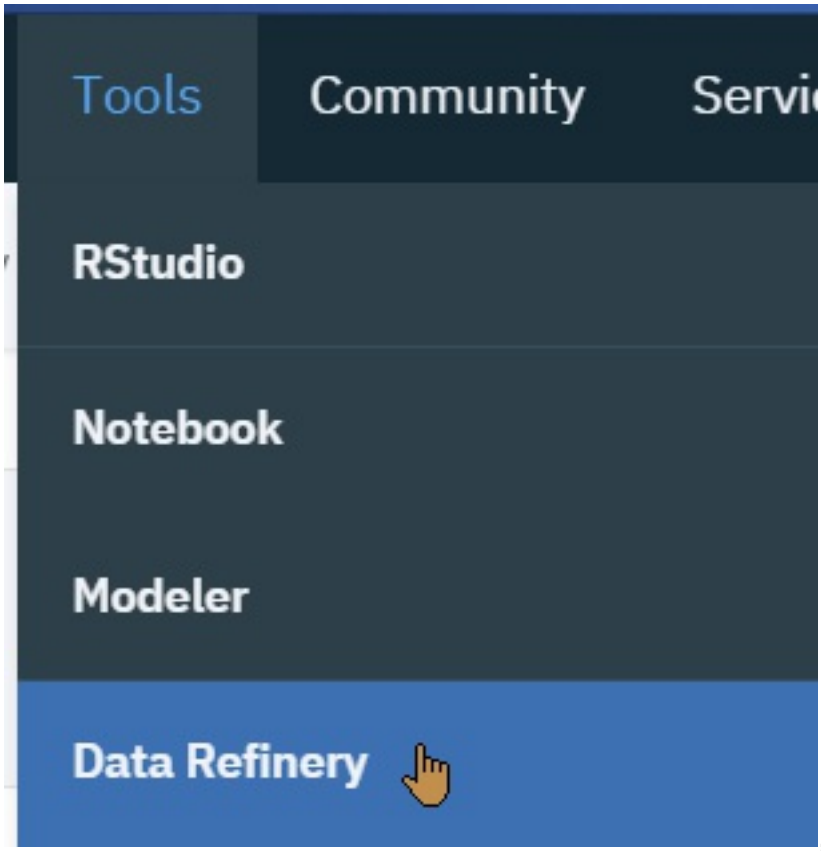
1. From the `Assets` tab in the project page, select the `cars.csv` data asset by clicking on it
2. This opens a preview of the data in tabular format. Data set has 9 columns and 406 rows.

> Note that you can change the Data Asset metadata such like the **Description** and the **Tags** from the **Information** side bar and clicking on the **pencil** to go in edit mode.

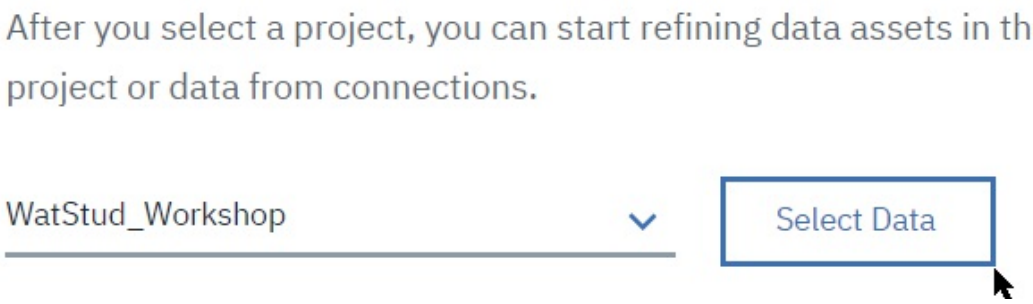


3. Select the `Refine` button 

> This will open the Data Refinery tools of **IBM Watson Studio** which allows to cleanse and shape data, customize it by filtering, sorting, combining or removing columns, and performing operations.
> NOTE: If the `[Refine]` button is not present or grayed-out, navigate to the `Tools/Data`



`Refinery` menu , then select your project

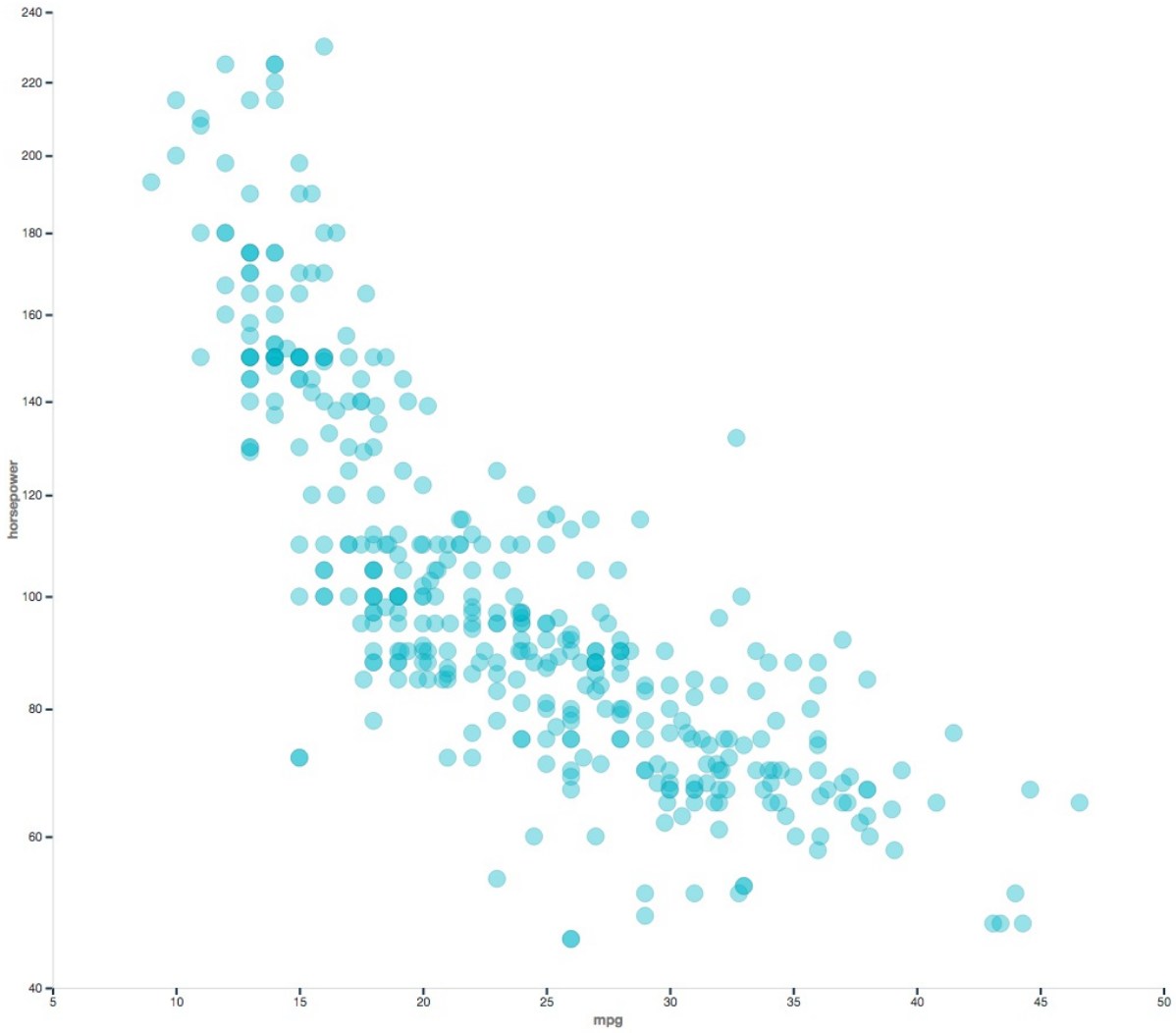


, and finally `[Add]` the intended file.
As you manipulate your data, you build a customized data flow that you can modify in real time and save for future re-use. When you save the refined data set, you typically load it to a different location than where you read it from. In this way, your source data can remain untouched by the refinement process.

4. switch to the `Vizualisations` tab in the view that opens

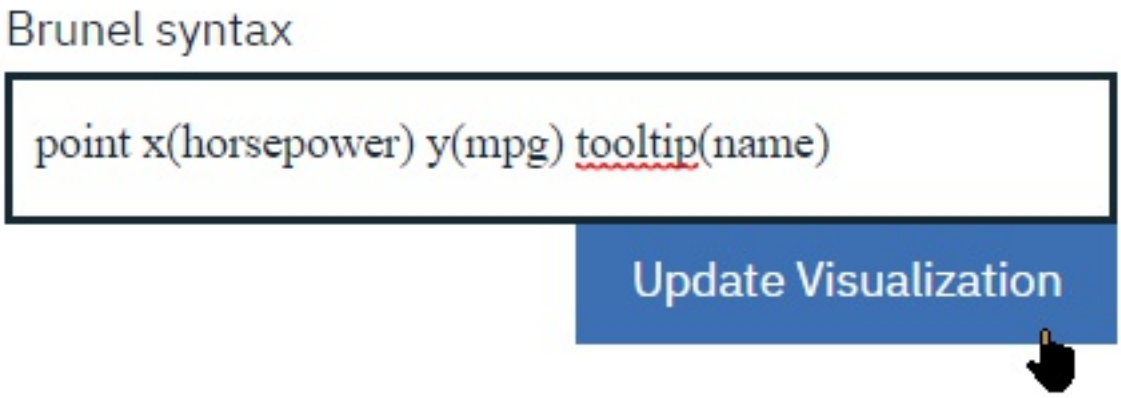
5. in columns, enter `mpg` , `horsepower` , the select graph type `Scatterplot`

6. the graph plots the two data columns to show their relationship. We will see in the Visualization Hands-On Lab how to programatically generate a similar graph. Note the **Brunel** notation generated to display the chart. There will be more on **Brunel** in the coming labs.



Interpretation of the horsepower/mpg scatter plot

Scatter plots are very useful diagrams to quickly show
if there is a relationship between two atributes.
Here we see that there is a general trend that cars with higher horsepower tend to have lower miles-per-gallon. This is kind of an expected outcome.
But we also see that the curve is not quite a straight line, it looks more hyperbolic.
Moreover, some points are clearly not on the general trend, these are called **'outliers'**. You can hover at the point at [hp: 132, mpg: 32.7].
In order to see the car brand and model, change the tooltip to `name` in the `Brunel syntax` entry, so that it shows `point x(horsepower) y(mpg) tooltip(name)`, and click `[Update Visualization]`:



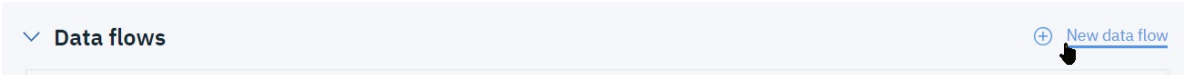
Hovering over the oulier point will show the `datsun 280-zx` as the car with high hp but relatively higher mpg than the other cars.
Similarly, below the curve, the `Ford Maverick` has low hp for low mpg.

Data Refinery

The Data Refinery in **IBM Watson Studio** is an integrated ETL feature which allows to easily implement data transformation pipelines in the form of a sequence of data operations applied to a data set called **data flows**.

In this section, we will use Data Refinery to cleanse and filter the contents of the `201701-citibike-tripdata.csv` data file. This file is one of the monthly reports of bike sharing usage for NYC, provided as an Open Data asset from https://www.citibikenyc.com/system-data.
This file is one of the monthly reports of bike sharing usage for NYC, provided as an Open Data asset from https://www.citibikenyc.com/system-data.
We will use **IBM Watson Studio** to get a first understanding of the data, and apply some transformations to reduce the volume and scope of data to analyze.
Note that this file is pretty large, with over 725000 lines of data, and a raw file size of over 120MB, in CSV format, which is not the most efficient to store data (the zipped content is about one fifth of the raw data)

1. From your project's `Assets` tab, scroll down to the `Data Flows` section and select `New data flow`:



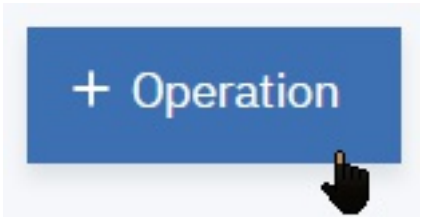
2. Select the `201701-citibike-tripdata.csv` data file, click on the small eye icon to have a preview, then click the `Add` button at the bottom right. Note that if the `[Add]` button is not active, you will have to select `[Add]` from the main panel.

3. Data Refinery will show a table with the 1000 first rows as a sample. As part of the operations we will want to apply to the data, we will:

    i. Rename the columns so as to remove blanks that could cause handling issues later on
    ii. Specify actual data types for non-string fields. This applies to the numeric 'Trip Duration', 'Birth Year' and the 4 station latitude and longitude columns.
    iii. Compute an Age column from birth date.
    iv. Extract date and time slot columns from the Start and Stop time columns.

    > Notice that as you perform data transformations, the steps of your data flow are added on the left side bar.

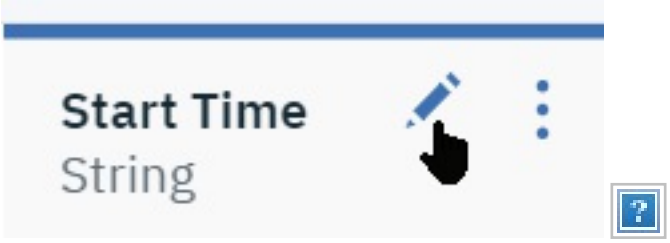
1. Columns renaming:

1. For the first column, select the `Add Operation` 

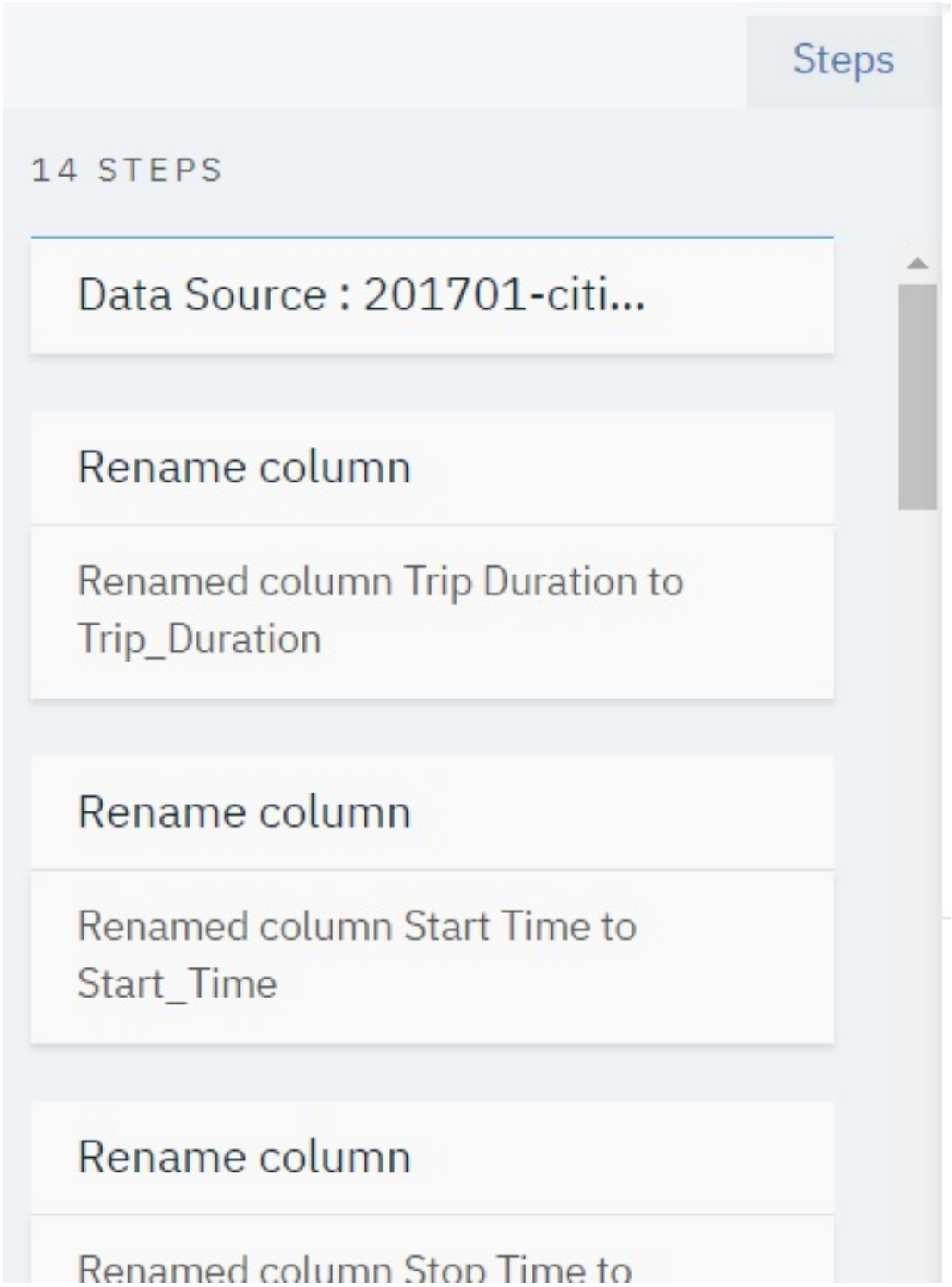
icon, then the `Rename`  operation, and replace the column name by the same with spaces replaced by underscores, e.g. `Trip Duration` becomes `Trip_Duration`.

> NOTE that it's a good idea to cut the column name before clicking the `Next` button on each rename to save retyping.

2. For the other columns, there is a faster way to add a rename operation, by clicking the **pencil** icon in the column header and changing the name there:



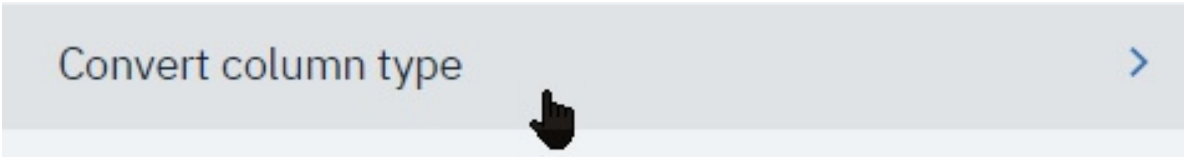
As you proceed through columns renaming, you will see operations being listed in the right-hand side panel. You should now have 14 operations listed in the steps list:



Save you work with the Disk icon.

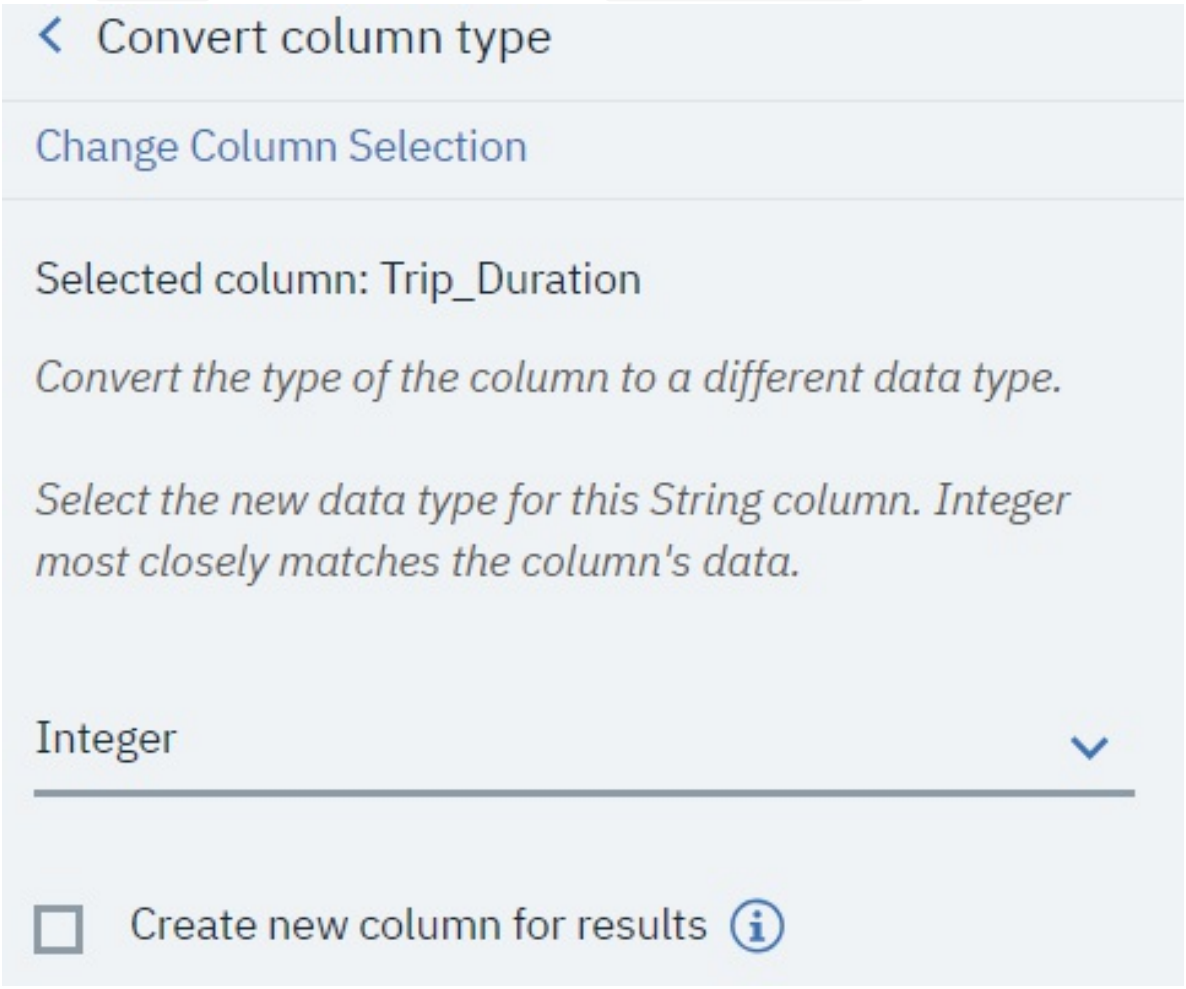
2. Data type changes

1. Now add `Convert Column type` operations for `Trip_Duration` and `Birth_Year` columns:

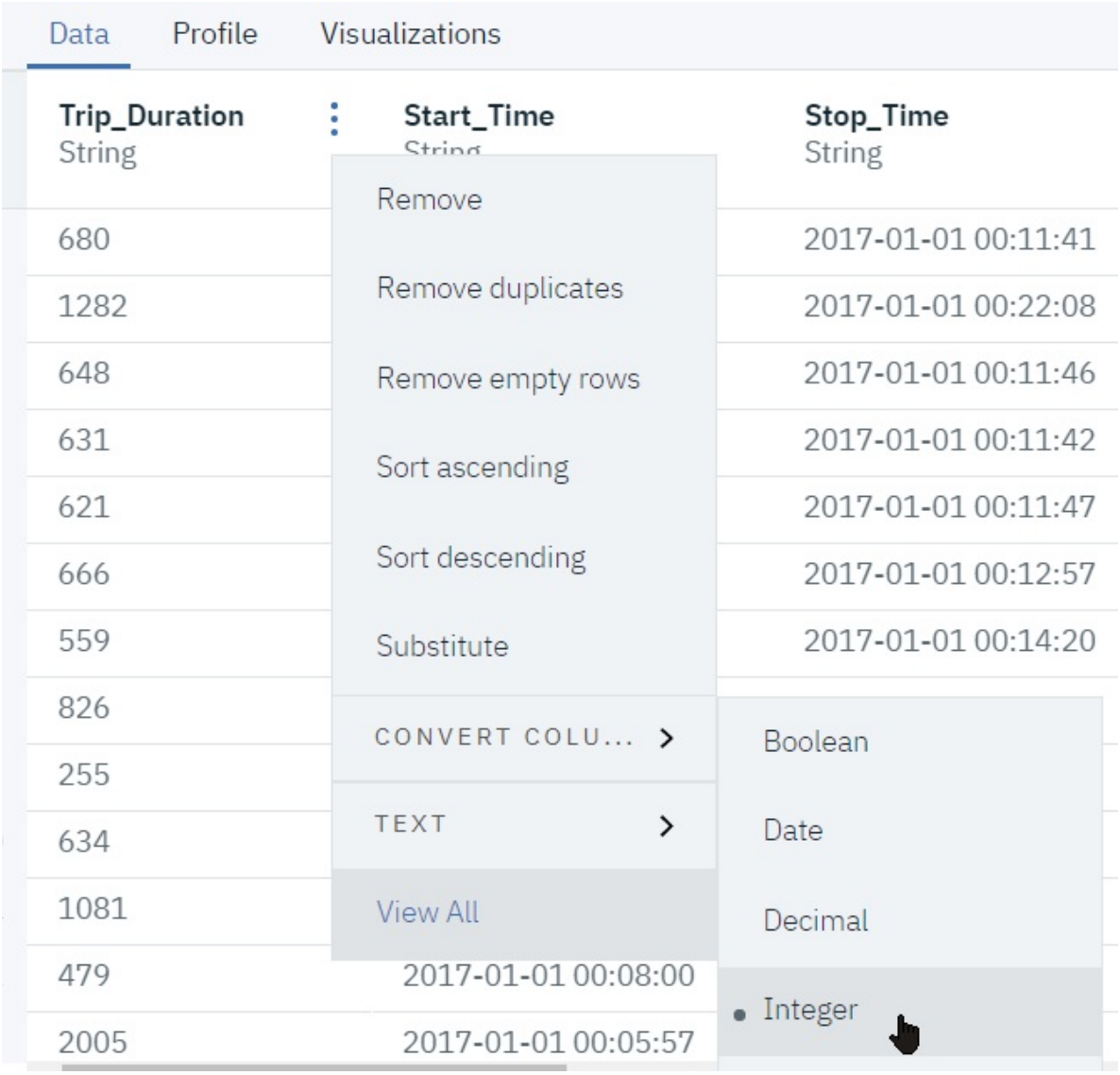


.

2. Select `Integer` as the target type. Leave the `Create new column` unselected:

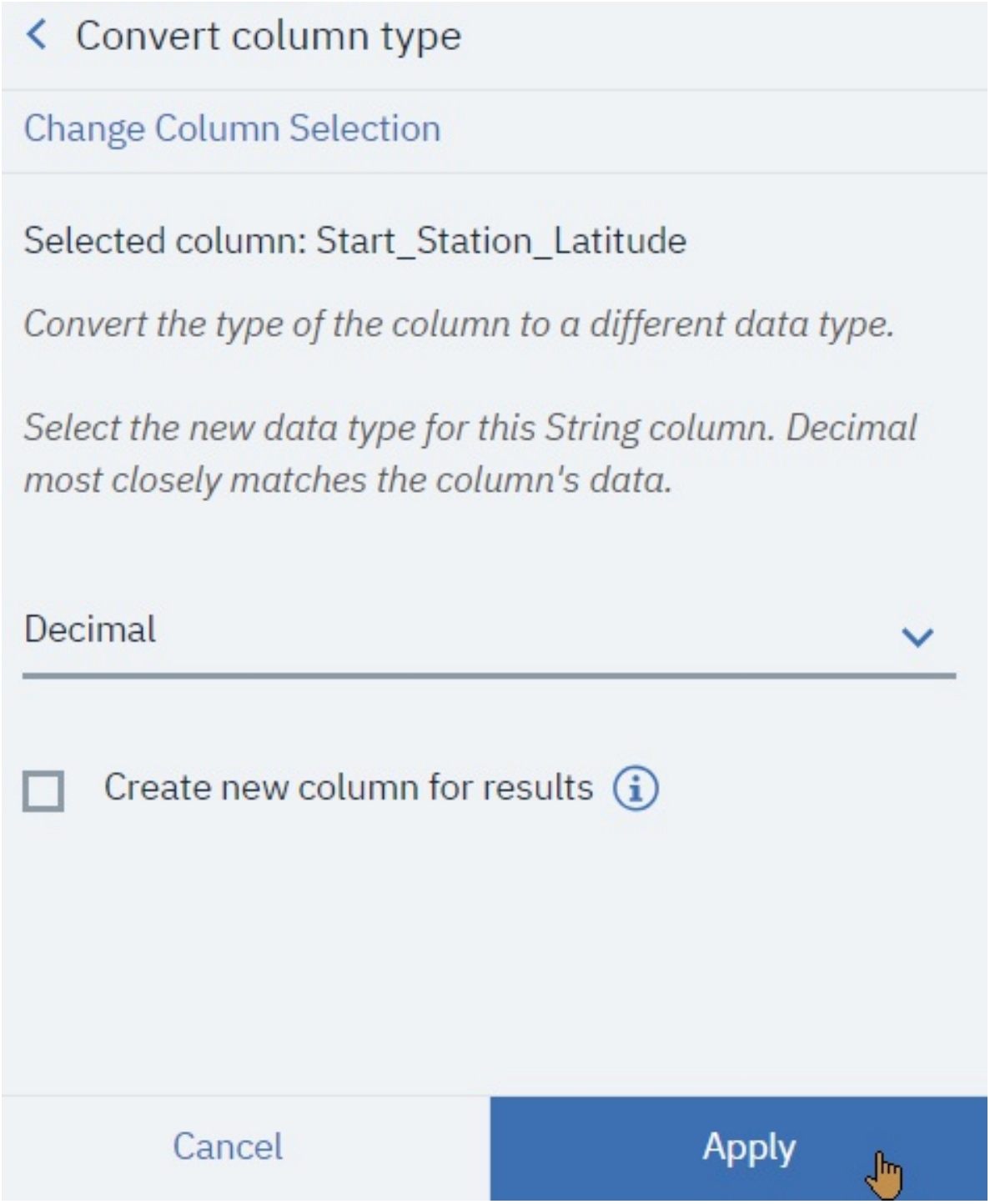


You can add the operation either from the `[+ Operation]` button at the top left, or from the column's header context menu:



. Note in this case how the `Integer` type is suggested with a small blue dot at its left.

3. Do the same for the 4 Start/End Latitude and Longitude columns, using `Decimal` as the type:



. Also note the suggested `Decimal` type here.
You should now have 20 steps recorded.

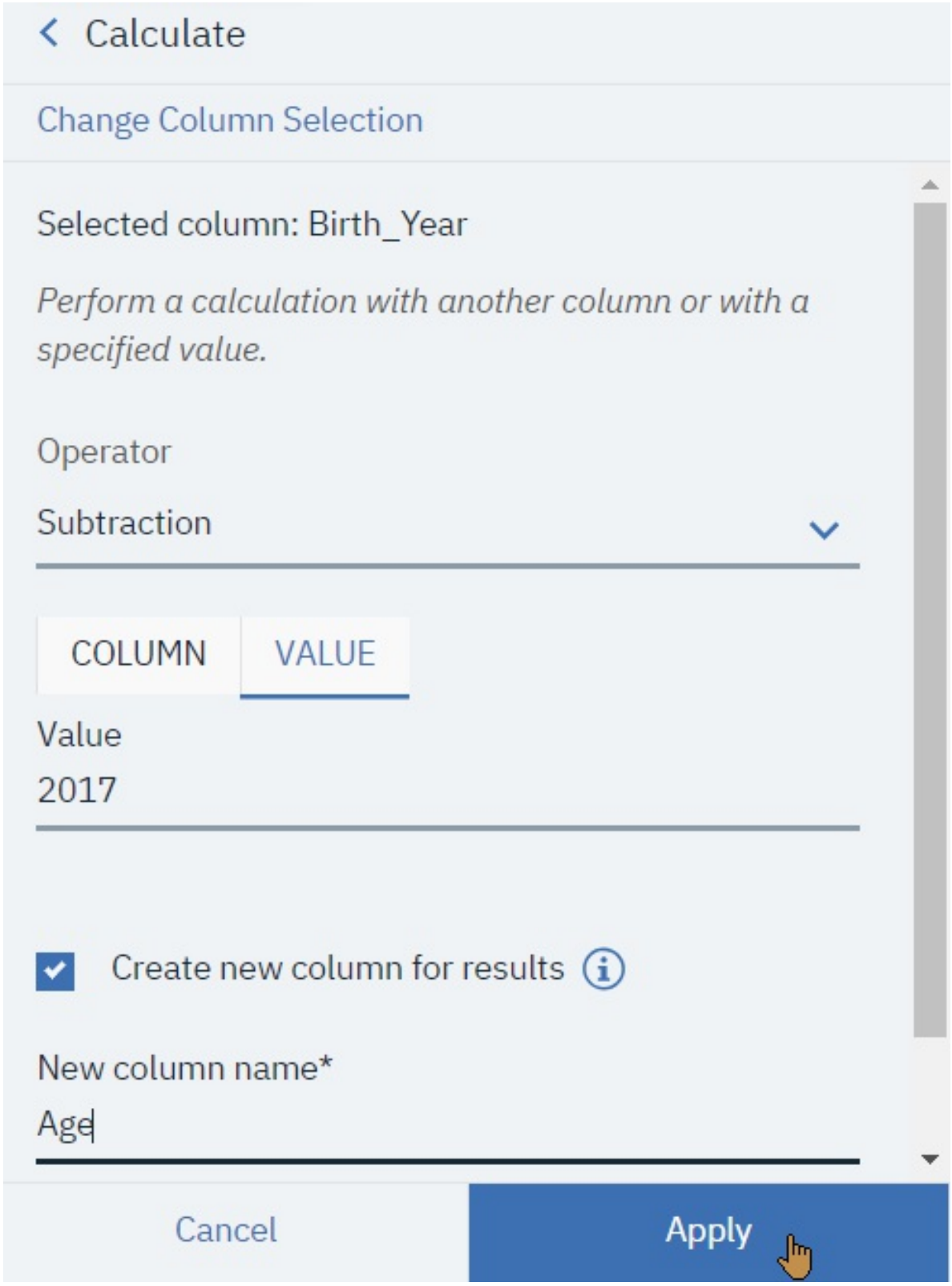
3. Feature Engineering: Additional computed column

We will compute the age from the birth year. Since we have only the birth year, we will just use 2017 as teh reference year from which to substract the birth and get an approximate age.
We will also remove all rows where `Age` is missing.

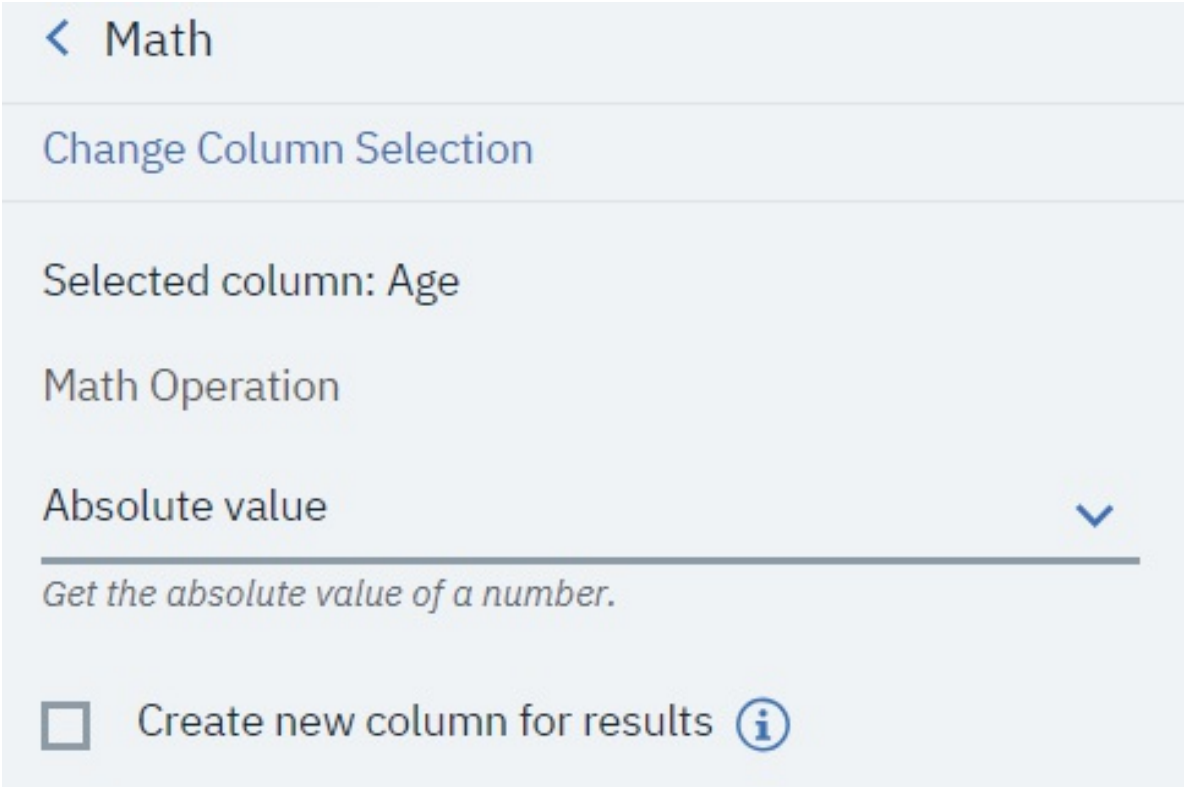
1. Add a `Calculate` operation, select `Birth_Year` as column, `Substraction` as operation,

and `value` 2017.

2. Check the `Create new column for result` checkbox and enter `Age` as the new column name:



3. The compute age comes out negative, we will add a `Math` / `Absolute Value` operation to the `Age` column:



> Note that at each step, you can see a preview of the data in the table.
> Verify that the values for `Age` column seem correct in the preview.

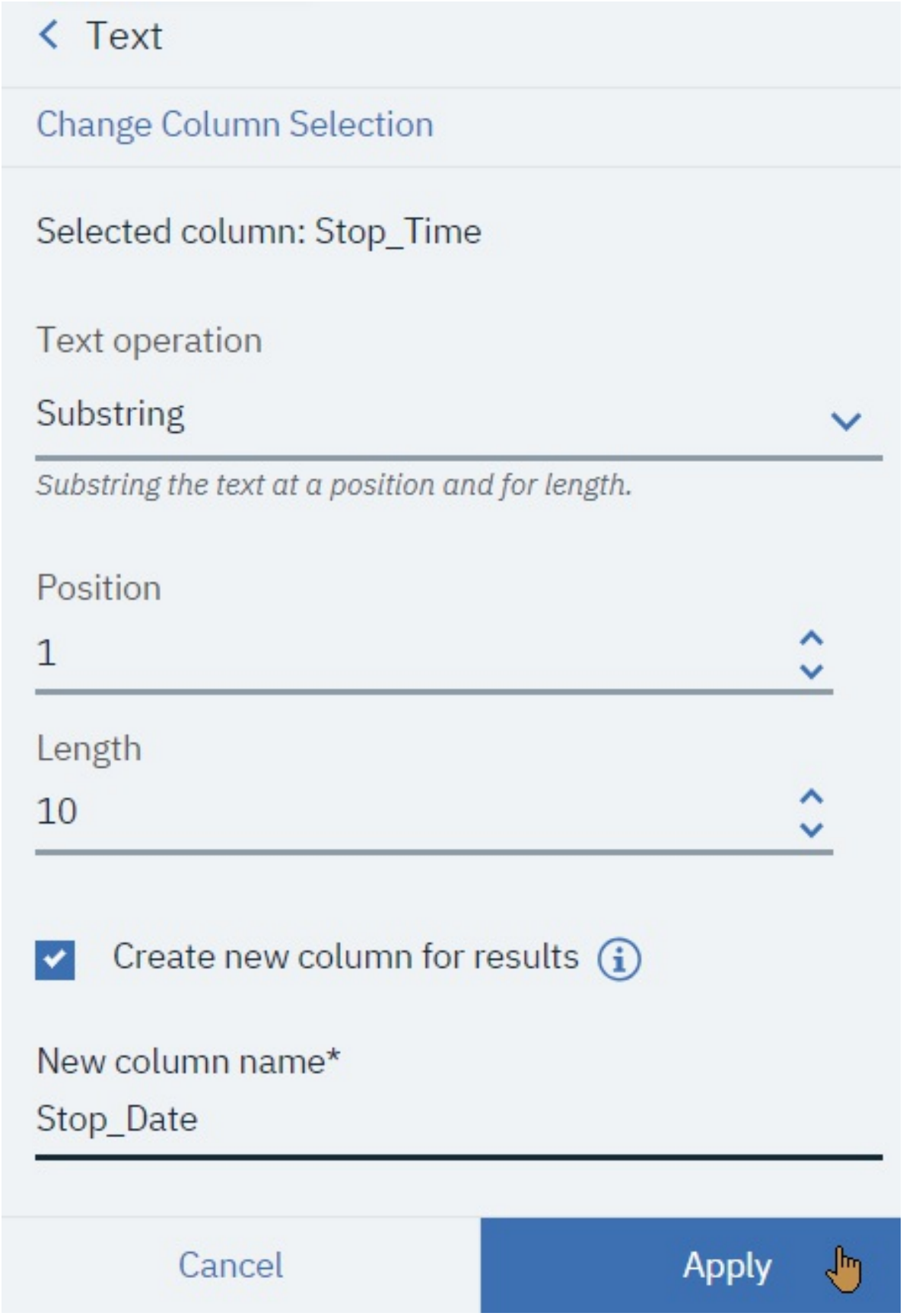
Also note that here we've used the UI-driven point-and-click style column ETL operations. It is also possible to add column operations using the guided formula operations entry at the top of the table preview.
For the age extraction operation, you could have entered a formula such as:

```
mutate(Age = 2017 - Birth_Year)
```

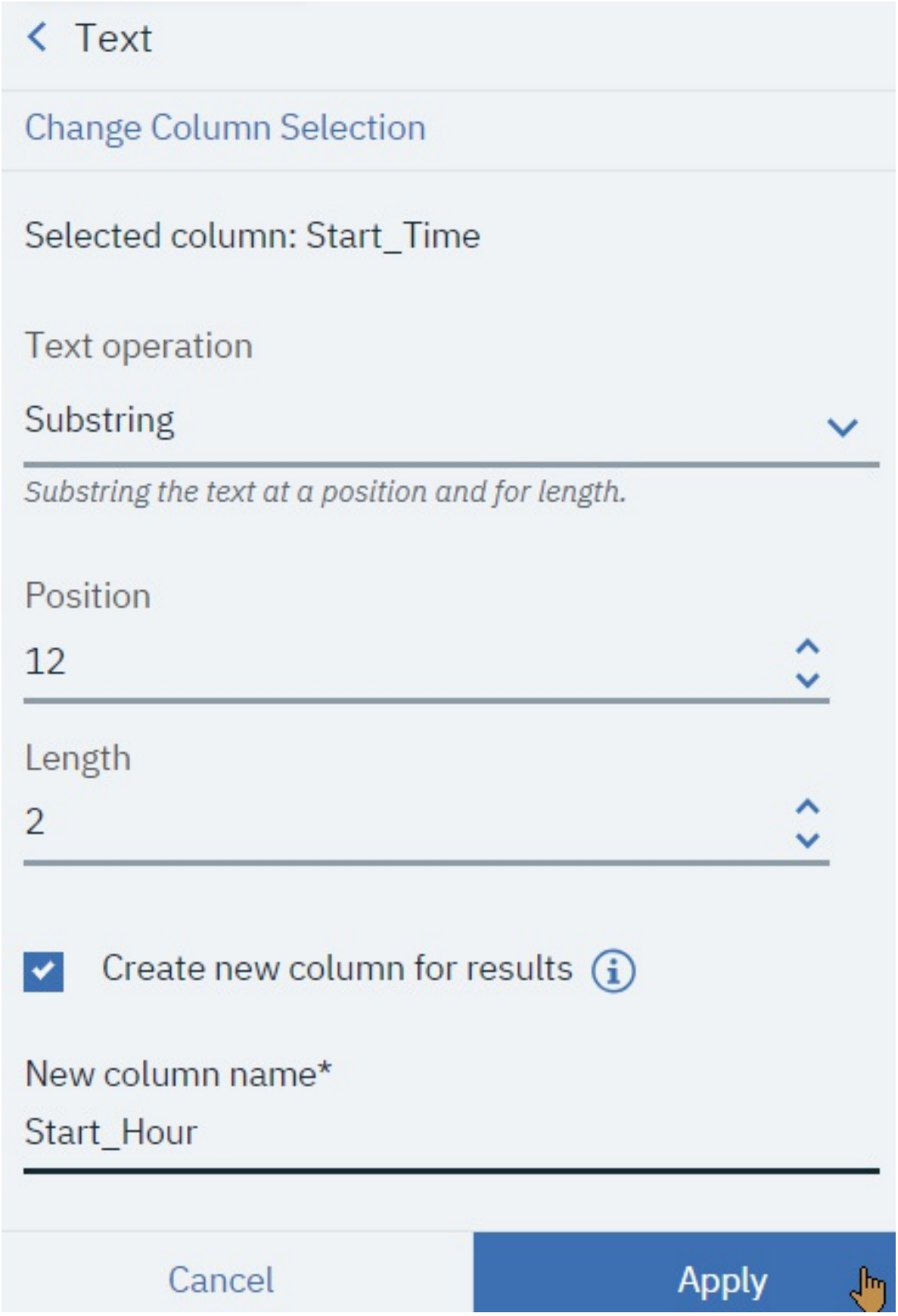
4. Feature Engineering: Process the time columns

Finally, we will process the time fields. We will split out the date in the 10 first characters into new columns, and convert to `Date` type. We will also extract the hour slot from the time into a new column typed `Integer`. For each of the `Start/Stop_Time` columns:

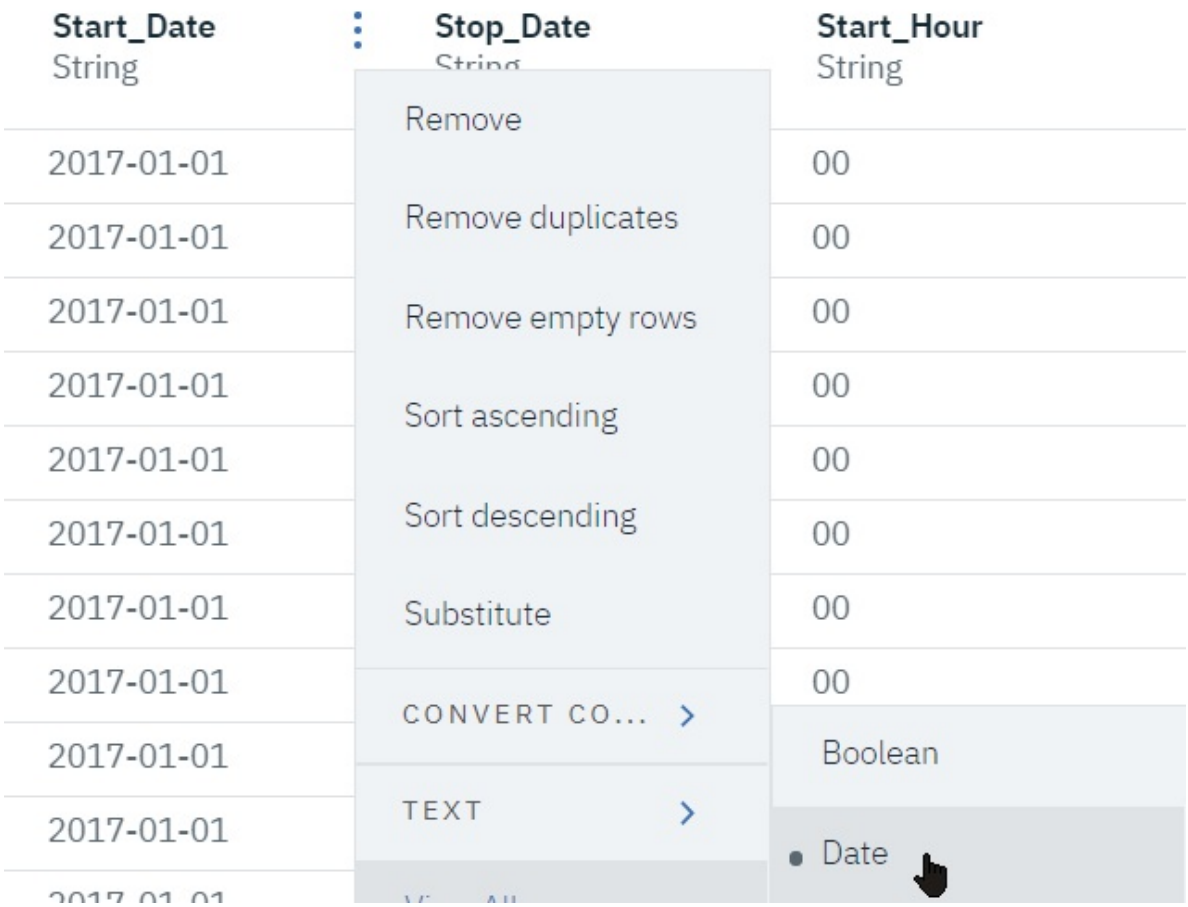
1. Add `Text` / `Substring` operations, which creates additional columns `Start_Date` and `End_Date`. We will take a substring from position 1 and length 10:



2. Similarly, Add a `Text` / `Substring` to create new columns `Start_Hour` and `Stop_Hour` from substring position 12, length 2:



3. Finally, change the type of the `Start/Stop_Date` columns to `Date` type using the menu from the column header:



, and `ymd` format.

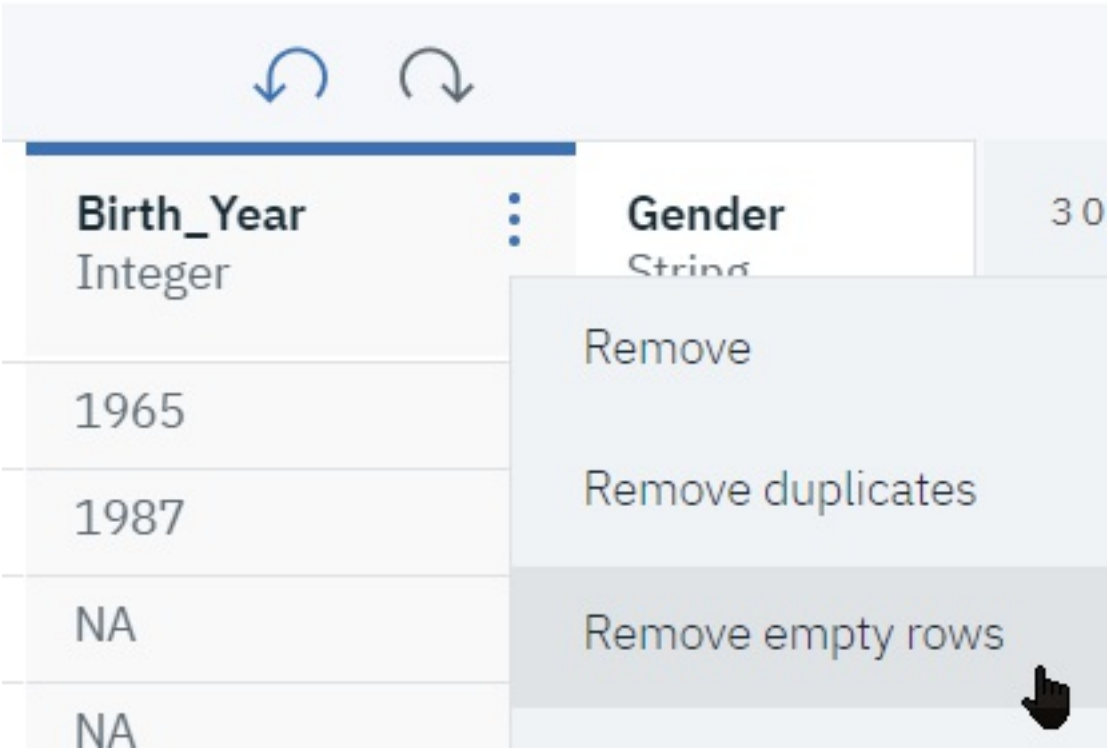
4. Same for the `Start/Stop_Hour` columns to type `Integer`

You should now have 30 steps defined.
Save your flow.

5. Data Cleansing: remove columns with no Birth_Date

Some columns are missing the Birth_Year demographics information. We will remove the rows that have this field empty.
Note that we could (and maybe should) have added this step before computing the Age column.

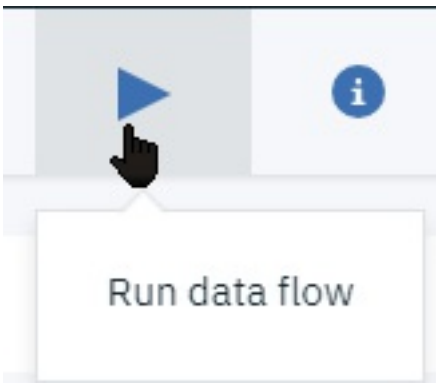
1. From the `Birth_Year` column header context menu, select the `Remove empty rows` operation:



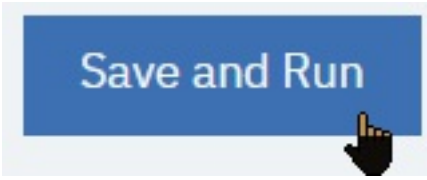
You should now have 31 steps defined.

Apply Data Flow pipeline to the input files

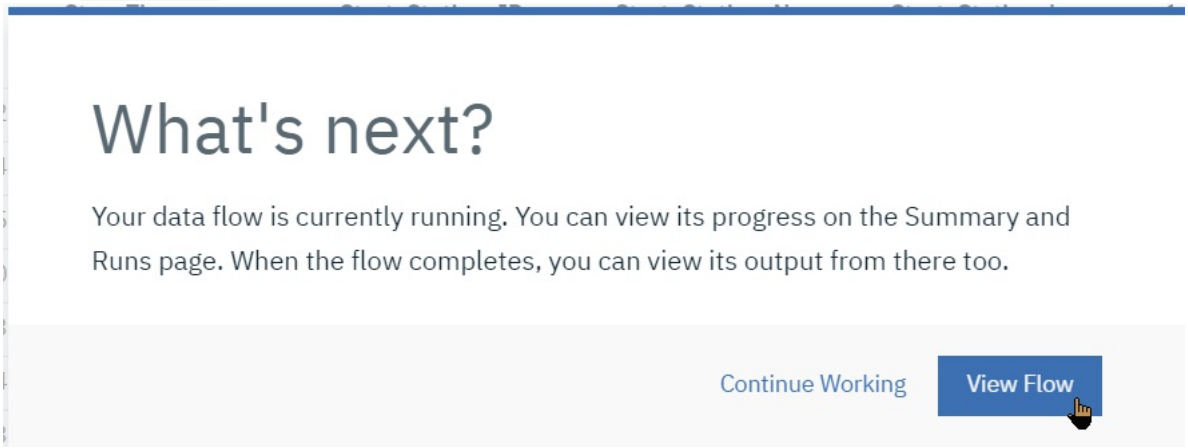
We will now process the entire file with our data cleansing and feature engineering pipeline.

1. Click on the 'Run' icon at the top right: 

2. Change the output target file name to `201701-citibike-tripdata_cleansed.csv`, and use format as `CSV` 

3. Finally click the `Save and Run` button: 

> Notice that you could have schedule your data flow to run on a defined time of the day.

1. Elect `View flow` on the next window:


2. Wait for the flow to complete processing. The flow executes in the Spark engine and should take less than a minute to execute over the 700 thousands records.
3. Once executed, you can go back to the project assets, and you will find the generated `201701-citibike-tripdata_cleansed.csv` file that you can browse by clicking on it. We will reuse this file in the second set of Labs.

Conclusion of Data Refinery section

We have experienced the Data Refinery which is Watson Studio's integrated ETL (Extract, Transform and Load) tool. You have seen that the tool is designed to define ETL operations without coding, even though it can be complemented by formulas.

In a Data Science pipeline, ETL tools are almost always required as first steps in the data processing. It allows to perform Data Cleansing and Feature Engineering.

A word on file type conversion

Data Refinery also allows to generate data Asset output in 'Parquet' file format, which is a file format specified as part of the Apache Hadoop project, optimized for columnar data storage and retrieval in a Hadoop or more generally Data Science environment.
Parquet is not as efficient as zipping a file, but can readily be used by data processing tools, and it carries meta data information such as column types. In the case of this input file, the resulting parquet conversion would yield a file of about 42 MB, vs 116 MB for the raw CSV file and 23 MB for the zipped CSV.

Strech lab for Data Refinery

Data Refinery also allows to perform aggregation of columns and join operations across two Data Assets. As a stretch lab, you can investigate how to create a Data Refinery Flow which generates a table which holds only the station names and IDs, and total number of bike departures and returns per day.

Using notebooks for data exploration

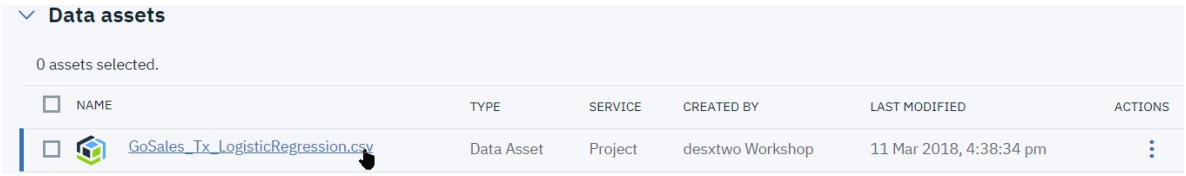
In this section, we will start exploring the data from a file which holds customer sales observations related to buying behavior of customers of an outdoor equipment company regarding tent purchases, using a Jupyter notebook.

This is a different approach to data analysis than the GUI-driven tools such as Data Refinery, here the paradigm is to perform programmatic operations on data files rather than GUI driven. Each approach has its pros and cons, and selecting one versus the other can be a matter of personal preference.

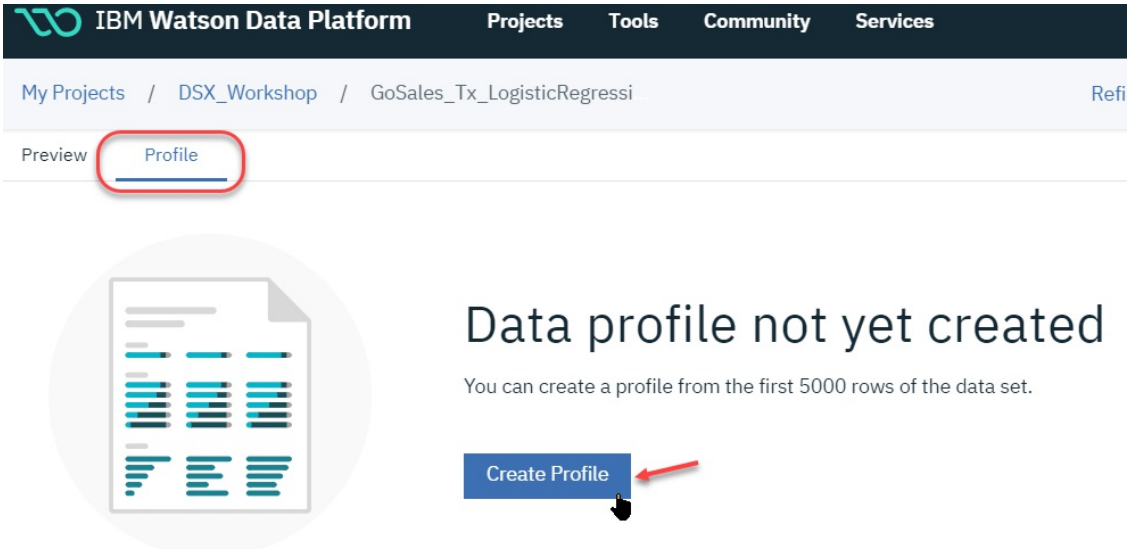
Explore the data set

Ensure that the `GoSales_Tx.csv` file is part of the data assets, so that we can start to have a look at the data:

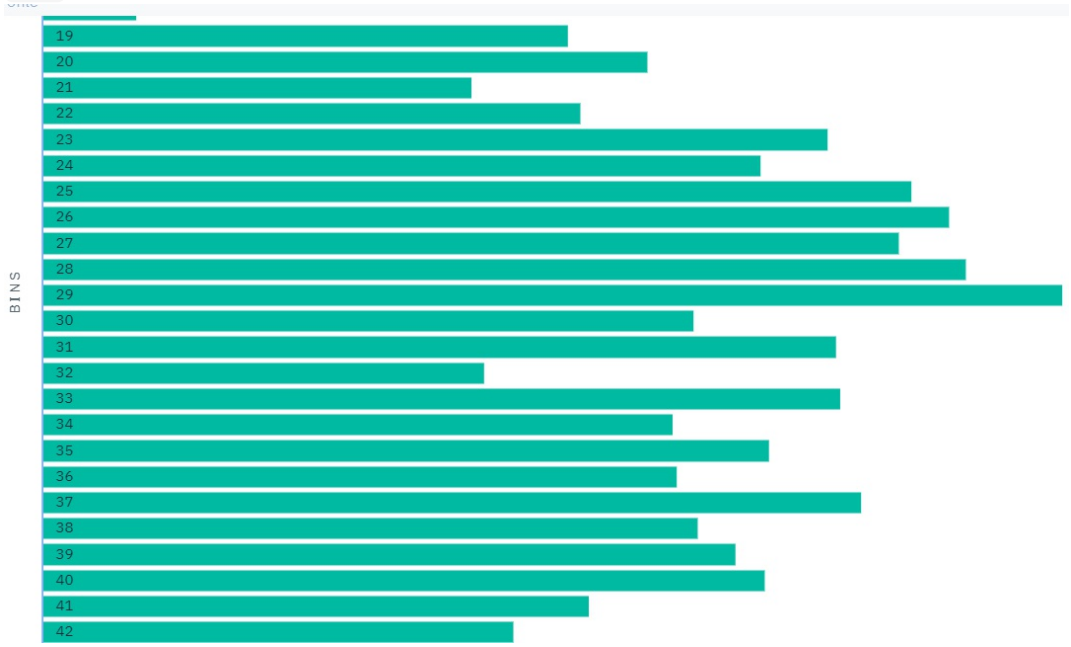
1. open the corresponding asset by clicking on the file name from the list



   This opens into the tabular preview, where we can discover the data structure:
   | `IS_TENT` | `GENDER` | `AGE` | `MARITAL_STATUS` | `PROFESSION` |
   | Type: String | Type: String | Type: String | Type: String | Type: String |
   So there are basically 4 features that can drive the buying decision held in the `IS_TENT` column.
2. To go further in the analysis, we will create the Profile for the data:
   - Select the `Profile` tab and then the `Create Profile` button.

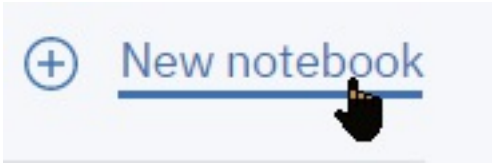


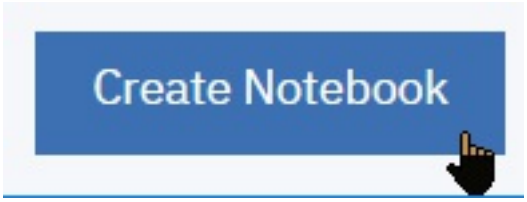
- After a while, the data profile is computed on the first 5000 lines.
- This gives a rough idea on the structure of the data through the content of the columns in statistical terms:
  - `IS_TENT` is detected as a boolean with roughly 10% occurences of `TRUE` (509 out of the 5000 sample)
  - `GENDER` has slightly more Male than Female.
  - `AGE` distribution shows a peak in the 24-30 years, with an average of 34:



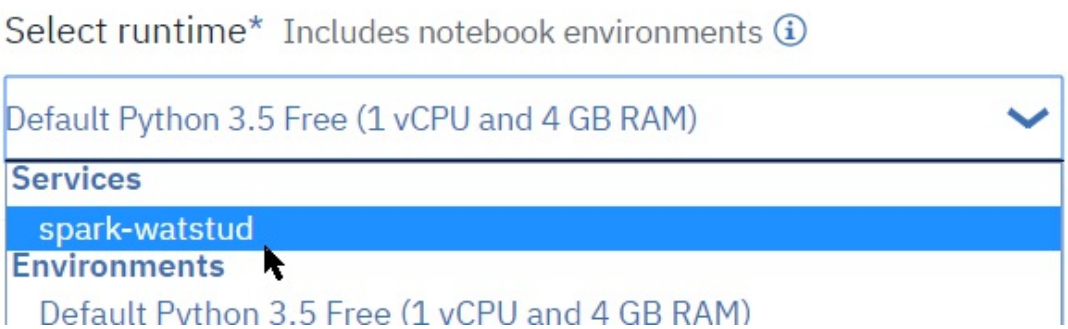
  - `MARITAL_STATUS` has half of the sample as married
  - `PROFESSION` shows almost half of the sample unspecified, with 8 distinct professions.

3. This gives a first-level overview of what to expect. We will now use the `GoSales_Tx_Analysis_cleared.ipynb` notebook for more data analysis:
   i. Go back to the Project page

ii. In the `Notebooks` section, select 

iii. Select the `From file` tab, scroll down to `Choose file` and select the `GoSales_Tx_Analysis_cleared.ipynb` file 

iv. In the bottom-right section below, select the Spark runtime:



v. **Open** the notebook. From that point on, follow the instructions that are within the Notebook.