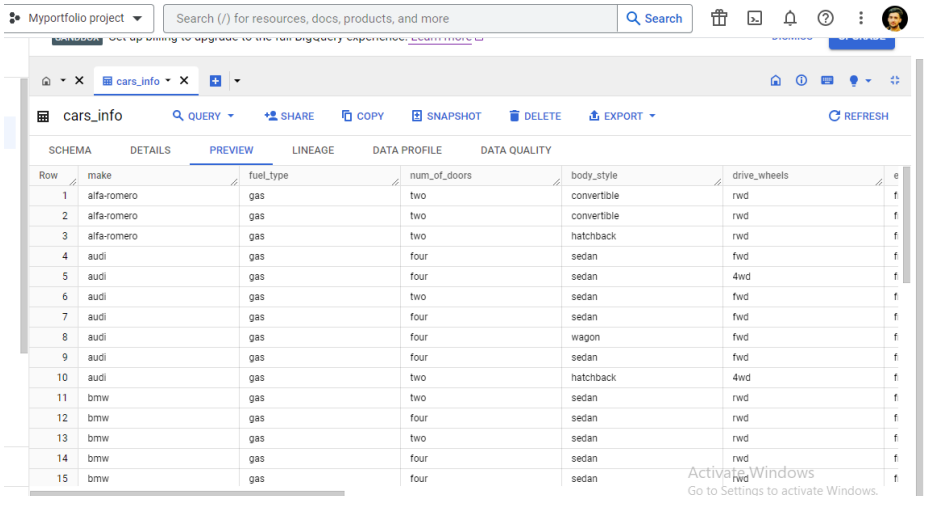
[Automobile Data]

Data Cleaning Project

- **Inspecting the fuel _type column**

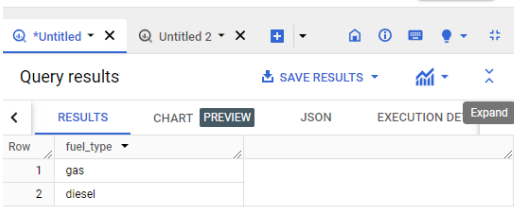
  Utilizing BigQuery to streamline data cleaning procedures, we have the following dataset to work with.



Row	make	fuel_type	num_of_doors	body_style	drive_wheels
1	alfa-romero	gas	two	convertible	rwd
2	alfa-romero	gas	two	convertible	rwd
3	alfa-romero	gas	two	hatchback	rwd
4	audi	gas	four	sedan	fwd
5	audi	gas	four	sedan	4wd
6	audi	gas	two	sedan	fwd
7	audi	gas	four	sedan	fwd
8	audi	gas	four	wagon	fwd
9	audi	gas	four	sedan	fwd
10	audi	gas	two	hatchback	4wd
11	bmw	gas	two	sedan	rwd
12	bmw	gas	four	sedan	rwd
13	bmw	gas	two	sedan	rwd
14	bmw	gas	four	sedan	rwd
15	bmw	gas	four	sedan	rwd

According to Data_description the *fuel_type* column should only have two unique string values: *diesel and gas*. R**unning the following query**.

```
SELECT Distinct  fuel_type FROM `myportfolio-proje.Cars.cars_info` LIMIT 1000
```



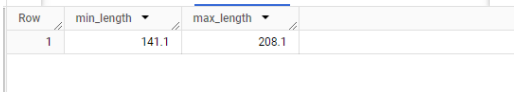
This confirms that the fuel_type column doesn't have any unexpected values.

- **Inspect the length column**

Inspecting a column with numerical data. The length column should contain numeric measurements of the cars. checking that the minimum and maximum lengths in the dataset align with the data description, which states that the lengths in this column should range from 141.1 to 208.1. **Running this query to confirm**

```
SELECT
MIN(length) AS min_length,
 MAX(length) AS max_length
FROM
 myportfolio-proje.Cars.cars_info;
```

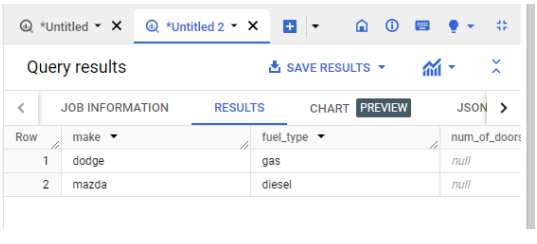
Results should confirm that 141.1 and 208.1 are the minimum and maximum values respectively in this column.



- **Fill in missing data**

Missing values can create errors or skew your results during analysis. Checking Data for null values or missing values.

```
select * from myportfolio-proje.Cars.cars_info  where num_of_doors is Null;
```



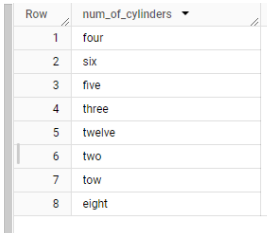
In order to fill in these missing values, checking with the sales manager, who states that all Dodge gas sedans and all Mazda diesel sedans sold had four doors.

```
update myportfolio-proje.Cars.cars_info
set num_of_doors = "four"
where  make = "dodge"
 AND fuel_type = "gas"
 AND body_style = "sedan";
```

Repeat this process to replace the null value for the Mazda

- ## Identify potential errors

Running this query to check the num_of_cylinders column**:**



Row	num_of_cylinders
1	four
2	six
3	five
4	three
5	twelve
6	two
7	tow
8	eight

There are two entries for two cylinders: rows 6 and 7. But the *two* in row 7 is misspelled. To **correct the misspelling for all rows**

```
UPDATE
 myportfolio-proje.Cars.cars_info
SET
 num_of_cylinders = "two"
WHERE
 num_of_cylinders = "tow";
```

checking the compression_ratio column. According to the data description, **the compression_ratio column values should range from 7 to 23.** Just like when I checked the length values , I can **use MIN and MAX to check if that's correct:**

```
SELECT
 MIN(compression_ratio) AS min_compression_ratio,
 MAX(compression_ratio) AS max_compression_ratio
FROM
  myportfolio-proje.Cars.cars_info;
```

this returns a maximum of 70.

this is an error because the maximum value in this column should be 23, not 70. So the 70 is most likely a 7.0

Running the above query again without the row with 70 to make sure that the rest of the values fall within the expected range of 7 to 23.

```
SELECT
```

```
 MIN(compression_ratio) AS min_compression_ratio,
 MAX(compression_ratio) AS max_compression_ratio
FROM
 myportfolio-proje.Cars.cars_info
WHERE
    compression_ratio <> 70;
```

Now the highest value is 23, which aligns with the data description. checking with the sales manager again, who says that this row was made in error and should be removed. Using the query below to count how many rows you would be deleting:

```
SELECT
COUNT(*) AS num_of_rows_to_delete
FROM
 myportfolio-proje.Cars.cars_info
WHERE
 compression_ratio = 70;
```

Turns out there is only one row with the erroneous 70 value. D**eleting that row using this query:**

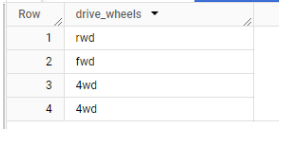
```
DELETE myportfolio-proje.Cars.cars_info
WHERE compression_ratio = 70;
```

- ## Ensure consistency

Check the drive_wheels column for inconsistencies by **running a query with a SELECT DISTINCT statement:**

```
 SELECT
  DISTINCT drive_wheels
FROM
  myportfolio-proje.Cars.cars_info;
```

It appears that 4wd appears twice in results., However, because we used a SELECT DISTINCT statement to return unique values this probably means there's an extra space in one of the 4wd entries that makes it different from the other 4wd.



Row	drive_wheels
1	rwd
2	fwd
3	4wd
4	4wd

To check if this is the case, I **used a LENGTH statement** to determine the length of how long each of these string variables:

```
SELECT
 DISTINCT drive_wheels,
```

```
  LENGTH(drive_wheels) AS string_length
FROM
  myportfolio-proje.Cars.cars_info;
```

According to these results, some instances of the 4wd string have four characters instead of the expected three (4wd has 3 characters). In that case, I **used the TRIM function to remove all extra spaces in the drive_wheels column :**

```
UPDATE
myportfolio-proje.Cars.cars_info
SET
 drive_wheels = TRIM(drive_wheels)
WHERE TRUE;
```

Then, I **run the SELECT DISTINCT statement again** to ensure that there are only three distinct values in the drive_wheels column:

```
SELECT
  DISTINCT drive_wheels
FROM
  myportfolio-proje.Cars.cars_info;
```

And now there should only be three unique values in this column! Which means my data is clean, consistent, and ready for analysis!