**Background**

The *Home\_Sales\_starter\_code\_colab.ipynb* file was provided for this challenge. I have coded in

Working with Big Data in this challenge, SparkSQL was used to determine key metrics about home sales data.

Then, I used Spark to create temporary views, to partition the data and to cache + uncache a temporary table.

The final step was to verify that the table had indeed been uncached.

**Instructions**

1. Renamed the *Home\_Sales\_starter\_code\_colab.ipynb* file which was provided as *Home\_Sales.ipynb*.
2. Imported the necessary PySpark SQL functions for this assignment.
3. Read the *home\_sales\_revised.csv* data in the starter code into a Spark DataFrame.
4. Created a temporary table called *home\_sales*.
5. Answered the following questions using SparkSQL:

* What is the average price for a four-bedroom house sold for each year? The answer was rounded off to two decimal places.

A screen shot of a computer screen

Description automatically generated

* What is the average price of a home for each year it was built that has three bedrooms and three bathrooms? The answer was rounded off to two decimal places.

A black screen with white text

Description automatically generated

* What is the average price of a home for each year built that has three bedrooms, three bathrooms, two floors, and is greater than or equal to 2,000 square feet? Rounded off the answer to two decimal places.

A screenshot of a computer screen

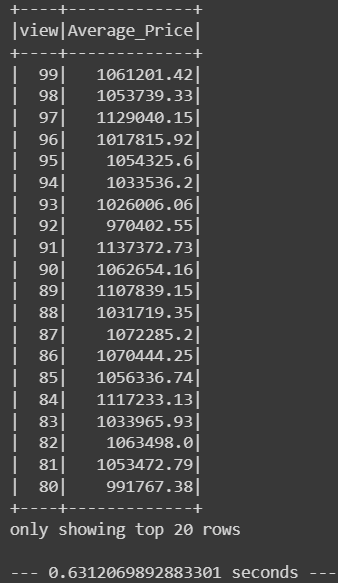
Description automatically generated

* What is the "view" rating for homes costing more than or equal to $350,000? The answer was rounded off to two decimal places and the runtime was determined.

A screenshot of a computer

Description automatically generated

1. Cached the temporary table *home\_sales\_data*.
2. Checked if the temporary table was cached.
3. Using the cached data, I ran a query that filtered out the view ratings with an average price of greater than or equal to $350,000 and determined the runtime. This runtime result was then compared to the uncached runtime.



The runtime for the cached data above was quicker at 0.6312069892883301 seconds while the uncached runtime was 0.8923213481903076 seconds.

1. Partitioned by the "*date\_built*" field on the formatted parquet home sales data.
2. Created a temporary table for the parquet data.
3. Ran the query that filtered out the view ratings with an average price of greater than or equal to $350,000 and determined the runtime and compared it to uncached runtime.

A screenshot of a computer

Description automatically generated

The runtime for the above analysis is 1.1581673622131348 seconds while the runtime for the cached version was faster at 0.6312069892883301 seconds.

1. Uncached the home\_sales temporary table.
2. Verified that the home\_sales temporary table is uncached using PySpark.
3. Download your *Home\_Sales.ipynb* file has been uploaded into this GitHub repository.