Cognizant Technology Solutions

A Project Report on

"FORMULA 1 PROJECT USING PYSPARK"

Prepared By

Ankita Shivane	2320390
Harshit Yadav	2320382
Mimansa Yadav	2320956
Nandhini T	2320539
Rajanya Kamilya	2320828
Shambhavi Shukla	2320529

INTRODUCTION

FORMULA RACING

Formula racing, exemplified by Formula 1 (F1), stands as the pinnacle of motorsport, blending cutting-edge technology with unparalleled athleticism and strategy. Originating in the post-war era, Formula racing has evolved into a global phenomenon, captivating audiences with its fusion of speed, precision engineering, and human skill.

At its heart, Formula racing features a series of high-profile races held on circuits around the world, with the Formula 1 World Championship being the most prestigious. This championship comprises a calendar of races held across diverse tracks, from iconic street circuits like Monaco to purpose-built facilities like Silverstone and Suzuka.

Behind the scenes, Formula racing is a relentless pursuit of perfection for constructors, who design and build the cars that compete on the track. Teams like Mercedes, Ferrari, and Red Bull invest heavily in research, development, and innovation, striving to create the fastest and most reliable machines possible.

Formula racing is also a stage for remarkable achievements, with records continually being shattered and history being made. From historic race wins to championship triumphs, the sport is a tapestry of unforgettable moments that etch themselves into the annals of motorsport history.

In essence, Formula racing is a captivating blend of technology, athleticism, and strategy, where races, drivers, constructors, tracks, and achievements converge to create an electrifying spectacle that captivates audiences around the globe.

PYSPARK

PySpark is a Python API to support Python with Apache Spark. PySpark provides Py4j library, with the help of this library, Python can be easily integrated with Apache Spark. PySpark plays an essential role when it needs to work with a vast dataset or analyze them. This feature of PySpark makes it a very demanding tool among data engineers.

A large amount of data is generated offline and online. It is necessary to extract valuable information from the raw data. We require a more efficient tool to perform different types of operations on the big data. There are various tools to perform the multiple tasks on the huge dataset, but these tools are not so appealing anymore. It is needed some scalable and flexible tools to crack big data and gain benefit from it.

BUSINESS REQUIREMENTS

Description:

Analyze and process Formula 1 data files to extract insights about dominant drivers, teams and their performances.

REQUIREMENTS:

Data Ingestion Requirements:

- Ingest the data from various source files
- Ingested data must have the schema applied
- Ingested data must have audit columns
- Ingested data must be stored in a columnar format

Data Transformation Requirements:

- Join the key information required for reporting to create a new table.
- Join the key information required for Analysis to create a new table.
- Transformed tables must have audit columns
- Must be able to analyze the transformed data via SQL
- Transformed data must be stored in columnar format (i.e. Parquet)

Reporting Requirements:

- Driver Standings
- Constructor Standings

Analysis Requirements:

- Dominant Drivers
- Dominant Teams
- Visualize the Outputs

F1 DATA SET

circuits table

Field	Type	Null	Key
circuitId	int(11)	NO	PRI
circuitRef	varchar(255)	NO	
name	varchar(255)	NO	
location	varchar(255)	YES	
country	varchar(255)	YES	
lat	float	YES	
lng	float	YES	
alt	int(11)	YES	
url	varchar(255)	NO	UNI

drivers table

Field	Type	Null	Key
driverId	int(11)	NO	PRI
driverRef	varchar(255)	NO	
number	int(11)	YES	
code	varchar(3)	YES	
forename	varchar(255)	NO	
surname	varchar(255)	NO	
dob	date	YES	
nationality	varchar(255)	YES	
url	varchar(255)	NO	UNI

constructors table

Field	Type	Null	Key
constructorId	int(11)	NO	PRI
constructorRef	varchar(255)	NO	
name	varchar(255)	NO	UNI
nationality	varchar(255)	YES	
url	varchar(255)	NO	

races table

Field	Type	Null	Key
raceId	int(11)	NO	PRI
year	int(11)	NO	
round	int(11)	NO	
circuitId	int(11)	NO	
name	varchar(255)	NO	
date	date	NO	
time	time	YES	
url	varchar(255)	YES	UNI
fp1_date	date	YES	
fp1_time	time	YES	
fp2_date	date	YES	
fp2_time	time	YES	
fp3_date	date	YES	
fp3_time	time	YES	
quali_date	date	YES	
quali_time	time	YES	
sprint_date	date	YES	
sprint_time	time	YES	

lap_times table

Field	Type	Null	Key
raceId	int(11)	NO	FK
driverId	int(11)	NO	FK
lap	int(11)	NO	PRI
position	int(11)	YES	
time	varchar(255)	YES	
milliseconds	int(11)	YES	

pit_stops table

Field	Type	Null	Key
raceId	int(11)	NO	FK
driverId	int(11)	NO	FK
stop	int(11)	NO	PRI
lap	int(11)	NO	
time	time	NO	
duration	varchar(255)	YES	
milliseconds	int(11)	YES	

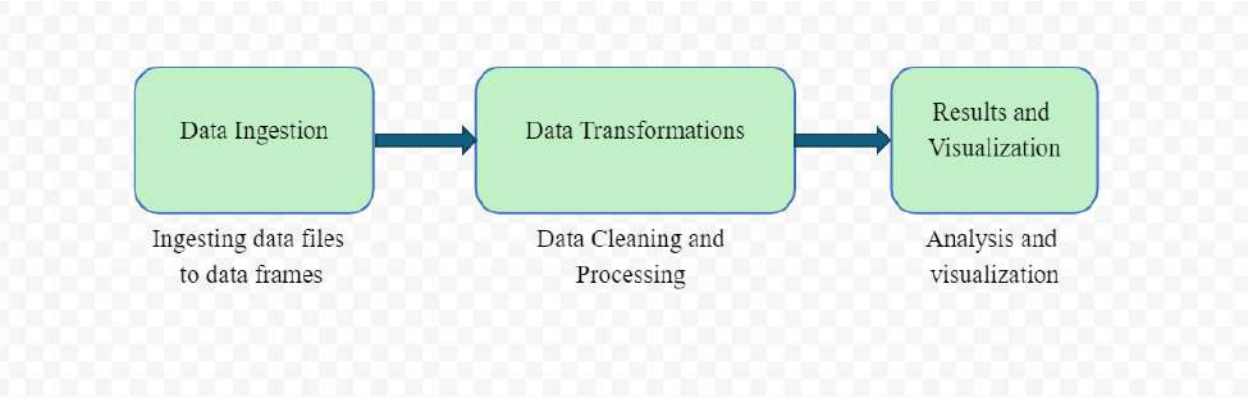
qualifying table

Field	Type	Null	Key
qualifyId	int(11)	NO	PRI
raceId	int(11)	NO	
driverId	int(11)	NO	
constructorId	int(11)	NO	
number	int(11)	NO	
position	int(11)	YES	
q1	varchar(255)	YES	
q2	varchar(255)	YES	
q3	varchar(255)	YES	

sprint_results table

Field	Type	Null	Key
sprintResultId	int(11)	NO	PRI
raceId	int(11)	NO	
driverId	int(11)	NO	
constructorId	int(11)	NO	
number	int(11)	YES	
grid	int(11)	NO	
position	int(11)	YES	
positionText	varchar(255)	NO	
positionOrder	int(11)	NO	
points	float	NO	
laps	int(11)	NO	
time	varchar(255)	YES	
milliseconds	int(11)	YES	
fastestLap	int(11)	YES	
fastestLapTime	varchar(255)	YES	
statusId	int(11)	NO	

PICTORIAL FLOWCHART:



IMPLEMENTATION:

- **REQUIREMENT 1:**

DATA INGESTION

Data ingestion is the process of importing and loading data into a system. It's a critical step in any data-centric workflow, ensuring that the correct information is available at the right time for analysis and decision-making.

To ingest a dataset into Databricks, you can follow these steps:

1. Create a Compute engine:

- Start by creating a Databricks cluster.
- This cluster will provide the compute resources needed to run your commands.

2. Enable DBFS in Databricks:

- Use Databricks features to explore your raw dataset.

3. Ingest the Raw Data:

- Load the raw data into a table to make it available for further processing.

Here, in this project we have 8 datafiles that we need to ingest into DBFS system.

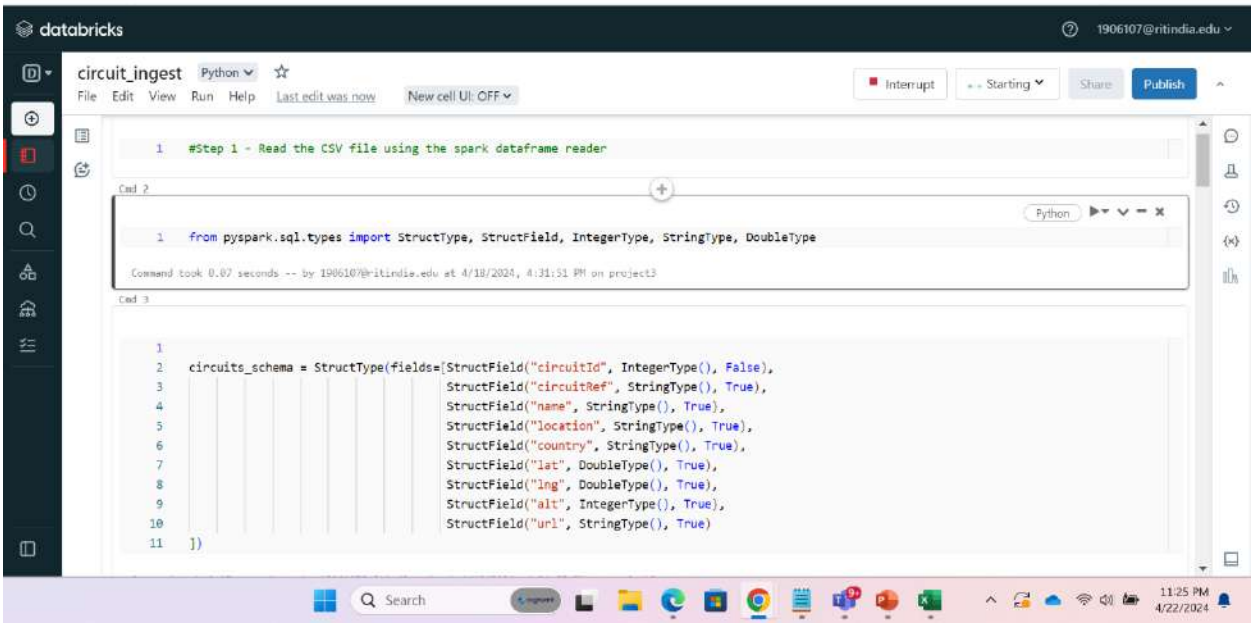
1. Circuit file ingestion:

Step 1: Open Databricks community and login to your account.

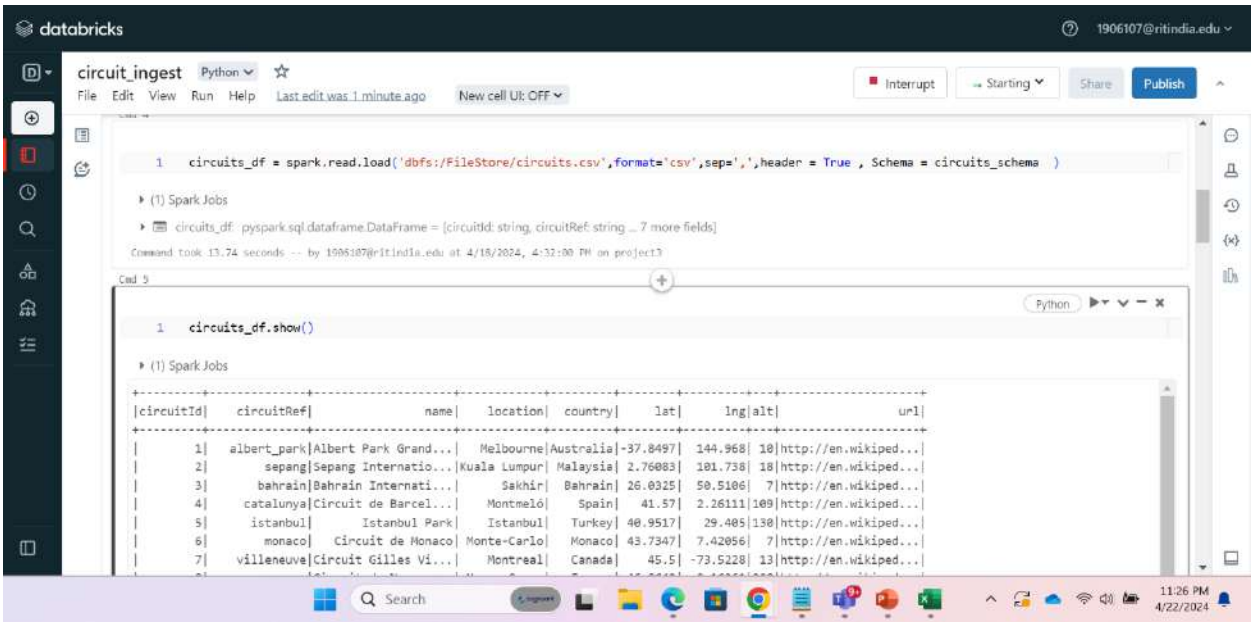
Create a compute engine for analysis.

Create new notebook in workspace and write the above syntax

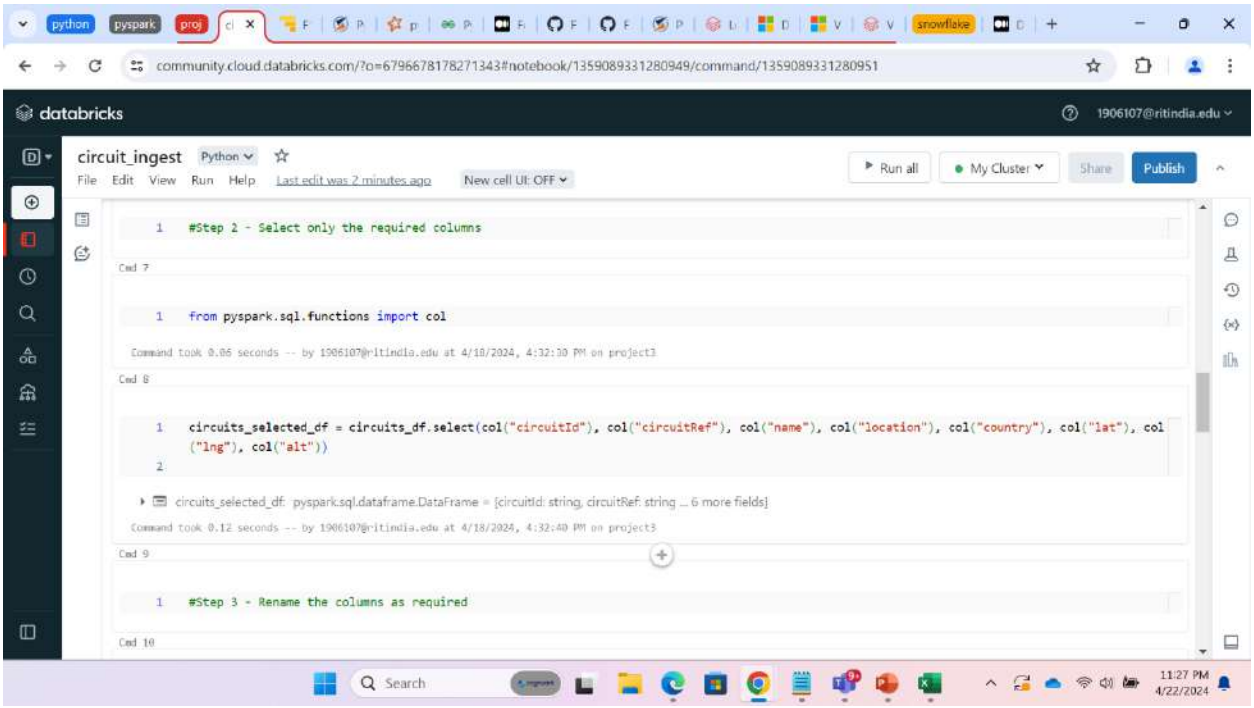
Step 2: Create the schema for Circuit file using StructType



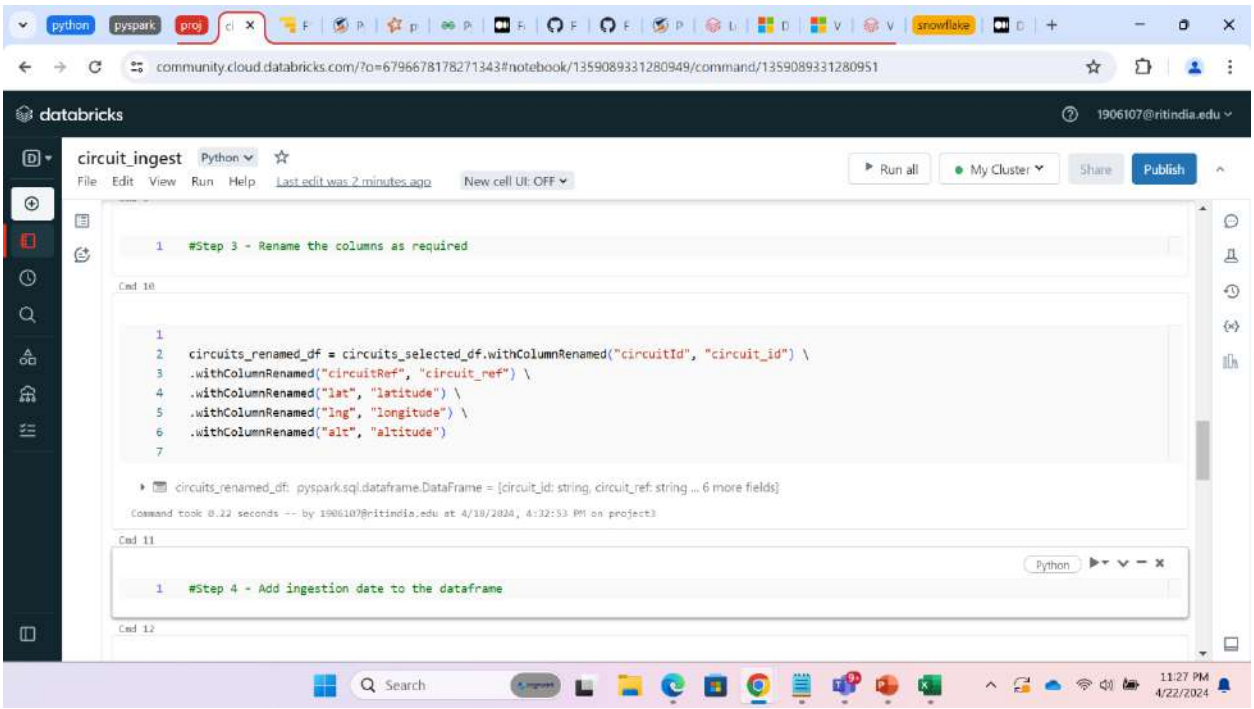
Step 3: We have created a dataframe by reading the data from circuit csv file loaded in DBFS system



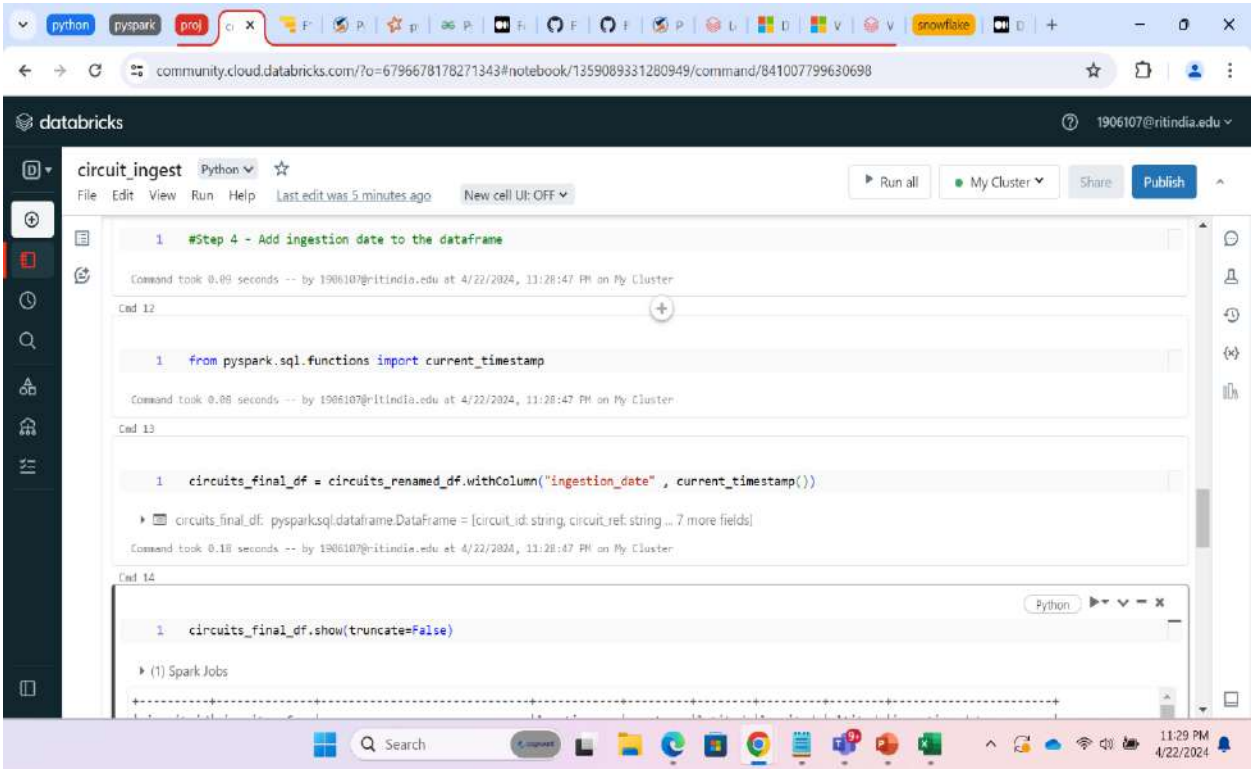
Step 4: Selecting only the required columns and creating a new dataframe.

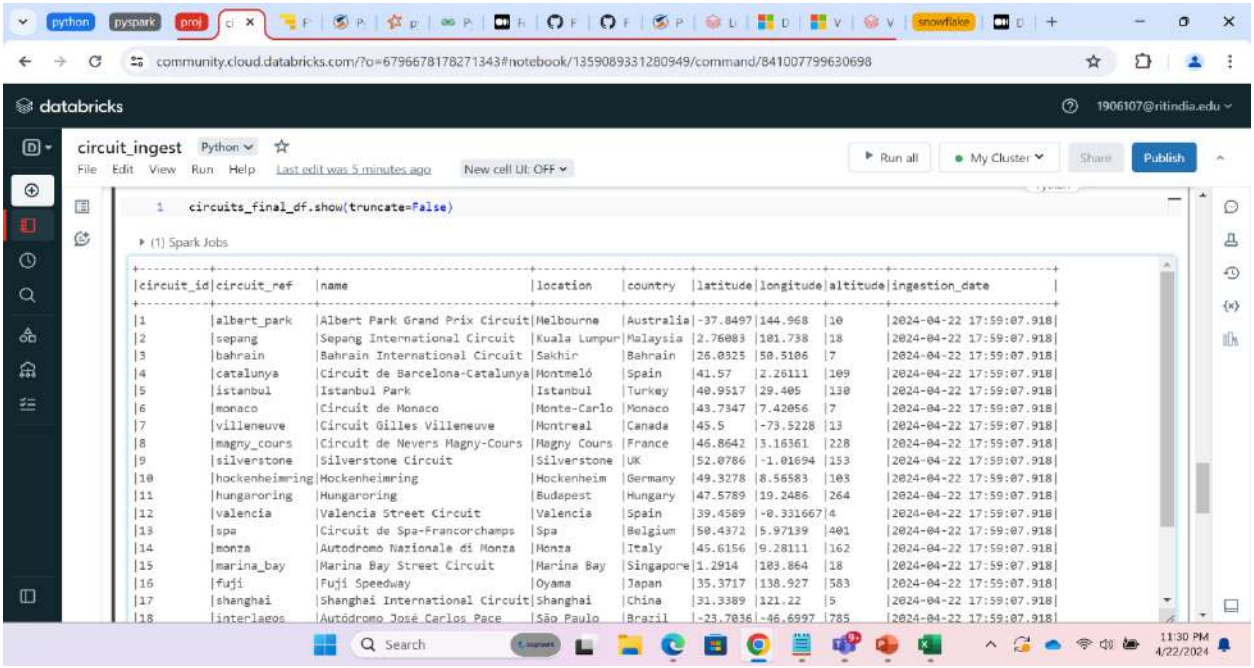


Step 5: Renaming the columns using withColumnRenamed command to match the naming standards.

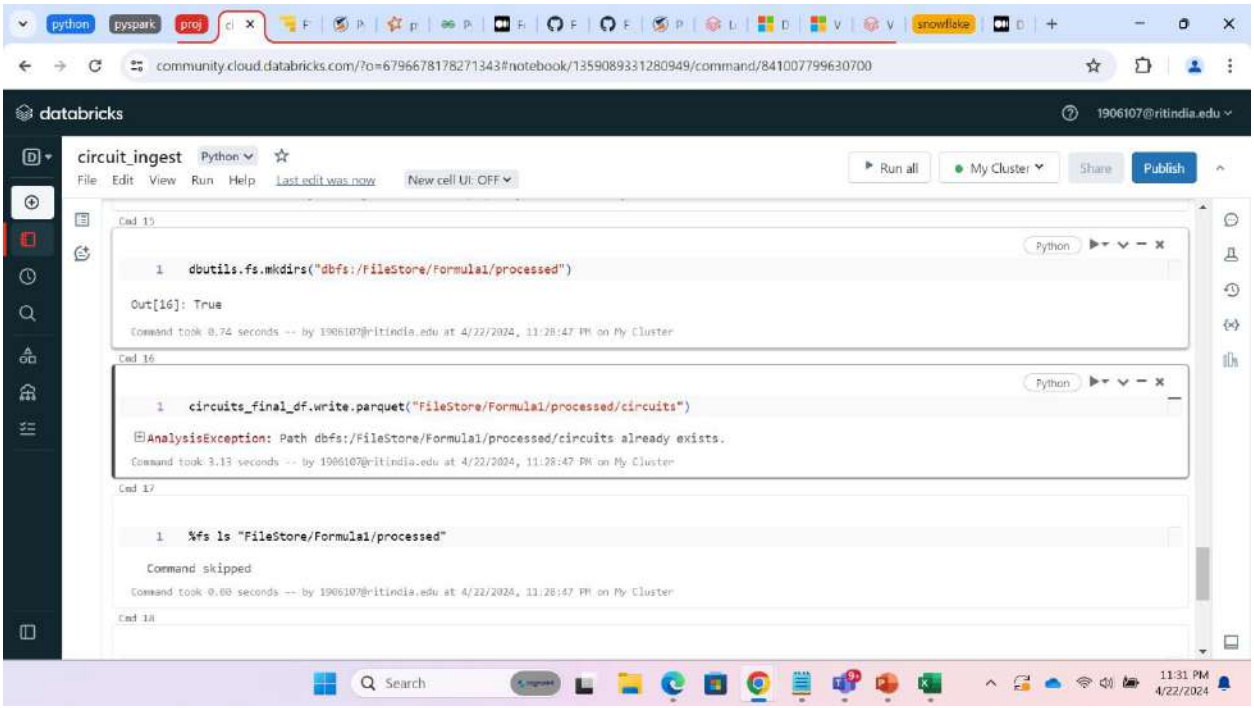


Step 6: Adding the ingestion_date column with current_tiemstamp and displaying the dataframe



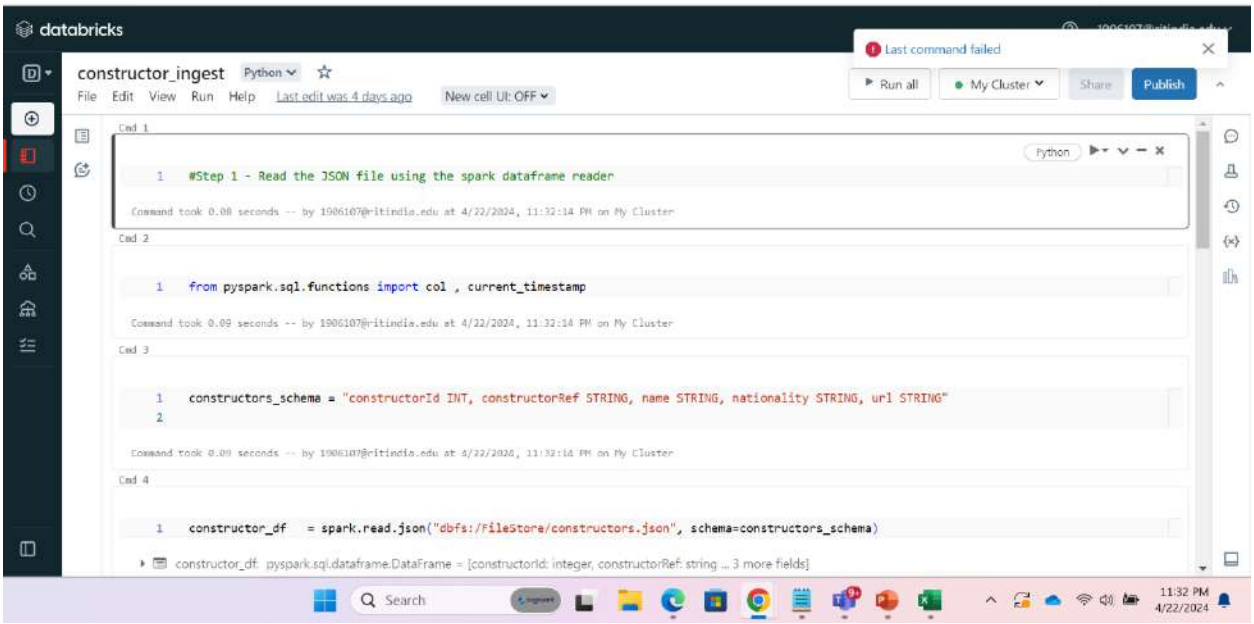


Step 7: Creating new directory in databricks filestore and storing our dataframe in columnar format that is parquet file.

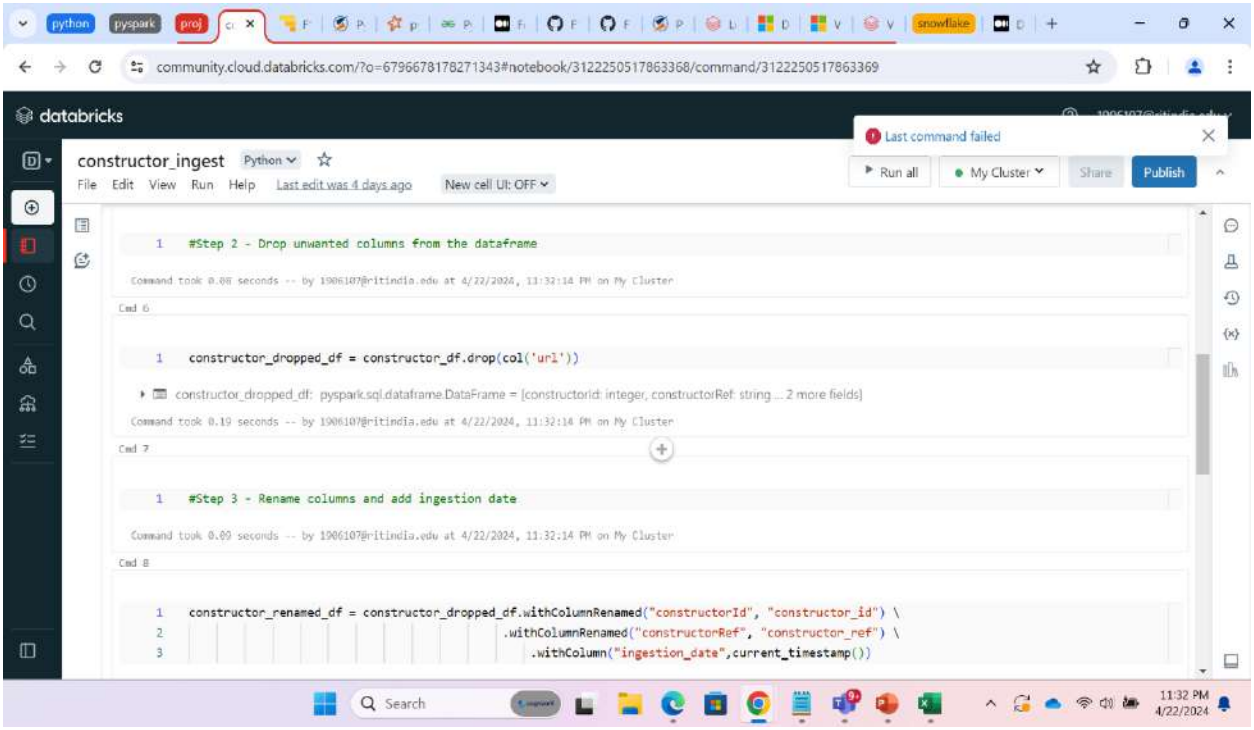


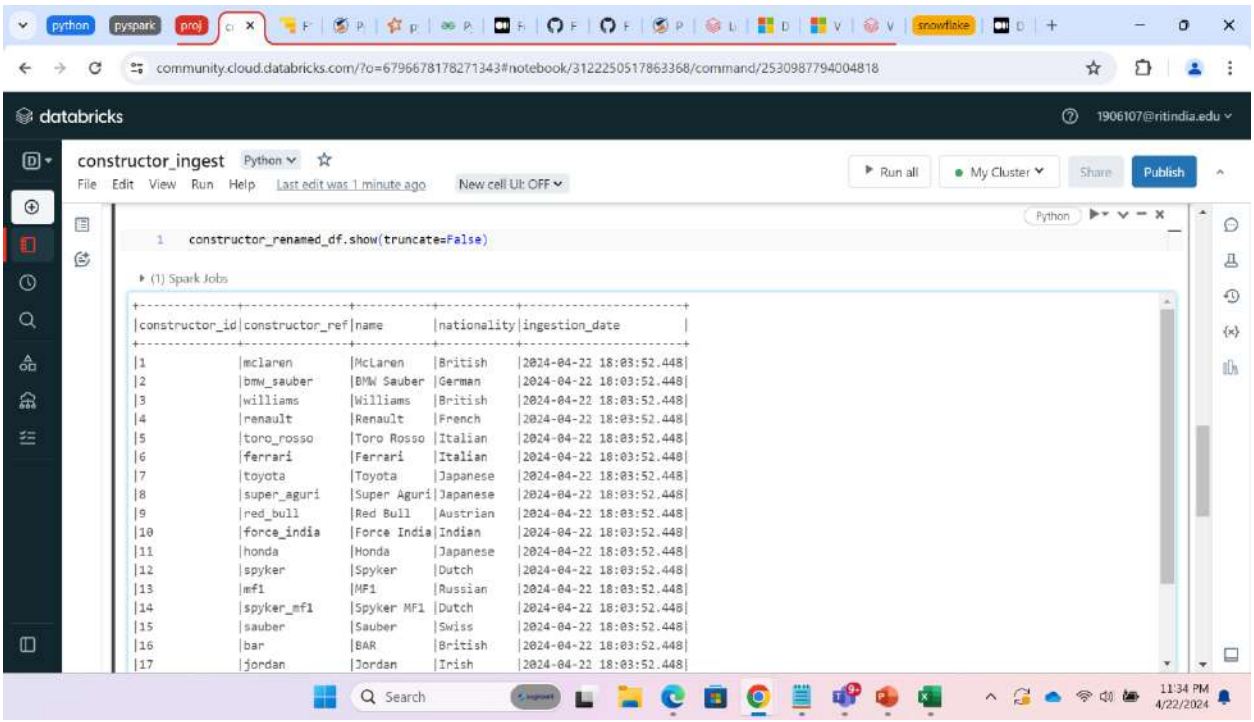
2. Constructor file ingestion:

Step 1: Create the schema for Constructor file using String format and then read the file from constructor JSON file.

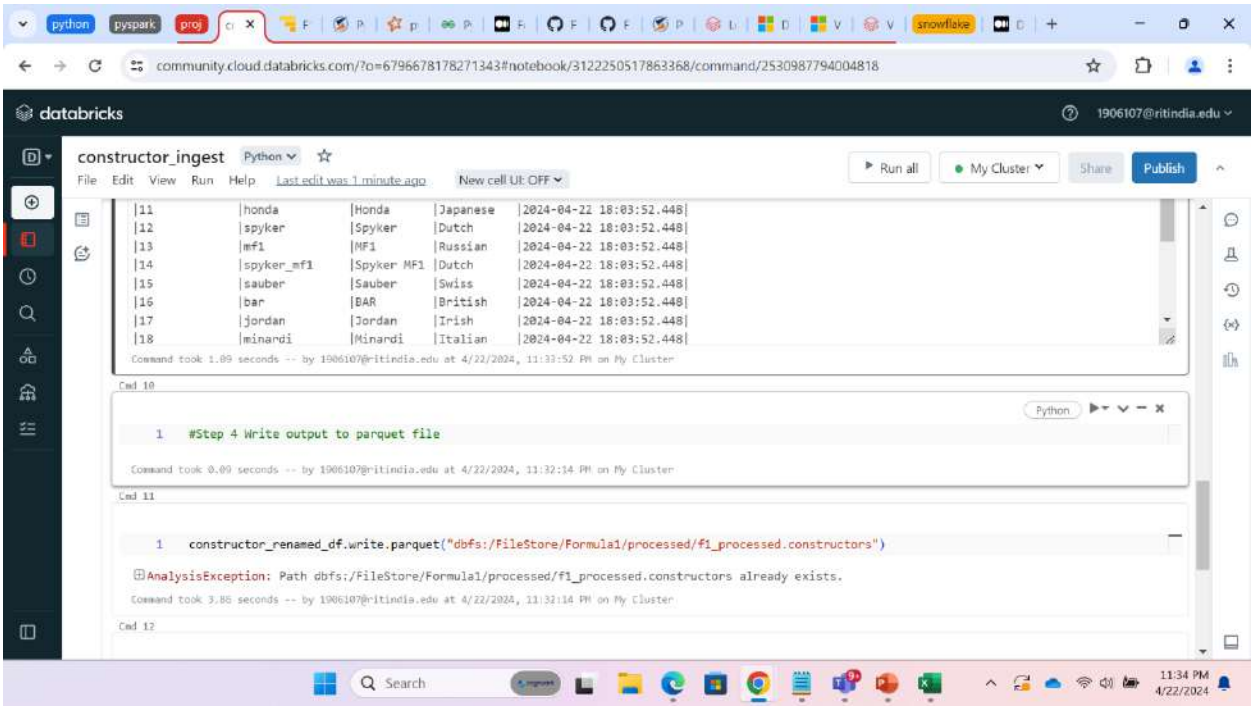


Step2: We are dropping the url column by drop command.



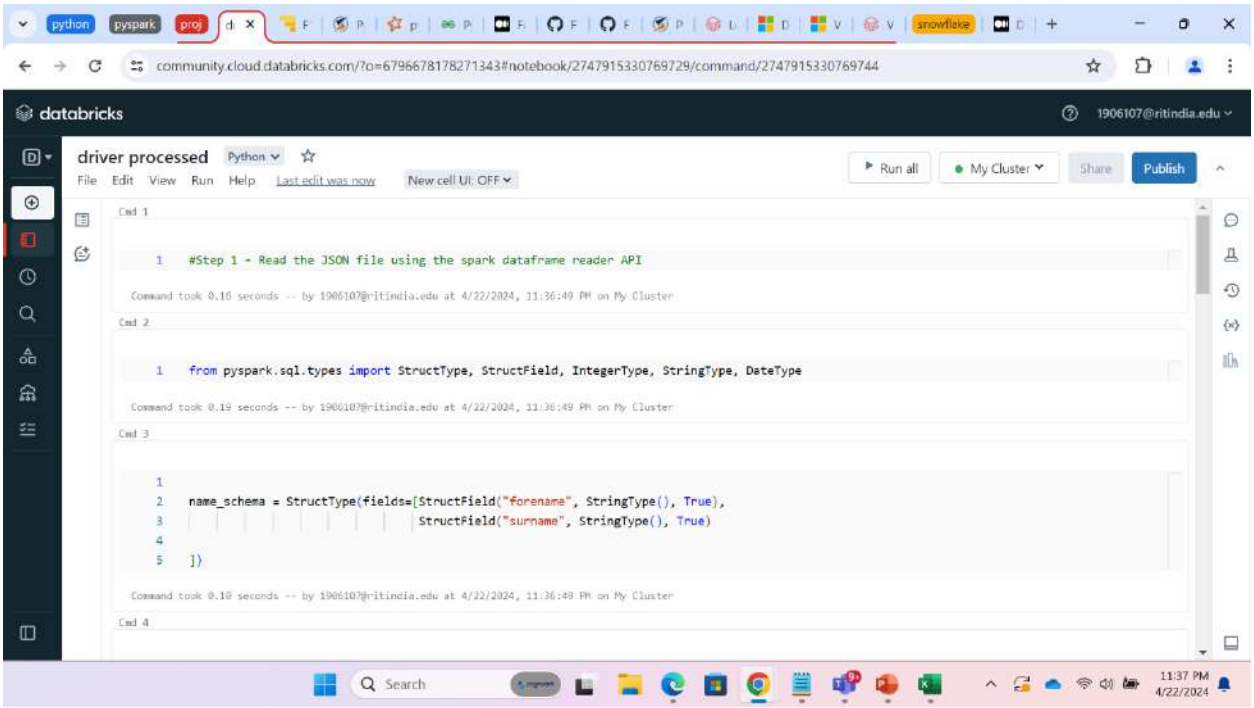


Step3: Storing our dataframe in columnar format that is parquet file.

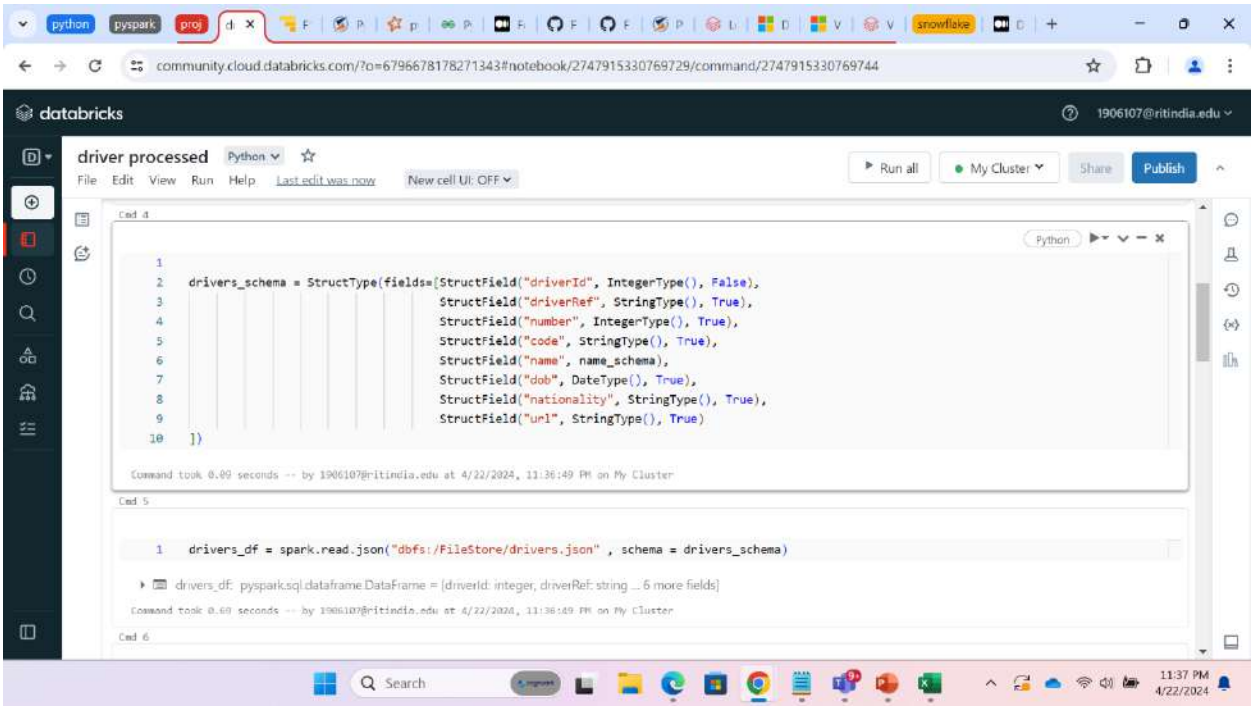


3. Driver file ingestion:

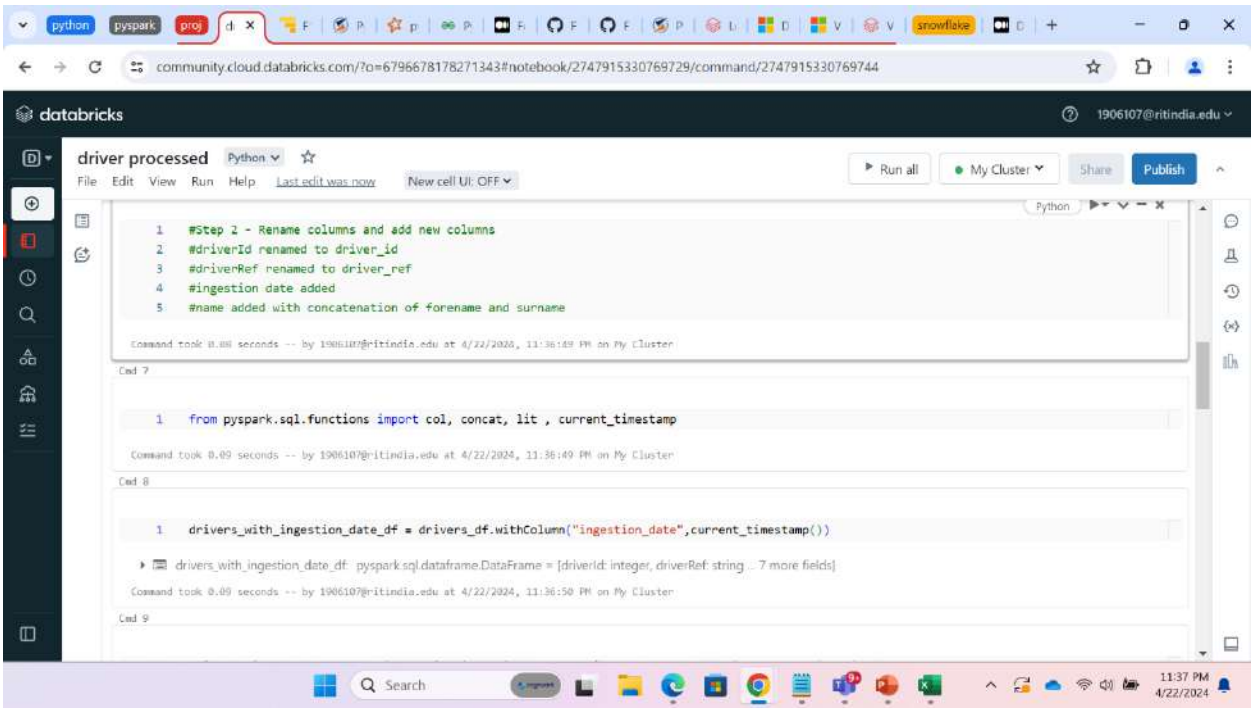
Step 1: Create the schema for the drivers file using StrucType

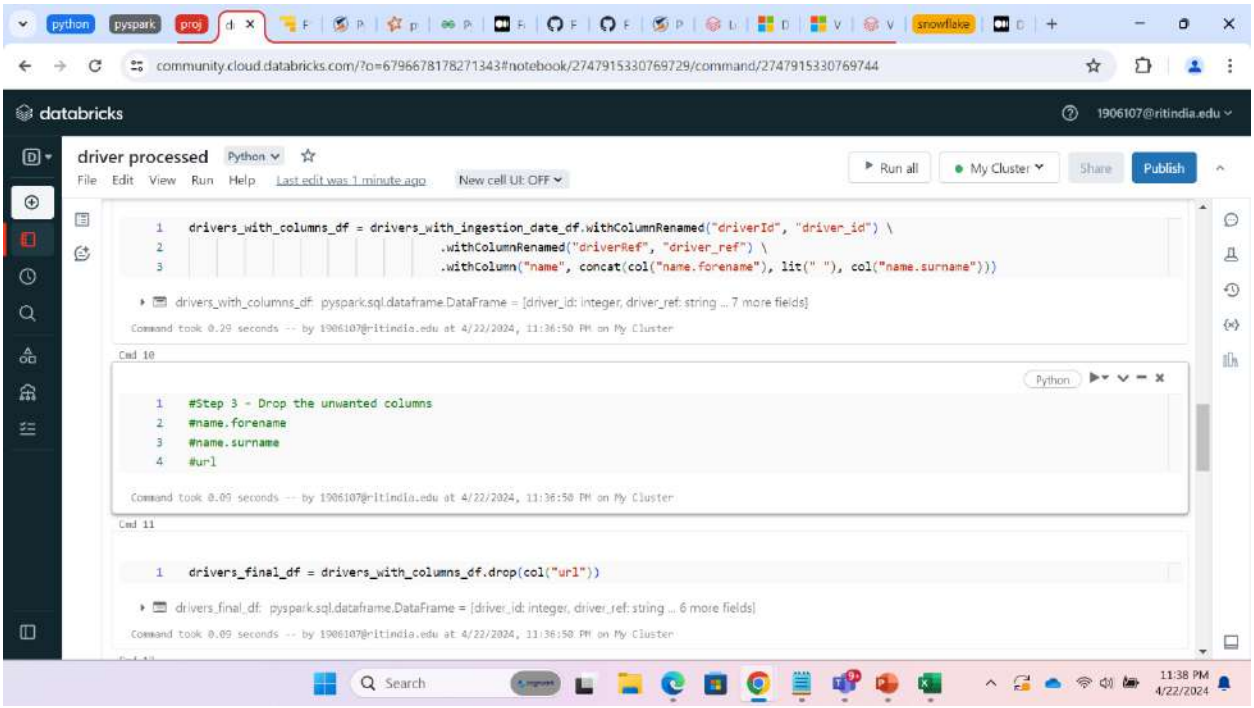


Step 3: We have created a dataframe by reading the data from json file loaded in DBFS system

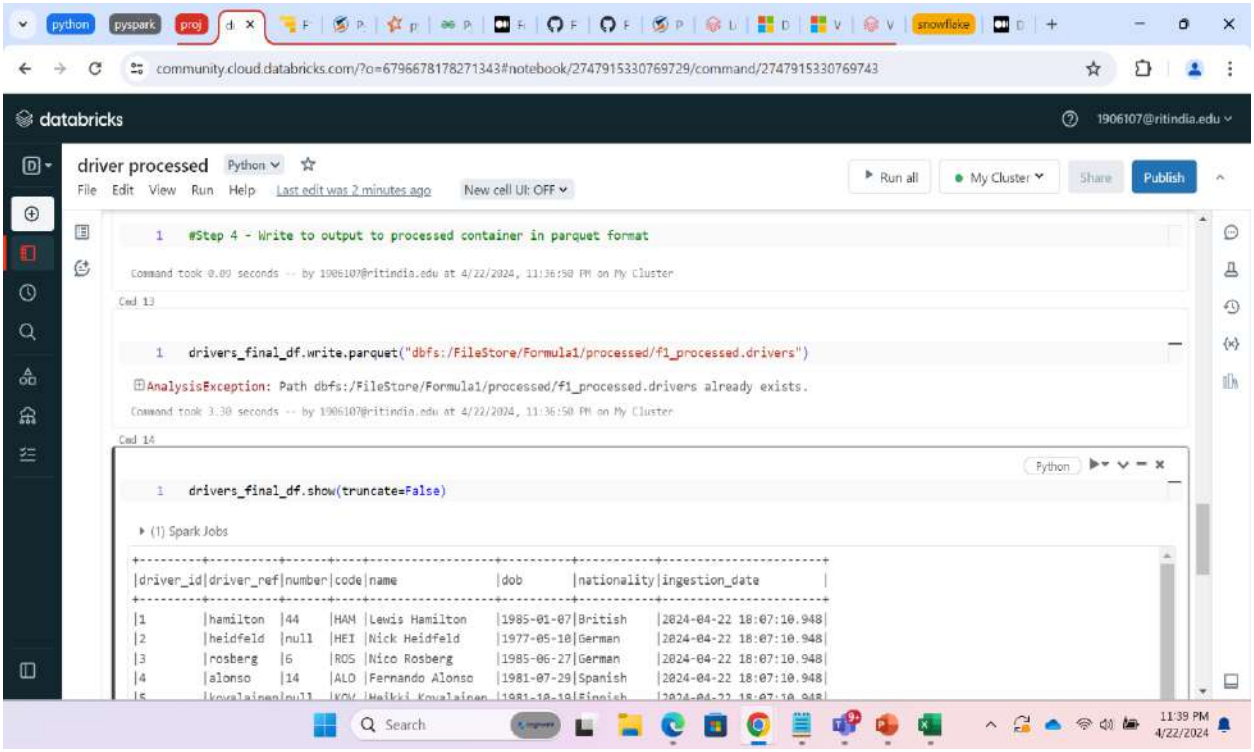


Step 4: Renaming the columns using withColumnRenamed command to match the naming standards.



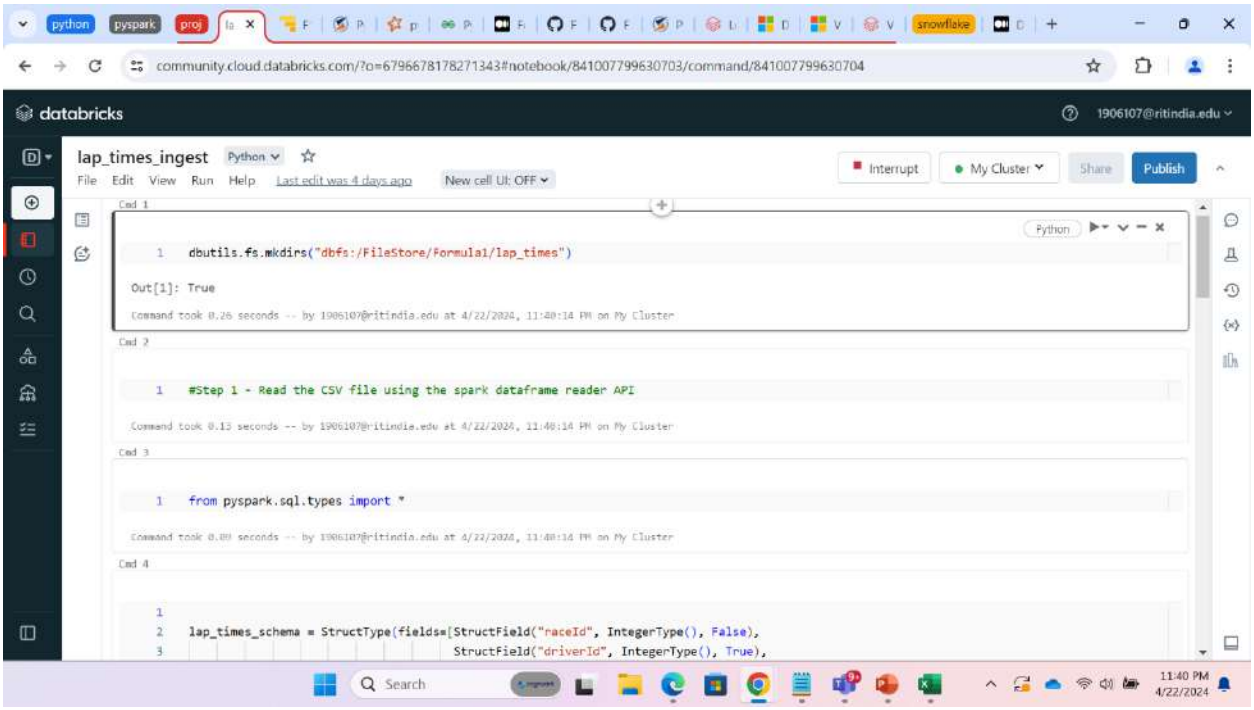


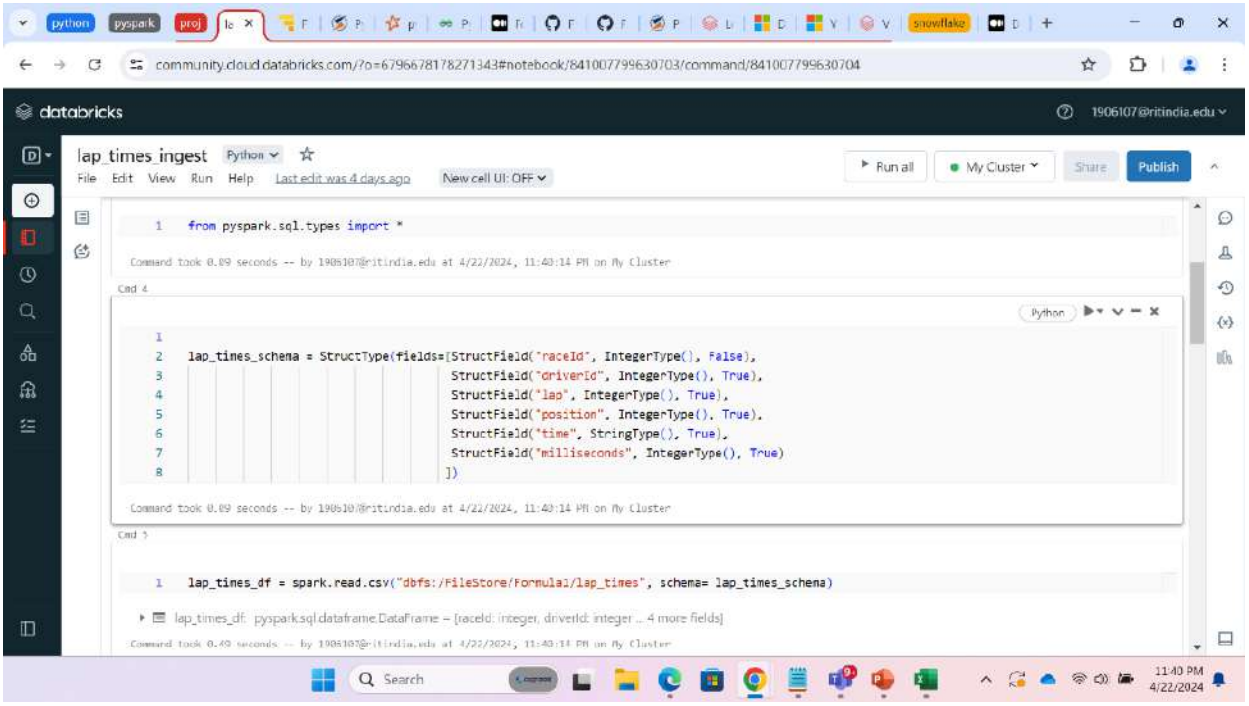
Step 5: Storing our dataframe in columnar format that is parquet file.

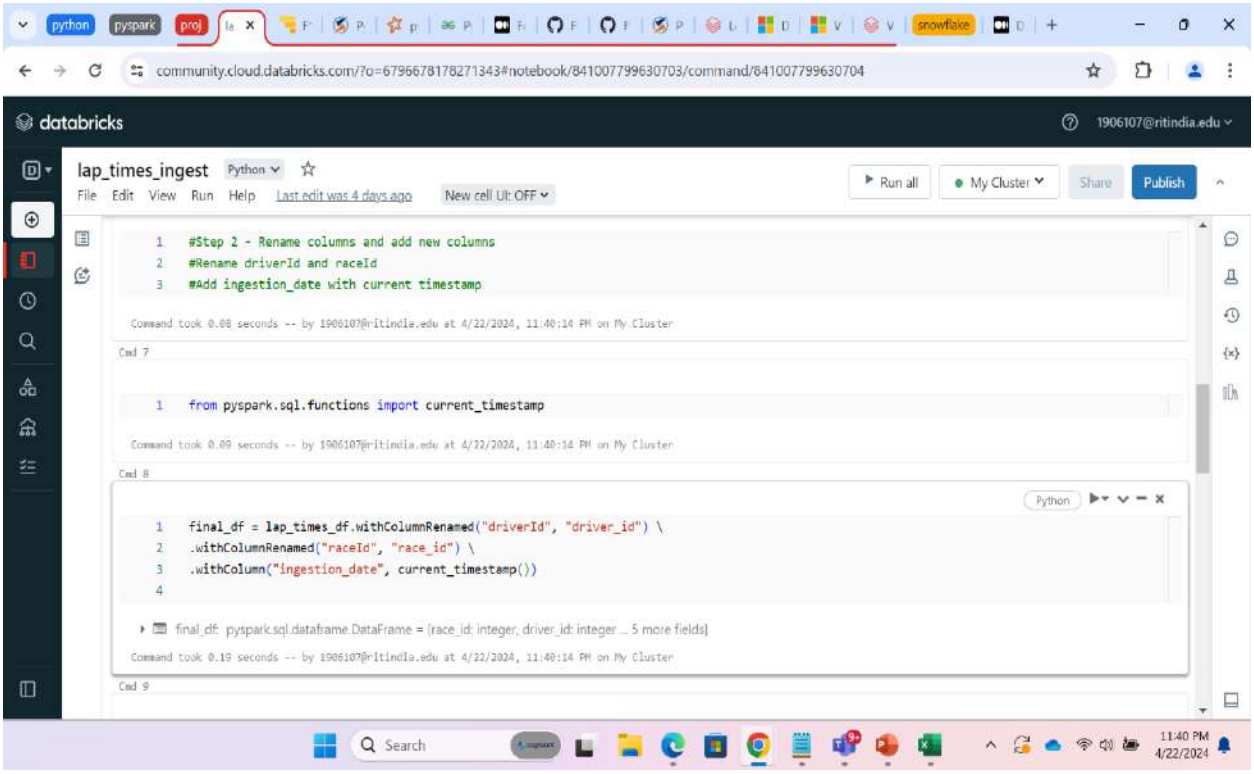


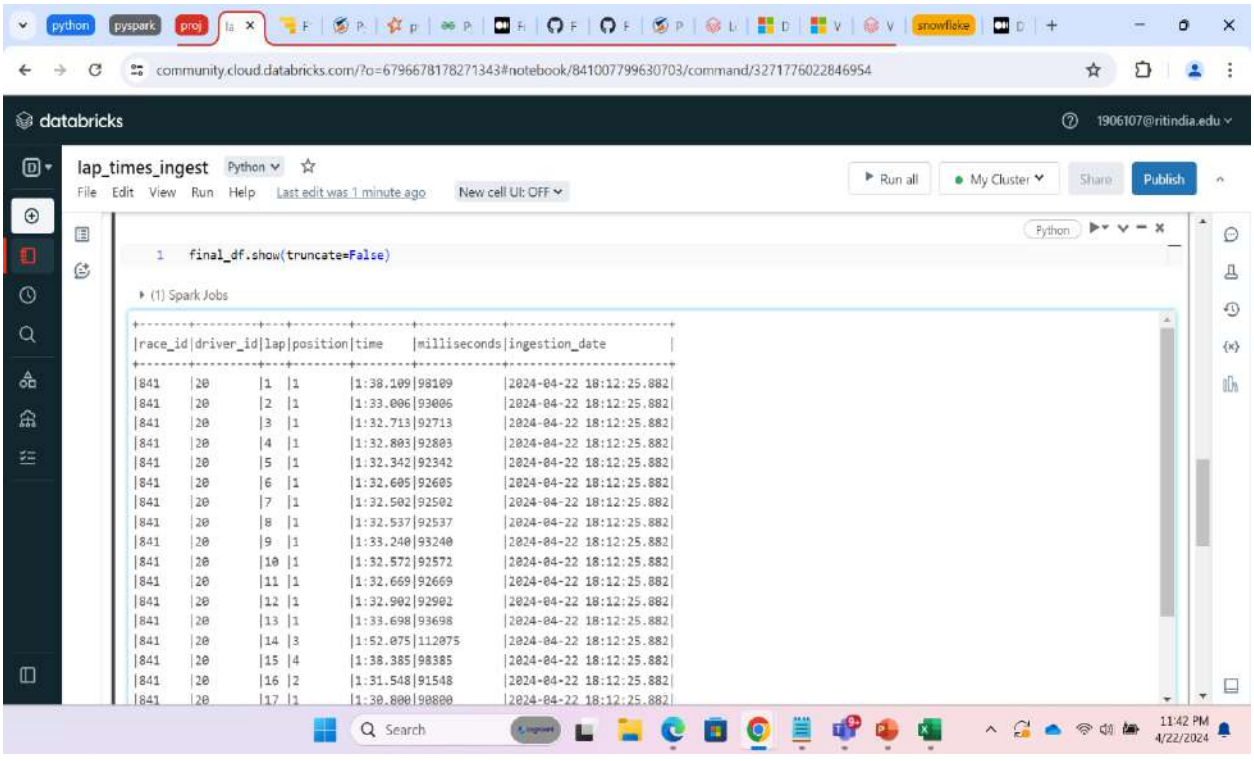
Similarly, we have performed ingestion for all the other files into the DBFS system.

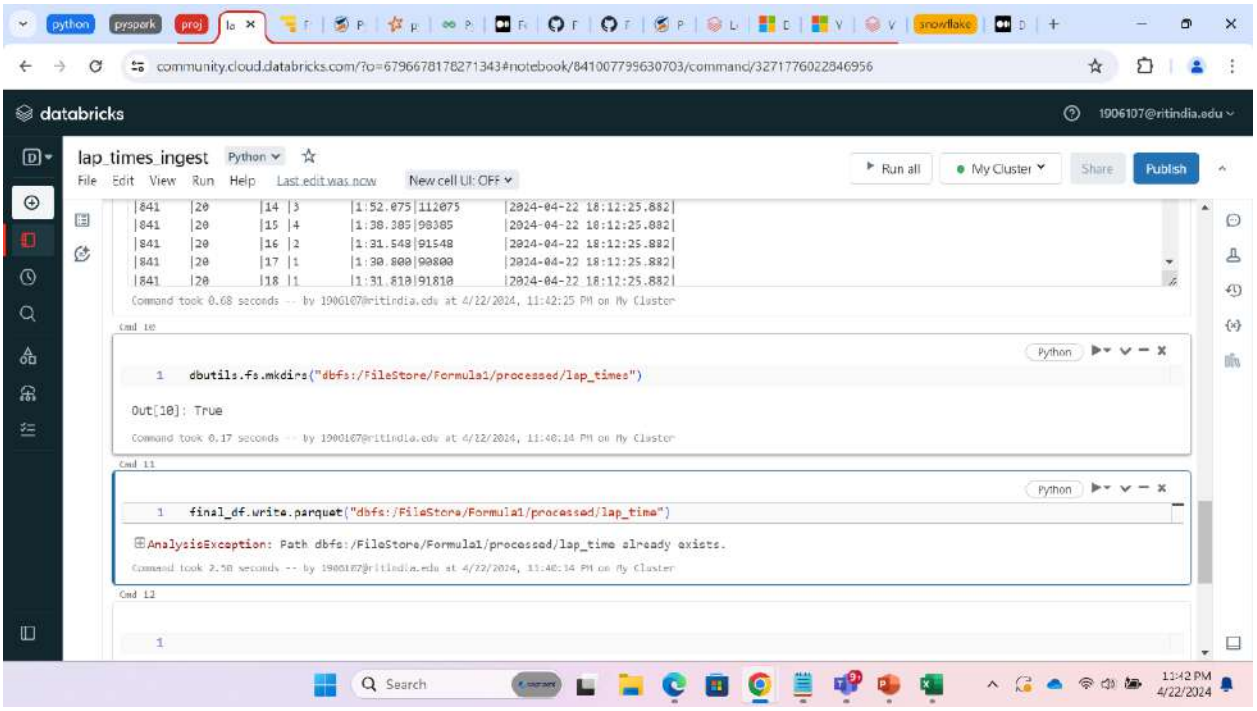
4. LAPTIME INGEST:



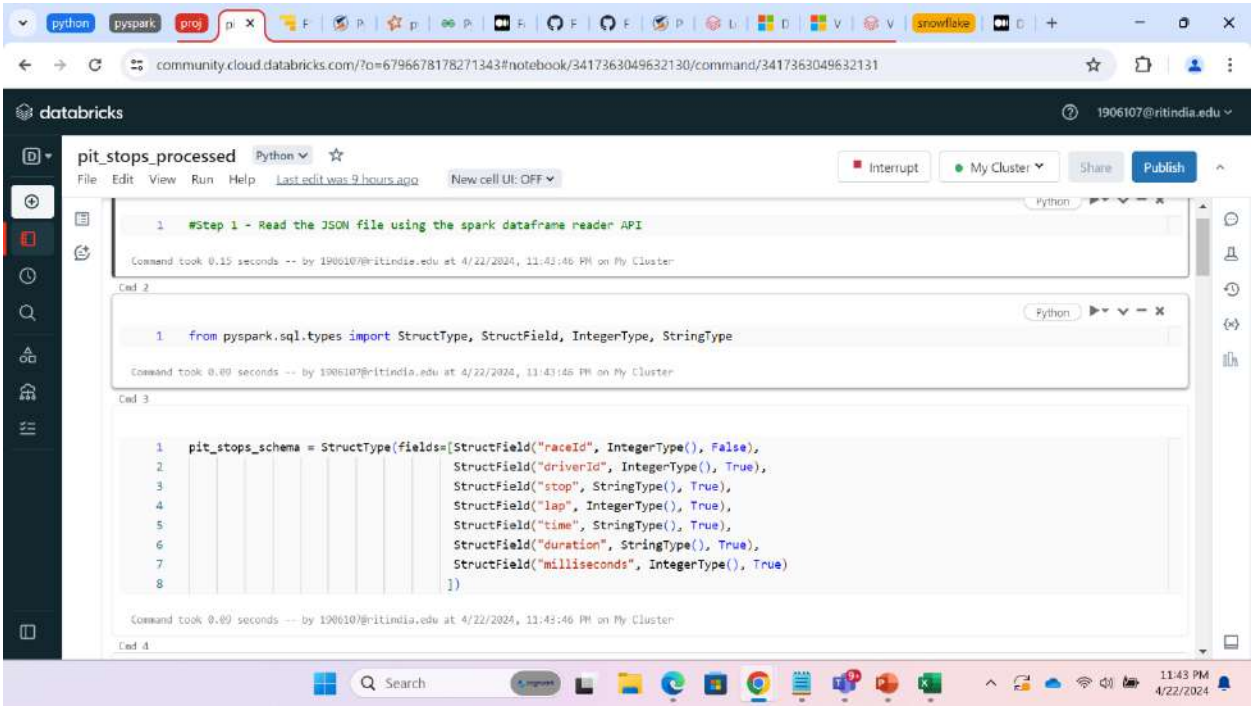


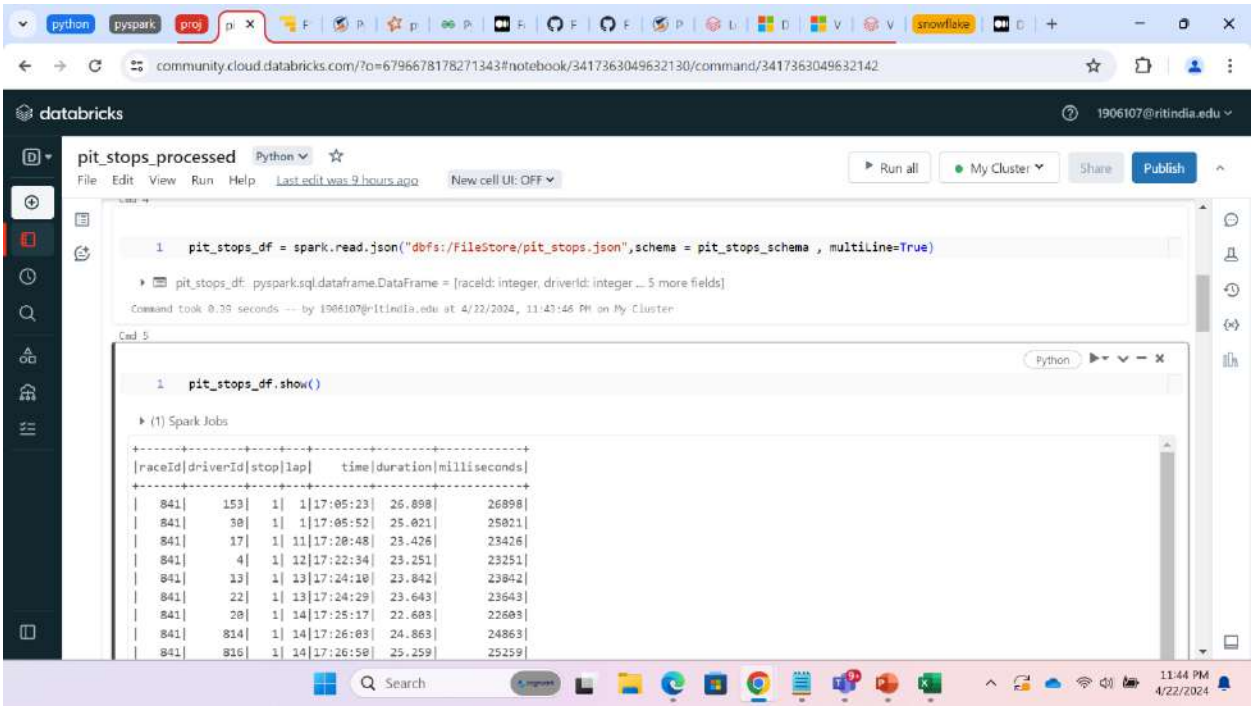

community.cloud.databricks.com/?o=6796678178271343#notebook/841007799630703/command/841007799630704
databricks
1906107@ritindia.edu
lap_times_ingest
Python
File Edit View Run Help
Last edit was 4 days ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
#Step 2 - Rename columns and add new columns
#Rename driverId and raceId
#Add ingestion_date with current timestamp
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:40:14 PM on My Cluster
Cmd 7
from pyspark.sql.functions import current_timestamp
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:40:14 PM on My Cluster
Cmd 8
final_df = lap_times_df.withColumnRenamed("driverId", "driver_id") \
.withColumnRenamed("raceId", "race_id") \
.withColumn("ingestion_date", current_timestamp())
final_df: pyspark.sql.dataframe.DataFrame = [race_id: integer, driver_id: integer ... 5 more fields]
Command took 0.19 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:40:14 PM on My Cluster
Cmd 9
Search
11:40 PM
4/22/2024

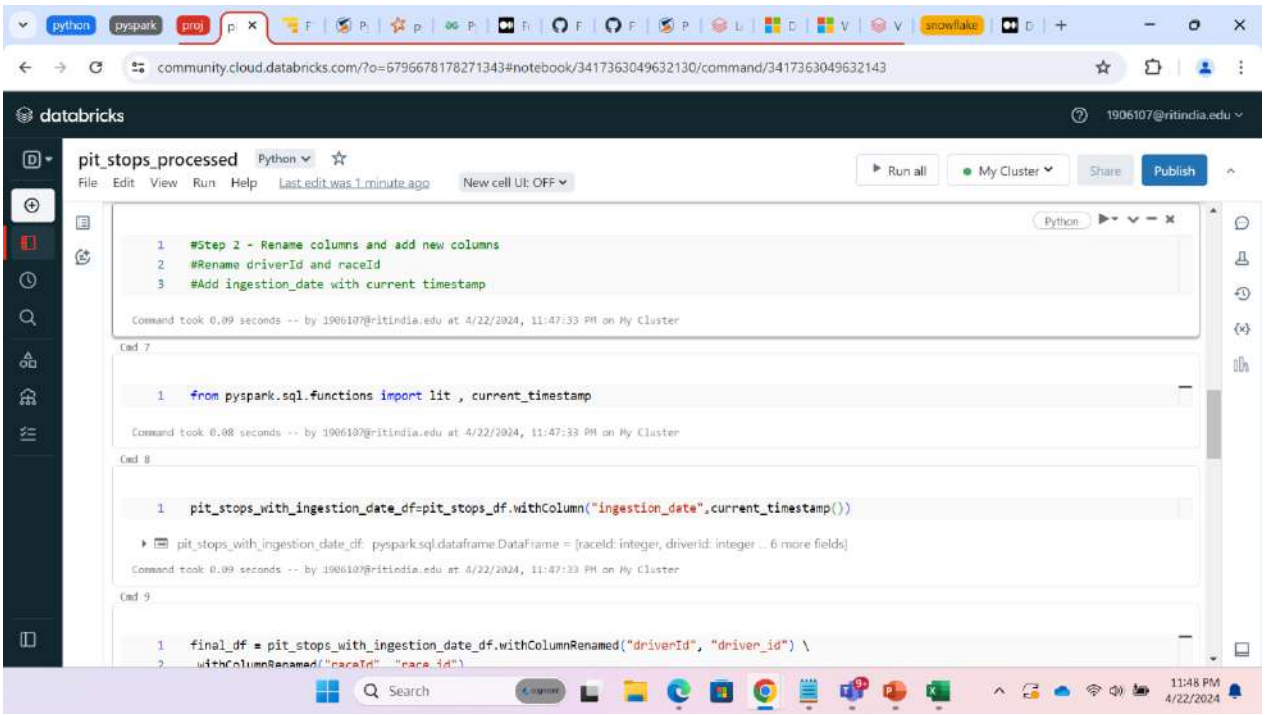

community.cloud.databricks.com/?o=6796678178271343#notebook/841007799630703/command/3271776022846954
databricks
1906107@ritindia.edu
lap_times_ingest
Python
File Edit View Run Help
Last edit was 1 minute ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
final_df.show(truncate=False)
(1) Spark Jobs
+-------+---------+---+--------+--------+------------+-----------------------+
|race_id|driver_id|lap|position|time |milliseconds|ingestion_date |
+-------+---------+---+--------+--------+------------+-----------------------+
841	20	1	1	1:38.109	98109	2024-04-22 18:12:25.882
841	20	2	1	1:33.006	93006	2024-04-22 18:12:25.882
841	20	3	1	1:32.713	92713	2024-04-22 18:12:25.882
841	20	4	1	1:32.803	92803	2024-04-22 18:12:25.882
841	20	5	1	1:32.342	92342	2024-04-22 18:12:25.882
841	20	6	1	1:32.605	92605	2024-04-22 18:12:25.882
841	20	7	1	1:32.502	92502	2024-04-22 18:12:25.882
841	20	8	1	1:32.537	92537	2024-04-22 18:12:25.882
841	20	9	1	1:33.240	93240	2024-04-22 18:12:25.882
841	20	10	1	1:32.572	92572	2024-04-22 18:12:25.882
841	20	11	1	1:32.669	92669	2024-04-22 18:12:25.882
841	20	12	1	1:32.902	92902	2024-04-22 18:12:25.882
841	20	13	1	1:33.698	93698	2024-04-22 18:12:25.882
841	20	14	3	1:52.075	112075	2024-04-22 18:12:25.882
841	20	15	4	1:38.385	98385	2024-04-22 18:12:25.882
841	20	16	2	1:31.548	91548	2024-04-22 18:12:25.882
841	20	17	1	1:30.800	90800	2024-04-22 18:12:25.882
Search
11:42 PM
4/22/2024

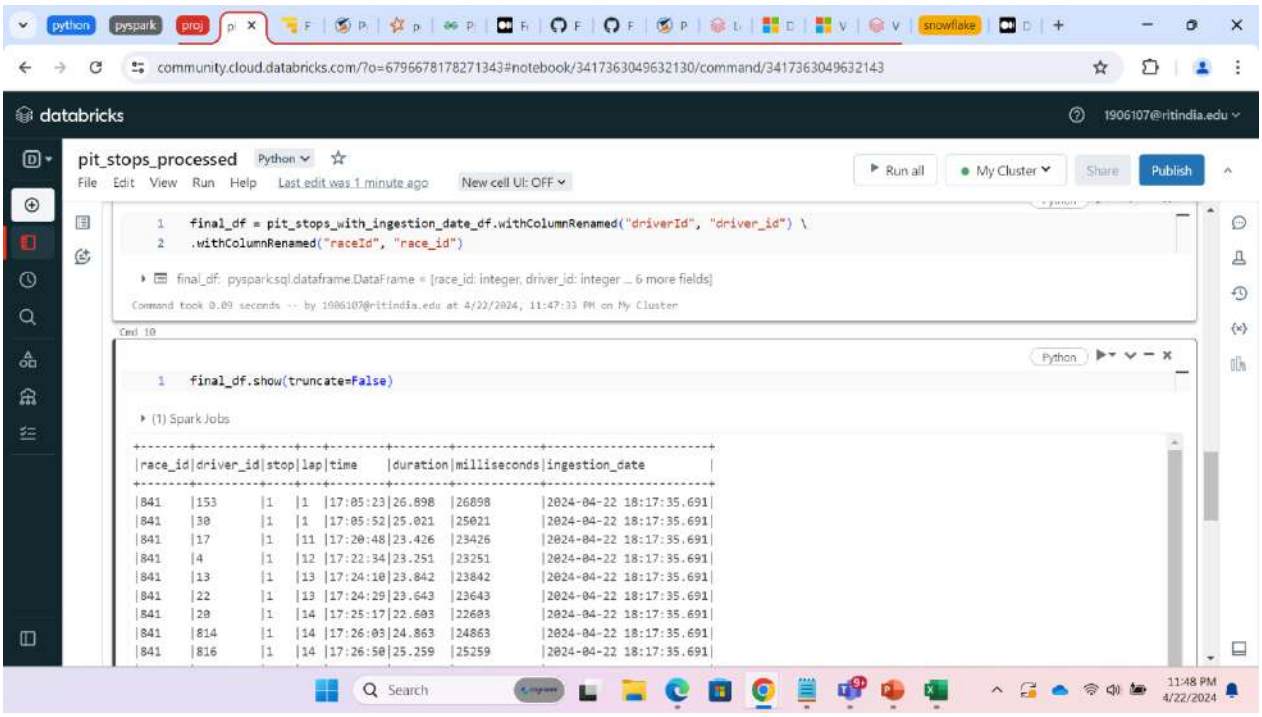


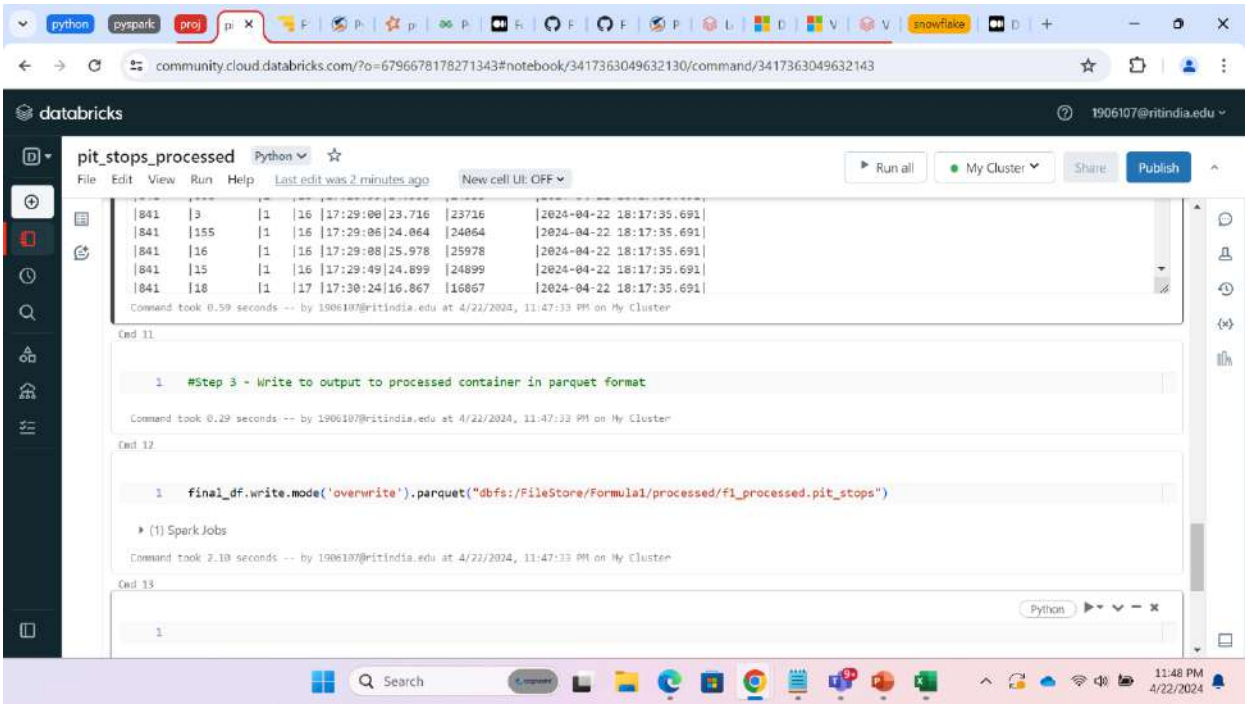
5. PIT STOP INGESTION:



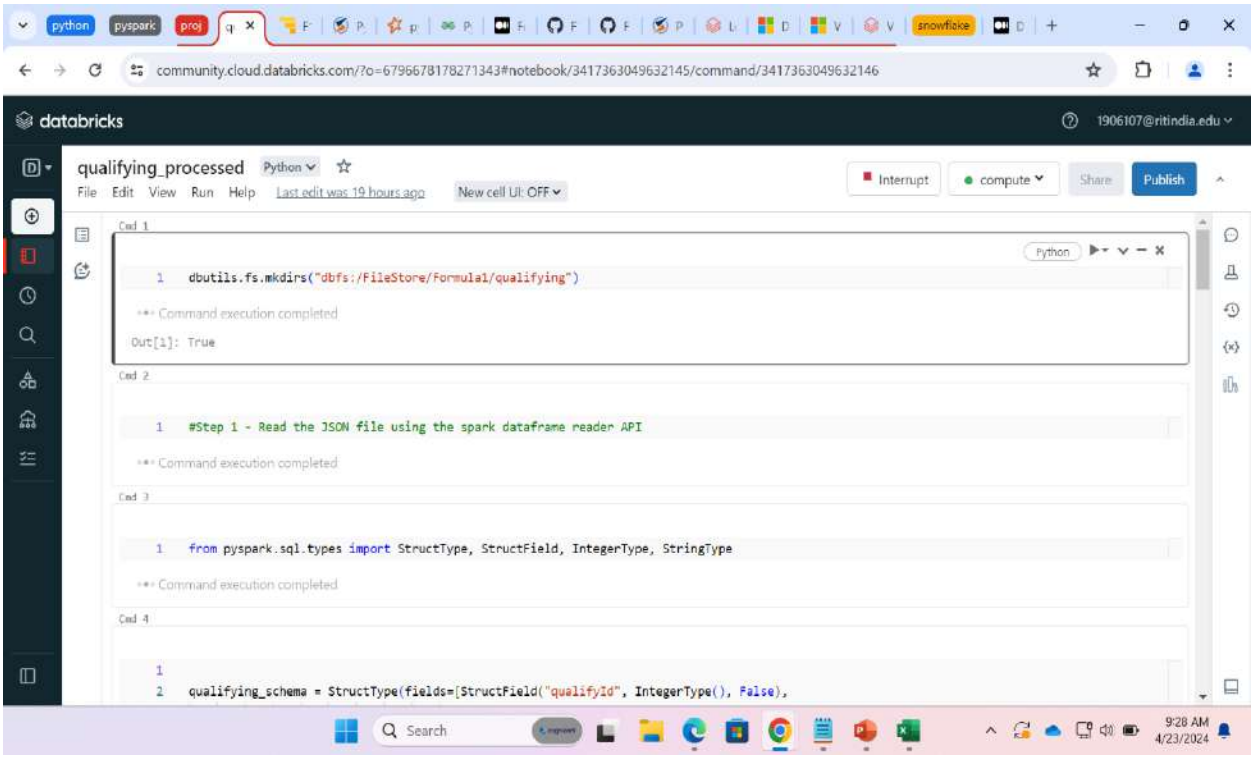

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632130/command/3417363049632142
databricks
1906107@ritindia.edu
pit_stops_processed
Python
File Edit View Run Help Last edit was 9 hours ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
1 pit_stops_df = spark.read.json("dbfs:/FileStore/pit_stops.json",schema = pit_stops_schema , multiLine=True)
pit_stops_df: pyspark.sql.dataframe.DataFrame = [raceId: integer, driverId: integer ... 5 more fields]
Command took 0.33 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:43:46 PM on My Cluster
Cmd 5
1 pit_stops_df.show()
(1) Spark Jobs
raceId	driverId	stop	lap	time	duration	milliseconds
841	153	1	1	17:05:23	26.898	26898
841	30	1	1	17:05:52	25.021	25021
841	17	1	11	17:20:48	23.426	23426
841	4	1	12	17:22:34	23.251	23251
841	13	1	13	17:24:10	23.842	23842
841	22	1	13	17:24:29	23.643	23643
841	20	1	14	17:25:17	22.603	22603
841	814	1	14	17:26:03	24.863	24863
841	816	1	14	17:26:50	25.259	25259
Search
11:44 PM
4/22/2024

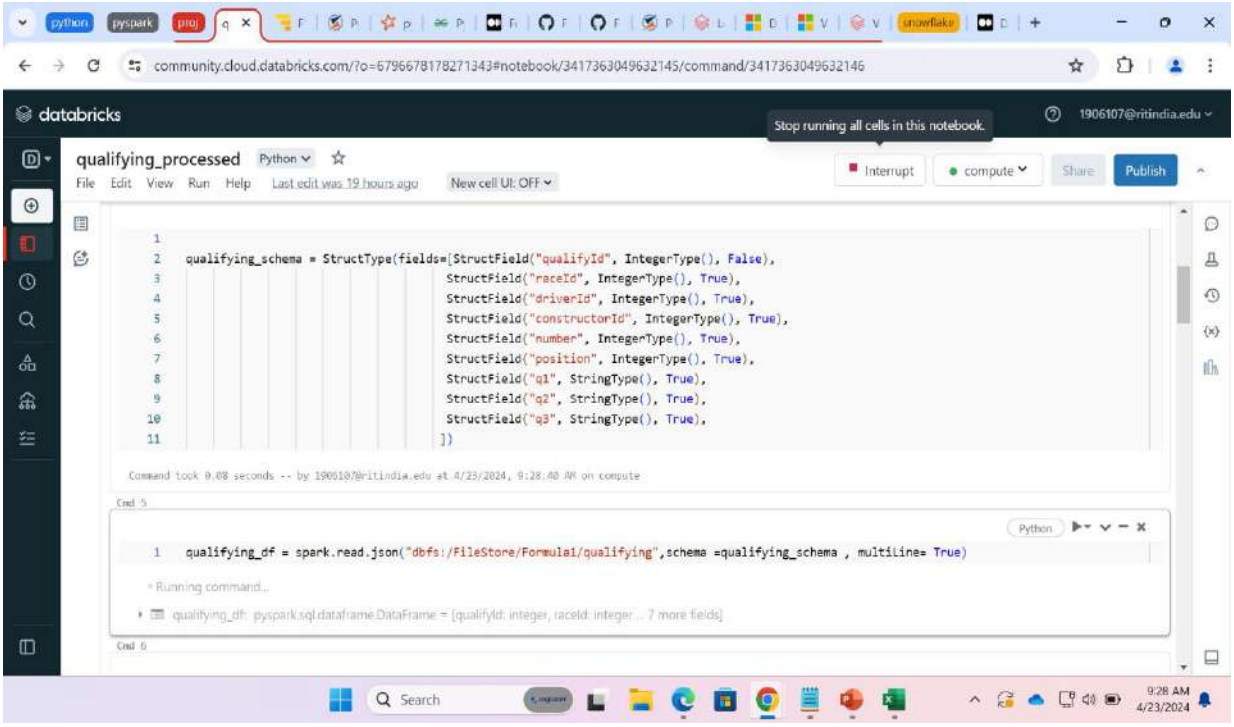

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632130/command/3417363049632143
databricks
1906107@ritindia.edu
pit_stops_processed
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
1 #Step 2 - Rename columns and add new columns
2 #Rename driverId and raceId
3 #Add ingestion_date with current timestamp
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 7
1 from pyspark.sql.functions import lit , current_timestamp
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 8
1 pit_stops_with_ingestion_date_df=pit_stops_df.withColumn("ingestion_date",current_timestamp())
pit_stops_with_ingestion_date_df: pyspark.sql.dataframe.DataFrame = [raceId: integer, driverId: integer ... 6 more fields]
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 9
1 final_df = pit_stops_with_ingestion_date_df.withColumnRenamed("driverId", "driver_id") \
2 .withColumnRenamed("raceId", "race_id")
Search
11:48 PM
4/22/2024

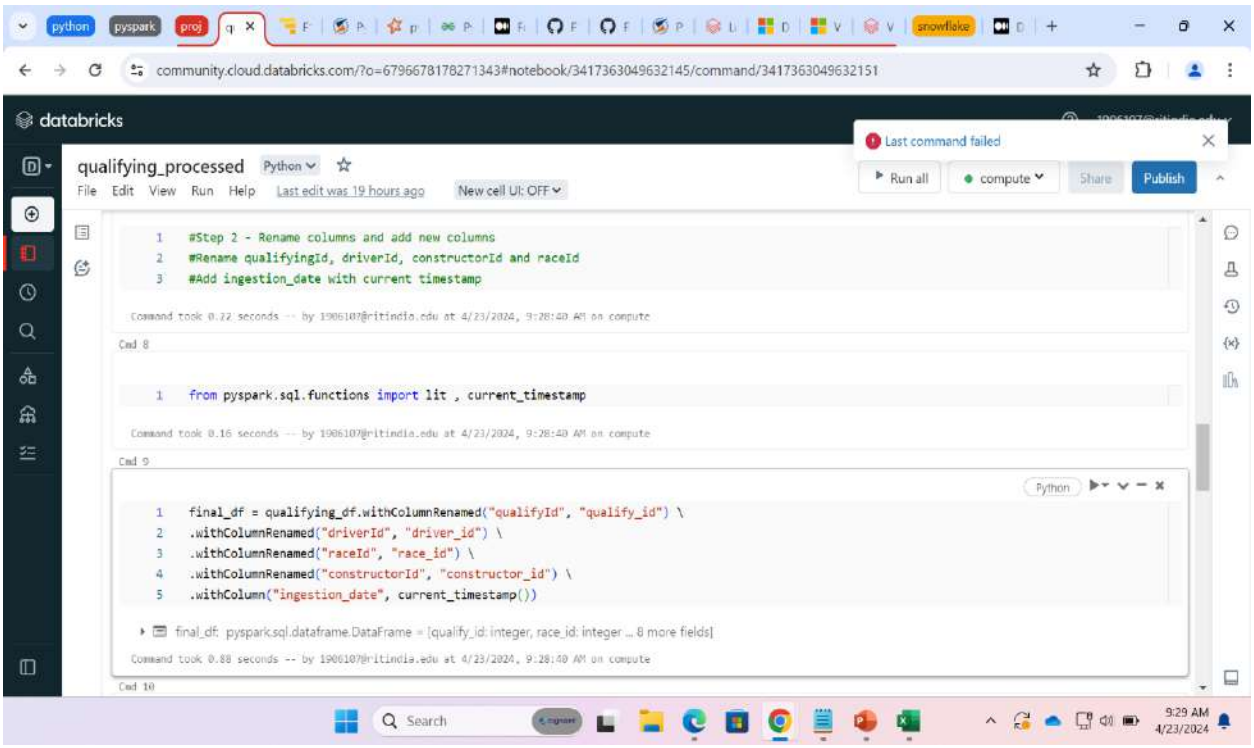

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632130/command/3417363049632143
databricks
1906107@ritindia.edu
pit_stops_processed
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
final_df = pit_stops_with_ingestion_date_df.withColumnRenamed("driverId", "driver_id") \
.withColumnRenamed("raceId", "race_id")
final_df: pyspark.sql.dataframe.DataFrame = [race_id: integer, driver_id: integer ... 6 more fields]
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 10
final_df.show(truncate=False)
(1) Spark Jobs
race_id	driver_id	stop	lap	time	duration	milliseconds	ingestion_date
841	153	1	1	17:05:23	26.898	26898	2024-04-22 18:17:35.691
841	30	1	1	17:05:52	25.021	25021	2024-04-22 18:17:35.691
841	17	1	11	17:20:48	23.426	23426	2024-04-22 18:17:35.691
841	4	1	12	17:22:34	23.251	23251	2024-04-22 18:17:35.691
841	13	1	13	17:24:10	23.842	23842	2024-04-22 18:17:35.691
841	22	1	13	17:24:29	23.643	23643	2024-04-22 18:17:35.691
841	20	1	14	17:25:17	22.603	22603	2024-04-22 18:17:35.691
841	814	1	14	17:26:03	24.863	24863	2024-04-22 18:17:35.691
841	816	1	14	17:26:50	25.259	25259	2024-04-22 18:17:35.691
Search
11:48 PM
4/22/2024

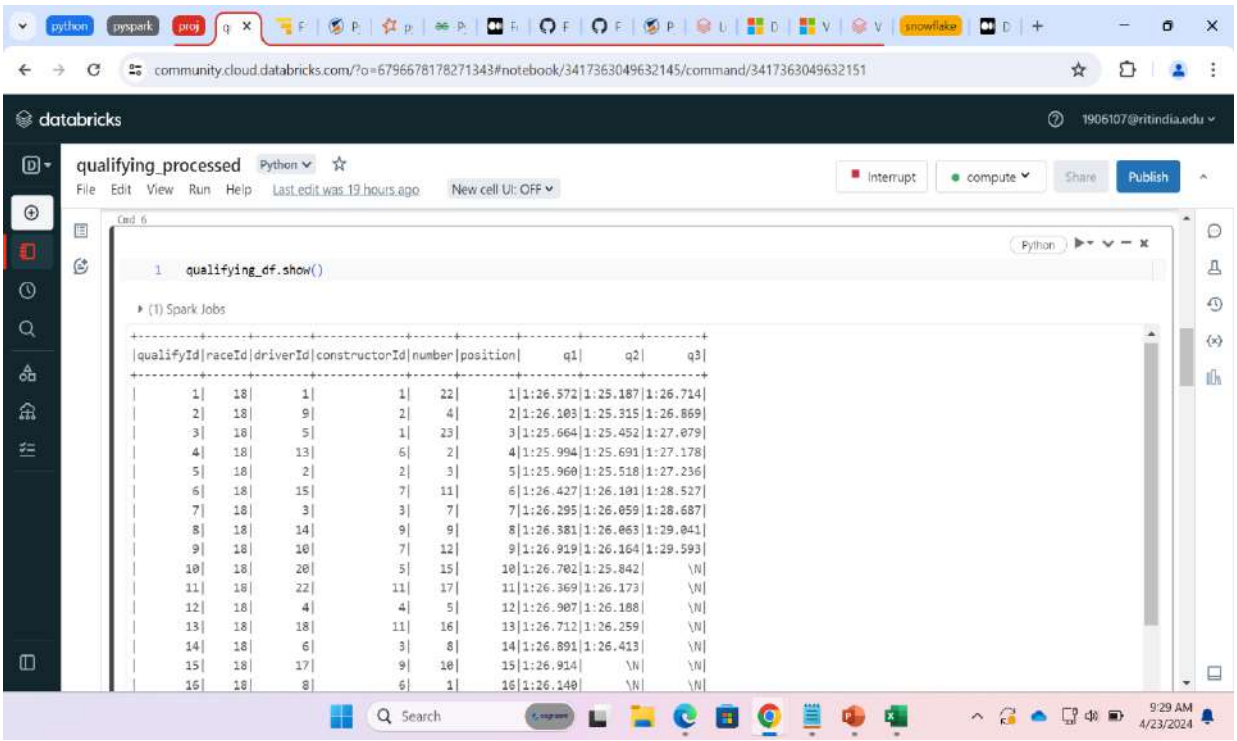

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632130/command/3417363049632143
databricks
1906107@ritindia.edu
pit_stops_processed
Python
File Edit View Run Help Last edit was 2 minutes ago
New cell UI: OFF
Run all
My Cluster
Share
Publish
841	3	1	16	17:29:00	23.716	23716	2024-04-22 18:17:35.691
841	155	1	16	17:29:06	24.064	24064	2024-04-22 18:17:35.691
841	16	1	16	17:29:08	25.978	25978	2024-04-22 18:17:35.691
841	15	1	16	17:29:49	24.899	24899	2024-04-22 18:17:35.691
841	18	1	17	17:30:24	16.867	16867	2024-04-22 18:17:35.691
Command took 0.59 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 11
#Step 3 - Write to output to processed container in parquet format
Command took 0.29 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 12
final_df.write.mode('overwrite').parquet("dbfs:/FileStore/Formula1/processed/f1_processed.pit_stops")
(1) Spark Jobs
Command took 2.10 seconds -- by 1906107@ritindia.edu at 4/22/2024, 11:47:33 PM on My Cluster
Cmd 13
Python
Search
11:48 PM
4/22/2024

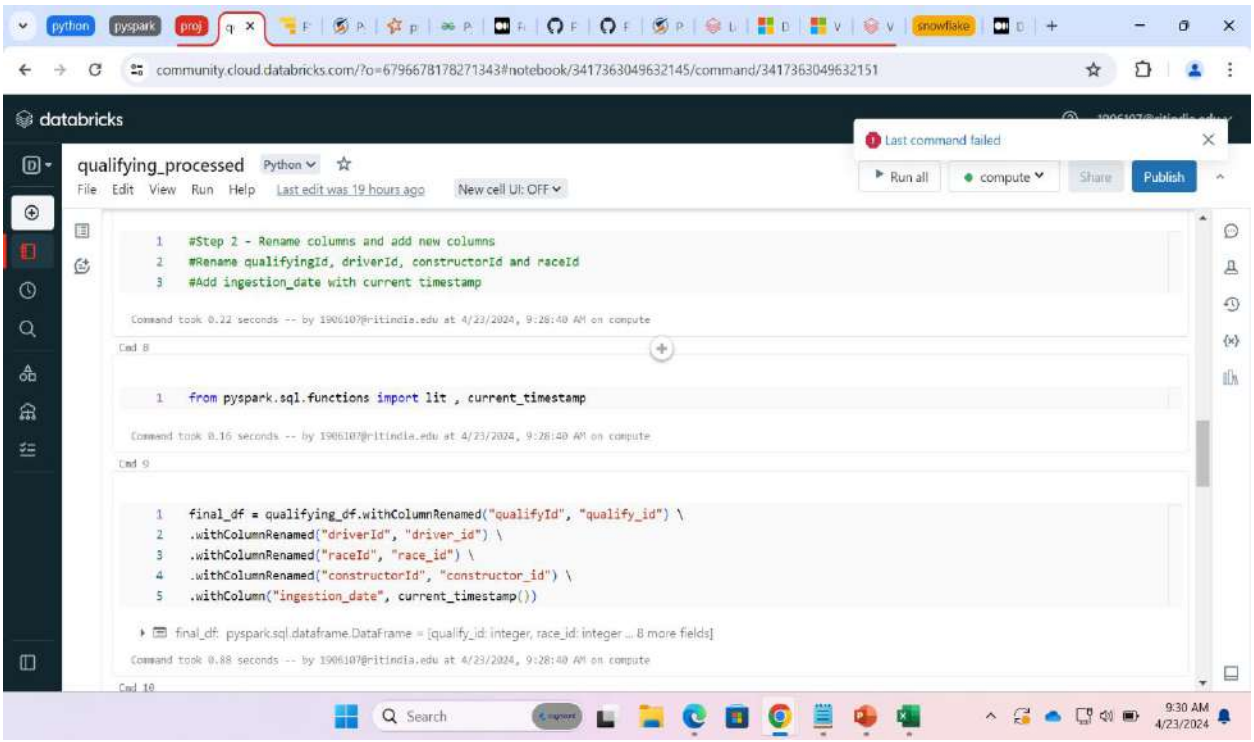
6. Qualifying ingest:

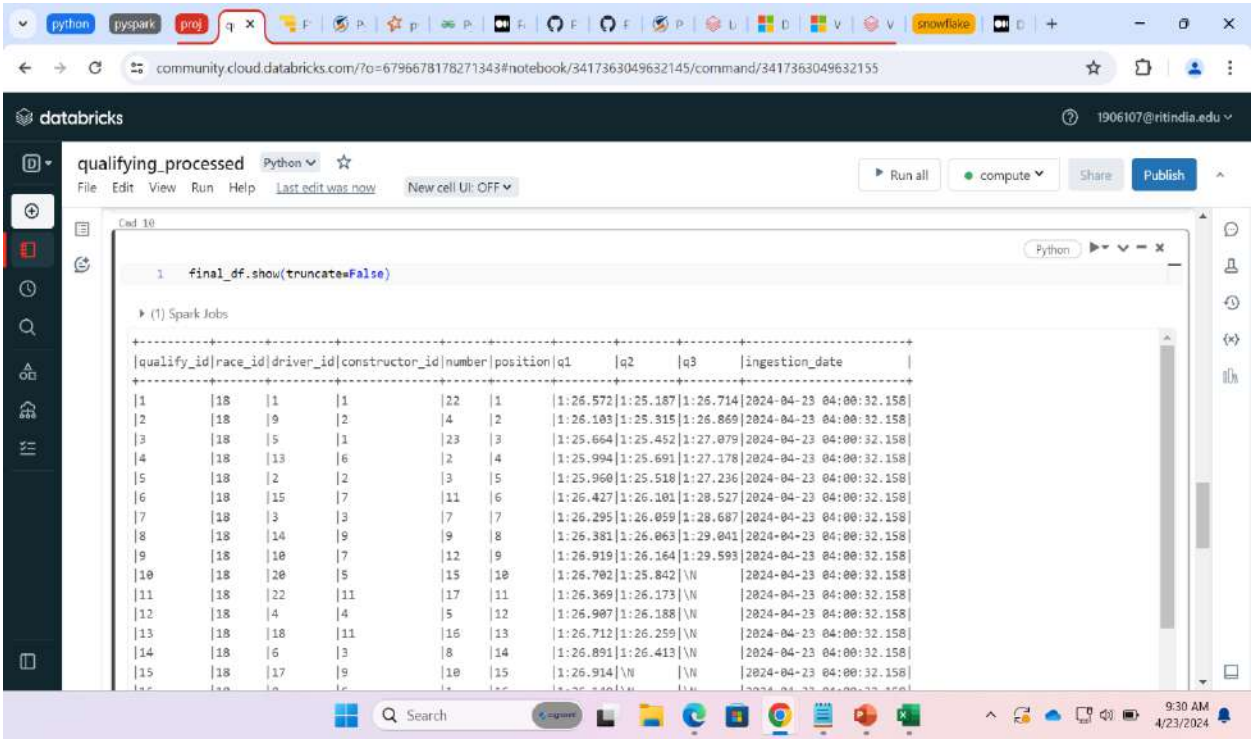


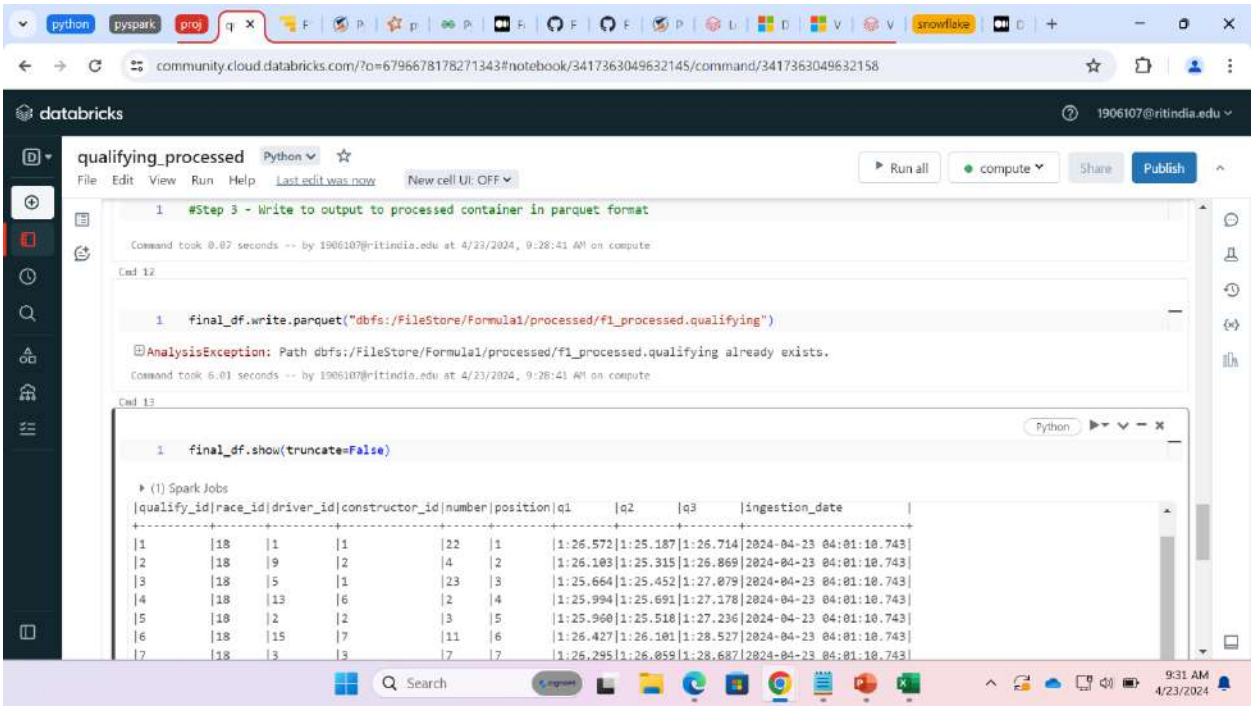



community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632145/command/3417363049632151
databricks
qualifying_processed
Python
File Edit View Run Help Last edit was 19 hours ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
#Step 2 - Rename columns and add new columns
#Rename qualifyingId, driverId, constructorId and raceId
#Add ingestion_date with current timestamp
Command took 0.22 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 8
from pyspark.sql.functions import lit , current_timestamp
Command took 0.16 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 9
final_df = qualifying_df.withColumnRenamed("qualifyId", "qualify_id") \
.withColumnRenamed("driverId", "driver_id") \
.withColumnRenamed("raceId", "race_id") \
.withColumnRenamed("constructorId", "constructor_id") \
.withColumn("ingestion_date", current_timestamp())
final_df: pyspark.sql.dataframe.DataFrame = [qualify_id: integer, race_id: integer ... 8 more fields]
Command took 0.88 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 10
Search
9:29 AM
4/23/2024

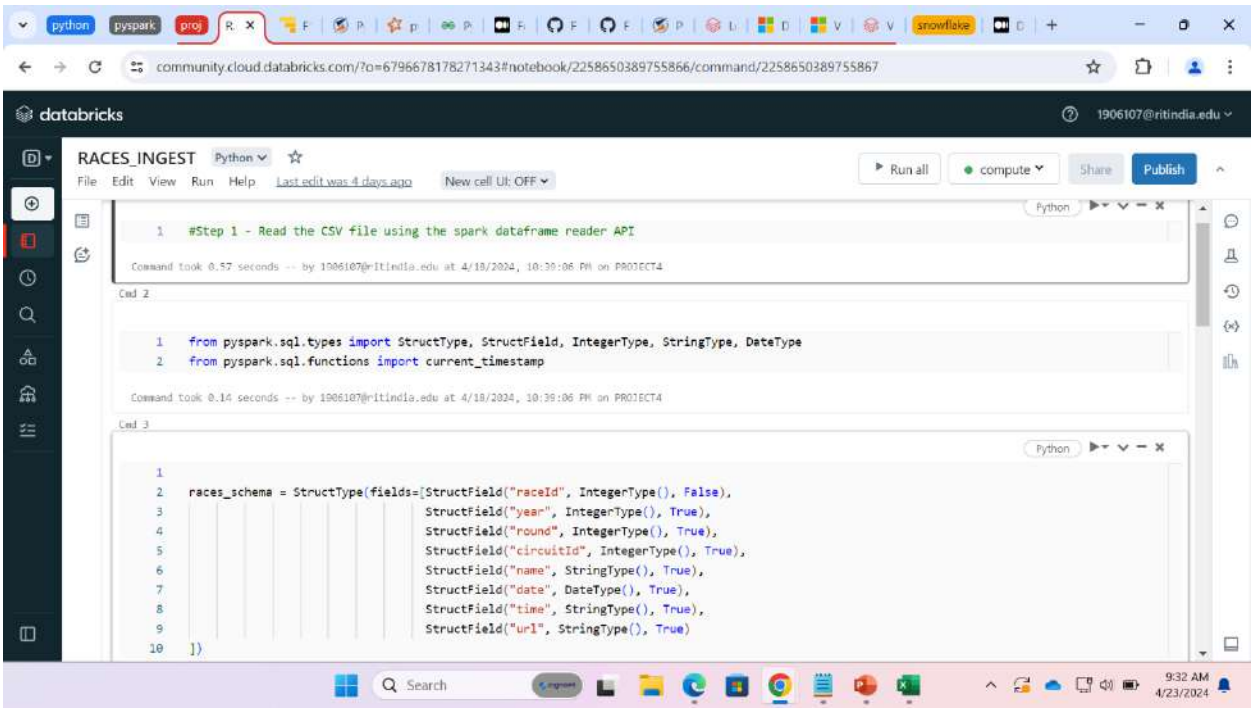

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632145/command/3417363049632151
databricks
1906107@ritindia.edu
qualifying_processed
Python
File Edit View Run Help Last edit was 19 hours ago
New cell UI: OFF
Interrupt
compute
Share
Publish
Cmd 6
qualifying_df.show()
(1) Spark Jobs
+---------+------+--------+-------------+------+--------+--------+--------+--------+
|qualifyId|raceId|driverId|constructorId|number|position| q1| q2| q3|
+---------+------+--------+-------------+------+--------+--------+--------+--------+
1	18	1	1	22	1	1:26.572	1:25.187	1:26.714
2	18	9	2	4	2	1:26.103	1:25.315	1:26.869
3	18	5	1	23	3	1:25.664	1:25.452	1:27.079
4	18	13	6	2	4	1:25.994	1:25.691	1:27.178
5	18	2	2	3	5	1:25.960	1:25.518	1:27.236
6	18	15	7	11	6	1:26.427	1:26.101	1:28.527
7	18	3	3	7	7	1:26.295	1:26.059	1:28.687
8	18	14	9	9	8	1:26.381	1:26.063	1:29.041
9	18	10	7	12	9	1:26.919	1:26.164	1:29.593
10	18	20	5	15	10	1:26.702	1:25.842	\N
11	18	22	11	17	11	1:26.369	1:26.173	\N
12	18	4	4	5	12	1:26.907	1:26.188	\N
13	18	18	11	16	13	1:26.712	1:26.259	\N
14	18	6	3	8	14	1:26.891	1:26.413	\N
15	18	17	9	10	15	1:26.914	\N	\N
16	18	8	6	1	16	1:26.140	\N	\N
Search
9:29 AM
4/23/2024

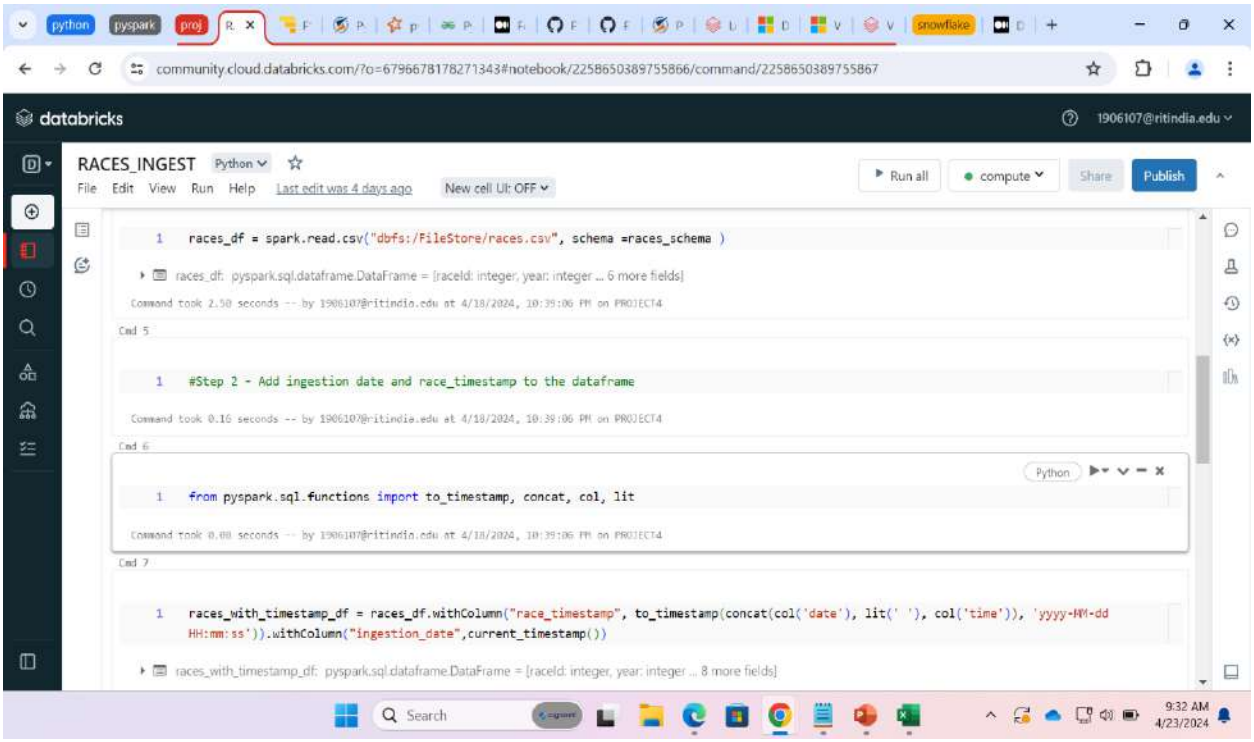

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632145/command/3417363049632151
databricks
qualifying_processed
Python
File Edit View Run Help
Last edit was 19 hours ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
#Step 2 - Rename columns and add new columns
#Rename qualifyingId, driverId, constructorId and raceId
#Add ingestion_date with current timestamp
Command took 0.22 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 8
from pyspark.sql.functions import lit , current_timestamp
Command took 0.16 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 9
final_df = qualifying_df.withColumnRenamed("qualifyId", "qualify_id") \
.withColumnRenamed("driverId", "driver_id") \
.withColumnRenamed("raceId", "race_id") \
.withColumnRenamed("constructorId", "constructor_id") \
.withColumn("ingestion_date", current_timestamp())
final_df: pyspark.sql.dataframe.DataFrame = [qualify_id: integer, race_id: integer ... 8 more fields]
Command took 0.88 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:28:40 AM on compute
Cmd 10
Search
9:30 AM
4/23/2024

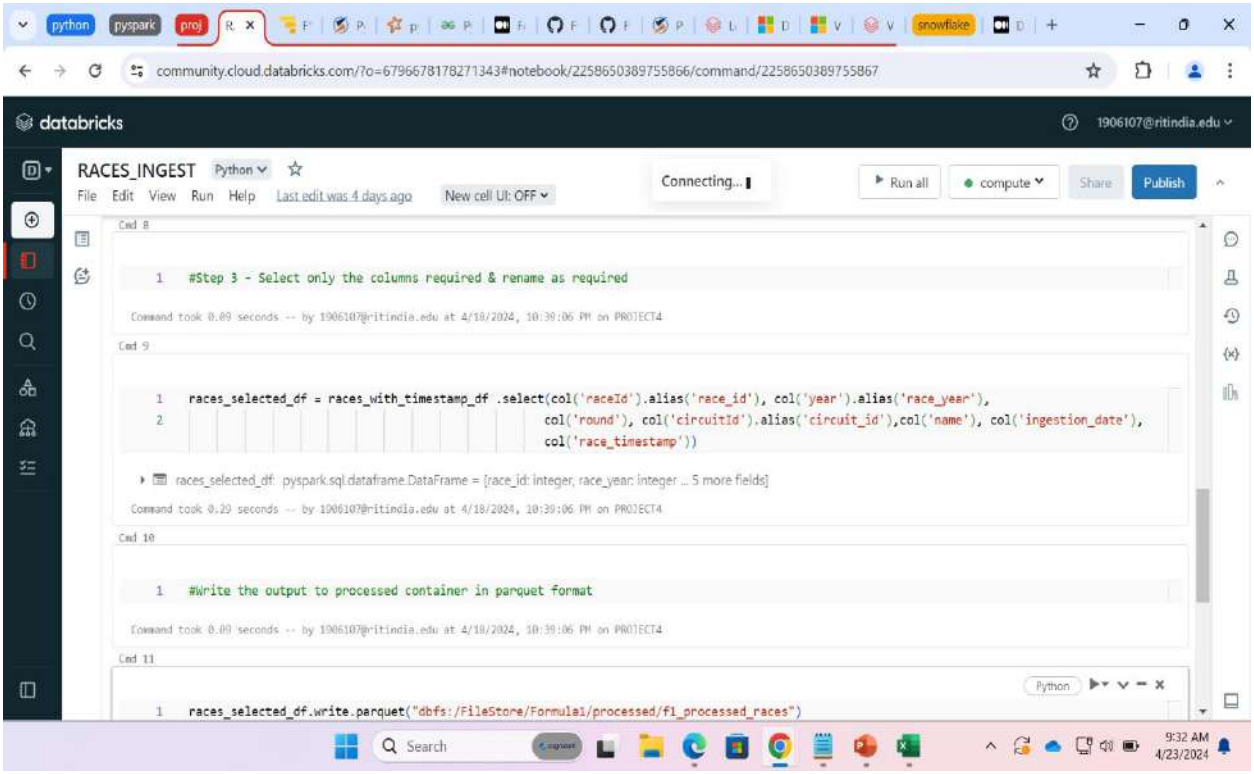

community.cloud.databricks.com/?o=6796678178271343#notebook/3417363049632145/command/3417363049632155
databricks
1906107@ritindia.edu
qualifying_processed
Python
File Edit View Run Help
Last edit was now
New cell UI: OFF
Run all
compute
Share
Publish
Cmd 10
final_df.show(truncate=False)
(1) Spark Jobs
+----------+-------+---------+--------------+------+--------+--------+--------+--------+-----------------------+
|qualify_id|race_id|driver_id|constructor_id|number|position|q1 |q2 |q3 |ingestion_date |
+----------+-------+---------+--------------+------+--------+--------+--------+--------+-----------------------+
1	18	1	1	22	1	1:26.572	1:25.187	1:26.714	2024-04-23 04:00:32.158
2	18	9	2	4	2	1:26.103	1:25.315	1:26.869	2024-04-23 04:00:32.158
3	18	5	1	23	3	1:25.664	1:25.452	1:27.079	2024-04-23 04:00:32.158
4	18	13	6	2	4	1:25.994	1:25.691	1:27.178	2024-04-23 04:00:32.158
5	18	2	2	3	5	1:25.960	1:25.518	1:27.236	2024-04-23 04:00:32.158
6	18	15	7	11	6	1:26.427	1:26.101	1:28.527	2024-04-23 04:00:32.158
7	18	3	3	7	7	1:26.295	1:26.059	1:28.687	2024-04-23 04:00:32.158
8	18	14	9	9	8	1:26.381	1:26.063	1:29.041	2024-04-23 04:00:32.158
9	18	10	7	12	9	1:26.919	1:26.164	1:29.593	2024-04-23 04:00:32.158
10	18	20	5	15	10	1:26.702	1:25.842	\N	2024-04-23 04:00:32.158
11	18	22	11	17	11	1:26.369	1:26.173	\N	2024-04-23 04:00:32.158
12	18	4	4	5	12	1:26.907	1:26.188	\N	2024-04-23 04:00:32.158
13	18	18	11	16	13	1:26.712	1:26.259	\N	2024-04-23 04:00:32.158
14	18	6	3	8	14	1:26.891	1:26.413	\N	2024-04-23 04:00:32.158
15	18	17	9	10	15	1:26.914	\N	\N	2024-04-23 04:00:32.158
Search
9:30 AM
4/23/2024

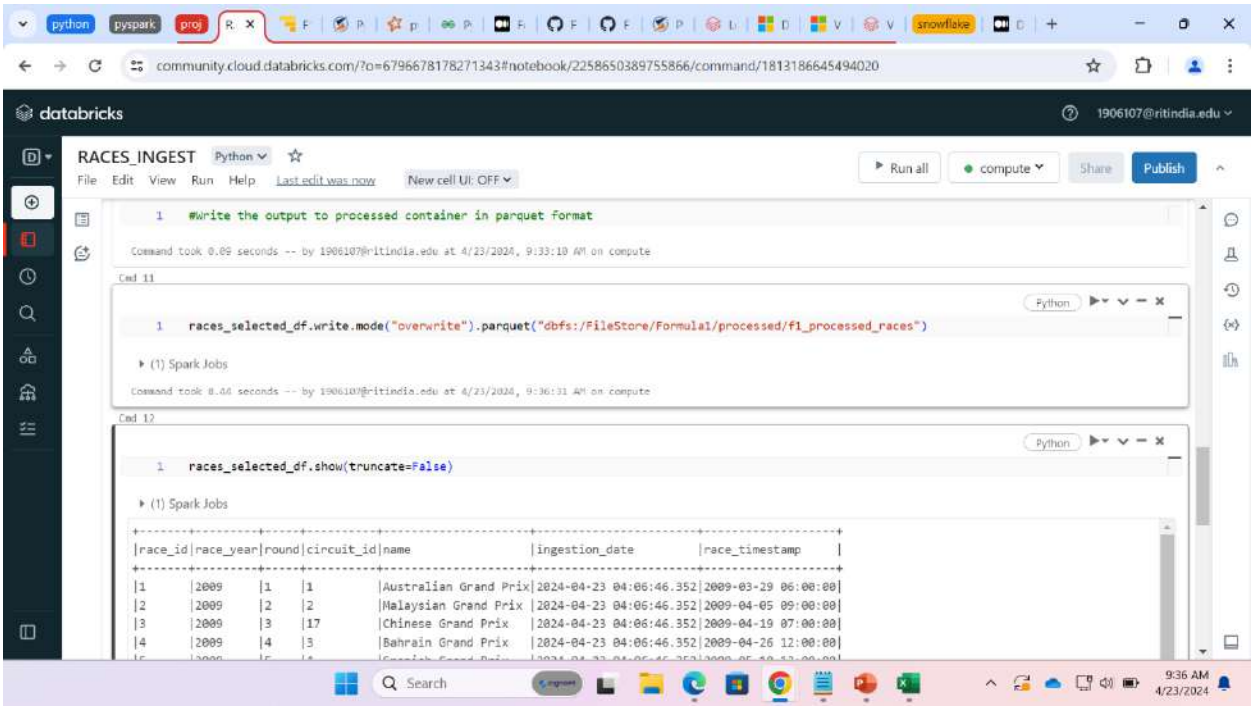


7. Race Ingest

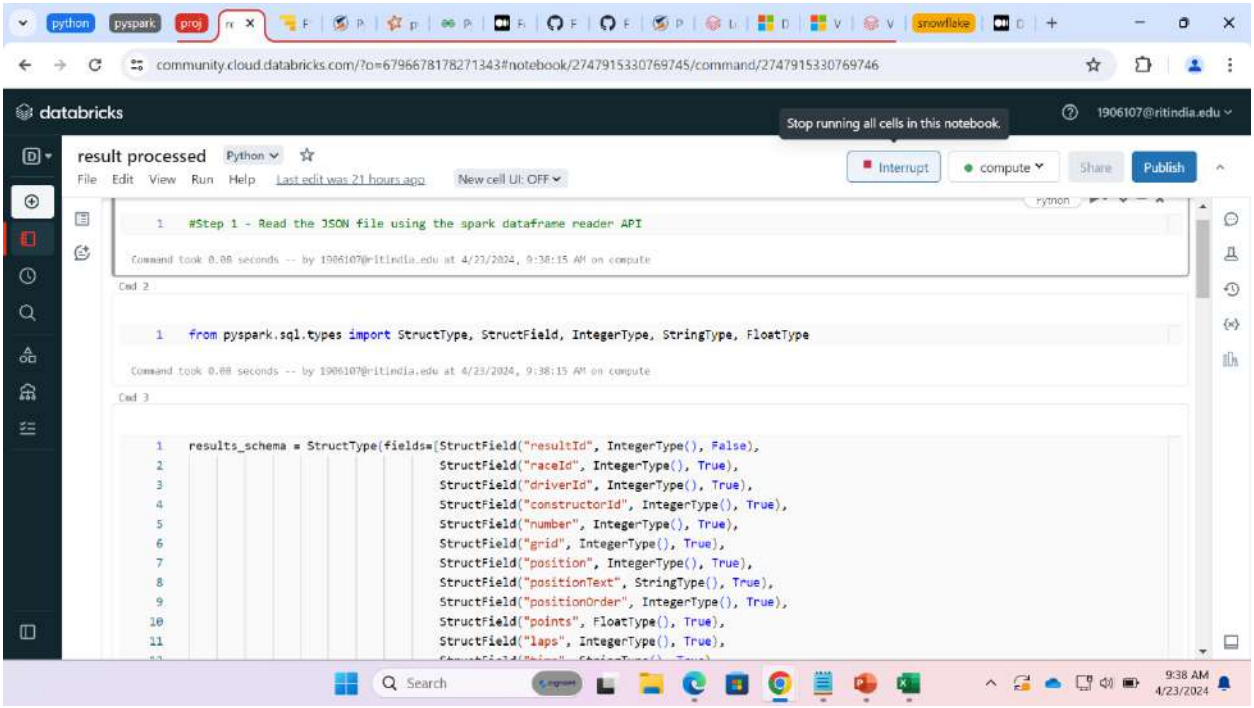


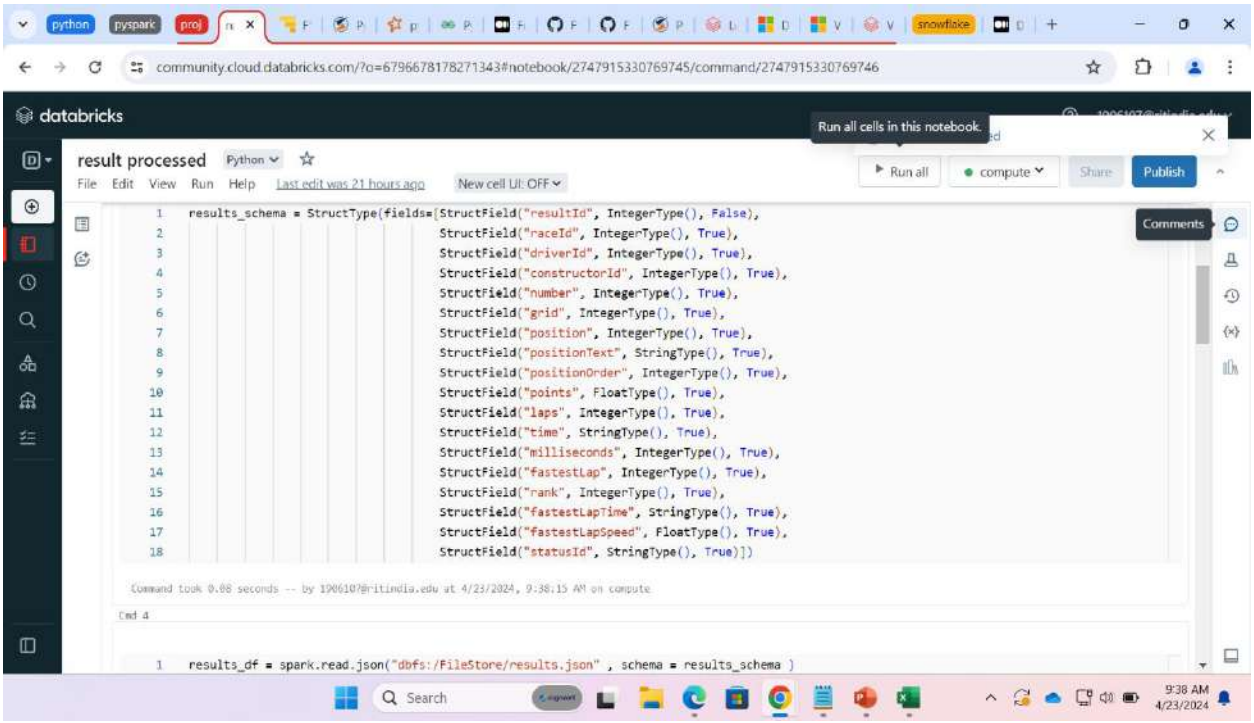

community.cloud.databricks.com/?o=6796678178271343#notebook/2258650389755866/command/2258650389755867
databricks
1906107@ritindia.edu
RACES_INGEST
Python
File Edit View Run Help Last edit was 4 days ago
New cell UI: OFF
Run all
compute
Share
Publish
1 races_df = spark.read.csv("dbfs:/FileStore/races.csv", schema =races_schema)
races_df: pyspark.sql.dataframe.DataFrame = [raceId: integer, year: integer ... 6 more fields]
Command took 2.50 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 5
1 #Step 2 - Add ingestion date and race_timestamp to the dataframe
Command took 0.16 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 6
Python
1 from pyspark.sql.functions import to_timestamp, concat, col, lit
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 7
1 races_with_timestamp_df = races_df.withColumn("race_timestamp", to_timestamp(concat(col('date'), lit(' '), col('time')), 'yyyy-MM-dd HH:mm:ss')).withColumn("ingestion_date",current_timestamp())
races_with_timestamp_df: pyspark.sql.dataframe.DataFrame = [raceId: integer, year: integer ... 8 more fields]
Search
9:32 AM
4/23/2024

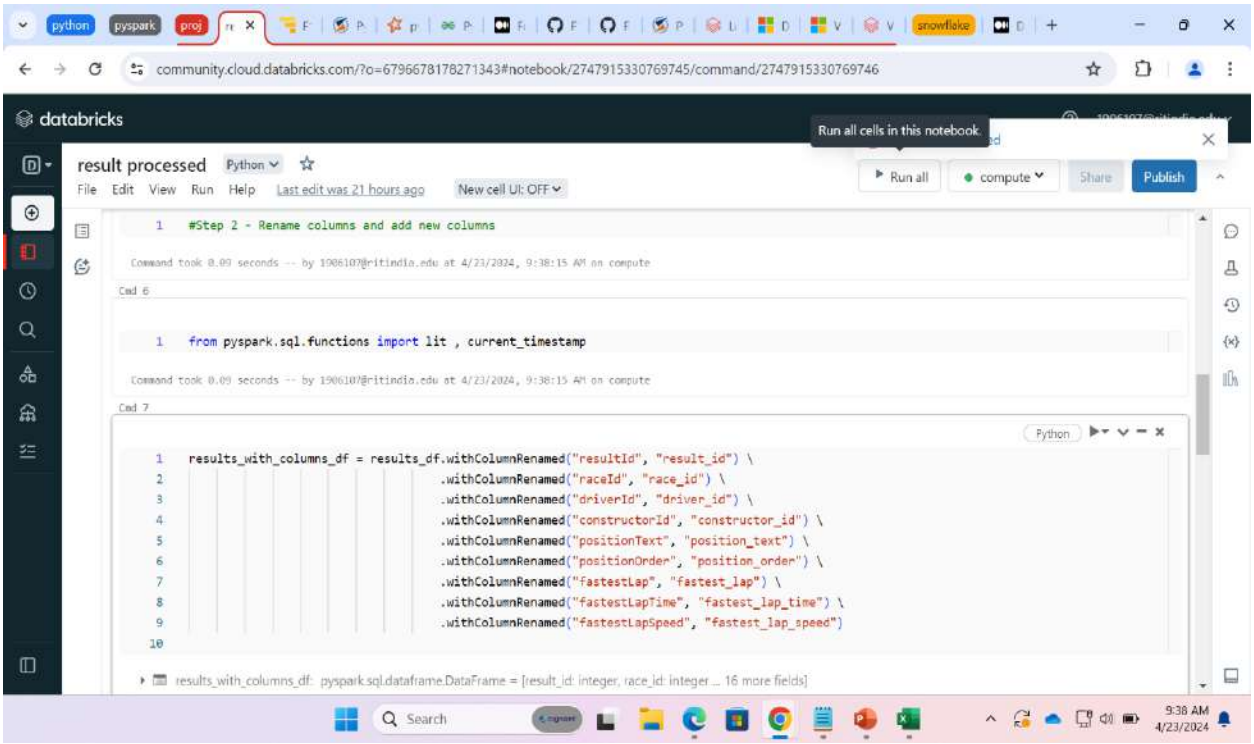

community.cloud.databricks.com/?o=6796678178271343#notebook/2258650389755866/command/2258650389755867
databricks
1906107@ritindia.edu
RACES_INGEST
Python
File Edit View Run Help Last edit was 4 days ago
New cell UI: OFF
Connecting...
Run all
compute
Share
Publish
Cmd 8
1 #Step 3 - Select only the columns required & rename as required
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 9
1 races_selected_df = races_with_timestamp_df .select(col('raceId').alias('race_id'), col('year').alias('race_year'),
2 col('round'), col('circuitId').alias('circuit_id'),col('name'), col('ingestion_date'),
col('race_timestamp'))
races_selected_df: pyspark.sql.dataframe.DataFrame = [race_id: integer, race_year: integer ... 5 more fields]
Command took 0.29 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 10
1 #Write the output to processed container in parquet format
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/18/2024, 10:39:06 PM on PROJECT4
Cmd 11
Python
1 races_selected_df.write.parquet("dbfs:/FileStore/Formula1/processed/f1_processed_races")
Search
9:32 AM
4/23/2024

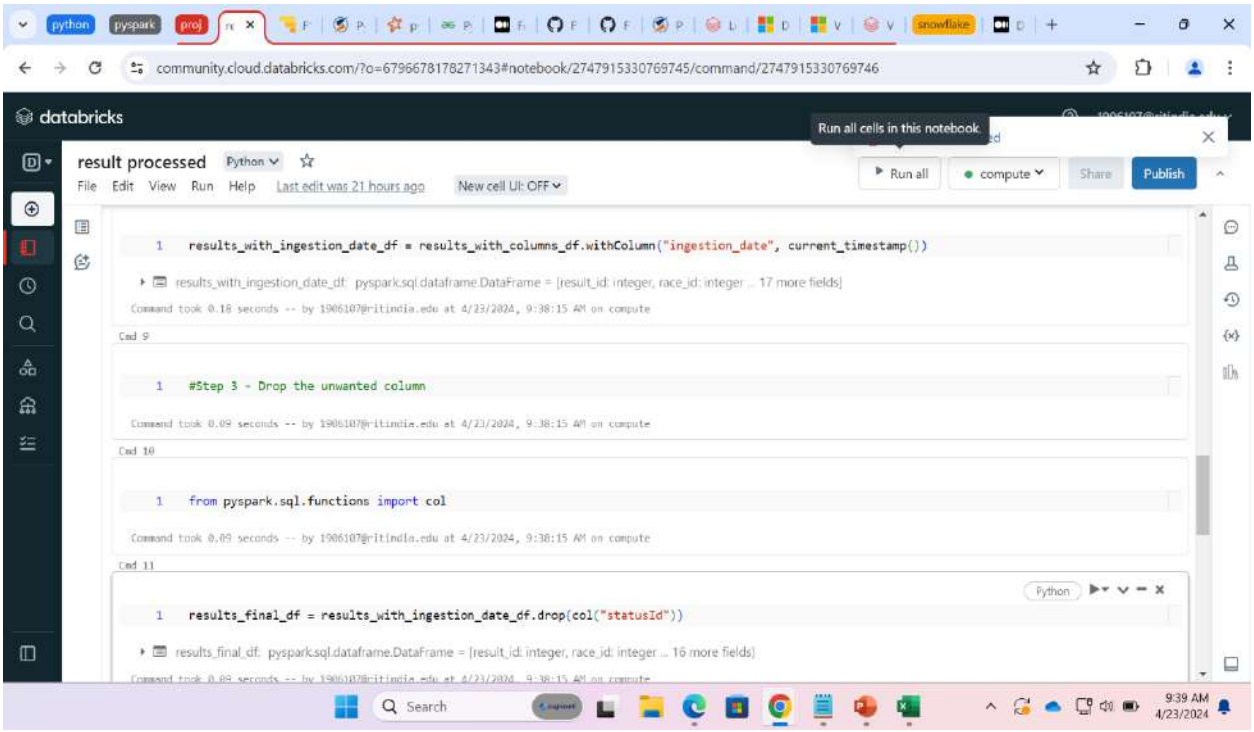


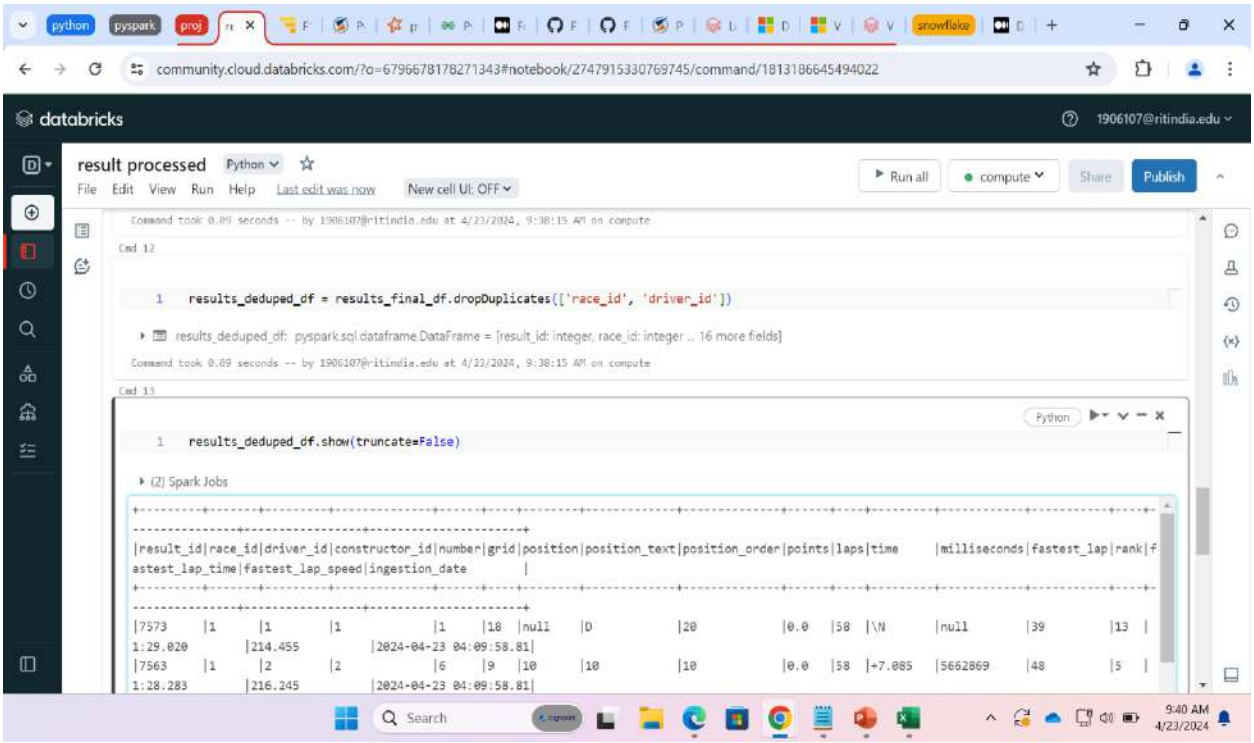
8. Result file ingestion

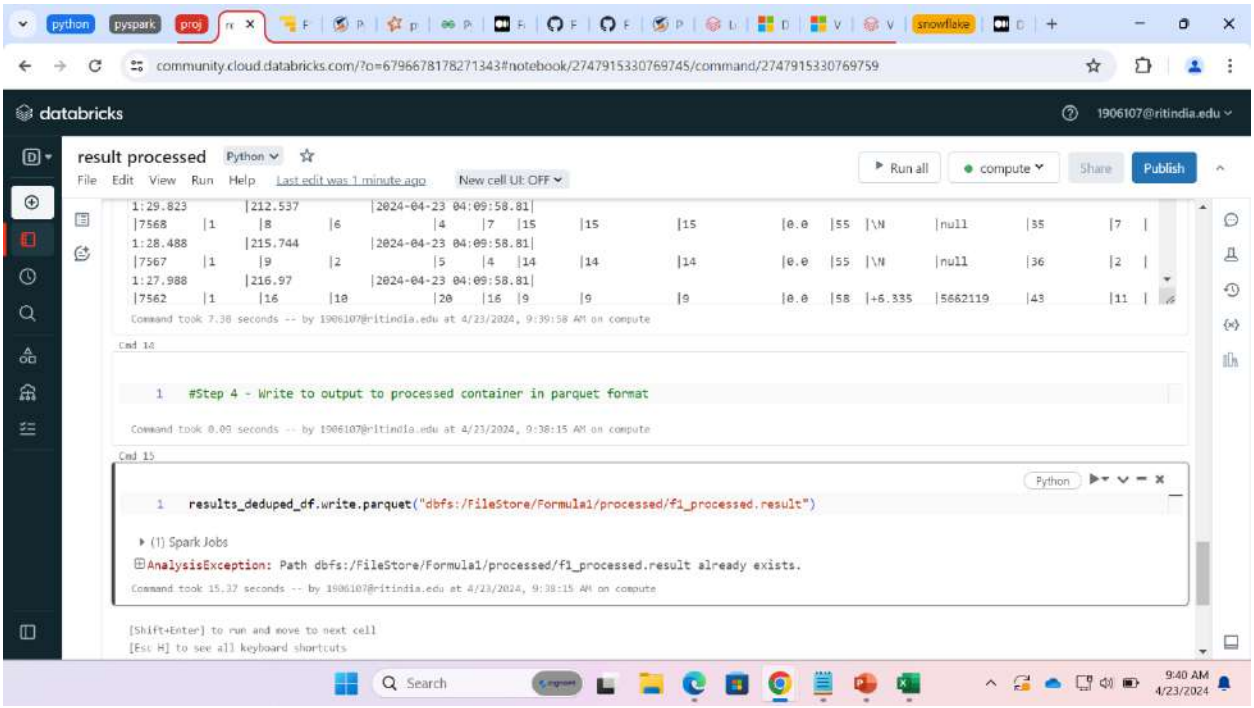



community.cloud.databricks.com/?o=6796678178271343#notebook/2747915330769745/command/2747915330769746
databricks
result processed
Python
File Edit View Run Help Last edit was 21 hours ago
New cell UI: OFF
Run all cells in this notebook.
Run all
compute
Share
Publish
Comments
1 results_schema = StructType(fields=[StructField("resultId", IntegerType(), False),
2 StructField("raceId", IntegerType(), True),
3 StructField("driverId", IntegerType(), True),
4 StructField("constructorId", IntegerType(), True),
5 StructField("number", IntegerType(), True),
6 StructField("grid", IntegerType(), True),
7 StructField("position", IntegerType(), True),
8 StructField("positionText", StringType(), True),
9 StructField("positionOrder", IntegerType(), True),
10 StructField("points", FloatType(), True),
11 StructField("laps", IntegerType(), True),
12 StructField("time", StringType(), True),
13 StructField("milliseconds", IntegerType(), True),
14 StructField("fastestLap", IntegerType(), True),
15 StructField("rank", IntegerType(), True),
16 StructField("fastestLapTime", StringType(), True),
17 StructField("fastestLapSpeed", FloatType(), True),
18 StructField("statusId", StringType(), True)])
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 4
1 results_df = spark.read.json("dbfs:/FileStore/results.json" , schema = results_schema)
Search
9:38 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/2747915330769745/command/2747915330769746
databricks
result processed
Python
File Edit View Run Help Last edit was 21 hours ago
New cell UI: OFF
Run all cells in this notebook.
Run all
compute
Share
Publish
1 #Step 2 - Rename columns and add new columns
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 6
1 from pyspark.sql.functions import lit , current_timestamp
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 7
Python
1 results_with_columns_df = results_df.withColumnRenamed("resultId", "result_id") \
2 .withColumnRenamed("raceId", "race_id") \
3 .withColumnRenamed("driverId", "driver_id") \
4 .withColumnRenamed("constructorId", "constructor_id") \
5 .withColumnRenamed("positionText", "position_text") \
6 .withColumnRenamed("positionOrder", "position_order") \
7 .withColumnRenamed("fastestLap", "fastest_lap") \
8 .withColumnRenamed("fastestLapTime", "fastest_lap_time") \
9 .withColumnRenamed("fastestLapSpeed", "fastest_lap_speed")
10
results_with_columns_df: pyspark.sql.dataframe.DataFrame = [result_id: integer, race_id: integer ... 16 more fields]
Search
9:38 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/2747915330769745/command/2747915330769746
databricks
result processed
Python
File Edit View Run Help Last edit was 21 hours ago
New cell UI: OFF
Run all cells in this notebook
Run all
compute
Share
Publish
1 results_with_ingestion_date_df = results_with_columns_df.withColumn("ingestion_date", current_timestamp())
results_with_ingestion_date_df: pyspark.sql.dataframe.DataFrame = [result_id: integer, race_id: integer ... 17 more fields]
Command took 0.18 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 9
1 #Step 3 - Drop the unwanted column
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 10
1 from pyspark.sql.functions import col
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 11
Python
1 results_final_df = results_with_ingestion_date_df.drop(col("statusId"))
results_final_df: pyspark.sql.dataframe.DataFrame = [result_id: integer, race_id: integer ... 16 more fields]
Search
9:39 AM
4/23/2024

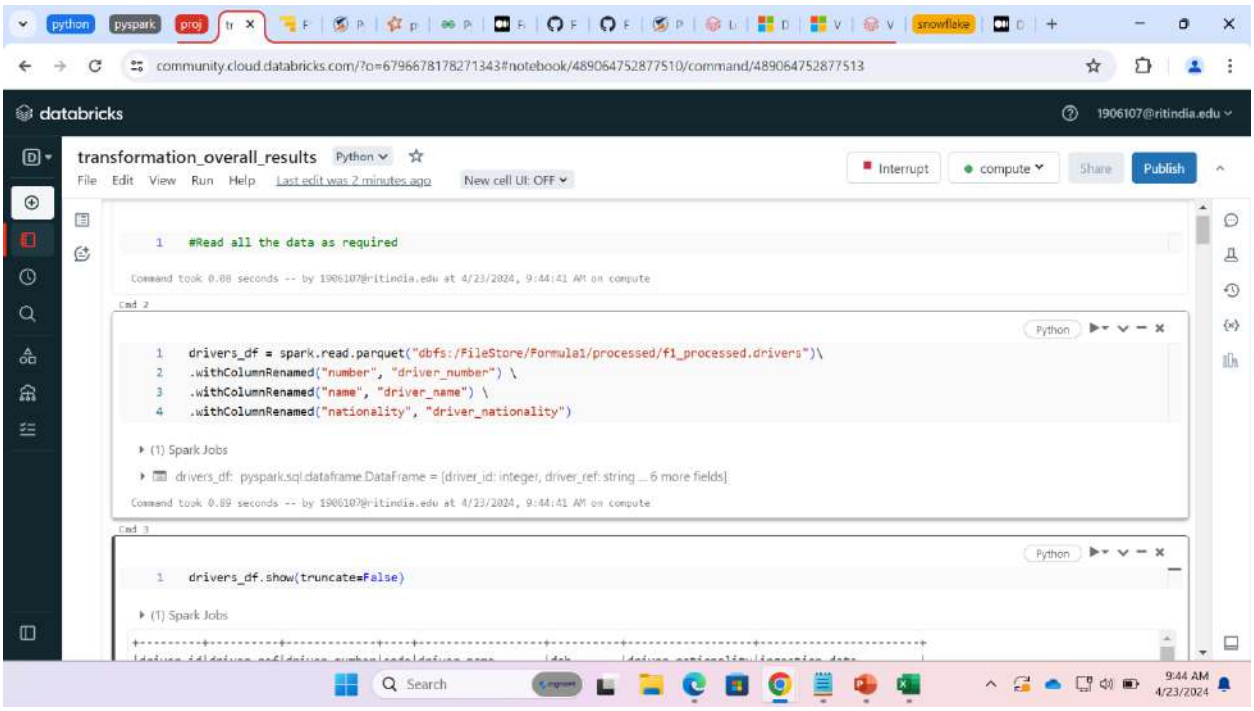

community.cloud.databricks.com/?o=6796678178271343#notebook/2747915330769745/command/1813186645494022
databricks
1906107@ritindia.edu
result processed
Python
File Edit View Run Help Last edit was now
New cell UI: OFF
Run all
compute
Share
Publish
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 12
1 results_deduped_df = results_final_df.dropDuplicates(['race_id', 'driver_id'])
results_deduped_df: pyspark.sql.dataframe.DataFrame = [result_id: integer, race_id: integer ... 16 more fields]
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 13
Python
1 results_deduped_df.show(truncate=False)
(2) Spark Jobs
result_id	race_id	driver_id	constructor_id	number	grid	position	position_text	position_order	points	laps	time	milliseconds	fastest_lap	rank	fastest_lap_time	fastest_lap_speed	ingestion_date
7573	1	1	1	18	null	D	20	0.0	58	\N	null	39	13	1:29.020	214.455	2024-04-23 04:09:58.81
7563	1	2	2	6	9	10	10	0.0	58	+7.085	5662869	48	5	1:28.283	216.245	2024-04-23 04:09:58.81
Search
9:40 AM
4/23/2024

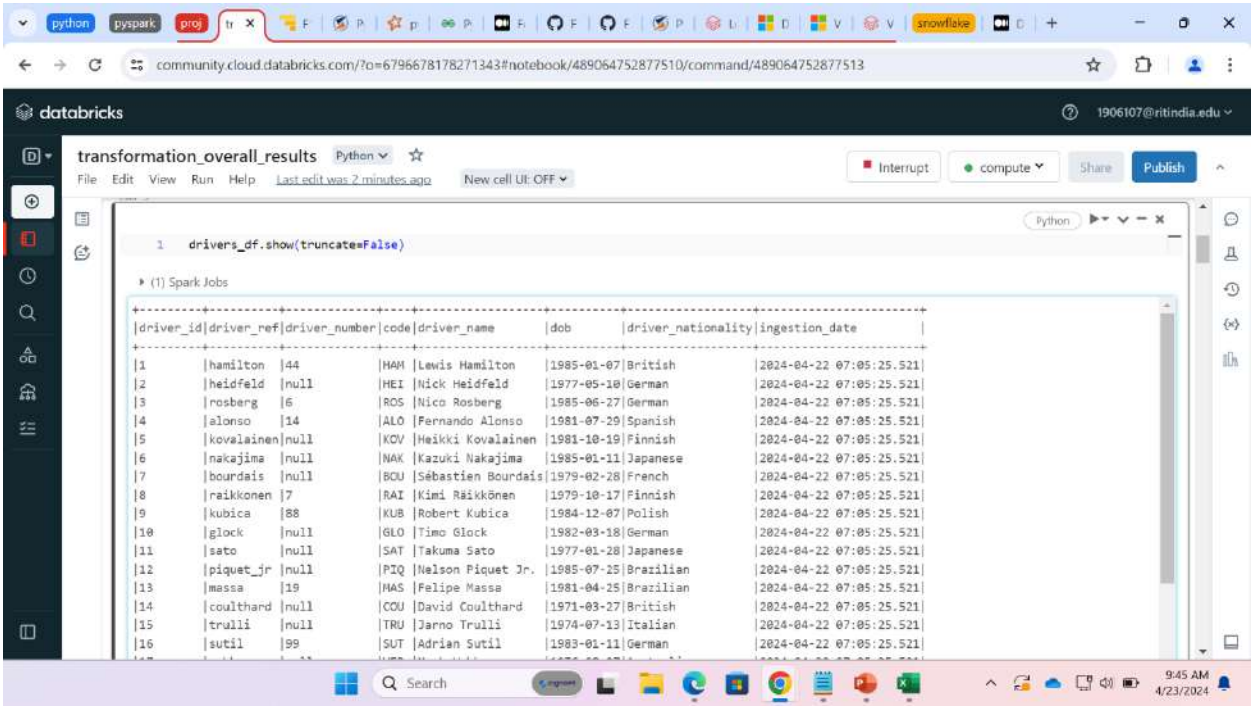

community.cloud.databricks.com/?o=6796678178271343#notebook/2747915330769745/command/2747915330769759
databricks
1906107@ritindia.edu
result processed
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
compute
Share
Publish
1:29.823	212.537	2024-04-23 04:09:58.81
7568 |1 |8 |6 |4 |7 |15 |15 |0.0 |55 |\N |null |35 |7
1:28.488	215.744	2024-04-23 04:09:58.81
7567 |1 |9 |2 |5 |4 |14 |14 |14 |0.0 |55 |\N |null |36 |2
1:27.988	216.97	2024-04-23 04:09:58.81
7562 |1 |16 |10 |20 |16 |9 |9 |9 |0.0 |58 |+6.335 |5662119 |43 |11
Command took 7.38 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:39:58 AM on compute
Cmd 14
1 #Step 4 - Write to output to processed container in parquet format
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
Cmd 15
Python
1 results_deduped_df.write.parquet("dbfs:/FileStore/Formula1/processed/f1_processed.result")
(1) Spark Jobs
AnalysisException: Path dbfs:/FileStore/Formula1/processed/f1_processed.result already exists.
Command took 15.37 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:38:15 AM on compute
[Shift+Enter] to run and move to next cell
[Esc H] to see all keyboard shortcuts
Search
9:40 AM
4/23/2024

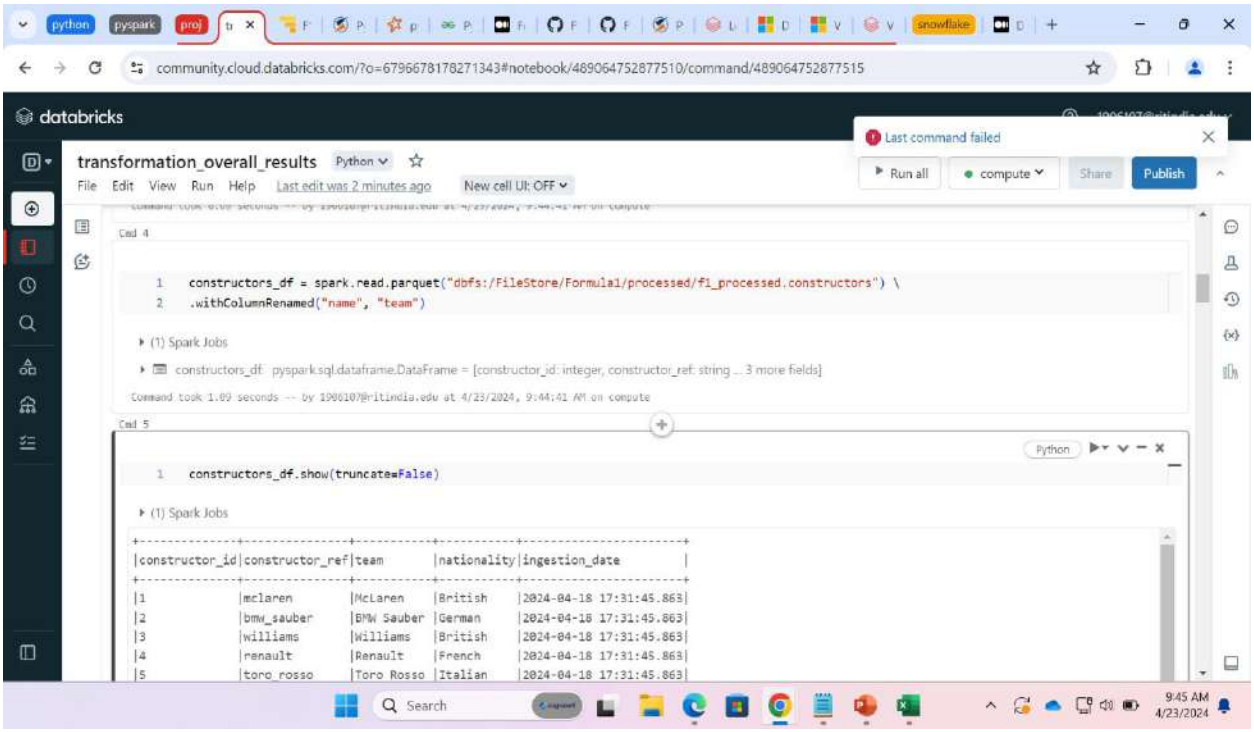
- # REQUIREMENT 2:

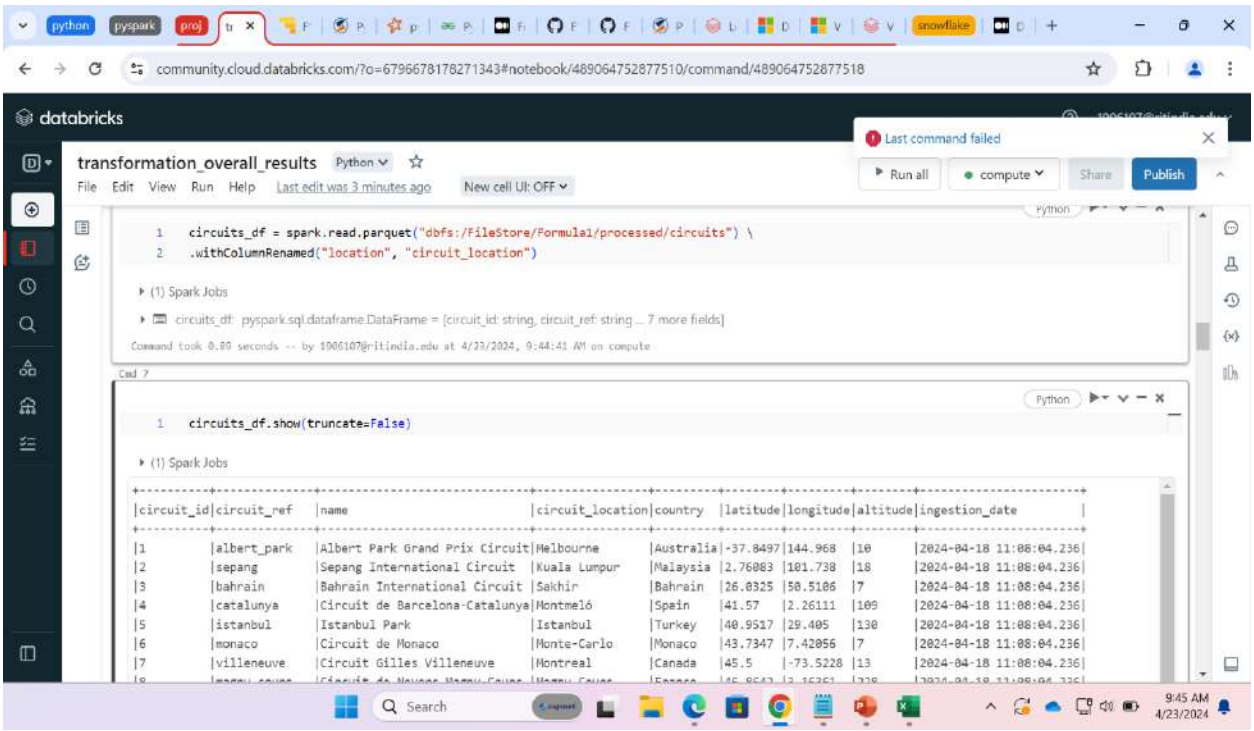
Data Transformations

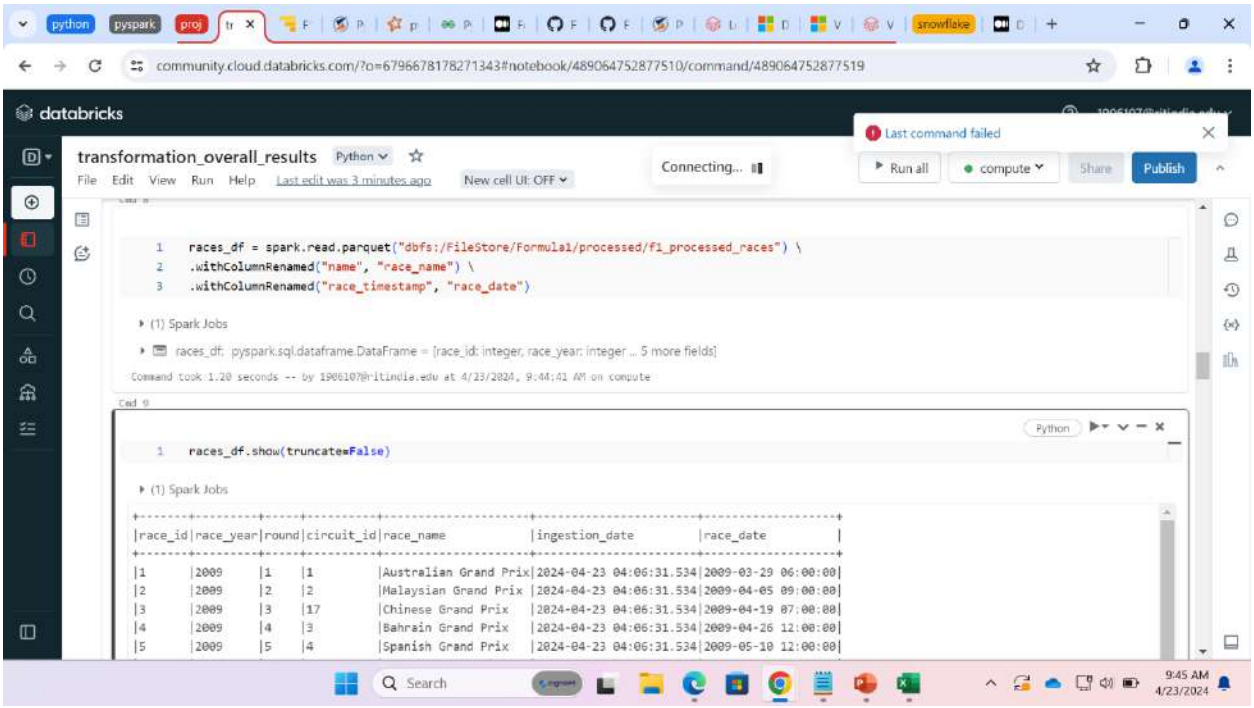
Step 1: Read all the parquet files from DBFS file store using the spark.read.parquet command

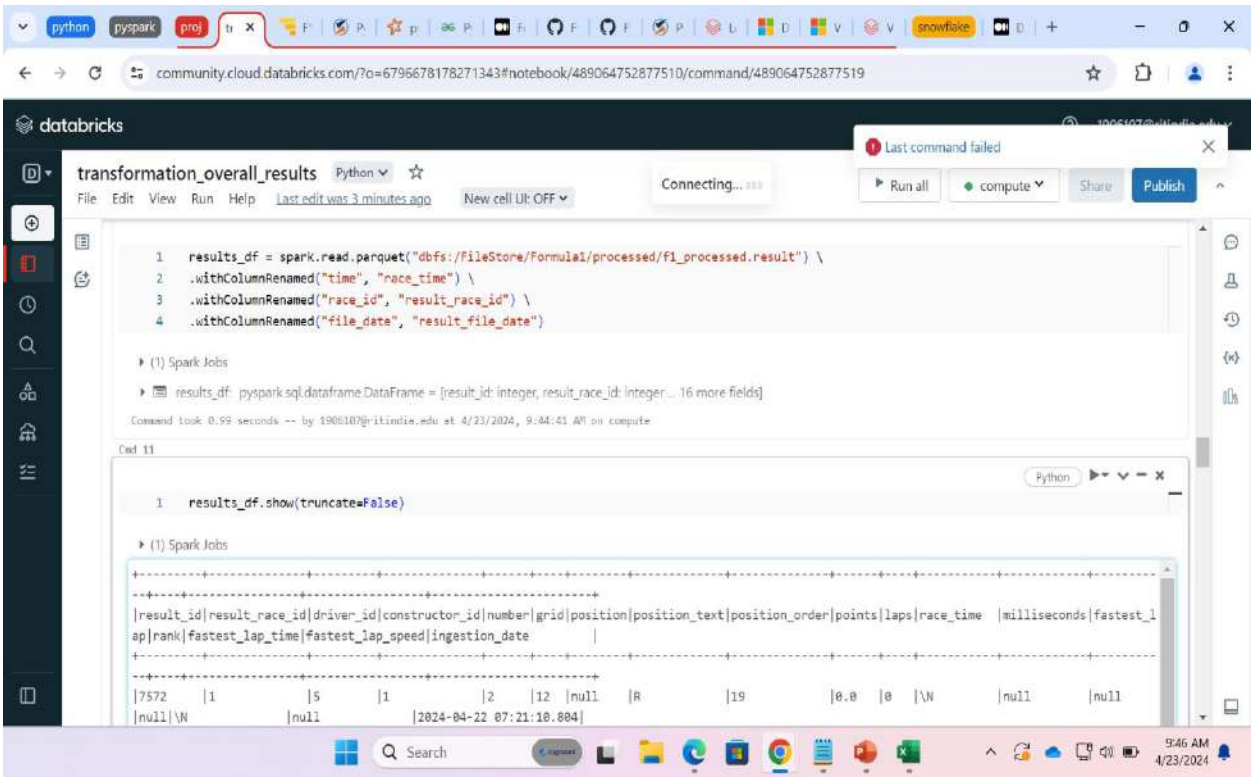




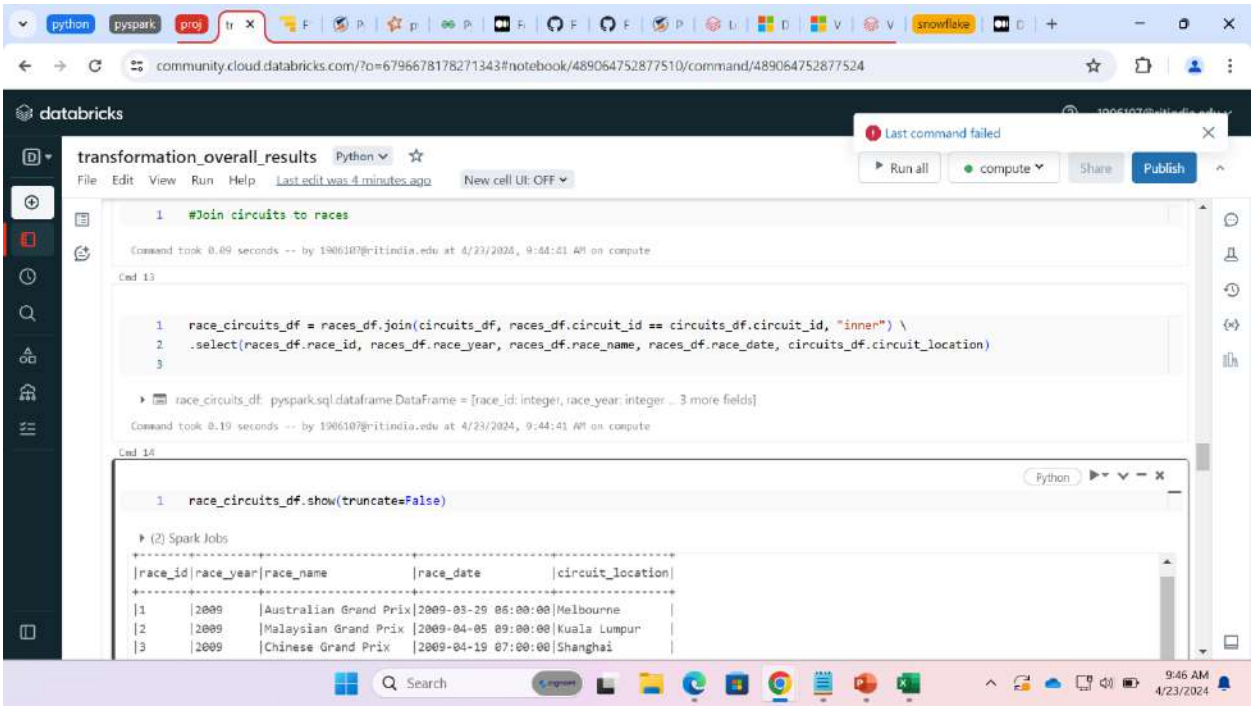

community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877510/command/489064752877515
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 2 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
Cmd 4
constructors_df = spark.read.parquet("dbfs:/FileStore/Formula1/processed/f1_processed.constructors") \
.withColumnRenamed("name", "team")
(1) Spark Jobs
constructors_df: pyspark.sql.dataframe.DataFrame = [constructor_id: integer, constructor_ref: string ... 3 more fields]
Command took 1.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 5
constructors_df.show(truncate=False)
(1) Spark Jobs
constructor_id	constructor_ref	team	nationality	ingestion_date
1	mclaren	McLaren	British	2024-04-18 17:31:45.863
2	bmw_sauber	BMW Sauber	German	2024-04-18 17:31:45.863
3	williams	Williams	British	2024-04-18 17:31:45.863
4	renault	Renault	French	2024-04-18 17:31:45.863
5	toro_rosso	Toro Rosso	Italian	2024-04-18 17:31:45.863
Search
9:45 AM
4/23/2024

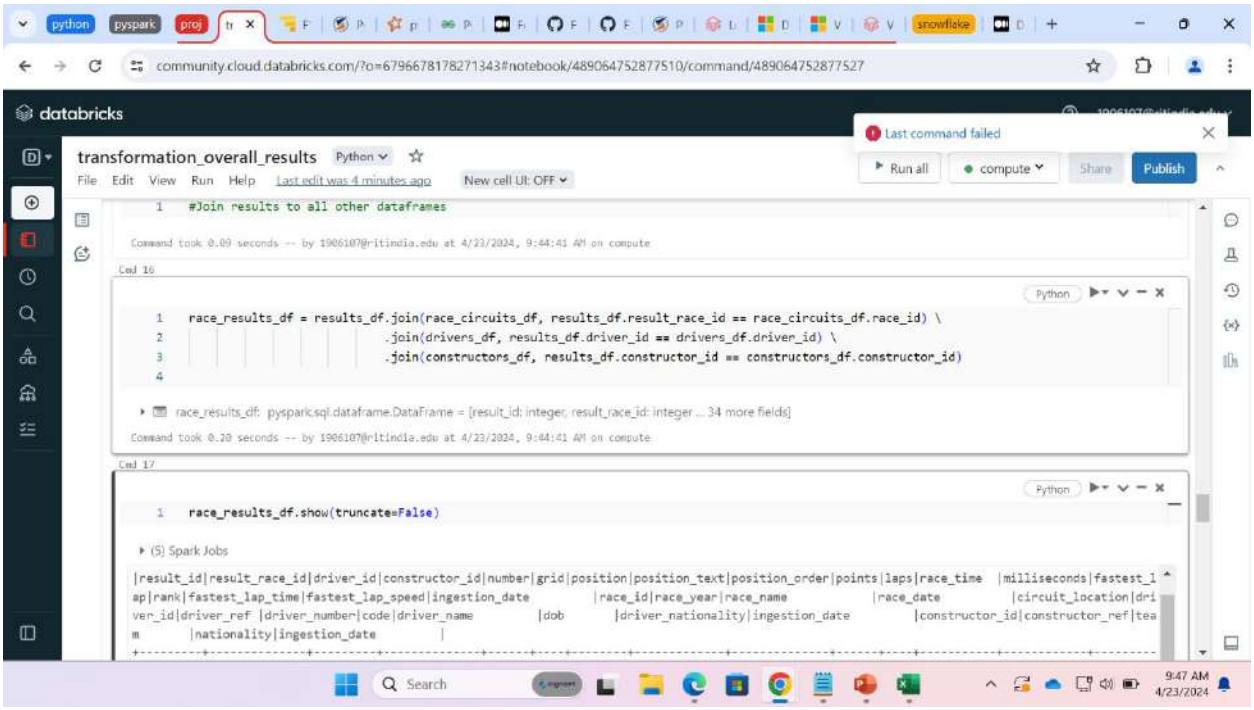

community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877510/command/489064752877518
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 3 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
circuits_df = spark.read.parquet("dbfs:/FileStore/Formula1/processed/circuits") \
.withColumnRenamed("location", "circuit_location")
(1) Spark Jobs
circuits_df: pyspark.sql.dataframe.DataFrame = [circuit_id: string, circuit_ref: string ... 7 more fields]
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 7
circuits_df.show(truncate=False)
(1) Spark Jobs
circuit_id	circuit_ref	name	circuit_location	country	latitude	longitude	altitude	ingestion_date
1	albert_park	Albert Park Grand Prix Circuit	Melbourne	Australia	-37.8497	144.968	10	2024-04-18 11:08:04.236
2	sepang	Sepang International Circuit	Kuala Lumpur	Malaysia	2.76083	101.738	18	2024-04-18 11:08:04.236
3	bahrain	Bahrain International Circuit	Sakhir	Bahrain	26.0325	50.5106	7	2024-04-18 11:08:04.236
4	catalunya	Circuit de Barcelona-Catalunya	Montmeló	Spain	41.57	2.26111	109	2024-04-18 11:08:04.236
5	istanbul	Istanbul Park	Istanbul	Turkey	40.9517	29.405	130	2024-04-18 11:08:04.236
6	monaco	Circuit de Monaco	Monte-Carlo	Monaco	43.7347	7.42056	7	2024-04-18 11:08:04.236
7	villeneuve	Circuit Gilles Villeneuve	Montreal	Canada	45.5	-73.5228	13	2024-04-18 11:08:04.236
Search
9:45 AM
4/23/2024

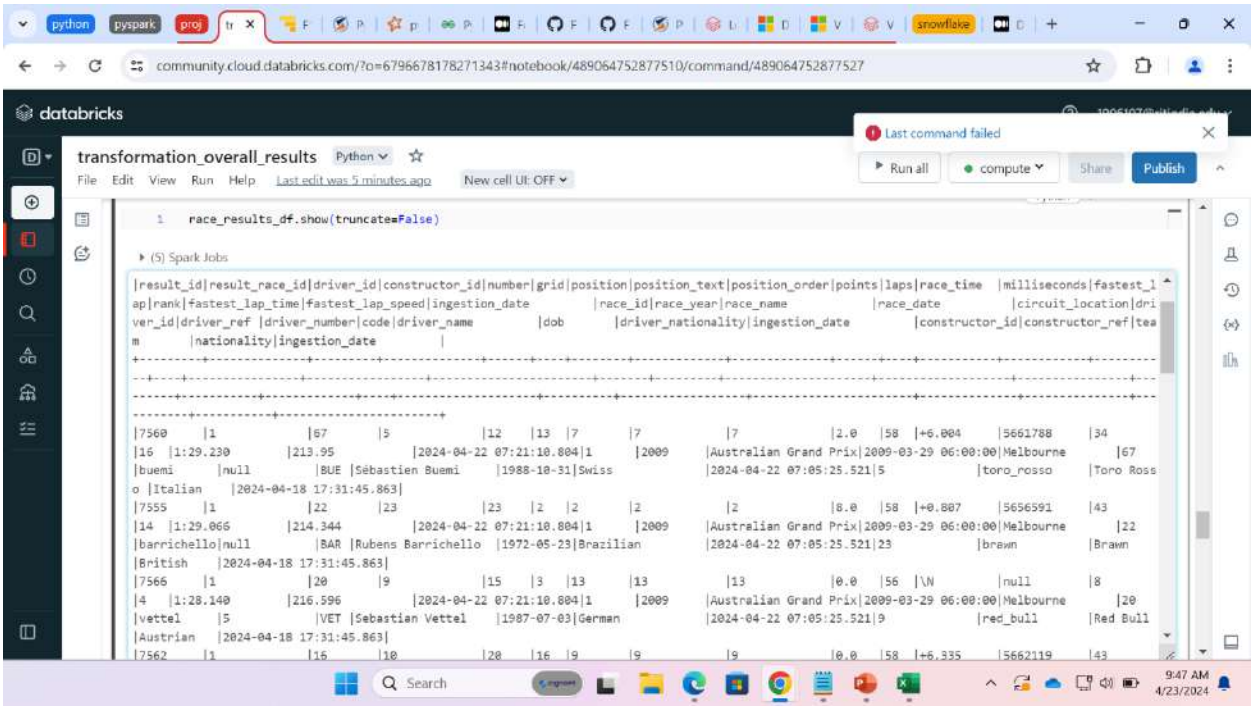


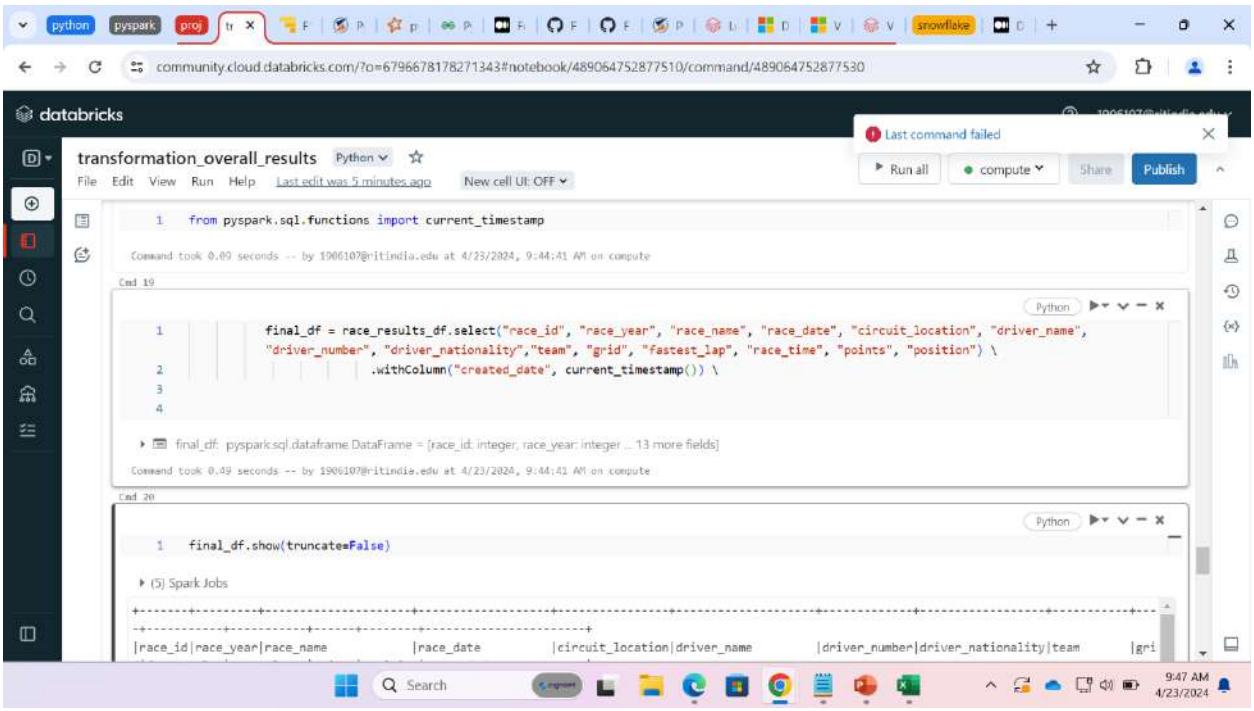


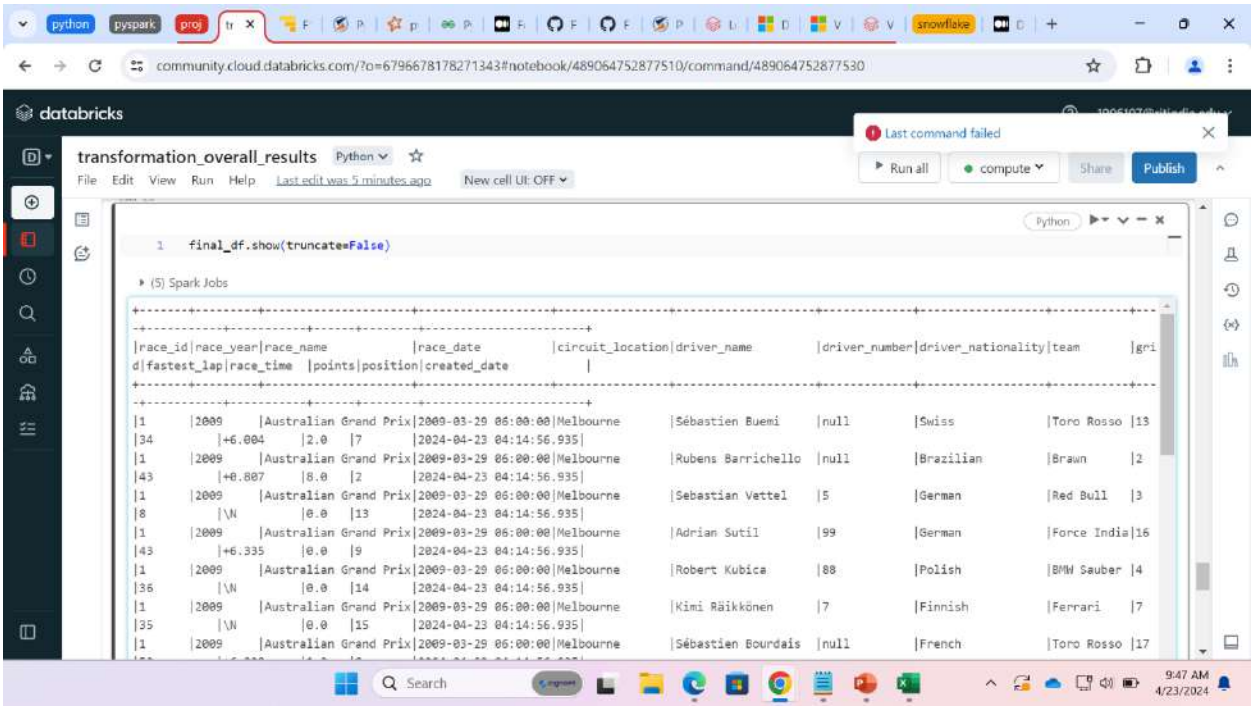
Step 2: Join the required tables for reporting and analysis requirements with only selected columns.


community.cloud.databricks.com/?o=6796678178271343#notebook/4890647528775l0/command/4890647528775l4
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 4 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
#Join circuits to races
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 13
race_circuits_df = races_df.join(circuits_df, races_df.circuit_id == circuits_df.circuit_id, "inner") \
.select(races_df.race_id, races_df.race_year, races_df.race_name, races_df.race_date, circuits_df.circuit_location)
race_circuits_df: pyspark.sql.dataframe.DataFrame = [race_id: integer, race_year: integer ... 3 more fields]
Command took 0.19 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 14
race_circuits_df.show(truncate=False)
(2) Spark Jobs
race_id	race_year	race_name	race_date	circuit_location
1	2009	Australian Grand Prix	2009-03-29 05:00:00	Melbourne
2	2009	Malaysian Grand Prix	2009-04-05 09:00:00	Kuala Lumpur
3	2009	Chinese Grand Prix	2009-04-19 07:00:00	Shanghai
Search
9:46 AM
4/23/2024

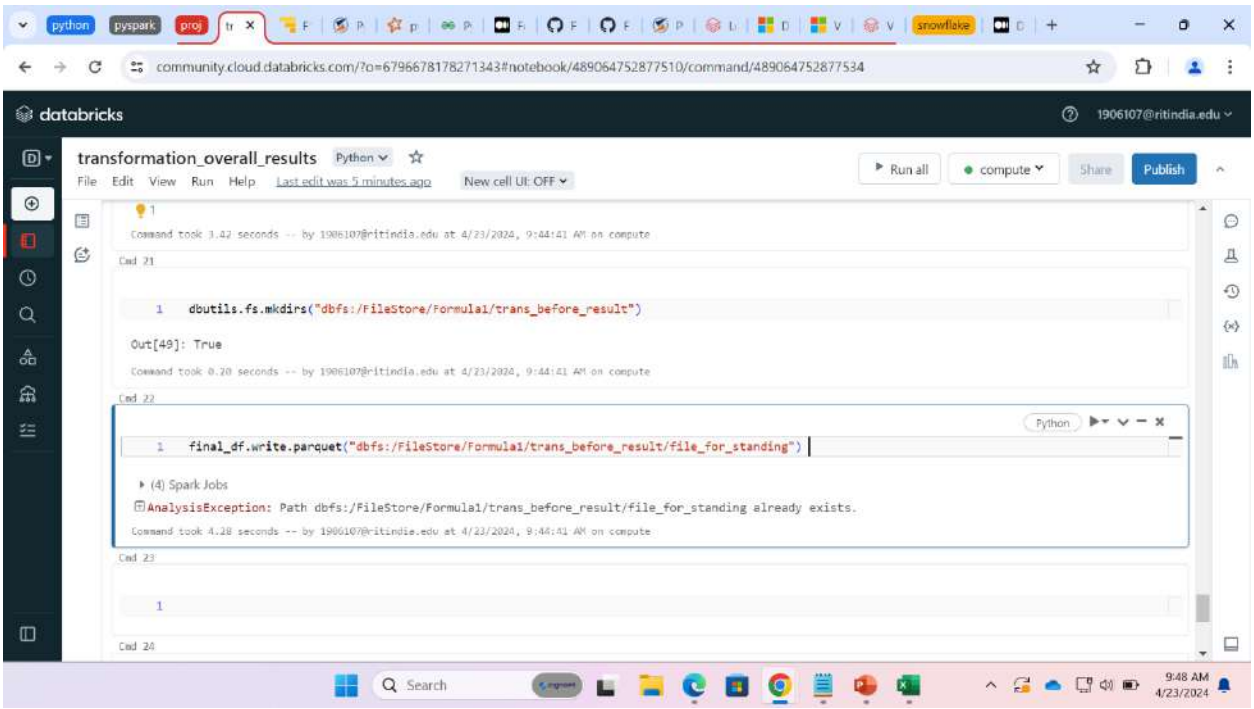

community.cloud.databricks.com/?o=6796678178271343#notebook/4890647528775l0/command/4890647528775l7
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 4 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
#Join results to all other dataframes
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 16
race_results_df = results_df.join(race_circuits_df, results_df.result_race_id == race_circuits_df.race_id) \
.join(drivers_df, results_df.driver_id == drivers_df.driver_id) \
.join(constructors_df, results_df.constructor_id == constructors_df.constructor_id)
race_results_df: pyspark.sql.dataframe.DataFrame = [result_id: integer, result_race_id: integer ... 34 more fields]
Command took 0.20 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 17
race_results_df.show(truncate=False)
(5) Spark Jobs
|result_id|result_race_id|driver_id|constructor_id|number|grid|position|position_text|position_order|points|laps|race_time |milliseconds|fastest_lap|rank|fastest_lap_time|fastest_lap_speed|ingestion_date |race_id|race_year|race_name |race_date |circuit_location|driver_id|driver_ref |driver_number|code|driver_name |dob |driver_nationality|ingestion_date |constructor_id|constructor_ref|team |nationality|ingestion_date |
Search
9:47 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877510/command/489064752877527
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 5 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
1 race_results_df.show(truncate=False)
(5) Spark Jobs
result_id	result_race_id	driver_id	constructor_id	number	grid	position	position_text	position_order	points	laps	race_time	milliseconds	fastest_lap	rank	fastest_lap_time	fastest_lap_speed	ingestion_date	race_id	race_year	race_name	race_date	circuit_location	driver_id	driver_ref	driver_number	code	driver_name	dob	driver_nationality	ingestion_date	constructor_id	constructor_ref	team	nationality	ingestion_date
7560	1	67	5	12	13	7	7	7	2.0	58	+6.004	5661788	34	16	1:29.230	213.95	2024-04-22 07:21:10.804	1	2009	Australian Grand Prix	2009-03-29 06:00:00	Melbourne	67	buemi	null	BUE	Sébastien Buemi	1988-10-31	Swiss	2024-04-22 07:05:25.521	5	toro_rosso	Toro Rosso	Italian	2024-04-18 17:31:45.863
7555	1	22	23	3	2	2	2	2	8.0	58	+0.807	5656591	43	14	1:29.066	214.344	2024-04-22 07:21:10.804	1	2009	Australian Grand Prix	2009-03-29 06:00:00	Melbourne	22	barrichello	null	BAR	Rubens Barrichello	1972-05-23	Brazilian	2024-04-22 07:05:25.521	23	brawn	Brawn	British	2024-04-18 17:31:45.863
7566	1	20	9	15	3	13	13	13	0.0	56	\N	null	8	4	1:28.140	216.596	2024-04-22 07:21:10.804	1	2009	Australian Grand Prix	2009-03-29 06:00:00	Melbourne	20	vettel	5	VET	Sebastian Vettel	1987-07-03	German	2024-04-22 07:05:25.521	9	red_bull	Red Bull	Austrian	2024-04-18 17:31:45.863
7562	1	16	10	20	16	9	9	9	0.0	58	+6.335	5662119	43
Search
9:47 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877510/command/489064752877530
databricks
transformation_overall_results
Python
File Edit View Run Help Last edit was 5 minutes ago
New cell UI: OFF
Last command failed
Run all
compute
Share
Publish
1 from pyspark.sql.functions import current_timestamp
Command took 0.09 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 19
Python
1 final_df = race_results_df.select("race_id", "race_year", "race_name", "race_date", "circuit_location", "driver_name",
"driver_number", "driver_nationality","team", "grid", "fastest_lap", "race_time", "points", "position") \
2 .withColumn("created_date", current_timestamp()) \
3
4
final_df: pyspark.sql.dataframe.DataFrame = [race_id: integer, race_year: integer ... 13 more fields]
Command took 0.49 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:44:41 AM on compute
Cmd 20
Python
1 final_df.show(truncate=False)
(5) Spark Jobs
|race_id|race_year|race_name |race_date |circuit_location|driver_name |driver_number|driver_nationality|team |gri
Search
9:47 AM
4/23/2024



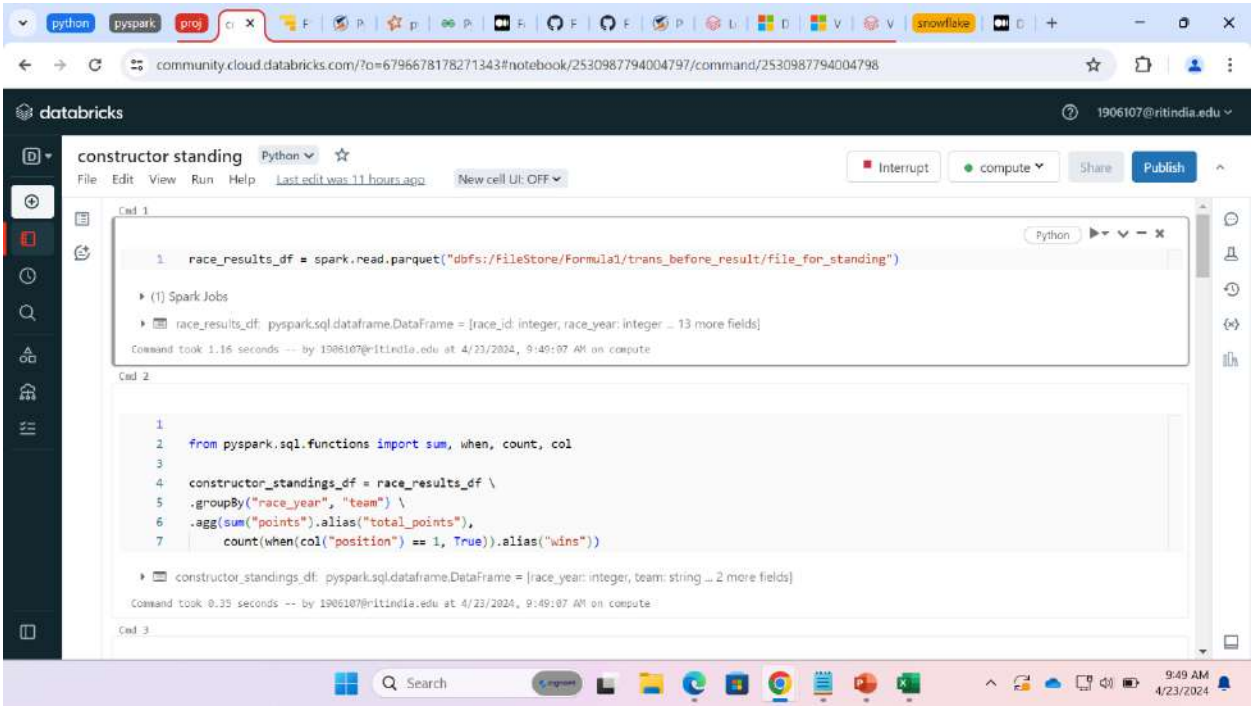
Step 3: Store the joined tables in columnar format (i.e. parquet files)

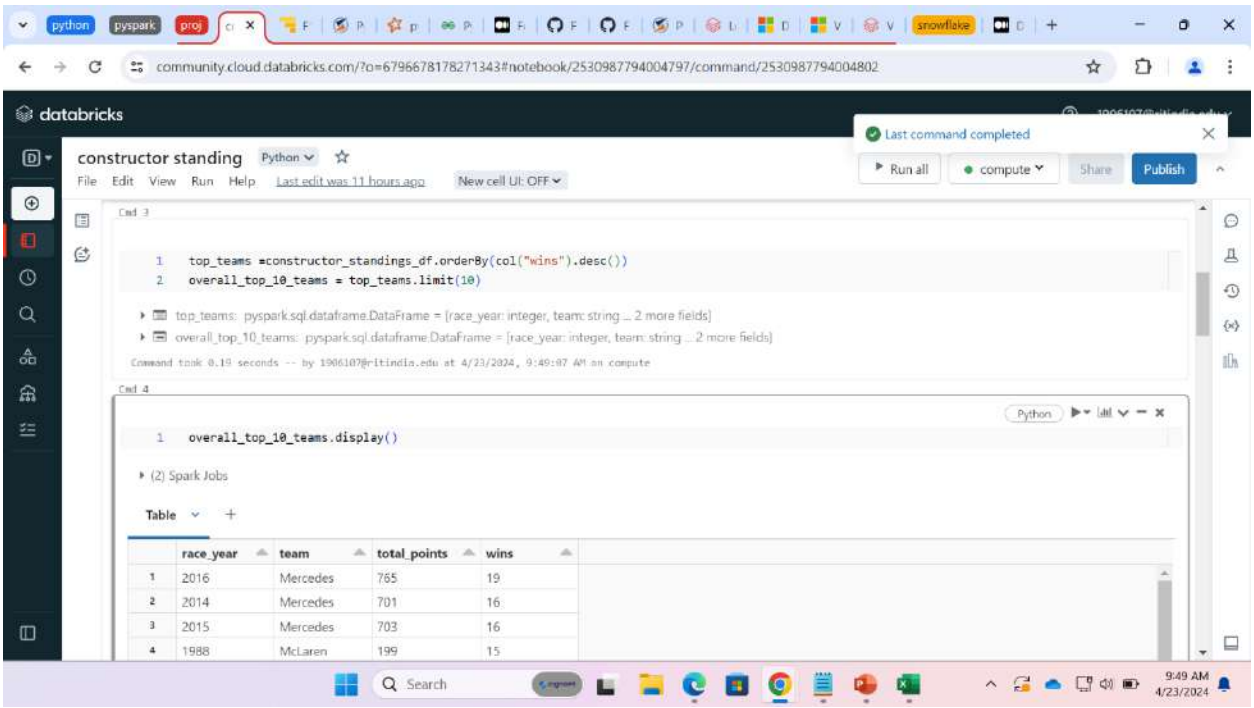


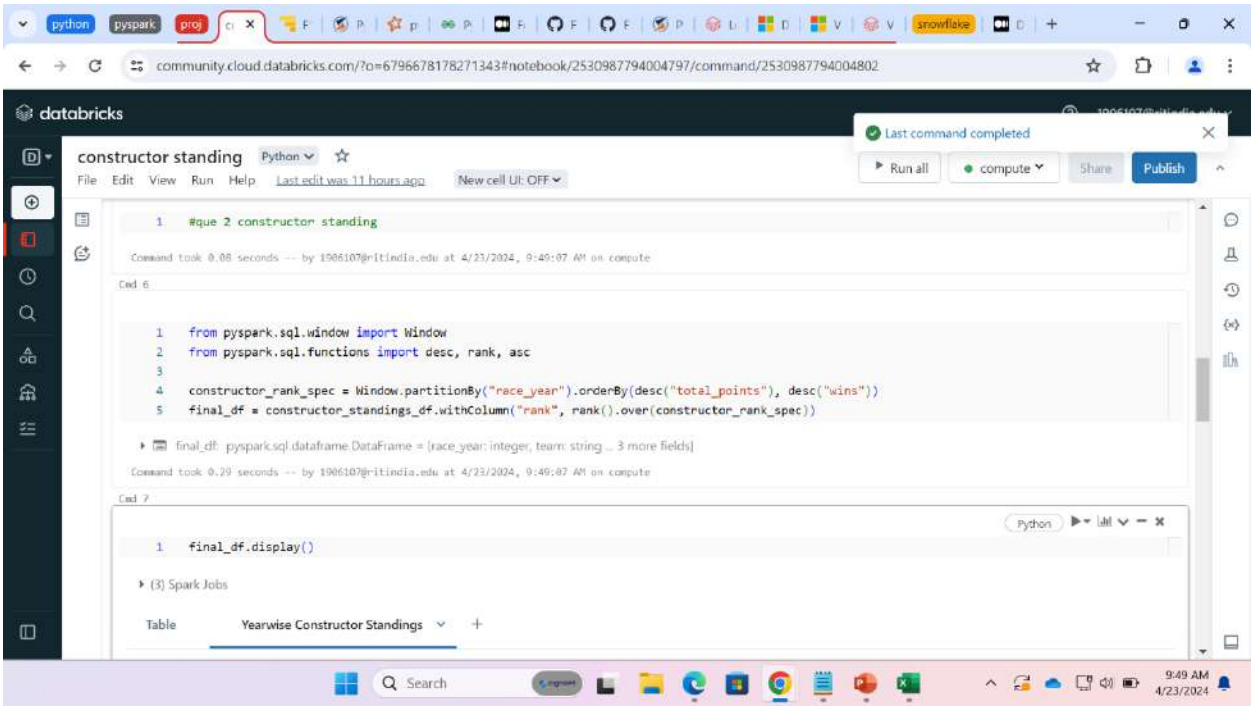
REQUIREMENT 3 & 4:

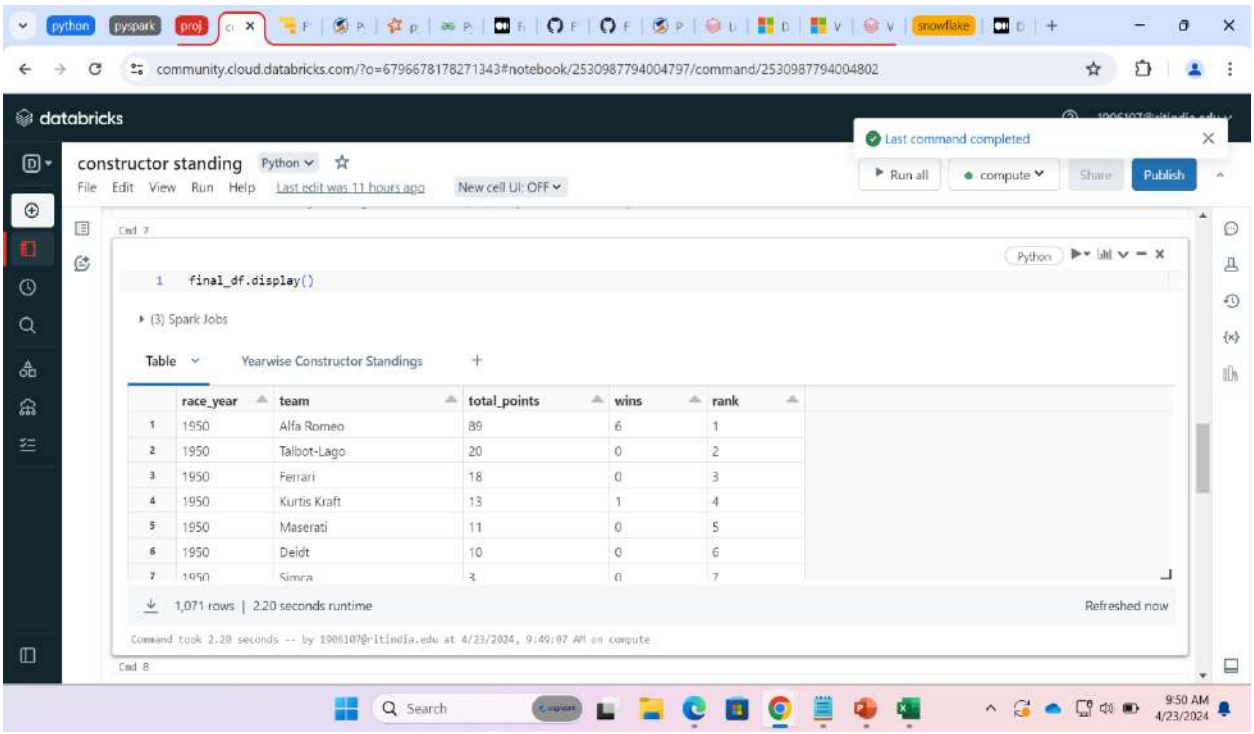
Reporting, Analysis and Visualization

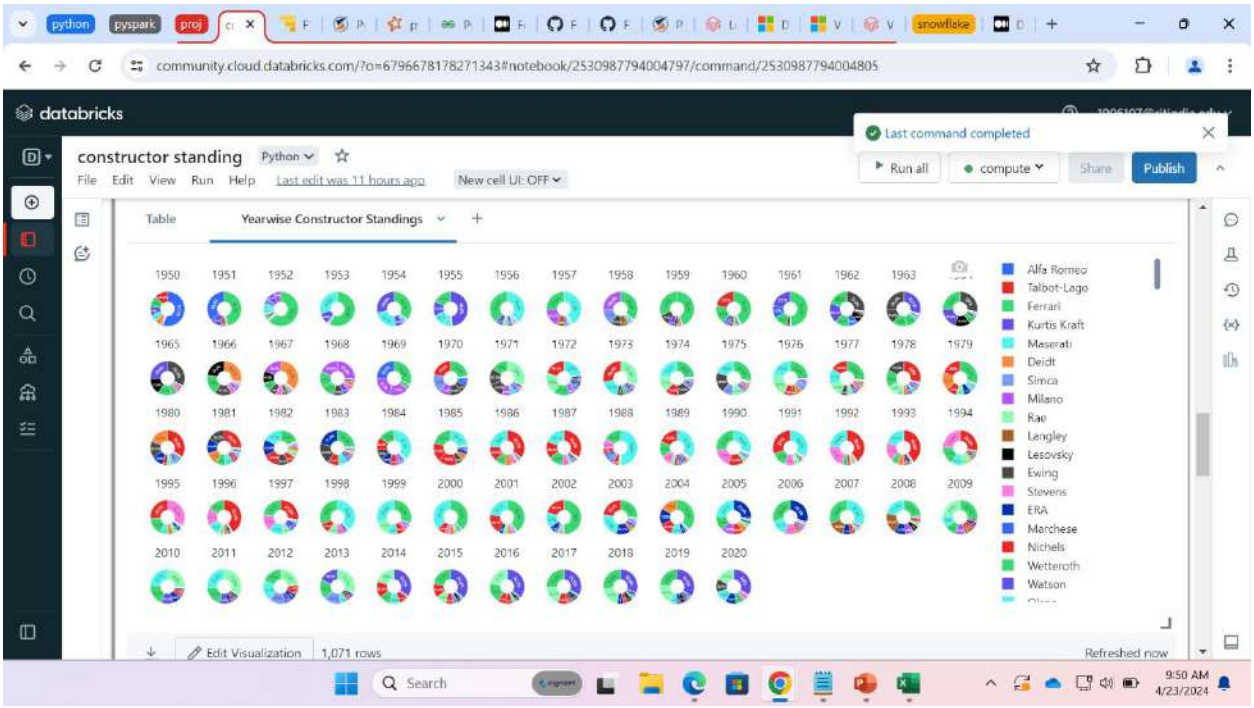
1. Constructor standings and dominant team.

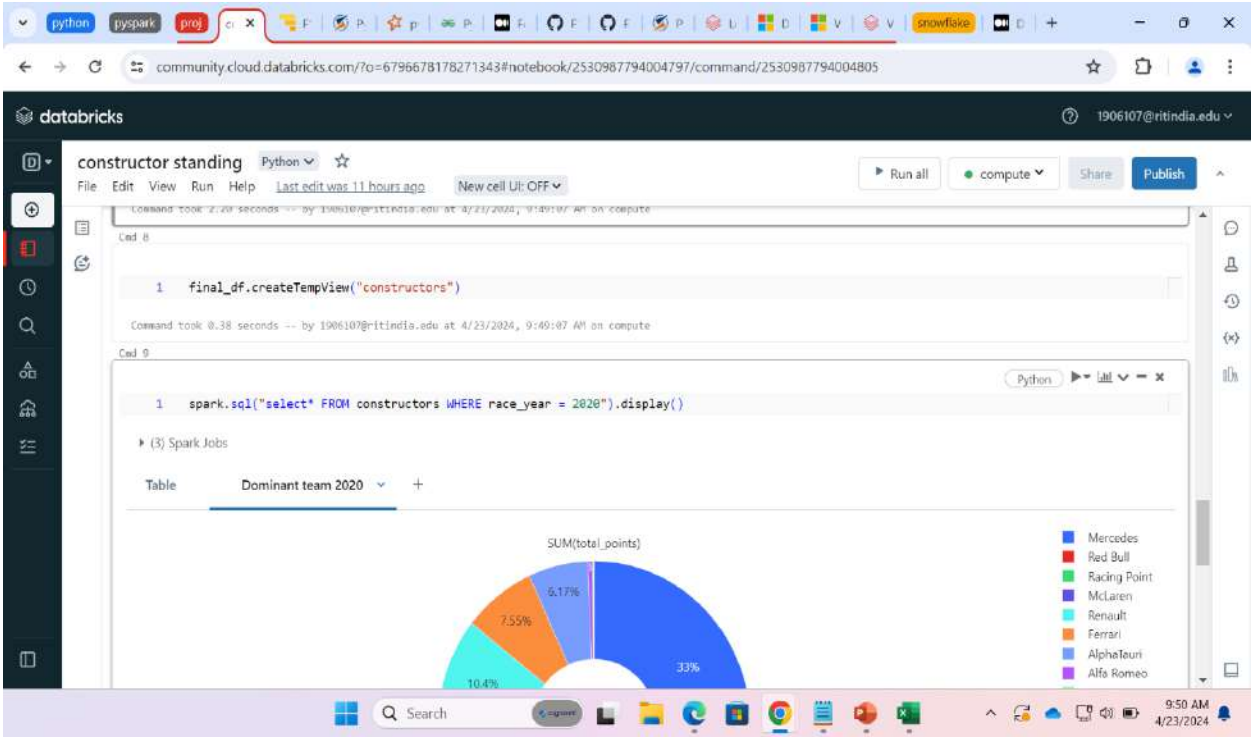


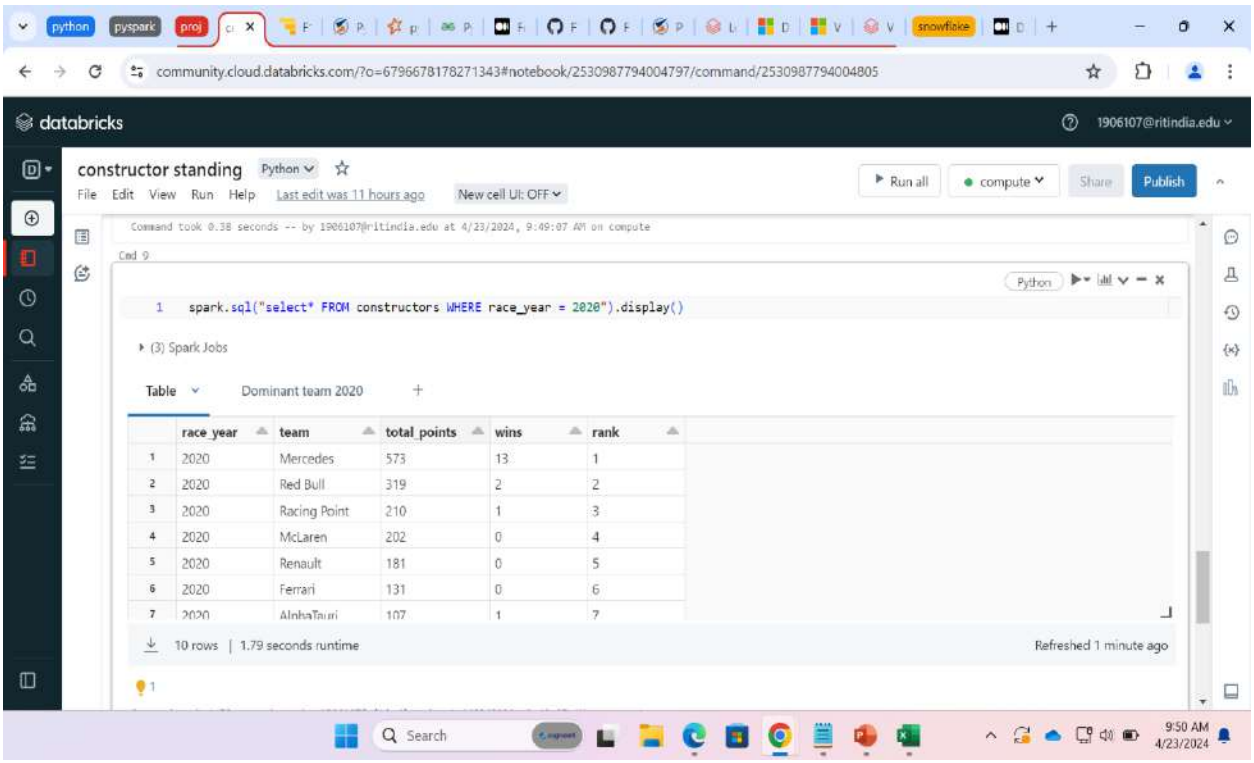


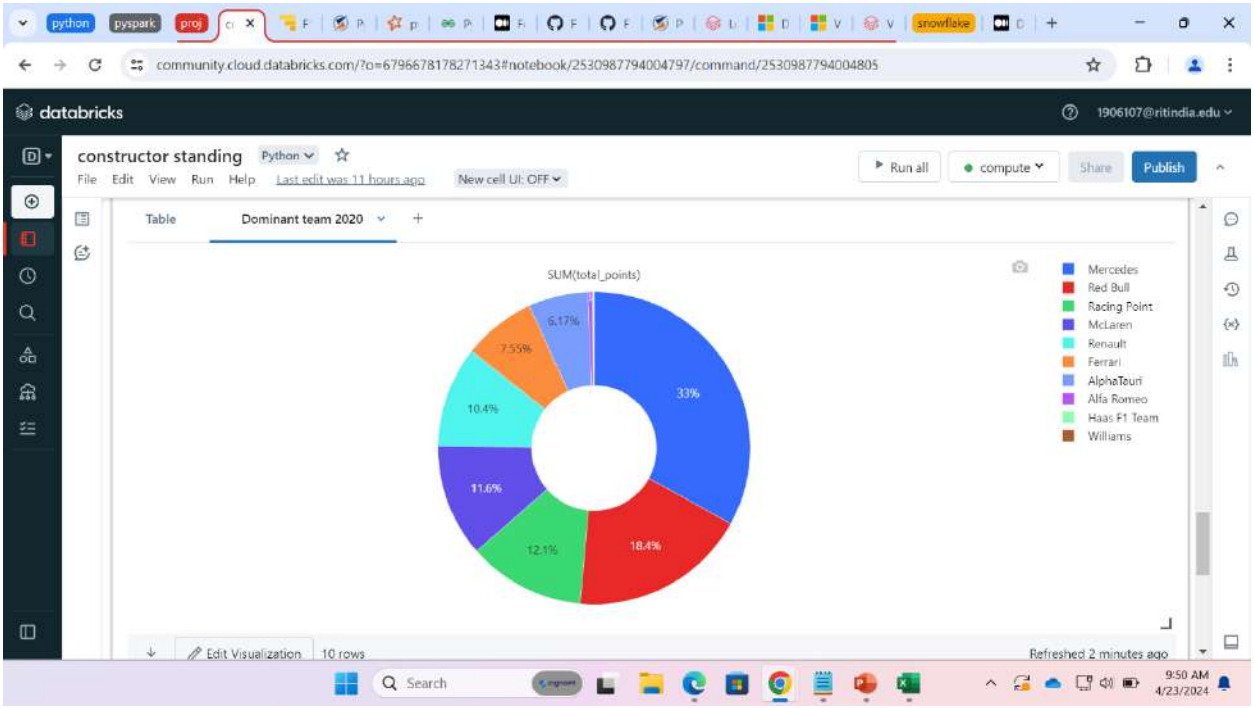

community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004802
databricks
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago
New cell UI: OFF
Last command completed
Run all
compute
Share
Publish
#que 2 constructor standing
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:49:07 AM on compute
Cmd 6
from pyspark.sql.window import Window
from pyspark.sql.functions import desc, rank, asc
constructor_rank_spec = Window.partitionBy("race_year").orderBy(desc("total_points"), desc("wins"))
final_df = constructor_standings_df.withColumn("rank", rank().over(constructor_rank_spec))
final_df: pyspark.sql.dataframe.DataFrame = [race_year: integer, team: string ... 3 more fields]
Command took 0.29 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:49:07 AM on compute
Cmd 7
final_df.display()
(3) Spark Jobs
Table
Yearwise Constructor Standings
Search
9:49 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004802
databricks
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago
New cell UI: OFF
Last command completed
Run all
compute
Share
Publish
Cmd 7
final_df.display()
(3) Spark Jobs
Table
Yearwise Constructor Standings
race_year team total_points wins rank
1 1950 Alfa Romeo 89 6 1
2 1950 Talbot-Lago 20 0 2
3 1950 Ferrari 18 0 3
4 1950 Kurtis Kraft 13 1 4
5 1950 Maserati 11 0 5
6 1950 Deidt 10 0 6
7 1950 Simca 3 0 7
1,071 rows | 2.20 seconds runtime
Refreshed now
Command took 2.20 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:49:07 AM on compute
Cmd 8
Search
9:50 AM
4/23/2024

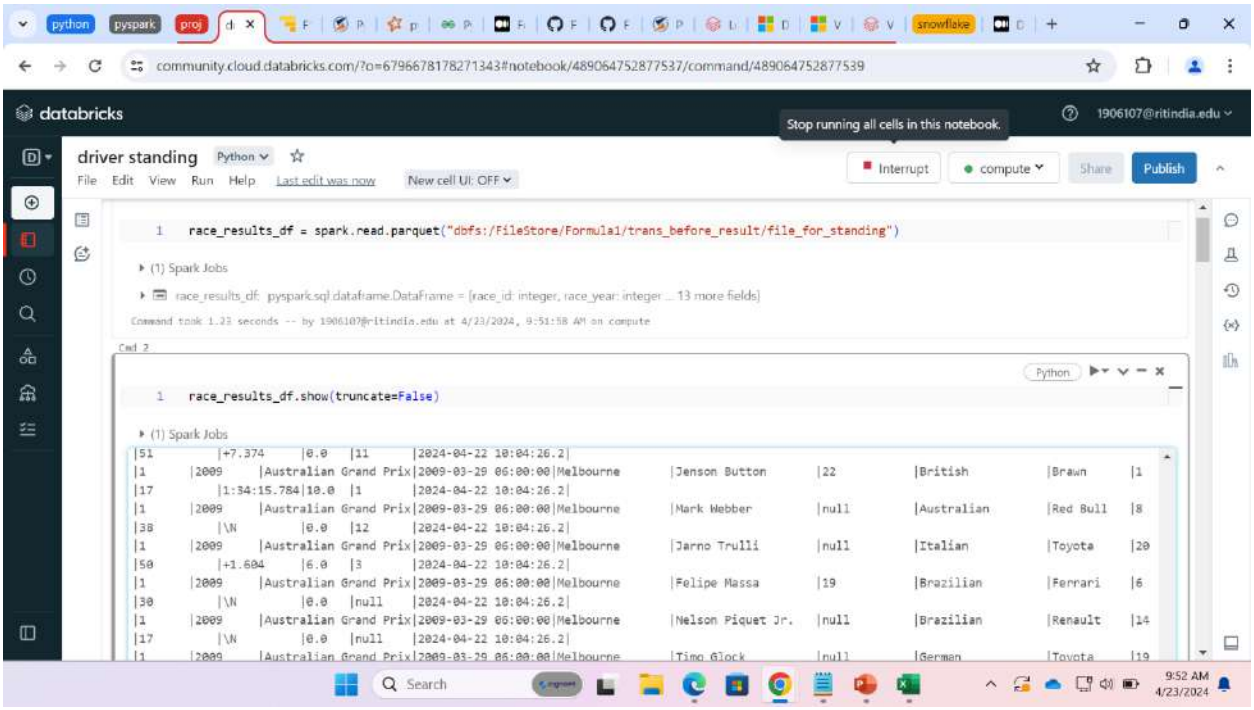

community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004805
databricks
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago
New cell UI: OFF
Last command completed
Run all
compute
Share
Publish
Table
Yearwise Constructor Standings
Alfa Romeo
Talbot-Lago
Ferrari
Kurtis Kraft
Maserati
Deidt
Simca
Milano
Rae
Langley
Lesovsky
Ewing
Stevens
ERA
Marchese
Nichels
Wetteroth
Watson
Edit Visualization
1,071 rows
Refreshed now
Search
9:50 AM
4/23/2024

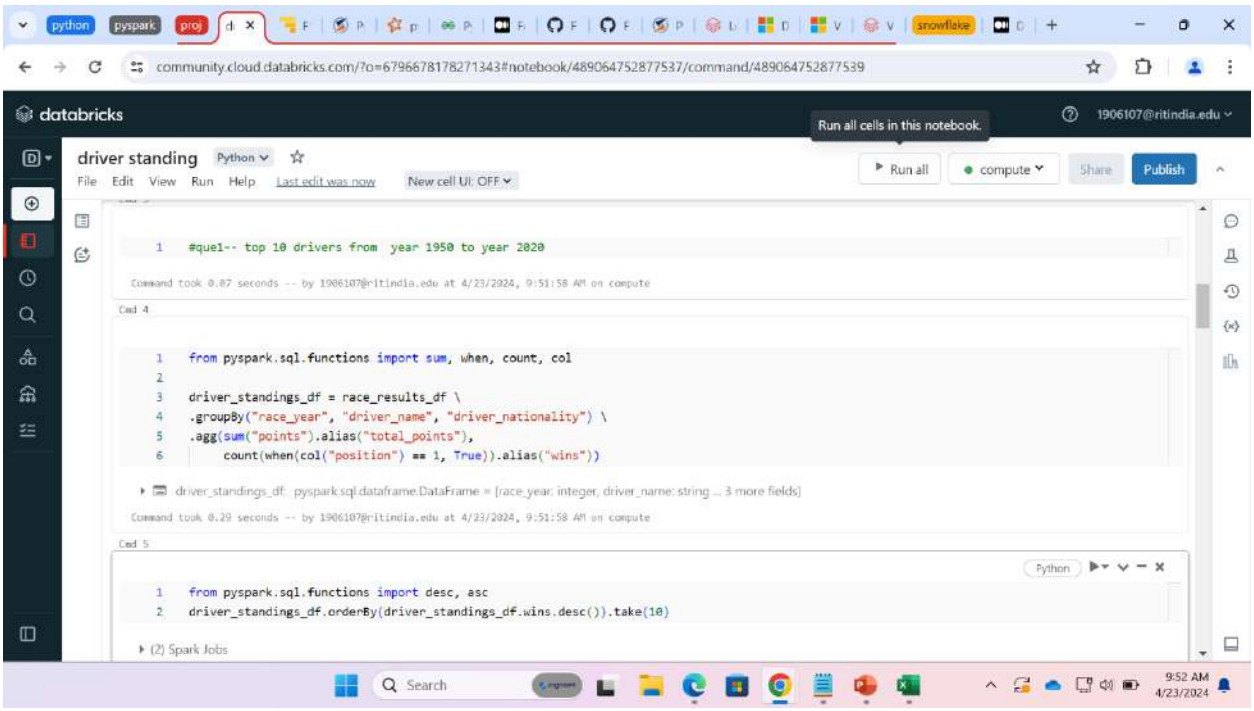

community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004805
databricks
1906107@ritindia.edu
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago
New cell UI: OFF
Run all
compute
Share
Publish
Cmd 8
1 final_df.createTempView("constructors")
Command took 0.38 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:49:07 AM on compute
Cmd 9
1 spark.sql("select* FROM constructors WHERE race_year = 2020").display()
(3) Spark Jobs
Table
Dominant team 2020
SUM(total_points)
33%
10.4%
7.55%
6.17%
Mercedes
Red Bull
Racing Point
McLaren
Renault
Ferrari
AlphaTauri
Alfa Romeo
Search
9:50 AM
4/23/2024

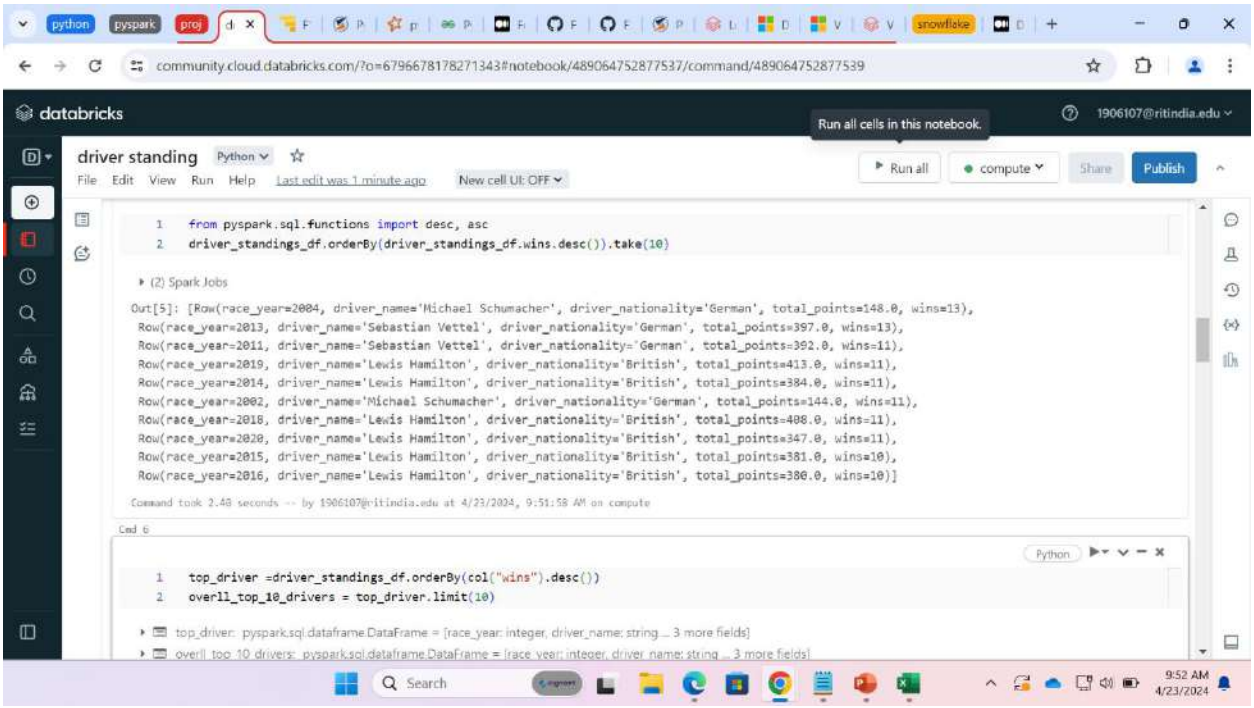

community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004805
databricks
1906107@ritindia.edu
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago New cell UI: OFF
Run all
compute
Share
Publish
Command took 0.38 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:49:07 AM on compute
Cmd 9
spark.sql("select* FROM constructors WHERE race_year = 2020").display()
(3) Spark Jobs
Table
Dominant team 2020
race_year team total_points wins rank
1 2020 Mercedes 573 13 1
2 2020 Red Bull 319 2 2
3 2020 Racing Point 210 1 3
4 2020 McLaren 202 0 4
5 2020 Renault 181 0 5
6 2020 Ferrari 131 0 6
7 2020 AlphaTauri 107 1 7
10 rows | 1.79 seconds runtime
Refreshed 1 minute ago
Search
9:50 AM
4/23/2024

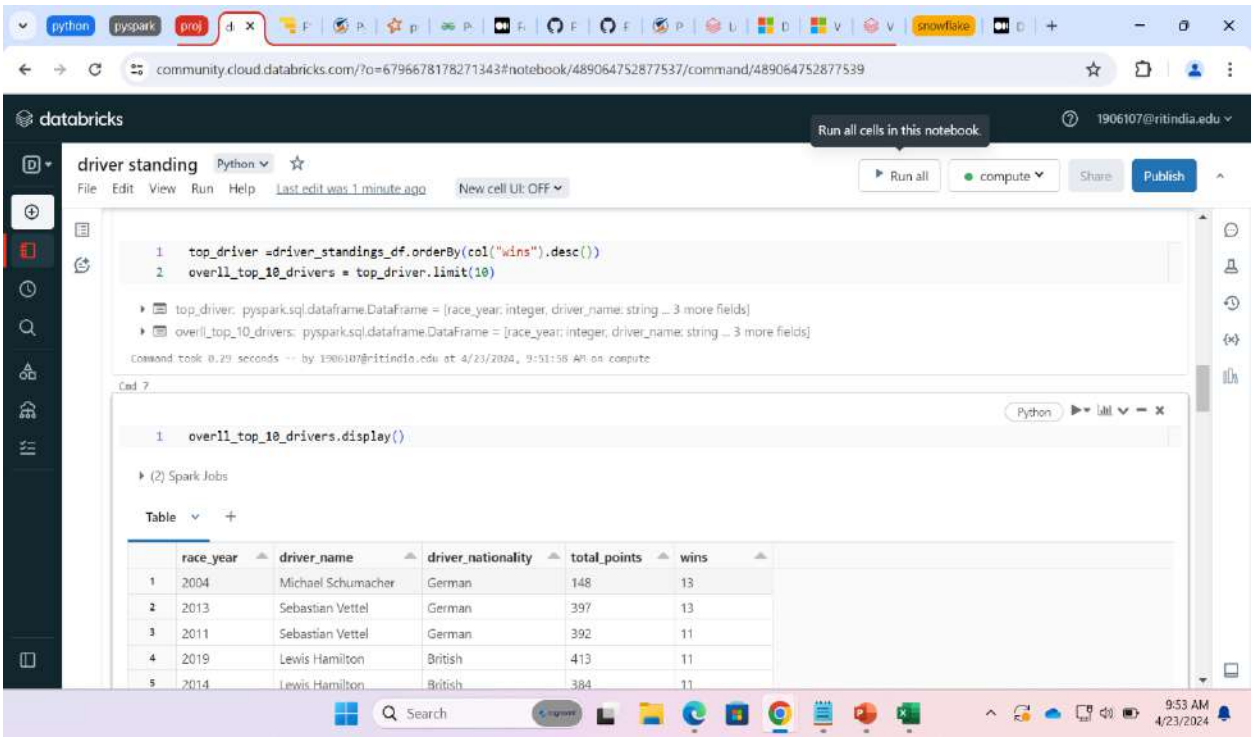

community.cloud.databricks.com/?o=6796678178271343#notebook/2530987794004797/command/2530987794004805
databricks
1906107@ritindia.edu
constructor standing
Python
File Edit View Run Help Last edit was 11 hours ago New cell UI: OFF
Run all
compute
Share
Publish
Table
Dominant team 2020
SUM(total_points)
33%
18.4%
12.1%
11.6%
10.4%
7.55%
6.17%
Mercedes
Red Bull
Racing Point
McLaren
Renault
Ferrari
AlphaTauri
Alfa Romeo
Haas F1 Team
Williams
Edit Visualization
10 rows
Refreshed 2 minutes ago
Search
9:50 AM
4/23/2024

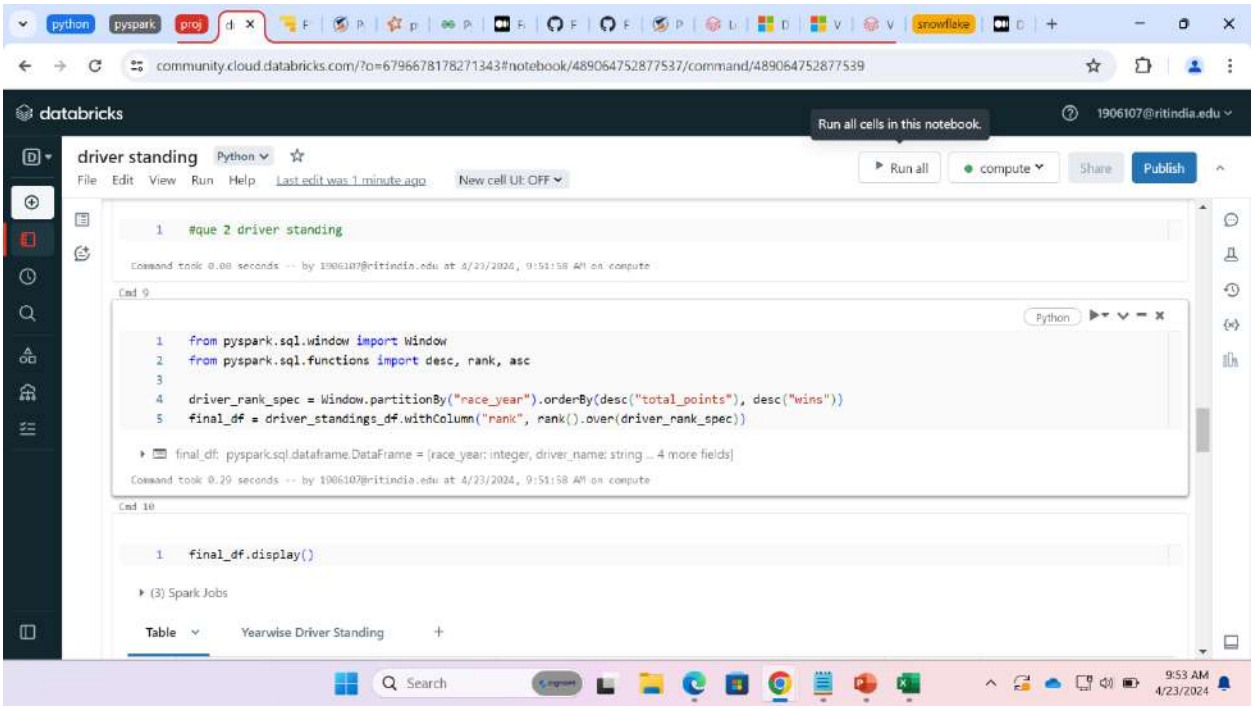
2. Driver standing and dominant driver.

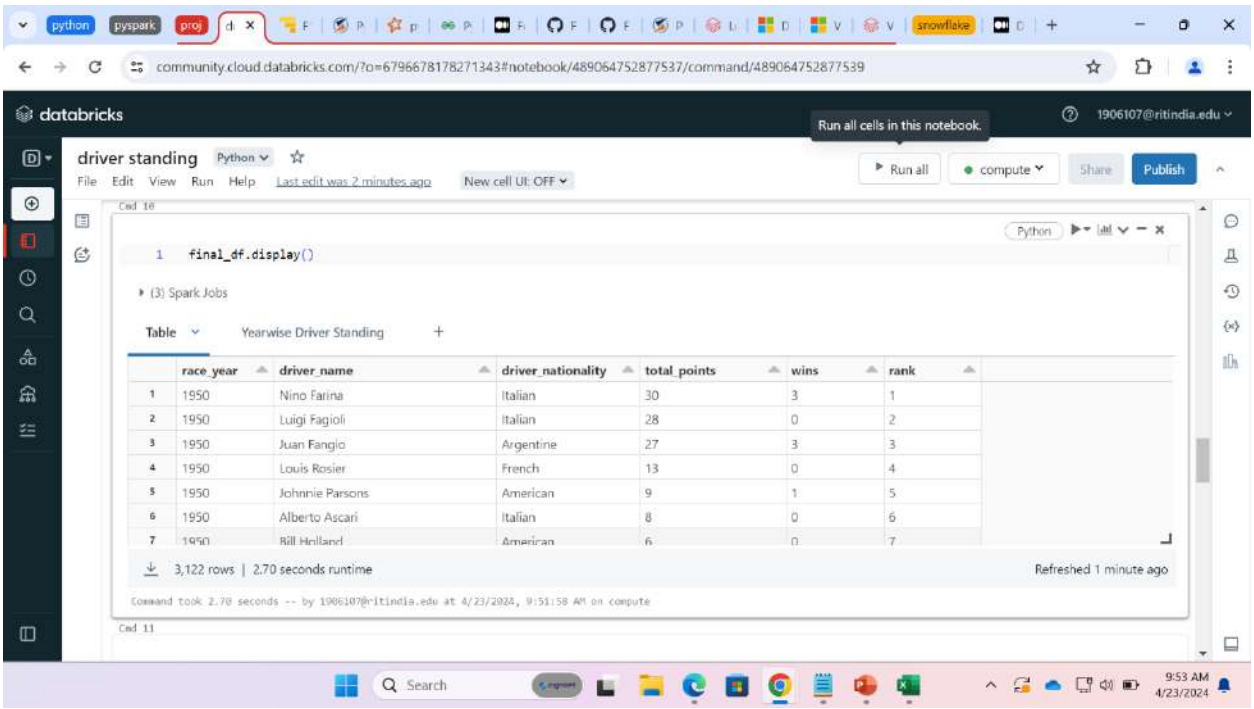


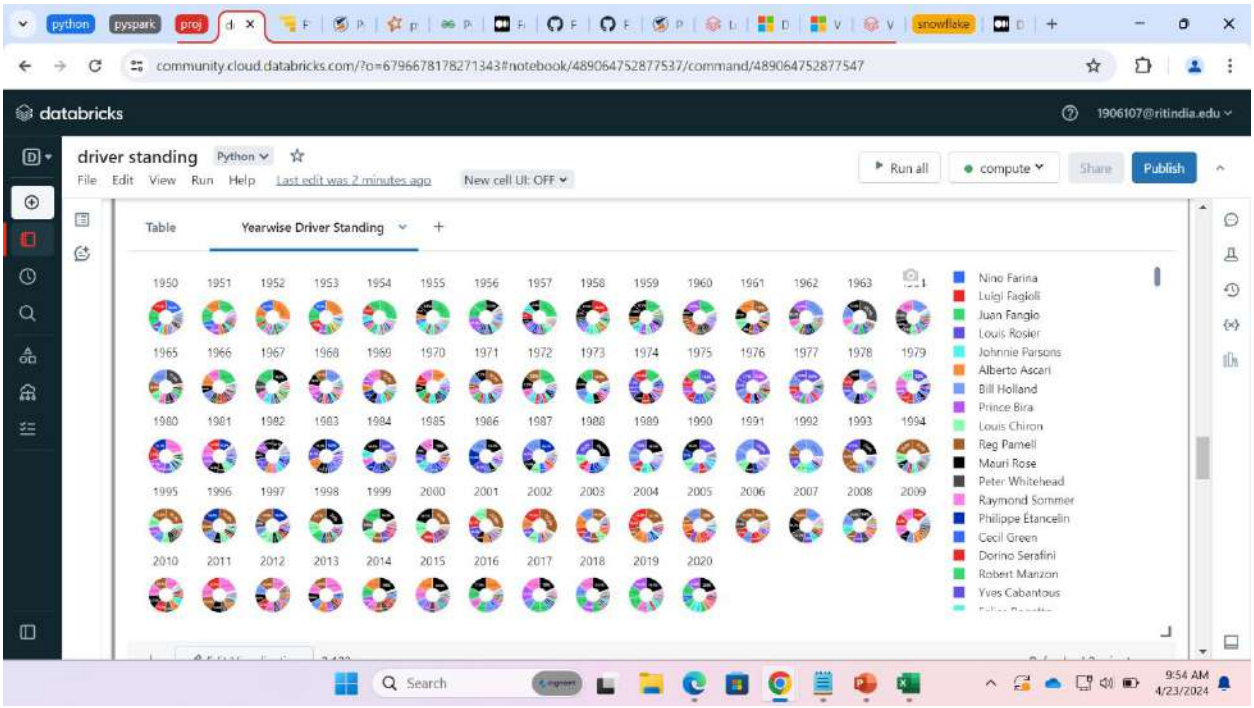


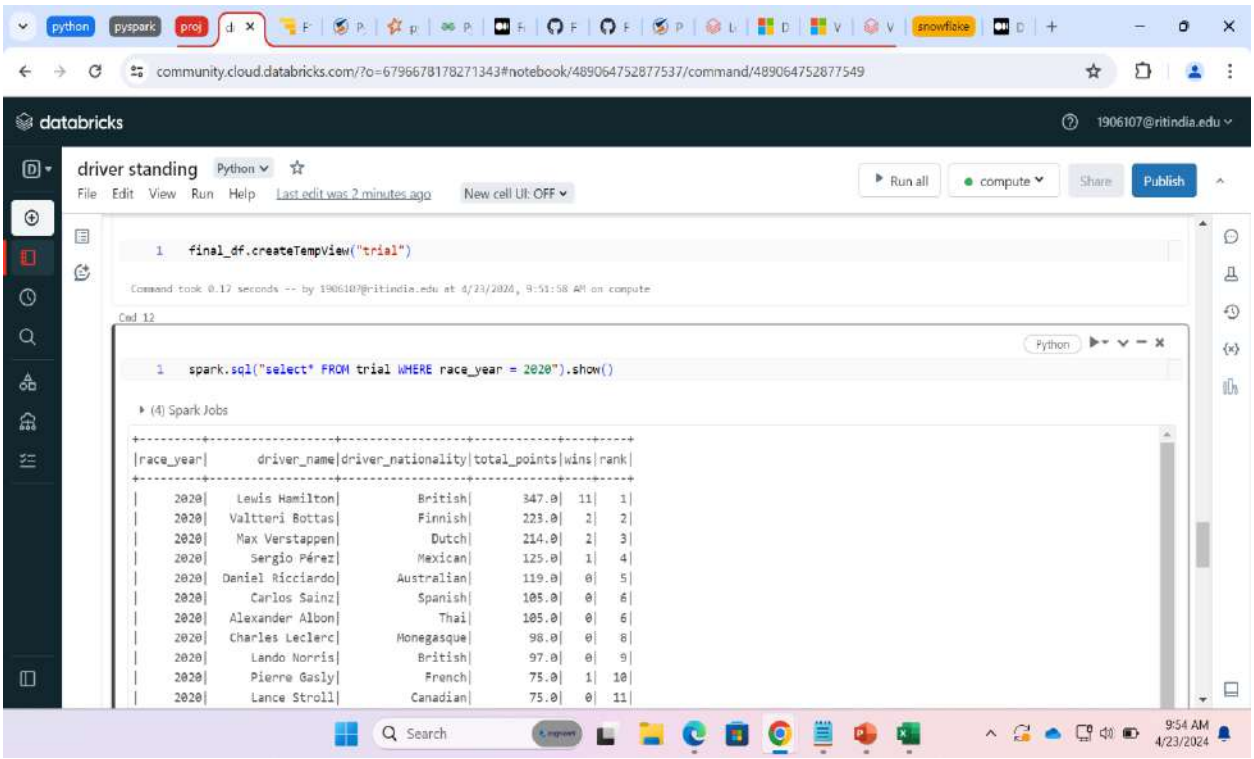

community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877539
databricks
1906107@ritindia.edu
Run all cells in this notebook.
driver standing
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
compute
Share
Publish
1 from pyspark.sql.functions import desc, asc
2 driver_standings_df.orderBy(driver_standings_df.wins.desc()).take(10)
(2) Spark Jobs
Out[5]: [Row(race_year=2004, driver_name='Michael Schumacher', driver_nationality='German', total_points=148.0, wins=13),
Row(race_year=2013, driver_name='Sebastian Vettel', driver_nationality='German', total_points=397.0, wins=13),
Row(race_year=2011, driver_name='Sebastian Vettel', driver_nationality='German', total_points=392.0, wins=11),
Row(race_year=2019, driver_name='Lewis Hamilton', driver_nationality='British', total_points=413.0, wins=11),
Row(race_year=2014, driver_name='Lewis Hamilton', driver_nationality='British', total_points=384.0, wins=11),
Row(race_year=2002, driver_name='Michael Schumacher', driver_nationality='German', total_points=144.0, wins=11),
Row(race_year=2018, driver_name='Lewis Hamilton', driver_nationality='British', total_points=408.0, wins=11),
Row(race_year=2020, driver_name='Lewis Hamilton', driver_nationality='British', total_points=347.0, wins=11),
Row(race_year=2015, driver_name='Lewis Hamilton', driver_nationality='British', total_points=381.0, wins=10),
Row(race_year=2016, driver_name='Lewis Hamilton', driver_nationality='British', total_points=380.0, wins=10)]
Command took 2.48 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 6
Python
1 top_driver =driver_standings_df.orderBy(col("wins").desc())
2 overll_top_10_drivers = top_driver.limit(10)
top_driver: pyspark.sql.dataframe.DataFrame = [race_year: integer, driver_name: string ... 3 more fields]
overll_top_10_drivers: pyspark.sql.dataframe.DataFrame = [race_year: integer, driver_name: string ... 3 more fields]
Search
9:52 AM
4/23/2024

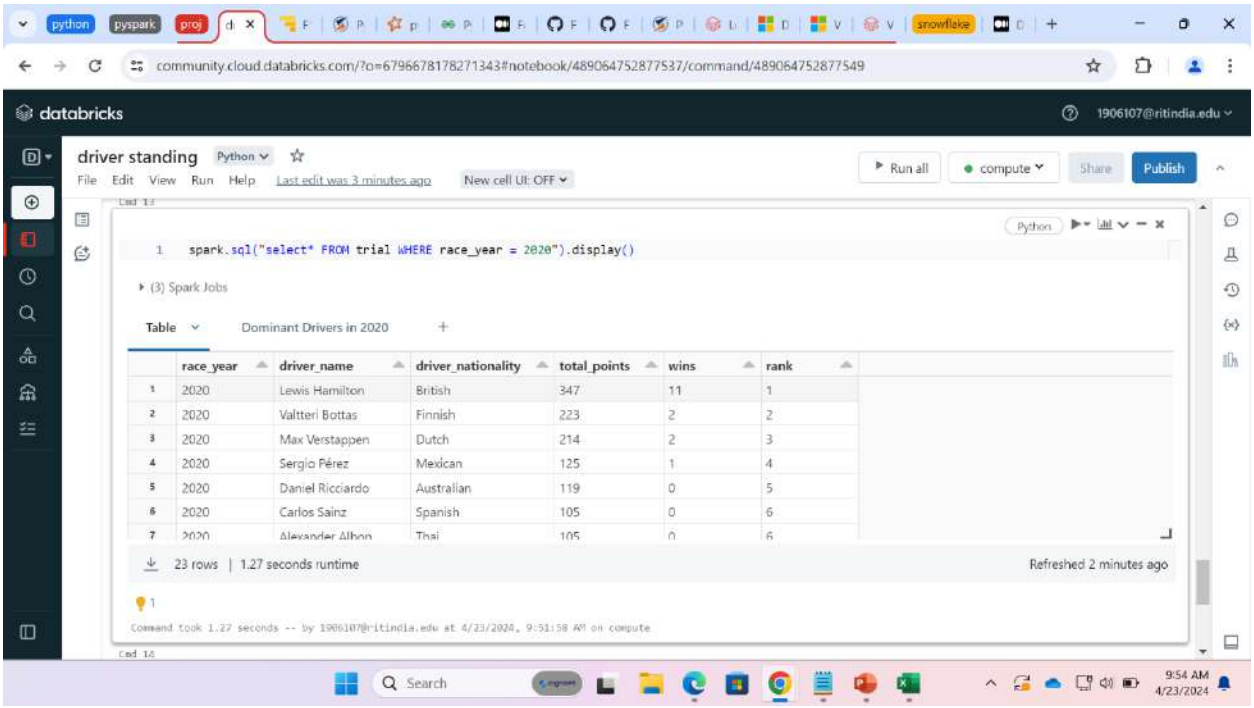

community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877539
databricks
1906107@ritindia.edu
Run all cells in this notebook.
driver standing
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
compute
Share
Publish
1 top_driver =driver_standings_df.orderBy(col("wins").desc())
2 overll_top_10_drivers = top_driver.limit(10)
top_driver: pyspark.sql.dataframe.DataFrame = [race_year: integer, driver_name: string ... 3 more fields]
overll_top_10_drivers: pyspark.sql.dataframe.DataFrame = [race_year: integer, driver_name: string ... 3 more fields]
Command took 0.29 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 7
Python
1 overll_top_10_drivers.display()
(2) Spark Jobs
Table
race_year driver_name driver_nationality total_points wins
1 2004 Michael Schumacher German 148 13
2 2013 Sebastian Vettel German 397 13
3 2011 Sebastian Vettel German 392 11
4 2019 Lewis Hamilton British 413 11
5 2014 Lewis Hamilton British 384 11
Search
9:53 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877539
databricks
1906107@ritindia.edu
Run all cells in this notebook.
driver standing
Python
File Edit View Run Help Last edit was 1 minute ago
New cell UI: OFF
Run all
compute
Share
Publish
#que 2 driver standing
Command took 0.08 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 9
from pyspark.sql.window import Window
from pyspark.sql.functions import desc, rank, asc
driver_rank_spec = Window.partitionBy("race_year").orderBy(desc("total_points"), desc("wins"))
final_df = driver_standings_df.withColumn("rank", rank().over(driver_rank_spec))
final_df: pyspark.sql.dataframe.DataFrame = [race_year: integer, driver_name: string ... 4 more fields]
Command took 0.29 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 10
final_df.display()
(3) Spark Jobs
Table
Yearwise Driver Standing
Search
9:53 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877539
databricks
1906107@ritindia.edu
Run all cells in this notebook.
driver standing
Python
File Edit View Run Help Last edit was 2 minutes ago
New cell UI: OFF
Run all
compute
Share
Publish
Cmd 10
final_df.display()
(3) Spark Jobs
Table
Yearwise Driver Standing
race_year driver_name driver_nationality total_points wins rank
1 1950 Nino Farina Italian 30 3 1
2 1950 Luigi Fagioli Italian 28 0 2
3 1950 Juan Fangio Argentine 27 3 3
4 1950 Louis Rosier French 13 0 4
5 1950 Johnnie Parsons American 9 1 5
6 1950 Alberto Ascari Italian 8 0 6
7 1950 Bill Holland American 6 0 7
3,122 rows | 2.70 seconds runtime
Refreshed 1 minute ago
Command took 2.70 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 11
Search
9:53 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877547
databricks
1906107@ritindia.edu
driver standing
Python
File Edit View Run Help Last edit was 2 minutes ago
New cell UI: OFF
Run all
compute
Share
Publish
Table
Yearwise Driver Standing
Nino Farina
Luigi Fagioli
Juan Fangio
Louis Rosier
Johnnie Parsons
Alberto Ascari
Bill Holland
Prince Bira
Louis Chiron
Reg Parnell
Mauri Rose
Peter Whitehead
Raymond Sommer
Philippe Étancelin
Cecil Green
Dorino Serafini
Robert Manzon
Yves Cabantous
Search
9:54 AM
4/23/2024


community.cloud.databricks.com/?o=6796678178271343#notebook/489064752877537/command/489064752877549
databricks
1906107@ritindia.edu
driver standing
Python
File Edit View Run Help Last edit was 2 minutes ago
New cell UI: OFF
Run all
compute
Share
Publish
final_df.createTempView("trial")
Command took 0.17 seconds -- by 1906107@ritindia.edu at 4/23/2024, 9:51:58 AM on compute
Cmd 12
spark.sql("select* FROM trial WHERE race_year = 2020").show()
(4) Spark Jobs
+---------+-----------------+-----------------+------------+----+----+
|race_year| driver_name|driver_nationality|total_points|wins|rank|
+---------+-----------------+-----------------+------------+----+----+
2020	Lewis Hamilton	British	347.0	11	1
2020	Valtteri Bottas	Finnish	223.0	2	2
2020	Max Verstappen	Dutch	214.0	2	3
2020	Sergio Pérez	Mexican	125.0	1	4
2020	Daniel Ricciardo	Australian	119.0	0	5
2020	Carlos Sainz	Spanish	105.0	0	6
2020	Alexander Albon	Thai	105.0	0	6
2020	Charles Leclerc	Monegasque	98.0	0	8
2020	Lando Norris	British	97.0	0	9
2020	Pierre Gasly	French	75.0	1	10
2020	Lance Stroll	Canadian	75.0	0	11
Search
9:54 AM
4/23/2024



	race_year	driver_name	driver_nationality	total_points	wins	rank
1	2020	Lewis Hamilton	British	347	11	1
2	2020	Valtteri Bottas	Finnish	223	2	2
3	2020	Max Verstappen	Dutch	214	2	3
4	2020	Sergio Pérez	Mexican	125	1	4
5	2020	Daniel Ricciardo	Australian	119	0	5
6	2020	Carlos Sainz	Spanish	105	0	6
7	2020	Alexander Albon	Thai	105	0	6