Week-2-3-Assignments-all-in-one Python

c

Cmd 1

```
1 %fs rm -r /user/hive/warehouse/demo_db.db/fire_service_calls_tbl
```

```
res2: Boolean = true
```

Command took 0.88 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 2

Assignment - 1

- You are given a dataset at the following location. /databricks-datasets/learning-spark-v2/sf-fire/sf-fire-calls.csv
- You are asked to do the following
  - Create a Spark data frame using the above file.
  - Create a Global Temporary View using the above data frame. Assume the view name is fire_service_calls_view.

Cmd 3

```
1 #Create a Spark data frame using the above file.
2 df1 = spark.read.format("csv").\
3     option("header", "true").\
4     option("inferSchema", "true").\
5     load("/databricks-datasets/learning-spark-v2/sf-fire/sf-fire-calls.csv");
```

▸ (2) Spark Jobs

Command took 49.36 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 4

```
1 #Create a Global Temporary View using the above data frame. Assume the view name is fire_service_calls_view
2 df1.createOrReplaceGlobalTempView("fire_service_calls_view");
```

Command took 0.34 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 5

```
1 df1.printSchema();
```

```
root
 |-- Call Number: integer (nullable = true)
 |-- Unit ID: string (nullable = true)
 |-- Incident Number: integer (nullable = true)
 |-- CallType: string (nullable = true)
 |-- Call Date: string (nullable = true)
 |-- Watch Date: string (nullable = true)
 |-- Call Final Disposition: string (nullable = true)
 |-- Available DtTm: string (nullable = true)
 |-- Address: string (nullable = true)
 |-- City: string (nullable = true)
 |-- Zipcode of Incident: integer (nullable = true)
 |-- Battalion: string (nullable = true)
 |-- Station Area: string (nullable = true)
 |-- Box: string (nullable = true)
 |-- OrigPriority: string (nullable = true)
 |-- Priority: string (nullable = true)
 |-- Final Priority: integer (nullable = true)
 |-- ALS Unit: boolean (nullable = true)
 |-- Call Type Group: string (nullable = true)
 |-- NumAlarms: integer (nullable = true)
```

Command took 0.11 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 6

Assignment - 2

- Create a Spark database demo_db
  - Create a spark table fire_service_calls_tbl in the demo_db
  - The schema for the table must match the view schema created in the previous assignment
  - The table data file must be a parquet file
  - Make sure your notebook is re-executable
- Load data into fire_service_calls from the fire_service_calls_view

Cmd 7

```
%sql
--Create a Spark database demo_db and table
drop table if exists demo_db.fire_service_calls_tbl;

create database if not exists demo_db ;

create table if not exists demo_db.fire_service_calls_tbl(
CallNumber integer,
  UnitID string,
  IncidentNumber integer,
  CallType string,
  CallDate string,
  WatchDate string,
  CallFinalDisposition string,
  AvailableDtTm string,
  Address string,
  City string,
  Zipcode integer,
  Battalion string,
  StationArea string,
  Box string,
  OriginalPriority string,
  Priority string,
  FinalPriority integer,
  ALSUnit boolean
```

Cmd 8

```
%sql
describe demo_db.fire_service_calls_tbl;
```

	col_name	data_type	comment
1	CallNumber	int	null
2	UnitID	string	null
3	IncidentNumber	int	null
4	CallType	string	null
5	CallDate	string	null
6	WatchDate	string	null
7	CallFinalDisposition	string	null

Showing all 28 rows.

Command took 0.28 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 9

```
%sql
--Load data into fire_service_calls from the fire_service_calls_view
insert into demo_db.fire_service_calls_tbl (select * from global_temp.fire_service_calls_view)
```

▸ (1) Spark Jobs

OK

Command took 2.25 minutes -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 10

```
%sql
select * from demo_db.fire_service_calls_tbl;
```

SQL

▸ (1) Spark Jobs

	CallNumber	UnitID	IncidentNumber	CallType	CallDate	WatchDate	CallFinalDisposition	Av
1	111050354	E14	11034920	Medical Incident	04/15/2011	04/15/2011	Other	04
2	111050355	E03	11034921	Structure Fire	04/15/2011	04/15/2011	Other	04
3	111050355	T03	11034921	Structure Fire	04/15/2011	04/15/2011	Other	04
4	111050356	73	11034922	Structure Fire	04/15/2011	04/15/2011	Other	04
5	111050356	B06	11034922	Structure Fire	04/15/2011	04/15/2011	Other	04
6	111050356	B10	11034922	Structure Fire	04/15/2011	04/15/2011	Other	04
7	111050356	D3	11034922	Structure Fire	04/15/2011	04/15/2011	Other	04

Truncated results, showing first 1000 rows.

Click to re-execute with maximum result limits.

Command took 9.01 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 11

Assignment - 3

Write Spark SQL queries to answer the following questions

1. How many distinct types of calls were made to the fire department?
2. What are distinct types of calls made to the fire department?
3. Find out all responses or delayed times greater than 5 mins?
4. What were the most common call types?
5. What zip codes accounted for the most common calls?
6. What San Francisco neighborhoods are in the zip codes 94102 and 94103
7. What was the sum of all calls, average, min, and max of the call response times?
8. How many distinct years of data are in the CSV file?
9. What week of the year in 2018 had the most fire calls?
10. What neighborhoods in San Francisco had the worst response time in 2018?

Cmd 12

Cmd 12

```
%sql
--1. How many distinct types of calls were made to the fire department?
select count(distinct CallType) from demo_db.fire_service_calls_tbl where CallType is not null;
```

▸ (3) Spark Jobs

	count(DISTINCT CallType)
1	32

Showing all 1 rows.

Command took 8.22 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 13

```
%sql
--2. What are distinct types of calls made to the fire department?
select distinct CallType from demo_db.fire_service_calls_tbl where CallType is not null;
```

▸ (2) Spark Jobs

	CallType
1	Elevator / Escalator Rescue
2	Marine Fire
3	Aircraft Emergency
4	Confined Space / Structure Collapse
5	Administrative
6	Alarms
7	Odor (Strange / Unknown)

Showing all 32 rows.

Command took 6.17 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 14

SQL

```
%sql
--3. Find out all responses or delayed times greater than 5 mins
select * from demo_db.fire_service_calls_tbl where where callType is not null and delay > 5
```

▸ (1) Spark Jobs

	CallNumber	UnitID	IncidentNumber	CallType	CallDate	WatchDate	CallFinalDisposition	AvailableDtTr
1	111060009	B03	11034934	Structure Fire	04/16/2011	04/15/2011	Other	04/16/2011 01:
2	111060015	T09	11034941	Other	04/16/2011	04/15/2011	Other	04/16/2011 01:
3	111060023	E41	11034947	Medical Incident	04/16/2011	04/15/2011	Other	04/16/2011 02
4	111060028	E36	11034952	Medical Incident	04/16/2011	04/15/2011	Other	04/16/2011 02
5	111060028	KM11	11034952	Medical Incident	04/16/2011	04/15/2011	Code 2 Transport	04/16/2011 02
6	111060065	AR1	11034985	Structure Fire	04/16/2011	04/15/2011	Other	04/16/2011 07
7	111060076	E18	11034994	Medical Incident	04/16/2011	04/15/2011	Medical Examiner	04/16/2011 09

Truncated results, showing first 1000 rows.

Click to re-execute with maximum result limits.

Command took 1.79 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 15

```
%sql
--4. What were the most common call types?
select CallType, count(*) as count from demo_db.fire_service_calls_tbl where callType is not null group by CallType order by count desc;
```

(2) Spark Jobs

	CallType	count
1	Medical Incident	2843475
2	Structure Fire	578998
3	Alarms	483518
4	Traffic Collision	175507
5	Citizen Assist / Service Call	65360
6	Other	56961
7	Outside Fire	51603

Showing all 32 rows.

Command took 5.74 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 16

```
%sql
--5. What zip codes accounted for the most common calls?
select Zipcode, CallType, count(*) as count from demo_db.fire_service_calls_tbl where callType is not null group by Zipcode,CallType order by count desc;
```

(2) Spark Jobs

	Zipcode	CallType	count
1	94102	Medical Incident	401457
2	94103	Medical Incident	370215
3	94110	Medical Incident	249279
4	94109	Medical Incident	238087
5	94124	Medical Incident	147564
6	94112	Medical Incident	139565
7	94115	Medical Incident	120087

Showing all 714 rows.

Edit Menu

Command took 6.30 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 17

```
%sql
--6. What San Francisco neighborhoods are in the zip codes 94102 and 94103
--Check this query???
select distinct Neighborhood, Zipcode  from demo_db.fire_service_calls_tbl where Zipcode in (94102,94103) and city in ('San Francisco', 'SFO', 'SF', 'SAN FRANCISCO');
```

(2) Spark Jobs

	Neighborhood	Zipcode
1	Potrero Hill	94103
2	Western Addition	94102
3	Tenderloin	94102
4	Nob Hill	94102
5	Castro/Upper Market	94103
6	South of Market	94102
7	South of Market	94103

Showing all 15 rows.

Edit Menu

Command took 4.17 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:53:04 on c

Cmd 18

```
%sql
--7. What was the sum of all calls, average, min, and max of the call response times?
select sum(NumAlarms), avg(delay), min(delay), max(delay) from demo_db.fire_service_calls_tbl;
```

▸ (2) Spark Jobs

	sum(NumAlarms)	avg(delay)	min(delay)	max(delay)
1	4403441	3.902170335891614	0.016666668	1879.6167

Showing all 1 rows.

Command took 4.31 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 19

SQL

```
%sql
--8. How many distinct years of data are in the CSV file?
select count(distinct (year(to_date(CallDate, "MM/dd/yyyy")))) from demo_db.fire_service_calls_tbl;
```

▸ (3) Spark Jobs

	count(DISTINCT year(to_date(CallDate, MM/dd/yyyy)))
1	19

Showing all 1 rows.

Command took 13.82 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 20

SQL

```
%sql
-- 9. What week of the year in 2018 had the most fire calls? (All are fire calls, so no need of separate condition of CallType like '%Fire%')
select weekofyear(to_timestamp(CallDate, "MM/dd/yyyy")) as week, count(*) as count from demo_db.fire_service_calls_tbl where
year(to_timestamp(CallDate, "MM/dd/yyyy")) == 2018 group by week order by count desc;
```

▸ (2) Spark Jobs

	week	count
1	1	6401
2	25	6163
3	13	6103
4	22	6060
5	44	6048
6	27	6042
7	16	6009

Showing all 45 rows.

Command took 7.60 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 23:13:59 on c

Cmd 21

Markdown

Assignment - 4

- You are given a dataset at the this location /databricks-datasets/learning-spark-v2/sf-fire/sf-fire-calls.csv
  - Create a data frame using the above data
  - Transform the data frame to rename columns removing the space in the column names
  - Transform the data frame to fix the date and timestamp column types

Cmd 22

```
assignment4_df = spark.read.format("csv").option("header", "true").option("inferSchema", "true").load("/databricks-datasets/learning-spark-v2/sf-fire/sf-fire-calls.csv");
```

Cmd 23

Python

```
renamed_df = assignment4_df\
            .withColumnRenamed("Call Number","CallNumber")\
            .withColumnRenamed("Unit ID","UnitID")\
            .withColumnRenamed("Incident Number","IncidentNumber")\
            .withColumnRenamed("Call Date","CallDate")\
            .withColumnRenamed("Watch Date","WatchDate")\
            .withColumnRenamed("Call Final Disposition","CallFinalDisposition")\
            .withColumnRenamed("Available DtTm","AvailableDtTm")\
            .withColumnRenamed("Zipcode of Incident","ZipcodeOfIncident")\
            .withColumnRenamed("Station Area","StationArea")\
            .withColumnRenamed("Final Priority","FinalPriority")\
            .withColumnRenamed("ALS Unit","ALSUnit")\
            .withColumnRenamed("Call Type","CallType")\
            .withColumnRenamed("Unit sequence in call dispatch","UnitSequenceInCallDispatch")\
            .withColumnRenamed("Fire Prevention D Delete ict","FirePreventionDistrict")\
            .withColumnRenamed("Supervisor District","SupervisorDistrict");
```

Command took 0.96 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 24

Cmd 24

```
from pyspark.sql.functions import to_date, to_timestamp
renamed_df = renamed_df\
            .withColumn("CallDate", to_date("WatchDate", "MM/dd/yyyy"))\
            .withColumn("WatchDate", to_date("WatchDate", "MM/dd/yyyy"))\
            .withColumn("AvailableDtTm", to_timestamp("AvailableDtTm", "MM/dd/yyyy hh:mm:ss a"));
```

Cmd 25

```
renamed_df.printSchema();
```

```
root
 |-- CallNumber: integer (nullable = true)
 |-- UnitID: string (nullable = true)
 |-- IncidentNumber: integer (nullable = true)
 |-- CallType: string (nullable = true)
 |-- CallDate: date (nullable = true)
 |-- WatchDate: date (nullable = true)
 |-- CallFinalDisposition: string (nullable = true)
 |-- AvailableDtTm: timestamp (nullable = true)
 |-- Address: string (nullable = true)
 |-- City: string (nullable = true)
 |-- ZipcodeOfIncident: integer (nullable = true)
 |-- Battalion: string (nullable = true)
 |-- StationArea: string (nullable = true)
 |-- Box: string (nullable = true)
 |-- OrigPriority: string (nullable = true)
 |-- Priority: string (nullable = true)
 |-- FinalPriority: integer (nullable = true)
 |-- ALSUnit: boolean (nullable = true)
 |-- Call Type Group: string (nullable = true)
 |-- NumAlarms: integer (nullable = true)
```

Command took 0.05 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 26

Markdown

Assignment - 5

Write Spark data frame queries to answer the following questions using the data frame you created in the previous assignment

1. How many distinct types of calls were made to the fire department?
2. What are distinct types of calls made to the fire department?
3. Find out all responses or delayed times greater than 5 mins?
4. What were the most common call types?
5. What zip codes accounted for the most common calls?
6. What San Francisco neighborhoods are in the zip codes 94102 and 94103
7. What was the sum of all calls, average, min, and max of the call response times?
8. How many distinct years of data are in the CSV file?
9. What week of the year in 2018 had the most fire calls?
10. What neighborhoods in San Francisco had the worst response time in 2018?

Cmd 27

```
# 1. How many distinct types of calls were made to the fire department?
q1_df = renamed_df.filter("CallType is not null").select("CallType").distinct().count();
display(q1_df);
```

▸ (3) Spark Jobs

32

Command took 47.21 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 28

```
# 2. What are distinct types of calls made to the fire department?
q2_df = renamed_df.select("CallType").distinct();
q2_df.show(q2_df.count());
```

▸ (5) Spark Jobs

```
+--------------------+
|            CallType|
+--------------------+
|Elevator / Escala...|
|         Marine Fire|
|  Aircraft Emergency|
|Confined Space / ...|
|      Administrative|
|              Alarms|
|Odor (Strange / U...|
|Citizen Assist / ...|
|              HazMat|
|Watercraft in Dis...|
|           Explosion|
|           Oil Spill|
|        Vehicle Fire|
|  Suspicious Package|
|Extrication / Ent...|
|               Other|
|        Outside Fire|
|   Traffic Collision|
```

Command took 1.51 minutes -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 29

Python

```
# 3. Find out all responses or delayed times greater than 5 mins?
q3_df = renamed_df.where("delay > 5");
display(q3_df);
```

▸ (1) Spark Jobs

	CallNumber	UnitID	IncidentNumber	CallType	CallDate	WatchDate	CallFinalDisposition	AvailableDtTm
1	20110014	M29	2003234	Medical Incident	2002-01-10	2002-01-10	Other	2002-01-11T01:58:43
2	20110017	M13	2003236	Medical Incident	2002-01-10	2002-01-10	Other	2002-01-11T02:27:14.
3	20110019	M36	2003238	Medical Incident	2002-01-10	2002-01-10	Other	2002-01-11T02:28:30
4	20110039	M41	2003257	Medical Incident	2002-01-10	2002-01-10	Other	2002-01-11T06:04:36

Cmd 30

```
# 4. What were the most common call types?
q4_df = renamed_df.select("CallType").groupBy("CallType").count().orderBy("count", ascending=False);
display(q4_df)
```

▸ (2) Spark Jobs

	CallType	count
1	Medical Incident	2843475
2	Structure Fire	578998
3	Alarms	483518
4	Traffic Collision	175507
5	Citizen Assist / Service Call	65360
6	Other	56961
7	Outside Fire	51603

Showing all 32 rows.

Command took 37.10 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 31

```python
# 5. What zip codes accounted for the most common calls?
q5_df = renamed_df.select("CallType", "ZipcodeOfIncident").groupBy("CallType", "ZipcodeOfIncident").count().orderBy("count", ascending=False);
display(q5_df);
```

(2) Spark Jobs

	CallType	ZipcodeOfIncident	count
1	Medical Incident	94102	401457
2	Medical Incident	94103	370215
3	Medical Incident	94110	249279
4	Medical Incident	94109	238087
5	Medical Incident	94124	147564
6	Medical Incident	94112	139565
7	Medical Incident	94115	120087

Showing all 714 rows.

Command took 33.55 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 32

```python
# 6. What San Francisco neighborhoods are in the zip codes 94102 and 94103
q6_df = renamed_df.select("Neighborhood", "ZipcodeOfIncident").where("ZipcodeOfIncident in (94102,94103) and city in ('San Francisco', 'SFO', 'SF', 'SAN FRANCISCO')").distinct();
display(q6_df);
```

(2) Spark Jobs

	Neighborhood	ZipcodeOfIncident
1	Potrero Hill	94103
2	Western Addition	94102
3	Tenderloin	94102
4	Nob Hill	94102
5	Castro/Upper Market	94103
6	Mission	94102
7	South of Market	94103

Showing all 15 rows.

Command took 46.38 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:53:36 on c

Cmd 33

```python
# 7. What was the sum of all calls, average, min, and max of the call response times?
from pyspark.sql.functions import expr;
q7_df = renamed_df.select(expr("sum(NumAlarms)"),expr("avg(Delay)"),expr("min(Delay)"),expr("max(Delay)"));
display(q7_df);
```

(2) Spark Jobs

	sum(NumAlarms)	avg(Delay)	min(Delay)	max(Delay)
1	4403441	3.902170335063649	0.016666666666666666	1879.6166666666666

Showing all 1 rows.

Command took 45.07 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 34

```python
# 8. How many distinct years of data are in the CSV file?
q8_df = renamed_df.select(expr("year(CallDate) as year")).distinct().orderBy("year", ascending=False);
display(q8_df);
```

(2) Spark Jobs

	year
1	2018
2	2017
3	2016
4	2015
5	2014
6	2013
7	2012

Showing all 19 rows.

Command took 44.72 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c

Cmd 35

```
# 9. What week of the year in 2018 had the most fire calls? (All are fire calls, so no need of separate condition of CallType like '%Fire%')
q9_df = renamed_df.select(expr("weekofyear(CallDate) as week")).filter("year(CallDate) == 2018").groupBy("week").count().orderBy("count",
ascending=False);
display(q9_df);
```

(2) Spark Jobs

	week	count
1	25	6206
2	1	6157
3	44	6063
4	13	6049
5	22	6029
6	43	5999
7	16	5996

Showing all 45 rows.

Command took 41.03 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 23:21:38 on c

Cmd 36

```
# 10. What neighborhoods in San Francisco had the worst response time in 2018?
q10_df = renamed_df.filter("year(CallDate) == 2018").groupBy("Neighborhood").sum("Delay");
display(q10_df);
```

(2) Spark Jobs

	Neighborhood	sum(Delay)
1	Inner Sunset	16696.100000000002
2	Haight Ashbury	12761.966666666665
3	Lincoln Park	1254.1833333333334
4	Japantown	8678.433333333336
5	None	844.2000000000002
6	North Beach	20743.649999999998
7	Lone Mountain/USF	11873.533333333344

Showing all 42 rows.

Command took 48.65 seconds -- by sachin.arpit.learning@gmail.com at 25/05/2022, 22:32:07 on c