Week4_Lab01_Simonsen

June 2, 2024

[ ]:
```
from pyspark.sql.types import *
import pyspark.sql.functions as F
```

[ ]:
```
fire_schema = StructType([StructField('CallNumber', IntegerType(), True),
                     StructField('UnitID', StringType(), True),
                     StructField('IncidentNumber', IntegerType(), True),
                     StructField('CallType', StringType(), True),
                     StructField('CallDate', StringType(), True),
                     StructField('WatchDate', StringType(), True),
                     StructField('CallFinalDisposition', StringType(), True),
                     StructField('AvailableDtTm', StringType(), True),
                     StructField('Address', StringType(), True),
                     StructField('City', StringType(), True),
                     StructField('Zipcode', IntegerType(), True),
                     StructField('Battalion', StringType(), True),
                     StructField('StationArea', StringType(), True),
                     StructField('Box', StringType(), True),
                     StructField('OriginalPriority', StringType(), True),
                     StructField('Priority', StringType(), True),
                     StructField('FinalPriority', IntegerType(), True),
                     StructField('ALSUnit', BooleanType(), True),
                     StructField('CallTypeGroup', StringType(), True),
                     StructField('NumAlarms', IntegerType(), True),
                     StructField('UnitType', StringType(), True),
                     StructField('UnitSequenceInCallDispatch', IntegerType(),␣
 ↪True),
                     StructField('FirePreventionDistrict', StringType(), True),
                     StructField('SupervisorDistrict', StringType(), True),
                     StructField('Neighborhood', StringType(), True),
                     StructField('Location', StringType(), True),
                     StructField('RowID', StringType(), True),
                     StructField('Delay', FloatType(), True)])
```

[ ]:
```
sf_fire_file = "dbfs:/FileStore/Merrimack/Week_4/sf_fire_calls.csv"
fire_df = spark.read.csv(sf_fire_file,header=True, schema=fire_schema)
```

[ ]:
```
fire_df.show(5)
```

```
+----------+------+--------------+----------------+----------+----------+-------
-------------+--------------------+--------------------+----+-------+---------+-
----------+----+----------------+--------+-------------+-------+-------------+--
-------+--------+--------------------------+----------------------+-------------
-----+--------------------+--------------------+-------------+---------+
|CallNumber|UnitID|IncidentNumber|        CallType|  CallDate|
WatchDate|CallFinalDisposition|       AvailableDtTm|
Address|City|Zipcode|Battalion|StationArea| Box|OriginalPriority|Priority|FinalP
riority|ALSUnit|CallTypeGroup|NumAlarms|UnitType|UnitSequenceInCallDispatch|Fire
PreventionDistrict|SupervisorDistrict|        Neighborhood|            Location|
RowID|    Delay|
+----------+------+--------------+----------------+----------+----------+-------
-------------+--------------------+--------------------+----+-------+---------+-
----------+----+----------------+--------+-------------+-------+-------------+--
-------+--------+--------------------------+----------------------+-------------
-----+--------------------+--------------------+-------------+---------+
|  20110016|   T13|       2003235|  Structure Fire|01/11/2002|01/10/2002|
Other|01/11/2002 01:51:...|2000 Block of CAL...|  SF|  94109|      B04|
38|3362|              3|       3|            3|  false|         NULL|        1|
TRUCK|                         2|                    4|                5|
Pacific Heights|(37.7895840679362...|020110016-T13|     2.95|
|  20110022|   M17|       2003241|Medical Incident|01/11/2002|01/10/2002|
Other|01/11/2002 03:01:...|0 Block of SILVER...|  SF|  94124|      B10|
42|6495|              3|       3|            3|   true|         NULL|        1|
MEDIC|                         1|                   10|
10|Bayview Hunters P...|(37.7337623673897...|020110022-M17|      4.7|
|  20110023|   M41|       2003242|Medical Incident|01/11/2002|01/10/2002|
Other|01/11/2002 02:39:...|MARKET ST/MCALLIS...|  SF|  94102|      B03|
01|1455|              3|       3|            3|   true|         NULL|        1|
MEDIC|                         2|                    3|                6|
Tenderloin|(37.7811772186856...|020110023-M41|2.4333334|
|  20110032|   E11|       2003250|    Vehicle Fire|01/11/2002|01/10/2002|
Other|01/11/2002 04:16:...|APPLETON AV/MISSI...|  SF|  94110|      B06|
32|5626|              3|       3|            3|  false|         NULL|        1|
ENGINE|                         1|                    6|                9|
Bernal Heights|(37.7388432849018...|020110032-E11|      1.5|
|  20110043|   B04|       2003259|          Alarms|01/11/2002|01/10/2002|
Other|01/11/2002 06:01:...|1400 Block of SUT...|  SF|  94109|      B04|
03|3223|              3|       3|            3|  false|         NULL|        1|
CHIEF|                         2|                    4|                2|
Western Addition|(37.7872890372638...|020110043-B04|3.4833333|
+----------+------+--------------+----------------+----------+----------+-------
-------------+--------------------+--------------------+----+-------+---------+-
----------+----+----------------+--------+-------------+-------+-------------+--
-------+--------+--------------------------+----------------------+-------------
-----+--------------------+--------------------+-------------+---------+
only showing top 5 rows
```

[ ]:
```
fire_ts_df = (fire_df
              .withColumn('IncidentDate',F.to_timestamp(F.col('CallDate'),'MM/dd/
 ↪yyyy'))
              .drop('CallDate')
              .withColumn('OnWatchDate',F.to_timestamp(F.col('WatchDate'),'MM/dd/
 ↪yyyy'))
              .drop('WatchDate')
              .withColumn('AvailableDtTS',F.to_timestamp(F.
 ↪col('AvailableDtTm'),'MM/dd/yyyy hh:mm:ss a'))
              .drop('AvailableDtTm')
             )
```

[ ]:
```
fire_ts_df.cache()
fire_ts_df.columns
```

[ ]:
```
['CallNumber',
 'UnitID',
 'IncidentNumber',
 'CallType',
 'CallFinalDisposition',
 'Address',
 'City',
 'Zipcode',
 'Battalion',
 'StationArea',
 'Box',
 'OriginalPriority',
 'Priority',
 'FinalPriority',
 'ALSUnit',
 'CallTypeGroup',
 'NumAlarms',
 'UnitType',
 'UnitSequenceInCallDispatch',
 'FirePreventionDistrict',
 'SupervisorDistrict',
 'Neighborhood',
 'Location',
 'RowID',
 'Delay',
 'IncidentDate',
 'OnWatchDate',
 'AvailableDtTS']
```

[ ]:
```
(fire_ts_df
    .select('IncidentDate','OnWatchDate','AvailableDtTS')
    .show(5, False)
```

```
)
```

```
+-------------------+-------------------+-------------------+
|IncidentDate       |OnWatchDate        |AvailableDtTS      |
+-------------------+-------------------+-------------------+
|2002-01-11 00:00:00|2002-01-10 00:00:00|2002-01-11 01:51:44|
|2002-01-11 00:00:00|2002-01-10 00:00:00|2002-01-11 03:01:18|
|2002-01-11 00:00:00|2002-01-10 00:00:00|2002-01-11 02:39:50|
|2002-01-11 00:00:00|2002-01-10 00:00:00|2002-01-11 04:16:46|
|2002-01-11 00:00:00|2002-01-10 00:00:00|2002-01-11 06:01:58|
+-------------------+-------------------+-------------------+
only showing top 5 rows
```

[ ]:
```
# 1)        What were all the different types of fire calls in 2018?

(fire_ts_df
    .select("CallType")
    .where(F.col('CallType').isNotNull())
    .where(F.year('IncidentDate') == 2018 )
    .distinct()
    .show(truncate=False)
)
```

```
+-------------------------------+
|CallType                       |
+-------------------------------+
|Elevator / Escalator Rescue    |
|Alarms                         |
|Odor (Strange / Unknown)       |
|Citizen Assist / Service Call  |
|HazMat                         |
|Vehicle Fire                   |
|Other                          |
|Outside Fire                   |
|Traffic Collision              |
|Assist Police                  |
|Gas Leak (Natural and LP Gases)|
|Water Rescue                   |
|Electrical Hazard              |
|Structure Fire                 |
|Medical Incident               |
|Fuel Spill                     |
|Smoke Investigation (Outside)  |
|Train / Rail Incident          |
|Explosion                      |
|Suspicious Package             |
+-------------------------------+
```

[ ]:
```
(fire_ts_df
 .select("CallType").where(F.col("CallType").isNotNull())
 .groupBy("CallType")
 .count()
 .orderBy("count", ascending=False)
 .show(n=10, truncate=False))
```

```
+-------------------------------+------+
|CallType                       |count |
+-------------------------------+------+
|Medical Incident               |113794|
|Structure Fire                 |23319 |
|Alarms                         |19406 |
|Traffic Collision              |7013  |
|Citizen Assist / Service Call  |2524  |
|Other                          |2166  |
|Outside Fire                   |2094  |
|Vehicle Fire                   |854   |
|Gas Leak (Natural and LP Gases)|764   |
|Water Rescue                   |755   |
+-------------------------------+------+
only showing top 10 rows
```

[ ]:
```
# 2) What months within the year 2018 saw the highest number of fire calls?␣
 ↪ANSWER: October, May, and March had the highest number of fire calls.

(fire_ts_df
 .select(F.month('IncidentDate').alias('Month'))
 .where(F.year('IncidentDate')==2018)
 .groupBy('Month')
 .count()
 .orderBy('count',ascending=False)
 .show()
)
```

```
+-----+-----+
|Month|count|
+-----+-----+
|   10| 1068|
|    5| 1047|
|    3| 1029|
|    8| 1021|
|    1| 1007|
|    6|  974|
|    7|  974|
|    9|  951|
```

```
|    4|  947|
|    2|  919|
|   11|  199|
+-----+-----+
```

[ ]:
```
# 3)        Which neighborhood in San Francisco generated the most fire calls in 2018? ANSWER: Tenderloin generated the most fire calls in 2018.

(fire_ts_df
 .select('Neighborhood')
 .where(F.year('IncidentDate')==2018)
 .groupBy('Neighborhood')
 .count()
 .orderBy('count',ascending=False)
 .show(n=5, truncate=False)
)
```

```
+------------------------------+-----+
|Neighborhood                  |count|
+------------------------------+-----+
|Tenderloin                    |1393 |
|South of Market               |1053 |
|Mission                       |913  |
|Financial District/South Beach|772  |
|Bayview Hunters Point         |522  |
+------------------------------+-----+
only showing top 5 rows
```

[ ]:
```
# 4)        Which neighborhoods had the worst response times to fire calls in 2018? ANSWER: In 2018, Chinatown, Financial District/South Beach, and Tenderloin had the worst response time to fire calls.

fire_df_response = fire_ts_df.withColumnRenamed("Delay","ResponseDelayedMins")

(fire_df_response
 .select('Neighborhood','ResponseDelayedMins')
 .where(F.year('IncidentDate')==2018)
 .orderBy('ResponseDelayedMins',ascending=False)
 .show(n=10,truncate=False)
)
```

```
+------------------------------+-------------------+
|Neighborhood                  |ResponseDelayedMins|
+------------------------------+-------------------+
|Chinatown                     |491.26666          |
|Financial District/South Beach|406.63333          |
```

```
|Tenderloin                     |340.48334          |
|Haight Ashbury                 |175.86667          |
|Bayview Hunters Point          |155.8              |
|Financial District/South Beach|135.51666          |
|Pacific Heights                |129.01666          |
|Potrero Hill                   |109.8              |
|Inner Sunset                   |106.13333          |
|South of Market                |94.71667           |
+-------------------------------+-------------------+
only showing top 10 rows
```

[ ]:
```
# 5)        Which week in the year in 2018 had the most fire calls? ANSWER: Week␣
↪22 had the most fire calls in the year 2018.

(fire_ts_df
 .select(F.weekofyear('IncidentDate').alias('Week'))
 .where(F.year('IncidentDate')==2018)
 .groupBy('Week')
 .count()
 .orderBy('count',ascending=False)
 .show(n=5, truncate=False)
)
```

```
+----+-----+
|Week|count|
+----+-----+
|22  |259  |
|40  |255  |
|43  |250  |
|25  |249  |
|1   |246  |
+----+-----+
only showing top 5 rows
```

[ ]:
```
# 6)        Is there a correlation between neighborhood, zip code, and number of␣
↪fire calls? ANSWER: There does appear to be a correlation. The zip codes with␣
↪the highest count of fire calls correlate to 3 out of the 5 same neighborhoods␣
↪identified as having the highest fire calls in question 3 above. Additionally,␣
↪two of the zip codes contain the Mission neighborhood.


(fire_ts_df
 .groupBy('ZipCode')
 .count()
 .orderBy('count',ascending=False)
 .show(n=5, truncate=False)
```

```
)

(fire_ts_df
 .select('Neighborhood','ZipCode')
 .where((F.col('Zipcode') == 94102) | (F.col('Zipcode') == 94103) | (F.
 ↪col('Zipcode') == 94110))
 .groupBy('Neighborhood','Zipcode')
 .count()
 .orderBy('count',ascending=False)
 .show(n=5, truncate=False)
)
```

```
+-------+-----+
|ZipCode|count|
+-------+-----+
|94102  |21840|
|94103  |20897|
|94110  |14801|
|94109  |14686|
|94124  |9236 |
+-------+-----+
only showing top 5 rows

+---------------+-------+-----+
|Neighborhood   |Zipcode|count|
+---------------+-------+-----+
|Tenderloin     |94102  |17084|
|South of Market|94103  |13762|
|Mission        |94110  |10444|
|Mission        |94103  |5445 |
|Bernal Heights |94110  |3109 |
+---------------+-------+-----+
only showing top 5 rows
```

[ ]:
```
# 7)        How can we use Parquet files or SQL tables to store this data and␣
 ↪read it back? ANSWER: The below code stores a dbfs file path as a path␣
 ↪variable, and then executes the code. The code itself first repartitions the␣
 ↪data to avoid creating small, sharded data files for each unit type. Then, the␣
 ↪code underneath creates an individual parquet file for each UnitType, saving␣
 ↪each parquet file in the path variable specified. The same execution could␣
 ↪also be represented in SQL to write a table to the hive_metastore. To do this,␣
 ↪all that would really change is the last line of code, instead running a␣
 ↪commmand that looks like .saveAsTable('hive_metastore_schema').
path = "dbfs:/FileStore/Merrimack/Week_4/data_parquet_week4"

(fire_df
```

```
  .repartition("UnitType")
  .write.format('parquet')
  .partitionBy("UnitType")
  .mode('overwrite')
  .option("header", "true")
  .save(path)
)
```