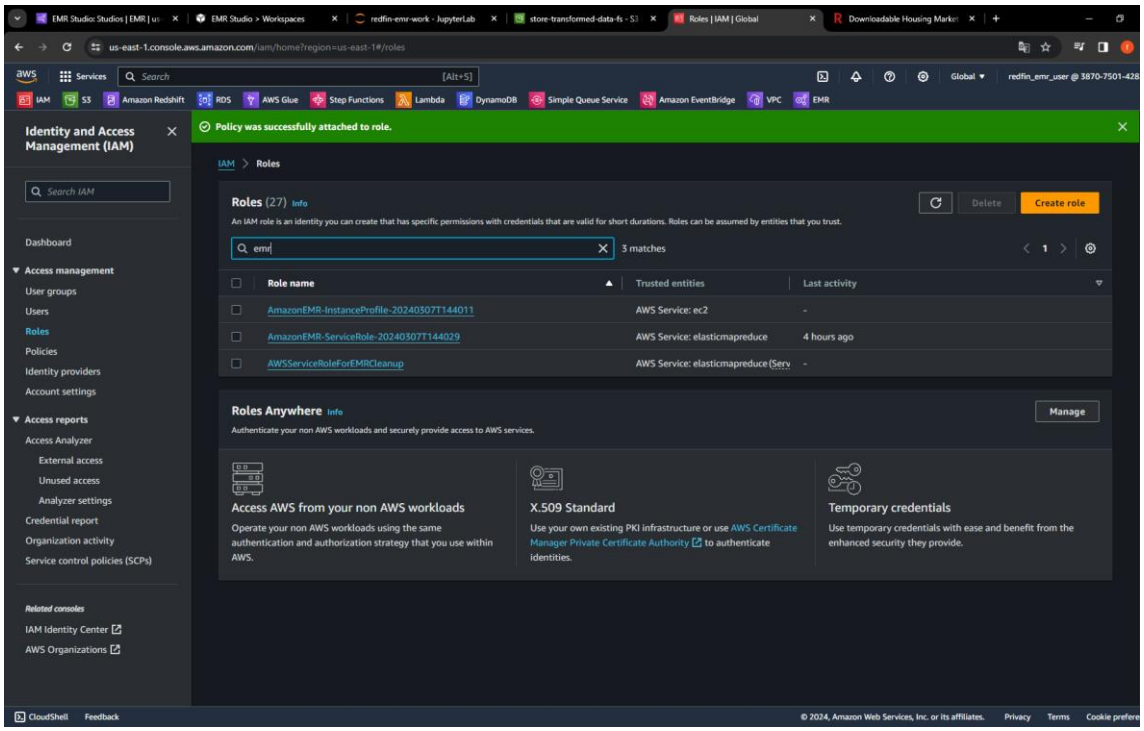
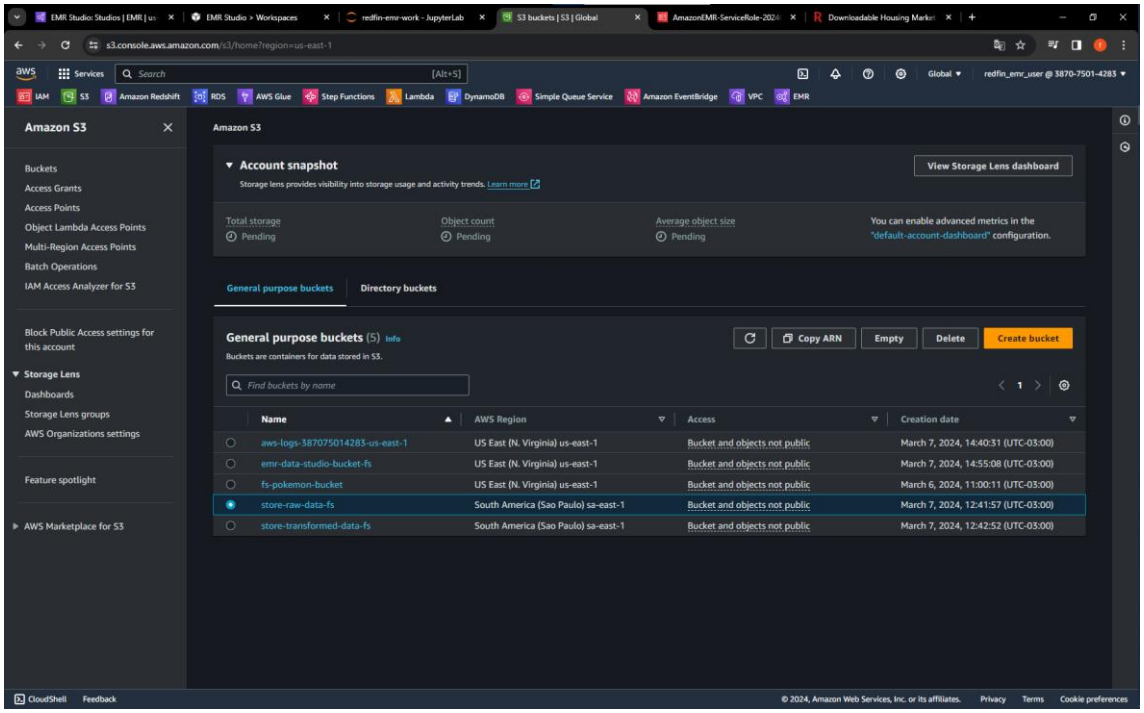
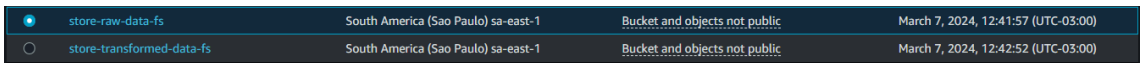
IAM Roles and permissions:

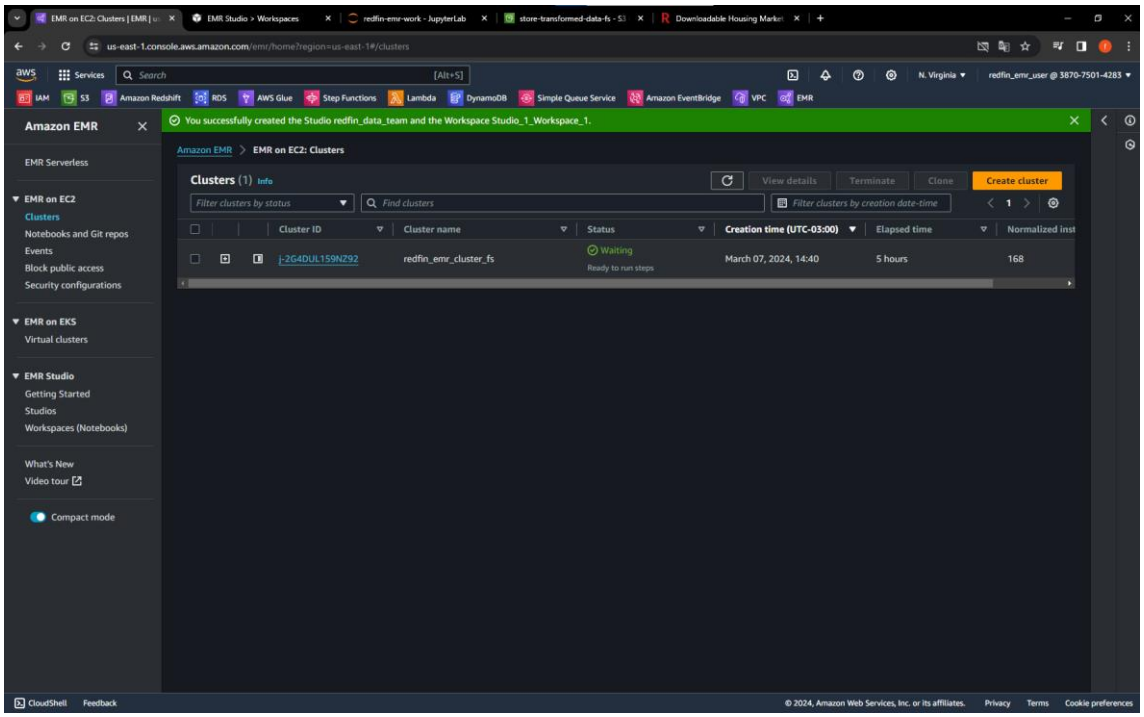


S3 Buckets:

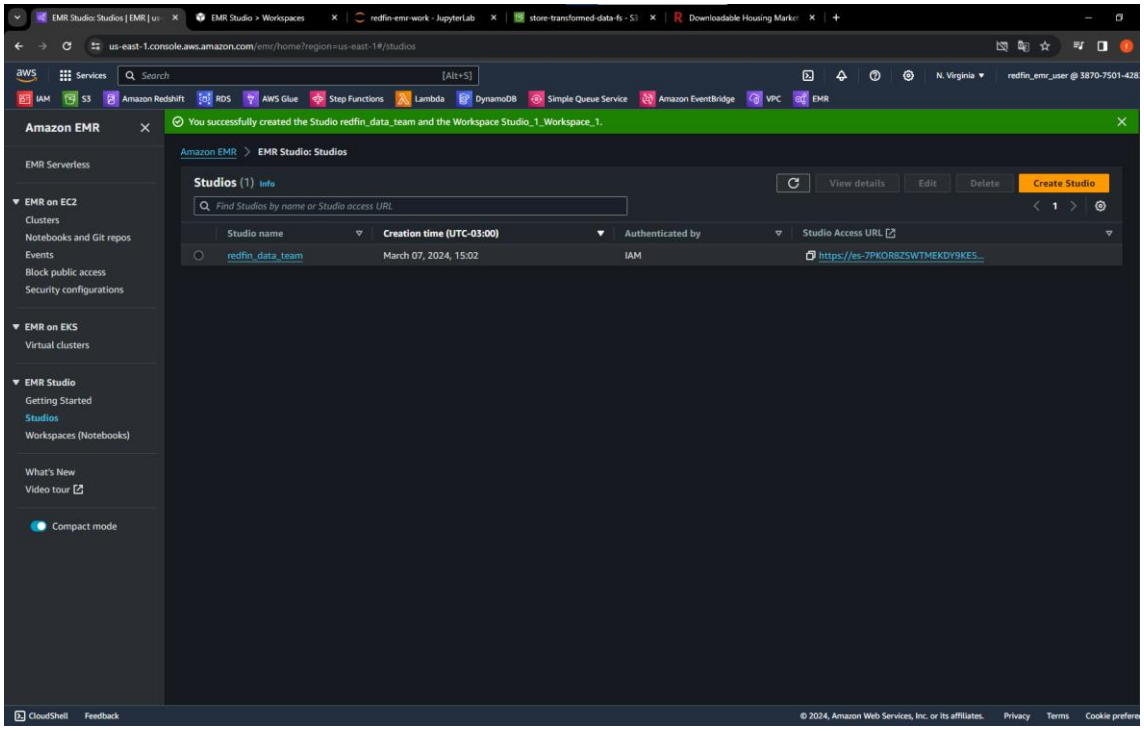




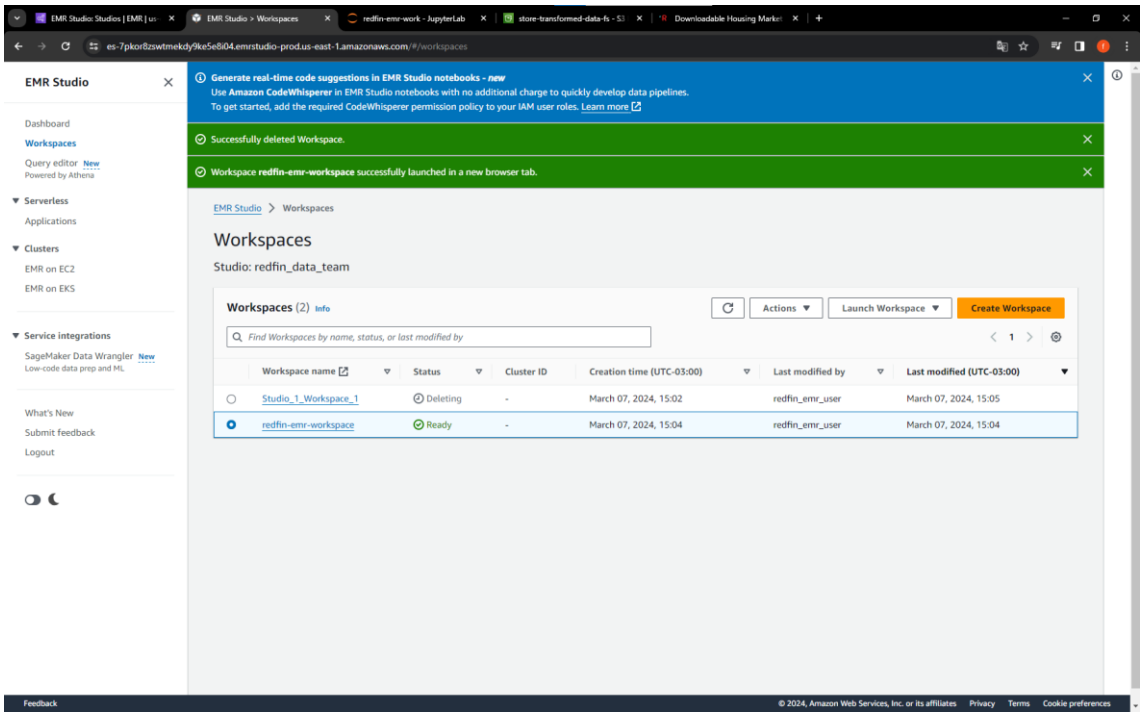
EMR on EC2 Cluster:



EMR Studio:

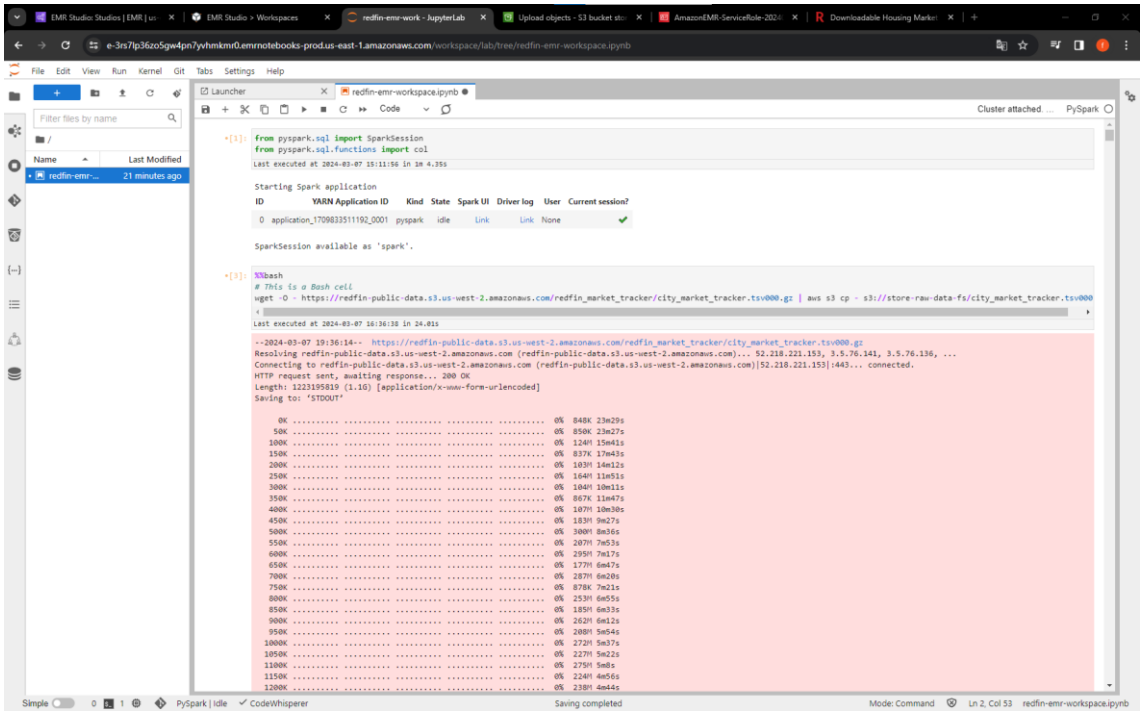


EMR Workspace:

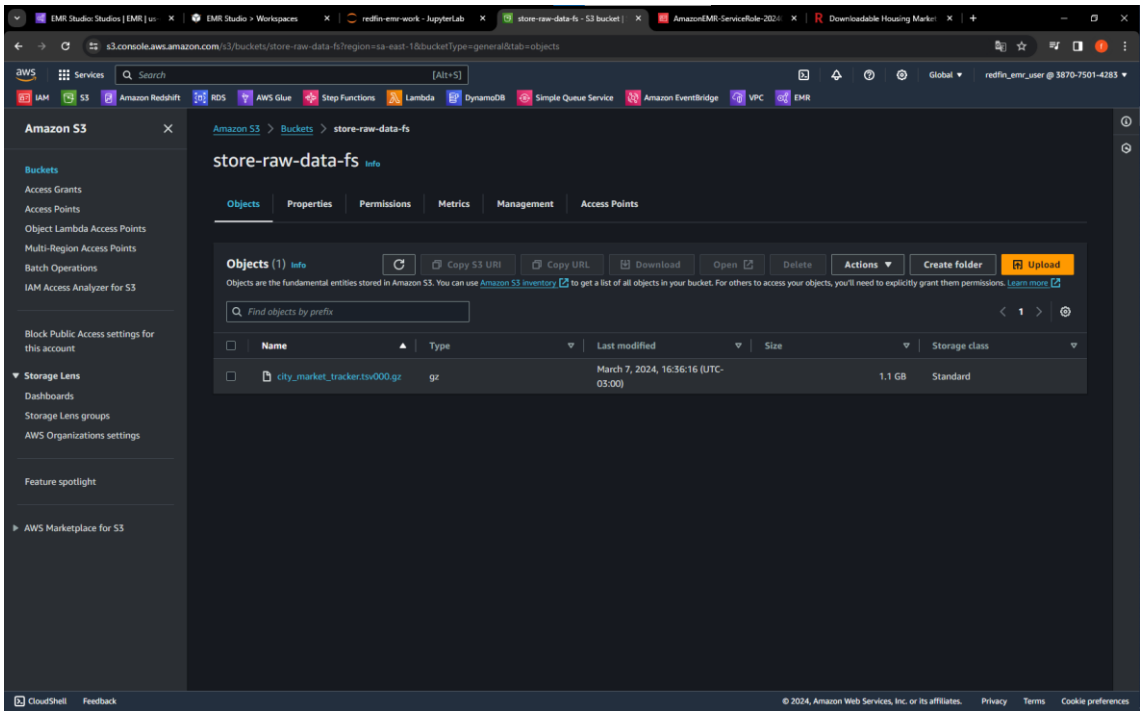


Pyspark Notebook:

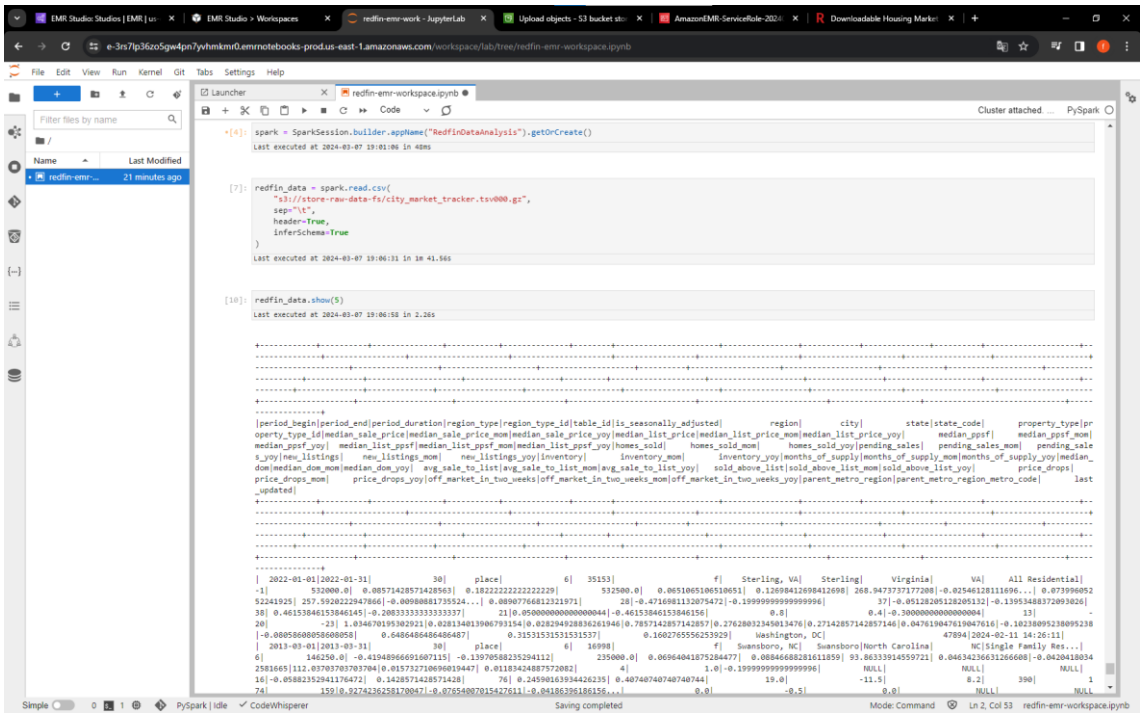
(Api request)

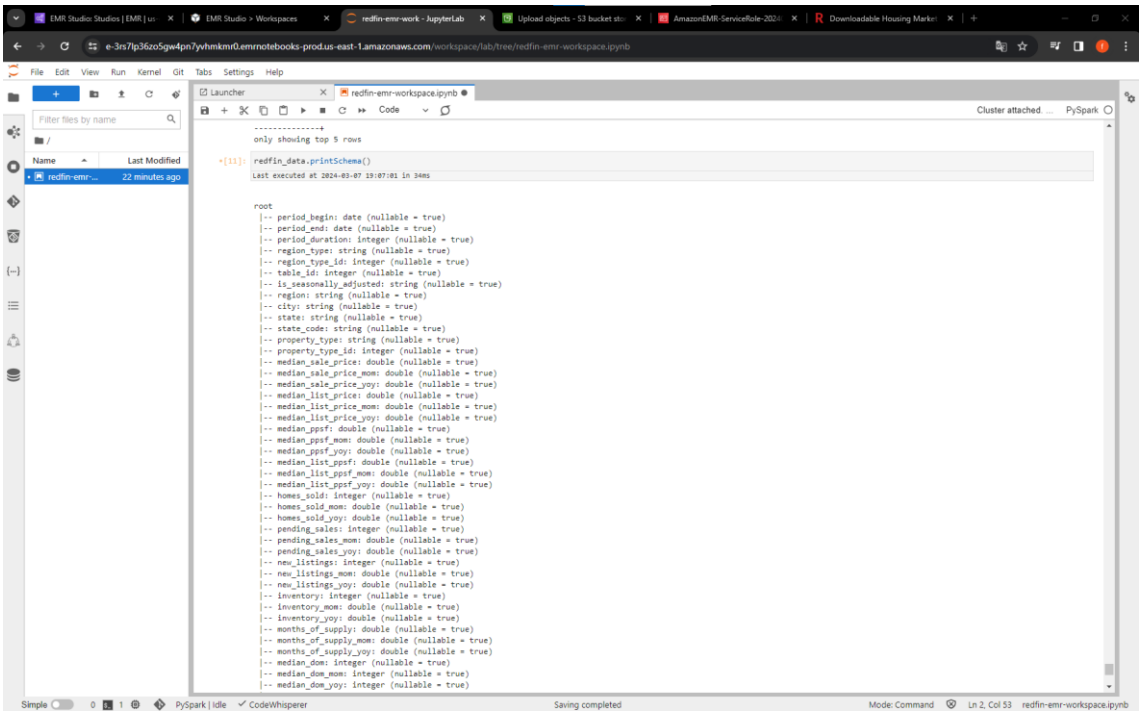


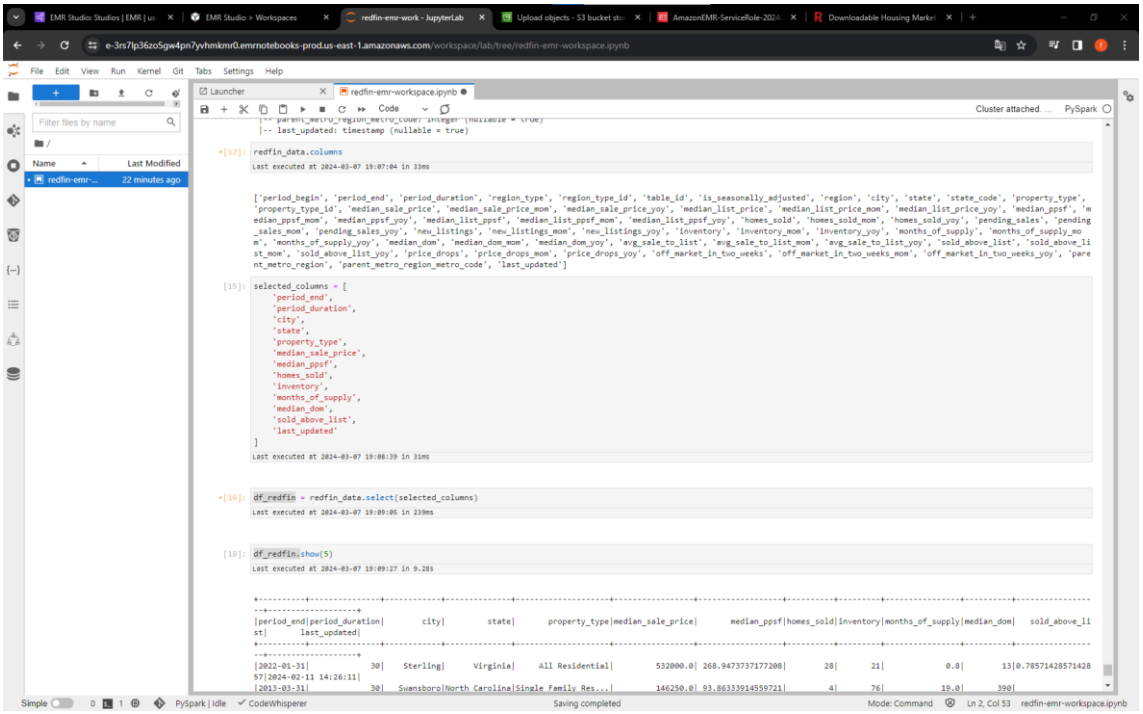
S3 Raw data bucket, file uploaded:

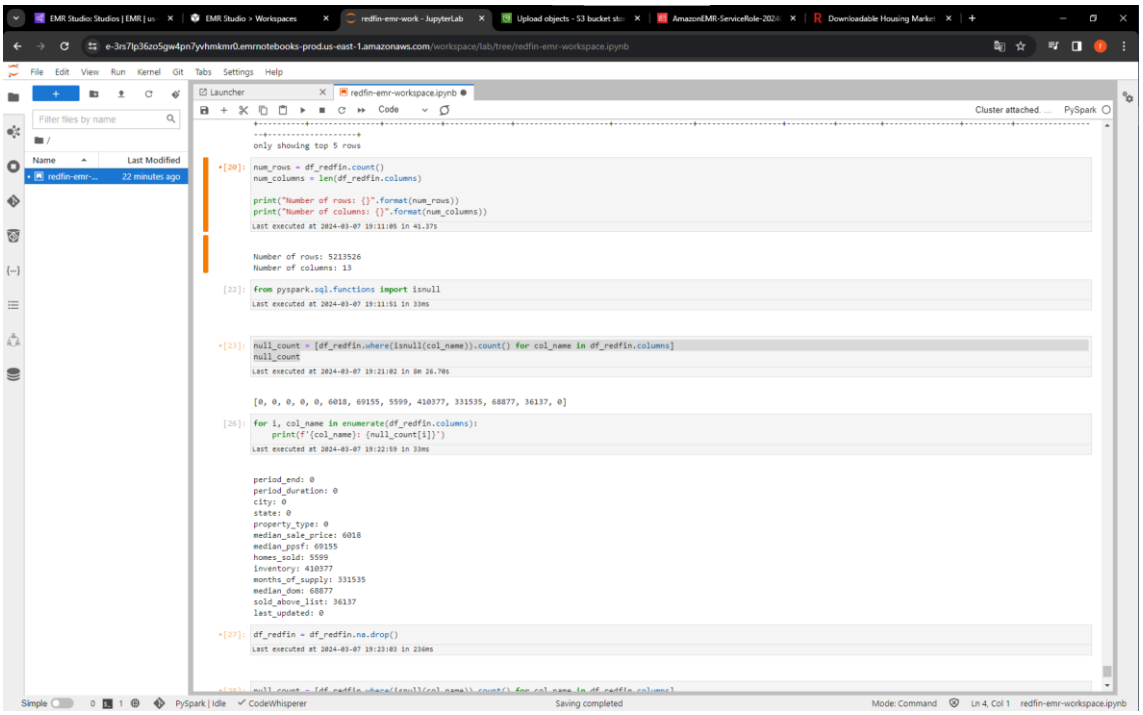


Pyspark Notebook transformations:

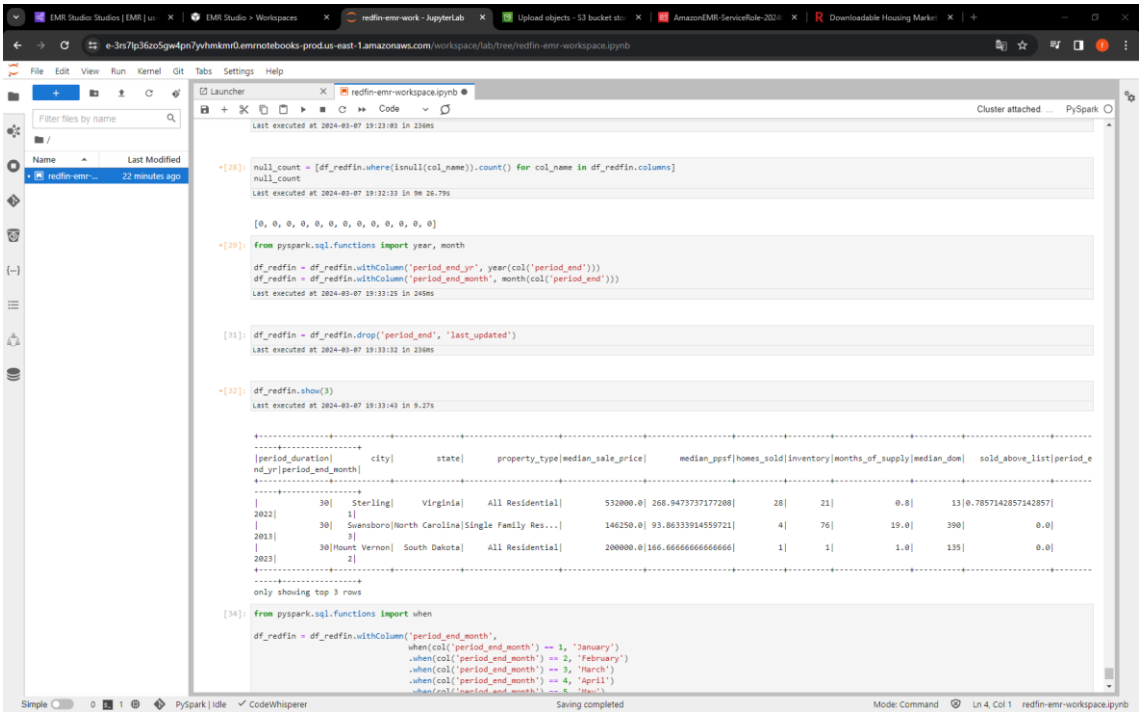



redfin-emr-workspace.ipynb
only showing top 5 rows
[11]: redfin_data.printSchema()
Last executed at 2024-03-07 19:07:01 in 34ms
root
|-- period_begin: date (nullable = true)
|-- period_end: date (nullable = true)
|-- period_duration: integer (nullable = true)
|-- region_type: string (nullable = true)
|-- region_type_id: integer (nullable = true)
|-- table_id: integer (nullable = true)
|-- is_seasonally_adjusted: string (nullable = true)
|-- region: string (nullable = true)
|-- city: string (nullable = true)
|-- state: string (nullable = true)
|-- state_code: string (nullable = true)
|-- property_type: string (nullable = true)
|-- property_type_id: integer (nullable = true)
|-- median_sale_price: double (nullable = true)
|-- median_sale_price_mom: double (nullable = true)
|-- median_sale_price_yoy: double (nullable = true)
|-- median_list_price: double (nullable = true)
|-- median_list_price_mom: double (nullable = true)
|-- median_list_price_yoy: double (nullable = true)
|-- median_ppsf: double (nullable = true)
|-- median_ppsf_mom: double (nullable = true)
|-- median_ppsf_yoy: double (nullable = true)
|-- median_list_ppsf: double (nullable = true)
|-- median_list_ppsf_mom: double (nullable = true)
|-- median_list_ppsf_yoy: double (nullable = true)
|-- homes_sold: integer (nullable = true)
|-- homes_sold_mom: double (nullable = true)
|-- homes_sold_yoy: double (nullable = true)
|-- pending_sales: integer (nullable = true)
|-- pending_sales_mom: double (nullable = true)
|-- pending_sales_yoy: double (nullable = true)
|-- new_listings: integer (nullable = true)
|-- new_listings_mom: double (nullable = true)
|-- new_listings_yoy: double (nullable = true)
|-- inventory: integer (nullable = true)
|-- inventory_mom: double (nullable = true)
|-- inventory_yoy: double (nullable = true)
|-- months_of_supply: double (nullable = true)
|-- months_of_supply_mom: double (nullable = true)
|-- months_of_supply_yoy: double (nullable = true)
|-- median_dom: integer (nullable = true)
|-- median_dom_mom: integer (nullable = true)
|-- median_dom_yoy: integer (nullable = true)

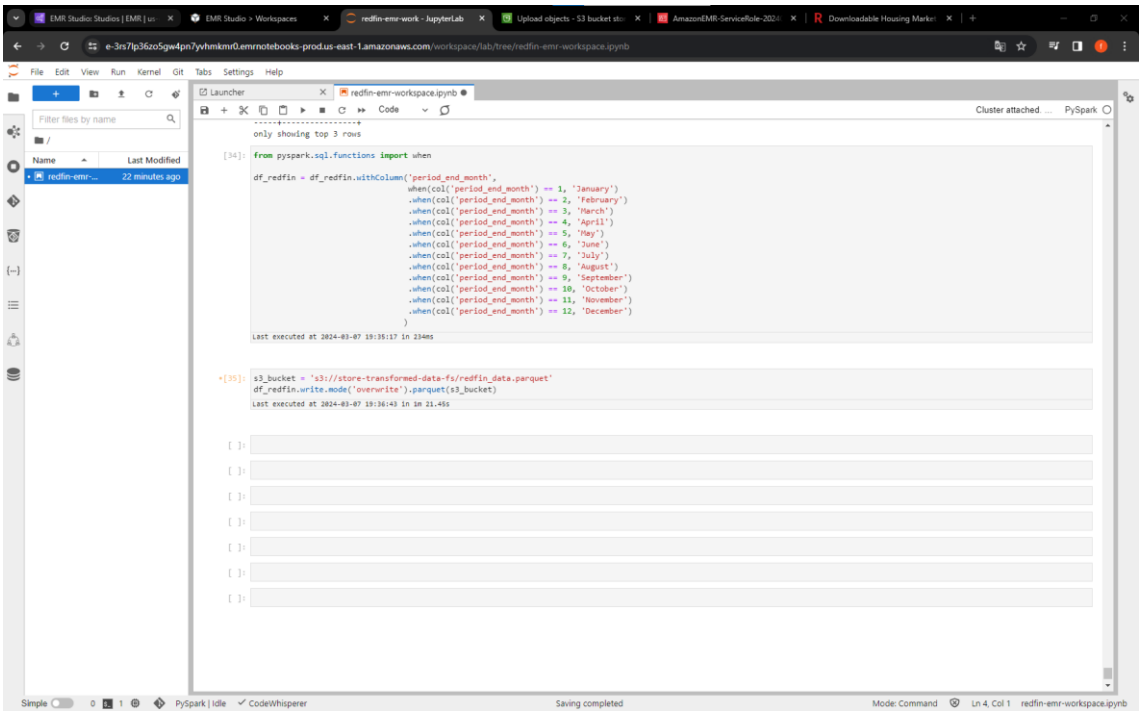

redfin-emr-workspace.ipynb
|-- last_updated: timestamp (nullable = true)
[12]: redfin_data.columns
Last executed at 2024-03-07 19:07:04 in 33ms
['period_begin', 'period_end', 'period_duration', 'region_type', 'region_type_id', 'table_id', 'is_seasonally_adjusted', 'region', 'city', 'state', 'state_code', 'property_type', 'property_type_id', 'median_sale_price', 'median_sale_price_mom', 'median_sale_price_yoy', 'median_list_price', 'median_list_price_mom', 'median_list_price_yoy', 'median_ppsf', 'median_ppsf_mom', 'median_ppsf_yoy', 'median_list_ppsf', 'median_list_ppsf_mom', 'median_list_ppsf_yoy', 'homes_sold', 'homes_sold_mom', 'homes_sold_yoy', 'pending_sales', 'pending_sales_mom', 'pending_sales_yoy', 'new_listings', 'new_listings_mom', 'new_listings_yoy', 'inventory', 'inventory_mom', 'inventory_yoy', 'months_of_supply', 'months_of_supply_mom', 'months_of_supply_yoy', 'median_dom', 'median_dom_mom', 'median_dom_yoy', 'avg_sale_to_list', 'avg_sale_to_list_mom', 'avg_sale_to_list_yoy', 'sold_above_list', 'sold_above_list_mom', 'sold_above_list_yoy', 'price_drops', 'price_drops_mom', 'price_drops_yoy', 'off_market_in_two_weeks', 'off_market_in_two_weeks_mom', 'off_market_in_two_weeks_yoy', 'parent_metro_region', 'parent_metro_region_metro_code', 'last_updated']
[15]: selected_columns = [
'period_end',
'period_duration',
'city',
'state',
'property_type',
'median_sale_price',
'median_ppsf',
'homes_sold',
'inventory',
'months_of_supply',
'median_dom',
'sold_above_list',
'last_updated'
]
Last executed at 2024-03-07 19:08:39 in 31ms
[16]: df_redfin = redfin_data.select(selected_columns)
Last executed at 2024-03-07 19:09:05 in 239ms
[18]: df_redfin.show(5)
Last executed at 2024-03-07 19:09:27 in 9.28s
|period_end|period_duration| city| state| property_type|median_sale_price| median_ppsf|homes_sold|inventory|months_of_supply|median_dom| sold_above_list| last_updated|
|2022-01-31| 30| Sterling| Virginia| All Residential| 532000.0| 268.9473737177208| 28| 21| 0.8| 13|0.78571428571428|2024-02-11 14:26:11|
|2013-03-31| 30| Swansboro|North Carolina|Single Family Res...| 146250.0| 93.86333914559721| 4| 76| 19.0| 390|



```
only showing top 5 rows

[20]: num_rows = df_redfin.count()
      num_columns = len(df_redfin.columns)

      print("Number of rows: {}".format(num_rows))
      print("Number of columns: {}".format(num_columns))
Last executed at 2024-03-07 19:11:05 in 41.37s

Number of rows: 5213526
Number of columns: 13

[22]: from pyspark.sql.functions import isnull
Last executed at 2024-03-07 19:11:51 in 33ms

[23]: null_count = [df_redfin.where(isnull(col_name)).count() for col_name in df_redfin.columns]
      null_count
Last executed at 2024-03-07 19:21:02 in 8m 26.70s

[0, 0, 0, 0, 0, 6018, 69155, 5599, 410377, 331535, 68877, 36137, 0]

[26]: for i, col_name in enumerate(df_redfin.columns):
          print(f'{col_name}: {null_count[i]}')
Last executed at 2024-03-07 19:22:59 in 33ms

period_end: 0
period_duration: 0
city: 0
state: 0
property_type: 0
median_sale_price: 6018
median_ppsf: 69155
homes_sold: 5599
inventory: 410377
months_of_supply: 331535
median_dom: 68877
sold_above_list: 36137
last_updated: 0

[27]: df_redfin = df_redfin.na.drop()
Last executed at 2024-03-07 19:23:03 in 236ms
```



```
Last executed at 2024-03-07 19:23:03 in 236ms

[28]: null_count = [df_redfin.where(isnull(col_name)).count() for col_name in df_redfin.columns]
      null_count
Last executed at 2024-03-07 19:32:33 in 9m 26.79s

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[29]: from pyspark.sql.functions import year, month

      df_redfin = df_redfin.withColumn('period_end_yr', year(col('period_end')))
      df_redfin = df_redfin.withColumn('period_end_month', month(col('period_end')))
Last executed at 2024-03-07 19:33:25 in 245ms

[31]: df_redfin = df_redfin.drop('period_end', 'last_updated')
Last executed at 2024-03-07 19:33:32 in 236ms

[32]: df_redfin.show(3)
Last executed at 2024-03-07 19:33:43 in 9.27s

+---------------+-----------+--------------+--------------------+-----------------+-----------------+---------+--------------+----------+--------------------+-----------+--------------+
|period_duration|       city|         state|       property_type|median_sale_price|       median_ppsf|homes_sold|inventory|months_of_supply|median_dom|   sold_above_list|period_end_yr|period_end_month|
+---------------+-----------+--------------+--------------------+-----------------+-----------------+---------+--------------+----------+--------------------+-----------+--------------+
|             30|    Sterling|      Virginia|    All Residential|         532000.0| 268.9473737177208|       28|          21|       0.8|                13|0.7857142857142857|        2022|             1|
|             30|  Swansboro|North Carolina|Single Family Res...|         146250.0| 93.86333914559721|        4|          76|      19.0|               390|        0.0|        2013|             3|
|             30|Mount Vernon|  South Dakota|    All Residential|         200000.0|166.66666666666666|        1|           1|       1.0|               135|        0.0|        2023|             2|
+---------------+-----------+--------------+--------------------+-----------------+-----------------+---------+--------------+----------+--------------------+-----------+--------------+
only showing top 3 rows

[34]: from pyspark.sql.functions import when

      df_redfin = df_redfin.withColumn('period_end_month',
                                       when(col('period_end_month') == 1, 'January')
                                       .when(col('period_end_month') == 2, 'February')
                                       .when(col('period_end_month') == 3, 'March')
                                       .when(col('period_end_month') == 4, 'April')
```



S3 Transformed bucket: