TEAM – 4

ADTA 5240 PROJECT

DOCUMENT WITH SCREENSHOTS

PRESENTED BY:

Abhiram Karthik Meduri

Abhiram Nallamothu

Pavan Kola

Jacob Larkin

Two Datasets:

Static dataset: https://data.cityofdenton.com/dataset/traffic-closed-cases/resource/b12cba00-24e4-4f45-89f8-ffd1c0bf6b95

Dynamic dataset: https://data.cityofdenton.com/dataset/denton-crime-data

WORKING WITH DYNAMIC DATASET

1. Fetching Data from the City of Denton API

Type of API Request: HTTP GET Request

Explanation:

So, what we're doing here is pulling some data from the City of Denton Open Data portal using a public API. Instead of grabbing everything in one go (which would be a lot and could easily overload the system), we're using SQL queries to get smaller, more manageable chunks. The API lets us use SQL right at the endpoint (https://data.cityofdenton.com/api/3/action/datastore_search_sql), which makes interacting with the dataset super straightforward.

Steps:

1. Build the SQL Query:

   - First, we put together an SQL query that pulls a set number of rows—10,000 at a time—and uses an OFFSET to make sure we're paging through the data step-by-step.

Sql:

```
SELECT * FROM "RESOURCE_ID" LIMIT 10000 OFFSET
```

   - Here, RESOURCE_ID represents the unique ID for the dataset we're working with in the City of Denton's data portal.
   - The LIMIT tells it to grab just 10,000 rows in each request. The OFFSET helps in fetching these rows in batches, so we don't end up asking for too much data all at once.

2. Send the GET Request:

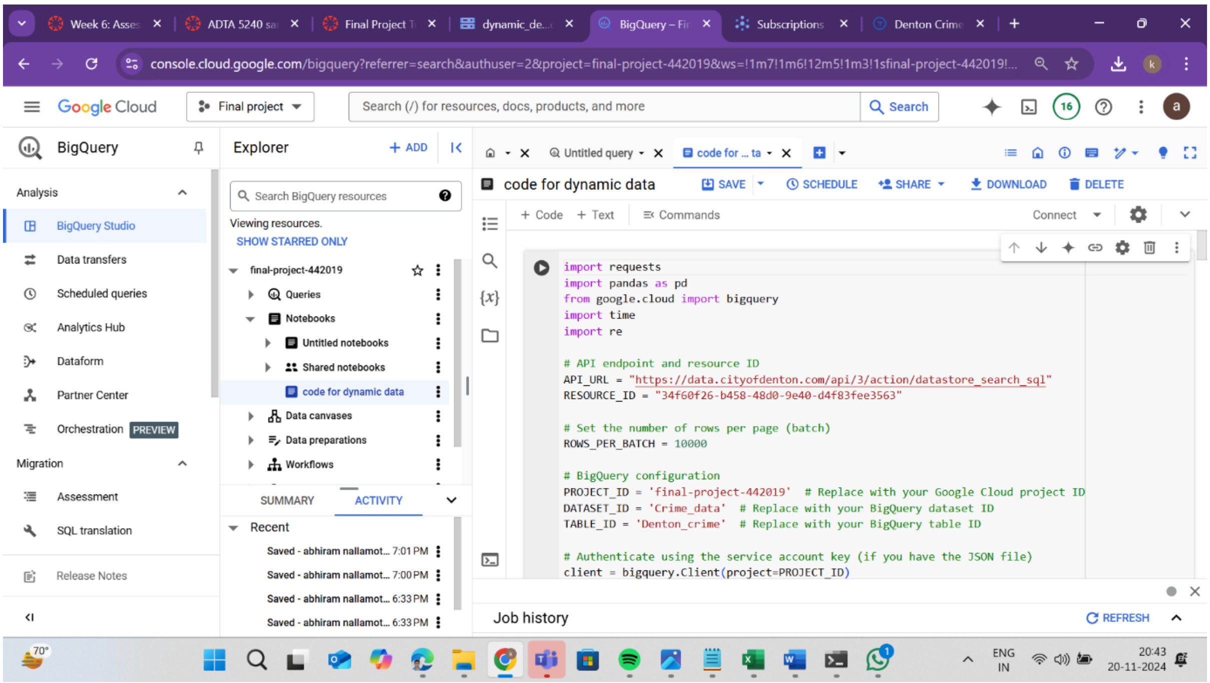
   - After getting the SQL query ready, we need to send it to the API. We're doing this using Python's requests.get() function:
   - Here, API_URL is the link to the API, and params includes the SQL query (params = {'sql': SQL_QUERY}) we're sending.
   - This function sends the request to the API and waits for it to respond.

3. Check the Response:

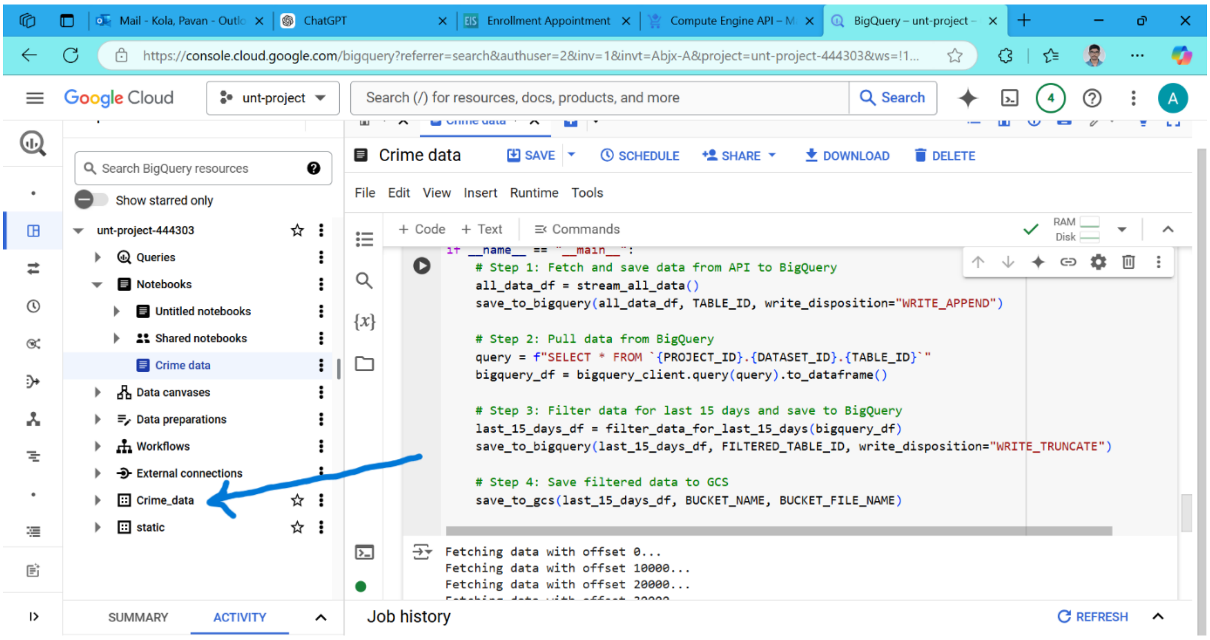
- Once we get a response, we need to make sure everything went okay.
- A status code of 200 means it worked! If that's the case, we convert the response from JSON into a format that Python can work with easily.
- The records key is where we find all the rows of data we just pulled.

Outcome: By the end of these steps, we've got the data in a JSON format that contains all the crime records. Now it's ready for us to do something useful with it, like analysis or processing.

DYNAMIC DATA PULLING IN BIGQUERY



```
import requests
import pandas as pd
from google.cloud import bigquery
import time
import re

# API endpoint and resource ID
API_URL = "https://data.cityofdenton.com/api/3/action/datastore_search_sql"
RESOURCE_ID = "34f60f26-b458-48d0-9e40-d4f83fee3563"

# Set the number of rows per page (batch)
ROWS_PER_BATCH = 10000

# BigQuery configuration
PROJECT_ID = 'final-project-442019'  # Replace with your Google Cloud project ID
DATASET_ID = 'Crime_data'  # Replace with your BigQuery dataset ID
TABLE_ID = 'Denton_crime'  # Replace with your BigQuery table ID

# Authenticate using the service account key (if you have the JSON file)
client = bigquery.Client(project=PROJECT_ID)
```



```
if __name__ == "__main__":
    # Step 1: Fetch and save data from API to BigQuery
    all_data_df = stream_all_data()
    save_to_bigquery(all_data_df, TABLE_ID, write_disposition="WRITE_APPEND")

    # Step 2: Pull data from BigQuery
    query = f"SELECT * FROM `{PROJECT_ID}.{DATASET_ID}.{TABLE_ID}`"
    bigquery_df = bigquery_client.query(query).to_dataframe()

    # Step 3: Filter data for last 15 days and save to BigQuery
    last_15_days_df = filter_data_for_last_15_days(bigquery_df)
    save_to_bigquery(last_15_days_df, FILTERED_TABLE_ID, write_disposition="WRITE_TRUNCATE")

    # Step 4: Save filtered data to GCS
    save_to_gcs(last_15_days_df, BUCKET_NAME, BUCKET_FILE_NAME)
```

```
Fetching data with offset 0...
Fetching data with offset 10000...
Fetching data with offset 20000...
```

2. Pagination (Handling Large Data)

Type of Request: GET Request with Pagination

Explanation:

When we're dealing with a lot of data, it's important not to try and grab everything at once. If we do, it could easily overwhelm the server or even cause our system to run out of memory. Instead, we use pagination—basically breaking the data into smaller chunks called pages. This way, we can pull the data piece by piece and make sure everything runs smoothly.

To do this, we use an OFFSET in our SQL query, which tells us where to start getting data from in the dataset.

Steps:

1. Set Initial Offset:
   - We start with an offset of 0, which means we're going to start pulling records from the beginning of the dataset.
2. Fetch Data in Batches:
   - To get the data in chunks, we use a loop (while True) to keep fetching 10,000 rows at a time.
3. Update Offset for Next Batch:
   - After pulling one batch, we update the offset by adding 10,000 (the number of rows in each batch) so that the next time around, we get the next set of rows.
4. Break on Last Page:
   - If the number of records we get is less than the batch size (i.e., fewer than 10,000 rows), that means we've reached the end, so we stop the loop.
5. Sleep to Simulate Streaming:
   - We add a time.sleep(2) call, which pauses the program for a couple of seconds between batches. This helps simulate a real-time streaming scenario where there's a small delay between each fetch.

Outcome: Using pagination like this, we can pull the data in manageable chunks without running into memory issues. This makes the whole process a lot more efficient and prevents overloading the system.

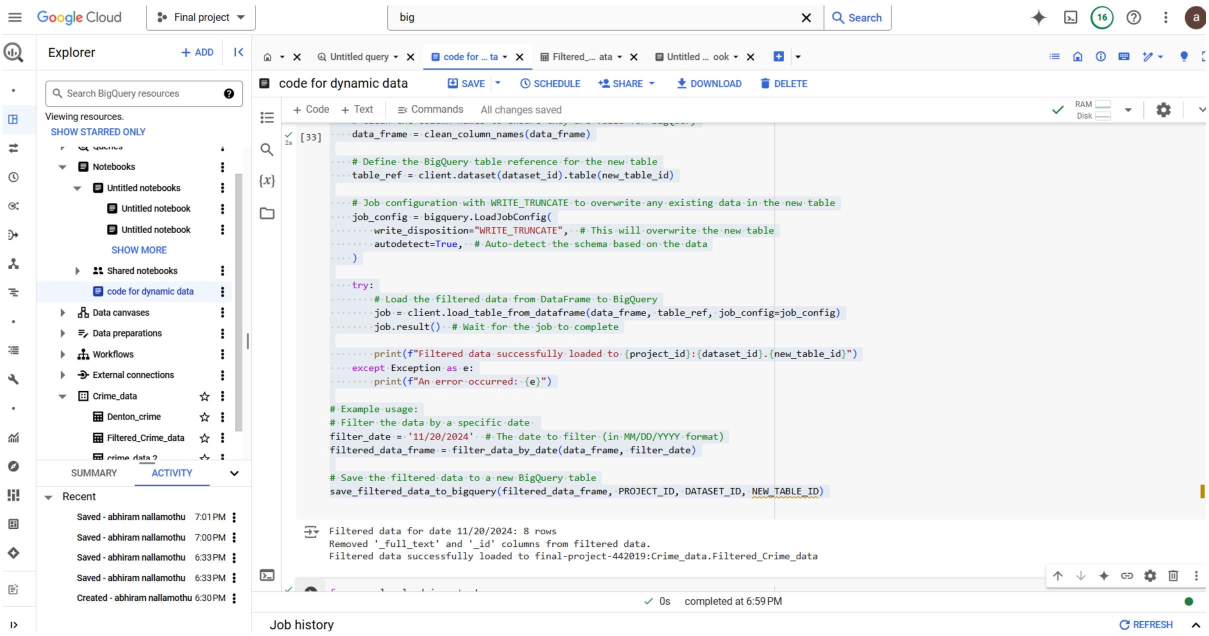
3. Cleaning Column Names

Type of Data Transformation: Data Normalization (Column Name Sanitization)

Explanation:

When we get data from the API, the column names might have spaces or special characters that don’t work well in BigQuery. BigQuery has strict rules for column names, so we need to clean them up to make sure they’re valid. To do that, we use a function called clean_column_names that replaces any invalid characters with an underscore (_).

Outcome: With the column names sanitized, the dataset becomes fully compatible with BigQuery and can be uploaded without issues. This step ensures that our data is clean, standardized, and ready for further processing.



```
[33]
    data_frame = clean_column_names(data_frame)

    # Define the BigQuery table reference for the new table
    table_ref = client.dataset(dataset_id).table(new_table_id)

    # Job configuration with WRITE_TRUNCATE to overwrite any existing data in the new table
    job_config = bigquery.LoadJobConfig(
        write_disposition="WRITE_TRUNCATE",  # This will overwrite the new table
        autodetect=True,  # Auto-detect the schema based on the data
    )

    try:
        # Load the filtered data from DataFrame to BigQuery
        job = client.load_table_from_dataframe(data_frame, table_ref, job_config=job_config)
        job.result()  # Wait for the job to complete

        print(f"Filtered data successfully loaded to {project_id}:{dataset_id}.{new_table_id}")
    except Exception as e:
        print(f"An error occurred: {e}")

# Example usage:
# Filter the data by a specific date
filter_date = '11/20/2024'  # The date to filter (in MM/DD/YYYY format)
filtered_data_frame = filter_data_by_date(data_frame, filter_date)

# Save the filtered data to a new BigQuery table
save_filtered_data_to_bigquery(filtered_data_frame, PROJECT_ID, DATASET_ID, NEW_TABLE_ID)
```

```
Filtered data for date 11/20/2024: 8 rows
Removed '_full_text' and '_id' columns from filtered data.
Filtered data successfully loaded to final-project-442019:Crime_data.Filtered_Crime_data
```

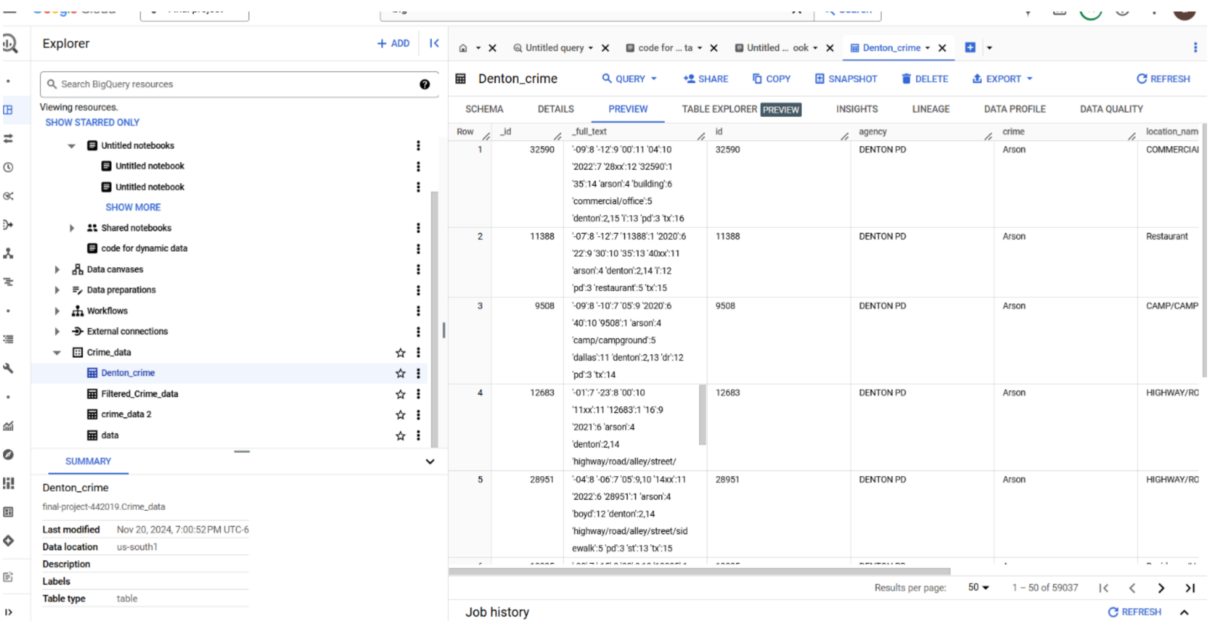
4. Saving Data to BigQuery

Type of Request: BigQuery Load Job API (using load_table_from_dataframe)

Explanation:

Once we have the data cleaned and ready, the next step is to upload it to BigQuery. To do this, we use the BigQuery API, specifically the load_table_from_dataframe() function, which lets us take a pandas DataFrame and load it straight into a BigQuery table.

Outcome: Once everything is complete, the data is in BigQuery and ready to be queried or analyzed further. This makes it really easy to run all kinds of SQL queries on the data, creating reports or doing in-depth analysis.



Denton_crime

final-project-442019.Crime_data

Last modified	Nov 20, 2024, 7:00:52 PM UTC-6
Data location	us-south1
Description	
Labels	
Table type	table

Row	_id	_full_text	id	agency	crime	location_nam
1	32590	'-09':8 '-12':9 '00':11 '04':10 '2022':7 '28xx':12 '32590':1 '35':14 'arson':4 'building':6 'commercial/office':5 'denton':2,15 'i':13 'pd':3 'tx':16	32590	DENTON PD	Arson	COMMERCIAl
2	11388	'-07':8 '-12':7 '11388':1 '2020':6 '22':9 '30':10 '35':13 '40xx':11 'arson':4 'denton':2,14 'i':12 'pd':3 'restaurant':5 'tx':15	11388	DENTON PD	Arson	Restaurant
3	9508	'-09':8 '-10':7 '05':9 '2020':6 '40':10 '9508':1 'arson':4 'camp/campground':5 'dallas':11 'denton':2,13 'dr':12 'pd':3 'tx':14	9508	DENTON PD	Arson	CAMP/CAMP
4	12683	'-01':7 '-23':8 '00':10 '11xx':11 '12683':1 '16':9 '2021':6 'arson':4 'denton':2,14 'highway/road/alley/street/	12683	DENTON PD	Arson	HIGHWAY/RO
5	28951	'-04':8 '-06':7 '05':9,10 '14xx':11 '2022':6 '28951':1 'arson':4 'boyd':12 'denton':2,14 'highway/road/alley/street/sidewalk':5 'pd':3 'st':13 'tx':15	28951	DENTON PD	Arson	HIGHWAY/RO

Results per page: 50 1 – 50 of 59037

5. Filtering Data by Date

Type of Data Transformation: Time-Based Filtering

Explanation:

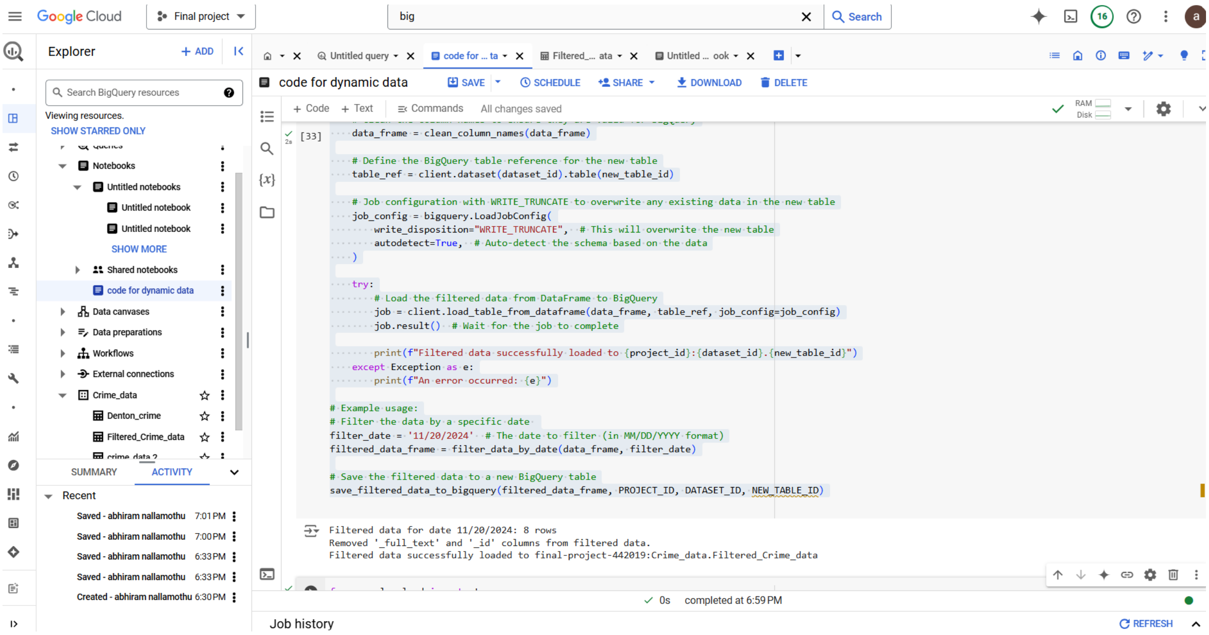
Here, we’re working on filtering the data to focus on records from a specific date. This means we’re going to use the date_time column in our dataset and convert it to a proper datetime object. Once that’s done, we can filter it down to match a particular date that we care about.

Steps:

1. Convert date_time to Datetime:
   - The first thing we need to do is take the date_time column and convert it from a string into an actual datetime object that Python can work with.
2. Filter by Date:

- Now that we have proper datetime objects, we can filter the data to keep only the rows where the date matches the specific date we're interested in.

Outcome: After filtering, we end up with a dataset that only contains records from the specified date. This makes it much more manageable and focuses our analysis on the relevant time period.



```
    data_frame = clean_column_names(data_frame)

    # Define the BigQuery table reference for the new table
    table_ref = client.dataset(dataset_id).table(new_table_id)

    # Job configuration with WRITE_TRUNCATE to overwrite any existing data in the new table
    job_config = bigquery.LoadJobConfig(
        write_disposition="WRITE_TRUNCATE",  # This will overwrite the new table
        autodetect=True,  # Auto-detect the schema based on the data
    )

    try:
        # Load the filtered data from DataFrame to BigQuery
        job = client.load_table_from_dataframe(data_frame, table_ref, job_config=job_config)
        job.result()  # Wait for the job to complete

        print(f"Filtered data successfully loaded to {project_id}:{dataset_id}.{new_table_id}")
    except Exception as e:
        print(f"An error occurred: {e}")

# Example usage:
# Filter the data by a specific date
filter_date = '11/20/2024'  # The date to filter (in MM/DD/YYYY format)
filtered_data_frame = filter_data_by_date(data_frame, filter_date)

# Save the filtered data to a new BigQuery table
save_filtered_data_to_bigquery(filtered_data_frame, PROJECT_ID, DATASET_ID, NEW_TABLE_ID)
```

```
Filtered data for date 11/20/2024: 8 rows
Removed '_full_text' and '_id' columns from filtered data.
Filtered data successfully loaded to final-project-442019:Crime_data.Filtered_Crime_data
```

6. Saving Filtered Data to BigQuery

Type of Request: BigQuery Load Job API (using load_table_from_dataframe)

Explanation:

This step is pretty much like the previous upload to BigQuery, but instead of the entire dataset, we're focusing on just the filtered data this time. The idea is to take that smaller, more relevant subset and store it in a new table in BigQuery.

Steps:

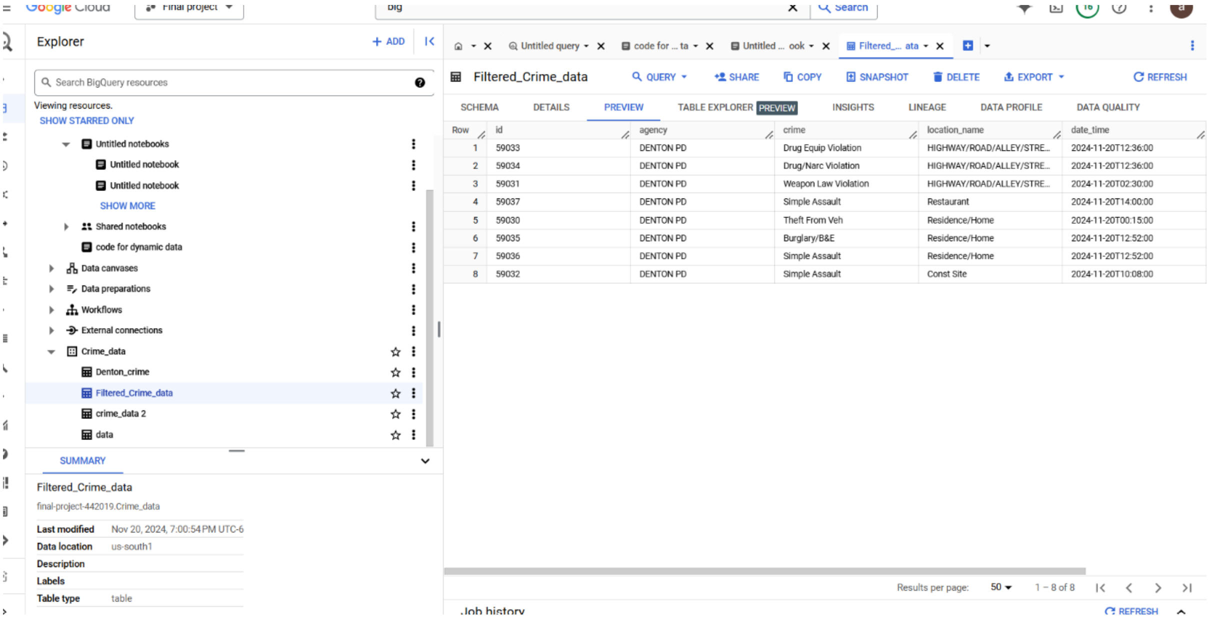
1. Clean Column Names:

- Before uploading, we go ahead and clean the column names again to make sure they're valid for BigQuery. This just means getting rid of any spaces or special characters that BigQuery doesn't like.

2. Load Filtered Data:

- Once the column names are good to go, we use load_table_from_dataframe() to upload the filtered data to BigQuery, just like we did before.

Outcome: By the end of this step, the filtered dataset is stored in BigQuery. Now it’s ready for us to run queries on or do some more analysis—focusing just on the specific slice of data we’re interested in.



Filtered_Crime_data

Row	id	agency	crime	location_name	date_time
1	59033	DENTON PD	Drug Equip Violation	HIGHWAY/ROAD/ALLEY/STRE...	2024-11-20T12:36:00
2	59034	DENTON PD	Drug/Narc Violation	HIGHWAY/ROAD/ALLEY/STRE...	2024-11-20T12:36:00
3	59031	DENTON PD	Weapon Law Violation	HIGHWAY/ROAD/ALLEY/STRE...	2024-11-20T02:30:00
4	59037	DENTON PD	Simple Assault	Restaurant	2024-11-20T14:00:00
5	59030	DENTON PD	Theft From Veh	Residence/Home	2024-11-20T00:15:00
6	59035	DENTON PD	Burglary/B&E	Residence/Home	2024-11-20T12:52:00
7	59036	DENTON PD	Simple Assault	Residence/Home	2024-11-20T12:52:00
8	59032	DENTON PD	Simple Assault	Const Site	2024-11-20T10:08:00

Filtered_Crime_data
final-project-442019.Crime_data

Last modified	Nov 20, 2024, 7:00:54 PM UTC-6
Data location	us-south1
Description	
Labels	
Table type	table

7. Uploading Filtered Data to Google Cloud Storage (GCS)

Type of API Request: GCS Blob Upload API

Explanation:

Once we have the filtered data, the next step is to save it as a CSV file and upload it to Google Cloud Storage (GCS). This step makes it easy to share the data with others or use it for further processing in different tools or applications. It’s basically about having a convenient backup or making the data accessible outside of BigQuery.

Steps:

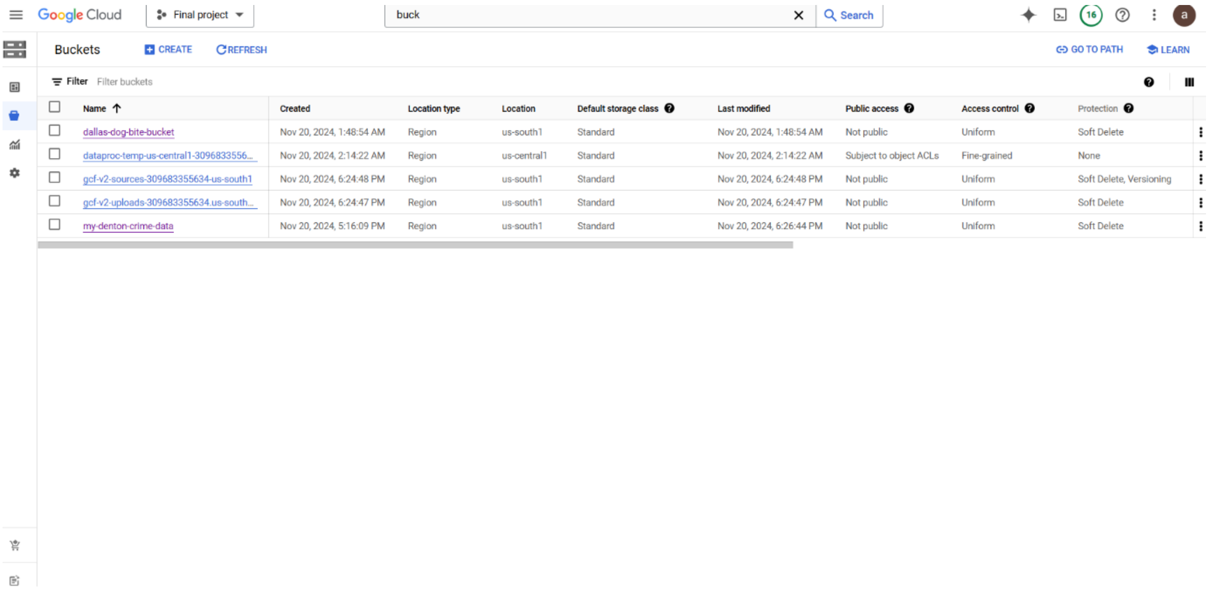
1. Save Filtered Data as CSV:

   - The filtered data is saved locally as a CSV file, which is a simple, widely-used format that most tools can understand.

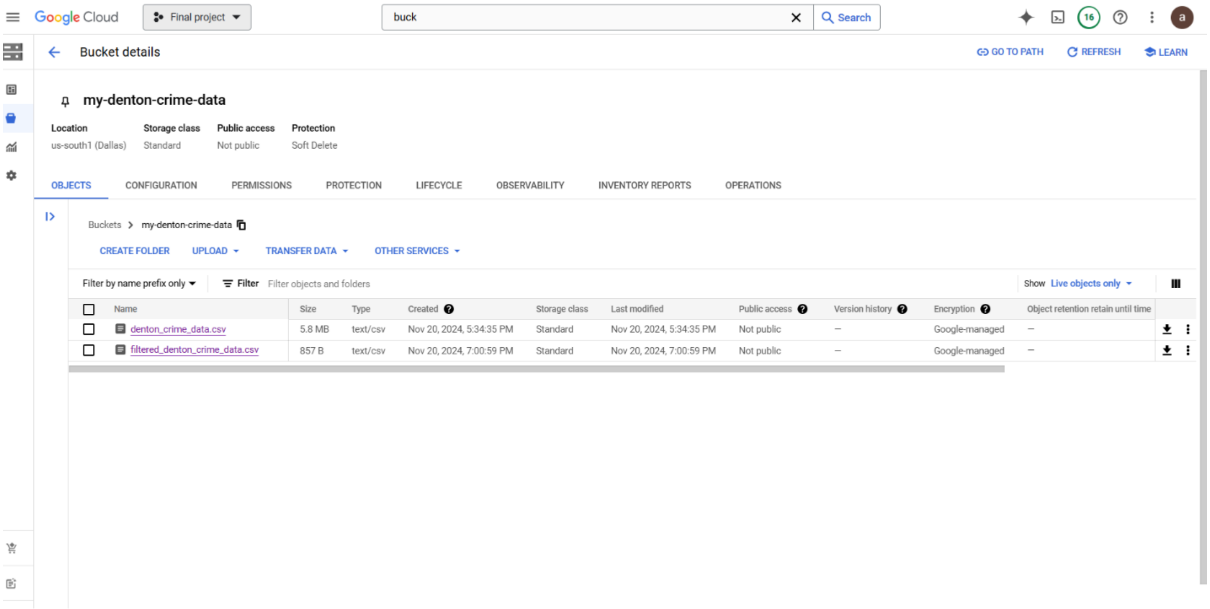
2. Upload to GCS:

   - After saving it, the CSV is uploaded to a GCS bucket. This means we can easily access it later, share it, or use it in other applications for more analysis.

Outcome: Now the filtered data is safely stored in Google Cloud Storage, ready for sharing or further use. It gives us flexibility—whether it's for archiving purposes, sharing with team members, or using it in other workflows.



☐	Name ↑	Created	Location type	Location	Default storage class	Last modified	Public access	Access control	Protection
☐	dallas-dog-bite-bucket	Nov 20, 2024, 1:48:54 AM	Region	us-south1	Standard	Nov 20, 2024, 1:48:54 AM	Not public	Uniform	Soft Delete
☐	dataproc-temp-us-central1-3096833556...	Nov 20, 2024, 2:14:22 AM	Region	us-central1	Standard	Nov 20, 2024, 2:14:22 AM	Subject to object ACLs	Fine-grained	None
☐	gcf-v2-sources-309683355634-us-south1	Nov 20, 2024, 6:24:48 PM	Region	us-south1	Standard	Nov 20, 2024, 6:24:48 PM	Not public	Uniform	Soft Delete, Versioning
☐	gcf-v2-uploads-309683355634.us-south...	Nov 20, 2024, 6:24:47 PM	Region	us-south1	Standard	Nov 20, 2024, 6:24:47 PM	Not public	Uniform	Soft Delete
☐	my-denton-crime-data	Nov 20, 2024, 5:16:09 PM	Region	us-south1	Standard	Nov 20, 2024, 6:26:44 PM	Not public	Uniform	Soft Delete



my-denton-crime-data

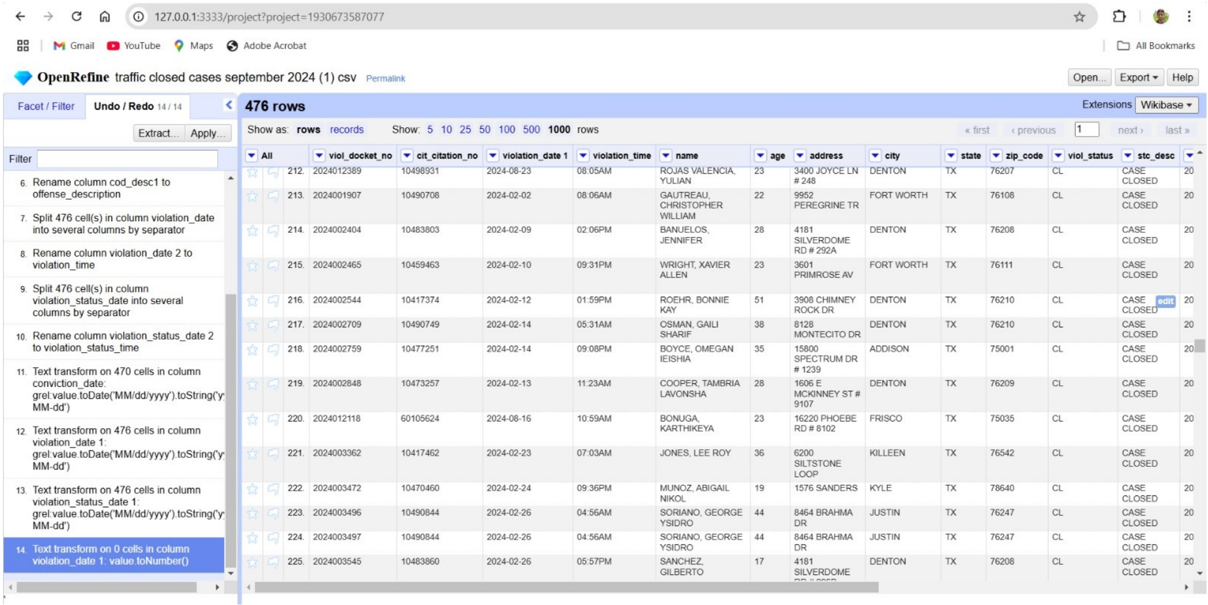
Location	Storage class	Public access	Protection
us-south1 (Dallas)	Standard	Not public	Soft Delete

☐	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Object retention retain until time
☐	denton_crime_data.csv	5.8 MB	text/csv	Nov 20, 2024, 5:34:35 PM	Standard	Nov 20, 2024, 5:34:35 PM	Not public	–	Google-managed	–
☐	filtered_denton_crime_data.csv	857 B	text/csv	Nov 20, 2024, 7:00:59 PM	Standard	Nov 20, 2024, 7:00:59 PM	Not public	–	Google-managed	–

WORKING WITH STATIC DATASET

Dataset: Traffic closed cases in denton-septmber 2024

STEP 1:



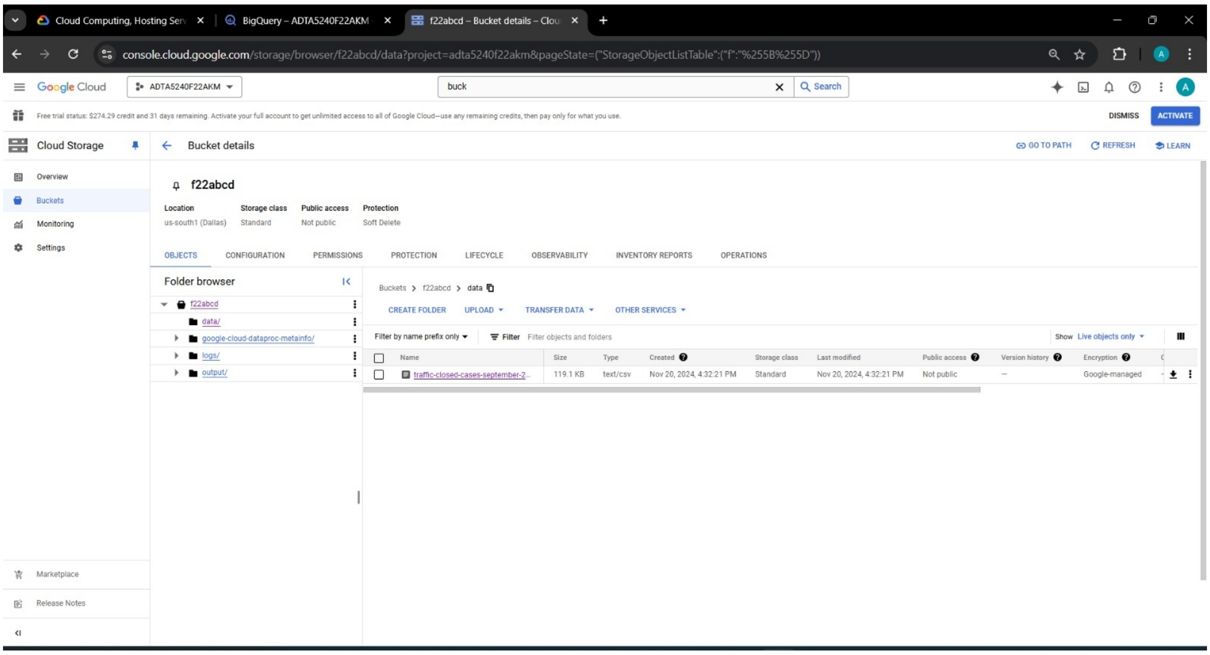
6. Rename column cod_desc1 to offense_description
7. Split 476 cell(s) in column violation_date into several columns by separator
8. Rename column violation_date 2 to violation_time
9. Split 476 cell(s) in column violation_status_date into several columns by separator
10. Rename column violation_status_date 2 to violation_status_time
11. Text transform on 470 cells in column conviction_date: grel:value.toDate('MM/dd/yyyy').toString('y MM-dd')
12. Text transform on 476 cells in column violation_date 1: grel:value.toDate('MM/dd/yyyy').toString('y MM-dd')
13. Text transform on 476 cells in column violation_status_date 1: grel:value.toDate('MM/dd/yyyy').toString('y MM-dd')
14. Text transform on 0 cells in column violation_date 1: value.toNumber()

All	viol_docket_no	cit_citation_no	violation_date 1	violation_time	name	age	address	city	state	zip_code	viol_status	stc_desc	
212.	2024012389	10498931	2024-08-23	08:05AM	ROJAS VALENCIA, YULIAN	23	3400 JOYCE LN # 248	DENTON	TX	76207	CL	CASE CLOSED	20
213.	2024001907	10490708	2024-02-02	08:06AM	GAUTREAU, CHRISTOPHER WILLIAM	22	9952 PEREGRINE TR	FORT WORTH	TX	76108	CL	CASE CLOSED	20
214.	2024002404	10483803	2024-02-09	02:06PM	BANUELOS, JENNIFER	28	4181 SILVERDOME RD # 292A	DENTON	TX	76208	CL	CASE CLOSED	20
215.	2024002465	10459463	2024-02-10	09:31PM	WRIGHT, XAVIER ALLEN	23	3601 PRIMROSE AV	FORT WORTH	TX	76111	CL	CASE CLOSED	20
216.	2024002544	10417374	2024-02-12	01:59PM	ROEHR, BONNIE KAY	51	3908 CHIMNEY ROCK DR	DENTON	TX	76210	CL	CASE CLOSED edit	20
217.	2024002709	10490749	2024-02-14	05:31AM	OSMAN, GAILI SHARIF	38	8128 MONTECITO DR	DENTON	TX	76210	CL	CASE CLOSED	20
218.	2024002759	10477251	2024-02-14	09:08PM	BOYCE, OMEGAN IEISHIA	35	15800 SPECTRUM DR # 1239	ADDISON	TX	75001	CL	CASE CLOSED	20
219.	2024002848	10473257	2024-02-13	11:23AM	COOPER, TAMBRIA LAVONSHA	28	1606 E MCKINNEY ST # 9107	DENTON	TX	76209	CL	CASE CLOSED	20
220.	2024012118	60105624	2024-08-16	10:59AM	BONUGA, KARTHIKEYA	23	16220 PHOEBE RD # 8102	FRISCO	TX	75035	CL	CASE CLOSED	20
221.	2024003362	10417462	2024-02-23	07:03AM	JONES, LEE ROY	36	6200 SILTSTONE LOOP	KILLEEN	TX	76542	CL	CASE CLOSED	20
222.	2024003472	10470460	2024-02-24	09:36PM	MUNOZ, ABIGAIL NIKOL	19	1576 SANDERS	KYLE	TX	78640	CL	CASE CLOSED	20
223.	2024003496	10490844	2024-02-26	04:56AM	SORIANO, GEORGE YSIDRO	44	8464 BRAHMA DR	JUSTIN	TX	76247	CL	CASE CLOSED	20
224.	2024003497	10490844	2024-02-26	04:56AM	SORIANO, GEORGE YSIDRO	44	8464 BRAHMA DR	JUSTIN	TX	76247	CL	CASE CLOSED	20
225.	2024003545	10483860	2024-02-26	05:57PM	SANCHEZ, GILBERTO	17	4181 SILVERDOME	DENTON	TX	76208	CL	CASE CLOSED	20

Here, we cleaned the datset in openrefine.

Step 2:

Now, we uploaded this datset in the bucket that we created in the google cloud platform.



console.cloud.google.com/storage/browser/f22abcd/data?project=adta5240f22akm&pageState=("StorageObjectListTable":("f":"%255B%255D"))

buck

Free trial status: $274.29 credit and 31 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

Bucket details

f22abcd

Location	Storage class	Public access	Protection
us-south1 (Dallas)	Standard	Not public	Soft Delete

Folder browser

- f22abcd
  - data/
  - google-cloud-dataproc-metainfo/
  - logs/
  - output/

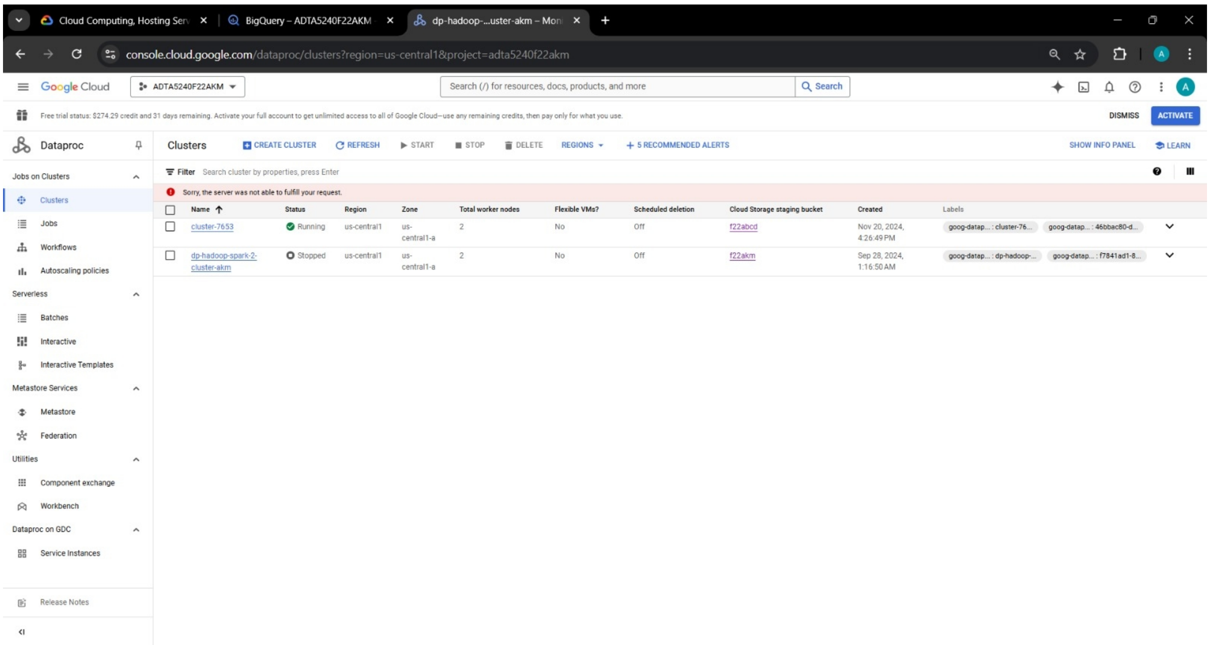
Buckets > f22abcd > data

Filter by name prefix only ▾ Filter Filter objects and folders Show Live objects only ▾

□	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption
□	traffic-closed-cases-september-2...	119.1 KB	text/csv	Nov 20, 2024, 4:32:21 PM	Standard	Nov 20, 2024, 4:32:21 PM	Not public	—	Google-managed

Step 3:

We created a cluster and then uploaded the bucket and then opened SSH window from VM instances.



console.cloud.google.com/dataproc/clusters?region=us-central1&project=adta5240f22akm

Search (/) for resources, docs, products, and more

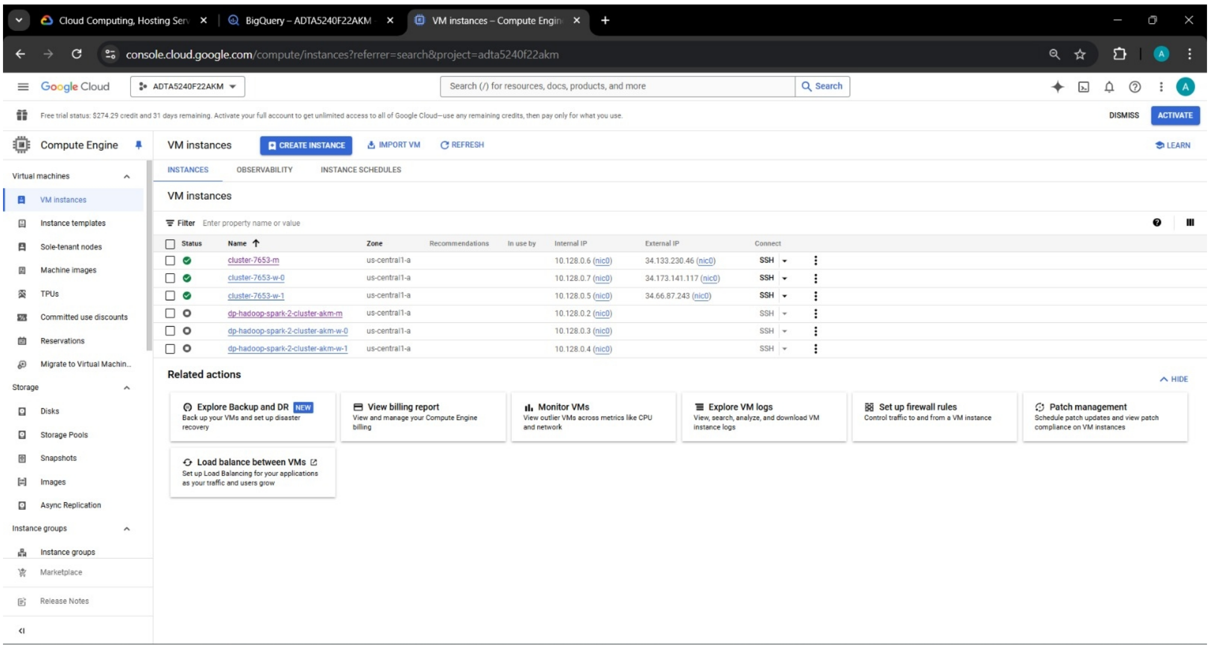
Free trial status: $274.29 credit and 31 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

Clusters

Filter Search cluster by properties, press Enter

Sorry, the server was not able to fulfill your request.

□	Name ↑	Status	Region	Zone	Total worker nodes	Flexible VMs?	Scheduled deletion	Cloud Storage staging bucket	Created	Labels
□	cluster-7653	Running	us-central1	us-central1-a	2	No	Off	f22abcd	Nov 20, 2024, 4:26:49 PM	goog-datap... : cluster-76... goog-datap... : 46bbac80-d...
□	dp-hadoop-spark-2-cluster-akm	Stopped	us-central1	us-central1-a	2	No	Off	f22akm	Sep 28, 2024, 1:16:50 AM	goog-datap... : dp-hadoop-... goog-datap... : f7841ad1-8...



Cloud Computing, Hosting Serv
BigQuery – ADTA5240F22AKM
VM instances – Compute Engin
console.cloud.google.com/compute/instances?referrer=search&project=adta5240f22akm
Google Cloud
ADTA5240F22AKM
Search (/) for resources, docs, products, and more
Search
Free trial status: $274.29 credit and 31 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.
DISMISS
ACTIVATE
Compute Engine
VM instances
CREATE INSTANCE
IMPORT VM
REFRESH
LEARN
Virtual machines
VM instances
Instance templates
Sole-tenant nodes
Machine images
TPUs
Committed use discounts
Reservations
Migrate to Virtual Machin...
Storage
Disks
Storage Pools
Snapshots
Images
Async Replication
Instance groups
Instance groups
Marketplace
Release Notes
INSTANCES
OBSERVABILITY
INSTANCE SCHEDULES
Filter
Enter property name or value
Related actions
HIDE
Explore Backup and DR
NEW
Back up your VMs and set up disaster recovery
View billing report
View and manage your Compute Engine billing
Monitor VMs
View outlier VMs across metrics like CPU and network
Explore VM logs
View, search, analyze, and download VM instance logs
Set up firewall rules
Control traffic to and from a VM instance
Patch management
Schedule patch updates and view patch compliance on VM instances
Load balance between VMs
Set up Load Balancing for your applications as your traffic and users grow

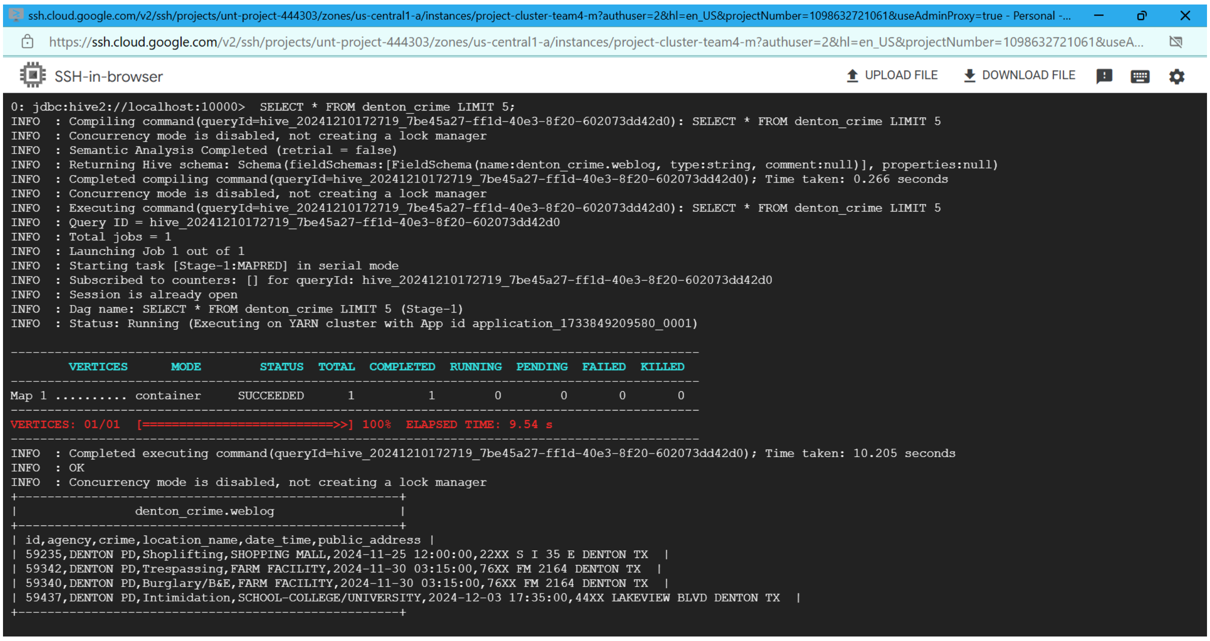
VM instances

□	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
□	✓	cluster-7653-m	us-central1-a			10.128.0.6 (nic0)	34.133.230.46 (nic0)	SSH
□	✓	cluster-7653-w-0	us-central1-a			10.128.0.7 (nic0)	34.173.141.117 (nic0)	SSH
□	✓	cluster-7653-w-1	us-central1-a			10.128.0.5 (nic0)	34.66.87.243 (nic0)	SSH
□	○	dp-hadoop-spark-2-cluster-akm-m	us-central1-a			10.128.0.2 (nic0)		SSH
□	○	dp-hadoop-spark-2-cluster-akm-w-0	us-central1-a			10.128.0.3 (nic0)		SSH
□	○	dp-hadoop-spark-2-cluster-akm-w-1	us-central1-a			10.128.0.4 (nic0)		SSH

Queries in Hive (Dynamic dataset)

A total of 342 rows of data is pulled from the streaming data (Denton crime cases) from 25 November.

1. Created a table

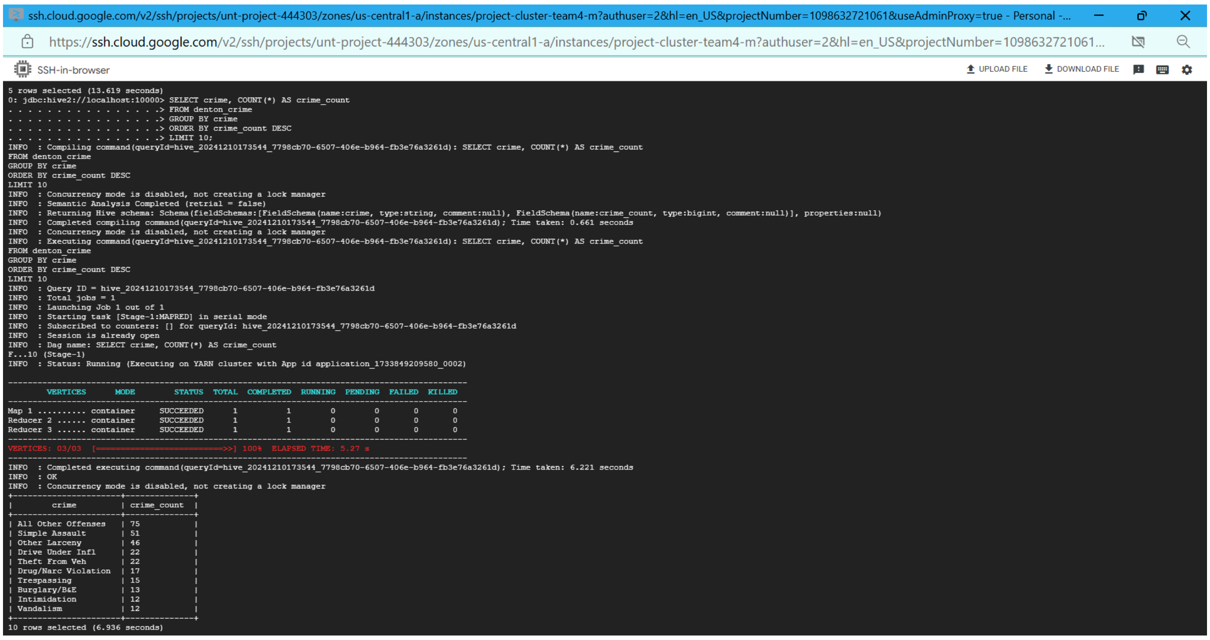


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true - Personal -...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useA...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
0: jdbc:hive2://localhost:10000>  SELECT * FROM denton_crime LIMIT 5;
INFO  : Compiling command(queryId=hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0): SELECT * FROM denton_crime LIMIT 5
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:denton_crime.weblog, type:string, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0); Time taken: 0.266 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0): SELECT * FROM denton_crime LIMIT 5
INFO  : Query ID = hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0
INFO  : Session is already open
INFO  : Dag name: SELECT * FROM denton_crime LIMIT 5 (Stage-1)
INFO  : Status: Running (Executing on YARN cluster with App id application_1733849209580_0001)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 9.54 s
----------------------------------------------------------------------------------------------
INFO  : Completed executing command(queryId=hive_20241210172719_7be45a27-ff1d-40e3-8f20-602073dd42d0); Time taken: 10.205 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+----------------------------------------------------+
|                 denton_crime.weblog                |
+----------------------------------------------------+
| id,agency,crime,location_name,date_time,public_address |
| 59235,DENTON PD,Shoplifting,SHOPPING MALL,2024-11-25 12:00:00,22XX S I 35 E DENTON TX  |
| 59342,DENTON PD,Trespassing,FARM FACILITY,2024-11-30 03:15:00,76XX FM 2164 DENTON TX  |
| 59340,DENTON PD,Burglary/B&E,FARM FACILITY,2024-11-30 03:15:00,76XX FM 2164 DENTON TX  |
| 59437,DENTON PD,Intimidation,SCHOOL-COLLEGE/UNIVERSITY,2024-12-03 17:35:00,44XX LAKEVIEW BLVD DENTON TX  |
+----------------------------------------------------+
```

Query 1:

Crime count (count of different crimes in the city)


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true - Personal -...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

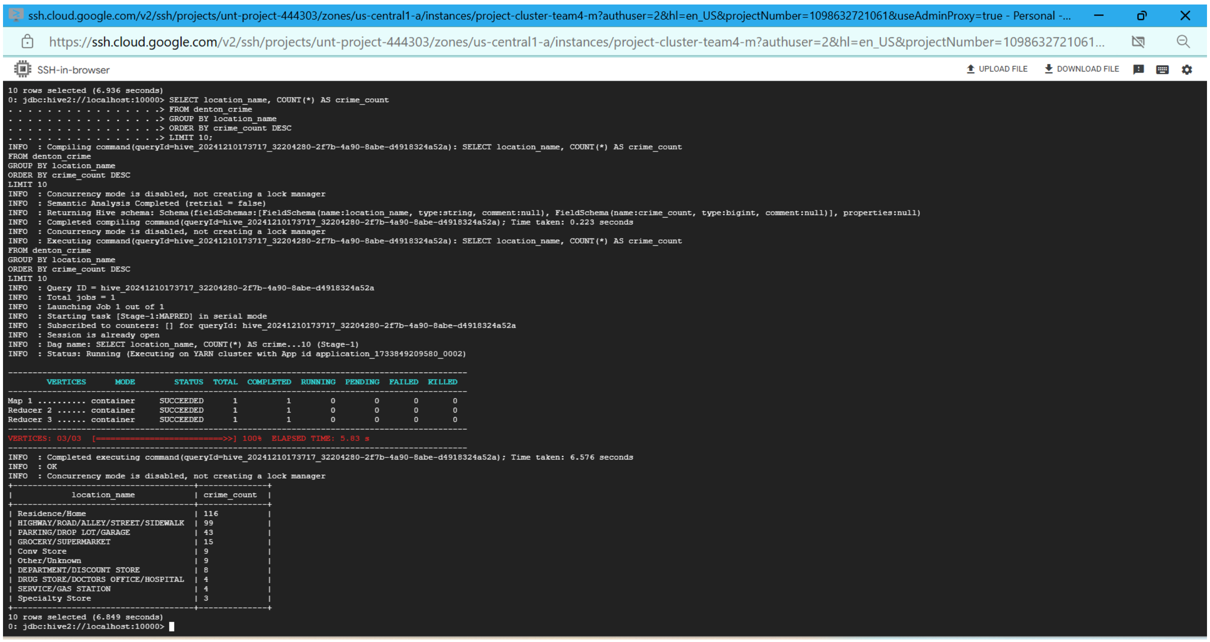
```
5 rows selected (13.619 seconds)
0: jdbc:hive2://localhost:10000> SELECT crime, COUNT(*) AS crime_count
. . . . . . . . . . . . . . . . .> FROM denton_crime
. . . . . . . . . . . . . . . . .> GROUP BY crime
. . . . . . . . . . . . . . . . .> ORDER BY crime_count DESC
. . . . . . . . . . . . . . . . .> LIMIT 10;
INFO  : Compiling command(queryId=hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d): SELECT crime, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY crime
ORDER BY crime_count DESC
LIMIT 10
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:crime, type:string, comment:null), FieldSchema(name:crime_count, type:bigint, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d); Time taken: 0.661 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d): SELECT crime, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY crime
ORDER BY crime_count DESC
LIMIT 10
INFO  : Query ID = hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d
INFO  : Session is already open
INFO  : Dag name: SELECT crime, COUNT(*) AS crime_count
F...10 (Stage-1)
INFO  : Status: Running (Executing on YARN cluster with App id application_1733849209580_0002)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 5.27 s
----------------------------------------------------------------------------------------------
INFO  : Completed executing command(queryId=hive_20241210173544_7798cb70-6507-406e-b964-fb3e76a3261d); Time taken: 6.221 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+----------------------+--------------+
|        crime         | crime_count  |
+----------------------+--------------+
| All Other Offenses   | 75           |
| Simple Assault       | 51           |
| Other Larceny        | 46           |
| Drive Under Infl     | 22           |
| Theft From Veh       | 22           |
| Drug/Narc Violation  | 17           |
| Trespassing          | 15           |
| Burglary/B&E         | 13           |
| Intimidation         | 12           |
| Vandalism            | 12           |
+----------------------+--------------+
10 rows selected (6.936 seconds)
```

These are the top 10 crime incidents, in the Denton city: simple assaults is the highest with 51 total cases, Driver under infl cases are about 46.

Query 2:

Crime count by location name

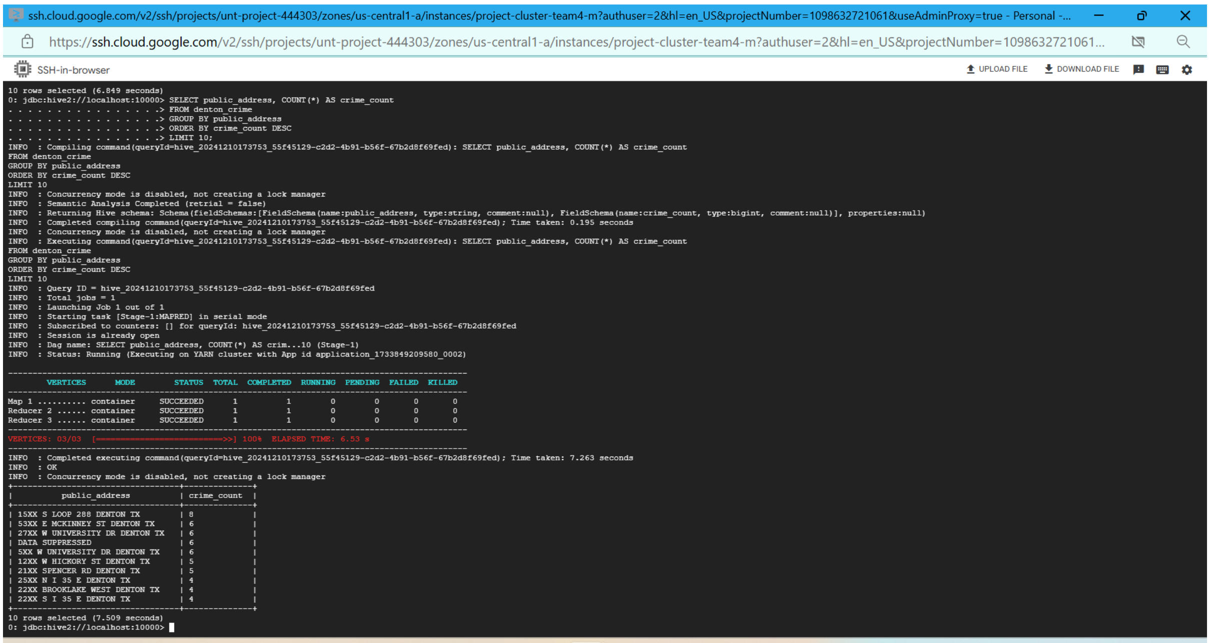
There are a total of 116 crimes that took place at home/residence, followed by the incidents that took place at highway/road/alley/street/sidewalk : 99 total crimes, 43 crimes at parking spots.


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true - Personal -...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
10 rows selected (6.936 seconds)
0: jdbc:hive2://localhost:10000> SELECT location_name, COUNT(*) AS crime_count
. . . . . . . . . . . . . . . . .> FROM denton_crime
. . . . . . . . . . . . . . . . .> GROUP BY location_name
. . . . . . . . . . . . . . . . .> ORDER BY crime_count DESC
. . . . . . . . . . . . . . . . .> LIMIT 10;
INFO  : Compiling command(queryId=hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a): SELECT location_name, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY location_name
ORDER BY crime_count DESC
LIMIT 10
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:location_name, type:string, comment:null), FieldSchema(name:crime_count, type:bigint, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a); Time taken: 0.223 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a): SELECT location_name, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY location_name
ORDER BY crime_count DESC
LIMIT 10
INFO  : Query ID = hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a
INFO  : Session is already open
INFO  : Dag name: SELECT location_name, COUNT(*) AS crime...10 (Stage-1)
INFO  : Status: Running (Executing on YARN cluster with App id application_1733849209580_0002)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 5.83 s
----------------------------------------------------------------------------------------------
INFO  : Completed executing command(queryId=hive_20241210173717_32204280-2f7b-4a90-8abe-d4918324a52a); Time taken: 6.576 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+-------------------------------------+--------------+
|            location_name            | crime_count  |
+-------------------------------------+--------------+
| Residence/Home                      | 116          |
| HIGHWAY/ROAD/ALLEY/STREET/SIDEWALK  | 99           |
| PARKING/DROP LOT/GARAGE             | 43           |
| GROCERY/SUPERMARKET                 | 15           |
| Conv Store                          | 9            |
| Other/Unknown                       | 9            |
| DEPARTMENT/DISCOUNT STORE           | 8            |
| DRUG STORE/DOCTORS OFFICE/HOSPITAL  | 4            |
| SERVICE/GAS STATION                 | 4            |
| Specialty Store                     | 3            |
+-------------------------------------+--------------+
10 rows selected (6.849 seconds)
0: jdbc:hive2://localhost:10000>
```

Query 3:

At which address most of the crimes took place


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true - Personal -...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

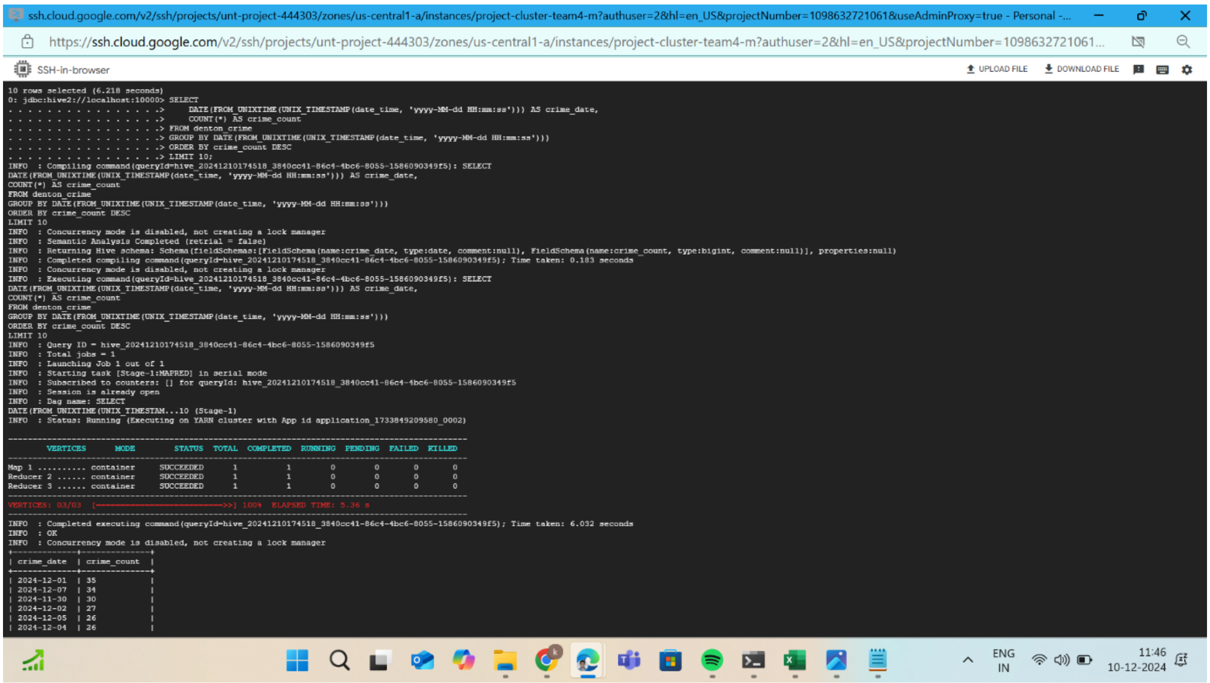
```
10 rows selected (6.849 seconds)
0: jdbc:hive2://localhost:10000> SELECT public_address, COUNT(*) AS crime_count
. . . . . . . . . . . . . . . . .> FROM denton_crime
. . . . . . . . . . . . . . . . .> GROUP BY public_address
. . . . . . . . . . . . . . . . .> ORDER BY crime_count DESC
. . . . . . . . . . . . . . . . .> LIMIT 10;
INFO  : Compiling command(queryId=hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed): SELECT public_address, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY public_address
ORDER BY crime_count DESC
LIMIT 10
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:public_address, type:string, comment:null), FieldSchema(name:crime_count, type:bigint, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed); Time taken: 0.195 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed): SELECT public_address, COUNT(*) AS crime_count
FROM denton_crime
GROUP BY public_address
ORDER BY crime_count DESC
LIMIT 10
INFO  : Query ID = hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed
INFO  : Session is already open
INFO  : Dag name: SELECT public_address, COUNT(*) AS crim...10 (Stage-1)
INFO  : Status: Running (Executing on YARN cluster with App id application_1733849209580_0002)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 6.53 s
----------------------------------------------------------------------------------------------
INFO  : Completed executing command(queryId=hive_20241210173753_55f45129-c2d2-4b91-b56f-67b2d8f69fed); Time taken: 7.263 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+----------------------------------+--------------+
|          public_address          | crime_count  |
+----------------------------------+--------------+
| 15XX S LOOP 288 DENTON TX        | 8            |
| 53XX E MCKINNEY ST DENTON TX     | 6            |
| 27XX W UNIVERSITY DR DENTON TX   | 6            |
| DATA SUPPRESSED                  | 6            |
| 5XX W UNIVERSITY DR DENTON TX    | 6            |
| 12XX W HICKORY ST DENTON TX      | 5            |
| 21XX SPENCER RD DENTON TX        | 5            |
| 25XX N I 35 E DENTON TX          | 4            |
| 22XX BROOKLAKE WEST DENTON TX    | 4            |
| 22XX S I 35 E DENTON TX          | 4            |
+----------------------------------+--------------+
10 rows selected (7.509 seconds)
0: jdbc:hive2://localhost:10000>
```

Most of the cases took place at the south loop, followed by east mckinnet streer and west university drive.

West university drive has also have significant number of crime cases in the span of 12 days.

Query 4:

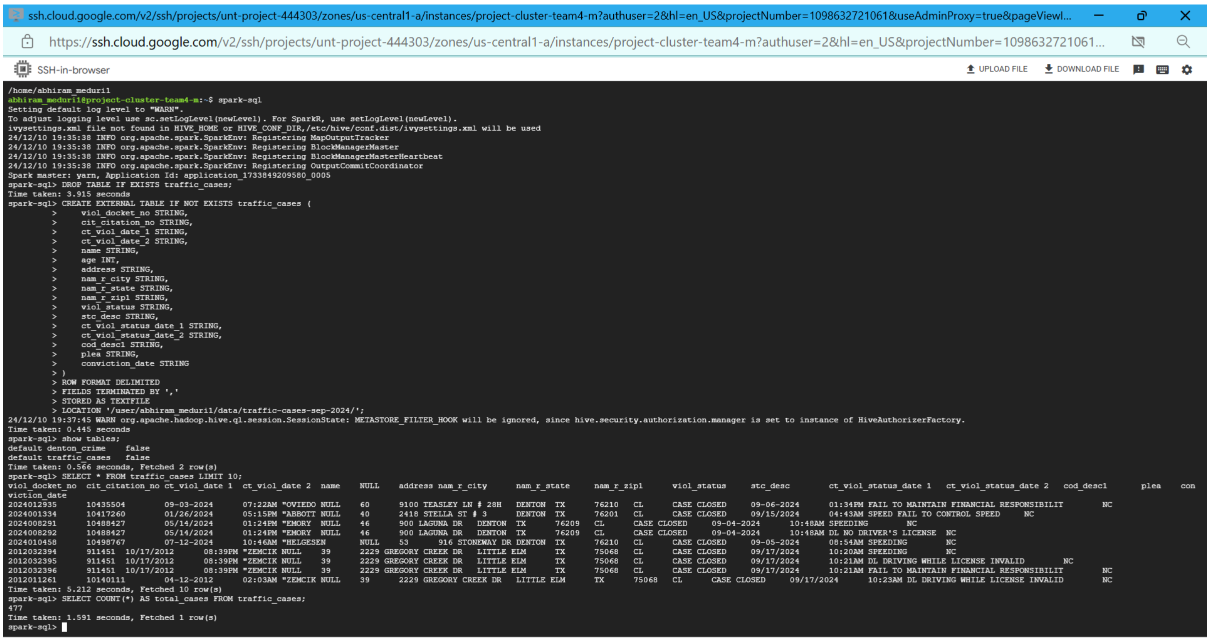
Crime count by date


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true - Personal -...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE
ENG
IN
11:46
10-12-2024

```
10 rows selected (6.218 seconds)
0: jdbc:hive2://localhost:10000> SELECT
. . . . . . . . . . . . . . . . .>     DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss'))) AS crime_date,
. . . . . . . . . . . . . . . . .>     COUNT(*) AS crime_count
. . . . . . . . . . . . . . . . .> FROM denton_crime
. . . . . . . . . . . . . . . . .> GROUP BY DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss')))
. . . . . . . . . . . . . . . . .> ORDER BY crime_count DESC
. . . . . . . . . . . . . . . . .> LIMIT 10;
INFO  : Compiling command(queryId=hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5): SELECT
DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss'))) AS crime_date,
COUNT(*) AS crime_count
FROM denton_crime
GROUP BY DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss')))
ORDER BY crime_count DESC
LIMIT 10
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:crime_date, type:date, comment:null), FieldSchema(name:crime_count, type:bigint, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5); Time taken: 0.183 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5): SELECT
DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss'))) AS crime_date,
COUNT(*) AS crime_count
FROM denton_crime
GROUP BY DATE(FROM_UNIXTIME(UNIX_TIMESTAMP(date_time, 'yyyy-MM-dd HH:mm:ss')))
ORDER BY crime_count DESC
LIMIT 10
INFO  : Query ID = hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5
INFO  : Session is already open
INFO  : Dag name: SELECT
DATE(FROM_UNIXTIME(UNIX_TIMESTAM...10 (Stage-1)
INFO  : Status: Running (Executing on YARN cluster with App id application_1733849209580_0002)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 5.36 s
----------------------------------------------------------------------------------------------
INFO  : Completed executing command(queryId=hive_20241210174518_3840cc41-86c4-4bc6-8055-1586090349f5); Time taken: 6.032 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+-------------+--------------+
| crime_date  | crime_count  |
+-------------+--------------+
| 2024-12-01  | 35           |
| 2024-12-07  | 34           |
| 2024-11-30  | 30           |
| 2024-12-02  | 27           |
| 2024-12-05  | 26           |
| 2024-12-04  | 26           |
```

The data is from 25 November 2024 when we look at this table, we can say that most of the cases took place on December 1st with a count of 35, then on December 7 there were 34 total cases.

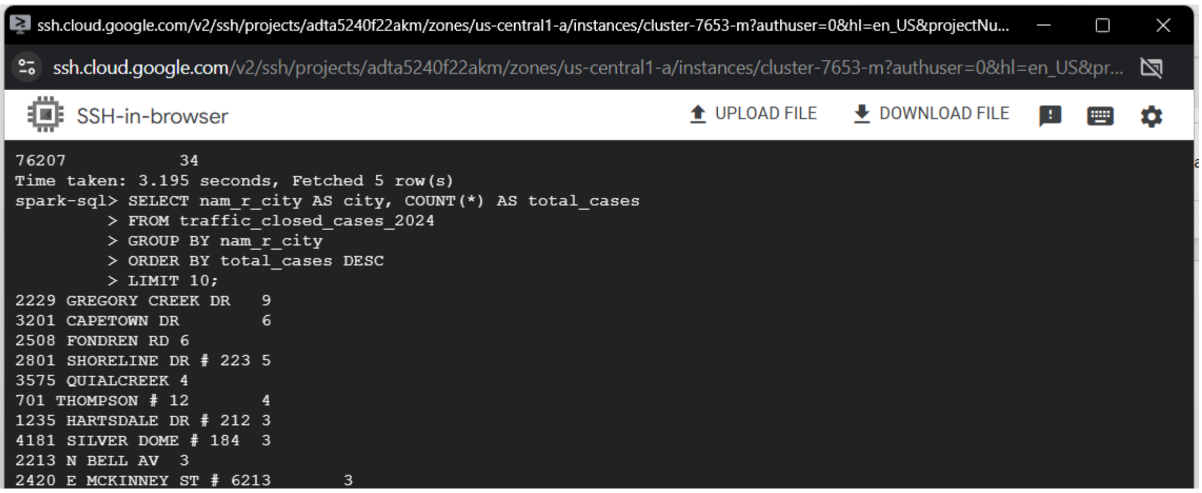
Queries in Spark (Static dataset):


ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061&useAdminProxy=true&pageViewl...
https://ssh.cloud.google.com/v2/ssh/projects/unt-project-444303/zones/us-central1-a/instances/project-cluster-team4-m?authuser=2&hl=en_US&projectNumber=1098632721061...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
/home/abhiram_meduri1
abhiram_meduri1@project-cluster-team4-m:~$ spark-sql
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
ivysettings.xml file not found in HIVE_HOME or HIVE_CONF_DIR,/etc/hive/conf.dist/ivysettings.xml will be used
24/12/10 19:35:38 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
24/12/10 19:35:38 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
24/12/10 19:35:38 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartbeat
24/12/10 19:35:38 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
Spark master: yarn, Application Id: application_1733849209580_0005
spark-sql> DROP TABLE IF EXISTS traffic_cases;
Time taken: 3.915 seconds
spark-sql> CREATE EXTERNAL TABLE IF NOT EXISTS traffic_cases (
         >     viol_docket_no STRING,
         >     cit_citation_no STRING,
         >     ct_viol_date_1 STRING,
         >     ct_viol_date_2 STRING,
         >     name STRING,
         >     age INT,
         >     address STRING,
         >     nam_r_city STRING,
         >     nam_r_state STRING,
         >     nam_r_zip1 STRING,
         >     viol_status STRING,
         >     stc_desc STRING,
         >     ct_viol_status_date_1 STRING,
         >     ct_viol_status_date_2 STRING,
         >     cod_desc1 STRING,
         >     plea STRING,
         >     conviction_date STRING
         > )
         > ROW FORMAT DELIMITED
         > FIELDS TERMINATED BY ','
         > STORED AS TEXTFILE
         > LOCATION '/user/abhiram_meduri1/data/traffic-cases-sep-2024/';
24/12/10 19:37:45 WARN org.apache.hadoop.hive.ql.session.SessionState: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.
Time taken: 0.445 seconds
spark-sql> show tables;
default denton_crime    false
default traffic_cases   false
Time taken: 0.566 seconds, Fetched 2 row(s)
spark-sql> SELECT * FROM traffic_cases LIMIT 10;
viol_docket_no  cit_citation_no ct_viol_date 1  ct_viol_date 2  name    NULL    address nam_r_city      nam_r_state     nam_r_zip1      viol_status     stc_desc        ct_viol_status_date 1   ct_viol_status_date 2   cod_desc1       plea    con
viction_date
2024012935      10435504        09-03-2024      07:22AM "OVIEDO NULL    60      9100 TEASLEY LN # 28H   DENTON  TX      76210   CL      CASE CLOSED     09-06-2024      01:34PM FAIL TO MAINTAIN FINANCIAL RESPONSIBILIT        NC
2024001334      10417260        01/26/2024      05:15PM "ABBOTT NULL    40      2418 STELLA ST # 3      DENTON  TX      76201   CL      CASE CLOSED     09/15/2024      04:43AM SPEED FAIL TO CONTROL SPEED     NC
2024008291      10488427        05/14/2024      01:24PM "EMORY  NULL    46      900 LAGUNA DR   DENTON  TX      76209   CL      CASE CLOSED     09-04-2024      10:48AM SPEEDING        NC
2024008292      10488427        05/14/2024      01:24PM "EMORY  NULL    46      900 LAGUNA DR   DENTON  TX      76209   CL      CASE CLOSED     09-04-2024      10:48AM DL NO DRIVER'S LICENSE  NC
2024010458      10498767        07-12-2024      10:46AM "HELGESEN       NULL    53      916 STONEWAY DR DENTON  TX      76210   CL      CASE CLOSED     09-05-2024      08:54AM SPEEDING        NC
2012032394      911451  10/17/2012      08:39PM "ZEMCIK NULL    39      2229 GREGORY CREEK DR   LITTLE ELM      TX      75068   CL      CASE CLOSED     09/17/2024      10:20AM SPEEDING        NC
2012032395      911451  10/17/2012      08:39PM "ZEMCIK NULL    39      2229 GREGORY CREEK DR   LITTLE ELM      TX      75068   CL      CASE CLOSED     09/17/2024      10:21AM DL DRIVING WHILE LICENSE INVALID        NC
2012032396      911451  10/17/2012      08:39PM "ZEMCIK NULL    39      2229 GREGORY CREEK DR   LITTLE ELM      TX      75068   CL      CASE CLOSED     09/17/2024      10:21AM FAIL TO MAINTAIN FINANCIAL RESPONSIBILIT        NC
2012011261      10140111        04-12-2012      02:03AM "ZEMCIK NULL    39      2229 GREGORY CREEK DR   LITTLE ELM      TX      75068   CL      CASE CLOSED     09/17/2024      10:23AM DL DRIVING WHILE LICENSE INVALID        NC
Time taken: 5.212 seconds, Fetched 10 row(s)
spark-sql> SELECT COUNT(*) AS total_cases FROM traffic_cases;
477
Time taken: 1.591 seconds, Fetched 1 row(s)
spark-sql>
```

Query 1:

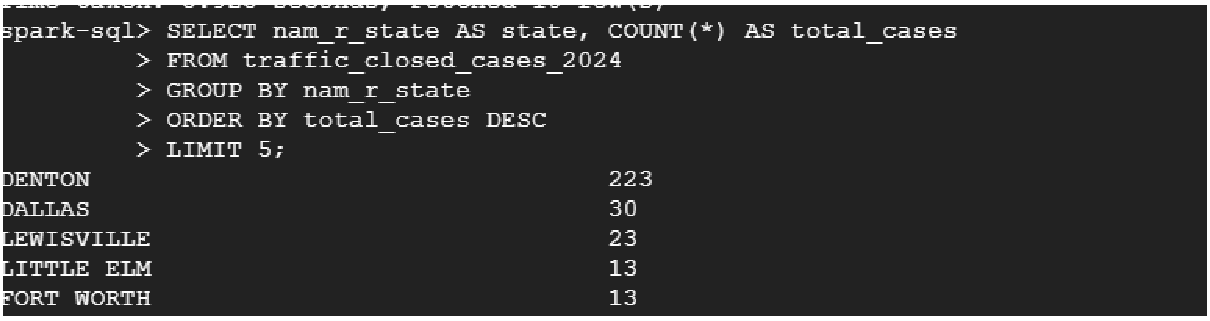
Address where highest cases were recorded:


ssh.cloud.google.com/v2/ssh/projects/adta5240f22akm/zones/us-central1-a/instances/cluster-7653-m?authuser=0&hl=en_US&projectNu...
ssh.cloud.google.com/v2/ssh/projects/adta5240f22akm/zones/us-central1-a/instances/cluster-7653-m?authuser=0&hl=en_US&pr...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
76207           34
Time taken: 3.195 seconds, Fetched 5 row(s)
spark-sql> SELECT nam_r_city AS city, COUNT(*) AS total_cases
         > FROM traffic_closed_cases_2024
         > GROUP BY nam_r_city
         > ORDER BY total_cases DESC
         > LIMIT 10;
2229 GREGORY CREEK DR   9
3201 CAPETOWN DR        6
2508 FONDREN RD 6
2801 SHORELINE DR # 223 5
3575 QUIALCREEK 4
701 THOMPSON # 12       4
1235 HARTSDALE DR # 212 3
4181 SILVER DOME # 184  3
2213 N BELL AV  3
2420 E MCKINNEY ST # 6213       3
```

Query 2:

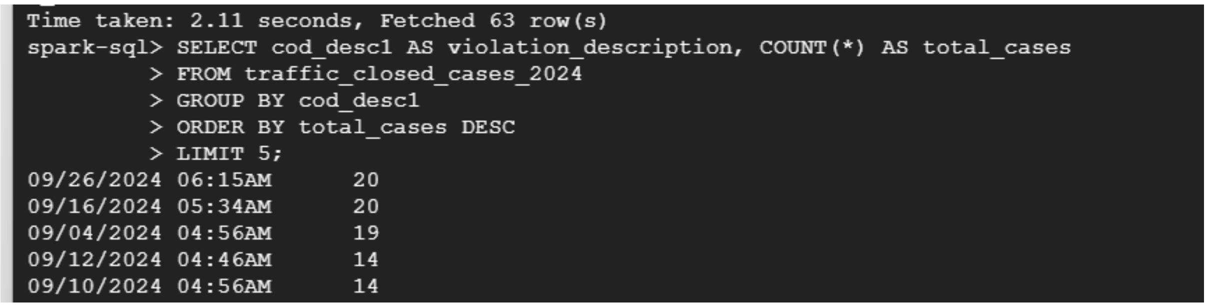
City where highest cases were recorded:



```
spark-sql> SELECT nam_r_state AS state, COUNT(*) AS total_cases
         > FROM traffic_closed_cases_2024
         > GROUP BY nam_r_state
         > ORDER BY total_cases DESC
         > LIMIT 5;
DENTON                              223
DALLAS                              30
LEWISVILLE                          23
LITTLE ELM                          13
FORT WORTH                          13
```

Query 3:

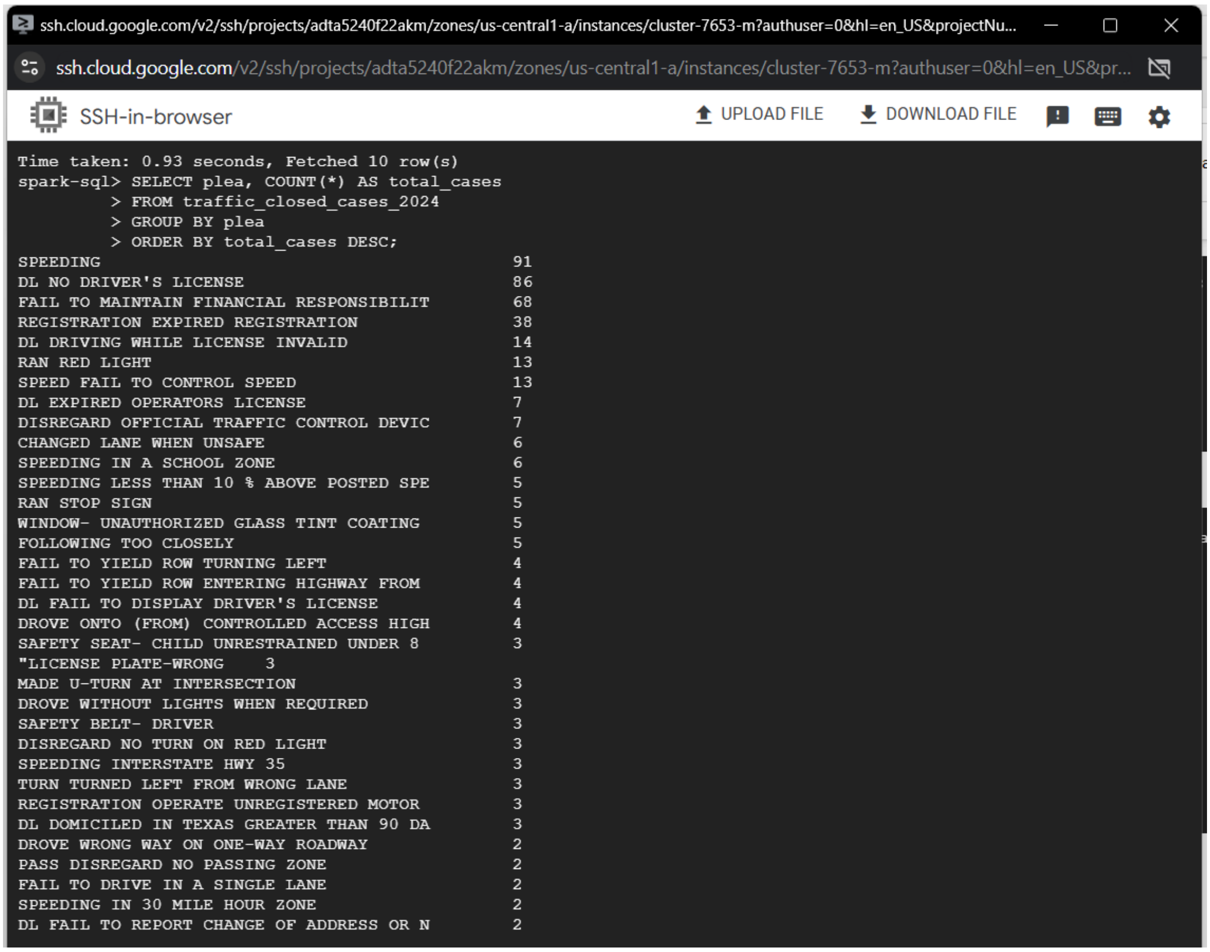
Date when highest cases were recorded:



```
Time taken: 2.11 seconds, Fetched 63 row(s)
spark-sql> SELECT cod_desc1 AS violation_description, COUNT(*) AS total_cases
         > FROM traffic_closed_cases_2024
         > GROUP BY cod_desc1
         > ORDER BY total_cases DESC
         > LIMIT 5;
09/26/2024 06:15AM      20
09/16/2024 05:34AM      20
09/04/2024 04:56AM      19
09/12/2024 04:46AM      14
09/10/2024 04:56AM      14
```

Query 4:

Reason for violation case



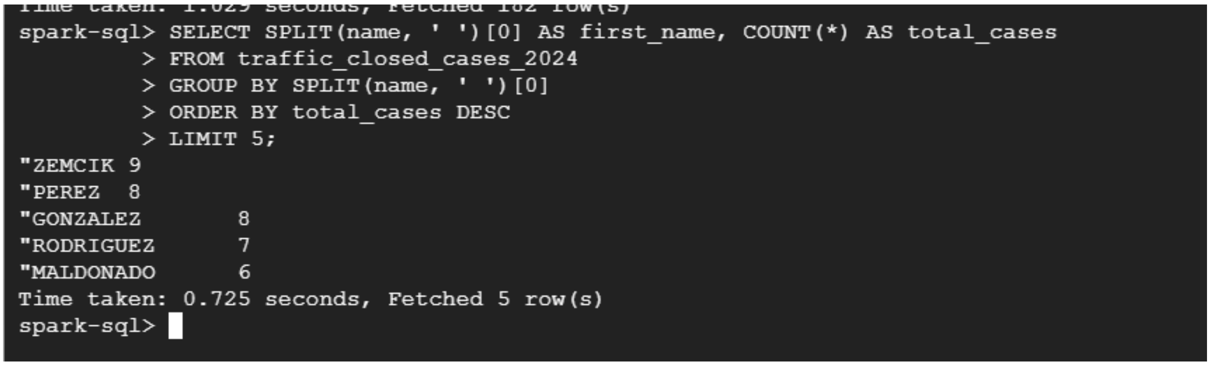
ssh.cloud.google.com/v2/ssh/projects/adta5240f22akm/zones/us-central1-a/instances/cluster-7653-m?authuser=0&hl=en_US&projectNu...
ssh.cloud.google.com/v2/ssh/projects/adta5240f22akm/zones/us-central1-a/instances/cluster-7653-m?authuser=0&hl=en_US&pr...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
Time taken: 0.93 seconds, Fetched 10 row(s)
spark-sql> SELECT plea, COUNT(*) AS total_cases
         > FROM traffic_closed_cases_2024
         > GROUP BY plea
         > ORDER BY total_cases DESC;
SPEEDING                                         91
DL NO DRIVER'S LICENSE                           86
FAIL TO MAINTAIN FINANCIAL RESPONSIBILIT         68
REGISTRATION EXPIRED REGISTRATION                38
DL DRIVING WHILE LICENSE INVALID                 14
RAN RED LIGHT                                    13
SPEED FAIL TO CONTROL SPEED                      13
DL EXPIRED OPERATORS LICENSE                     7
DISREGARD OFFICIAL TRAFFIC CONTROL DEVIC         7
CHANGED LANE WHEN UNSAFE                         6
SPEEDING IN A SCHOOL ZONE                        6
SPEEDING LESS THAN 10 % ABOVE POSTED SPE         5
RAN STOP SIGN                                    5
WINDOW- UNAUTHORIZED GLASS TINT COATING          5
FOLLOWING TOO CLOSELY                            5
FAIL TO YIELD ROW TURNING LEFT                   4
FAIL TO YIELD ROW ENTERING HIGHWAY FROM          4
DL FAIL TO DISPLAY DRIVER'S LICENSE              4
DROVE ONTO (FROM) CONTROLLED ACCESS HIGH         4
SAFETY SEAT- CHILD UNRESTRAINED UNDER 8          3
"LICENSE PLATE-WRONG    3
MADE U-TURN AT INTERSECTION                      3
DROVE WITHOUT LIGHTS WHEN REQUIRED               3
SAFETY BELT- DRIVER                              3
DISREGARD NO TURN ON RED LIGHT                   3
SPEEDING INTERSTATE HWY 35                       3
TURN TURNED LEFT FROM WRONG LANE                 3
REGISTRATION OPERATE UNREGISTERED MOTOR          3
DL DOMICILED IN TEXAS GREATER THAN 90 DA         3
DROVE WRONG WAY ON ONE-WAY ROADWAY               2
PASS DISREGARD NO PASSING ZONE                   2
FAIL TO DRIVE IN A SINGLE LANE                   2
SPEEDING IN 30 MILE HOUR ZONE                    2
DL FAIL TO REPORT CHANGE OF ADDRESS OR N         2
```

Speeding violation cases top the list with 91, followed by no drivers license.

Query 5:

Top people with most violations

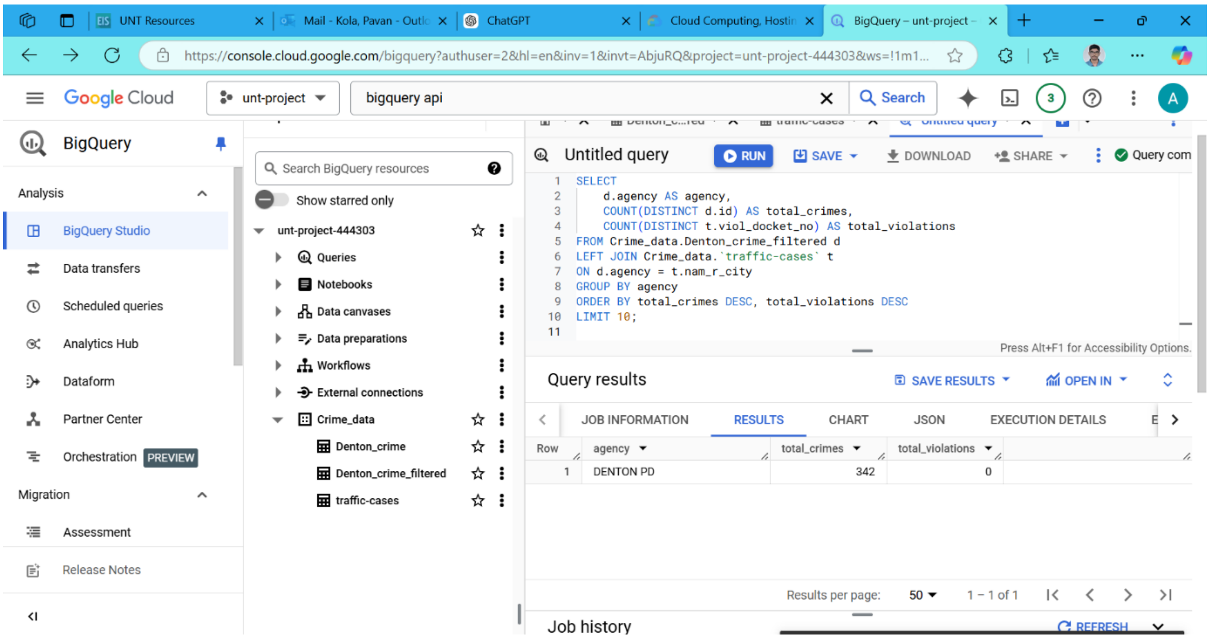


```
Time taken: 1.029 seconds, Fetched 182 row(s)
spark-sql> SELECT SPLIT(name, ' ')[0] AS first_name, COUNT(*) AS total_cases
         > FROM traffic_closed_cases_2024
         > GROUP BY SPLIT(name, ' ')[0]
         > ORDER BY total_cases DESC
         > LIMIT 5;
"ZEMCIK 9
"PEREZ  8
"GONZALEZ       8
"RODRIGUEZ      7
"MALDONADO      6
Time taken: 0.725 seconds, Fetched 5 row(s)
spark-sql> 
```

Zemick toped with 9, followed by Perez and Gonzalez with 8 each.

Queries in BigQuery (combined Queries of both datasets)

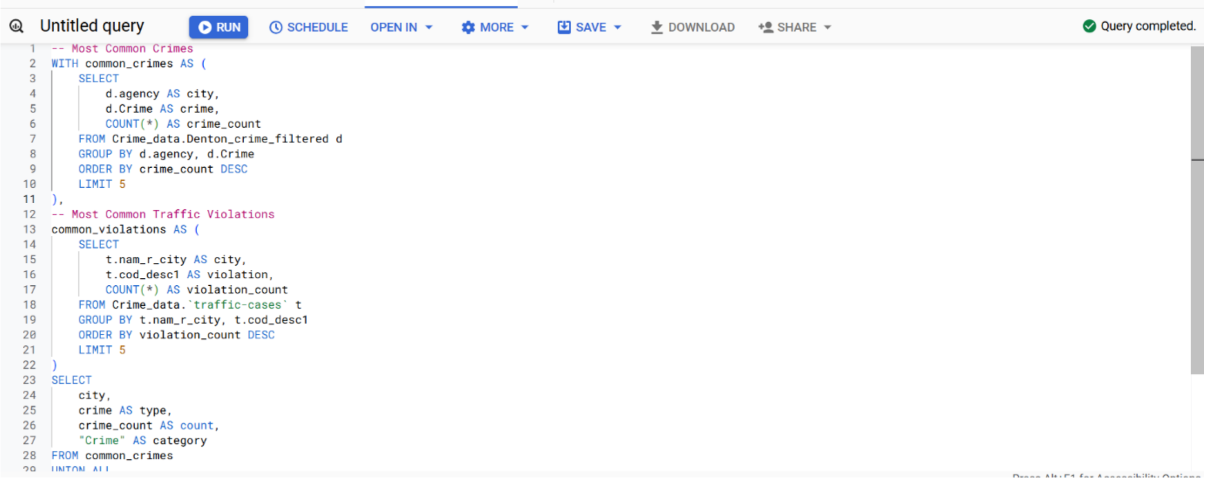
City wise crime and total number of crime type count



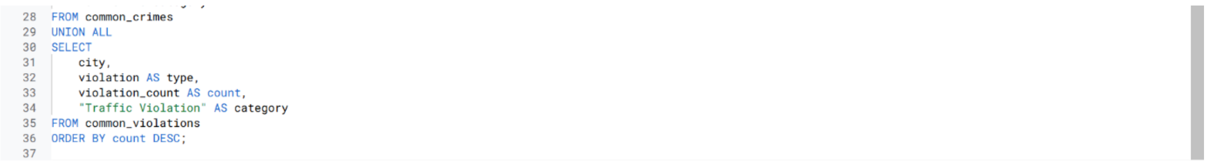
```sql
SELECT
    d.agency AS agency,
    COUNT(DISTINCT d.id) AS total_crimes,
    COUNT(DISTINCT t.viol_docket_no) AS total_violations
FROM Crime_data.Denton_crime_filtered d
LEFT JOIN Crime_data.`traffic-cases` t
ON d.agency = t.nam_r_city
GROUP BY agency
ORDER BY total_crimes DESC, total_violations DESC
LIMIT 10;
```

Row	agency	total_crimes	total_violations
1	DENTON PD	342	0

Most common crimes and their violations

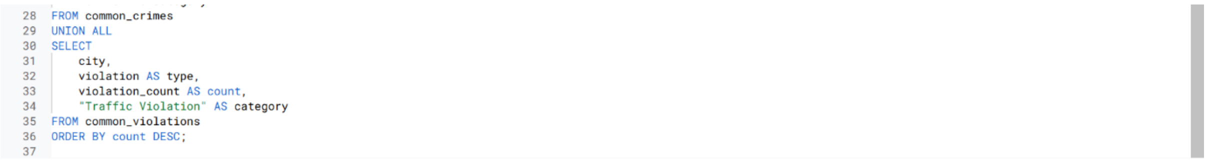


```sql
-- Most Common Crimes
WITH common_crimes AS (
    SELECT
        d.agency AS city,
        d.Crime AS crime,
        COUNT(*) AS crime_count
    FROM Crime_data.Denton_crime_filtered d
    GROUP BY d.agency, d.Crime
    ORDER BY crime_count DESC
    LIMIT 5
),
-- Most Common Traffic Violations
common_violations AS (
    SELECT
        t.nam_r_city AS city,
        t.cod_desc1 AS violation,
        COUNT(*) AS violation_count
    FROM Crime_data.`traffic-cases` t
    GROUP BY t.nam_r_city, t.cod_desc1
    ORDER BY violation_count DESC
    LIMIT 5
)
SELECT
    city,
    crime AS type,
    crime_count AS count,
    "Crime" AS category
FROM common_crimes
```

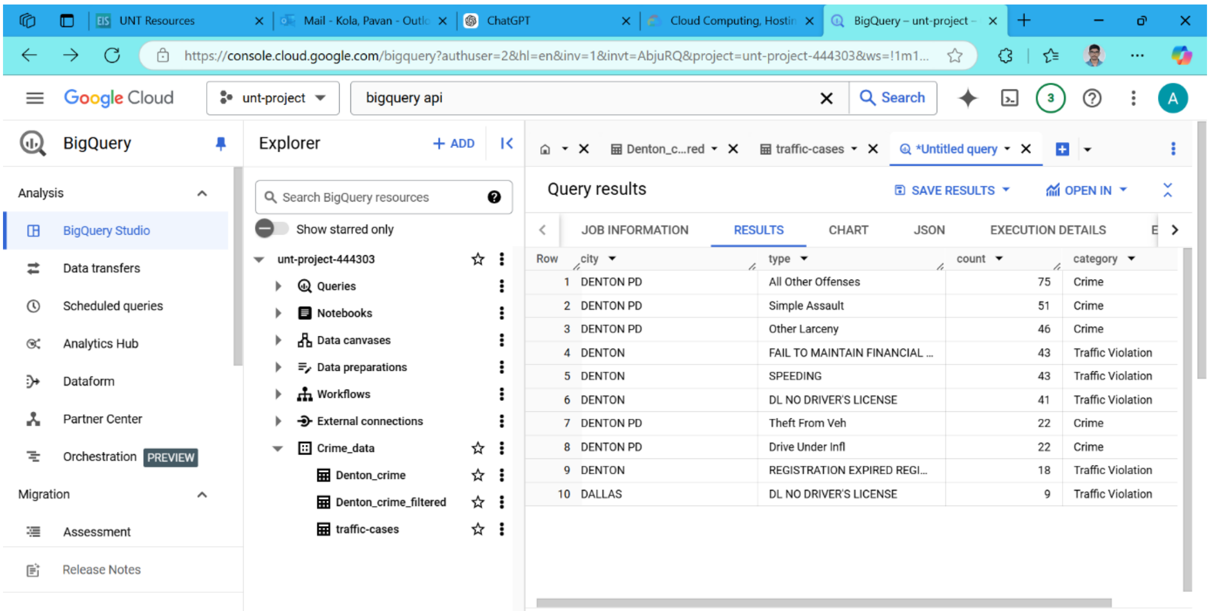


FROM common_crimes

```sql
UNION ALL
SELECT
    city,
    violation AS type,
    violation_count AS count,
    "Traffic Violation" AS category
FROM common_violations
ORDER BY count DESC;
```

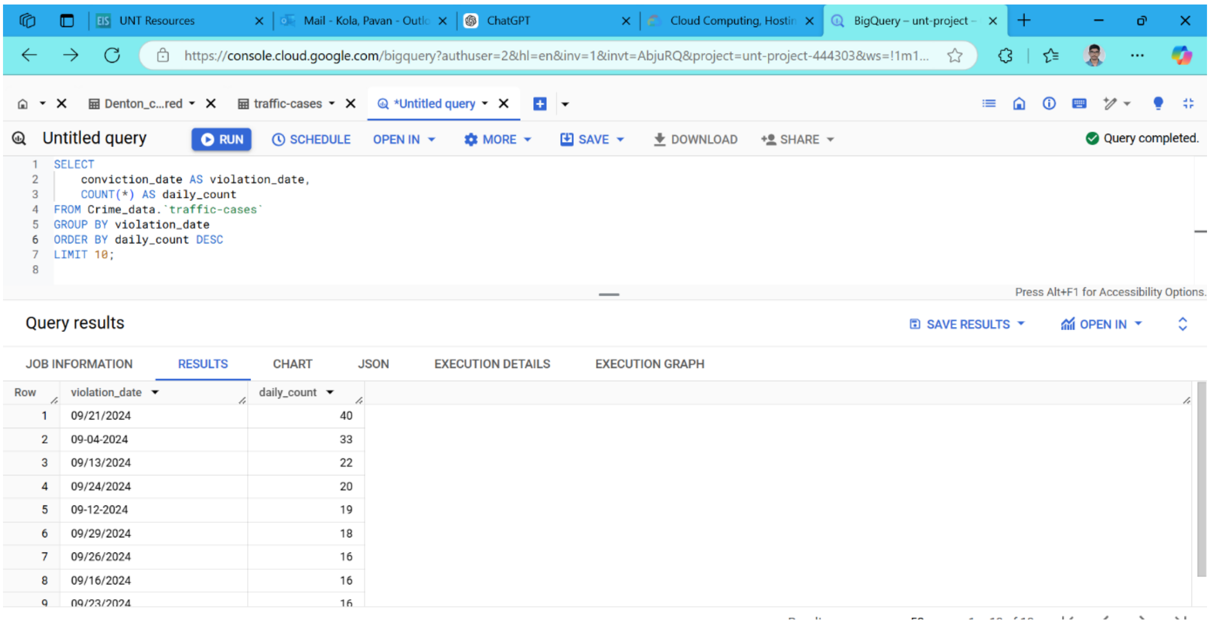


```sql
FROM common_crimes
UNION ALL
SELECT
    city,
    violation AS type,
    violation_count AS count,
    "Traffic Violation" AS category
FROM common_violations
ORDER BY count DESC;
```



Row	city	type	count	category
1	DENTON PD	All Other Offenses	75	Crime
2	DENTON PD	Simple Assault	51	Crime
3	DENTON PD	Other Larceny	46	Crime
4	DENTON	FAIL TO MAINTAIN FINANCIAL ...	43	Traffic Violation
5	DENTON	SPEEDING	43	Traffic Violation
6	DENTON	DL NO DRIVER'S LICENSE	41	Traffic Violation
7	DENTON PD	Theft From Veh	22	Crime
8	DENTON PD	Drive Under Infl	22	Crime
9	DENTON	REGISTRATION EXPIRED REGI...	18	Traffic Violation
10	DALLAS	DL NO DRIVER'S LICENSE	9	Traffic Violation

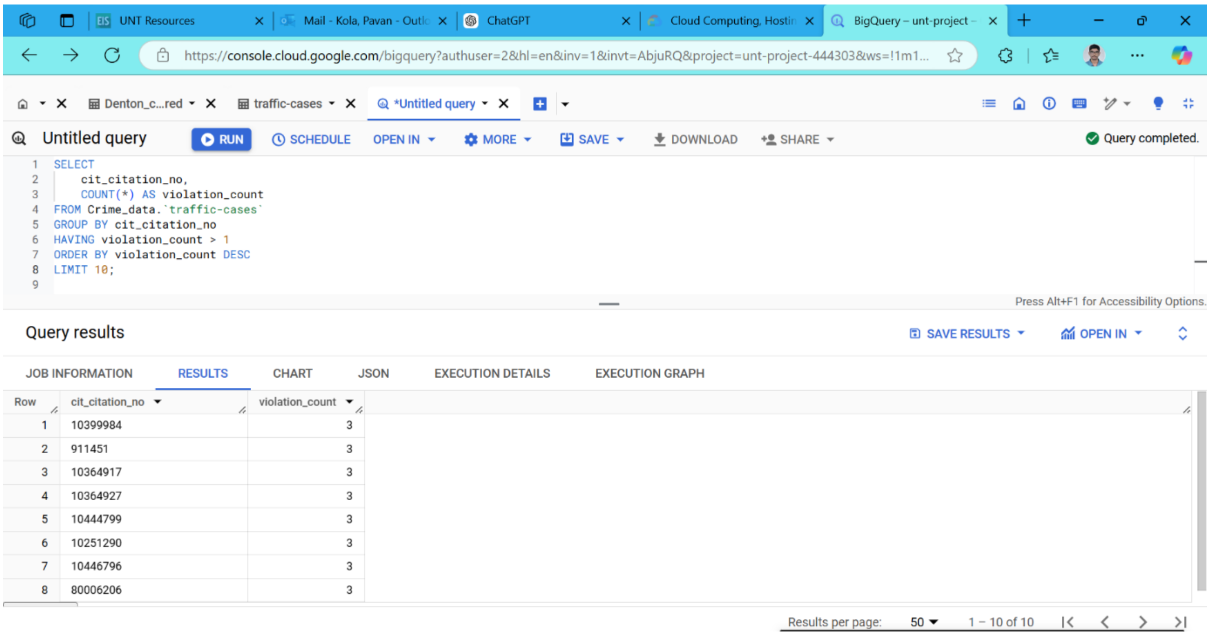
Daily trends in Traffic cases:



```sql
SELECT
    conviction_date AS violation_date,
    COUNT(*) AS daily_count
FROM Crime_data.`traffic-cases`
GROUP BY violation_date
ORDER BY daily_count DESC
LIMIT 10;
```

Row	violation_date	daily_count
1	09/21/2024	40
2	09-04-2024	33
3	09/13/2024	22
4	09/24/2024	20
5	09-12-2024	19
6	09/29/2024	18
7	09/26/2024	16
8	09/16/2024	16
9	09/23/2024	16

Highest number of violations for a citation number:



```sql
SELECT
    cit_citation_no,
    COUNT(*) AS violation_count
FROM Crime_data.`traffic-cases`
GROUP BY cit_citation_no
HAVING violation_count > 1
ORDER BY violation_count DESC
LIMIT 10;
```

Row	cit_citation_no	violation_count
1	10399984	3
2	911451	3
3	10364917	3
4	10364927	3
5	10444799	3
6	10251290	3
7	10446796	3
8	80006206	3