Crime Data Analyst Project by Wynter Gray

Portfolio Report — SQL analysis, Tableau visualization, and insights from public crime data.

Project Overview

This project analyzes public crime data from the City of Aurora using SQL for analytical querying and Tableau for visualization. The analysis combines three datasets — incidents, calls for service, and locations — to explore geographic trends in reported incidents.

Using SQL, each query explores a key analytical question: identifying cleared cases, counting incidents by date, finding missing values, and more. The cleaned outputs were visualized through Tableau, producing an interactive dashboard summarizing key trends and spatial patterns.

Tableau Dashboard

The published Tableau dashboard visualizes crime patterns using five sheets:
• **Total Incidents by Status** — shows the distribution of Open, Cleared, and Unfounded cases.
• **Top ZIP Codes by Incident Volume** — highlights ZIP codes with the highest counts, grouped by case status.
• **Top Offense Codes** — lists the 10 most frequent offense categories.
• **Map of Incidents** — plots all cases geographically across Aurora and nearby areas, colored by case status.
• **Incidents in the Last 12 Months** — a trend line visualizing monthly incident counts.

Together, these visualizations provide a dynamic overview of public safety activity, supporting data-driven insights and communication.

SQL Queries and Results

1) 1) View all columns from the incidents table.

- SQL Query:

```
SELECT *
FROM incidents$
```

- Result (preview):

incident_id	incident_number	offense_code	occurred_dt	cleared_dt	status	beat	census_tract	location_id
1	AUR-251103-8069	190.0	2024-11-03 22:22:16		Unfounded	B09	968.2	48

2	AUR-250422-6742	180.0	2024-11-04 17:48:16	2024-11-16 19:19:16	Cleared	B10	873.8	13
3	AUR-250922-5641	140.0	2024-11-05 09:48:16	2024-11-11 04:58:16	Cleared	B10	672.6	53
4	AUR-241105-1116	110.0	2024-11-05 19:16:16		Open	B10	783.9	16
5	AUR-250526-7698	220.0	2024-11-06 14:05:16	2024-11-15 03:03:16	Cleared	B06	835.7	20
6	AUR-250401-6990	190.0	2024-11-06 19:29:16	2024-11-20 03:20:16	Cleared	B10	897.7	88
7	AUR-250301-3928	190.0	2024-11-06 23:13:16	2024-11-11 14:37:16	Cleared	B06	672.6	72
8	AUR-250414-3262	220.0	2024-11-07 05:00:16	2024-11-14 04:38:16	Cleared	B11	835.7	96

2) 2) Show only incident_id, status, and occurred_dt.

- SQL Query:

```
SELECT incident_id, status, occurred_dt
FROM incidents$
```

- Result (preview):

incident_id	status	occurred_dt
1	Unfounded	2024-11-03 22:22:16
2	Cleared	2024-11-04 17:48:16
3	Cleared	2024-11-05 09:48:16
4	Open	2024-11-05 19:16:16
5	Cleared	2024-11-06 14:05:16
6	Cleared	2024-11-06 19:29:16
7	Cleared	2024-11-06 23:13:16
8	Cleared	2024-11-07 05:00:16

3) 3) Find incidents that are Cleared.

- SQL Query:

```
SELECT incident_id, status
FROM incidents$
WHERE status = 'Cleared'
```

- Result (preview):

incident_id	status
2	Cleared
3	Cleared
5	Cleared
6	Cleared
7	Cleared
8	Cleared
10	Cleared
13	Cleared

4) 4) Find incidents that occurred after January 1, 2025.

- SQL Query:

```
SELECT *
FROM incidents$
WHERE occurred_dt >= '2025-01-01'
```

- Result (preview):

incident_id	incident_number	offense_code	occurred_dt	cleared_dt	status	beat	census_tract	location_id
158	AUR-250423-2108	160.0	2025-01-01 15:41:16		Open	B07	672.6	9
159	AUR-250902-5496	130.0	2025-01-02 11:21:16		Open	B06	783.9	53
160	AUR-250115-9666	120.0	2025-01-02 12:00:16	2025-01-07 06:20:16	Cleared	B02	920.8	93
161	AUR-250624-9608	160.0	2025-01-02 18:30:16	2025-01-09 07:30:16	Cleared	B10	727.9	46
162	AUR-250119-9132	200.0	2025-01-04 19:06:16		Unfounded	B09	920.8	64

163	AUR-250727-1025	150.0	2025-01-05 05:15:16	2025-01-19 02:14:16	Cleared	B02	586.2	60
164	AUR-241223-7793	140.0	2025-01-05 05:58:16	2025-01-07 22:32:16	Cleared	B07	566.7	43
165	AUR-250718-1185	210.0	2025-01-05 19:12:16		Open	B11	771.1	89

5) 5) Find incidents with no offense code (missing data).

- SQL Query:

```
SELECT *
FROM incidents$
WHERE offense_code IS NULL
```

- Result (preview):

incident_id	incident_number	offense_code	occurred_dt	cleared_dt	status	beat	census_tract	location_id
24	AUR-250629-5706		2024-11-12 02:34:16		Open	B02	670.4	87
44	AUR-250821-3745		2024-11-20 03:41:16		Open	B11	923.9	54
206	AUR-241202-8155		2025-01-17 14:22:16	2025-01-20 21:10:16	Cleared	B03	744.4	18
345	AUR-251014-2347		2025-03-05 06:05:16		Open	B06	920.8	15
406	AUR-241124-9096		2025-03-26 20:18:16	2025-04-05 18:07:16	Cleared	B02	868.2	92
533	AUR-250727-6483		2025-05-14 07:34:16		Open	B02	873.8	104

580	AUR-250117-1527		2025-06-01 06:15:16		Open	B03	606.7	43
829	AUR-241214-8361		2025-08-31 07:04:16		Unfounded	B11	873.8	13

6) 6) Show the 5 most recent incidents.

- SQL Query:

```
SELECT TOP 5 *
FROM incidents$
ORDER BY occurred_dt DESC
```

- Result (preview):

incident_id	incident_number	offense_code	occurred_dt	cleared_dt	status	beat	census_tract	location_id
1000	AUR-241220-1510	150.0	2025-11-03 17:10:16	2025-11-14 10:23:16	Cleared	B03	575.1	6
999	AUR-241228-6902	160.0	2025-11-03 05:53:16	2025-11-09 16:39:16	Cleared	B06	968.2	115
998	AUR-250521-7084	140.0	2025-11-02 01:04:16		Open	B11	968.2	10
997	AUR-250925-9305	120.0	2025-11-01 18:34:16	2025-11-15 10:43:16	Cleared	B08	835.7	66
996	AUR-250624-5217	220.0	2025-11-01 06:02:16	2025-11-06 10:41:16	Cleared	B01	575.1	70
994	AUR-241203-1812	200.0	2025-10-31 23:38:16		Open	B04	835.7	84
995	AUR-250217-8214	210.0	2025-10-31 23:38:16		Open	B10	923.9	6

993	AUR-250624-9108	150.0	2025-10-31 05:50:16		Open	B03	868.2	81

7) 7) Show all incidents sorted by status then by date.

- SQL Query:

```
SELECT *
FROM incidents$
ORDER BY status, occurred_dt
```

- Result (preview):

incident_id	incident_number	offense_code	occurred_dt	cleared_dt	status	beat	census_tract	location_id
2	AUR-250422-6742	180.0	2024-11-04 17:48:16	2024-11-16 19:19:16	Cleared	B10	873.8	13
3	AUR-250922-5641	140.0	2024-11-05 09:48:16	2024-11-11 04:58:16	Cleared	B10	672.6	53
5	AUR-250526-7698	220.0	2024-11-06 14:05:16	2024-11-15 03:03:16	Cleared	B06	835.7	20
6	AUR-250401-6990	190.0	2024-11-06 19:29:16	2024-11-20 03:20:16	Cleared	B10	897.7	88
7	AUR-250301-3928	190.0	2024-11-06 23:13:16	2024-11-11 14:37:16	Cleared	B06	672.6	72
8	AUR-250414-3262	220.0	2024-11-07 05:00:16	2024-11-14 04:38:16	Cleared	B11	835.7	96
10	AUR-241107-1852	210.0	2024-11-07 07:51:16	2024-11-19 20:42:16	Cleared	B03	670.4	82
13	AUR-250610-1941	190.0	2024-11-08 02:30:16	2024-11-17 05:28:16	Cleared	B03	920.8	57

8) 8) Count total number of incidents.

- SQL Query:

```
SELECT COUNT(*) AS total_incidents
FROM incidents$
```

- Result (preview):

total_incidents
1000

9) 9) Count number of incidents per status.

- SQL Query:

```
SELECT status, COUNT(*) AS incidents_per_status
FROM incidents$
GROUP BY status
```

- Result (preview):

status	incidents_per_status
Cleared	541
Open	370
Unfounded	89

10) 10) Find the top 3 most common offense codes.

- SQL Query:

```
SELECT TOP 3 offense_code, COUNT(*) AS total
FROM incidents$
WHERE offense_code IS NOT NULL
GROUP BY offense_code
ORDER BY total DESC
```

- Result (preview):

offense_code	total
210.0	101.0
220.0	93.0
180.0	87.0
130.0	86.0
120.0	86.0
160.0	83.0
110.0	81.0
150.0	79.0

11) 11) Join incidents with locations to show address.

- SQL Query:

```
SELECT address, incident_id, status
FROM locations$
JOIN incidents$ ON locations$.location_id=incidents$.location_id
```

- Result (preview):

address	incident_id	status
1924 Colfax Ave	97	Cleared
1924 Colfax Ave	443	Cleared
1924 Colfax Ave	519	Open
1924 Colfax Ave	631	Cleared
1924 Colfax Ave	738	Cleared
1924 Colfax Ave	783	Cleared
4606 Alameda Ave	56	Cleared
4606 Alameda Ave	57	Open

12) 12) Show incidents that occurred in ZIP code 80013.

- SQL Query:

```
SELECT incident_id, occurred_dt, zip_code
FROM incidents$
JOIN locations$ ON incidents$.location_id=locations$.location_id
WHERE zip_code = '80013'
```

- Result (preview):

incident_id	occurred_dt	zip_code
49	2024-11-22 13:10:16	80013
155	2024-12-30 21:00:16	80013
325	2025-02-27 20:37:16	80013
706	2025-07-20 09:55:16	80013
897	2025-09-27 20:33:16	80013
919	2025-10-04 04:57:16	80013
976	2025-10-24 13:41:16	80013
174	2025-01-07 20:02:16	80013

13) 13) Replace missing offense_code values with 'Unknown'.

- SQL Query:

```
SELECT ISNULL(offense_code, 'Uknown') AS offense_code,
COUNT(*) as count_row
FROM incidents$
GROUP BY ISNULL(offense_code, 'Uknown')
```

```
Note: Execution failed on sql 'SELECT ISNULL(offense_code, 'Uknown') AS offense_code,
COUNT(*) as count_row
FROM incidents$
GROUP BY ISNULL(offense_code, 'Uknown')': near "ISNULL": syntax error
```

14) 14) Find earliest and latest incident dates.

- SQL Query:

```
SELECT MIN(occurred_dt) AS earliest_incident,
MAX(occurred_dt) AS latest_incident
FROM incidents$
```

- Result (preview):

earliest_incident	latest_incident
2024-11-03 22:22:16	2025-11-03 17:10:16

15) 15) Count how many incidents have missing cleared_dt values.

- SQL Query:

```
SELECT COUNT(*) AS missing_clear_dates
FROM incidents$
WHERE cleared_dt IS NULL
```

- Result (preview):

missing_clear_dates
459

16) 16) Show call_id, final_call_type, and matching incident_id.

- SQL Query:

```
SELECT call_id, final_call_type, incident_id
FROM calls_for_service$
JOIN incidents$ ON calls_for_service$.location_id=incidents$.location_id
```

- Result (preview):

call_id	final_call_type	incident_id
1	Domestic Dispute	43
1	Domestic Dispute	84
1	Domestic Dispute	113
1	Domestic Dispute	200
1	Domestic Dispute	807
1	Domestic Dispute	950
2	Welfare Check	6
2	Welfare Check	187

17) 17) Count how many calls have a matching incident by location.

- SQL Query:

```
SELECT COUNT(DISTINCT call_id) AS calls_with_incidents
FROM calls_for_service$
JOIN incidents$ on calls_for_service$.location_id=incidents$.location_id
```

- Result (preview):

calls_with_incidents
700

18) 18) Find top 5 addresses with the most incidents.

- SQL Query:

```
SELECT TOP 5 address, COUNT(incident_id) AS total_incidents
FROM locations$
JOIN incidents$ ON locations$.location_id=incidents$.location_id
GROUP BY address
ORDER BY total_incidents DESC
```

- Result (preview):

address	total_incidents
6973 Alameda Ave	16
7673 Iliff Ave	14
4606 Alameda Ave	14
4415 6th Ave	14
3708 Colfax Ave	14
1927 Chambers Rd	14
1184 Alameda Ave	14
5674 Chambers Rd	13

19) Incidents by Weekday and Average Hours to Clear

- SQL Query:

```
SELECT weekday_name, COUNT(*) AS Total FROM incidents GROUP BY weekday_name
ORDER BY Total DESC;
SELECT AVG(hours_to_clear) AS avg_hrs_to_clear FROM incidents WHERE status = 'Cleared';
```

- Incidents by Weekday:

weekday_name	Total
Monday	153
Thursday	150
Sunday	147
Saturday	144
Wednesday	144
Tuesday	132
Friday	130

20) 20) Find average time to clear (hours).

- SQL Query:

```
SELECT AVG(DATEDIFF(hour, occurred_dt, cleared_dt)) AS avg_hrs_to_clear
FROM incidents$
WHERE cleared_dt IS NOT NULL
```

Average Hours to Clear: 167

Average Hours to Clear	167