SQL Portfolio Project – Patient Scheduling & Billing Analysis

This document presents a focused SQL portfolio project built around a simulated healthcare patient scheduling and billing dataset. The work emphasizes analytical reasoning, relational integrity, and edge-case handling rather than application development.

Project Objectives

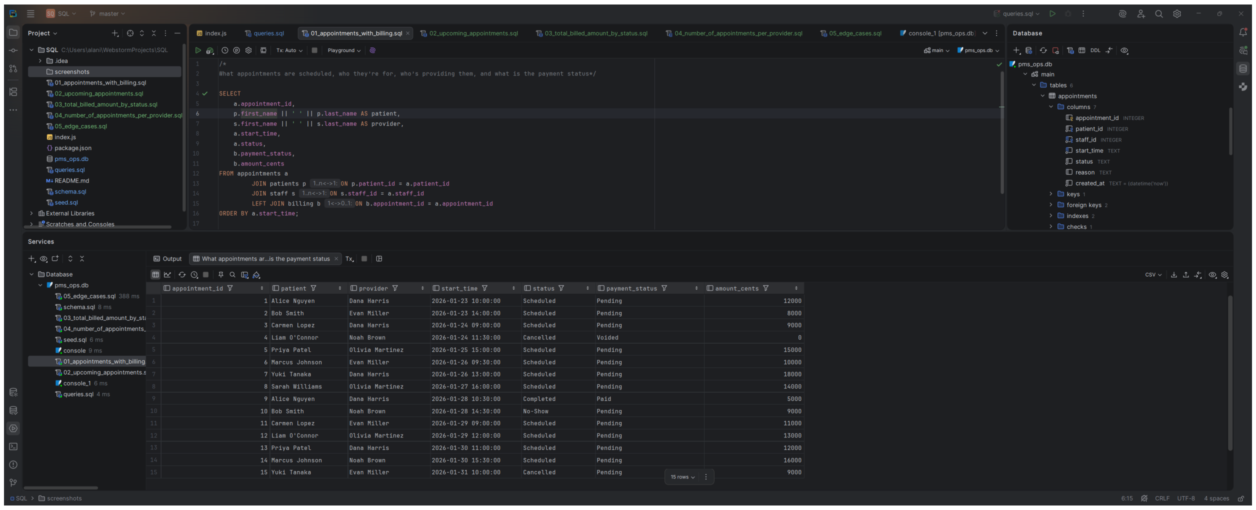
The objective of this project is to demonstrate how structured SQL analysis can be used to answer real operational questions. The dataset supports analysis of appointment outcomes, billing status consistency, and data quality edge cases common in transactional systems.

Data Model Overview

The schema includes patients, staff, appointments, and billing tables with enforced foreign-key relationships and uniqueness constraints. Data was seeded intentionally to support both normal and edge-case scenarios.

Query Result Example 1

This query output illustrates how specific business questions are validated through SQL. Results are interpreted to confirm correct join behavior, filtering logic, and handling of incomplete or conflicting records.



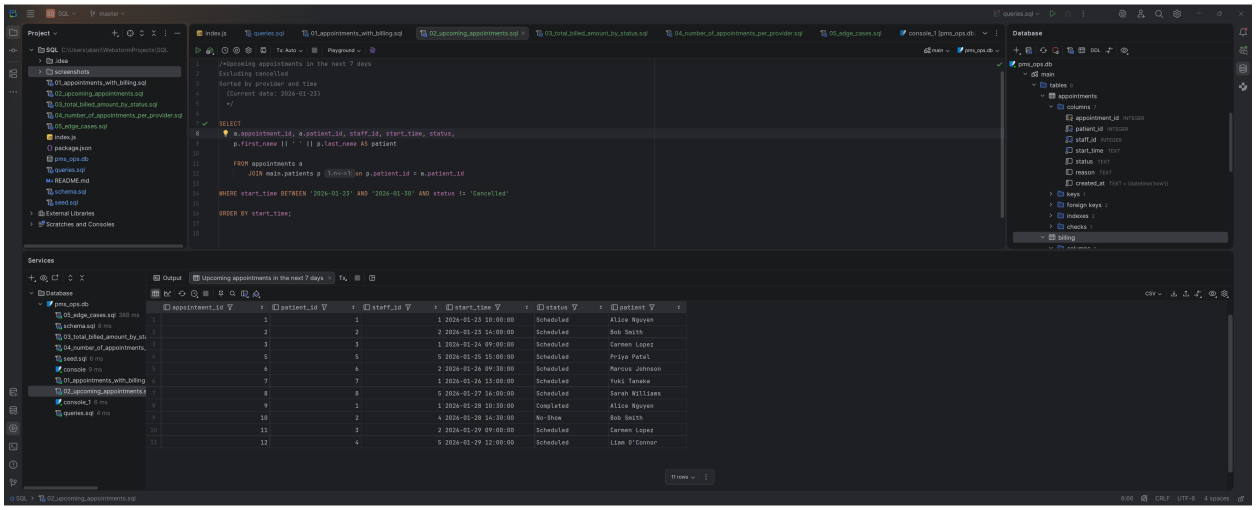
```sql
/*
What appointments are scheduled, who they're for, who's providing them, and what is the payment status*/

SELECT
    a.appointment_id,
    p.first_name || ' ' || p.last_name AS patient,
    s.first_name || ' ' || s.last_name AS provider,
    a.start_time,
    a.status,
    b.payment_status,
    b.amount_cents
FROM appointments a
         JOIN patients p ON p.patient_id = a.patient_id
         JOIN staff s ON s.staff_id = a.staff_id
         LEFT JOIN billing b ON b.appointment_id = a.appointment_id
ORDER BY a.start_time;
```

	appointment_id	patient	provider	start_time	status	payment_status	amount_cents
1	1	Alice Nguyen	Dana Harris	2026-01-23 10:00:00	Scheduled	Pending	12000
2	2	Bob Smith	Evan Miller	2026-01-23 14:00:00	Scheduled	Pending	8000
3	3	Carmen Lopez	Dana Harris	2026-01-24 09:00:00	Scheduled	Pending	9000
4	4	Liam O'Connor	Noah Brown	2026-01-24 11:30:00	Cancelled	Voided	0
5	5	Priya Patel	Olivia Martinez	2026-01-25 15:00:00	Scheduled	Pending	15000
6	6	Marcus Johnson	Evan Miller	2026-01-26 09:30:00	Scheduled	Pending	10000
7	7	Yuki Tanaka	Dana Harris	2026-01-26 13:00:00	Scheduled	Pending	18000
8	8	Sarah Williams	Olivia Martinez	2026-01-27 16:00:00	Scheduled	Pending	14000
9	9	Alice Nguyen	Dana Harris	2026-01-28 10:30:00	Completed	Paid	5000
10	10	Bob Smith	Noah Brown	2026-01-28 14:30:00	No-Show	Pending	9000
11	11	Carmen Lopez	Evan Miller	2026-01-29 09:00:00	Scheduled	Pending	11000
12	12	Liam O'Connor	Olivia Martinez	2026-01-29 12:00:00	Scheduled	Pending	13000
13	13	Priya Patel	Dana Harris	2026-01-30 11:00:00	Scheduled	Pending	12000
14	14	Marcus Johnson	Noah Brown	2026-01-30 15:30:00	Scheduled	Pending	16000
15	15	Yuki Tanaka	Evan Miller	2026-01-31 10:00:00	Cancelled	Pending	9000

15 rows

Query Result Example 2

This query output illustrates how specific business questions are validated through SQL. Results are interpreted to confirm correct join behavior, filtering logic, and handling of incomplete or conflicting records.



```sql
/*Upcoming appointments in the next 7 days
Excluding cancelled
Sorted by provider and time
  (Current date: 2026-01-23)
  */

SELECT
    a.appointment_id, a.patient_id, staff_id, start_time, status,
    p.first_name || ' ' || p.last_name AS patient

    FROM appointments a
        JOIN main.patients p on p.patient_id = a.patient_id

WHERE start_time BETWEEN '2026-01-23' AND '2026-01-30' AND status != 'Cancelled'

ORDER BY start_time;
```

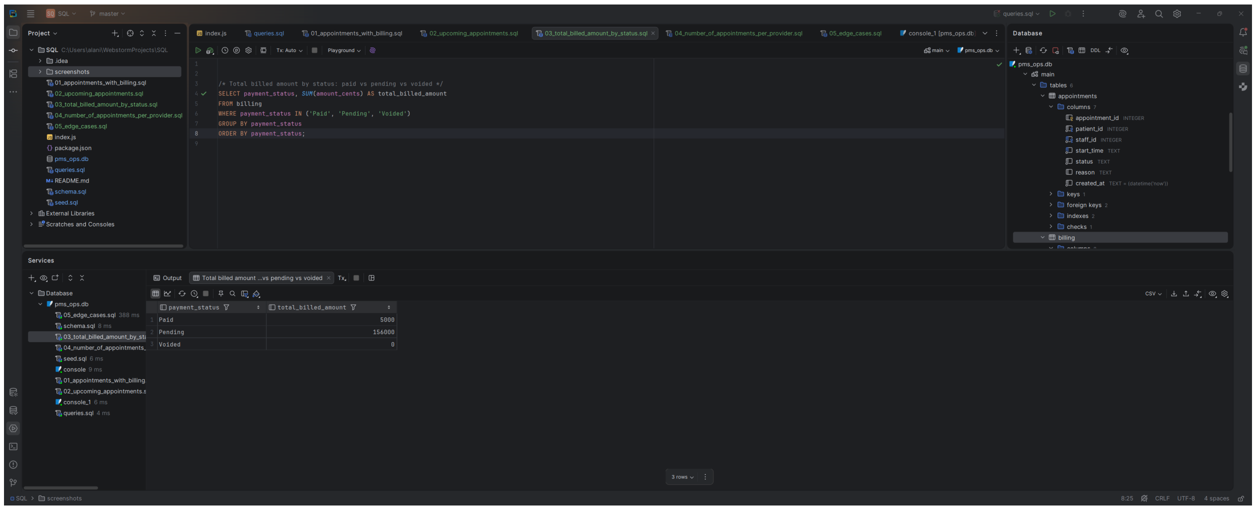
Upcoming appointments in the next 7 days

	appointment_id	patient_id	staff_id	start_time	status	patient
1	1	1	1	2026-01-23 10:00:00	Scheduled	Alice Nguyen
2	2	2	2	2026-01-23 14:00:00	Scheduled	Bob Smith
3	3	3	1	2026-01-24 09:00:00	Scheduled	Carmen Lopez
4	5	5	5	2026-01-25 15:00:00	Scheduled	Priya Patel
5	6	6	2	2026-01-26 09:30:00	Scheduled	Marcus Johnson
6	7	7	1	2026-01-26 13:00:00	Scheduled	Yuki Tanaka
7	8	8	5	2026-01-27 16:00:00	Scheduled	Sarah Williams
8	9	1	1	2026-01-28 10:30:00	Completed	Alice Nguyen
9	10	2	4	2026-01-28 14:30:00	No-Show	Bob Smith
10	11	3	2	2026-01-29 09:00:00	Scheduled	Carmen Lopez
11	12	4	5	2026-01-29 12:00:00	Scheduled	Liam O'Connor

11 rows

Query Result Example 3

This query output illustrates how specific business questions are validated through SQL. Results are interpreted to confirm correct join behavior, filtering logic, and handling of incomplete or conflicting records.

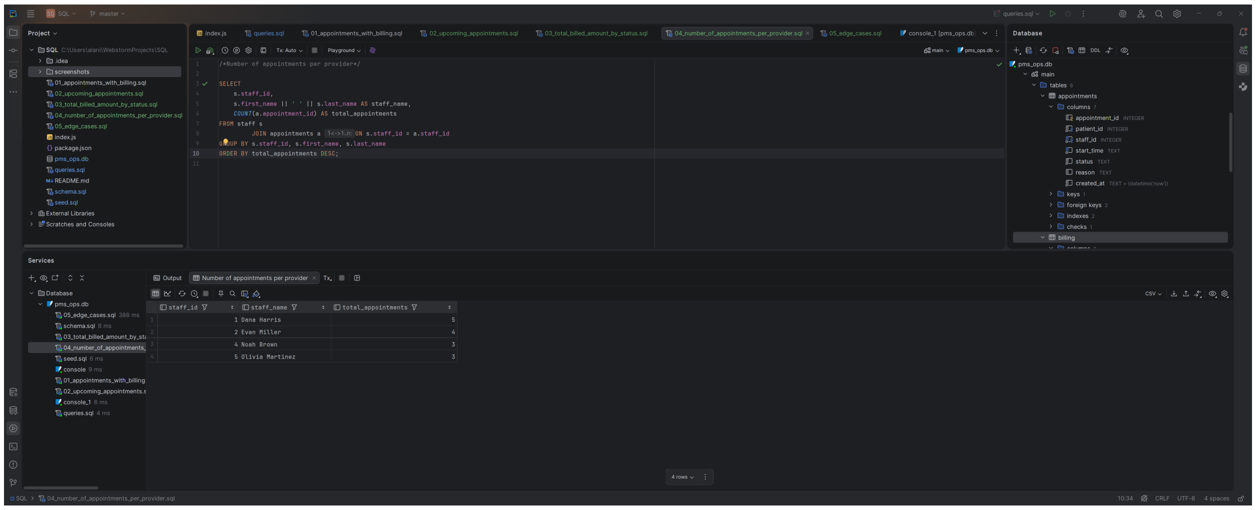


```sql
/* Total billed amount by status: paid vs pending vs voided */
SELECT payment_status, SUM(amount_cents) AS total_billed_amount
FROM billing
WHERE payment_status IN ('Paid', 'Pending', 'Voided')
GROUP BY payment_status
ORDER BY payment_status;
```

	payment_status	total_billed_amount
1	Paid	5000
2	Pending	156000
3	Voided	0

Query Result Example 4

This query output illustrates how specific business questions are validated through SQL. Results are interpreted to confirm correct join behavior, filtering logic, and handling of incomplete or conflicting records.



```sql
/*Number of appointments per provider*/

SELECT
    s.staff_id,
    s.first_name || ' ' || s.last_name AS staff_name,
    COUNT(a.appointment_id) AS total_appointments
FROM staff s
         JOIN appointments a ON s.staff_id = a.staff_id
GROUP BY s.staff_id, s.first_name, s.last_name
ORDER BY total_appointments DESC;
```

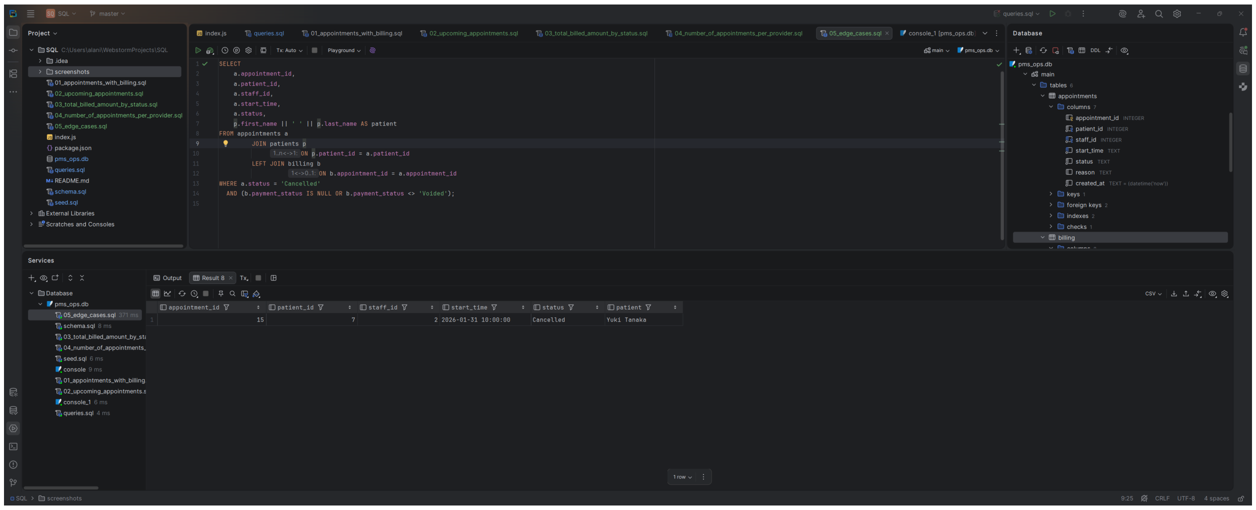
Number of appointments per provider

	staff_id	staff_name	total_appointments
1	1	Dana Harris	5
2	2	Evan Miller	4
3	4	Noah Brown	3
4	5	Olivia Martinez	3

4 rows

Query Result Example 5

This query output illustrates how specific business questions are validated through SQL. Results are interpreted to confirm correct join behavior, filtering logic, and handling of incomplete or conflicting records.



```sql
SELECT
    a.appointment_id,
    a.patient_id,
    a.staff_id,
    a.start_time,
    a.status,
    p.first_name || ' ' || p.last_name AS patient
FROM appointments a
         JOIN patients p
              ON p.patient_id = a.patient_id
         LEFT JOIN billing b
                   ON b.appointment_id = a.appointment_id
WHERE a.status = 'Cancelled'
  AND (b.payment_status IS NULL OR b.payment_status <> 'Voided');
```

```
  appointment_id  patient_id  staff_id  start_time           status     patient
1             15           7         2  2026-01-31 10:00:00  Cancelled  Yuki Tanaka
```

Analytical Focus & Edge Cases

Special attention was given to scenarios such as cancelled appointments with active billing, missing billing records, and enforcement of one-to-one relationships where appropriate. These cases demonstrate deliberate query design and data validation practices.

Conclusion

This SQL portfolio artifact complements broader systems analysis work by showing practical data reasoning, disciplined schema design, and clarity in analytical outputs. It reflects how SQL is used in real environments to support decision-making and reporting.