Emergency Room Operations & Patient Flow Dashboard

Operational Performance & Patient Experience Analysis

Role: Data Analyst
Tools: SQL | Excel (Power Query, Power Pivot, DAX)
Industry: Healthcare Analytics
Time Period: Jan 2026
Data Source: Kaggle

Project Overview

Project Title

Emergency Room Operations & Patient Flow Dashboard

Objective

The objective of this project is to analyze Emergency Room (ER) operations using hospital visit data to:

- Monitor patient flow
- Reduce waiting time
- Improve patient satisfaction
- Support operational decision-making

SQL is used as the **backend analytics layer**, while Excel is used for dashboard visualization.

Business Problem

Emergency Rooms face challenges such as:

- Long patient wait times
- Overcrowding during peak hours
- Uneven department workload
- Low patient satisfaction

Business Questions

- **Patient Volume** – How many patients visit the ER daily and monthly?
- **Average Wait Time** – How long do patients wait on average before being seen?
- **SLA Compliance** – What percentage of patients are seen within 30 minutes?
- **Department Load** – Which departments or services have the highest ER visits?
- **Impact on Satisfaction** – Does a longer wait time reduce patient satisfaction?

Dataset Overview

Source: Hospital ER operational data
Records: Patient-level visit data
Key Columns:
Patient ID
Visit Date & Time
Gender
Age
Department Referral
Admission Flag
Patient Wait Time (minutes)
Satisfaction Score

Tools & Technologies

SQL: Data cleaning, transformation, KPI calculation
Excel Power Query: ETL & preprocessing
Power Pivot (DAX): KPIs & calculated columns
Pivot Tables & Charts: Analysis
Excel Dashboard: Final visualization

Overall Workflow

1. Raw data ingested into SQL
2. Data cleaned & transformed using SQL queries
3. Aggregated tables exported to Excel
4. Power Query used for final cleaning
5. Power Pivot used for calculations
6. Dashboard created for stakeholders

SQL: Staging Table Creation

```sql
CREATE TABLE public.staging_er_data (
    patient_id              VARCHAR(50),
    admission_datetime      VARCHAR(50),
    first_initial           VARCHAR(5),
    last_name               VARCHAR(100),
    gender                  VARCHAR(10),
    age                     INTEGER,
    race                    VARCHAR(50),
    department              VARCHAR(100),
    admission_flag          VARCHAR(10),
    satisfaction_score      INTEGER,
    wait_time               INTEGER,
    patients_cm             VARCHAR(10)
);
```

Why?

To load raw CSV data without data loss.

1.Initial Data Profiling & Validation

1.1 Preview Data

```sql
SELECT *
FROM staging_er_data
LIMIT 5;
```

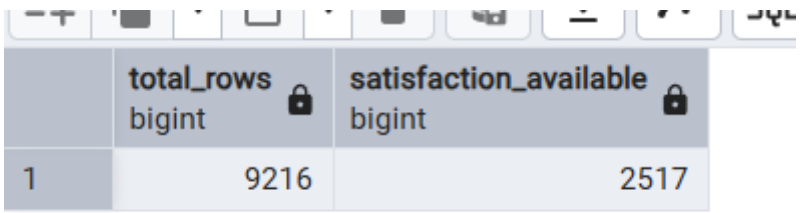
1.2 Record Count Validation

```sql
SELECT COUNT(*) AS total_patients
FROM staging_er_data;
```

	total_patients bigint
1	9216

1.3 Satisfaction Score Availability

```sql
SELECT
    COUNT(*) AS total_rows,
    COUNT(satisfaction_score) AS satisfaction_available
FROM staging_er_data;
```



	total_rows bigint	satisfaction_available bigint
1	9216	2517

📌 **Insight**
Not all patients provided satisfaction scores — handled later using filters.

2. Data Cleaning & Transformation

2.1 Gender Standardization

```sql
UPDATE staging_er_data
SET gender = CASE
    WHEN gender = 'M' THEN 'Male'
    WHEN gender = 'F' THEN 'Female'
    ELSE 'Other'
END;
```

📌 Ensures consistent gender reporting for analysis & charts.

2.2 Patient Name Creation

```
ALTER TABLE staging_er_data
ADD patient_name VARCHAR(100);
```

```
UPDATE staging_er_data
SET patient_name = CONCAT(first_initial, '. ', last_name);
```

📌 Improves readability for reporting (used in Excel views).

2.3 Admission Status Conversion

```
ALTER TABLE staging_er_data
ADD admission_status VARCHAR(20);

UPDATE public.staging_er_data
SET admission_status = CASE
    WHEN admission_flag ILIKE 'Y%' THEN 'Admitted'
    ELSE 'Not Admitted'
END;
```

📌 Converts cryptic flags into **business-friendly labels**.

2.4 Date & Time Extraction

```
ALTER TABLE staging_er_data
ADD COLUMN visit_date DATE,
ADD COLUMN visit_time TIME;

UPDATE staging_er_data
SET visit_date = TO_TIMESTAMP(admission_datetime, 'DD-MM-YYYY HH24:MI')::DATE,
    visit_time = TO_TIMESTAMP(admission_datetime, 'DD-MM-YYYY HH24:MI')::TIME;
```

📌 Enables:

- Daily trends
- Monthly analysis
- Peak hour analysis

2.5 Age Group Bucketing

```sql
ALTER TABLE staging_er_data
ADD COLUMN age_group VARCHAR(20);

UPDATE staging_er_data
SET age_group = CASE
    WHEN age IS NULL THEN 'Unknown'
    WHEN age BETWEEN 0 AND 9 THEN '0-9'
    WHEN age BETWEEN 10 AND 19 THEN '10-19'
    WHEN age BETWEEN 20 AND 29 THEN '20-29'
    WHEN age BETWEEN 30 AND 39 THEN '30-39'
    WHEN age BETWEEN 40 AND 49 THEN '40-49'
    WHEN age BETWEEN 50 AND 59 THEN '50-59'
    WHEN age BETWEEN 60 AND 69 THEN '60-69'
    ELSE '70+'
END;
```

- 📌 Aligns with healthcare demographic reporting standards.

2.6 Wait Time SLA Classification

```sql
ALTER TABLE staging_er_data
ADD COLUMN wait_time_status VARCHAR(20);


UPDATE staging_er_data
SET wait_time_status = CASE
    WHEN wait_time IS NULL THEN 'Unknown'
    WHEN wait_time <= 30 THEN 'On Time'
    ELSE 'Delayed'
END;
```

📌 Critical KPI for **ER service efficiency**.

3. Calendar Dimension (Time Intelligence)

3.1 Calendar Table Creation

```sql
CREATE TABLE dim_calendar (
    date_id DATE PRIMARY KEY,
    year INT,
    month INT,
    month_name VARCHAR(15),
    weekday_name VARCHAR(15)
);
```

3.2 Populate Calendar

```sql
INSERT INTO dim_calendar
(date_id, year, month, month_name, weekday_name)
SELECT
    d::DATE,
    EXTRACT(YEAR FROM d)::INT,
    EXTRACT(MONTH FROM d)::INT,
    TO_CHAR(d, 'FMMonth'),
    TO_CHAR(d, 'FMDay')
FROM generate_series(
        '2023-01-01'::DATE,
        '2024-12-31'::DATE,
        INTERVAL '1 day'
    ) AS d;
```

📌 Enables:

- Monthly trends
- Year-over-year comparison
- Consistent time slicing

4. KPI Calculations (SQL Metrics)

4.1 KPI Calculations (SQL Metrics)

```sql
SELECT COUNT(DISTINCT patient_id)
AS total_patients
FROM staging_er_data;
```

	total_patients bigint
1	9216

4.2 Average Wait Time

```sql
--Average Wait Time
SELECT ROUND(AVG(wait_time),2) AS avg_wait_time
FROM staging_er_data;
```

Data Output | Messages | Notifications

	avg_wait_time numeric
1	35.26

4.3 Average Satisfaction Score

```sql
--Average Satisfaction Score
SELECT ROUND(AVG(satisfaction_score),2) AS avg_satisfaction
FROM staging_er_data
WHERE satisfaction_score IS NOT NULL;
```

Data Output | Messages | Notifications

	avg_satisfaction numeric
1	4.99

4.4 SLA Performance

```sql
SELECT wait_time_status, COUNT(*) AS patients
FROM staging_er_data
GROUP BY wait_time_status;
```

Data Output | Messages | Notifications

	wait_time_status character varying (20)	patients bigint
1	On Time	3749
2	Delayed	5467

4.5 Department Referral Analysis

```sql
--Department Referrals
SELECT department, COUNT(*) AS total_patients
FROM staging_er_data
GROUP BY department
ORDER BY total_patients DESC;
```

Data Output | Messages | Notifications

	department character varying (100)	total_patients bigint
1	None	5400
2	General Practice	1840
3	Orthopedics	995
4	Physiotherapy	276
5	Cardiology	248
6	Neurology	193
7	Gastroenterology	178
8	Renal	86

📌 Identifies high-load departments.

4.6 Satisfaction vs Wait Time

```sql
SELECT
    CASE
        WHEN wait_time < 20 THEN 'Low Wait'
        WHEN wait_time BETWEEN 20 AND 40 THEN 'Medium Wait'
        ELSE 'High Wait'
    END AS wait_category,
    ROUND(AVG(satisfaction_score),2) AS avg_satisfaction
FROM er_data_cleaned
GROUP BY wait_category;
```

📌 Demonstrates direct impact of wait time on satisfaction.

4.7 Monthly Patient Trend

```sql
--Monthly Trend
SELECT
    c.year,
    c.month_name,
    COUNT(*) AS total_patients
FROM staging_er_data s
JOIN dim_calendar c
ON s.visit_date = c.date_id
GROUP BY c.year, c.month_name, c.month
ORDER BY c.year, c.month;
```

📌 Supports **capacity planning & seasonal analysis**.

Excel Power Query (ETL)

Power Query was used to:

- Remove blank rows
- Convert data types
- Rename columns
- Split date & time
- Create calendar table

Patient Name Creation

```
= Text.Combine({[First Initial], ". ", [Last Name]})
```

Calendar Table (Excel)

```
= List.Dates( #date(2023,1,1),731,
  #duration(1,0,0,0))
```

Used for:

- Monthly trends
- Year slicers
- Time intelligence

Pivot (DAX Calculations)

```
Total Patients =
DISTINCTCOUNT(ER_Data[Patient ID])


Average Wait Time =
AVERAGE(ER_Data[Patient Wait Time])
```

DAX Age GroupAge

```
Group =
IF([Patient Age] > 70, "70+",
IF([Patient Age] > 60, "60-69",
IF([Patient Age] > 50, "50-59",
IF([Patient Age] > 40, "40-49",
IF([Patient Age] > 30, "30-39",
IF([Patient Age] > 20, "20-29",
IF([Patient Age] > 10, "10-19", "0-9")))))))
```

DAX SLA Measure

```
Wait Time Status =
IF([Patient Wait Time] > 30, "Delayed", "On Time")
```

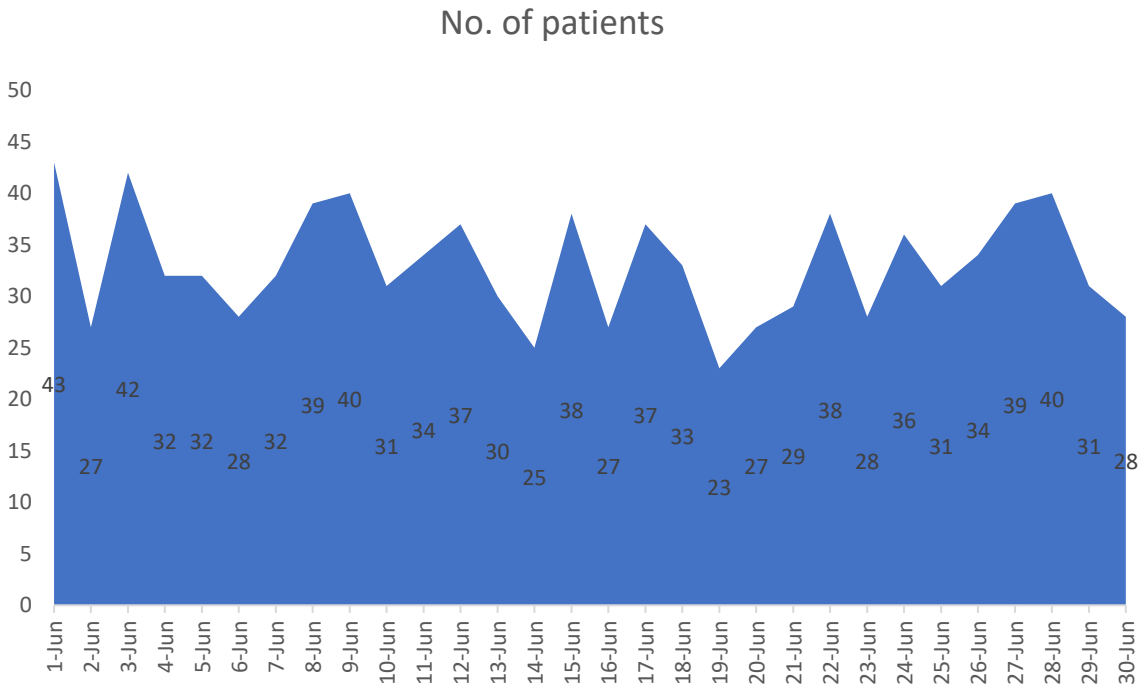
Dashboard KPIs

- Total Patients
- Average Wait Time
- Average Satisfaction Score
- % Delayed Patients

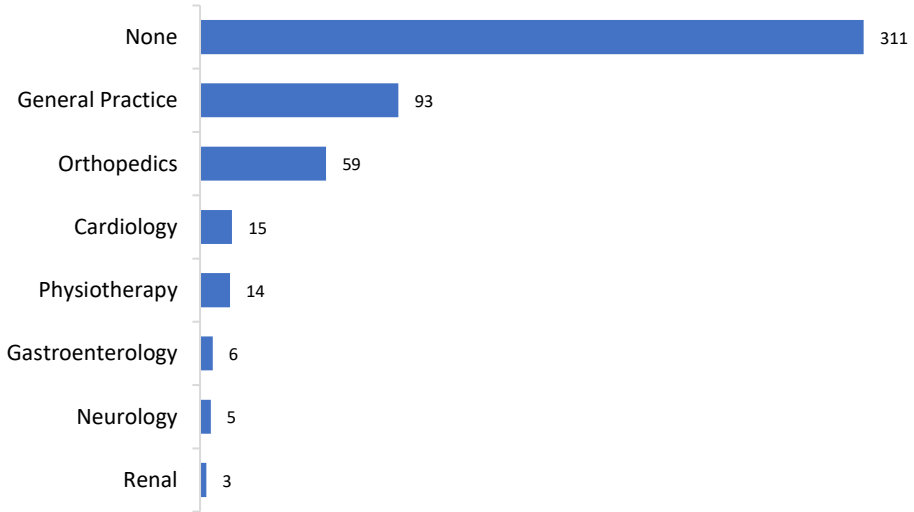
📌 KPIs update dynamically using slicers

Dashboard Visuals

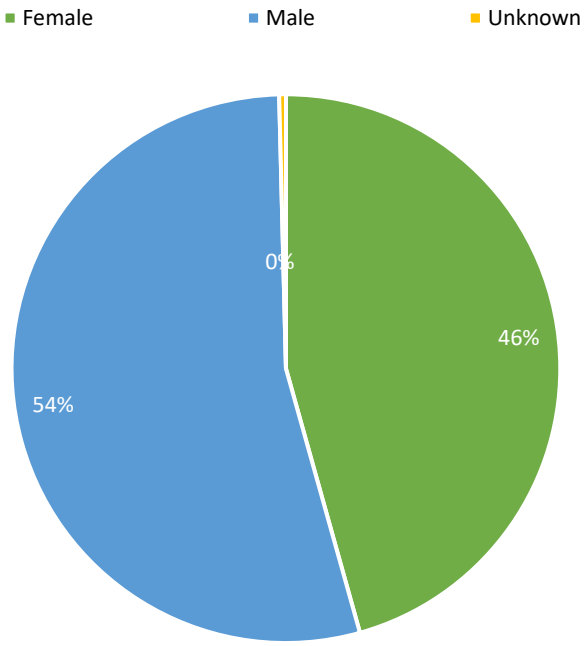
Daily patient trend



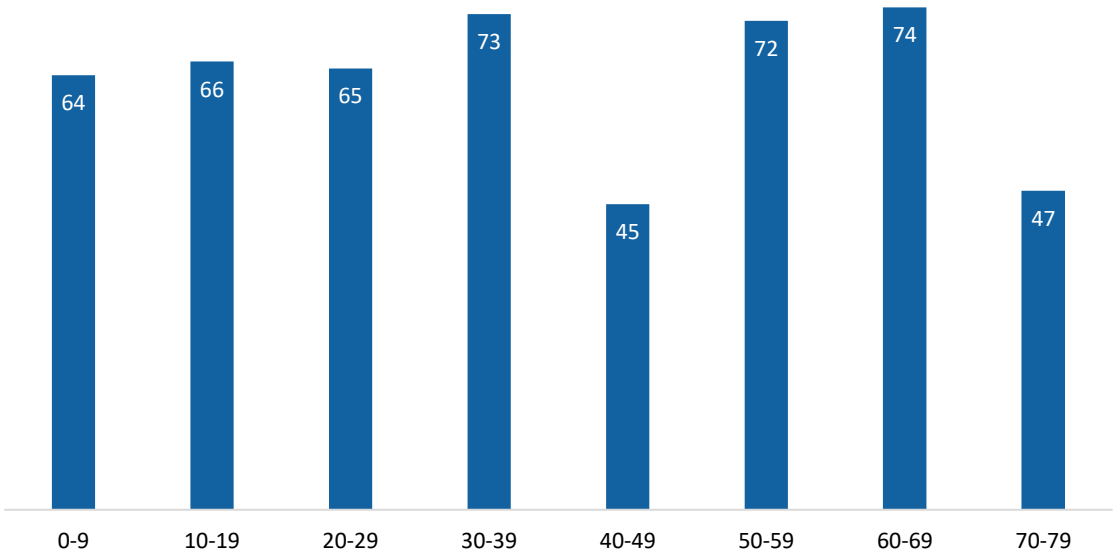
Department referrals



Gender distribution



Age groups



SLA performance

Admission Statu	Patient Count	Percentage (%)
Not Admitted	253	50.00%
Admitted	253	50.00%

Interactivity

- Year slicer
- Month slicer
- Cross-filtering across all visuals

Slicer → Report Connections → All Pivot Tables

Key Insights

- Peak patient inflow during mid-week
- Majority patients aged 30–49
- Certain departments consistently overloaded
- Delayed patients show lower satisfaction scores

Business Recommendations

- Increase staffing during peak days
- Improve triage for delayed patients
- Optimize high-referral departments
- Monitor SLA breaches daily

- **Project Outcome**
- ✅ **Key Outcomes Delivered**
- **SQL-driven analysis**
  Used SQL to clean, transform, and analyze raw ER patient data, ensuring accurate and reliable insights.
- **Excel dashboard for stakeholders**
  Built an interactive Excel dashboard with KPIs and slicers to enable hospital management to monitor ER performance in real time.
- **Data-driven decision support**
  Provided actionable insights on patient wait times, department workload, and satisfaction levels to support operational improvements.
- **Delivered** an end-to-end healthcare analytics project using SQL and Excel, covering data cleaning, modeling, KPI design, and dashboarding for stakeholder decision-making.

Skills Demonstrated

Technical & Analytical Skills

- **SQL querying & data modeling**
  Wrote optimized SQL queries for data cleaning, feature engineering, aggregations, and KPI calculations.
- **KPI design & business metrics**
  Defined and calculated key healthcare KPIs such as total patients, average wait time, SLA performance, and satisfaction score.
- **Power Query (ETL)**
  Performed data extraction, cleaning, transformation, and calendar table creation using Power Query.
- **DAX calculations (Power Pivot)**
  Created calculated columns and measures for age groups, wait-time status, and dynamic KPIs.
- **Dashboard storytelling & presentation**
  Designed a user-friendly dashboard and translated data insights into clear business recommendations.

Thank You

Questions?