## Explanation of Queries:

### Task 1: . Doctor-wise OPD load monthly; top 5 busiest per branch.

This query measures the monthly patient load for every doctor.
By using strftime('%Y-%m', visit_datetime), all visits are aggregated at the month level.
Joining the **Doctor** and **OPD_Visit** tables ensures that the workload is mapped correctly with doctor details such as name and specialization.

This analysis helps management:

- Identify doctors with consistently high workloads
- Balance staffing between branches
- Detect seasonal variations in patient volume

It forms the foundation for strategic workforce planning.

### Task 2: New vs follow-up ratio per branch per month

This task enhances Task 1 using **Window Functions** to rank high-performing doctors each month.

The CTEs used—monthly_load and ranked—segment the data into manageable stages.
The key concept is:

ROW_NUMBER() OVER (PARTITION BY month ORDER BY total_visits DESC)

- PARTITION BY month resets ranking for each month
- Doctors are ranked based on total visits
- Filtering for ROW_NUMBER() <= 5 selects the top 5

This report is extremely useful for identifying:

- Monthly high-performers
- Doctors who may require support due to overload
- Candidates for incentives or recognition

### Task 3 – Top 3 diagnoses per specialization.

This query identifies the **most common diagnoses** within each medical specialization.

By joining **OPD_Diagnosis**, **OPD_Visit**, and **Doctor**, every diagnosis gets mapped to the doctor who made it and their specialty.
Aggregating these counts gives a clear view of disease prevalence.

Using ROW_NUMBER() OVER (PARTITION BY specialization ORDER BY count_diag DESC) selects the top 3 diagnoses for every category.

This insight supports:

- Predictive treatment planning
- Pharmacy inventory optimization
- Identification of seasonal or specialization-specific disease trends

### Task 4 – Most prescribed medicines with patient count.

This query counts the number of **unique patients** prescribed each medicine.

Using COUNT(DISTINCT visit_id) or COUNT(DISTINCT patient_id) prevents inflated numbers caused by multiple prescriptions within the same visit.

Sorting the results identifies the most frequently prescribed medicines, supporting:

- Pharmacy stocking decisions
- Drug procurement and negotiation with vendors
- Monitoring adherence to standard treatment protocols

### Task 5 – Monthly Revenue per Branch (Gross & Net)

This query calculates two major financial KPIs:

1. **Gross Revenue** – income before discounts
2. **Net Revenue** – actual realized revenue after discount adjustment

By grouping at the branch-month level, this analysis gives:

- Financial performance of each branch
- Trends over months
- Insights for budgeting and cost control

Using the billing table ensures accurate, transaction-level revenue reporting.

### Task 6 – Average Ticket Size by Payment Mode

This query computes the **average revenue per bill** based on the payment mode.

AVG(consultation_fee + additional_charges - discount_amount) represents the true financial value of each bill.

This helps identify:

- Whether insurance cases have higher bills
- Whether UPI or cash transactions show different spending patterns
- Consistency of pricing across payment methods

This information is relevant for financial strategy and billing policy decisions.

## Task 7 – Doctor Performance (Visits, Revenue & Average Ticket Size)

This query provides a complete performance scorecard for every doctor.

Metrics calculated:

- **Total Visits** – workload/volume contribution
- **Total Revenue** – financial contribution
- **Average Ticket Size** – efficiency/quality of consultations

Using LEFT JOIN ensures that even doctors who had zero visits appear in the report.

Ranking doctors by total revenue helps management:

- Identify top contributors
- Evaluate high-earning specialties
- Allocate resources more effectively

## Task 7.1 – Top 3 Doctors per Specialization

This task ranks doctors **within their own specialization**, making comparisons fair and relevant.

ROW_NUMBER() OVER (PARTITION BY specialization ORDER BY total_revenue DESC) ensures:

- A cardiologist is compared only with other cardiologists
- A general physician is not compared with a surgeon

This produces accurate top-performer lists for each department and assists in:

- Department-level evaluation
- Role-specific incentives
- Annual appraisal reviews

## Task 8 – Peak Hour Analysis per Branch

This query identifies the **busiest hour** for patient visits in each branch.

Step 1: CTE hourly_visits computes visits per hour using strftime('%H', visit_datetime)
Step 2: Window function ranks hours for each branch
Step 3: Filtering for rank=1 gives the **peak hour**

This insight is essential for:

- Optimizing staff schedules
- Managing front-desk and nursing workloads
- Reducing patient wait time
- Improving resource availability during rush hours