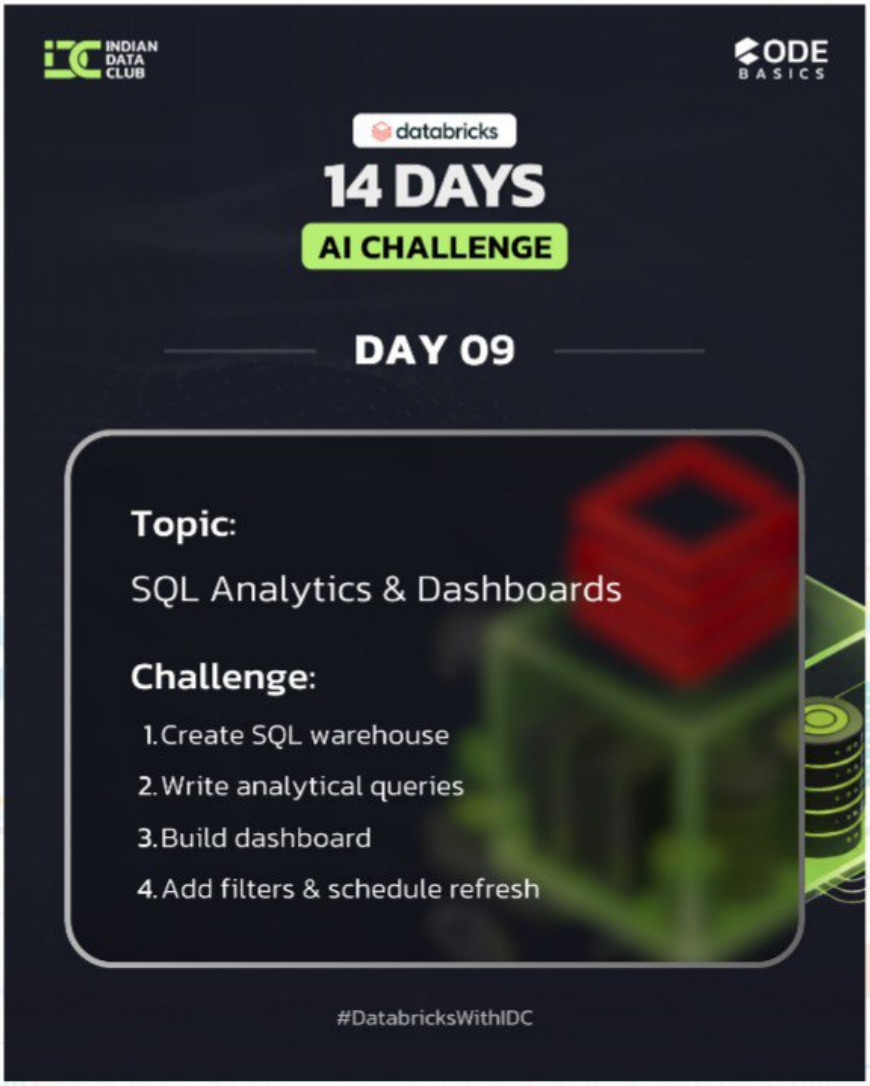
Day 9 - SQL Analytics & Dashboards



SQL Warehouses

- *What is a SQL Warehouse?*

A SQL Warehouse in Databricks is a compute engine optimized for:
•Running SQL queries
•Powering dashboards
•BI tools (Databricks dashboards, Power BI, Tableau)

Think of it as:
💡 *A fast SQL-only engine that reads your Delta tables and returns results for analytics.*

Why do we need SQL Warehouses?

Because:
- *Notebooks = development*
- *SQL Warehouses = analytics & reporting*

They are:
- ✔ *Faster for SQL*
- ✔ *Auto-scalable*
- ✔ *Cost-efficient*
- ✔ *Required for dashboards*

Where it fits in Medallion Architecture

- *Bronze/Silver/Gold tables → stored in Delta*
- *SQL Warehouse → queries Gold tables*
- *Dashboards → built on SQL Warehouse*

How to create SQL Warehouse (UI steps)

1.Go to Databricks → SQL
2.Click SQL Warehouses
3.Click Create Warehouse
4.Choose:
Size: Small (for learning)
Auto-stop: 5–10 minutes
5.Click Create

✅ Now your warehouse is READY

What are Analytical Queries?

🔍 *Analytical Queries*

SQL queries used to:

- *Analyze large datasets*
- *Discover trends & patterns*
- *Support business decisions*

📌 *Focus on:*

- *Aggregations*
- *Rankings*
- *Trends*
- *Comparisons*

Why Analytical Queries Matter?

💡 *Business Value*

Analytical queries help answer:

- *How much revenue did we make?*
- *Which products perform best?*
- *Are sales growing or declining?*
- *Where are we losing customers?*

📈 *Foundation for dashboards & reports*

Basic Aggregations

🧮 Core SQL Aggregates

Common functions:

- SUM() → Total revenue
- COUNT() → Number of orders
- AVG() → Average order value
- MIN() / MAX() → Range analysis

📌 Used with GROUP BY

Revenue Analysis

📈 *Revenue by Day*

```
SELECT order_date,
       SUM(total_amount) AS
daily_revenue
FROM gold_orders
GROUP BY order_date;
```

📌 *Helps track:*

- *Sales trends*
- *Business growth over time*

Grouped Analytics

📦 *Business-Level Insights*

Examples:

- *Revenue by category*
- *Revenue by product*
- *Orders by region*

```
SELECT category,
       SUM(total_amount) AS revenue
FROM gold_orders
GROUP BY category;
```

📌 *Answers "Where is money coming from?"*

Top Products Analysis

🏆 *Best Performing Products*

```sql
SELECT product_name,
       SUM(total_amount) AS revenue
FROM gold_orders
GROUP BY product_name
ORDER BY revenue DESC
LIMIT 5;
```

📌 *Used in:*

Leaderboards

Top products dashboards

Window Functions (Advanced SQL)

⭐ *Why Window Functions?*

They allow:

- *Ranking*
- *Running totals*
- *Comparisons across rows*
- *Without losing row-level data*

📌 *Syntax:*

```
FUNCTION() OVER (PARTITION BY ...
ORDER BY ...)
```

Running Total Revenue

📈 *Growth Tracking*

```
SUM(SUM(total_amount))
OVER (ORDER BY order_date)
```

📌 *Used for:*
Cumulative revenue charts
Business growth analysis

Ranking Products

🥇 Product Ranking

```
RANK() OVER (ORDER BY revenue DESC)
```

📌 Helps identify:
Top performers
Low-performing products

Funnel Analysis

🔄 *Order Funnel*

Stages:

- *Placed*
- *Shipped*
- *Delivered*
- *Cancelled*

```sql
SELECT order_status, COUNT(*)
FROM gold_orders
GROUP BY order_status;
```

How This Powers Dashboards

 *From SQL → Dashboard*

Analytical queries feed:

- *Revenue trend charts*
- *Top product visuals*
- *Funnel & KPI tiles*

🚀 *SQL = backbone of analytics dashboards*

What is a Databricks Dashboard?

A Databricks SQL Dashboard is a collection of:

- *SQL queries*
- *Visualizations*
- *Filters*

Connected to:

✔ *SQL Warehouse*

✔ *Gold tables*

Prerequisites (Very Important)

Before creating dashboard, ensure :

✅ *SQL Warehouse is running*
✅ *Gold tables exist*
✅ *Analytical SQL queries are ready*

Step 1 – SQL Editor

👩‍💻 *Where Dashboards Begin*

1.Go to Databricks → SQL → SQL Editor

2.Select SQL Warehouse

3.Run analytical SQL queries

📌 *Each metric = one SQL query*

Step 2 – Create Metric Queries

📌 *Example Metrics*

Total Revenue (KPI)

•Daily Revenue Trend

•Top Products

•Order Status Funnel

Each query is:

•Run

•Saved

•Reused in dashboards

Step 3 – Create Dashboard

📊 *Dashboard Setup*

Steps:

1.Go to Dashboards

2.Click Create 3.Dashboard

4.Give a meaningful name

📌 *Example: E-Commerce Sales Dashboard*

Step 4 – Add Visualizations

📈 *From Query to Chart*

·Add saved SQL query

·Choose visualization type

·Preview & adjust

📌 *One visualization per query*

Step 5 – Dashboard Layout

🧩 *Designing the Dashboard*

Best layout:

- *KPIs at the top*
- *Trends in the middle*
- *Detailed breakdowns below*

📌 *Keep it simple & readable*

Step 6 – Save & Share

🔐 *Collaboration*

- *Save dashboard*
- *Share with users or groups*
- *Set permissions (View / Edit)*

📌 *Dashboards are meant to be shared*

Best Practices ⭐

🚀 Dashboard Design Tips

- ✔ One metric per tile
- ✔ Clear titles & labels
- ✔ Avoid clutter
- ✔ Business-friendly visuals
- ✔ Always use Gold data

Visualization – What & Why

What is Visualization?
Visual representation of data using charts & graphs.

Why it matters:
- *Makes insights easy to understand*
- *Highlights trends & patterns*
- *Helps faster decision-making*

📌 *One chart = one clear business question*

Visualization – Types & Best Use

Common Visualization Types:

KPI → Key numbers
Line → Trends over time
Bar → Comparison
Pie → Proportion
Funnel → Process stages

📌 *Choosing the right chart = correct insight*

Filters – What & Why

What are Filters?
Filters allow users to dynamically slice dashboard data.

Why filters matter:
- *One dashboard → multiple views*
- *Personalized analysis*
- *No SQL changes needed*

📌 *Filters make dashboards interactive*

Filters – Examples & Usage

Common Filters:

- *Date range*
- *Product category*
- *Order status*
- *Region*

📌 *Filters connect to SQL parameters and update visuals instantly*

Scheduling – What & Why

What is Scheduling?
Automated refresh of dashboards at fixed intervals.

Why scheduling matters:
- *Always fresh data*
- *No manual refresh*
- *Reliable reporting*

📌 *Automation builds trust in data*

Scheduling – Best Practices

Scheduling Tips:

- *Choose frequency wisely (daily/hourly)*
- *Avoid peak hours*
- *Optimize heavy queries*
- *Monitor failures*

📌 *Smart scheduling = cost + performance efficiency*