📌 Data Warehousing & ETL – Interview Preparation Guide with Real-time Examples

1️⃣ What is a Data Warehouse?

A **Data Warehouse (DWH)** is a centralized repository designed for **analytical processing** rather than transactional processing. It helps businesses store **historical data** for insights, reporting, and decision-making.

✅ **Real-time Example:**
Imagine you run an **e-commerce company like Amazon**. Every day, millions of users place orders, browse products, and leave reviews. This data is stored in **transactional databases (OLTP)**, but for business analytics (e.g., monthly sales reports, customer trends, and product performance), it needs to be stored in a **data warehouse (OLAP)**.

2️⃣ Database vs. Data Warehouse

Feature	Database (OLTP) 🔷	Data Warehouse (OLAP) 🔶
Purpose	Transactional operations (CRUD)	Analytical processing & reporting
Data Type	Current data (day-to-day transactions)	Historical data (aggregated over time)
Example	Banking system storing customer deposits	Business intelligence dashboard analyzing customer trends
Performance	Optimized for quick transactions	Optimized for complex queries & aggregations

Storage Cost	Low	High (due to large historical data)

✅ **Real-time Example:**
A bank uses a **database** to process real-time transactions (withdrawals, deposits). However, for analyzing customer spending patterns over the past 5 years, they use a **data warehouse**.

---

3 What is Data Warehousing?

Data Warehousing is the **process of collecting, storing, and managing** large amounts of data from different sources into a **centralized repository**.

✅ **Real-time Example:**
A **hospital chain** collects patient records from multiple hospitals and clinics. By integrating all data into a **data warehouse**, they can analyze disease patterns, optimize resource allocation, and improve healthcare services.

---

4 ETL (Extract, Transform, Load) Layers

ETL is a **process** used to extract data from multiple sources, transform it into a structured format, and load it into a data warehouse.

◈ Extract – Retrieve data from multiple sources (databases, APIs, logs, files).

◈ Transform – Clean, filter, deduplicate, and convert data into a structured format.

◈ Load – Store the processed data into the data warehouse.

✅ **Real-time Example:**
A **retail company like Walmart** collects data from:

- **POS (Point-of-Sale) systems** for daily sales
- **Supplier databases** for inventory
- **Customer databases** for purchase history

Using ETL, this data is processed and stored in a **data warehouse** for analytics.

---

5 Incremental Loading

Instead of loading all data repeatedly, **incremental loading** updates only the new or modified records.

✅ **Real-time Example:**
A **ride-sharing app like Uber** stores ride history in a database. Instead of loading the entire data daily, they use **incremental ETL** to update only **new rides** into the data warehouse.

---

6 Databricks Overview & Free Account Setup

Databricks is a **cloud-based big data processing** platform built on Apache Spark, designed for fast, scalable, and collaborative data processing.

✅ **Real-time Example:**
A **social media company like Instagram** analyzes user activity (likes, shares, comments) using **Databricks + Spark SQL** to generate personalized content recommendations.

---

7 Incremental Data Loading using Spark SQL

- Use `MERGE` in **Spark SQL** to perform **incremental updates**
- Identify new records using a **timestamp** or **primary key**

✅ **Real-time Example:**
A **stock market analytics platform** updates stock prices every second. Using **incremental loading with Spark SQL**, they only update changed prices instead of reloading the entire dataset.

---

8 What is Data Modeling?

Data Modeling is the **process of designing** the structure of a database or data warehouse.

✅ **Real-time Example:**
A **food delivery company like Swiggy/Zomato** models their data to track:

- **Customers** (customer_id, name, address)
- **Orders** (order_id, customer_id, total_amount, timestamp)
- **Restaurants** (restaurant_id, name, cuisine)

This structured approach enables **faster reporting and analysis**.

---

9 What is Dimensional Data Modeling?

A technique used in data warehouses to organize data into **facts and dimensions**.

✅ **Real-time Example:**
A **supermarket chain** wants to analyze sales trends:

- **Fact Table (Sales_Fact):** Contains sales amount, quantity sold
- **Dimension Tables:** Product, Store, Customer, Time

This allows analysis like: **"Which product sells the most in December?"**

---

10 Fact Table & Dimension Tables

🔷 **Fact Table** – Stores measurable data (e.g., sales, revenue).
🔷 **Dimension Table** – Stores descriptive data (e.g., customer, location, product).

✅ **Real-time Example:**
An **airline company** tracks flights:

- **Fact Table:** Flight sales, ticket revenue
- **Dimensions:** Date, Airline, Destination, Customer

---

11 STAR Schema vs. SNOWFLAKE Schema

⭐ **STAR Schema:** A single **fact table** with multiple **dimension tables**.
❄️ **SNOWFLAKE Schema:** Dimensions are **normalized** to reduce redundancy.

✅ **Real-time Example:**
A **Netflix-style streaming platform**:

- **STAR Schema:** Simpler queries for quick reports
- **SNOWFLAKE Schema:** Reduces storage cost for massive datasets

---

1️⃣2️⃣ Types of Fact & Dimension Tables

📊 Fact Tables:

- **Transactional Fact Table** – Stores business transactions (e.g., purchases).
- **Snapshot Fact Table** – Stores periodic snapshots (e.g., monthly account balances).
- **Accumulating Fact Table** – Tracks the progress of events over time (e.g., order processing).

✅ **Real-time Example:**
A **loan processing system** uses an **Accumulating Fact Table** to track the status of loans: **Applied → Approved → Disbursed → Closed**.

📊 Dimension Tables:

- **Conformed Dimension** – Used across multiple fact tables (e.g., Customer).
- **Junk Dimension** – Stores miscellaneous attributes (e.g., order status codes).
- **Role-Playing Dimension** – Used for different purposes (e.g., Order Date vs. Delivery Date).

1️⃣3️⃣ Slowly Changing Dimensions (SCD)

SCD handles **historical changes** in dimension data.

- **Type 1:** Overwrites old data.
- **Type 2:** Keeps history with versioning.
- **Type 3:** Keeps both old and new values in the same row.

✅ **Real-time Example:**
A **telecom company** tracks customers:

- **SCD Type 1:** Updates new addresses, losing history.
- **SCD Type 2:** Maintains history of address changes.

1️⃣4️⃣ Implementing SCD Type 1 in Databricks with Spark SQL

sql

```
MERGE INTO customer_dim AS target

USING customer_updates AS source

ON target.customer_id = source.customer_id

WHEN MATCHED THEN

    UPDATE SET target.address = source.address

WHEN NOT MATCHED THEN

    INSERT (customer_id, name, address) VALUES (source.customer_id,
source.name, source.address)
```

✅ **Real-time Example:**
An **online banking system** updates customer details but doesn't keep historical records, using **SCD Type 1**.