Q1) You are hired as a data engineer for ShopSmart, a national retail chain that operates 100+ stores and an online e-commerce platform. ShopSmart wants to build a central analytics warehouse to analyze sales performance, customer behavior, and inventory trends across multiple channels.

1 Identify Fact and Dimension Tables

Fact Tables

Fact Table	Measures / Metrics
`Sales_Fact`	`sales_amount`, `quantity_sold`, `discount_applied`, `profit`
`Inventory_Fact`	`stock_on_hand`, `reorder_level`, `units_sold`
`Promotion_Fact`	`promotion_discount`, `promotion_revenue`, `product_count`

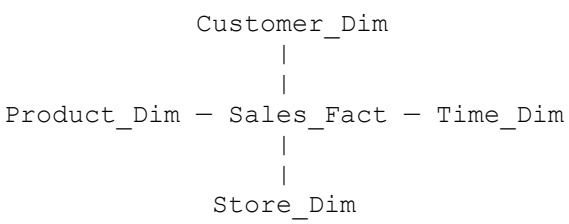
Dimension Tables

Dimension Table	Attributes
`Customer_Dim`	`customer_id`, `first_name`, `last_name`, `gender`, `birth_date`, `loyalty_level`, `city`, `state`, `region`
`Product_Dim`	`product_id`, `product_name`, `category`, `subcategory`, `brand`, `supplier_id`, `price`
`Store_Dim`	`store_id`, `store_name`, `city`, `state`, `region`, `store_manager`
`Time_Dim`	`date_key`, `date`, `day`, `week`, `month`, `quarter`, `year`, `holiday_flag`
`Promotion_Dim`	`promotion_id`, `promotion_name`, `start_date`, `end_date`, `discount_percent`

---

2 Star Schema Design

Star Schema: Fact tables in the center, directly connected to dimension tables.



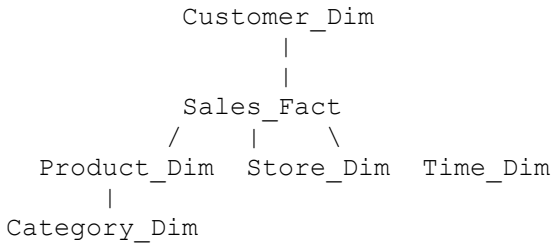
- **Fact table:** `Sales_Fact`
- **Dimensions:** `Customer_Dim`, `Product_Dim`, `Store_Dim`, `Time_Dim`
- Advantages:
  - Simple structure for **fast querying**
  - Denormalized dimensions reduce joins

- Easy to write **analytical queries**

---

3 Snowflake Schema Design

Snowflake Schema: Dimension tables normalized into sub-dimensions.



- `Product_Dim` normalized to `Category_Dim`
- `Store_Dim` could normalize city/state/region
- Advantages:
  - Reduces **data redundancy**
  - Smaller storage footprint
- Disadvantages:
  - More joins → **slower query performance**

---

4 Slowly Changing Dimensions (SCD)

- **Customer_Dim:** Address or loyalty level may change → Type 2 SCD
- **Product_Dim:** Product category updates → Type 2 SCD
- This ensures **historical analytics** are accurate.

---

5 Example Queries

Total Sales per Region per Month

```
SELECT
    t.year,
    t.month,
    s.region,
    SUM(f.sales_amount) AS total_sales
FROM Sales_Fact f
JOIN Time_Dim t ON f.time_key = t.date_key
JOIN Store_Dim s ON f.store_key = s.store_id
GROUP BY t.year, t.month, s.region
ORDER BY t.year, t.month, s.region;
```

Top 5 Products by Revenue

```
SELECT
    p.product_name,
    SUM(f.sales_amount) AS revenue
FROM Sales_Fact f
JOIN Product_Dim p ON f.product_key = p.product_id
GROUP BY p.product_name
ORDER BY revenue DESC
LIMIT 5;
```

Customer Retention Analysis

```
SELECT
    c.loyalty_level,
    COUNT(DISTINCT f.customer_key) AS active_customers
FROM Sales_Fact f
JOIN Customer_Dim c ON f.customer_key = c.customer_id
WHERE f.time_key BETWEEN '2025-01-01' AND '2025-12-31'
GROUP BY c.loyalty_level;
```

---

6 Justification of Schema Choice

Schema	Use Case	Pros	Cons
Star	Fast reporting, dashboards	Simple queries, good for BI	Slight data redundancy
Snowflake	Normalized data warehouse	Saves storage, reduces redundancy	Complex queries, slower joins

Q2 You are a data engineer for QuickEats, an online food delivery platform operating in multiple cities. QuickEats collects and processes data from multiple sources. Currently, the system struggles with scalability, real-time processing, and analytics performance. Suggest a suitable model.

1️⃣ Identify Fact and Dimension Tables

✅ Fact Tables

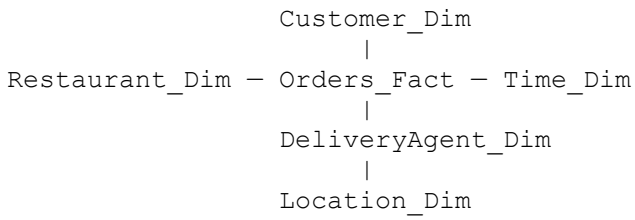
Fact Table	Measures / Metrics
Orders_Fact:	order_amount, delivery_fee, discount_amount, total_payment delivery_time, order_status
Delivery_Fact:	delivery_time, distance_traveled, pickup_time, drop_time, delivery_rating
Payment_Fact:	payment_amount, tax_amount, commission, payment_status
App_Events_Fact:	click_count, session_duration, device_type, action_type

---

✅ Dimension Tables

Dimension Table	Attributes
Customer_Dim:	customer_id, name, phone, email, signup_date, city, loyalty_level
Restaurant_Dim:	restaurant_id, restaurant_name, cuisine_type, city, rating, partner_since
DeliveryAgent_Dim:	agent_id, agent_name, vehicle_type, city, experience_level
Time_Dim:	time_id, date, day, week, month, quarter, year
Location_Dim:	location_id, city, state, region
PaymentMethod_Dim:	payment_method_id, method_name, provider

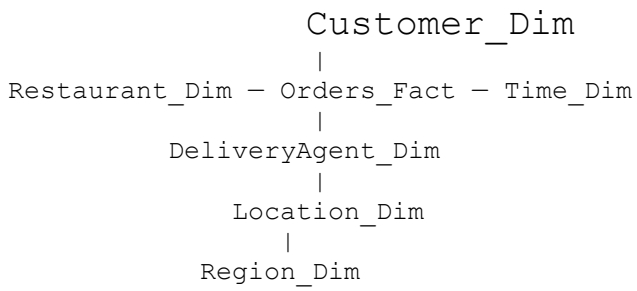
2️⃣ Star Schema Design (Recommended for BI Dashboards)



✅ Advantages:

- Fast performance for reporting.
- Simple joins.
- Best for dashboards (Power BI, Tableau, Looker).

3️⃣ Snowflake Schema Design (Normalized)



✅ Features:

- Normalized dimensions like **Location → Region → Country**.
- Reduces duplicate data.

4️⃣ Slowly Changing Dimensions (SCD)

Dimension	Attribute Change Example	SCD Type
Customer_Dim	Loyalty level upgrade from Silver → Gold	Type 2
Restaurant_Dim	Updated rating or menu expansion	Type 2
DeliveryAgent_Dim	Vehicle type change	Type 2
Location_Dim	City name correction	Type 1

5️⃣ Example Queries

✅ Total Revenue by City

```
SELECT l.city, SUM(o.total_payment) AS revenue
FROM Orders_Fact o
JOIN Location_Dim l ON o.location_key = l.location_id
GROUP BY l.city
ORDER BY revenue DESC;
```

✅ Top 5 Restaurants by Orders

```
SELECT r.restaurant_name, COUNT(o.order_id) AS total_orders
FROM Orders_Fact o
JOIN Restaurant_Dim r ON o.restaurant_key = r.restaurant_id
GROUP BY r.restaurant_name
ORDER BY total_orders DESC
LIMIT 5;
```

✅ Average Delivery Time by Agent

```
SELECT d.agent_name, AVG(f.delivery_time) AS avg_delivery_time
FROM Delivery_Fact f
JOIN DeliveryAgent_Dim d ON f.agent_key = d.agent_id
GROUP BY d.agent_name
ORDER BY avg_delivery_time;
```

6️⃣ Suggested Modern Architecture for QuickEats

Layer	Technology
Data Ingestion	Kafka / AWS Kinesis (Real-time), Airbyte/Fivetran (Batch)
Data Storage	Data Lake (S3/Google Cloud), Warehouse (Snowflake/BigQuery)
Processing	Apache Spark, Flink (Real-time stream processing)
ETL/ELT	dbt + Airflow
BI Dashboard	Power BI, Tableau, Looker
Orchestration	Apache Airflow

7️⃣ Justification of Data Model Choice

Schema	Use Case	Advantages	Disadvantages
Star	Sales dashboards, business reporting	Fast queries, easy to maintain	Some redundancy
Snowflake	Large dimensional data	Saves space, normalized	More joins
Data Lakehouse (Recommended)	Real-time + Analytics	Supports batch + streaming, scalable	Slightly complex setup

Q3) You are a data engineer for StreamFlix, a global video streaming platform (like Netflix). StreamFlix collects millions of events per day. The company wants to build a high-performance analytics warehouse to support:

- Real-time viewer engagement analytics
- Top trending videos per region
- AI models for recommendation engines

1️⃣ Identify Fact and Dimension Tables

✅ Fact Tables

Fact Table	Measures / Metrics
Viewership_Fact:	watch_duration, progress_percent, buffering_time, resolution_played, watch_status
StreamingSession_Fact:	session_duration, device_time_spent, data_consumed_mb
Subscription_Fact:	subscription_amount, discount, renewal_status
Search_Fact:	total_searches, click_throughs, search_time
Recommendation_Fact:	recommendation_clicks, recommendation_impressions

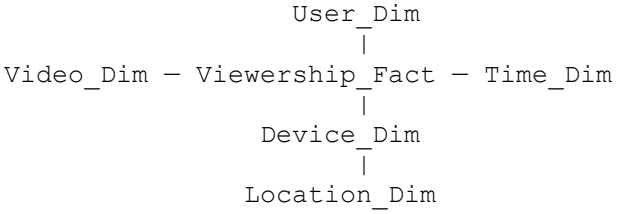
---

✅ Dimension Tables

Dimension Table	Attributes
User_Dim:	user_id, name, gender, age_group, country, subscription_type, join_date
Video_Dim:	video_id, title, genre, sub_genre, language, release_year, maturity_rating
Device_Dim:	device_id, device_type, os, app_version
Time_Dim:	time_id, date, hour, day, week, month, quarter, year
Location_Dim:	location_id, country, region, city
SubscriptionPlan_Dim:	plan_id, plan_name, price, resolution_limit, screens_allowed

---

2️⃣ Star Schema Design (For Fast BI Reporting)

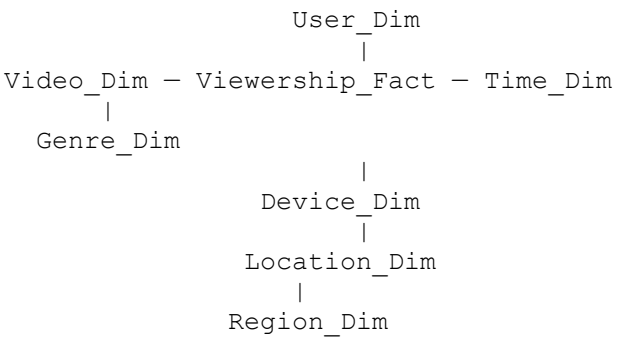


✅ **Advantages**

- Optimized for query performance
- Simple structure for dashboards
- Ideal for daily analytics

---

3 Snowflake Schema Design (Normalized Model)



✅ **Advantages:** Reduces redundancy
❌ More joins → Slight slower performance

4 Handling Slowly Changing Dimensions (SCD)

Dimension	Change Example	SCD Type
User_Dim	Subscription changes from Basic → Premium	Type 2
Video_Dim	Video updated from SD → HD version	Type 2
Device_Dim	App version updates	Type 1
SubscriptionPlan_Dim	Plan pricing updates	Type 2

---

5 Example Analytical Queries

✅ Top Trending Videos Per Region

```
SELECT l.region, v.title, COUNT(f.video_id) AS total_views
FROM Viewership_Fact f
JOIN Video_Dim v ON f.video_key = v.video_id
JOIN Location_Dim l ON f.location_key = l.location_id
GROUP BY l.region, v.title
ORDER BY total_views DESC
LIMIT 10;
```

✅ Average Watch Time by Subscription Type

```
SELECT u.subscription_type, AVG(f.watch_duration) AS avg_watch_time
FROM Viewership_Fact f
JOIN User_Dim u ON f.user_key = u.user_id
GROUP BY u.subscription_type;
```

✅ Device Usage Analysis

```
SELECT d.device_type, COUNT(f.session_id) AS sessions
FROM StreamingSession_Fact f
JOIN Device_Dim d ON f.device_key = d.device_id
GROUP BY d.device_type;
```

6️⃣ Recommended Modern Architecture (Streaming + Batch)

Layer	Technology
Event Ingestion	Kafka / AWS Kinesis / Pub/Sub
Real-Time Processing	Apache Spark Streaming / Flink
Batch ETL	Airflow + dbt
Storage	Data Lake (S3/GCS) + Delta Lake
Warehouse	Snowflake / BigQuery / Redshift
Feature Store	Feast (for ML Recommendations)
BI Layer	Tableau / Power BI / Looker

7️⃣ Justification of Schema Choice

Model	Use Case	Pros	Cons
Star Schema	Viewer reports, dashboards	Fast query, simple	Some redundancy
Snowflake	Large metadata handling	Efficient storage	Complex joins
Lakehouse (✅ Recommended)	Real-time + ML use cases	Supports streaming + batch + AI	Setup complexity