

## **Data Modeling**

### **SQL:**

#### **1) Flat Main Table (Railways)**

- Transaction\_ID
- Date\_of\_Purchase
- Time\_of\_Purchase
- Departure\_Station
- Arrival\_Destination
- Date\_of\_Journey
- Departure\_Time,
- Arrival\_Time
- Actual\_Arrival\_Time
- Journey\_Status
- Reason\_for\_Delay, Refund\_Request
- Purchase\_Type
- Payment\_Method
- Railcard
- Ticket\_Class
- Ticket\_Type,
- Price

## 2) Cleaning

We performed multiple cleaning steps:

- **Checked for duplicates:** verified Transaction\_ID uniqueness.
- **Checked for NULLs:** especially in Transaction\_ID, Actual\_Arrival\_Time, and Reason\_for\_Delay.
- **Decided NULL handling:** left NULLs in columns like Reason\_for\_Delay to preserve meaning (NULL = No Delay).
- **Standardized data types:**
  - Converted date/time columns → DATE, TIME.
- **Added constraints:** prepared for primary key (Transaction\_ID).
- **Price Validation:** price data checked that it's above zero and valid

### **3) Dimensions and Facts**

#### **a) TicketInfo (Fact-like)**

- Transaction\_ID (PK)
  - Date\_of\_Purchase
  - Time\_of\_Purchase
  - Purchase\_Type
  - Payment\_Method
  - Railcard
  - Ticket\_Class
  - Ticket\_Type
  - Price
- 

#### **b) RouteInfo (Dimension)**

- Route\_ID (PK, surrogate with IDENTITY)
  - Departure\_Station
  - Arrival\_Destination
  - Added UNIQUE constraint that each pair exists only once (64 Routes).
-

### **c) Journey (Dimension)**

- Journey\_ID (PK, surrogate with IDENTITY)
  - Transaction\_ID (FK from TicketInfo)
  - Departure\_Station
  - Arrival\_Destination
  - Date\_of\_Journey
  - Departure\_Time
  - Arrival\_Time
  - Actual\_Arrival\_Time
  - Journey\_Status
- 

### **d) Delay (Dimension Table)**

- Delay\_ID (PK, surrogate with IDENTITY)
  - Transaction\_ID (FK from TicketInfo)
  - Journey\_ID (FK from Journey)
  - Route\_ID (FK from RouteInfo)
  - Journey\_Status
  - Reason\_for\_Delay
  - Refund\_Request
-

#### **d) Date (Dimension Table)**

- DateKey (PK)
- FullDate
- Day
- Month
- Year

---

#### **Python:**

We used python to ensure the validation of the cleaning done by the SQL

##### **A) Duplicate Check:**

Ensured no repeated Transaction IDs.

Guarantee: each transaction is unique.

##### **B) Null Check:**

Checked critical fields (ID, Dates, Times, Price, Status).

Guarantee: no missing data in essential columns.

##### **C) Primary Key Check:**

Verified Transaction ID is unique and not null.

Guarantee: dataset has a valid primary key for joins/analysis.

##### **D) Price Check:**

Confirmed Price is present and non-negative.

Guarantee: financial values are reliable.

