Diagram

Description automatically generated

# Creating a star schema

We begin with a **normalized database schema**, where information is stored across many related tables to avoid duplication. In our case, we had a structure where: Sales data is recorded in the SALES table. Information about customers, employees, stores, products, and locations is stored across several other tables like CUSTOMERS, EMPLOYEES, PRODUCTS, CATEGORIES, ADDRESSES, CITIES, COUNTRIES, etc.

To convert this schema to a star schema I followed these steps:

**Step 1: Picked the Fact Table**

The SALES table is chosen as the main fact table because it records key events — each sale, its amount, and when/where it happened.

**Step 2: Chose Dimension Tables**

I selected the following tables to describe the sales: CUSTOMERS, PRODUCTS, EMPLOYEES, STORES, DATE .

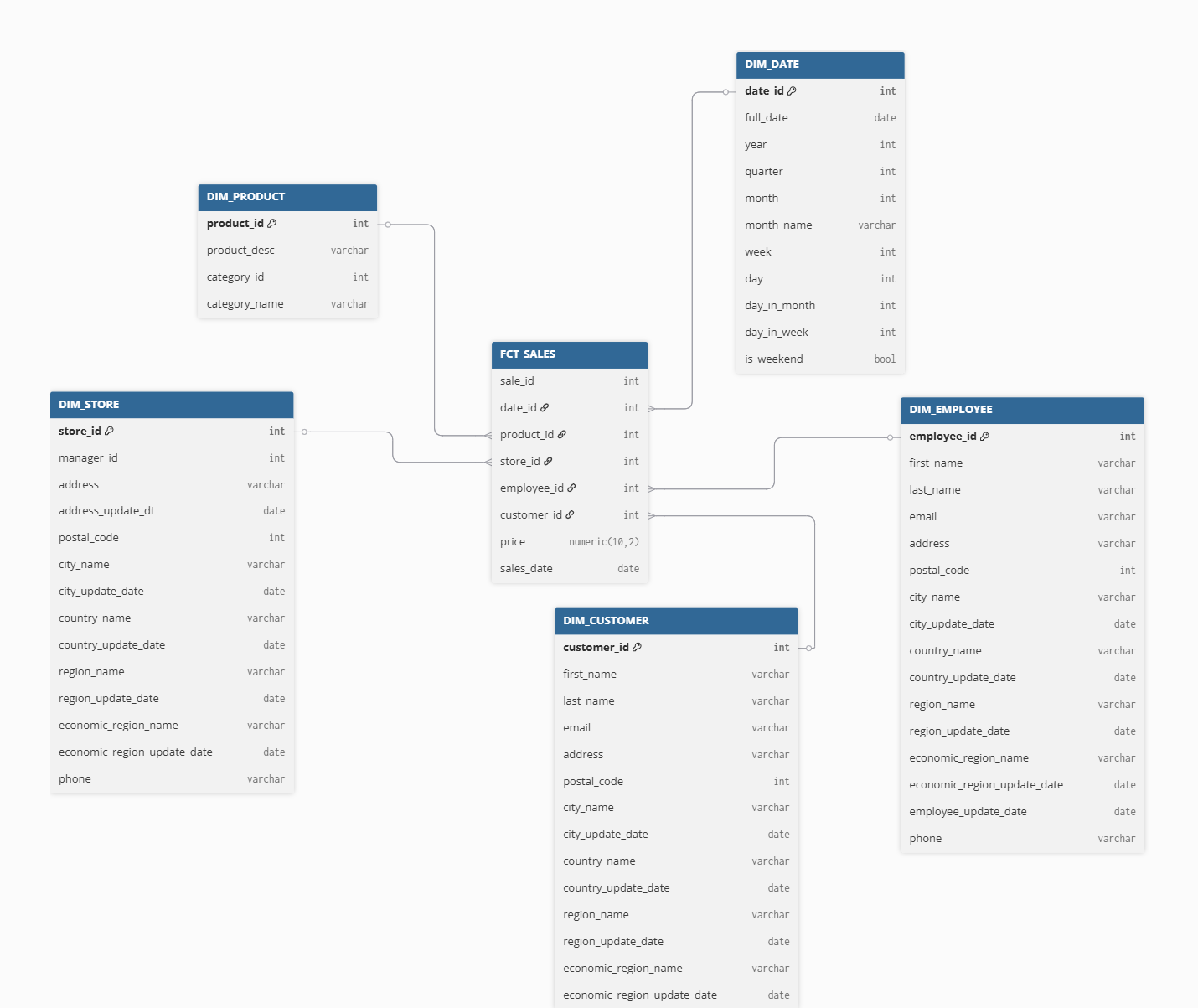
**Step 3: Combined Related Information**

To make reporting easier, I **combined** related tables into single, flat tables. For example:

1. In the CUSTOMERS table, we added details from ADDRESSES, CITIES, COUNTRIES, and REGIONS.
2. In the PRODUCTS table, we added the product category name directly.

**Step 4: Connected All to the Fact Table**

We linked all the dimension tables directly to the SALES table. The result looks like a star, with SALES in the center and dimensions around it — this is why it's called a **star schema.**

Result:

Transforming the star schema to a snowflake:

The star schema is simple and good for performance, but it can lead to data repetition. To avoid this, we can convert it into a **snowflake schema** by breaking large dimension tables into smaller related ones again. For the result I did the following:

**Step 1: Identified redundant attribute groups and hierarchies**

From the denormalized Star Schema, we spot repeating groups and hierarchies:

**Product Category** appears in every product row

**Address, City, Country, Region, Economic Region** repeat across DIM\_CUSTOMER, DIM\_EMPLOYEE, DIM\_STORE

**Price** is duplicated per transaction but may change over time (needs SCD)

**Step 2: Normalized product dimension**

I broke out **Product Category** into a separate dimension:

DIM\_PRODUCT now references DIM\_CATEGORY via category\_id.

**Step 3: Normalize price using type 2 SCD**

I move price from FCT\_SALES into a separate SCD-managed dimension:

Added start\_date, end\_date, is\_active for version tracking.

FCT\_SALES now references price\_id, not product\_id.

**Step 4: Normalized address and geography hierarchy**

Break down full geographic info into multiple normalized tables:

| **From Star Dimensions** | **To Snowflake Hierarchy** |
| --- | --- |
| address, postal\_code | DIM\_ADDRESS |
| city\_name | DIM\_CITY |
| country\_name | DIM\_COUNTRY |
| region\_name | DIM\_REGION |
| economic\_region\_name | DIM\_ECONOMIC\_REGION |

DIM\_ADDRESS references DIM\_CITY

DIM\_CITY → DIM\_COUNTRY → DIM\_REGION → DIM\_ECONOMIC\_REGION

DIM\_CUSTOMER, DIM\_EMPLOYEE, and DIM\_STORE reference DIM\_ADDRESS

**Step 5: Updated fact table**

I replaced foreign keys to flattened dimensions with links to the lowest-level normalized versions:

Result:

