Report on Building a Database for Super Store Sales Data

GROUP 2

Karan Bhosale

Yingzi Yuan

Priyanka Chahande

Yiming Sun

ISQS 6338 Database Concepts

Introduction

The purpose of this report is to document the process of designing and implementing a relational database for managing the sales data of a Super Store. The primary objective of this endeavor was to construct a robust and efficient database system capable of handling and analyzing vast quantities of sales-related information.

Objective

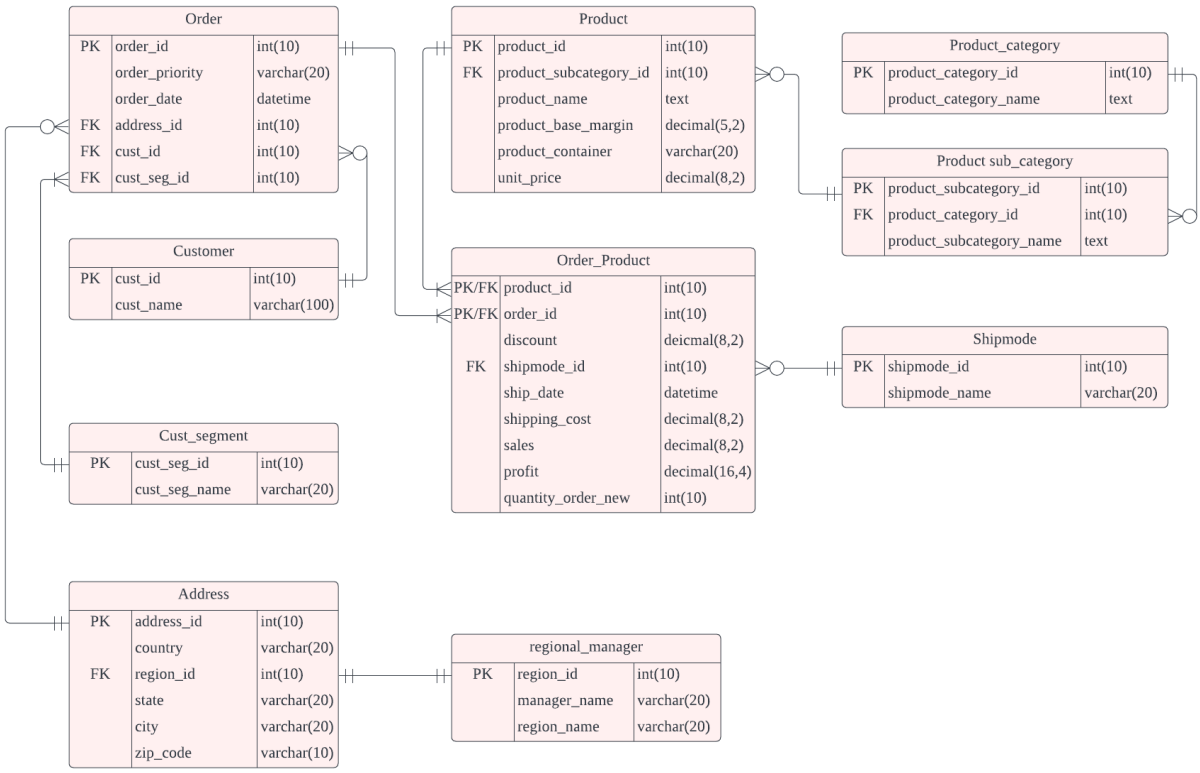
The main objective of this project was to design and build an Entity-Relationship (ER) model and subsequently implement a relational database to efficiently store and manage the sales data of the Super Store. The database was envisioned to offer seamless data retrieval, support complex queries, and enable data analysis.

Methodology

1. **Data Selection:**
   We found a Super Store sales dataset. We chose 150 rows of data as a dataset and stored them in a CSV file.

2. **Entity-Relationship (ER) Model:**
   The next phase entailed constructing an ER model to represent the entities, attributes, and relationships within the Super Store's sales domain. The ER model served as the blueprint for designing the database schema.

   Here is our ER model.



3. **Database Design:**

Based on the ER model, we proceeded to design the relational database schema. The schema comprised a set of normalized tables, each representing a specific entity, and incorporated appropriate relationships and constraints to ensure data integrity.

4. **Database Creation:**

With the schema in place, we created 10 tables.

Here are the SQL query statements of table creation.

```
CREATE TABLE 201_G02.regional_manager(
  region_id INT(10) NOT NULL AUTO_INCREMENT,
   manager_name VARCHAR(20),
   region_name VARCHAR(20),
   PRIMARY KEY(region_id)
   );
```

```
CREATE TABLE 201_G02.cust_segment(
   cust_seg_id INT(10) NOT NULL AUTO_INCREMENT,
   cust_seg_name VARCHAR(20),
   PRIMARY KEY(cust_seg_id)
   );

CREATE TABLE 201_G02.customer(
   cust_id INT(10) NOT NULL,
   cust_name VARCHAR(100),
   PRIMARY KEY(cust_id)
         );

CREATE TABLE 201_G02.shipmode(
   shipmode_id INT(10) NOT NULL AUTO_INCREMENT,
   shipmode_name VARCHAR(20),
   PRIMARY KEY(shipmode_id)
   );

CREATE TABLE 201_G02.product_category(
   product_category_id INT(10) NOT NULL AUTO_INCREMENT,
   product_category_name TEXT,
   PRIMARY KEY(product_category_id)
   );

CREATE TABLE 201_G02.product_subcategory(
   product_subcategory_id INT(10) NOT NULL AUTO_INCREMENT,
   product_category_id INT(10) NOT NULL,
   product_subcategory_name TEXT,
   PRIMARY KEY(product_subcategory_id),
   CONSTRAINT product_category_fk FOREIGN KEY(product_category_id)
                  REFERENCES 201_G02.product_category(product_category_id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
         );
```

```
CREATE TABLE 201_G02.address(
   address_id INT(10) NOT NULL AUTO_INCREMENT,
   country VARCHAR(20),
   region_id INT(10) NOT NULL,
   state VARCHAR(20),
   city VARCHAR(20),
   zip_code VARCHAR(10),
   PRIMARY KEY(address_id),
   CONSTRAINT regional_manager_fk FOREIGN KEY(region_id)
                  REFERENCES 201_G02.regional_manager(region_id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
         );

CREATE TABLE 201_G02.product(
   product_id INT(10) NOT NULL AUTO_INCREMENT,
   product_subcategory_id INT(10) NOT NULL,
   product_name TEXT,
   product_base_margin DECIMAL(5,2),
   product_container VARCHAR(20),
   unit_price DECIMAL(8,2),
   PRIMARY KEY(product_id),
   CONSTRAINT product_subcategory_fk FOREIGN KEY(product_subcategory_id)
                  REFERENCES
201_G02.product_subcategory(product_subcategory_id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
         );


CREATE TABLE 201_G02.order(
         order_id INT(10) NOT NULL,
         order_priority VARCHAR(20),
         order_date datetime,
         address_id INT(10) NOT NULL,
         cust_id INT(10) NOT NULL,
         cust_seg_id INT(10) NOT NULL,
```

```
    PRIMARY KEY(order_id),
    CONSTRAINT address_fk FOREIGN KEY(address_id)
        REFERENCES 201_G02.address(address_id)
        ON DELETE NO ACTION ON UPDATE NO ACTION,
    CONSTRAINT customer_fk FOREIGN KEY(cust_id)
        REFERENCES 201_G02.customer(cust_id)
  ON DELETE NO ACTION ON UPDATE NO ACTION,
    CONSTRAINT cust_segment_fk FOREIGN KEY(cust_seg_id)
        REFERENCES 201_G02.cust_segment(cust_seg_id)
  ON DELETE NO ACTION ON UPDATE NO ACTION
    );


CREATE TABLE 201_G02.order_product(
  product_id INT(10) NOT NULL,
  order_id INT(10) NOT NULL,
  discount DECIMAL(8,2),
  shipmode_id INT(10) NOT NULL,
  ship_date DATETIME,
  shipping_cost DECIMAL(8,2),
  sales DECIMAL(8,2),
  profit DECIMAL(16,4),
  quantity_order_new INT(10),
  PRIMARY KEY(product_id, order_id),
  CONSTRAINT product_fk FOREIGN KEY(product_id)
        REFERENCES 201_G02.product(product_id)
    ON DELETE NO ACTION ON UPDATE NO ACTION,
    CONSTRAINT order_fk FOREIGN KEY(order_id)
        REFERENCES 201_G02.order(order_id)
    ON DELETE NO ACTION ON UPDATE NO ACTION,
    CONSTRAINT shipmode_fk FOREIGN KEY(shipmode_id)
        REFERENCES 201_G02.shipmode(shipmode_id)
    ON DELETE NO ACTION ON UPDATE NO ACTION
    );
```

5. **Data Insertion:**

To populate the database, we wrote custom scripts to import the sales data from the CSV files into the respective tables.

Here are the SQL query statements of data insertion.

```
insert into 201_G02.regional_manager (region_name,manager_name)
values ('Central','Chris'),
        ('East','Erin'),
        ('South','Sam'),
        ('West','William')
;

insert into 201_G02.cust_segment (cust_seg_name)
select distinct `Customer Segment`
from 201_G02.superstore
;

insert into 201_G02.shipmode(shipmode_name)
select distinct `Ship Mode`
from 201_G02.superstore
;

insert into 201_G02.product_category(product_category_name)
select distinct `Product Category`
from 201_G02.superstore
;

insert into 201_G02.product_subcategory(product_category_id,
product_subcategory_name)
select distinct pc.product_category_id, s.`Product Sub-Category`
from 201_G02.superstore s, 201_G02.product_category pc
where s.`Product Category` = pc.product_category_name
;
```

```
insert into 201_G02.address (country,region_id,state,city,zip_code)
select distinct s.Country, r.region_id, s.`State or Province` as state, s.City, s.`Postal
Code` as zip_code
from 201_G02.superstore s, 201_G02.regional_manager r
where s.region = r.region_name
;

insert into 201_G02.customer(cust_id, cust_name)
select distinct `Customer ID`, `Customer Name`
from 201_G02.superstore s
;

insert into 201_G02.order(order_id, order_priority, order_date, address_id,
cust_id, cust_seg_id)
select distinct s.`Order ID`, s.`Order Priority`, str_to_date(s.`Order
Date`,'%m/%d/%y')  `Order Date`, a.address_id, s.`Customer ID`, cs.cust_seg_id
from 201_G02.superstore s,  201_G02.address a , 201_G02.regional_manager r,
201_G02.cust_segment cs
where s.Country=a.country
and s.region=r.region_name
and s.`State Or Province`=a.state
and s.city=a.city
and s.`Postal Code`=a.zip_code
and a.region_id =r.region_id
and s.`Customer Segment` =cs.cust_seg_name
;

insert into 201_G02.product(product_subcategory_id, product_name,
product_base_margin, product_container, unit_price)
select distinct ps.product_subcategory_id, s.`Product Name`, s.`Product Base
Margin`, s.`Product Container`, s.`Unit Price`
from 201_G02.superstore s, 201_G02.product_subcategory ps
where s.`Product Sub-Category` = ps.product_subcategory_name
;
```

```
insert into 201_G02.order_product(product_id, order_id, discount, shipmode_id,
ship_date, shipping_cost, sales, profit, quantity_order_new)
select distinct p.product_id, s.`Order ID`, s.`Discount`, sm.shipmode_id,
str_to_date(s.`Ship Date`,'%m/%d/%y')  `Ship Date`, s.`Shipping Cost`, s.`Sales`,
s.`Profit`, s.`Quantity ordered new`
from 201_G02.superstore s, 201_G02.product p, 201_G02.shipmode sm
where s.`Product Name` = p.product_name
and s.`Ship Mode` =sm.shipmode_name
;
```

Challenges and Solutions

Challenge 1:

At the beginning, we try to find health care dataset, but many cells are empty, anonymized, or incomplete. What's more, different countries and hospitals use varying medical terminologies and metrics, leading to data redundancy. And we have no idea about some medical terminologies, so it is hard to figure out the relationship between different attributes.

Challenge 2:

While inserting data for regional_manager table from dataset table, we come to know about region_name which was not there in the dataset table as well as in ER model.
So, we use 'insert' into values instead of select statement as we just have 4 values to insert.
It was easy to insert.

Challenge 3:

At first, we forgot to mention auto-increment to primary key, so we need to drop all the tables and create tables again.
But we already had order_id in our database so again we need to remove auto increment from orders table.

Challenge 4:

When we inserted the data, we found every customer may have more than one customer segment, it is different from the logic we build the ER model. We assumed that

each customer has one segment, but, in reality, a customer can have multiple segments, and these segments are related to specific orders.
So, we changed our ER model. We move cust_seg_id from customer entity to order entity, and link the customer_segment entity to order entity.

Challenge 5:

During table construction, we made assumptions about data types. However, when inserting data, we encountered space constraints that caused data truncation. As a result, we had to modify the data types.
We changed cust_name varchar(100) due to space constraints.