SQL Database for a Vintage Thrift Store

Introduction:
In today's era of sustainability and conscious consumption, thrift stores play a pivotal role in providing affordable and environmentally friendly shopping options. Today, in the United States, there are more than 28,000 thrift stores. However, managing the diverse inventory and streamlining operations within thrift stores can be challenging without a robust database management system (DBMS). Our thrift store DBMS aims to address these challenges by providing a centralized platform for efficient inventory management, sales processing, and customer relationship management.

Advantages:

- Centralized inventory management: The DBSM will allow thrift stores to efficiently track donated items, categorize them based on type, brand, condition, and price, and update inventory in real-time.
- Streamlines Sales Process: Integration of point-of-sale systems which enables seamless transactions, including but not limited to, discounts, promotions, and payment options.
- Customer Relationship Management: Recording customers' purchase histories enables personalized interactions and targeted marketing campaigns to foster customer loyalty and drive sales.
- Reporting and Analytics: Can help generate valuable insights into sales trends, popular items and operational efficiency. This is vital for store managers to make data-driven decisions, which can optimize the business.

Use Cases:

- Retail Industry:
    - Retail chains
    - Department stores
    - Boutiques
- E-commerce:
    - Amazon
    - Shopify
    - Alibaba
    - Etsy
- Non-profit organizations:
    - Salvation Army
    - Goodwill
    - AMVETs

Business Rules

Inventory management rules

- Inventory includes clothing, accessories, antiques, and record players
- Unique productID, ProductName, Price
- Unique inventoryID
- Categorize based on: vintage t-shirt, retro dresses, vinyl records, antique watches, vintage sunglasses, and vinyl player
- Each unique product has unique names and prices depending on how valuable they are such as the watch

Sales and transactions

- Are categorized by customer ID, the products unique ID, and the quantity bought of the product
- Other unique identifiers for the transaction include the dates it was sold and price

Customer rules

- Customers information is categorized by first and last name such as in the database referred to as Alice Johnson and Bob Smith
- Their address and unique CustomerID is identified too

Employee Rules

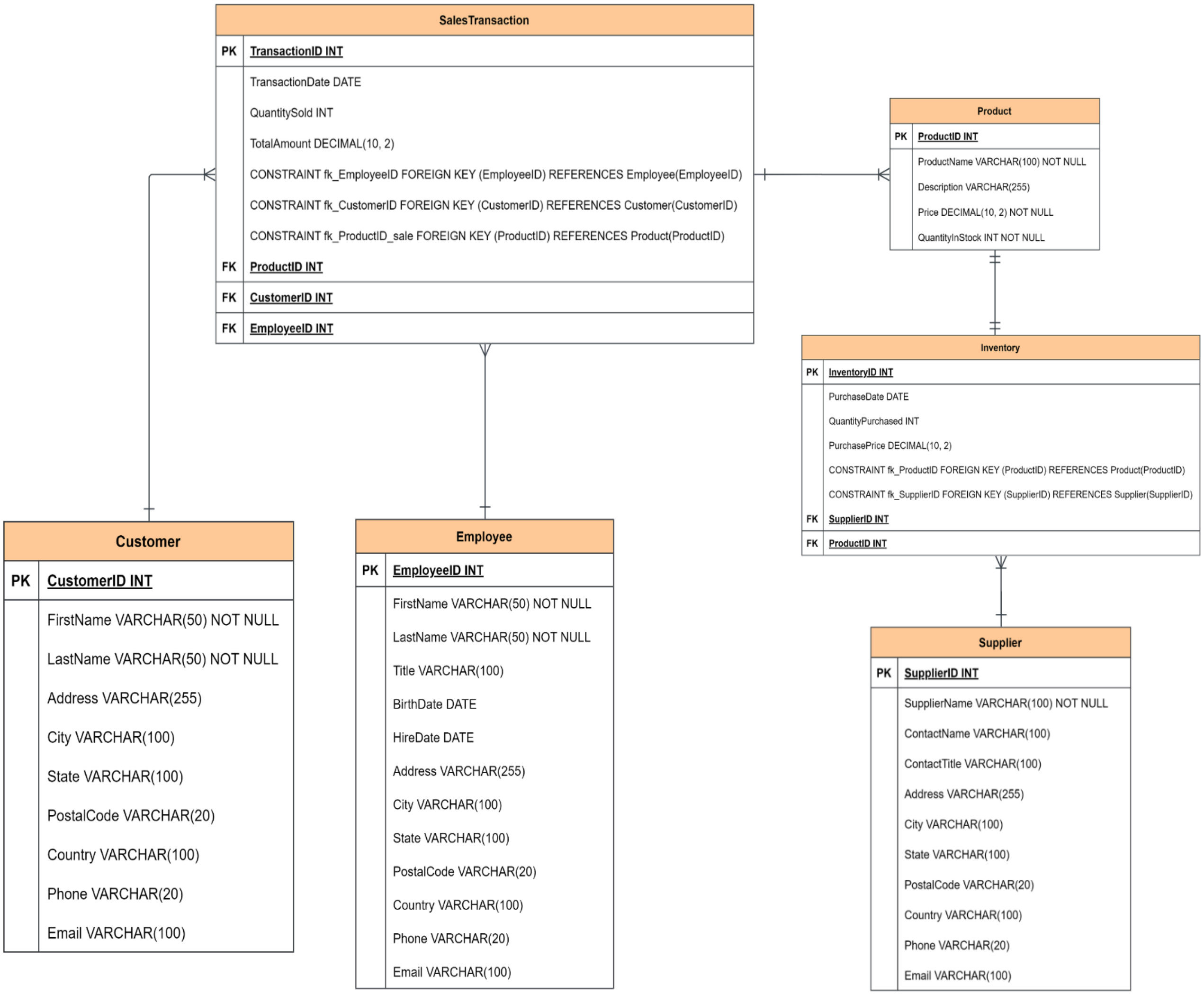
- Each employee must have a unique ID and their information is provided
- Employee such as Michael Smith referred to as the manager and Emily Davis as a sales associate

These business rules allows us to properly identify what product is being sold, which category it lies in, the valuableness, what customer bought it and how many, and the employees taking part of the business operation

Conceptual Data Modeling and Database Design:

- **Product**:
    - Attributes: Product ID (Primary Key), Name, Description, Price, Quantity in Stock
    - Relationships:
        - One-to-One relationship with Inventory (each product can have only one inventory records)
        - One or Many-to-One relationship with Sales Transaction (One or multiple products can be part of one sales transactions)
- **Supplier**:
    - Attributes: Supplier ID (Primary Key), Name, Contact Information
    - Relationships:
        - One-to-One or Many relationship with Inventory (each Supplier supply one or multiple unique products)
- **Inventory**:
    - Attributes: Product ID (Primary Key), Quantity in Stock, Supplier ID (Foreign Key)
    - Relationships:
        - One-to-One relationship with Product (each inventory record corresponds only to one product)
        - One or Many-to-One relationship with Supplier (One or multiple inventory records are associated with one Supplier)
- **Customer**:
    - Attributes: Customer ID (Primary Key), Name, Contact Information, Purchase History
    - Relationships:
        - One-to-One or Many relationship with Sales Transaction (each customer can have one or multiple sales transactions)
- **Sales Transaction**:
    - Attributes: Transaction ID (Primary Key), Date and Time of Sale, Customer ID (Foreign Key), Total Amount, Payment Method
    - Relationships:
        - One or Many-to-One relationship with Customer (One or Many transactions are associated with one customer)
        - One-to-One or Many relationship with Product (each transaction involves one or multiple products)
        - Many-to-One relationship with Employee (Multiple transactions are completed by one Employee)
- **Employee**:

- Attributes: Employee ID (Primary Key), Name, Position, Contact Information, Work Schedule
- Relationships:
- One-to-Many relationship with Sales Transaction (each Employee can complete multiple transactions)



	SalesTransaction
PK	**TransactionID INT**
	TransactionDate DATE
	QuantitySold INT
	TotalAmount DECIMAL(10, 2)
	CONSTRAINT fk_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
	CONSTRAINT fk_CustomerID FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
	CONSTRAINT fk_ProductID_sale FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
FK	**ProductID INT**
FK	**CustomerID INT**
FK	**EmployeeID INT**

	Product
PK	**ProductID INT**
	ProductName VARCHAR(100) NOT NULL
	Description VARCHAR(255)
	Price DECIMAL(10, 2) NOT NULL
	QuantityInStock INT NOT NULL

	Inventory
PK	**InventoryID INT**
	PurchaseDate DATE
	QuantityPurchased INT
	PurchasePrice DECIMAL(10, 2)
	CONSTRAINT fk_ProductID FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
	CONSTRAINT fk_SupplierID FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)
FK	**SupplierID INT**
FK	**ProductID INT**

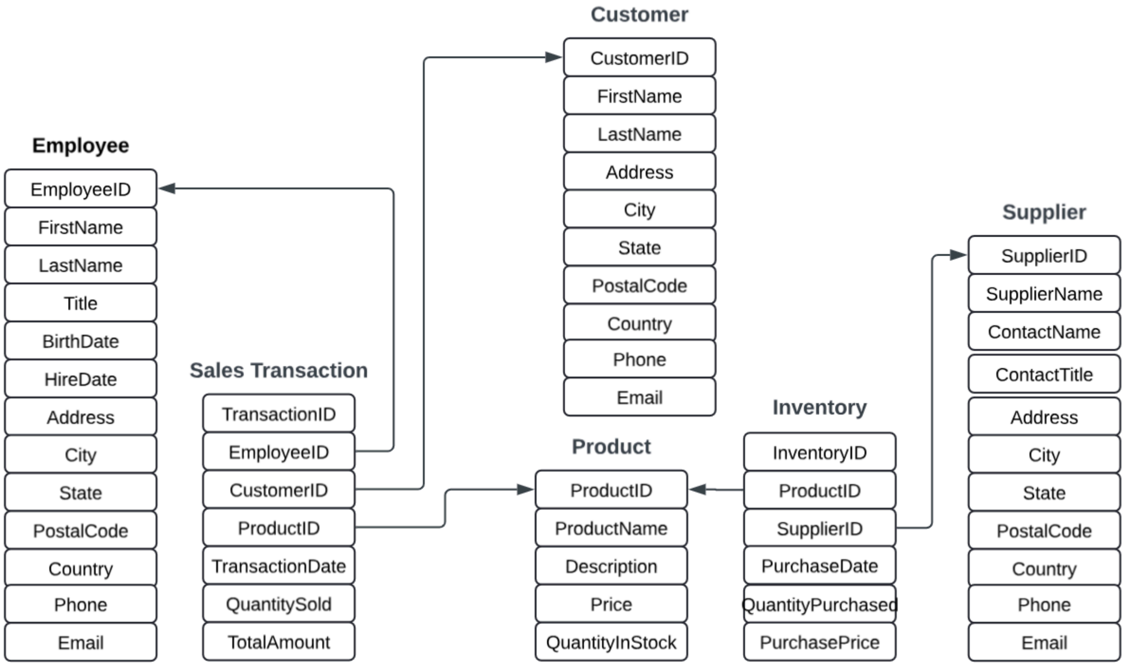
	Customer
PK	**CustomerID INT**
	FirstName VARCHAR(50) NOT NULL
	LastName VARCHAR(50) NOT NULL
	Address VARCHAR(255)
	City VARCHAR(100)
	State VARCHAR(100)
	PostalCode VARCHAR(20)
	Country VARCHAR(100)
	Phone VARCHAR(20)
	Email VARCHAR(100)

	Employee
PK	**EmployeeID INT**
	FirstName VARCHAR(50) NOT NULL
	LastName VARCHAR(50) NOT NULL
	Title VARCHAR(100)
	BirthDate DATE
	HireDate DATE
	Address VARCHAR(255)
	City VARCHAR(100)
	State VARCHAR(100)
	PostalCode VARCHAR(20)
	Country VARCHAR(100)
	Phone VARCHAR(20)
	Email VARCHAR(100)

	Supplier
PK	**SupplierID INT**
	SupplierName VARCHAR(100) NOT NULL
	ContactName VARCHAR(100)
	ContactTitle VARCHAR(100)
	Address VARCHAR(255)
	City VARCHAR(100)
	State VARCHAR(100)
	PostalCode VARCHAR(20)
	Country VARCHAR(100)
	Phone VARCHAR(20)
	Email VARCHAR(100)

Transform ER/EER Model to Relational Model:

1. **Product Table:**
    - Attributes: Product_ID (Primary Key), Name, Description, Price, Quantity_In_Stock
    - Primary Key: Product_ID
2. **Supplier Table:**
    - Attributes: Supplier_ID (Primary Key), Name, Contact_Information
    - Primary Key: Supplier_ID
3. **Inventory Table:**
    - Attributes: Product_ID (Foreign Key), Quantity_In_Stock, Reorder_Level, Supplier_ID (Foreign Key)
    - Primary Key: (Product_ID, Supplier_ID)
    - Foreign Keys:
        - Product_ID references Product(Product_ID)
        - Supplier_ID references Supplier(Supplier_ID)
4. **Customer Table:**
    - Attributes: Customer_ID (Primary Key), Name, Contact_Information, Purchase_History
    - Primary Key: Customer_ID
5. **Sales Transaction Table:**
    - Attributes: Transaction_ID (Primary Key), Date_Time, Customer_ID (Foreign Key), Total_Amount, Payment_Method, Employee_ID (Foreign Key)
    - Primary Key: Transaction_ID
    - Foreign Keys:
        - Customer_ID references Customer(Customer_ID)
        - Employee_ID references Employee(Employee_ID)
6. **Employee Table:**
    - Attributes: Employee_ID (Primary Key), Name, Position, Contact_Information, Work_Schedule
    - Primary Key: Employee_ID



In this model:

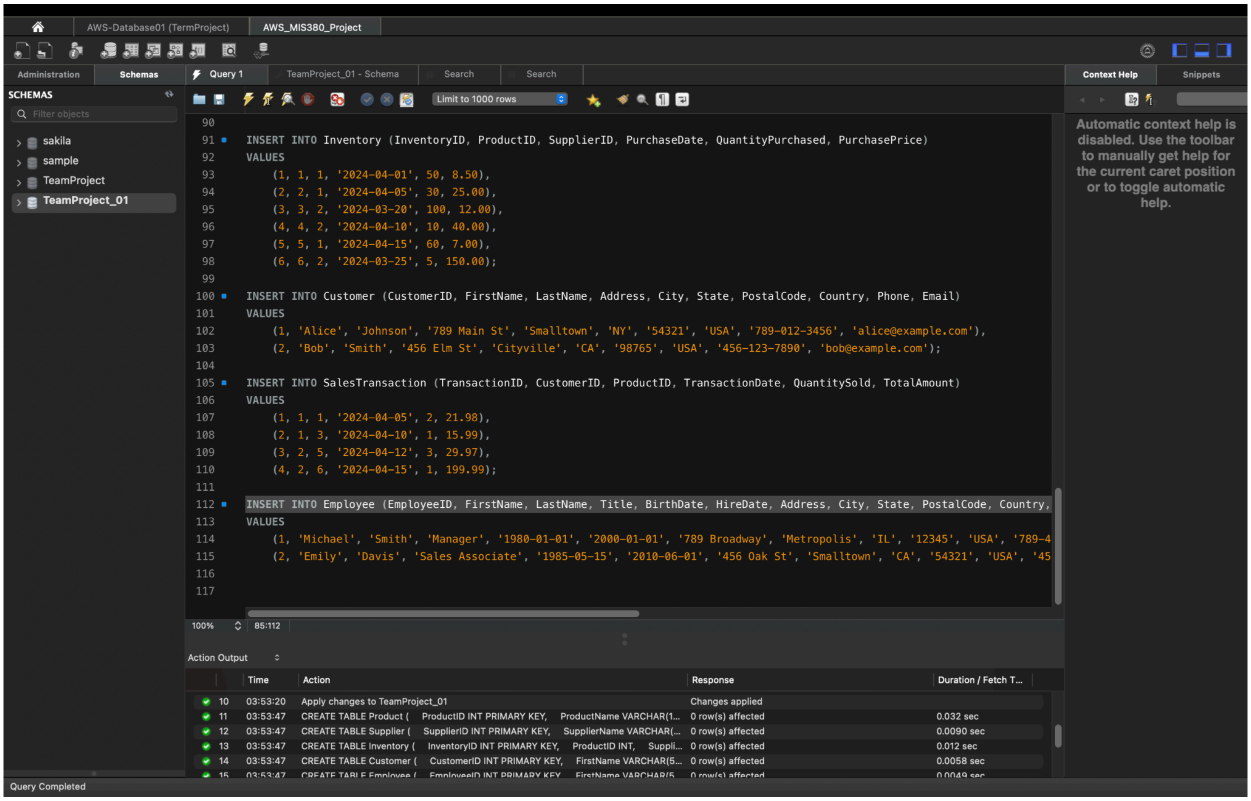
- Each table represents an entity in the scenario.
- Primary keys are indicated.
- Foreign keys establish relationships between tables, referencing primary keys in related tables.
- The Inventory table uses a composite primary key consisting of Product_ID and Supplier_ID to uniquely identify each inventory entry for a specific product from a specific supplier.
- Relationships are established between the Sales Transaction table and both the Customer and Employee tables using foreign keys.

Database Implementation

```
CREATE TABLE Product (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100) NOT NULL,
    Description VARCHAR(255),
    Price DECIMAL(10, 2) NOT NULL,
    QuantityInStock INT NOT NULL
    );

CREATE TABLE Supplier (
    SupplierID INT PRIMARY KEY,
    SupplierName VARCHAR(100) NOT NULL,
    ContactName VARCHAR(100),
    ContactTitle VARCHAR(100),
    Address VARCHAR(255),
    City VARCHAR(100),
    State VARCHAR(100),
    PostalCode VARCHAR(20),
    Country VARCHAR(100),
    Phone VARCHAR(20),
    Email VARCHAR(100)
);

CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY,
    ProductID INT,
    SupplierID INT,
    PurchaseDate DATE,
    QuantityPurchased INT,
    PurchasePrice DECIMAL(10, 2),
    CONSTRAINT fk_ProductID FOREIGN KEY (ProductID) REFERENCES Product(ProductID),
    CONSTRAINT fk_SupplierID FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)
);

CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Address VARCHAR(255),
    City VARCHAR(100),
    State VARCHAR(100),
    PostalCode VARCHAR(20),
    Country VARCHAR(100),
    Phone VARCHAR(20),
    Email VARCHAR(100)
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Title VARCHAR(100),
    BirthDate DATE,
    HireDate DATE,
    Address VARCHAR(255),
    City VARCHAR(100),
    State VARCHAR(100),
    PostalCode VARCHAR(20),
    Country VARCHAR(100),
    Phone VARCHAR(20),
    Email VARCHAR(100)
    );


CREATE TABLE SalesTransaction (
    TransactionID INT PRIMARY KEY,
    EmployeeID INT,
    CustomerID INT,
    ProductID INT,
    TransactionDate DATE,
    QuantitySold INT,
    TotalAmount DECIMAL(10, 2),
    CONSTRAINT fk_EmployeeID FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
    CONSTRAINT fk_CustomerID FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    CONSTRAINT fk_ProductID_sale FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);


INSERT INTO Product (ProductID, ProductName, Description, Price, QuantityInStock)
VALUES
    (1, 'Vintage T-shirt', 'Classic vintage t-shirt', 10.99, 100),
    (2, 'Retro Dress', 'Stylish retro dress', 29.99, 50),
    (3, 'Vinyl Record', 'Original vinyl record', 15.99, 200),
    (4, 'Antique Watch', 'Elegant antique watch', 49.99, 20),
    (5, 'Vintage Sunglasses', 'Cool vintage sunglasses', 9.99, 80),
    (6, 'Classic Vinyl Player', 'High-quality vintage vinyl player', 199.99, 10);


INSERT INTO Supplier (SupplierID, SupplierName, ContactName, ContactTitle, Address, City, State, PostalCode, Country, Phone, Email)
VALUES
    (1, 'Vintage Emporium', 'John Smith', 'Owner', '123 Vintage St', 'Vintage City', 'VA', '12345', 'USA', '123-456-7890', 'info@vintageemporium.com'),
    (2, 'Retro Finds Co.', 'Emily Johnson', 'Manager', '456 Retro Ave', 'Retro Town', 'CA', '67890', 'USA', '456-789-0123', 'sales@retrofinds.com');
```

```
INSERT INTO Inventory (InventoryID, ProductID, SupplierID, PurchaseDate, QuantityPurchased, PurchasePrice)

VALUES

    (1, 1, 1, '2024-04-01', 50, 8.50),

    (2, 2, 1, '2024-04-05', 30, 25.00),

    (3, 3, 2, '2024-03-20', 100, 12.00),

    (4, 4, 2, '2024-04-10', 10, 40.00),

    (5, 5, 1, '2024-04-15', 60, 7.00),

    (6, 6, 2, '2024-03-25', 5, 150.00);


INSERT INTO Customer (CustomerID, FirstName, LastName, Address, City, State, PostalCode, Country, Phone, Email)

VALUES

    (1, 'Alice', 'Johnson', '789 Main St', 'Smalltown', 'NY', '54321', 'USA', '789-012-3456', 'alice@example.com'),

    (2, 'Bob', 'Smith', '456 Elm St', 'Cityville', 'CA', '98765', 'USA', '456-123-7890', 'bob@example.com');


INSERT INTO SalesTransaction (TransactionID, CustomerID, ProductID, TransactionDate, QuantitySold, TotalAmount)

VALUES

    (1, 1, 1, '2024-04-05', 2, 21.98),

    (2, 1, 3, '2024-04-10', 1, 15.99),

    (3, 2, 5, '2024-04-12', 3, 29.97),

    (4, 2, 6, '2024-04-15', 1, 199.99);


INSERT INTO Employee (EmployeeID, FirstName, LastName, Title, BirthDate, HireDate, Address, City, State, PostalCode, Country, Phone, Email)

VALUES

    (1, 'Michael', 'Smith', 'Manager', '1980-01-01', '2000-01-01', '789 Broadway', 'Metropolis', 'IL', '12345', 'USA', '789-456-1230', 'michael@example.com'),

    (2, 'Emily', 'Davis', 'Sales Associate', '1985-05-15', '2010-06-01', '456 Oak St', 'Smalltown', 'CA', '54321', 'USA', '456-789-0123', 'emily@example.com');
```

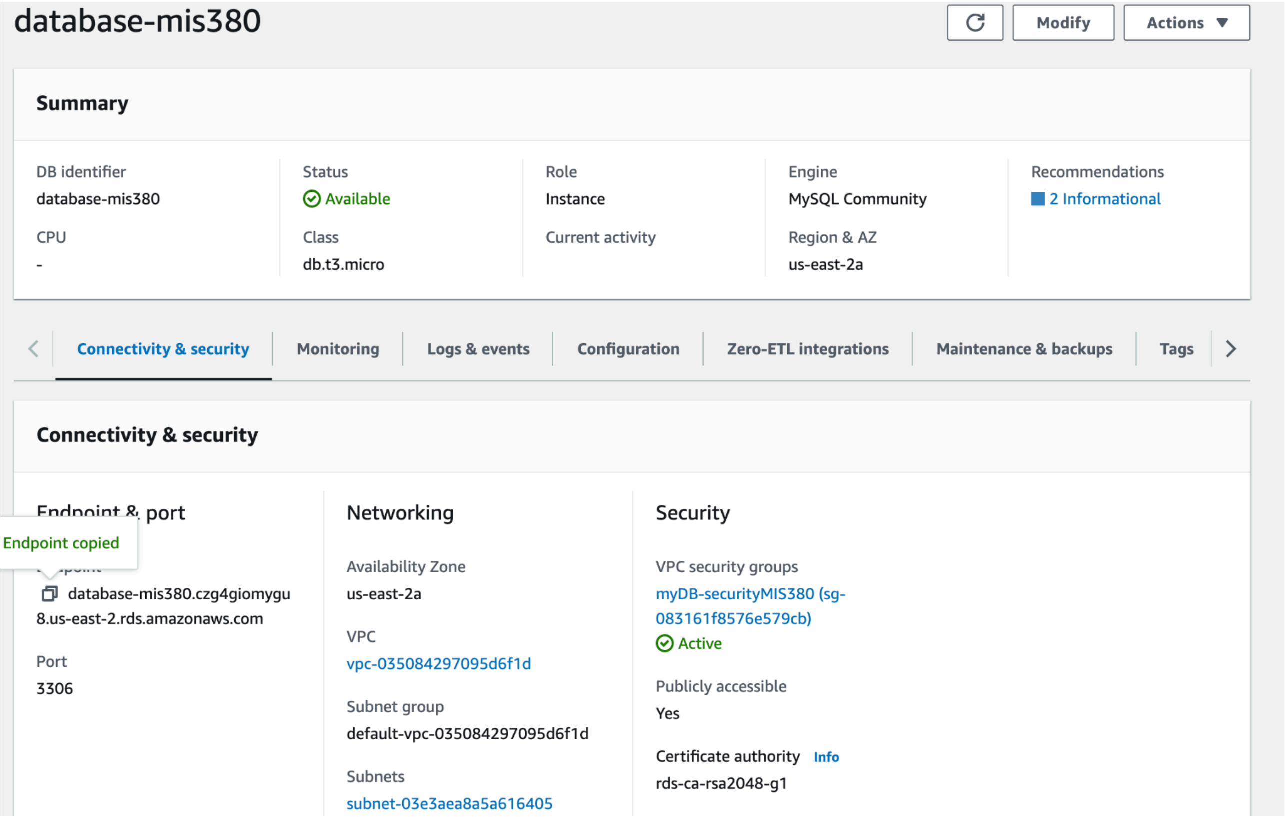
The SQL Code Passed all tests:



```sql
INSERT INTO Inventory (InventoryID, ProductID, SupplierID, PurchaseDate, QuantityPurchased, PurchasePrice)
VALUES
    (1, 1, 1, '2024-04-01', 50, 8.50),
    (2, 2, 1, '2024-04-05', 30, 25.00),
    (3, 3, 2, '2024-03-20', 100, 12.00),
    (4, 4, 2, '2024-04-10', 10, 40.00),
    (5, 5, 1, '2024-04-15', 60, 7.00),
    (6, 6, 2, '2024-03-25', 5, 150.00);

INSERT INTO Customer (CustomerID, FirstName, LastName, Address, City, State, PostalCode, Country, Phone, Email)
VALUES
    (1, 'Alice', 'Johnson', '789 Main St', 'Smalltown', 'NY', '54321', 'USA', '789-012-3456', 'alice@example.com'),
    (2, 'Bob', 'Smith', '456 Elm St', 'Cityville', 'CA', '98765', 'USA', '456-123-7890', 'bob@example.com');

INSERT INTO SalesTransaction (TransactionID, CustomerID, ProductID, TransactionDate, QuantitySold, TotalAmount)
VALUES
    (1, 1, 1, '2024-04-05', 2, 21.98),
    (2, 1, 3, '2024-04-10', 1, 15.99),
    (3, 2, 5, '2024-04-12', 3, 29.97),
    (4, 2, 6, '2024-04-15', 1, 199.99);

INSERT INTO Employee (EmployeeID, FirstName, LastName, Title, BirthDate, HireDate, Address, City, State, PostalCode, Country,
VALUES
    (1, 'Michael', 'Smith', 'Manager', '1980-01-01', '2000-01-01', '789 Broadway', 'Metropolis', 'IL', '12345', 'USA', '789-4
    (2, 'Emily', 'Davis', 'Sales Associate', '1985-05-15', '2010-06-01', '456 Oak St', 'Smalltown', 'CA', '54321', 'USA', '45
```

```
    Time      Action                                                                               Response            Duration / Fetch T...
✓ 10 03:53:20  Apply changes to TeamProject_01                                                      Changes applied
✓ 11 03:53:47  CREATE TABLE Product (    ProductID INT PRIMARY KEY,     ProductName VARCHAR(1...    0 row(s) affected   0.032 sec
✓ 12 03:53:47  CREATE TABLE Supplier (    SupplierID INT PRIMARY KEY,     SupplierName VARCHAR(...  0 row(s) affected   0.0090 sec
✓ 13 03:53:47  CREATE TABLE Inventory (    InventoryID INT PRIMARY KEY,     ProductID INT,    Suppli... 0 row(s) affected   0.012 sec
✓ 14 03:53:47  CREATE TABLE Customer (    CustomerID INT PRIMARY KEY,     FirstName VARCHAR(5...    0 row(s) affected   0.0058 sec
✓ 15 03:53:47  CREATE TABLE Employee (    EmployeeID INT PRIMARY KEY,     FirstName VARCHAR(5...    0 row(s) affected   0.0049 sec
```

Deploy The Database



database-mis380

Summary

DB identifier
database-mis380

CPU
-

Status
Available

Class
db.t3.micro

Role
Instance

Current activity

Engine
MySQL Community

Region & AZ
us-east-2a

Recommendations
2 Informational

Connectivity & security

Endpoint & port

Endpoint copied

database-mis380.czg4giomygu8.us-east-2.rds.amazonaws.com

Port
3306

Networking

Availability Zone
us-east-2a

VPC
vpc-035084297095d6f1d

Subnet group
default-vpc-035084297095d6f1d

Subnets
subnet-03e3aea8a5a616405

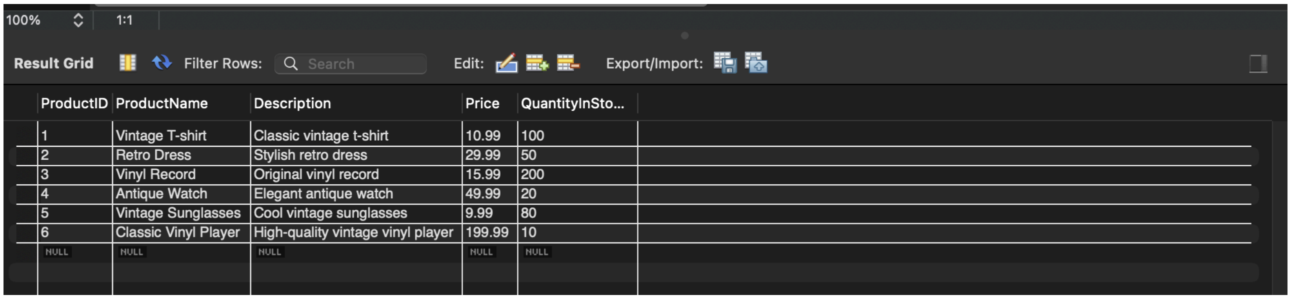
Security

VPC security groups
myDB-securityMIS380 (sg-083161f8576e579cb)
Active

Publicly accessible
Yes

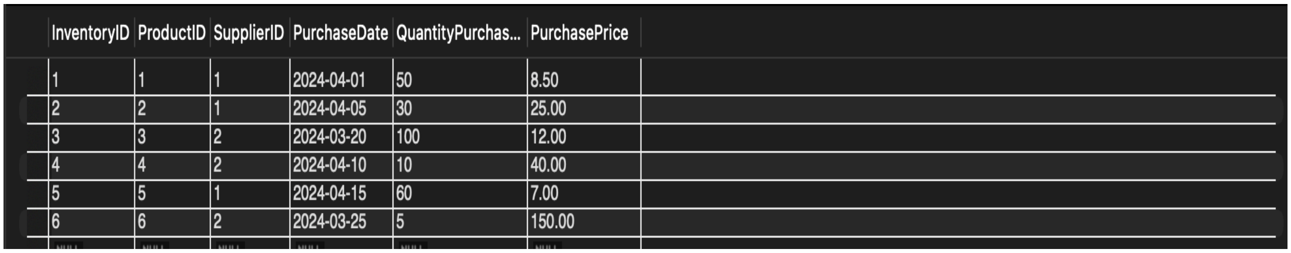
Certificate authority Info
rds-ca-rsa2048-g1

Product Table:



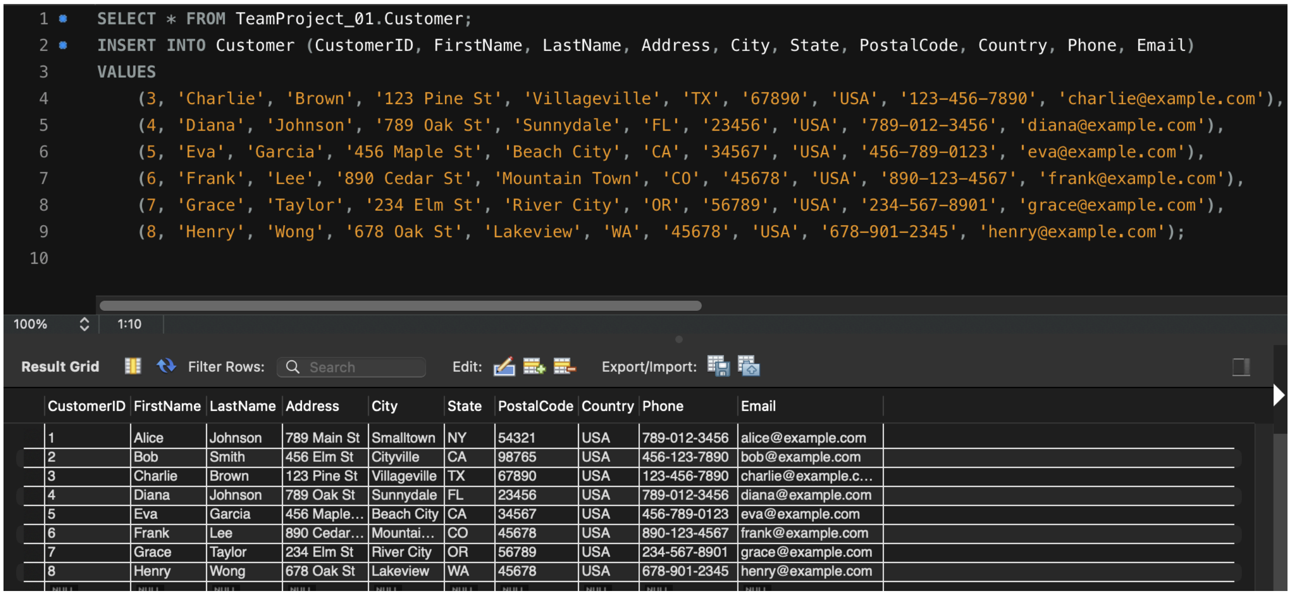
ProductID	ProductName	Description	Price	QuantityInSto...
1	Vintage T-shirt	Classic vintage t-shirt	10.99	100
2	Retro Dress	Stylish retro dress	29.99	50
3	Vinyl Record	Original vinyl record	15.99	200
4	Antique Watch	Elegant antique watch	49.99	20
5	Vintage Sunglasses	Cool vintage sunglasses	9.99	80
6	Classic Vinyl Player	High-quality vintage vinyl player	199.99	10
NULL	NULL	NULL	NULL	NULL

Inventory Table:



InventoryID	ProductID	SupplierID	PurchaseDate	QuantityPurchas...	PurchasePrice
1	1	1	2024-04-01	50	8.50
2	2	1	2024-04-05	30	25.00
3	3	2	2024-03-20	100	12.00
4	4	2	2024-04-10	10	40.00
5	5	1	2024-04-15	60	7.00
6	6	2	2024-03-25	5	150.00

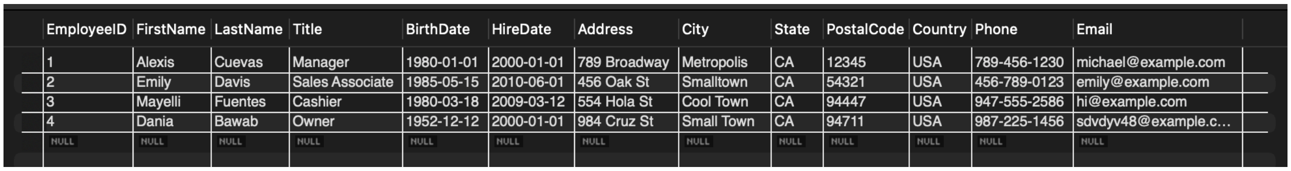
Customer Table:



```sql
SELECT * FROM TeamProject_01.Customer;
INSERT INTO Customer (CustomerID, FirstName, LastName, Address, City, State, PostalCode, Country, Phone, Email)
VALUES
    (3, 'Charlie', 'Brown', '123 Pine St', 'Villageville', 'TX', '67890', 'USA', '123-456-7890', 'charlie@example.com'),
    (4, 'Diana', 'Johnson', '789 Oak St', 'Sunnydale', 'FL', '23456', 'USA', '789-012-3456', 'diana@example.com'),
    (5, 'Eva', 'Garcia', '456 Maple St', 'Beach City', 'CA', '34567', 'USA', '456-789-0123', 'eva@example.com'),
    (6, 'Frank', 'Lee', '890 Cedar St', 'Mountain Town', 'CO', '45678', 'USA', '890-123-4567', 'frank@example.com'),
    (7, 'Grace', 'Taylor', '234 Elm St', 'River City', 'OR', '56789', 'USA', '234-567-8901', 'grace@example.com'),
    (8, 'Henry', 'Wong', '678 Oak St', 'Lakeview', 'WA', '45678', 'USA', '678-901-2345', 'henry@example.com');
```

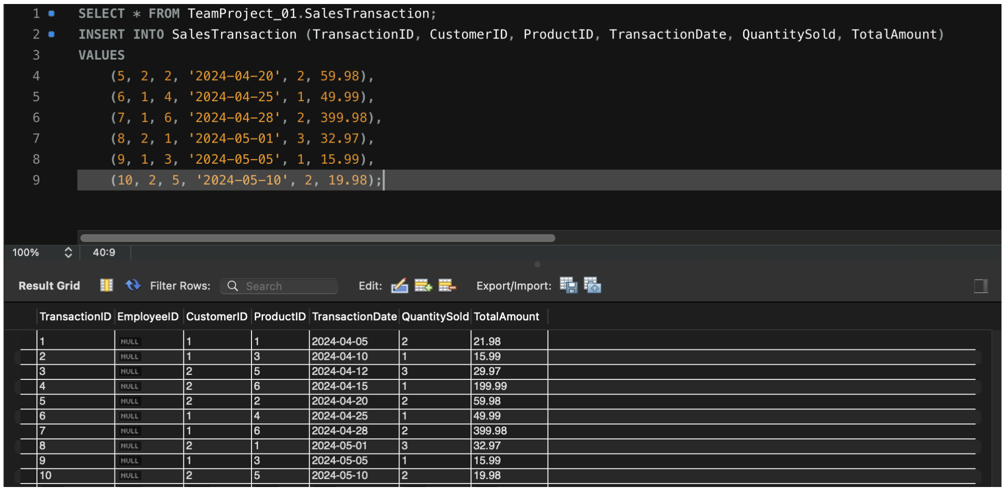
CustomerID	FirstName	LastName	Address	City	State	PostalCode	Country	Phone	Email
1	Alice	Johnson	789 Main St	Smalltown	NY	54321	USA	789-012-3456	alice@example.com
2	Bob	Smith	456 Elm St	Cityville	CA	98765	USA	456-123-7890	bob@example.com
3	Charlie	Brown	123 Pine St	Villageville	TX	67890	USA	123-456-7890	charlie@example.c...
4	Diana	Johnson	789 Oak St	Sunnydale	FL	23456	USA	789-012-3456	diana@example.com
5	Eva	Garcia	456 Maple...	Beach City	CA	34567	USA	456-789-0123	eva@example.com
6	Frank	Lee	890 Cedar...	Mountai...	CO	45678	USA	890-123-4567	frank@example.com
7	Grace	Taylor	234 Elm St	River City	OR	56789	USA	234-567-8901	grace@example.com
8	Henry	Wong	678 Oak St	Lakeview	WA	45678	USA	678-901-2345	henry@example.com

Employee Table:



EmployeeID	FirstName	LastName	Title	BirthDate	HireDate	Address	City	State	PostalCode	Country	Phone	Email
1	Alexis	Cuevas	Manager	1980-01-01	2000-01-01	789 Broadway	Metropolis	CA	12345	USA	789-456-1230	michael@example.com
2	Emily	Davis	Sales Associate	1985-05-15	2010-06-01	456 Oak St	Smalltown	CA	54321	USA	456-789-0123	emily@example.com
3	Mayelli	Fuentes	Cashier	1980-03-18	2009-03-12	554 Hola St	Cool Town	CA	94447	USA	947-555-2586	hi@example.com
4	Dania	Bawab	Owner	1952-12-12	2000-01-01	984 Cruz St	Small Town	CA	94711	USA	987-225-1456	sdvdyv48@example.c...
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

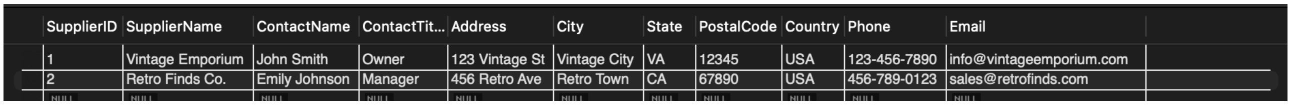
Transaction Table:



```sql
SELECT * FROM TeamProject_01.SalesTransaction;
INSERT INTO SalesTransaction (TransactionID, CustomerID, ProductID, TransactionDate, QuantitySold, TotalAmount)
VALUES
    (5, 2, 2, '2024-04-20', 2, 59.98),
    (6, 1, 4, '2024-04-25', 1, 49.99),
    (7, 1, 6, '2024-04-28', 2, 399.98),
    (8, 2, 1, '2024-05-01', 3, 32.97),
    (9, 1, 3, '2024-05-05', 1, 15.99),
    (10, 2, 5, '2024-05-10', 2, 19.98);
```

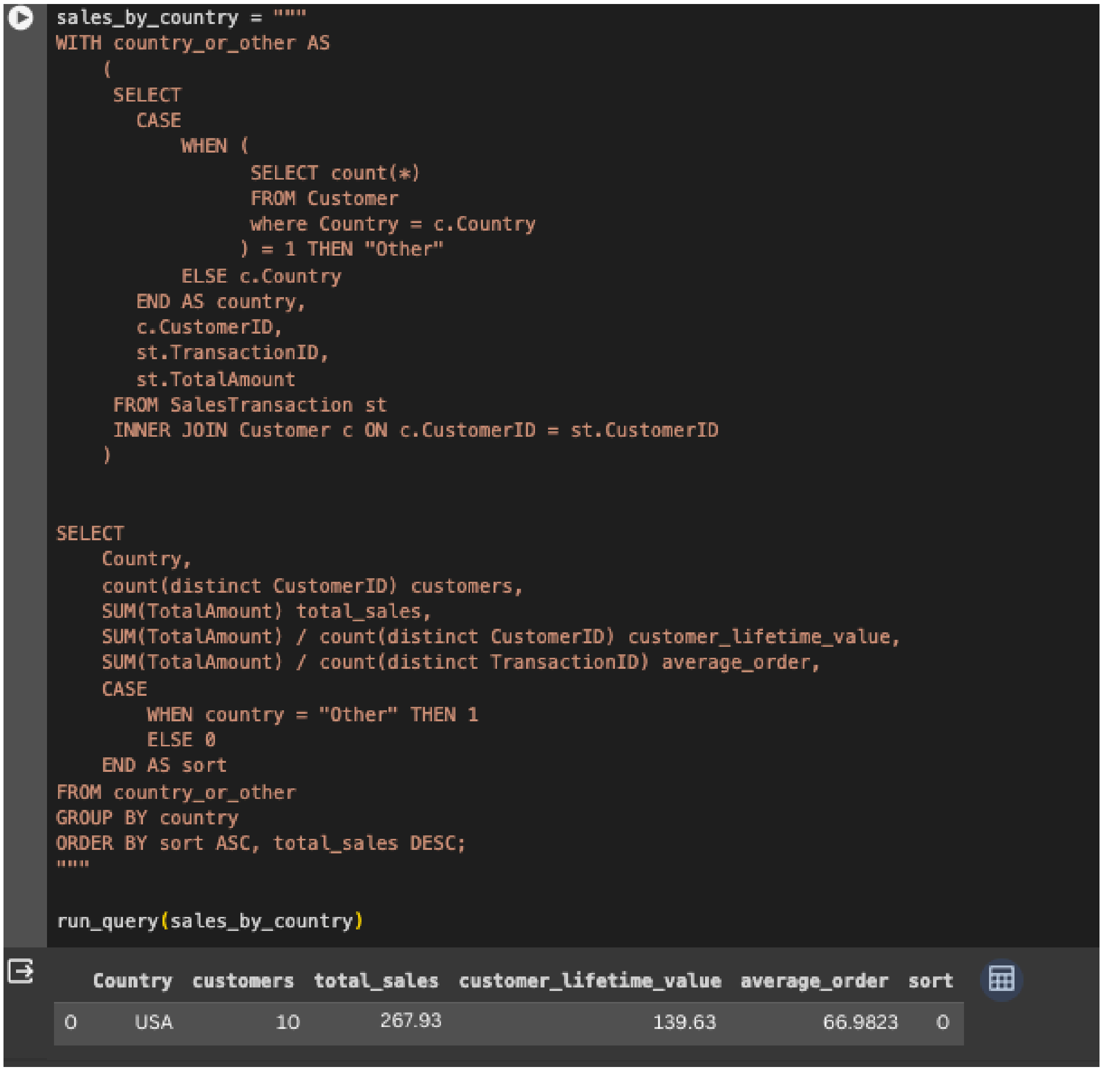
TransactionID	EmployeeID	CustomerID	ProductID	TransactionDate	QuantitySold	TotalAmount
1	NULL	1	1	2024-04-05	2	21.98
2	NULL	1	3	2024-04-10	1	15.99
3	NULL	2	5	2024-04-12	3	29.97
4	NULL	2	6	2024-04-15	1	199.99
5	NULL	2	2	2024-04-20	2	59.98
6	NULL	1	4	2024-04-25	1	49.99
7	NULL	1	6	2024-04-28	2	399.98
8	NULL	2	1	2024-05-01	3	32.97
9	NULL	1	3	2024-05-05	1	15.99
10	NULL	2	5	2024-05-10	2	19.98

Supplier Table:



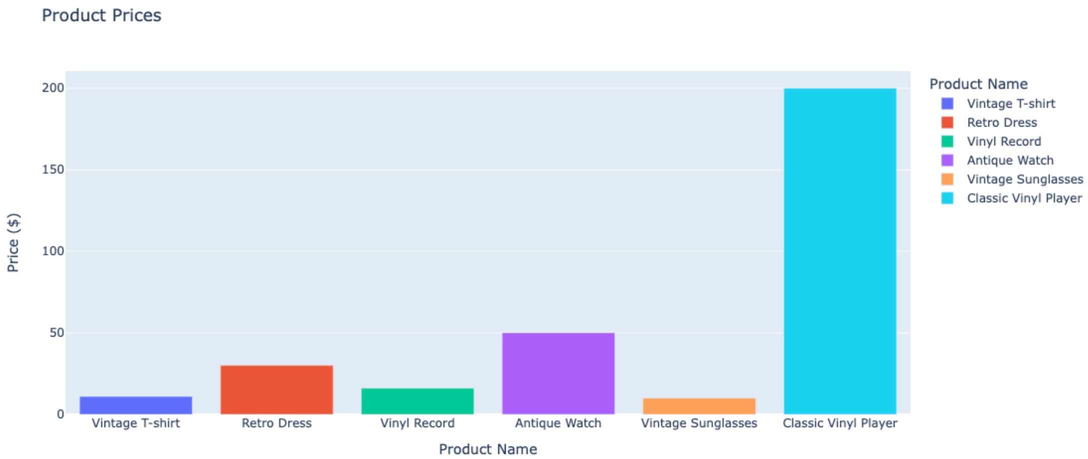
SupplierID	SupplierName	ContactName	ContactTit...	Address	City	State	PostalCode	Country	Phone	Email
1	Vintage Emporium	John Smith	Owner	123 Vintage St	Vintage City	VA	12345	USA	123-456-7890	info@vintageemporium.com
2	Retro Finds Co.	Emily Johnson	Manager	456 Retro Ave	Retro Town	CA	67890	USA	456-789-0123	sales@retrofinds.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

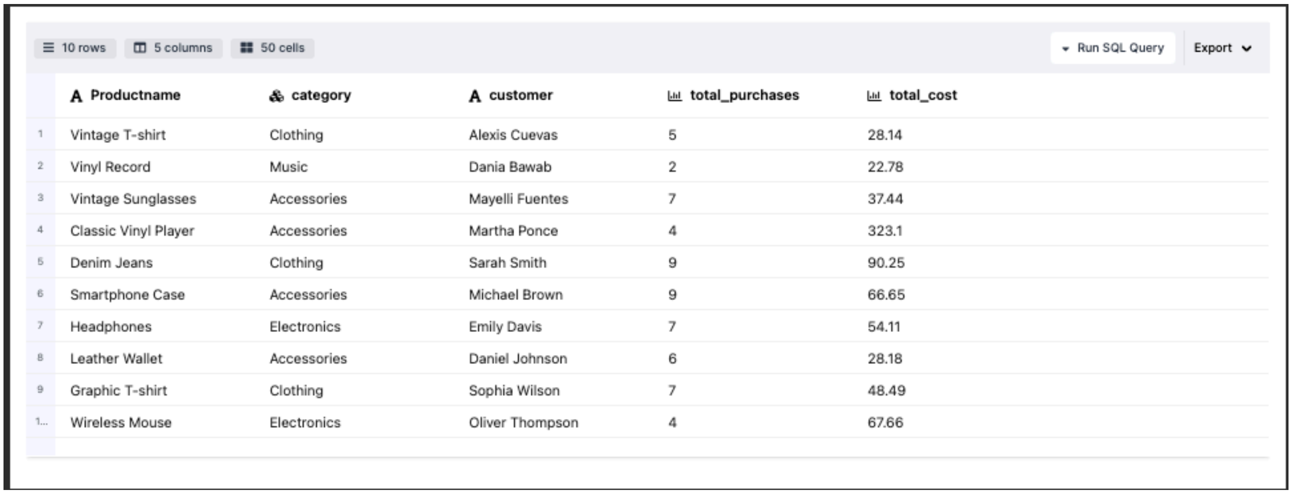
Analytic Dashboard



```
sales_by_country = """
WITH country_or_other AS
    (
     SELECT
       CASE
           WHEN (
                 SELECT count(*)
                 FROM Customer
                 where Country = c.Country
                ) = 1 THEN "Other"
           ELSE c.Country
       END AS country,
       c.CustomerID,
       st.TransactionID,
       st.TotalAmount
     FROM SalesTransaction st
     INNER JOIN Customer c ON c.CustomerID = st.CustomerID
    )


SELECT
    Country,
    count(distinct CustomerID) customers,
    SUM(TotalAmount) total_sales,
    SUM(TotalAmount) / count(distinct CustomerID) customer_lifetime_value,
    SUM(TotalAmount) / count(distinct TransactionID) average_order,
    CASE
        WHEN country = "Other" THEN 1
        ELSE 0
    END AS sort
FROM country_or_other
GROUP BY country
ORDER BY sort ASC, total_sales DESC;
"""

run_query(sales_by_country)
```

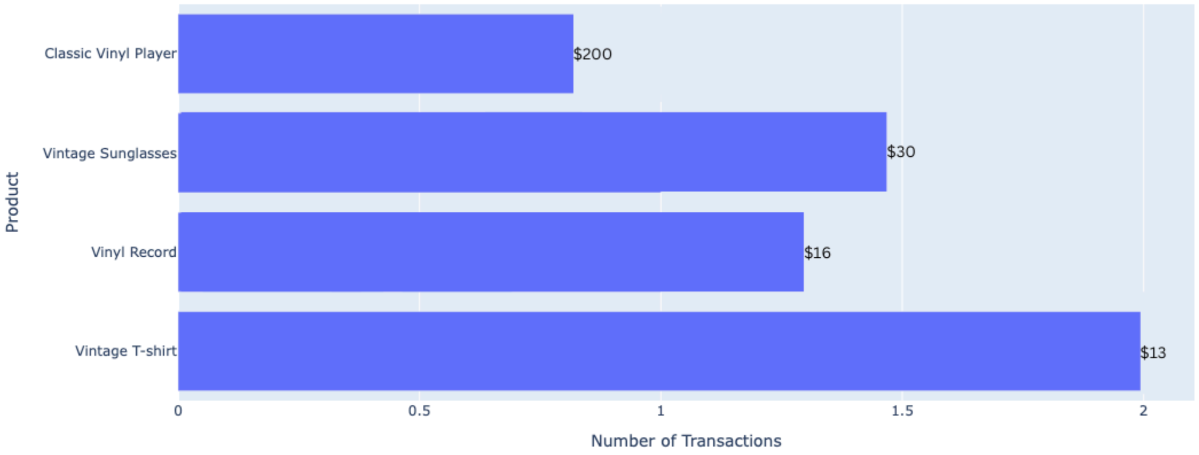
```
   Country  customers  total_sales  customer_lifetime_value  average_order  sort
0  USA             10       267.93                   139.63        66.9823     0
```



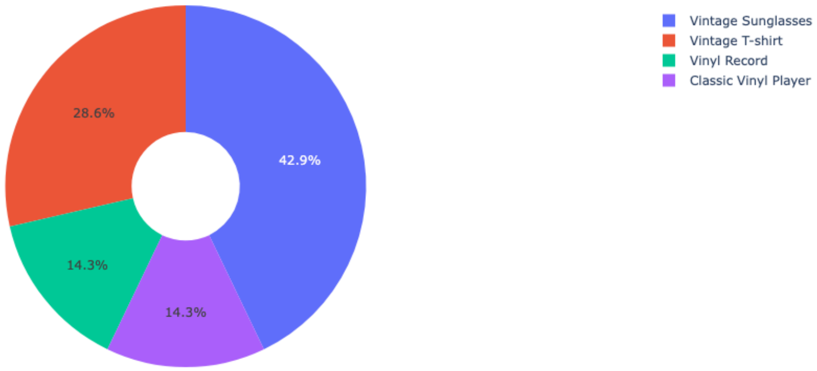


	Productname	category	customer	total_purchases	total_cost
1	Vintage T-shirt	Clothing	Alexis Cuevas	5	28.14
2	Vinyl Record	Music	Dania Bawab	2	22.78
3	Vintage Sunglasses	Accessories	Mayelli Fuentes	7	37.44
4	Classic Vinyl Player	Accessories	Martha Ponce	4	323.1
5	Denim Jeans	Clothing	Sarah Smith	9	90.25
6	Smartphone Case	Accessories	Michael Brown	9	66.65
7	Headphones	Electronics	Emily Davis	7	54.11
8	Leather Wallet	Accessories	Daniel Johnson	6	28.18
9	Graphic T-shirt	Clothing	Sophia Wilson	7	48.49
1...	Wireless Mouse	Electronics	Oliver Thompson	4	67.66

Top Selling Products (Randomized)



Total Quantity Sold by Product



🟦 Vintage Sunglasses
🟥 Vintage T-shirt
🟩 Vinyl Record
🟪 Classic Vinyl Player