**Copyright Incorporated Database**

**Group Members:**

Caleb Walls

Corin Canepa

John Luke Denny

Cole McKay

**Domain Application:**

Our data base is an Ecommerce data base that will be used for tracking information. The data will be used to track customer purchases. In this table customers may also be sellers. Therefore the data will also keep track of what the sellers sell. It will be able to give information such as delivery dates and even tell the user if the package has been delivered.

**Domain Design:**

We have a total of 7 tables:

* Account
* Address
* Cart
* Cart\_Item
* Customer
* Item
* Seller

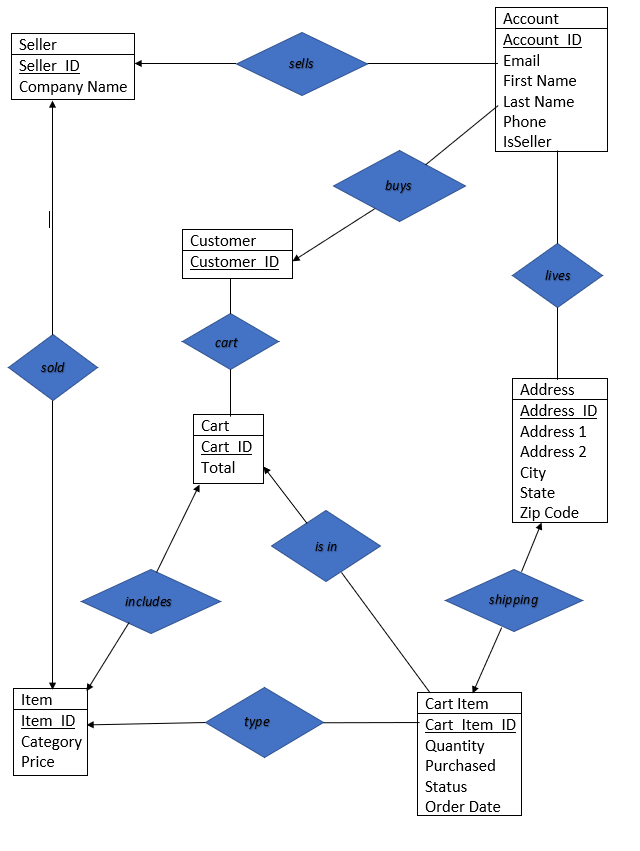
**Attributes of each table:**

* **Account-**
* Account\_ID INT not Null,
* Email Varchar(30),
* FirstName Varchar(10),
* LastName Varchar(10),
* Phone Varchar(12),
* IsSeller Boolean not null,
* Primary key (Account\_ID)
* **Customer-**
* Customer\_ID varchar(20) not null,
* Cart\_ID varchar(20) not null,
* Account\_ID int not null,
* Primary Key (Customer\_ID),
* Foreign Key (Account\_ID) references Account(Account\_ID)
* **Seller-**
* Seller\_ID int not null,
* Company\_Name Varchar(30),
* Account\_ID int not null,
* Primary Key (Seller\_ID),
* Foreign Key (Account\_ID) references Account(Account\_ID)
* **Address-**
* Address\_ID int not null,
* Address  varchar(30) not null,
* City varchar(20) not null,
* State varchar(20) not null,
* Account\_ID int,
* Primary Key (Address\_ID),
* Foreign Key (Account\_ID) references Account(Account\_ID)
* **Item-**
* Item\_ID int not null,
* Category varchar(20) not null,
* ItemPrice float not null,
* Seller\_ID int not null,
* Primary key (Item\_ID),
* Foreign key (Seller\_ID) references Seller(Seller\_ID)
* **Cart-**
* Account\_ID int not null
* Customer\_ID varchar(20) not null,
* Cart\_ID varchar(12) not null,
* Foreign Key(Account\_ID) references Account(Account\_ID)
* **Cart\_Item-**
* Quantity int,
* CartItem\_ID varchar(12) not null,
* Purchased boolean,
* Status varchar(10) not null,
* Cart\_ID varchar(12) not null,
* Item\_ID  int not null,
* Address ID  int not null,
* Order Date date varchar(12),
* Primary Key(CartItem\_ID),
* Foreign Key (Cart\_ID) references Cart(Cart\_ID),
* Foreign Key (Item\_ID) references Item(Item\_ID),
* Foreign Key (Address\_ID) references Address(Address\_ID)

**Design Description:**

* The normalization of this database is Level 2.
* The Account table is for all 400 users whether they are a seller or just simply a customer. The primary key for this table is the Account\_ID because the account id will determine each characteristic.
* The Address table contains 400 users and whether they are a seller or a customer. The primary key is once again the Account\_ID and it is also the foreign key of the table. This table will give the address of every person in the table.
* The Cart table contains 400 entries . This table contains what each user has in their cart and how much in total they have in their cart. The primary key for the table is the Cart\_ID.
* The Cart\_Item table contains 100 entries. This table contains the quantity, a Boolean to check whether the item was purchased, the status of the item, the cart id , the item id, the address id and the order date. Cart\_ID,Item\_ID, and Adress\_ID are the foreign keys of this table .
* The Customer table contains 400 users and contains the customer’s id and their cart id. It also contains their account id. The primary key for this table is the Customer\_ID. The foreign keys would be Cart\_ID and Account\_ID.
* The Item table contains 100 items . It also contains the price, seller id , and item id. The primary key for this table would be the Item\_ID and the foreign key would be Address\_ID.
* The Seller table contains 100 sellers. It also contains the company name and the account id. The primary key would be the Seller\_ID and the foreign key would be the Account\_ID.
* In these tables sellers are also allowed to be buyers . Therefore ID’s for customers and sellers will overlap.

**ER Diagram**



**Relational Diagram**

**Diagram

Description automatically generated**

**Constraints and Assumptions**

* Every account has a unique ID.
* No account can have a NULL value
* Buyers and Sellers ID will overlap

**Table Creation**

Drop Table If Exists CartItem, Item, Address, Seller, Customer, Cart, Account;

Create Table Account(

Account\_ID INT not Null,

FirstName Varchar(10),

LastName Varchar(10),

Email Varchar(30),

Phone Varchar(12),

IsSeller Boolean not null,

Primary key (Account\_ID)

);

Create table Cart(

Cart\_ID varchar(12) not null,

Total float,

Primary key (Cart\_ID)

);

Create Table Customer(

Account\_ID int not null,

Customer\_ID varchar(20) not null,

Cart\_ID varchar(12) not null,

Primary Key (Customer\_ID),

Foreign Key (Account\_ID) references Account(Account\_ID)

);

Create Table Seller(

Account\_ID int not null,

Seller\_ID int not null,

Company\_Name Varchar(30),

Primary Key (Seller\_ID),

Foreign Key (Account\_ID) references Account(Account\_ID)

);

Create table Address(

Address\_ID int not null,

Address varchar(30) not null,

City varchar(20) not null,

State varchar(20) not null,

Account\_ID int,

Primary Key (Address\_ID),

Foreign Key (Account\_ID) references Account(Account\_ID)

);

Create table Item(

Item\_ID int not null,

Category varchar(20) not null,

ItemPrice float not null,

Seller\_ID int not null,

Primary key (Item\_ID),

Foreign key (Seller\_ID) references Seller(Seller\_ID)

);

Create Table CartItem(

CartItem\_ID varchar(12) not null,

Quantity int,

Purchased boolean,

OrderDate varchar(12),

OrderStatus varchar(10) not null,

Item\_ID int not null,

Cart\_ID varchar(12) not null,

Address\_ID int not null,

Primary Key (CartItem\_ID),

Foreign Key (Cart\_ID) references Cart(Cart\_ID),

Foreign Key (Item\_ID) references Item(Item\_ID),

Foreign Key (Address\_ID) references Address(Address\_ID)

);

**SQL Queries**

* **The top 5 Cities with the most items being shipped there**
* Tables Query Uses:
  + Address
  + CartItem
* Purpose: Great for tracking the cities or cities with the most (or lowest) purchases in quantity (not necessarily based on revenue. This could be used with helping with shipping changes or increasing/decreasing warehouse space in certain areas due to volume.

Select A.City, A.State, sum(CI.Quantity)

from Address A, CartItem CI

Where A.Address\_ID = CI.Address\_ID

group by A.City

Order by sum(CI.Quantity)

Limit 5;

Text

Description automatically generated

**Top 5 revenue items**

* Tables Query Uses:
  + CartItem
  + Account
* Purpose: Lets Sellers know what items are making the most money which helps to decide where production costs go.
* select CI.Item\_ID, round(sum(I.ItemPrice \* CI.Quantity),2) as Revenue from CartItem CI, Item I Where CI.Item\_ID = I.Item\_ID and CI.Purchased = TRUE Group by CI.Item\_ID Order by Revenue desc Limit 5;
* Text

  Description automatically generated

**Locate all the sellers in Louisiana**

* Tables Query Uses:
  + Address
  + Account
* Purpose: As a customer query, some people like to buy local, or an advertiser can use this query to help show customers.
* select ac.Account\_ID
* from Account ac
* left join Address ad
* on ac.Account\_ID = ad.Account\_ID
* where ad.State = "Louisiana"
* and ac.IsSeller = TRUE;
* A picture containing diagram

  Description automatically generated
* **Top 50% of states with highest number of orders**
* Tables Query Uses:
  + CartItem
  + Address

Purpose: Allows Sellers to know where distributors should be set up to optimize shipping costs

Select A.State, Count(A.State) as numOrders

From (Address as A)

Right Join (CartItem as CI)

on A.Address\_ID = CI.Address\_ID

Where CI.OrderStatus in ("pending", "processed")

Group by A.State

Having numOrders > Avg(numOrders)

Order by numOrders desc;

A picture containing text

Description automatically generated

* **Most Expensive Item Per Category**
* Tables Query Uses:
  + Item
* Purpose: Great for tracking the most expensive Item in each categories. Queries similar to this could be used for sorting from lowest to highest price

Select Category,Item\_ID, MAX(ItemPrice) as Tot\_Price

From Item

Group by Category;

A picture containing chart

Description automatically generated

* **Top Revenue Generating Cities**
* Tables Query Uses:
  + Address
  + CartItem
* Purpose: Great for tracking the cities generating the most revenue from purchases

Select A.City, A.State, ROUND(I.ItemPrice \* CI.Quantity, 1) AS Revenue

From Address A, Item I, CartItem CI

Where A.Address\_ID = CI.Address\_ID AND I.Item\_ID = CI.Item\_ID

Group by A.City

Order by Revenue desc

Limit 5;

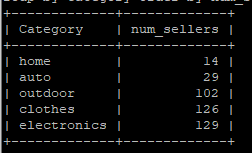
A picture containing calendar

Description automatically generated

**Item categories with highest number of sellers**

* Tables Query Uses:
  + Item

Purpose: Shows which markets might be easy to enter or be oversaturated.

* select Category, count(Seller\_ID) as num\_sellers From Item Group by Category Order by num\_sellers;
* 
* **The Top 5 Spenders most purchased Item Category**
* Tables Query Uses:
  + Customer
  + Account
  + Item
  + Cart
  + CartItem
* Purpose: Great for tracking your top spenders, but also knowing what category they are spending the most money in. You can use this or a similar query to target ads for other items in the categories top spenders purchase most from.

Select a.FirstName, a.LastName, ct.Total, I.Category AS Top\_Category

From Item I, CartItem CI, Account a, Customer c, Cart ct,

(Select count(I.Category) From Item I, CartItem CI, Account a, Customer c, Cart ct Where I.Item\_ID = CI.Item\_ID AND c.Cart\_ID = CI.Cart\_ID AND CI.Cart\_ID = ct.Cart\_ID AND c.Account\_ID = a.Account\_ID Group by ct.Cart\_ID, I.Category Order by count(I.Category) desc Limit 1) as top

Where I.Item\_ID = CI.Item\_ID AND c.Cart\_ID = CI.Cart\_ID AND CI.Cart\_ID = ct.Cart\_ID AND c.Account\_ID = a.Account\_ID

Group by CI.Cart\_ID

Order by ct.total desc

Limit 5;

A picture containing text, scoreboard

Description automatically generated

**Checks if a product is pending and the purchase is true and the total is over 10 dollars**

* Tables Query Uses:
  + CartItem
  + Item

Purpose: Lets Seller know what orders they need to fulfill and are making a reasonable sale on.

select  CI.CartItem\_ID, round(sum(CI.Quantity\*I.ItemPrice),2) as total

from (CartItem CI)

Left join Item I

on CI.Item\_ID = I.Item\_ID

where CI.OrderStatus = “pending“ and CI.Purchased = TRUE

group by CI.CartItem\_ID

having total >= 10.0;

A picture containing table

Description automatically generated

**Find the seller with the most sells**

* Tables Query Uses:
  + Item

Purpose: Helps to identify Seller with largest market share.

* select Seller\_ID,count(Seller\_ID) as Total\_Sells from Item Group by Seller\_ID Order by Total\_Sells desc  Limit 5;

