Benjamin Michalowicz, Ian Peitzsch, Veronica Quintana

CSE 305: Principles of Database Systems

Spring 2020

Homework 2 Documentation

Our group used MySQL Workbench (8.0) to create our database. MySQL worked well for us as a program with an easy-to-understand UI and SQL writing/testing/debugging features. Our database has Customer, Item, Seller, Address, Reviews, Shipment, Employee, CreditCard, Payment, and Buys tables. The design focuses around the Customer, as the center of an E-Commerce application will always be the people who use the application.

Each customer has their personal details (such as name, payment, email) and references an address. This allows for the customer to be registered in the database, but the address need not be immediately added to their information. This additionally requires either all or none of the address to be used, rather than allowing portions to be left out. The Seller has a name and an ID, and the name defaults to Anonymous in the case that a seller does not input a name for themselves. Items in the database link to the Seller; they also have other identifying information such as price, type, quantity available, and name. The quantity is used in conjunction with the Inventory to determine how much of an item is present in the database. The types are also useful in this regard, as they help a user to see similarly typed items.

With the Reviews functionality of the E-Commerce system, each customer is permitted a single review per item. Reviews are linked to a customer’s ID, and the customer need not give a written review but must submit a rating out of 10. Each purchase of course comes with a Shipment, which has its own ID and reference’s a customer’s address. Shipments have a status of PROCESSED, SHIPPED, or ARRIVED, for the convenience of the user eagerly awaiting their purchase from the application. The Buys table keeps track of what the customer has bought/wants to buy. It links the customer to an item, gives the quantity of the items to the user desires to purchase, then gives the price of buying the quantity of that given item. Payment references the CreditCard table, as they go hand in hand to allow users to actually purchase their items and provide a type of payment. The CreditCard table references a customer, and the payment references the card and the amount.

Create Statements:

drop database if exists mydb;

create Database myDB;

use myDB;

CREATE TABLE Seller(

SellerId INT AUTO\_INCREMENT,

SellerName VARCHAR(64) DEFAULT "Anonymous",

Primary Key(SellerId) );

CREATE TABLE Item(

ItemId INT AUTO\_INCREMENT,

Price DECIMAL(10,2) DEFAULT 0.00,

ItemType VARCHAR(45) default '',

Quantity INT DEFAULT 0,

ItemName VARCHAR(45) default 'Item',

SellerId INT NOT NULL,

Primary Key(ItemId),

Foreign Key(SellerId) References Seller(SellerId) );

create table Address(

AddId int auto\_increment,

Address varchar(45) not null,

Town varchar(45) not null,

State char(2) not null,

ZIP int not null,

primary key(AddID) );

create table Customer(

CustomerId int auto\_increment,

FirstName varchar(45) not null,

LastName varchar(45) not null,

Email varchar(45),

Address int,

primary key(CustomerId),

foreign key(Address) references Address(AddId));

create table Reviews(

CustomerId int,

ItemId int,

Rating int not null,

Review varchar(256),

primary key(CustomerId, ItemId),

foreign key(CustomerId) references Customer(CustomerId),

foreign key(ItemId) references Item(ItemId) );

create table Shipment(

ShipmentId int auto\_increment,

ShipmentAddress int not null,

ShipmentStatus char(9) not null,

primary key(ShipmentId),

check(ShipmentStatus in ('ARRIVED', 'PROCESSED', 'SHIPPED')),

foreign key(ShipmentAddress) references Address(AddID) );

create table Employee(

EmployeeId int auto\_increment,

EmployeeRole varchar(64),

FirstName varchar(45) not null,

LastName varchar(45) not null,

Joined date not null,

SupervisorId int,

primary key(EmployeeId),

foreign key(SupervisorId) references Employee(EmployeeId),

check(SupervisorId != EmployeeId) );

create table CreditCard(

Num bigint,

Own int not null,

PaymentType varchar(10) not null,

ExpirationDtae date not null,

Primary key(Num),

foreign key(Own) references Customer(CustomerId),

check(PaymentType in ('MasterCard', 'Visa', 'Discover', 'Chase')) );

create table Payment(

PaymentId int auto\_increment,

CreditCard bigint not null,

Amount decimal(10,2),

primary key(PaymentId),

foreign key(CreditCard) references CreditCard(Num) );

create table Buys(

CustomerId int,

ItemId int,

Quantity int not null,

Price decimal(10,2),

PaymentId int,

primary key(CustomerId, ItemId),

foreign key(CustomerId) references Customer(CustomerId),

foreign key(ItemId) references Item(ItemId),

foreign key(PaymentId) references Payment(PaymentId) );

Insert Statements:

use myDB;

insert into Seller values (null, "Adidas");

insert into Seller values (null, "Nike");

insert into Seller values (null, "Reebok");

insert into Seller values (null, "Champion");

insert into Item values (null, 20.95, 'Track Jacket', 50, 'Adidas Track Jacket Blue', (select SellerId from seller where SellerName = 'Adidas'));

insert into Item values (null, 15.05, 'Track Pants', 40, 'Adidas Track Pants Blue', (select SellerId from seller where SellerName = 'Adidas'));

insert into Item values (null, 78.00, 'Shoes', 25, 'Nike Air Max', (select SellerId from seller where SellerName = 'Nike'));

insert into Item values (null, 53.55, 'Shoes', 37, 'Nike Flex', (select SellerId from seller where SellerName = 'Nike'));

insert into Item values (null, 98.55, 'Shoes', 40, 'Reebok Lifestyle Leather', (select SellerId from seller where SellerName = 'Reebok'));

insert into Item values (null, 32.99, 'Shoes', 45, 'Reebok Club C Vintage', (select SellerId from seller where SellerName = 'Reebok'));

insert into Item values (null, 12.03, 'Sweatshirt', 48, 'Champion Sweatshirt Blue', (select SellerId from seller where SellerName = 'Champion'));

insert into Item values (null, 14.03, 'Sweatshirt', 60, 'Champion Sweatshirt Grey', (select SellerId from seller where SellerName = 'Champion'));

insert into Address values (null, '123 Lane St.', 'Stony Brook', 'NY', 11790);

insert into Address values (null, '444 Mann Rd.', 'Niantic', 'CT', 06357);

insert into Customer values (null, 'John', 'Doe', 'johndoe@email.com', (select AddId from Address where Address = '123 Lane St.' and Town = 'Stony Brook' and State = 'NY' and ZIP = 11790));

insert into Customer values (null, 'Man', 'McMann', 'man@manman.org', null);

insert into Customer values (null, 'Liam', 'Johnson', 'buddyman@hotmail.com', (select AddId from Address where Address = '444 Mann Rd.' and Town = 'Niantic' and State = 'CT' and ZIP = 06357));

insert into Customer values (null, 'Geoff', 'Hernandez', 'jeff@jeff.jeff', null);

insert into Reviews values (1, 2, 7, "They're comfortable pants.");

insert into Reviews values (3, 5, 3, null);

insert into Reviews values (2, 3, 8, 'Best. Shoes. Ever.');

insert into Reviews values (1, 3, 5, 'Could be better.');

insert into Shipment values (null, 1, 'SHIPPED');

insert into Shipment values (null, 2, 'ARRIVED');

insert into Employee values (null, 'Supervisor', 'Praveen', 'Tripathi', STR\_TO\_DATE('1-01-2019', '%m-%d-%Y'), null);

insert into Employee values (null, 'Programmer', 'Ian', 'Peitzsch', STR\_TO\_DATE('1-28-2019', '%m-%d-%Y'), 1);

insert into Employee values (null, 'Programmer', 'Ben', 'Benson', STR\_TO\_DATE('1-28-2019', '%m-%d-%Y'), 1);

insert into Employee values (null, 'Programmer', 'V', 'Vvvvvvvvvvvv', STR\_TO\_DATE('1-28-2019', '%m-%d-%Y'), 1);

insert into CreditCard values (1111111111, 1, 'Discover', STR\_TO\_DATE('1-08-2224', '%m-%d-%Y'));

insert into CreditCard values (2222222222, 2, 'Visa', STR\_TO\_DATE('1-02-2222', '%m-%d-%Y'));

insert into CreditCard values (3333333333, 1, 'Chase', STR\_TO\_DATE('1-03-2220', '%m-%d-%Y'));

insert into Payment values (null, 1111111111, 36.00);

insert into Payment values (null, 2222222222, 78.00);

insert into Buys values (1, 2, 1, 15.05, 1);

insert into Buys values (1, 1, 1, 20.95, 1);

insert into Buys values (2, 3, 1, 78.00, 2);

insert into Buys values (4, 8, 3, 42.09, null);