PDB Fall 2020 Project Report

E-commerce Website Manager

Karthik Panambur <kp2492>
Omkar Kumbhar <opk208>
Port Number: 8581

Introduction

a) E-commerce website database manager will allow managing retail business for different views for customers and offices. A customer will be able to place an order, see his order details which will include product information and payment details. Both the customer and the office will be able to see different metrics related to revenue and products ordered.
b) The Entity Sets in our applications are
    1) Customers
    2) Products
    3) Offices
    4) Payments
    5) Order
    6) Order Details
    7) Employees
    8) Customer Address
    9) Office Address
    10) Inventory

Customers can have multiple addresses associated with them, for example Ship to and Bill to address, There is a participation constraint on the Customer with the Customer Address Entity set i.e every customer must have some address associated with them. Customer Address has both key and participation constraint on the Customer table i.e 1 Address can belong to only 1 customer and all address in the address table must belong to some customer

Offices can have multiple addresses associated with them, for example Retail and Warehouse address, There is a participation constraint on the Offices with the Offices Address Entity set i.e every Office must have some address associated with them. Office Address has both key and participation constraints on the Office table i.e 1 Address can belong to only 1 Office and all addresses in the address table must belong to some Office.

Customers can Order products which are recorded in the Orders and order details table, there are no constraints here. A customer can order any number of products. A product can be part of any number of orders.

Order table is associated with Order details table which will have line level details about the order, like product ordered and the quantity. Order detail table has participation constraint with the Order table, i.e every detail is associated with some Order in the Order table. Similarly there is a participation constraint on Order table, i,e every Order in the Order table also has at least one Order detail record associated with it.

Order table also has a relation with the payments table, which will record whether a payment was made for the order or not. There is a key and participation constraint on the payment table, i.e every Record in the payment table must be associated with some entry in the Order table and Customer cannot combine payments of multiple orders and pay in one payment record. There is no constraint for the Order table with Payments table.

Order Details is associated with the Products table, There is key and participation constraint on the Order details table with the product table because one order detail record can contain exactly one product. There are no such constraints on the product table.

Products are sold by Offices, There is a participation constraint on products, every product is sold by some office and the same product can be sold by multiple offices. There can be offices without any products hence no participation constraint and an office can have multiple products hence there is no key constraint.

An Office can have multiple Employees, hence there is no key constraint and there can be offices with no employees hence no participation constraint. An employee can belong to only one office hence key constraint.

Inventory table has details about what products are available and which offices are selling these products and the quantity available on hand.

c) We curated the data needed for this application and made sure they seemed as realistic as possible.

d) There can be basically two ways of operations: For Offices, For Customers.

Business Rules

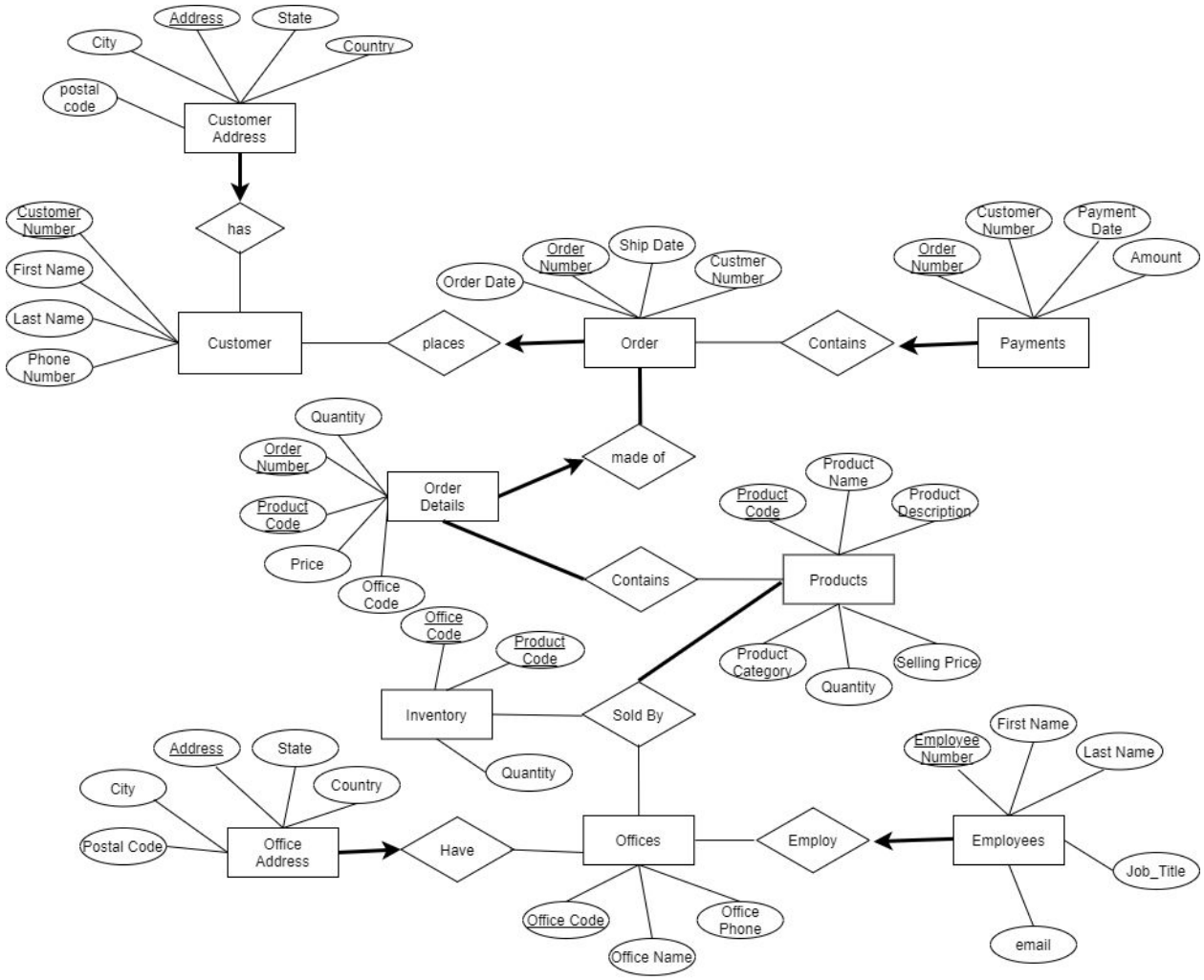
1) For customers in our database, they can select a product available in the inventory across offices and place an order for them.
2) Display Total spending for multiple customers
3) Display Total spending by cities in our DB.
4) View Purchase history for a customer
5) Show top selling products for an office based on the amount spent on them.
6) Show the performance of the offices based on their sales.
7) Contact store and a representative for any queries.

Data Loading Procedure (Port Number: 8581)
Outside the data folder, run the following codes

cat data/offices.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY offices from STDIN CSV HEADER"
cat data/employees.csv | psql --U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY employees from STDIN CSV HEADER"

cat data/products.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY products from STDIN CSV HEADER"
cat data/customers.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY customers from STDIN CSV HEADER"
cat data/orders.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY orders from STDIN CSV HEADER"
cat data/order_details.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY order_details from STDIN CSV HEADER"
cat data/inventory.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY inventory from STDIN CSV HEADER"
cat data/payments.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY payments from STDIN CSV HEADER"
cat data/customer_addresses.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY customer_addresses from STDIN CSV HEADER"
cat data/office_addresses.csv | psql -U kp2492 -d kp2492_project -h localhost -p 5432 -c "COPY office_addresses from STDIN CSV HEADER"

ER Diagram



schema.sql

```
-- drop table if exists Offices;
drop table if exists offices cascade;
create table Offices(
        office_code varchar(64) primary key,
        office_name varchar(128) not null,
        phone varchar not null
);
drop table if exists office_Addresses cascade;
create table Office_addresses(
        office_address varchar(128) primary key,
        city varchar(64),
        stat varchar(128),
        country varchar(128),
        postal_code varchar(32),
        office_code varchar(64) not null,
        foreign key (office_code) references Offices(office_code)
);


drop table if exists employees cascade;
create table Employees(
        employee_number integer primary key,
        lastname varchar(64) not null,
        firstname varchar(64) not null,
        email varchar(64) unique not null,
        office_code varchar(64) not null,
        job_title varchar(64) not null,
        foreign key (office_code) references Offices(office_code)
);

drop table if exists products cascade;
create table Products(
        product_code varchar(64) primary key,
        product_name varchar(64) not null,
        product_categories varchar(64) not null,
        product_description varchar(64) not null,
        selling_price varchar(64) not null
);

drop table if exists customers cascade;
create table Customers(
        customer_number integer primary key,
        firstname varchar(128) not null,
        lastname varchar(128) not null,
        phone varchar(64) unique not null,
        foreign key(sales_representative) references Employees(employee_number)
);

drop table if exists inventory cascade;
create table Inventory(
        product_code varchar(64),
        office_code varchar(64),
        quantity integer not null,
```

```
        primary key(product_code,office_code),
        foreign key(product_code) references Products(product_code),
        foreign key(office_code) references Offices(office_code)
);

drop table if exists customer_addresses cascade;
create table Customer_Addresses(
        customer_address varchar(128) primary key,
        city varchar(64) not null,
        state varchar(128) not null,
        country varchar(128) not null,
        postal_code varchar(32) not null,
        customer_number integer not null,
        foreign key(customer_number) references Customers(customer_number)
);

drop table if exists orders cascade;
create table Orders(
        order_number int primary key,
        order_date date not null,
        shipped_date date not null,
        customer_number integer not null,
        foreign key (customer_number) references Customers(customer_number)
);

drop table if exists order_Details cascade;
create table Order_Details(
        order_number integer,
        product_code varchar(32),
        office_code varchar(64),
        quantity integer not null,
        price decimal not null,
        primary key(order_number,product_code),
        foreign key (order_number) references Orders(order_number),
        foreign key (product_code) references Products(product_code),
        foreign key (office_code) references Offices(office_code)
);

drop table if exists payments cascade;
create table Payments(
        customer_number integer,
        order_number integer,
        payment_date date not null,
        amount decimal not null,
        primary key(order_number),
        foreign key (customer_number) references Customers(customer_number),
        foreign key (order_number) references Orders(order_number)
);
```