University of Arkansas at Little Rock

Designing and Implementing an Application Database System: Online Restaurant Ordering Database System

Charlessia Robinson

College of Engineering and Information Technology

Department of Information Science

IFSC 3330 Current Trends in Database Technology

Dr. Ningning Wu

April 30th, 2020

Overview of Database System Report

The purpose of this report is to provide a complete documentation of the design and implementation of this project. The report demonstrates the use of basic theories of database system and design including using the high-level conceptual models such as ER models for database design, designing a relational database by ER-to-Relational mapping and basic to complex SQL programming. This document provides a detailed analysis of the approach and results of each phase throughout the project. It starts with the description of data requirement of the application database, followed by the conceptual design of the database in ER diagrams, relational schemas in 3NF, SQL codes for implementing the database, and sample queries and their results.

Table of Contents

Database System Objectives

The main purpose of the Online Restaurant Ordering Database System is to help a restaurant develop and manage a company database for day-to-day operations. The system should manage online to-go orders and customer data efficiently and effectively. Overall, the goal of the restaurant database system is to enable the business to grow and continue to meet the needs of all their customers and the customer orders.

- To build and maintain a high-speed and accurate system
- To create, read, update and delete information about customers and orders
- To provide a searchable database for all users who interact with the system
- To increase and maintain customer satisfaction
- To reduce cost of operation and increase profits
- To make communication between different parts of the system more efficient

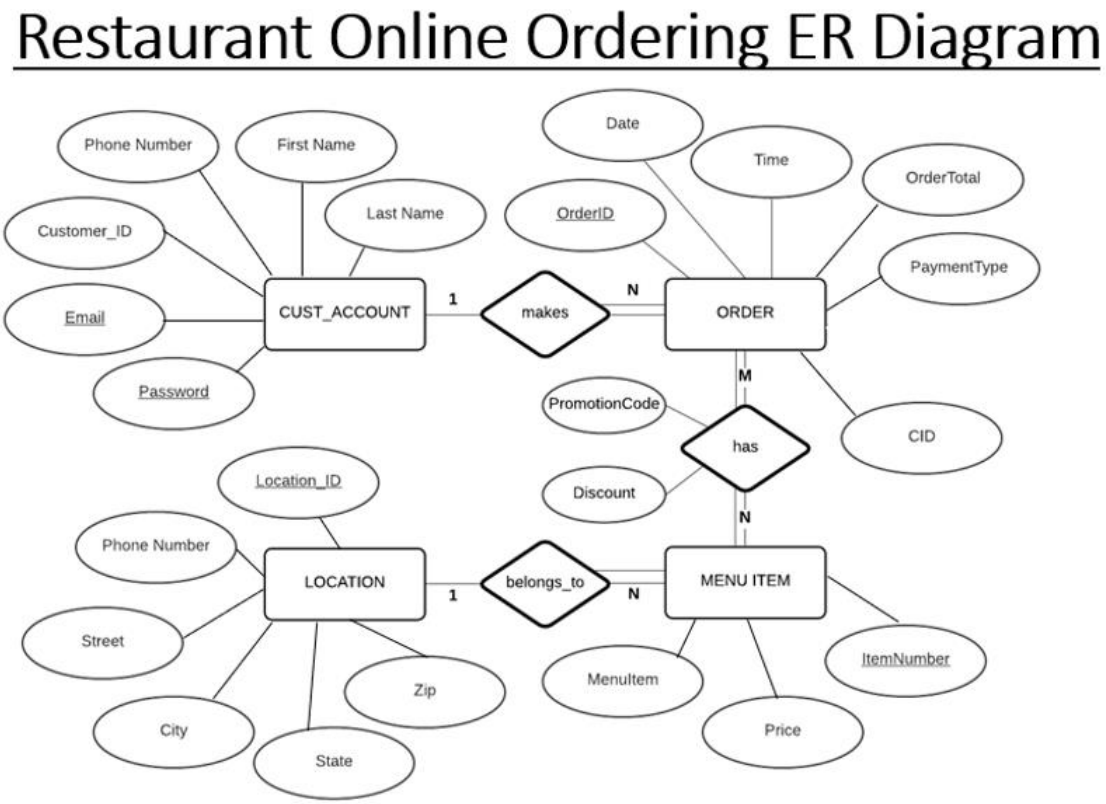
Database Requirements

The online ordering restaurant system is organized into Customer accounts. Each ACCOUNT has a first and last name, phone number and a customer ID which is accessed and available with a customer's email address and password information. Also, each account keeps track of order history. Each LOCATION has an address, phone number, hours of operation for multiple days and times throughout the week. For every ORDER, there is a pickup date, order number, location order id and one or more menu items chosen. For each MENU, there is a menu item name, item number and a price. For every PAYMENT, there is a customer id, a payment type, an order number and an amount. For every PROMOTION, there is a promo name, code, and a discount.

A customer account has multiple restaurant locations to select from. A customer account can place multiple orders. However, there is only one account for each order. For every order, there is only one menu to choose from. A customer account can make multiple payments, but for each payment, there is only one customer account. Also, for every promotion, only one promo code can be applied or redeemed for one online payment.

System Design

Phase 2: *Design ER diagram for a DB application*



Phase 3: *Design relation schemas of the database*

CustomerAccount(Email, Password, CustomerID, Fname, Lname, PhoneNumber)

Order(OrderID, Date, Time, OrderTotal, PaymentType, CID)

MenuItem(ItemNumber, ItemName, Price, Location_ID)

Location(LocationID, Street, City, State, Zipcode, PhoneNumber)

OrderInfo(OID, ItemNum, PromoID)

PromoOrder(Promocode, Discount)

System Implementation

Phase 4: *Create and populate database*

```
create table CustomerAccount(
    Email varchar2(30),
    Customer_Password varchar2(30),
    Customer_Id number(9) unique,
    First_name varchar2(30),
    Last_name varchar2(30),
    Phone_number number(10),
    constraint cust_pk primary key(Email, Customer_Password)
);
```

```
create table CustomerOrder(
    OrderID number(10),
    Customer_date varchar2(15),
    Customer_time varchar2(15),
    Order_total number(6,2),
    Payment_type varchar2(20),
    constraint ord_pk primary key(OrderID),
    constraint fk_customerid foreign key(CID) references CustomerAccount(Customer_Id) on delete cascade
);
```

```
create table MenuItem(
    Item_number number(2),
    Item_name varchar2(20),
    Price number(6,2) check(Price >= 0.0),
    Location_ID number(5),
    constraint menuItem_pk primary key(Item_number),
    constraint locate_fk foreign key(Location_ID) references Location(LocationID) on delete cascade
);
```

```
create table Location(
    LocationID number(5),
    Street varchar2(30),
    City varchar2(30),
    State varchar2(30),
    Zipcode number(5),
    Restaurant_number number(10) unique,
    constraint locate_pk primary key(LocationID)
);
```

```
create table OrderInfo(
    OID number(10),
    ItemNum number(3),
    PromoID number(8),
    constraint ordInfo_pk primary key(OID, ItemNum),
    constraint oid_fk foreign key(OID) references CustomerOrder(OrderID) on delete cascade,
    constraint item_fk foreign key(ItemNum) references MenuItem(Item_number) on delete cascade,
    constraint promo_fk foreign key(PromoID) references PromoCode(Promo_code) on delete cascade
);
```

```
create table PromoCode(
    Promo_code number(8),
    Discount number(2,2),
    constraint promo_pk primary key(Promo_code)
);

INSERT INTO CUSTOMERACCOUNT VALUES
('cxrobinson@yahoo.com', 'ualrInfoScience', 100018349, 'Charlessia', 'Robinson', 5013509899);
INSERT INTO CUSTOMERORDER VALUES
(9888133900, '29-MAR-2020', '6:45:15', 35.25, 'Debit');
INSERT INTO LOCATION VALUES
(10055, 'West Markham', 'Little Rock', 'Arkansas', 72203, 5017348000);
INSERT INTO MENUITEM VALUES
(12, 'Chicken Alfredo', 10.05, 10055);
INSERT INTO PROMOCODE VALUES
(77773300, .15);
INSERT INTO ORDERINFO VALUES
(9888133900, 12, 77773300);
```

Phase 5: *Write application program*

```
SELECT item_number, item_name
FROM MENUITEM, LOCATION
WHERE menuitem.location_id = location.locationid and location.locationid = 10050;

SELECT first_name, last_name
FROM CUSTOMERACCOUNT
WHERE customer_id in (SELECT CID FROM CUSTOMERORDER Group By CID Having count(*) > 2);

SELECT location.street, location.restaurant_number
FROM LOCATION
WHERE location.city LIKE '%Little Rock';

SELECT cid
FROM CUSTOMERORDER
ORDER BY customer_date;

SELECT min(order_total), max(order_total), avg(order_total)
FROM CUSTOMERORDER, ORDERINFO, MENUITEM
WHERE customerorder.orderid = orderinfo.oid and menuitem.item_number = orderinfo.itemnum
and menuitem.location_id = 10055;

SELECT orderinfo.oid, customerorder.order_total, promocode.discount
FROM PROMOCODE, ORDERINFO, CUSTOMERORDER
WHERE promocode.promo_code = orderinfo.promoid and customerorder.orderid = orderinfo.oid;
```

```
SELECT street, city, restaurant_number
FROM LOCATION, MENUITEM
WHERE location.locationid = menuitem.location_id and menuitem.item_name = 'Chicken Alfredo';

SELECT customer_id, sum(order_total)
FROM CUSTOMERORDER, CUSTOMERACCOUNT
WHERE customerorder.cid = customeraccount.customer_id
GROUP BY customer_id
ORDER BY customer_id ASC;

SELECT count(*)
FROM CUSTOMERORDER, ORDERINFO, MENUITEM
WHERE customerorder.orderid = orderinfo.oid and orderinfo.itemnum = menuitem.item_number
and menuitem.item_name LIKE '%Pizza';

create or replace procedure change_customerPassword(custEmail in customeraccount.email%type,
newPassword in customeraccount.customer_password%type) as
begin
update customeraccount
set customeraccount.customer_password = newPassword
where customeraccount.email = custEmail;
end;

BEGIN
change_customerPassword('calebjohnson67@gmail.com', 'caleblovestacos');
END;

create or replace procedure customorOrderInfo(customID in customerorder.cid%type,
orderCount out number, totalSpent out customerorder.order_total%type) as
invalid_customID EXCEPTION;
Begin
    if customID is null then
    raise invalid_customID;
    end if;
    select count(*), sum(customerorder.order_total)
        into orderCount, totalSpent
    from customerorder
    where customerorder.cid = customID;
    EXCEPTION
    when invalid_customID then
        raise_application_error(-20002, 'Customer ID cannot be NULL');
    when NO_DATA_FOUND then
        raise_application_error(-20001, 'No order matches Customer ID');
End;
```

```
Declare
custOrders number;
custTotalAmount number(6,2);
begin
customerOrderInfo(100018845, custOrders, custTotalAmount);
dbms_output.put_line(custOrders || custTotalAmount);
end;
```

Limitations

Throughout the scope of this project, there were a few anticipated and unanticipated challenges that surfaced in the design and implementation phases. These constraints influenced the approach and results of the database system. Most of these occurrences happened primarily in design. When it came time to gather data requirements to model these real-world entities for this application, I was aware of my prior knowledge and exposure interacting with similar systems. Therefore, I was confident in my ability to design this particular database. As I have been a user of a restaurant database, I was able to collect and analyze the needs of a customer including the restaurant operation. As this may suggest, the scope of the design process is significantly limited to my own perspective and experience. However, I did take the time to think critically about the system from a large and varied group of users. For instance, a restaurant database system needs to consider how this will impact not just the customer, but also the owner, the workers, investors and the restaurant business strategy.

As I mentioned, the design process showed weaknesses in my database from the ER diagrams to the mapping of relational schemas. The process revealed some revision to redundant data attributes in the model and connecting the data with its relationships. There was some trial and error when it came time to describe the relationship types and their participation constraints. As previously learned, the primary keys needed to be unique identifiers for multiple objects of that specific type. The foreign keys were also necessary to refer the table dataset to another table dataset to make the database applicable in the business. The implementation process had less roadblocks to get pass due to the amount of time spent mapping the data. However, there were some issues with datatypes such as getting the time and date which caused a delay trying to implement. As expected, there were some errors using Oracle that I had to fix. There was one problem that I noticed in implementation which was between the order information and the promocode. For every order, not all customers will be applying a promocode to their order. So, it could lead to multiple null values for many tuples in the dataset.

Conclusion

Overall, the database requirements detailed in the description complied with the design and implementation. The main goal of this database system was to build a working system that would meet the needs of customers and the restaurant. Other goals for this application included the ability to insert and extract specified data for all end users in case-by-case scenarios. The project presented challenges in following the standard approach for building a relational database. These problems offered a learning curve for my knowledge of databases. However, I do not believe that these problems hindered the progress of the system. It only enhanced its usability. In final, these goals were accomplished in a timely manner with limited constraints. The Online Restaurant Ordering Database System is fully capable to support the growth of a restaurant operation.