Olive Garden Database

Final Report

BADM 352 - Database Design and Management

Group 7

Ababacar Diagne, Amanda DuVall, Bhumi Gutpa, Megha Patel

**Section 1: Context and Business Background Information**

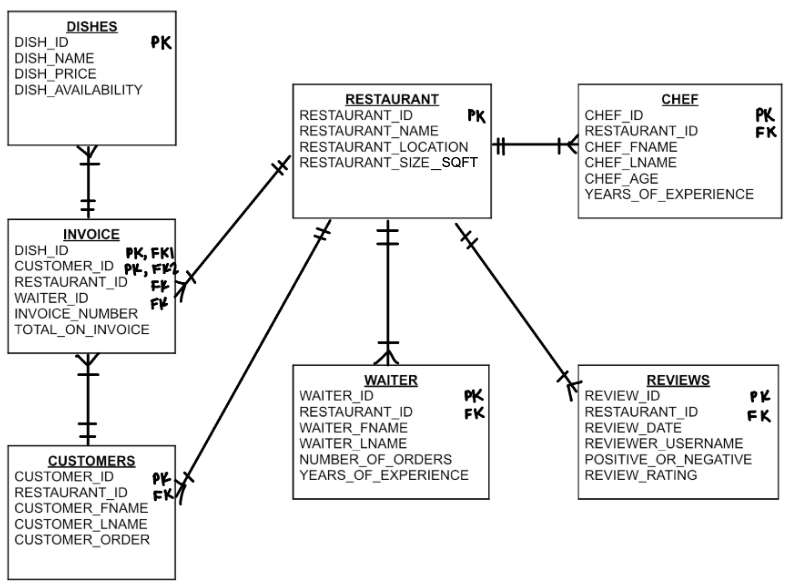
The business of this database is based on the Olive Garden restaurants. It was founded in Orlando, Florida in 1932. The restaurant business has 145 locations, with a presence of 850 locations worldwide. It provides a wide range of cuisines based on it’s locations and communities.

Olive Garden is an American casual dining restaurant chain that specializes in Italian-American cuisine. The business idea is to provide customers with a comfortable, family-oriented atmosphere and to serve affordable Italian-American food. The competitive advantage of Olive Garden is value pricing, menu variety, and consistency.

**Section 2: Business Rules**

We assume that every restaurant within the Olive Garden chain has the same menu; therefore, every restaurant has the same type of dishes. One invoice can belong to only one restaurant, and one restaurant can have many invoices. One restaurant can have many customers, and one customer can belong to one restaurant. One waiter can work at one restaurant, and one restaurant can employ many waiters. One restaurant can have many reviews, and one review can belong to only one restaurant. One chef can work at one restaurant, and one restaurant can employ many chefs.

**Section 3: Relational Model and Underlying Assumptions**

****

The relational model consists of six tables, including dishes, customers, restaurant, waiter, chef, and reviews. There is one junction table, which is the invoice table.

**Section 4: Database Tables**

**Chef Table**

****

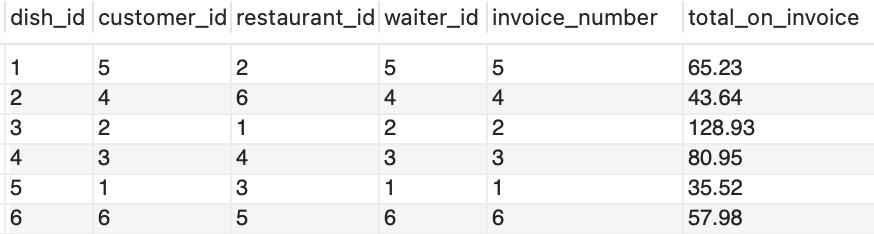
**Customers Table**

****

**Dishes Table**

****

**Invoice Table**

****

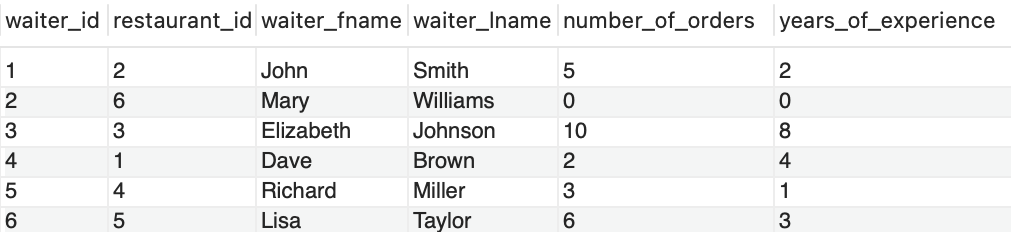
**Restaurant Table**

****

**Reviews Table**

****

**Waiter Table**

****

**Section 5: Code for Creation of Tables and Insertion of Data**

**Creation:**

**Database**

CREATE DATABASE megha3\_restaurant;

**Chef Table**

CREATE TABLE chef(

chef\_id int,

restaurant\_id int,

chef\_fname varchar(20),

chef\_lname varchar(20),

chef\_age int,

years\_of\_experience int,

PRIMARY KEY (chef\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant (restaurant\_id)

);

**Customers Table**

CREATE TABLE customers(

Ccustomer\_id int,

Rrestaurant\_id int,

Ccustomer\_fname varchar(20),

Ccustome\_lname varchar(20),

Ccustomer\_order varchar(30),

PRIMARY KEY (customer\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant (restaurant\_id)

);

**Dishes Table**

CREATE TABLE dishes(

dish\_id int,

dish\_name varchar(30),

dish\_price decimal(10,2),

dish\_availability varchar(10),

PRIMARY KEY (dish\_id)

);

**Invoice Table**

CREATE TABLE invoice(

dish\_id int,

customer\_id int,

restaurant\_id int,

waiter\_id int,

invoice\_number int,

total\_on\_invoice decimal(10,2),

PRIMARY KEY (dish\_id, customer\_id),

FOREIGN KEY (dish\_id) REFERENCES dishes (dish\_id),

FOREIGN KEY (customer\_id) REFERENCES customers (customer\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant (restaurant\_id),

FOREIGN KEY (waiter\_id) REFERENCES waiter (waiter\_id)

);

**Restaurant Table**

CREATE TABLE restaurant(

restaurant\_id int,

restaurant\_name varchar(15),

restaurant\_location varchar(20),

restaurant\_size\_sqft int,

PRIMARY KEY (restaurant\_id)

);

**Reviews Table**

CREATE TABLE reviews(

review\_id int,

restaurant\_id int,

review\_date date,

reviewer\_username varchar(20),

positive\_or\_negatiive varchar(12),

review\_ratinig int,

PRIMARY KEY (review\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant (restaurant\_id)

);

**Waiter Table**

CREATE TABLE waiter(

waiter\_id int,

restaurant\_id int,

waiter\_fname varchar(20),

waiter\_lname varchar(20),

number\_of\_orders int,

years\_of\_experience int,

PRIMARY KEY (waiter\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant (restaurant\_id)

);

**Insertion:**

**Chef Table**

INSERT INTO chef VALUES (01, 05, "Alex", "Garcia", 29, 3);

INSERT INTO chef VALUES (02, 03, "Michael", "Jones", 42, 15);

INSERT INTO chef VALUES (03, 01, "Daniel", "Davis", 35, 10);

INSERT INTO chef VALUES (04, 02, "Nancy", "Moore", 28, 4);

INSERT INTO chef VALUES (05, 06, "Andrew", "Martin", 32, 1);

INSERT INTO chef VALUES (06, 04, "Ashley", "Thomas", 23, 0);

**Customers Table**

INSERT INTO customers VALUES(01, 02, "Emily", "Rodriguez", "Lasagne alla

Bolognese");

INSERT INTO customers VALUES(02, 01, "Jacob", "Kim", "Fettuccine al Pomodoro");

INSERT INTO customers VALUES(03, 04, "Ava", "Patel", "Gnocchi di Patate");

INSERT INTO customers VALUES(04, 06, "Ethan", "Nguyen", "Pollo alla Caccicatora" );

INSERT INTO customers VALUES(05, 02, "Olivia", "Garcia", "Pizza Margherita");

INSERT INTO customers VALUES(06, 05, "William", "Smith", "Viello Tonnato");

**Dishes Table**

INSERT INTO dishes VALUES(01, "Lasagne alla Bolognese", 12.00, "Available");

INSERT INTO dishes VALUES(02, "Fettuccine al Pomodoro", 12.00, "Available");

INSERT INTO dishes VALUES(03, "Gnocchi di Patate", 12.00, "Available");

INSERT INTO dishes VALUES(04, "Pollo alla Caccicatora", 12.00, "Available");

INSERT INTO dishes VALUES(05, "Pizza Margherita", 12, "Available");

INSERT INTO dishes VALUES(06, "Viello Tonnato", 12.00, "Available");

**Invoice Table**

INSERT INTO invoice VALUES (05, 01, 03, 01, 01, 35.52);

INSERT INTO invoice VALUES (03, 02, 01, 02, 02, 128.93);

INSERT INTO invoice VALUES (04, 03, 04, 03, 03, 80.95);

INSERT INTO invoice VALUES (02, 04, 06, 04, 04, 43.64);

INSERT INTO invoice VALUES (01, 05, 02, 05, 05, 65.23);

INSERT INTO invoice VALUES (06, 06, 05, 06, 06, 57.98);

**Restaurant Table**

INSERT INTO restaurant VALUES (01, "Olive Garden1", "Champaign", 225);

INSERT INTO restaurant VALUES (02, "Olive Garden2", "Urbana", 225);

INSERT INTO restaurant VALUES (03, "Olive Garden3", "Chicago", 400);

INSERT INTO restaurant VALUES (04, "Olive Garden4", "Evanston", 325);

INSERT INTO restaurant VALUES (05, "Olive Garden5", "Bloomington", 225);

INSERT INTO restaurant VALUES (06, "Olive Garden6", "Peoria", 225);

**Reviews Table**

INSERT INTO reviews VALUES (01, 01, '2001-04-23', "Lisa\_Martin", "positive", 4);

INSERT INTO reviews VALUES (02, 02, '2005-08-04', "John\_Chris", "positive", 5);

INSERT INTO reviews VALUES (03, 03, '2022-03-29', "Mary\_Adam", "negative", 2);

INSERT INTO reviews VALUES (04, 04, '2020-01-30', "Christina\_Reynolds", "negative",

3);

INSERT INTO reviews VALUES (05, 05, '2018-05-13', "Bree\_David", "positive", 4);

INSERT INTO reviews VALUES (06, 06, '2015-07-05', "Gabi\_Stevenson", "negative", 1);

**Waiter Table**

INSERT INTO waiter VALUES (01, 02, "John", "Smith", 5, 2);

INSERT INTO waiter VALUES (02, 06, "Mary", "Williams", 0, 0);

INSERT INTO waiter VALUES (03, 03, "Elizabeth", "Johnson", 10, 8);

INSERT INTO waiter VALUES (04, 01, "Dave", "Brown", 2, 4);

INSERT INTO waiter VALUES (05, 04, "Richard", "Miller", 3, 1);

INSERT INTO waiter VALUES (06, 05, "Lisa", "Taylor", 6, 3);

**Section 6: Queries and Results**

1. **List the oldest and the youngest chef that work at Olive Garden.**

SELECT \* FROM chef

WHERE chef\_age = (SELECT MAX(chef\_age) FROM chef)

OR chef\_age = (SELECT MIN(chef\_age) FROM chef);

1. **List all customers and their waiters for their time at Olive Garden.**

SELECT W.waiter\_id, W.waiter\_fname, W.waiter\_lname, C.customer\_id, C.customer\_fname, C.customer\_lname

FROM waiter as W INNER JOIN customers as C

ON W.waiter\_id = C.customer\_id;

1. **List all customers, their dishes, and the total invoice of their order.**

SELECT C.customer\_id, C.customer\_fname, C.customer\_lname, D.dish\_id, D.dish\_name, I.invoice\_number, I.total\_on\_invoice

FROM customers as C INNER JOIN dishes as D INNER JOIN invoice as I

ON C.customer\_id = D.dish\_id AND D.dish\_id = I.invoice\_number;

1. **List the Chef’s id, first/last name, years of experience, and the restaurant id.**

SELECT chef\_id, chef\_fname, chef\_lname, years\_of\_experience, restaurant\_id

FROM chef;

1. **List all of the customer, waiters, and chefs who’s last name contains the letter A.**

SELECT customer\_lname

FROM customers

WHERE customer\_lname LIKE '%A%'

UNION

SELECT waiter\_lname

FROM waiter

WHERE waiter\_lname LIKE '%A%'

UNION

SELECT chef\_lname

FROM chef

WHERE chef\_lname LIKE '%A%';

1. **List all of the reviews, review dates, review rating and reviewer usernames.**

SELECT review\_id, review\_date, review\_rating,reviewer\_username

FROM reviews;

1. **List all from chef and order by last name.**

SELECT \*

FROM chef

ORDER BY chef\_lname;

1. **List all of waiters first/last name, years of experience, number of orders, and their id.**

SELECT waiter\_id, waiter\_fname, waiter\_lname, years\_of\_experience, number\_of\_orders

FROM waiter;

1. **Count the number invoices from the restaurant.**

SELECT COUNT(invoice\_number)

FROM invoice;

1. **List the invoice totals that are more than 50.**

SELECT total\_on\_invoice

FROM invoice

WHERE total\_on\_invoice > 50;

**Section 7: Limitations and Scope for Extension**

**Limitations**

The limitations consist of three. First, is that it lacks data values, there is lack of depth in reviewing tables, and the ids could be more unique. A lack of data values is unrealistic. Additionally, with little data, insights can only be obtained at surface level. Which means that with a larger amount, conclusions of benefit/certainty can be derived. The lack of depth in the review table poses two issues. The first is that there is lack of variety in the reviewing tables. This means that the customer should be asked about their experience with timing, taste of meal, price satisfaction, etc. Hence, such questions could be considered as columns. The second issue is that the current model’s values are vague. The column available have the options of 1-5, positive, or negative. Such values make it difficult to interpret a tangible improvement of the restaurant. The final limitation is the lack of uniqueness of the ids. The ids for the customers, waiters, and chefs are all single digits. The ids are also numerically the same for different entities, though they do not represent the same meanings.

**Scope for Extension**

In terms of extensions we could add 3 entities, have maintenance protocols or systems, update/add the review table, and to improve the ids. The entities that could be added are menus, inventory, and suppliers. The menus could consist of many dishes. As for inventory, the ingredients to are apart of the dishes would be stored. The inventory would be one per restaurant. However, the supplier entity would be many per inventory. Though, ideally there would be a specific set of companies that consistently provide the goods and services. The review table could be adjusted in two ways. The first could be to add columns that have specific questions. Secondly, customers would be provided values that are simple and reasonable to answer the questions. The ids could be complexified through a few principles. The principles could be that an id must have sixteen character, with special characters, numbers and letters. This will prevent confusion while working with the database and even within the restaurant.