JUNK

RELATIONAL DATABASE FOR A FAST FOOD RESTAURANT

SQL II FINAL PROJECT

Prepared by :
Ziyad Albaadi, Emilio Capitan, Marilyn El Kassis
Sami Boustani, Pablo Ferraro, Carlota Perez

TABLE OF CONTENT

ABOUT JUNK BURGER

RELEVANT INFORMATION FOR OUR DATABASE

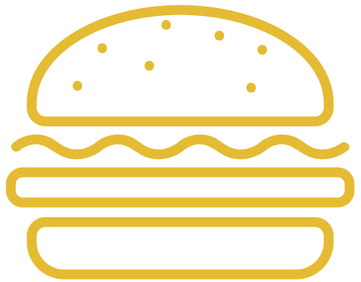
5 Restaurants in Madrid

5 Product Categories Different Menu Options

Many Customers +40 Employees

ASSUMPTIONS

TO BUILD OUR MODEL

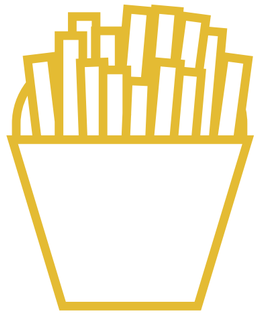


Branch-Specific Data

Each restaurant branch has unique identifiers, and operational data like orders and employees are linked to specific branches.

Customer-Centric Design

The database tracks customer interactions across all branches, facilitating insights into customer preferences and behavior.



Structured Menu Catalog

The database assumes a well-defined menu where each item is categorized (e.g., burgers, fries), aiding in straightforward inventory management and menu planning.

ASSUMPTIONS

TO BUILD OUR MODEL

Geographical Customer Insights

By linking customers to specific locations, the database is designed to provide insights into customer distribution and popular areas, aiding in targeted marketing strategies

Employee Management

The inclusion of employee data tied to specific store locations allows for efficient staff management and deployment based on store needs and locations.

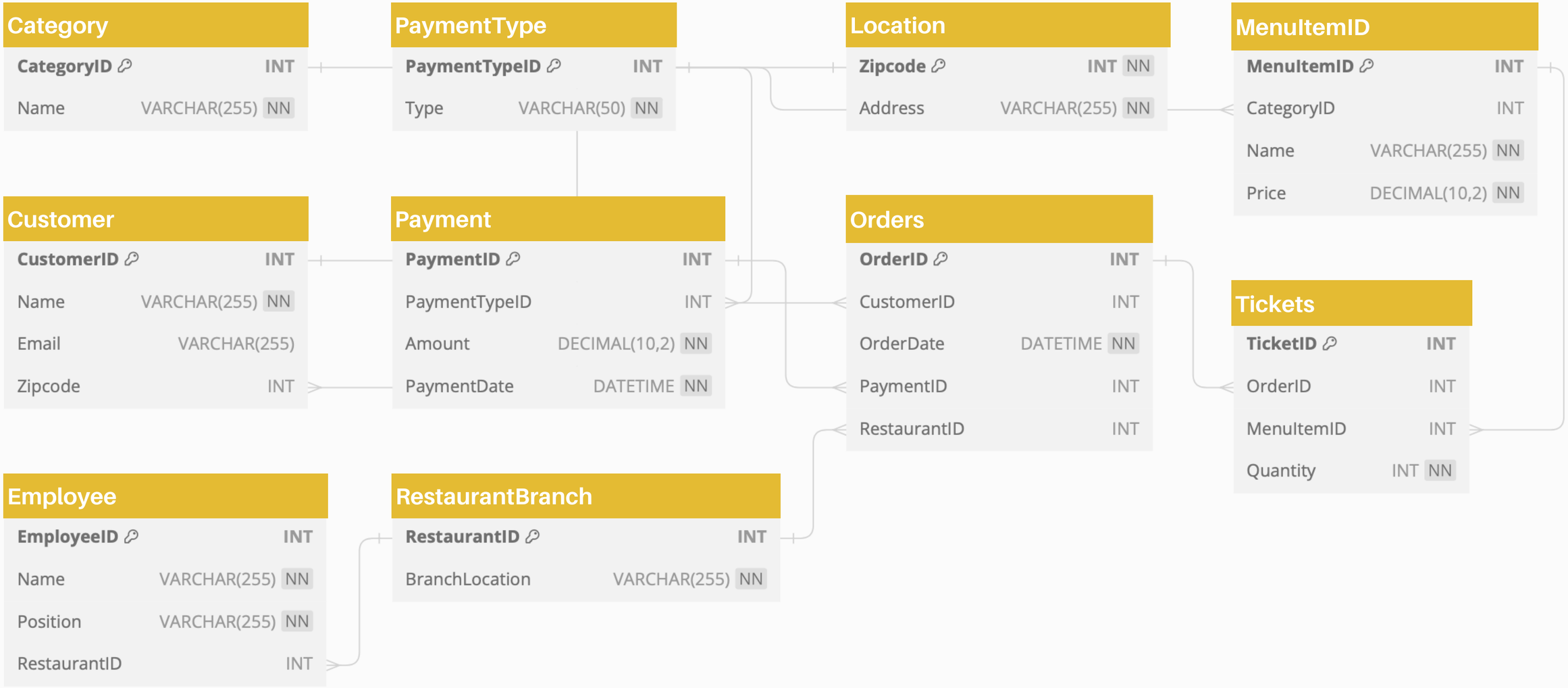
Order Tracking

The database tracks each order's details, including the items ordered and the associated payments, enabling a clear view of sales trends and customer preferences.

3NF COMPLIANCE

- **Elimination of Redundancy:** By segregating data into distinct tables (e.g., separating menu items, orders, and customer information), the database avoids unnecessary duplication of data. This structure ensures that updates to one piece of data (like a customer's email address) only need to be made in one place.
- **Dependency on Primary Keys:** Each table is constructed such that every non-key attribute is fully functionally dependent on the primary key. For instance, in the `Customer` table, customer details like name and email depend solely on the `CustomerID`.
- **Removal of Transitive Dependency:** The database design meticulously avoids transitive dependencies, where non-key attributes depend on other non-key attributes. This is achieved by properly segmenting data across tables and establishing direct relationships. For example, order details are directly linked to specific orders rather than indirectly through customers.

Entity Relation Diagram



https://dbdiagram.io/d/Junk-Burger-656e2c2b56d8064ca05deb94

DATA INSERTION

RESTAURANT BRANCH

The five restaurants in Madrid have been incorporated along with their respective location and zip code.

CATEGORIES & MENU ITEMS

The product categories and the various menu options, along with their original menu prices, have been included.

PAYMENT INFORMATION

In the payment table, authentic payment options have been simulated, incorporating a variety of products from the menu.

TICKETS

The order records, menu items and quantity of products have been entered in the ticket table, all corresponding to the simulated orders in the reference tables.

CUSTOMERS & EMPLOYEES

Customer information has been incorporated, simulating authentic orders, as well as details about employees and their respective positions within the restaurants.

*Queries for inserted data can be seen in **insert_data.sql**

TOP 3 FAVORITE DISHES

NAME	TOTALSOLD
LE GAGNANT	9
JACKI BBQ	8
TARTA DE QUESO	8

```
SELECT MI.Name, SUM(OD.Quantity) AS TotalSold
FROM Ticket OD
JOIN MenuItem MI ON OD.MenuItemID = MI.MenuItemID
GROUP BY MI.Name
ORDER BY TotalSold DESC
LIMIT 3;
```

*Queries can be found in **analytics.sql**

REVENUE PER TYPE OF CREDIT CARD

TYPE	TOTALREVENUE
VISA	251.1
MASTERCARD	192.18
AMEX	30.4

```
SELECT PT.Type, SUM(P.Amount) AS TotalRevenue
FROM Payment P
JOIN PAYMENTTYPE PT ON P.PaymentTypeID = PT.PaymentTypeID
WHERE PT.Type <> 'Cash'
GROUP BY PT.Type
ORDER BY TotalRevenue DESC;
```

*Queries can be found in **analytics.sql**

AVERAGE AMOUNT SPENT ON EACH PURCHASE

AVERAGESPENDING
24.48

```
SELECT AVG(P.Amount) AS AverageSpending
FROM Payment P;
```

*Queries can be found in **analytics.sql**

CUSTOMER DISTRICT/AREA

```
SELECT L.Address, COUNT(*) AS
NumberOfCustomers
FROM Customer C
JOIN Location L ON C.Zipcode = L.Zipcode
GROUP BY L.Address
ORDER BY NumberOfCustomers DESC;
```

ADDRESS	NUMBEROFCUSTOMERS
Abascal	4
El Retiro	4
El Viso	4
Gran Via	4
Salamanca	4

*Queries can be found in **analytics.sql**

REPEATED CUSTOMERS

```
SELECT C.CustomerID, C.Name, C.Email,
COUNT(O.OrderID) AS NumberOfOrders
FROM Orders O
JOIN Customer C ON O.CustomerID = C.CustomerID
GROUP BY C.CustomerID, C.Name, C.Email
HAVING COUNT(O.OrderID) > 1
ORDER BY Numberoforders DESC;
```

*Queries can be found in **analytics.sql**

CUSTOMERID	NAME	EMAIL	NORDERS
1	Ziyad Albaadi	ziyad@albaadi.com	2
2	Carlota Pérez	Carlota@Perez.com	2
3	Marilyn ElKassis	Marilyn@ElKassis.com	2
4	Sami Boustani	Sami@Boustani.com	2
5	Emilio Capitan	Emilio@Capitan.com	2
6	Pablo Ferraro	Pablo@Ferraro.com	2
7	Olivia Harris	olivia@example.com	2

MOST PROFITABLE BRANCH

```
SELECT RestaurantBranch.RestaurantID, BranchLocation,
SUM(Payment.Amount) AS TotalRevenue
FROM Orders
JOIN Payment ON Orders.PaymentID = Payment.PaymentID
JOIN RestaurantBranch ON Orders.RestaurantID =
RestaurantBranch.RestaurantID
GROUP BY RestaurantBranch.RestaurantID, BranchLocation
ORDER BY TotalRevenue DESC;
```

*Queries can be found in **analytics.sql**

RESTAURANTID	BRANCHLOCATION	TOTALREVENUE
5	Penalver	184.5
1	Abascal	145.7
4	Doctor Cortezo	141.28
3	Princesa	126.1
2	Gta. de Bilbao	63.35

MOST SOLD ITEM 'QUANTITY'

MENUITEMID	NAME	TOTAQSOLD
1	LE GAGNANT	9
3	JACKIE BBQ	8
4	Tarta de Queso	8
14	MOSTAZA HEINZ	7
2	TIMELESS	6
5	Founders All Day IPA	6

```
SELECT MI.MenuItemID, MI.Name, SUM(OD.Quantity) AS
TotalQuantitySold
FROM TICKET OD
JOIN MenuItem MI ON OD.MenuItemID = MI.MenuItemID
GROUP BY MI.MenuItemID, MI.Name
ORDER BY TotalQuantitySold DESC;
```

*Queries can be found in **analytics.sql**

ITEMS THAT PROVIDE HIGHER REVENUE

MENUITEMID	NAME	TOTAQSOLD	TREVENUE
1	LE GAGNANT	9	125.1
3	JACKIE BBQ	8	107.2
2	TIMELESS	6	83.4
4	Tarta de Queso	8	55.6
5	Founders All Day IPA	6	27
17	JUNK FRIES	5	17
7	Tercio Mahou Maestra	4	14
18	JUNK CAJUN FRIES	4	13.6

```
SELECT MI.MenuItemID, MI.Name,
SUM(OD.Quantity) AS TotalQuantitySold,SUM(OD.Quantity * MI.Price) AS TotalRevenue
FROM Ticket OD
JOIN MenuItem MI ON OD.MenuItemID = MI.MenuItemID
GROUP BY MI.MenuItemID, MI.Name
ORDER BY TotalRevenue DESC;
```

CONCLUSION

The designed database for the fast-food restaurant exemplifies a highly efficient and structured system, crucial for informed business decision-making. Adhering to the Third Normal Form (3NF), it guarantees data integrity, minimizes redundancy, and ensures efficient data access. This database adeptly captures essential aspects of the business, including menu offerings, customer transactions, and sales trends, facilitating insights into customer preferences and operational performance.

Its scalability and adaptability make it a valuable asset for the restaurant's growth, enabling data- driven strategies and optimizing operations. In essence, this database is a cornerstone for enhancing customer satisfaction and driving business success in a competitive market.