McDonalds System

System Concept

An online booking system for the customers. The booking process requires customers to start by setting the date and time of coming to the restaurant from the saved restaurant operating hours. They then choose what to order from the menu list, the list is stored in the database and its price information, so that the customers can see an invoice with the total amount of money that they need to pay.

Restaurant Operations

The McDonald restaurant operations are run by its staff who make sure a customer order is fulfilled. If I leave out those staff members who are not actively involved in the process of day-to-day order fulfilment, there is the head chef who will plan the meals and directs the cooking staff, a line cook who prepares the ordered meals, and/or someone who makes sure the food get to the waiters.

The waiters serve the patrons armed with knowledge of the customer’s name and with how many guests did the customer come with. With this knowledge the waiter can lead the patrons to a table that will better accommodate them, waiting time is significantly reduced to almost zero.

A table is served by one waiter at a time so as to not confuse the patrons, but a waiter can serve one or more customer tables. When finished dining, the customer is presented with an invoice by the serving waiter, one invoice reflects the items that the customer ordered from the menu, their quantities, unit prices, and the overall total of a single order.

Business Rules

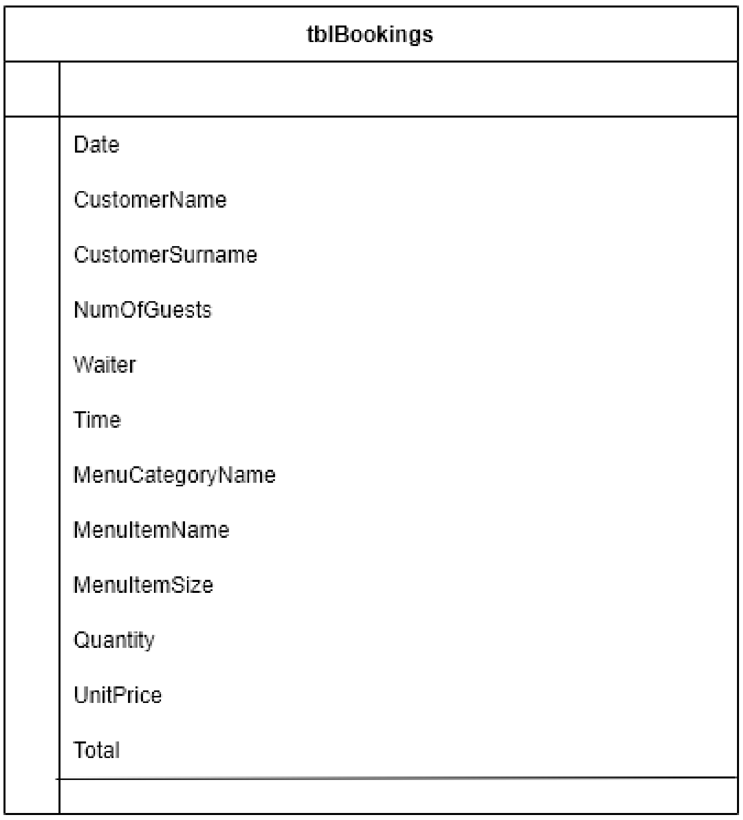
- A customer may make a booking that has a specific time and date of fulfilment.
- Each booking at a specific date and time that is compliant with the restaurant’s operating hours is associated with one customer
- Each customer’s booking is handled by a single waiter, one or more bookings may be handled by one waiter.

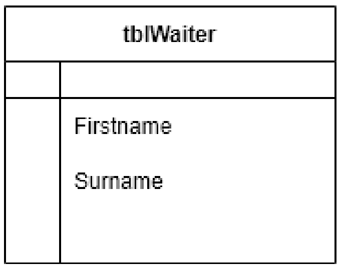
- Each booking generates an invoice that is to be prepared by the waiter for the customer.
- One invoice shows a list of lines of all the ordered menu items, their quantity, unit prices, and total cost of the ordered items.
- Each invoice line references a single menu item from the menu and each menu item may be found in zero or many invoice lines.

Database Structural Design

Normalization:

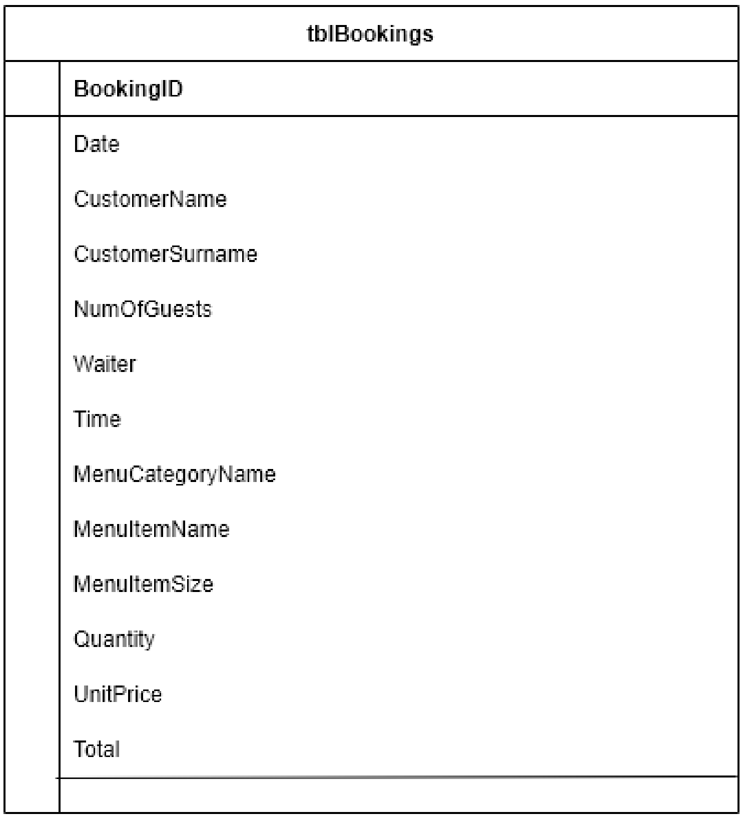
From the above explanation of business processes and business rules, entities and their relationships can be extracted. The following is our initial entities, a booking table that stores all the information required to fulfil an order transaction and waiter table to store waiter information:



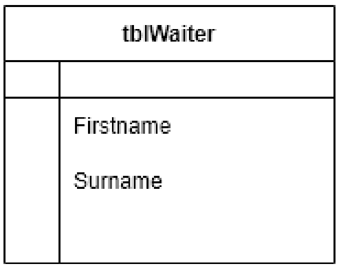


1ST Normal Form

The original tables have no identifying columns, an indicator that there needs to be table normalisation, a process with at most four normal forms. For the bookings and waiter tables to satisfy the first normal form they must have primary keys to uniquely identify each table row, and as can be observed there are no repeating columns containing same kind of data, and all columns contain a single value.



	tblBookings
	BookingID
	Date
	CustomerName
	CustomerSurname
	NumOfGuests
	Waiter
	Time
	MenuCategoryName
	MenuItemName
	MenuItemSize
	Quantity
	UnitPrice
	Total



	tblWaiter
	Firstname
	Surname

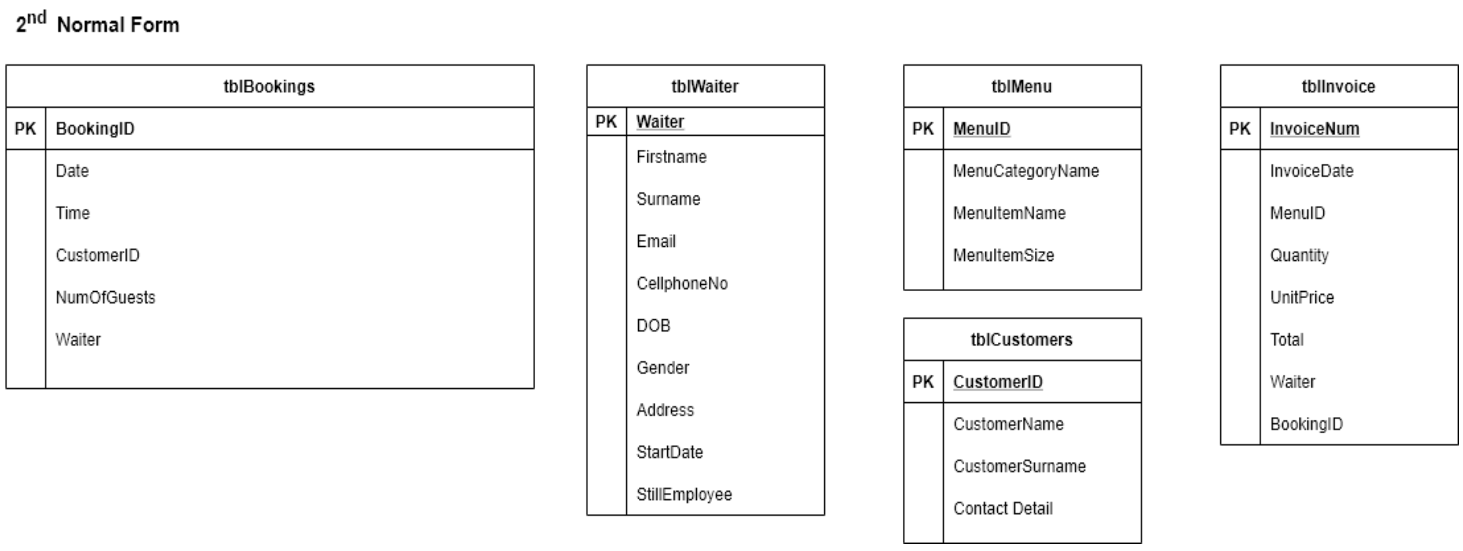
2nd Normal Form

Both tables have primary keys and satisfy the first normal form which deals with duplicate data (or redundancies) across multiple columns, but the bookings table requires further normalisation as there are redundancies across multiple rows.

In the waiter table more columns are added to store actionable information about the waiter as an employee.

To achieve Second Normal Form, the tables must be in First Normal Form, of which they are, the next step is to identify rows whose data repeats in different places and then remove them to their own table. A look at the bookings table and it is clear that customer name and surname will repeat, so will the menu option details.

Upon normalising our table as per second normal form standards, the following tables are defined:



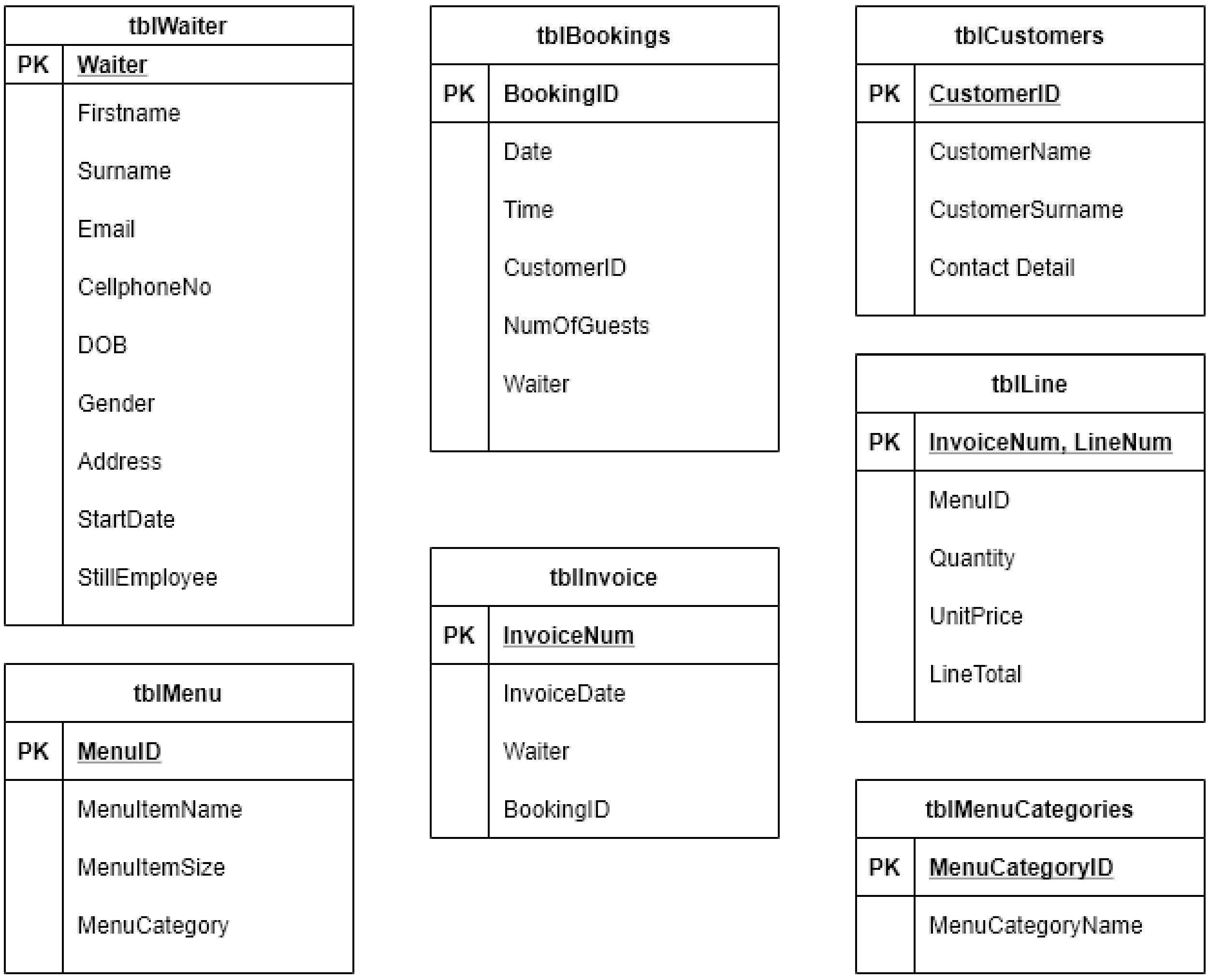
The above tables do not have repeating rows except the invoice and the menu tables which can be further normalized in the third normal form.

3rd Normal Form

This normal form requires that data that is not directly dependent on the primary key but is dependent on another value in the table should be moved with that value according to the dependency.

Important to notice:

- The removal of the Invoice total column from tblInvoice to tblLine but renamed as LineTotal, the change in naming is necessitated by the entity name.
- The new relationship suggested by table menu and table menu categories.



tblWaiter	
PK	Waiter
	Firstname
	Surname
	Email
	CellphoneNo
	DOB
	Gender
	Address
	StartDate
	StillEmployee

tblBookings	
PK	BookingID
	Date
	Time
	CustomerID
	NumOfGuests
	Waiter

tblCustomers	
PK	CustomerID
	CustomerName
	CustomerSurname
	Contact Detail

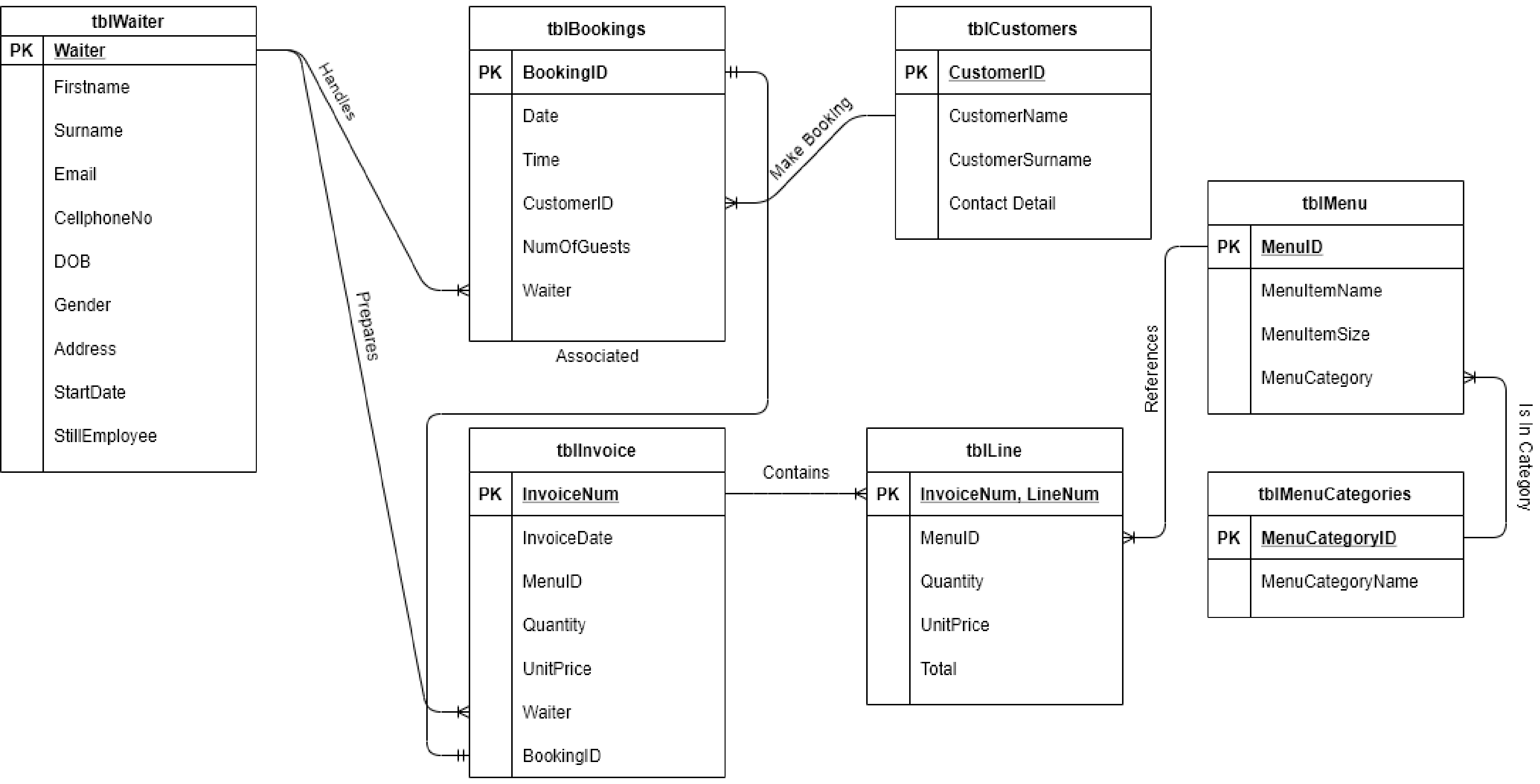
tblLine	
PK	InvoiceNum, LineNum
	MenuID
	Quantity
	UnitPrice
	LineTotal

tblInvoice	
PK	InvoiceNum
	InvoiceDate
	Waiter
	BookingID

tblMenu	
PK	MenuID
	MenuItemName
	MenuItemSize
	MenuCategory

tblMenuCategories	
PK	MenuCategoryID
	MenuCategoryName

Entity Relationship Diagram

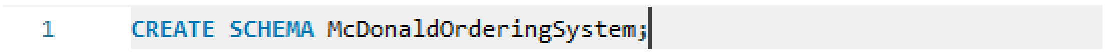


- A One-to-Many relationship exists between the waiter and the bookings entities: One waiter may handle one or many bookings, and one or many bookings may be handled by a single waiter.
- A One-to-Many relationship exists between entity waiter and entity invoice: One waiter may prepare one or many invoices, and one or many invoices may be prepared by a single waiter.
- A One-to-Many relationship exists between the customers table and the bookings table: One customer may make one or many bookings, and many or one booking may be associated with one customer.
- A One-to-One relationship exists between the invoice entity and the bookings entity: A single invoice is associated with only one booking, and one booking is associated with only one invoice.
- A One-to-Many relationship exists between table invoice and table line: One invoice may contain one or many invoice lines, and many or one line may be contained in an invoice. Since table line is using a surrogate key, this means line numbers will be unique and invoice numbers on multiple lines may reference a single invoice number.

- A One-to-Many relationship exists between table menu and table line: One menu item may be ordered one or many times, and one or many invoice lines may reference a single menu item.
- A One-to-Many relationship exists between table menu categories and table menu: one menu category may appear one or many times in the table menu, and one or many menu items may belong under one menu category.

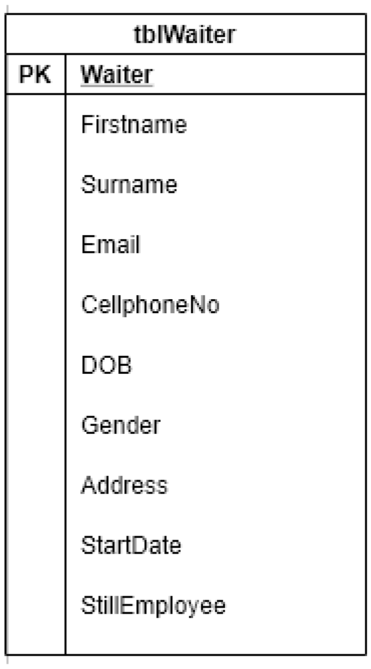
Database Structural Implementation

- Database Schema – this is a the database shell that is defined using the CREATE data definition command.

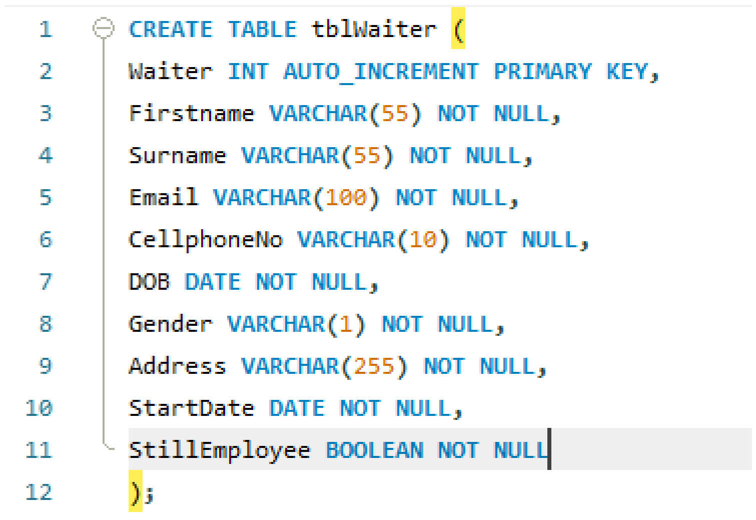


```
CREATE SCHEMA McDonaldOrderingSystem;
```

- Table Waiter – this table is used to store a waiter’s information as an employee of the restaurant.

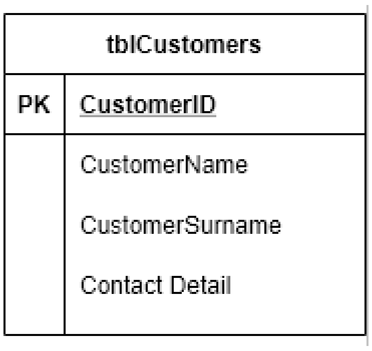


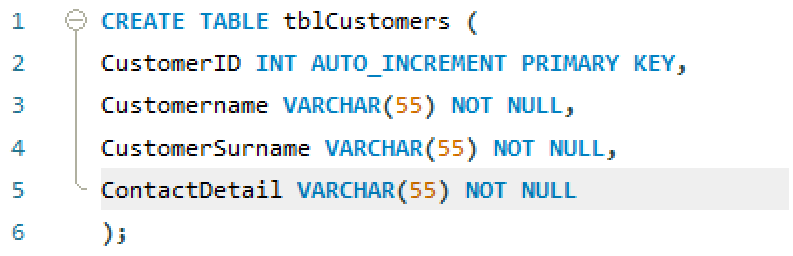
tblWaiter	
PK	Waiter
	Firstname
	Surname
	Email
	CellphoneNo
	DOB
	Gender
	Address
	StartDate
	StillEmployee



```
CREATE TABLE tblWaiter (
Waiter INT AUTO_INCREMENT PRIMARY KEY,
Firstname VARCHAR(55) NOT NULL,
Surname VARCHAR(55) NOT NULL,
Email VARCHAR(100) NOT NULL,
CellphoneNo VARCHAR(10) NOT NULL,
DOB DATE NOT NULL,
Gender VARCHAR(1) NOT NULL,
Address VARCHAR(255) NOT NULL,
StartDate DATE NOT NULL,
StillEmployee BOOLEAN NOT NULL
);
```

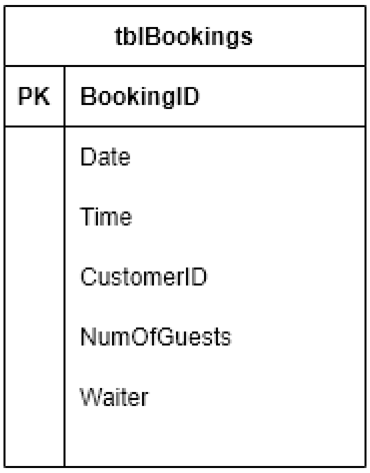
- Table Customers – this table is used to store customer personal details such as,, their name, surname, and contact detail, this data will be stored in such a way that it will not be duplicated instead if a particular customer's details already exist they will be referenced.

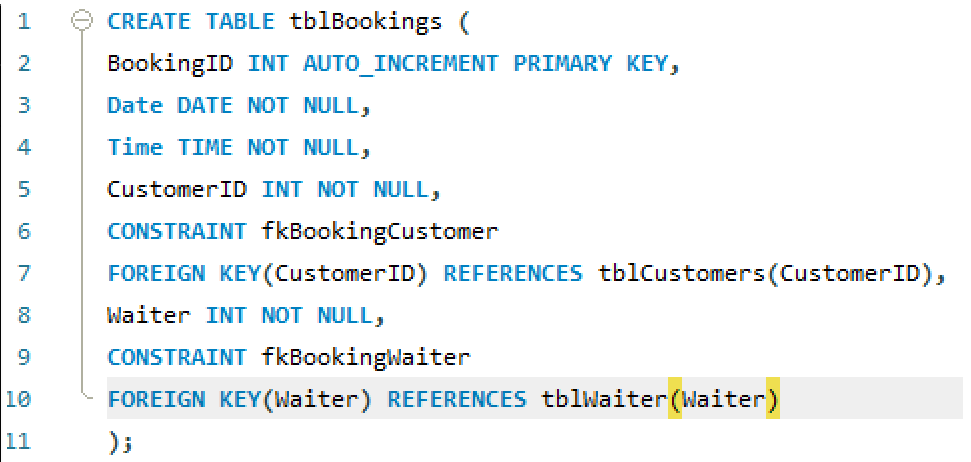




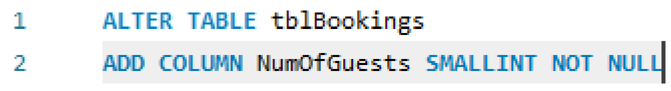
```
CREATE TABLE tblCustomers (
CustomerID INT AUTO_INCREMENT PRIMARY KEY,
Customername VARCHAR(55) NOT NULL,
CustomerSurname VARCHAR(55) NOT NULL,
ContactDetail VARCHAR(55) NOT NULL
);
```

- Table Bookings – this table Stores the date and time of a booking, reference to the customer that made the booking, number of guests the customer will bring, and a reference to a waiter who is going to handle the booking and provide service.



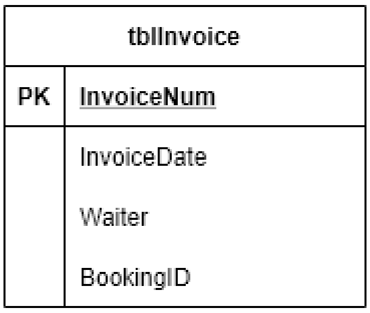


```
CREATE TABLE tblBookings (
BookingID INT AUTO_INCREMENT PRIMARY KEY,
Date DATE NOT NULL,
Time TIME NOT NULL,
CustomerID INT NOT NULL,
CONSTRAINT fkBookingCustomer
FOREIGN KEY(CustomerID) REFERENCES tblCustomers(CustomerID),
Waiter INT NOT NULL,
CONSTRAINT fkBookingWaiter
FOREIGN KEY(Waiter) REFERENCES tblWaiter(Waiter)
);
```

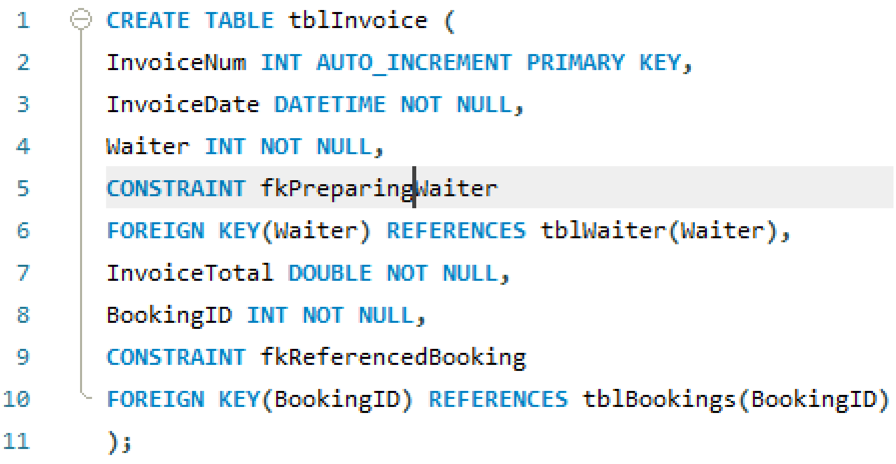


```
ALTER TABLE tblBookings
ADD COLUMN NumOfGuests SMALLINT NOT NULL
```

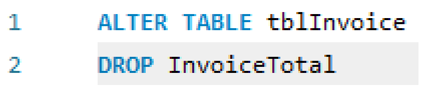
- Table Invoice – this table stores data about a booking payment made, the waiter that prepared an invoice, and invoice total. A fully prepared invoice gets its data from this table and a Line table which was a result of further normalisation of tthis table.



tblInvoice	
PK	InvoiceNum
	InvoiceDate Waiter BookingID

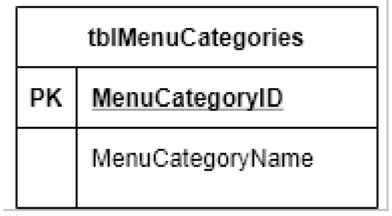


```
CREATE TABLE tblInvoice (
InvoiceNum INT AUTO_INCREMENT PRIMARY KEY,
InvoiceDate DATETIME NOT NULL,
Waiter INT NOT NULL,
CONSTRAINT fkPreparingWaiter
FOREIGN KEY(Waiter) REFERENCES tblWaiter(Waiter),
InvoiceTotal DOUBLE NOT NULL,
BookingID INT NOT NULL,
CONSTRAINT fkReferencedBooking
FOREIGN KEY(BookingID) REFERENCES tblBookings(BookingID)
);
```

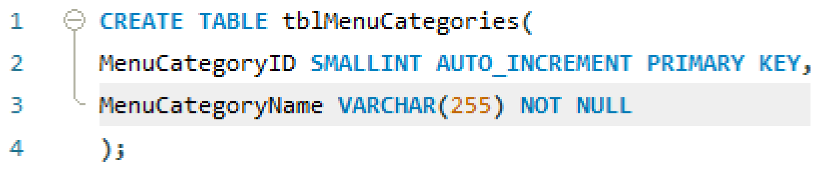


```
ALTER TABLE tblInvoice
DROP InvoiceTotal
```

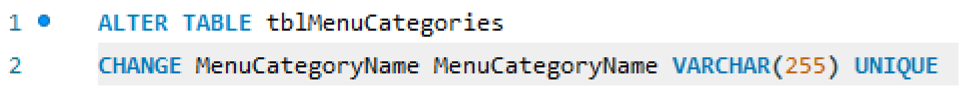
- Table Menu Categories – this table stores menu category names.



tblMenuCategories	
PK	MenuCategoryID
	MenuCategoryName

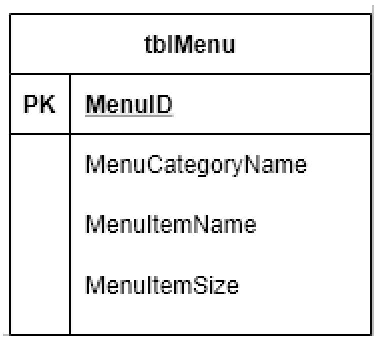


```
CREATE TABLE tblMenuCategories(
MenuCategoryID SMALLINT AUTO_INCREMENT PRIMARY KEY,
MenuCategoryName VARCHAR(255) NOT NULL
);
```

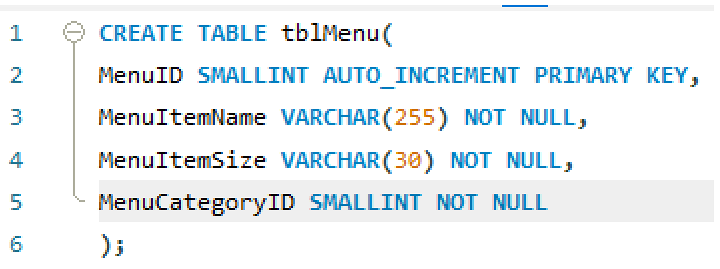


```
ALTER TABLE tblMenuCategories
CHANGE MenuCategoryName MenuCategoryName VARCHAR(255) UNIQUE
```

- Table Menu – this table stores menu option and categorisation details, menu item name, and menu item size.

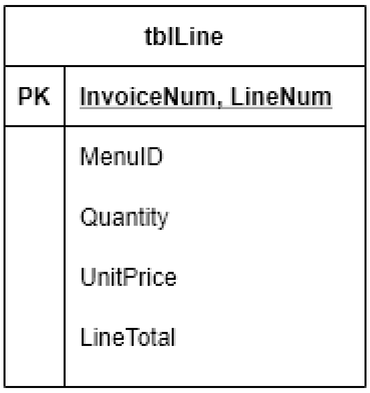


tblMenu	
PK	MenuID
	MenuCategoryName MenuItemName MenuItemSize

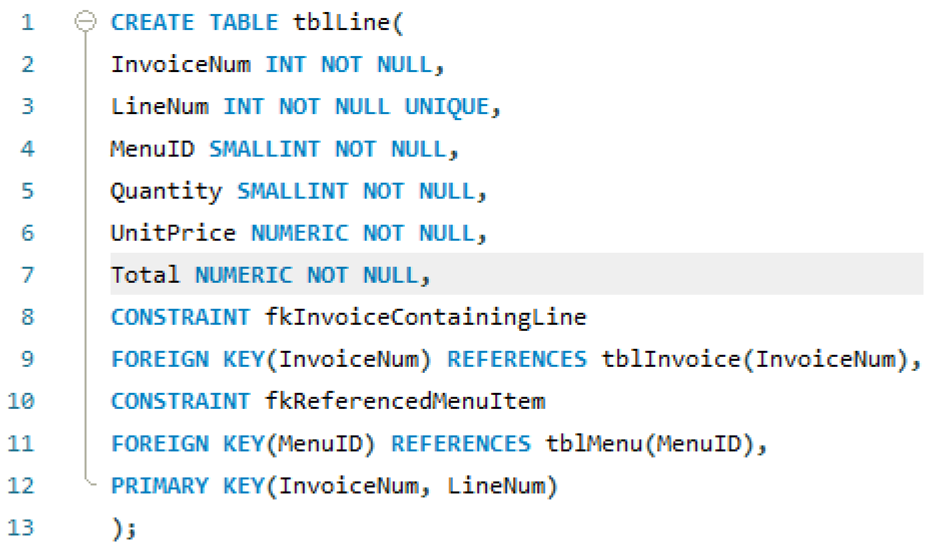


```
CREATE TABLE tblMenu(
MenuID SMALLINT AUTO_INCREMENT PRIMARY KEY,
MenuItemName VARCHAR(255) NOT NULL,
MenuItemSize VARCHAR(30) NOT NULL,
MenuCategoryID SMALLINT NOT NULL
);
```

- Table Line – this table has a surrogate key which is a combination of the line id and invoice number, this is because on its own, each line id does not provide enough information to know in which invoice the line is contained. This table stores a reference to table menu, unit price of the ordered and/or referenced menu item and its quantity.



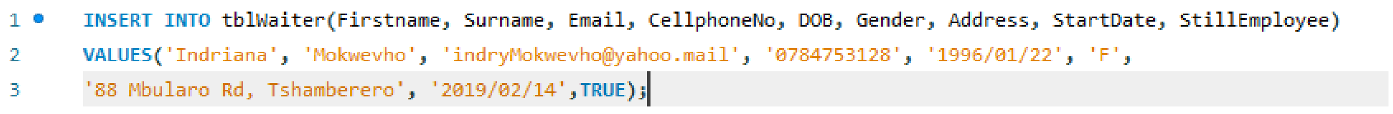
tblLine	
PK	InvoiceNum, LineNum
	MenuID
	Quantity
	UnitPrice
	LineTotal



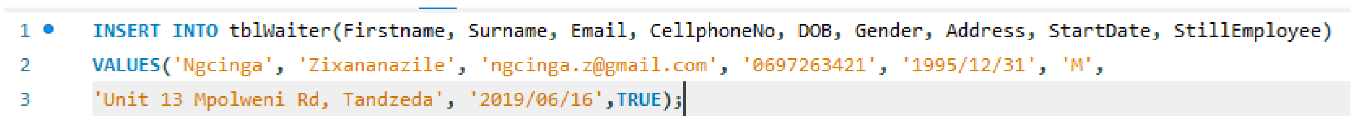
```sql
CREATE TABLE tblLine(
InvoiceNum INT NOT NULL,
LineNum INT NOT NULL UNIQUE,
MenuID SMALLINT NOT NULL,
Quantity SMALLINT NOT NULL,
UnitPrice NUMERIC NOT NULL,
Total NUMERIC NOT NULL,
CONSTRAINT fkInvoiceContainingLine
FOREIGN KEY(InvoiceNum) REFERENCES tblInvoice(InvoiceNum),
CONSTRAINT fkReferencedMenuItem
FOREIGN KEY(MenuID) REFERENCES tblMenu(MenuID),
PRIMARY KEY(InvoiceNum, LineNum)
);
```

Data Insertion and Table Display

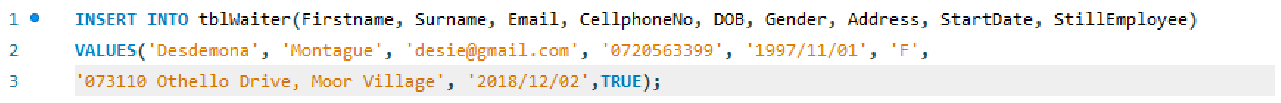
- Table Waiter



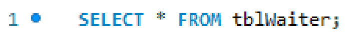
```sql
INSERT INTO tblWaiter(Firstname, Surname, Email, CellphoneNo, DOB, Gender, Address, StartDate, StillEmployee)
VALUES('Indriana', 'Mokwevho', 'indryMokwevho@yahoo.mail', '0784753128', '1996/01/22', 'F',
'88 Mbularo Rd, Tshamberero', '2019/02/14',TRUE);
```



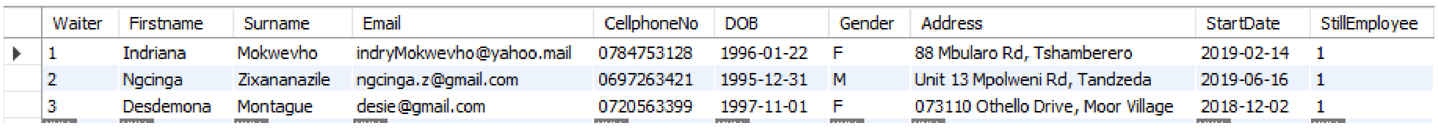
```sql
INSERT INTO tblWaiter(Firstname, Surname, Email, CellphoneNo, DOB, Gender, Address, StartDate, StillEmployee)
VALUES('Ngcinga', 'Zixananazile', 'ngcinga.z@gmail.com', '0697263421', '1995/12/31', 'M',
'Unit 13 Mpolweni Rd, Tandzeda', '2019/06/16',TRUE);
```



```sql
INSERT INTO tblWaiter(Firstname, Surname, Email, CellphoneNo, DOB, Gender, Address, StartDate, StillEmployee)
VALUES('Desdemona', 'Montague', 'desie@gmail.com', '0720563399', '1997/11/01', 'F',
'073110 Othello Drive, Moor Village', '2018/12/02',TRUE);
```

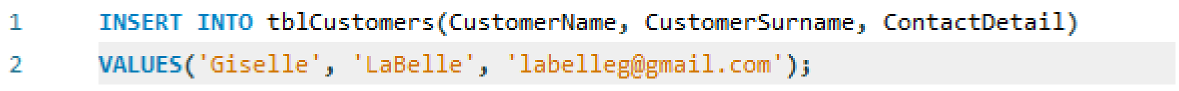


```sql
SELECT * FROM tblWaiter;
```

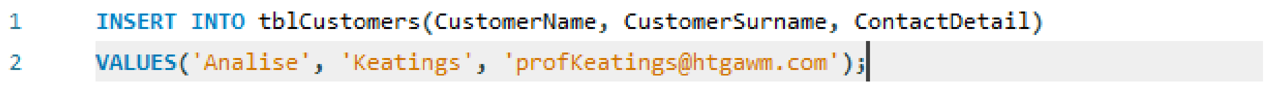


Waiter	Firstname	Surname	Email	CellphoneNo	DOB	Gender	Address	StartDate	StillEmployee
1	Indriana	Mokwevho	indryMokwevho@yahoo.mail	0784753128	1996-01-22	F	88 Mbularo Rd, Tshamberero	2019-02-14	1
2	Ngcinga	Zixananazile	ngcinga.z@gmail.com	0697263421	1995-12-31	M	Unit 13 Mpolweni Rd, Tandzeda	2019-06-16	1
3	Desdemona	Montague	desie@gmail.com	0720563399	1997-11-01	F	073110 Othello Drive, Moor Village	2018-12-02	1

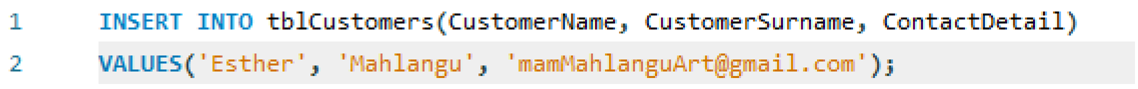
- Table Customers



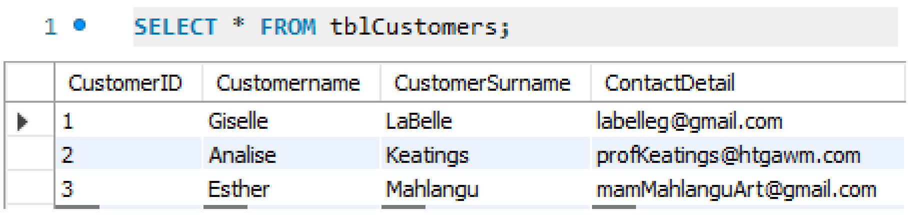
```sql
INSERT INTO tblCustomers(CustomerName, CustomerSurname, ContactDetail)
VALUES('Giselle', 'LaBelle', 'labelleg@gmail.com');
```



```sql
INSERT INTO tblCustomers(CustomerName, CustomerSurname, ContactDetail)
VALUES('Analise', 'Keatings', 'profKeatings@htgawm.com');
```



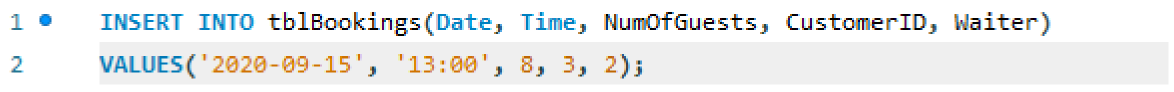
```sql
INSERT INTO tblCustomers(CustomerName, CustomerSurname, ContactDetail)
VALUES('Esther', 'Mahlangu', 'mamMahlanguArt@gmail.com');
```



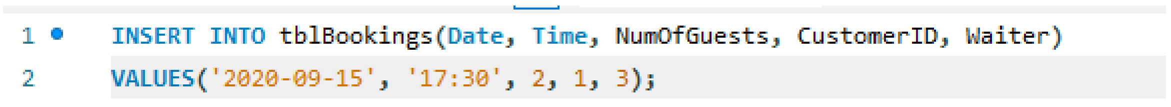
```sql
SELECT * FROM tblCustomers;
```

CustomerID	Customername	CustomerSurname	ContactDetail
1	Giselle	LaBelle	labelleg@gmail.com
2	Analise	Keatings	profKeatings@htgawm.com
3	Esther	Mahlangu	mamMahlanguArt@gmail.com

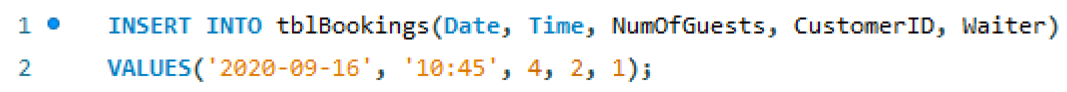
- Table Bookings



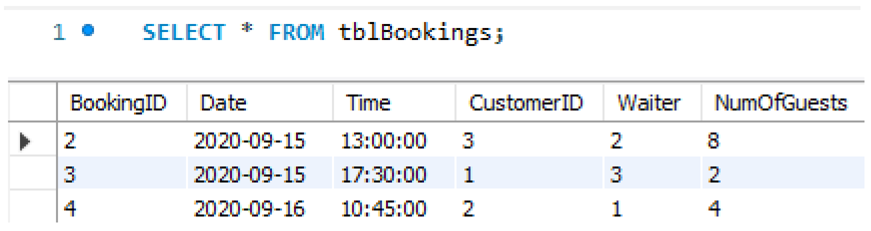
```sql
INSERT INTO tblBookings(Date, Time, NumOfGuests, CustomerID, Waiter)
VALUES('2020-09-15', '13:00', 8, 3, 2);
```



```sql
INSERT INTO tblBookings(Date, Time, NumOfGuests, CustomerID, Waiter)
VALUES('2020-09-15', '17:30', 2, 1, 3);
```



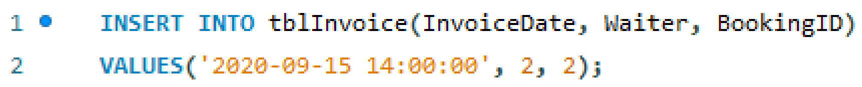
```sql
INSERT INTO tblBookings(Date, Time, NumOfGuests, CustomerID, Waiter)
VALUES('2020-09-16', '10:45', 4, 2, 1);
```



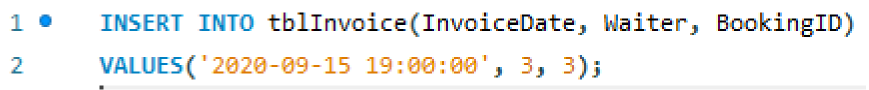
```sql
SELECT * FROM tblBookings;
```

BookingID	Date	Time	CustomerID	Waiter	NumOfGuests
2	2020-09-15	13:00:00	3	2	8
3	2020-09-15	17:30:00	1	3	2
4	2020-09-16	10:45:00	2	1	4

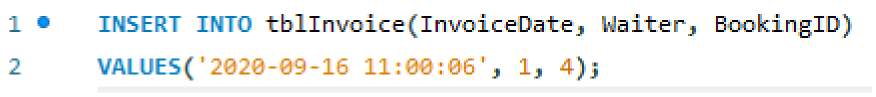
- Table Invoice



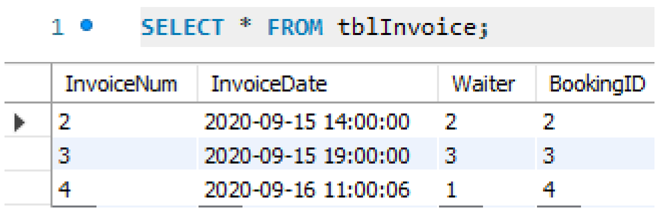
```sql
INSERT INTO tblInvoice(InvoiceDate, Waiter, BookingID)
VALUES('2020-09-15 14:00:00', 2, 2);
```



```sql
INSERT INTO tblInvoice(InvoiceDate, Waiter, BookingID)
VALUES('2020-09-15 19:00:00', 3, 3);
```



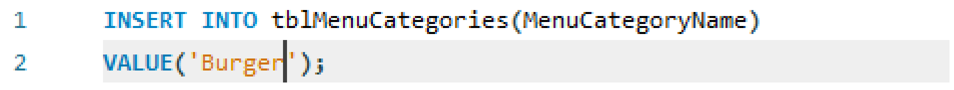
```sql
INSERT INTO tblInvoice(InvoiceDate, Waiter, BookingID)
VALUES('2020-09-16 11:00:06', 1, 4);
```



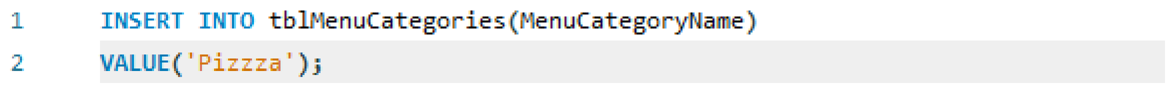
```sql
SELECT * FROM tblInvoice;
```

InvoiceNum	InvoiceDate	Waiter	BookingID
2	2020-09-15 14:00:00	2	2
3	2020-09-15 19:00:00	3	3
4	2020-09-16 11:00:06	1	4

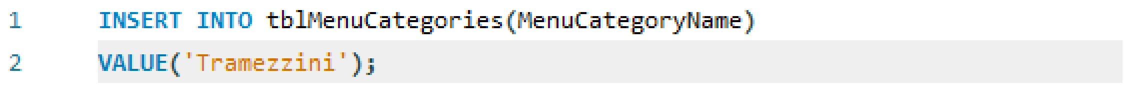
- Table Menu Categories



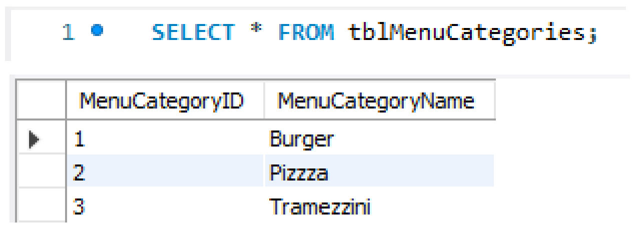
```
INSERT INTO tblMenuCategories(MenuCategoryName)
VALUE('Burger');
```



```
INSERT INTO tblMenuCategories(MenuCategoryName)
VALUE('Pizzza');
```



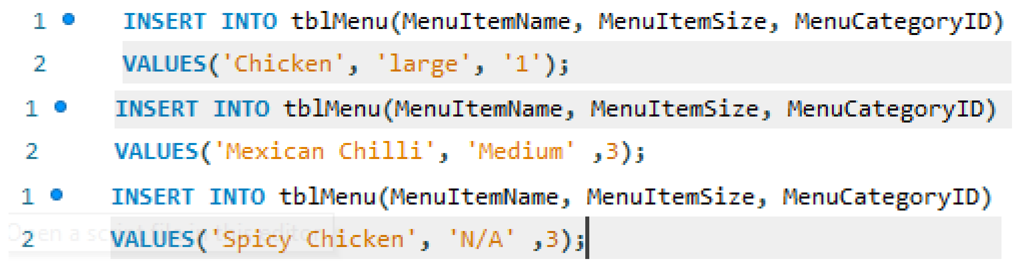
```
INSERT INTO tblMenuCategories(MenuCategoryName)
VALUE('Tramezzini');
```



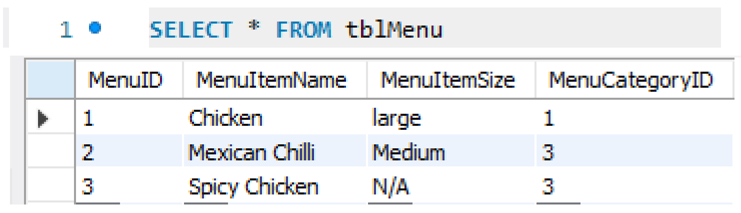
```
SELECT * FROM tblMenuCategories;
```

	MenuCategoryID	MenuCategoryName
▶	1	Burger
	2	Pizzza
	3	Tramezzini

- Table Menu



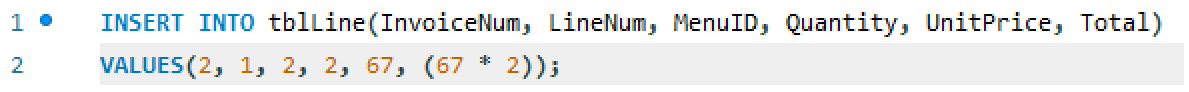
```
INSERT INTO tblMenu(MenuItemName, MenuItemSize, MenuCategoryID)
VALUES('Chicken', 'large', '1');
INSERT INTO tblMenu(MenuItemName, MenuItemSize, MenuCategoryID)
VALUES('Mexican Chilli', 'Medium' ,3);
INSERT INTO tblMenu(MenuItemName, MenuItemSize, MenuCategoryID)
VALUES('Spicy Chicken', 'N/A' ,3);
```



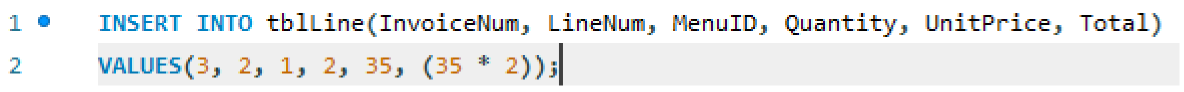
```
SELECT * FROM tblMenu
```

	MenuID	MenuItemName	MenuItemSize	MenuCategoryID
▶	1	Chicken	large	1
	2	Mexican Chilli	Medium	3
	3	Spicy Chicken	N/A	3

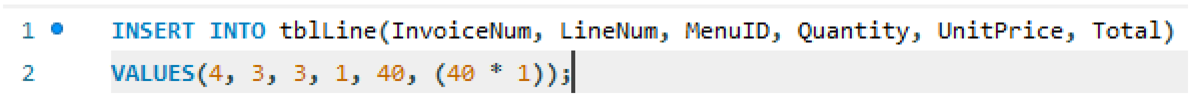
- Table Line



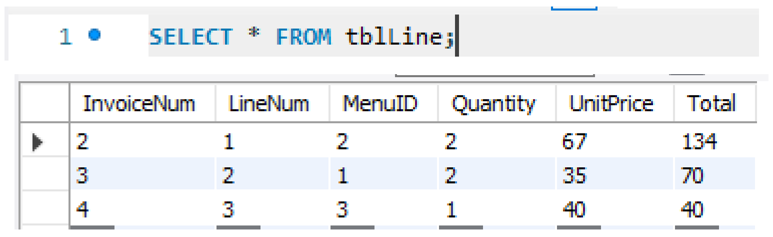
```
INSERT INTO tblLine(InvoiceNum, LineNum, MenuID, Quantity, UnitPrice, Total)
VALUES(2, 1, 2, 2, 67, (67 * 2));
```



```
INSERT INTO tblLine(InvoiceNum, LineNum, MenuID, Quantity, UnitPrice, Total)
VALUES(3, 2, 1, 2, 35, (35 * 2));
```



```
INSERT INTO tblLine(InvoiceNum, LineNum, MenuID, Quantity, UnitPrice, Total)
VALUES(4, 3, 3, 1, 40, (40 * 1));
```



```
SELECT * FROM tblLine;
```

InvoiceNum	LineNum	MenuID	Quantity	UnitPrice	Total
2	1	2	2	67	134
3	2	1	2	35	70
4	3	3	1	40	40