IST 659 – Project Final Report

A Restaurant Database Management System



-Aneesh Phatak(M002)

**Project Summary**

My project involves building a database model for a restaurant. I plan to use this database model to collect data necessary for deriving knowledge about key numbers such as daily revenue, number of orders, distribution of guests and staff payment information according to tasks completed. This is a massive upgrade to the existing state of affairs in which all information about the orders of the restaurant is stored manually.

Coming to the layout of the restaurant, this restaurant has three different seating areas. One prohibits alcohol and smoking, the second allows drinking but not smoking and the third allows both. The procedure of placing an order is as follows:

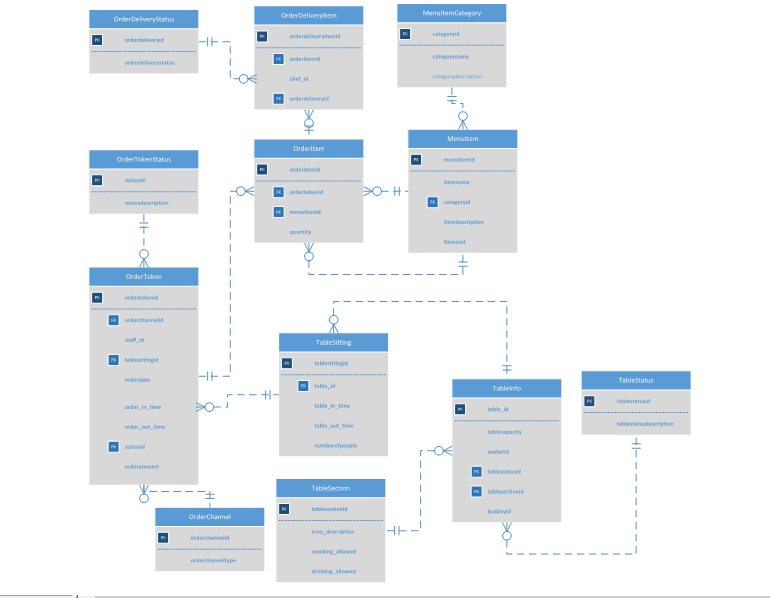
1. Arriving customers will be greeted by the host, ask about number of people in the group and which section they would like to be seated in. The host will also assign waiters to groups of diners.
2. The waiter will take orders from tables and bring prepared orders to the table.
3. The supervisor will keep a track of incoming orders and accordingly assign tasks to the chefs.
4. The chefs take instructions from the supervisor, work on their tasks and report to the supervisor after completing their tasks.
5. The waiters are in charge of billing and payment of a customer. After the payment is done, the waiter informs the manager who then assigns busboys to clean tables after a group leaves. The busboy cleans tables and informs manager who then updates status of the table.

**Entity and Attribute Glossary:**

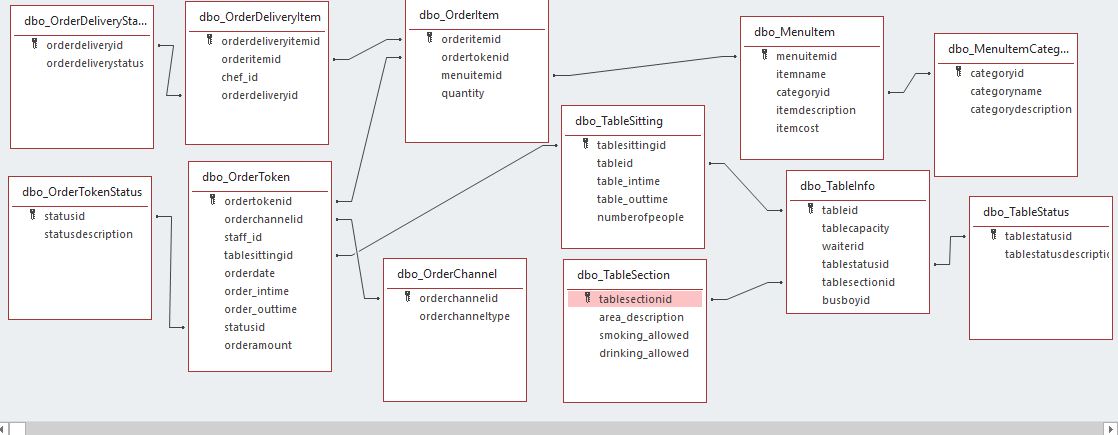
Entities are marked in bold while attributes are not. Attributes which are primary keys contain PK as a prefix while those which are foreign keys contain FK as a prefix

|  |  |
| --- | --- |
| Entity and attributes | Description |
| **OrderToken** | Contains preliminary order details |
| **PK**-ordertokenid  **FK**-orderchannelid  staff\_id  **FK**-tablesittingid  orderdate  order\_in\_time  order\_out\_time  **FK**-statusid  orderamount | Unique id used to identify each order  Channel through which order is placed(References PK orderchannelid in OrderChannel,1 order channel can have many order tokens)  Id of the waiter who is assigned  Id of table where guests are seated(References PK tablesittingid in TableSitting, 1 table can have many order tokens)  Date on which order was placed  Timestamp when guests arrive at the restaurant  Timestamp when guests leave the restaurant  Defines current status of an order(References statusid in OrderTokenStatus, 1 order status can have many order tokens)  Total amount due by the customer |
| **OrderChannel** | Information about channel through which order is placed |
| **PK**-orderchannelid  orderchanneltype | Unique id to identify each order channel  Type of channel like ‘online’, ’takeaway’, ‘dine-in’ |
| **OrderTokenStatus** | Information about what each statusid means |
| **PK**-statusid  statusdescription | Unique id used to identify each type of order status  Contains all possible statuses like ‘order received’, ‘order in progress’, ‘order delivered’ |
| Contd: |  |
| **OrderItem** | Contains information on items ordered |
| **PK**-orderitemid  **FK**-ordertokenid  **FK**-menuitemid  quantity | Unique id used to identify each instance of an order(eg- A second order by a customer added to the same bill will contain same ordertokenid but different orderitemid)  Helps us to identify which ordertokenid this particular orderitemid belongs to(References ordertokenid from OrderToken, 1 order token can have many order items)  ID of the menu item this order item contains(References menuitemid in MenuItem, 1 menu item can belong to multiple order items)  Quantity of item ordered from menu |
| **OrderDeliveryStatus** | Contains information on what each delivery status means |
| **PK** - orderdeliveryid  orderdeliverystatus | Uniquely identifies each order delivery status  Type of status of order delivery(eg- 1 means ‘Order just received and 5 means ‘Ready’) |
| **OrderDeliveryItem** |  |
| **PK**- orderdeliveryid  **FK** – menuitemid  chef\_id  **FK** - orderdeliverystatus | Uniqely identifies each status of each order item for delivery  ID of item in menu(References menuitemid in MenuItem, 1 menu item can belong to multiple order delivery items)  ID of chef who is assigned to an item  Status of order item for delivery(References orderdeliverystatus in OrderDeliveryStatus,1 order delivery status can have many order delivery items) |
| **MenuItem** | Contains all menu items and their description,category and cost |
| **PK** - menuitemid  itemname  **FK** - categoryid  itemdescription  itemcost | Uniquely identifies each item on the menu  Name of the item  Category of the item(References categoryid in MenuItemCategory, 1 category can have multiple menu items)  Description of item  Cost of item |
| **MenuItemCategory** | Contains all possible categories of menu items |
| **PK** - categoryid  categoryname  categorydescription | Uniquely identifies each menu category  Name of the category  Description of the category |
| **TableInfo** | Information on table in which guests are seated |
| **PK** – table\_id  tablecapacity  waiterid  **FK** - tablestatusid  **FK** - tablesectionid  busboyid | Uniquely identifies each table  Capacity of table  Id of waiter assigned  Table status like ‘Occupied’, ‘Free & clean’, ‘Unclean’  ( References tablestatusid in TableStatus, 1 table status can belong to many tables)  Shows section in which table is(References tablesectionid in TableSection, 1 Section can have multiple tables)  Id of busboy assigned |
| **TableSection** | Contains all sections and their descriptions |
| **PK** - tablesectionid  area\_description  smoking\_allowed  drinking\_allowed | Uniquely identifies each section  Description of section  1 indicates that smoking is allowed, 0 otherwise  1 indicates that drinking is allowed, 0 otherwise |
| **TableStatus** | Contains all possible statuses for a table |
| **PK** - tablestatusid  tablestatusdescription | Uniquely identifies each status for a table like Table status = ‘Occupied’, ‘Free & clean’, ‘Unclean’  Description of status |
| **Tablesitting** | Contains information on number of guests, the table they were seated on, their in and out times. |
| **PK** - tablesittingid  **FK** – table\_id  table\_in\_time  table\_out\_time  numberofpeople | Uniqely identifies each instance of a table being occupied  Contains ID of table(References table\_id in TableInfo, 1 table\_id can have multiple tablesittingid’s)  Time when guests are seated  Time when guests leave the table  Number of guests |

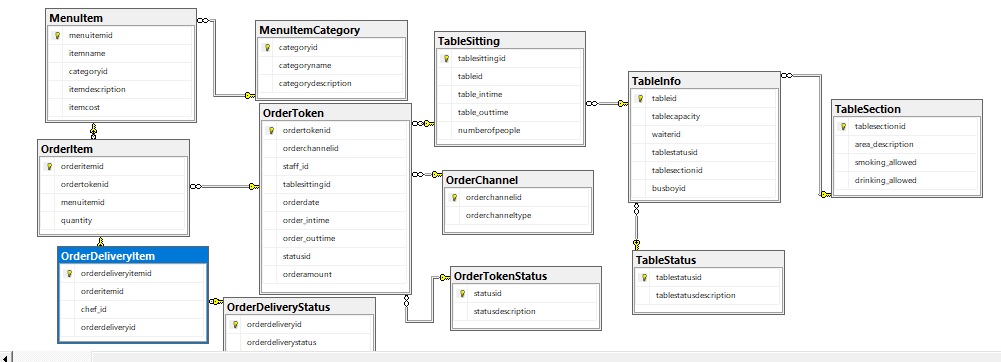
**Entity Relationship Diagram in Visio**

****

**Entity Relationship Diagram in Access**



**Entity Relationship Diagram in MS SQL Server**

****

**SQL Script for creating tables**

DROP TABLES:

/\*DROP TABLE OrderDeliveryItem

DROP TABLE OrderItem

DROP TABLE OrderToken

DROP TABLE TableSitting

DROP TABLE TableInfo

DROP TABLE MenuItem

DROP TABLE TableSection

DROP TABLE TableStatus

DROP TABLE OrderChannel

DROP TABLE OrderTokenStatus

DROP TABLE MenuItemCategory

DROP TABLE OrderDeliveryStatus

\*/

CREATE TABLES:

CREATE TABLE OrderDeliveryStatus

(

orderdeliveryid INTEGER PRIMARY KEY,

orderdeliverystatus CHAR(20) NOT NULL

);

CREATE TABLE MenuItemCategory

(

categoryid INTEGER PRIMARY KEY,

categoryname CHAR(15) NOT NULL,

categorydescription CHAR(30) NOT NULL

);

CREATE TABLE OrderTokenStatus

(

statusid INTEGER PRIMARY KEY,

statusdescription CHAR(15) NOT NULL

);

CREATE TABLE OrderChannel

(

orderchannelid INTEGER PRIMARY KEY,

orderchanneltype CHAR(10) NOT NULL

);

CREATE TABLE TableStatus

(

tablestatusid INTEGER PRIMARY KEY,

tablestatusdescription CHAR(10) NOT NULL

);

CREATE TABLE TableSection

(

tablesectionid INTEGER PRIMARY KEY,

area\_description CHAR(20) NOT NULL,

smoking\_allowed INTEGER NOT NULL,

drinking\_allowed INTEGER NOT NULL

);

CREATE TABLE MenuItem

(

menuitemid INTEGER PRIMARY KEY,

itemname CHAR(15) NOT NULL,

categoryid INTEGER NOT NULL,

itemdescription CHAR(30) NOT NULL,

itemcost DECIMAL(6,2) NOT NULL,

constraint MenuItem\_FK FOREIGN KEY (categoryid) REFERENCES MenuItemCategory(categoryid)

);

CREATE TABLE TableInfo

(

tableid INTEGER PRIMARY KEY,

tablecapacity INTEGER NOT NULL,

waiterid INTEGER NOT NULL,

tablestatusid INTEGER NOT NULL,

tablesectionid INTEGER NOT NULL,

busboyid INTEGER NOT NULL,

constraint TableInfo\_FK\_tablestatusid FOREIGN KEY (tablestatusid) REFERENCES TableStatus(tablestatusid),

constraint TableInfo\_FK\_tablesectionid FOREIGN KEY (tablesectionid) REFERENCES TableSection(tablesectionid)

);

CREATE TABLE TableSitting

(

tablesittingid INTEGER PRIMARY KEY,

tableid INTEGER NOT NULL,

table\_intime TIME(0) NOT NULL,

table\_outtime TIME(0) NOT NULL,

numberofpeople INTEGER NOT NULL,

constraint TableSitting\_FK FOREIGN KEY (tableid) REFERENCES TableInfo(tableid)

);

CREATE TABLE OrderToken

(

ordertokenid INTEGER PRIMARY KEY,

orderchannelid INTEGER NOT NULL,

staff\_id INTEGER NOT NULL,

tablesittingid INTEGER,

orderdate DATE NOT NULL,

order\_intime TIME(0) NOT NULL,

order\_outtime TIME(0) NOT NULL,

statusid INTEGER NOT NULL,

orderamount DECIMAL(6,2),

constraint OrderToken\_FK\_orderchannelid FOREIGN KEY (orderchannelid) REFERENCES OrderChannel(orderchannelid),

constraint OrderToken\_FK\_tablesittingid FOREIGN KEY (tablesittingid) REFERENCES TableSitting(tablesittingid),

constraint OrderToken\_FK\_statusid FOREIGN KEY (statusid) REFERENCES OrderTokenStatus(statusid)

);

CREATE TABLE OrderItem

(

orderitemid INTEGER PRIMARY KEY,

ordertokenid INTEGER NOT NULL,

menuitemid INTEGER NOT NULL,

quantity INTEGER NOT NULL,

constraint OrderItem\_FK\_ordertokenid FOREIGN KEY (ordertokenid) REFERENCES OrderToken(ordertokenid),

constraint OrderItem\_FK\_menuitemid FOREIGN KEY (menuitemid) REFERENCES MenuItem(menuitemid)

);

CREATE TABLE OrderDeliveryItem

(

orderdeliveryitemid INTEGER PRIMARY KEY,

orderitemid INTEGER NOT NULL,

chef\_id INTEGER NOT NULL,

orderdeliveryid INTEGER NOT NULL,

constraint OrderDeliveryItem\_FK\_orderitemid FOREIGN KEY (orderitemid) REFERENCES OrderItem(orderitemid),

constraint OrderDeliveryItem\_FK\_orderdeliveryid FOREIGN KEY (orderdeliveryid) REFERENCES OrderDeliveryStatus(orderdeliveryid)

);

**SQL Script for inserting data into tables**

INSERT INTO OrderDeliveryStatus Values (1,'Order just received')

INSERT INTO OrderDeliveryStatus Values (2,'Prepping')

INSERT INTO OrderDeliveryStatus Values (3,'Cooking')

INSERT INTO OrderDeliveryStatus Values (4,'Plating')

INSERT INTO OrderDeliveryStatus Values (5,'Ready')

SELECT \* FROM OrderDeliveryStatus

INSERT INTO MenuItemCategory Values (1,'Italian','Pizza and Pasta')

INSERT INTO MenuItemCategory Values (2,'American','Burgers and Mac&Cheese')

INSERT INTO MenuItemCategory Values (3,'Mexican','Burritos,Tacos & Quessadilas')

INSERT INTO MenuItemCategory Values (4,'Indian','Breads and Curries')

SELECT \* FROM MenuItemCategory

INSERT INTO OrderTokenStatus Values (1,'Order Received')

INSERT INTO OrderTokenStatus Values (2,'Preparing Order')

INSERT INTO OrderTokenStatus Values (3,'Finishing')

INSERT INTO OrderTokenStatus Values (4,'Order Ready')

SELECT \* FROM OrderTokenStatus

INSERT INTO OrderChannel Values (1,'online')

INSERT INTO OrderChannel Values (2,'takeaway')

INSERT INTO OrderChannel Values (3,'dine-in')

SELECT \* FROM OrderChannel

INSERT INTO TableStatus Values(1,'Occupied')

INSERT INTO TableStatus Values(2,'Free&Clean')

INSERT INTO TableStatus Values(3,'Unclean')

SELECT \* FROM TableStatus

INSERT INTO TableSection Values (1,'Indoor',0,0)

INSERT INTO TableSection Values (2,'Outdoor',1,1)

INSERT INTO TableSection Values (3,'Lounge',0,1)

SELECT \* FROM TableSection

INSERT INTO MenuItem Values(1,'Cheese Pizza', 1, 'Mozzarella Cheese', 14.00)

INSERT INTO MenuItem Values(2,'Mac&Cheese', 2, 'Macaron with cheese sauce', 8.00)

INSERT INTO MenuItem Values(3,'Bean Burrito', 3, 'Veggies,Rice and Beans', 9.00)

INSERT INTO MenuItem Values(4,'Chole Bhature', 4, 'Chickpea gravy with bread', 7.50)

SELECT \* FROM MenuItem

INSERT INTO TableInfo Values (1,4,1,1,1,1)

INSERT INTO TableInfo Values (2,6,2,3,2,2)

INSERT INTO TableInfo Values (3,8,3,2,3,3)

INSERT INTO TableInfo Values (4,4,4,1,2,2)

SELECT \* FROM TableInfo

INSERT INTO TableSitting Values (1,1,'12:43:21','13:21:21',3)

INSERT INTO TableSitting Values (2,2,'12:52:24','13:42:56',6)

INSERT INTO TableSitting Values (3,3,'13:41:12','14:30:43',6)

INSERT INTO TableSitting Values (4,4,'14:53:21','15:11:41',2)

SELECT \* FROM TableSitting

INSERT INTO OrderToken Values (1,1,1,1,'2019-11-11','12:43:21','13:21:21',1,0)

INSERT INTO OrderToken Values (2,2,2,2,'2019-11-11','13:33:42','13:45:24',1,0)

INSERT INTO OrderToken Values (3,1,3,3,'2019-11-11','14:24:11','14:51:23',1,0)

INSERT INTO OrderToken Values (4,1,3,3,'2019-11-11','14:24:11','14:51:23',1,0)

INSERT INTO OrderToken Values (5,1,1,1,'2019-11-12','12:43:21','13:21:21',1,0)

INSERT INTO OrderToken Values (6,2,2,2,'2019-11-12','13:33:42','13:45:24',1,0)

INSERT INTO OrderToken Values (7,1,3,3,'2019-11-13','14:24:11','14:51:23',1,0)

INSERT INTO OrderToken Values (8,1,3,3,'2019-11-13','14:24:11','14:51:23',1,0)

SELECT \* FROM OrderToken

INSERT INTO OrderItem Values(1,1,1,3)

INSERT INTO OrderItem Values(2,2,2,4)

INSERT INTO OrderItem Values(3,3,4,2)

INSERT INTO OrderItem Values(4,3,1,4)

INSERT INTO OrderItem Values(5,4,1,4)

INSERT INTO OrderItem Values(6,5,4,2)

INSERT INTO OrderItem Values(7,6,1,3)

INSERT INTO OrderItem Values(8,7,3,4)

INSERT INTO OrderItem Values(9,8,2,4)

SELECT \* FROM OrderItem

INSERT INTO OrderDeliveryItem Values (1,1,1,1)

INSERT INTO OrderDeliveryItem Values (2,2,1,1)

INSERT INTO OrderDeliveryItem Values (3,3,1,1)

INSERT INTO OrderDeliveryItem Values (4,4,1,1)

INSERT INTO OrderDeliveryItem Values (5,2,1,1)

INSERT INTO OrderDeliveryItem Values (6,3,4,1)

INSERT INTO OrderDeliveryItem Values (7,1,3,1)

INSERT INTO OrderDeliveryItem Values (8,2,2,1)

INSERT INTO OrderDeliveryItem Values (9,3,2,1)

SELECT \* FROM OrderDeliveryItem

**Major Data Questions and Reports**

**Major data questions:**

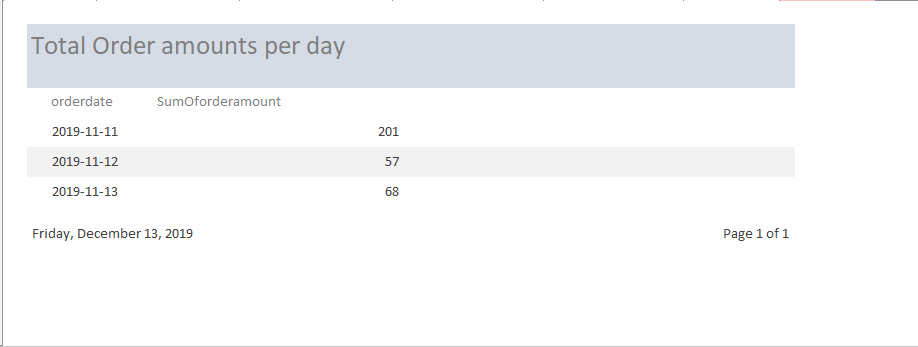
1. **What is the total amount of money earned per day?**

This is useful for monitoring daily revenue

SELECT Sum(OrderToken.orderamount) AS SumOforderamount, OrderToken.orderdate

FROM OrderToken

GROUP BY OrderToken.orderdate;



1. **What is the total quantity of menu items each ordered per day?**

This will help us identify which menu items are popular and which aren’t. The unpopular items can then be replaced with new recipes.

SELECT Sum(OrderItem.quantity) AS SumOfquantity, MenuItem.itemname, OrderToken.orderdate

FROM (MenuItem INNER JOIN OrderItem ON MenuItem.menuitemid = OrderItem.menuitemid)

INNER JOIN OrderToken ON OrderItem.ordertokenid = OrderToken.ordertokenid

GROUP BY MenuItem.itemname, OrderToken.orderdate;



1. **What is the total number of orders from each section specifically per day?**

Using this, we can keep track of the activity in each section and consider expanding one particular section if it is doing really well.

SELECT Count(OrderToken.ordertokenid) AS CountOfordertokenid, OrderToken.orderdate, TableSection.area\_description

FROM OrderToken, TableSection

GROUP BY OrderToken.orderdate, TableSection.area\_description;



1. **What is the total number of customers per day?**

This will help us understand the trend of arriving customers(whether the footfall higher on holidays or weekends?

SELECT Sum(TableSitting.numberofpeople) AS SumOfnumberofpeople, OrderToken.orderdate

FROM OrderToken INNER JOIN TableSitting ON OrderToken.tablesittingid = TableSitting.tablesittingid

GROUP BY OrderToken.orderdate;



1. **How many customers/tables did each waiter serve per day?**

This will help us calculate amount to paid to each waiter.

SELECT Count(OrderToken.ordertokenid) AS CountOfordertokenid, OrderToken.orderdate, OrderToken.staff\_id

FROM OrderToken

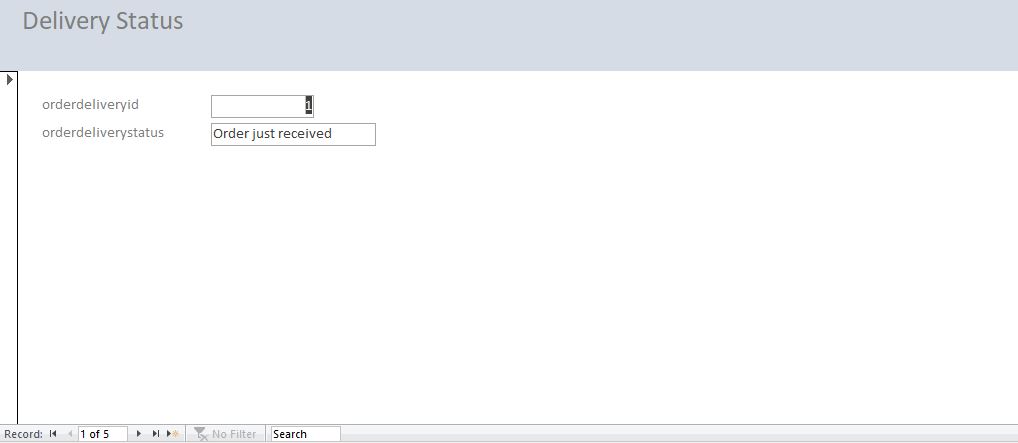
GROUP BY OrderToken.orderdate, OrderToken.staff\_id;



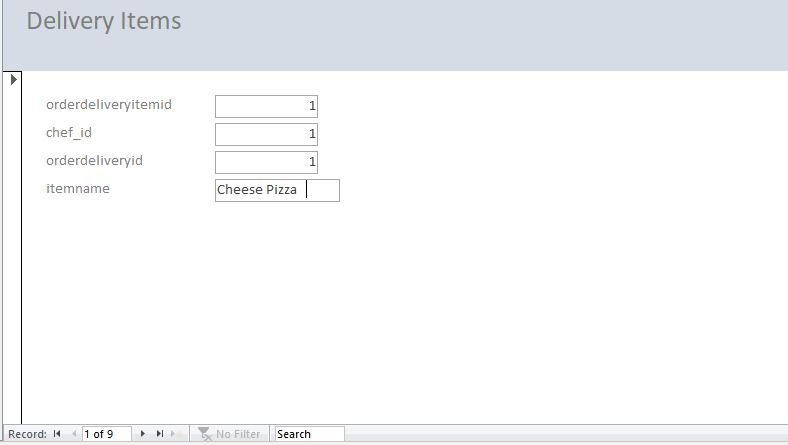
**Interfaces**

**Forms:**

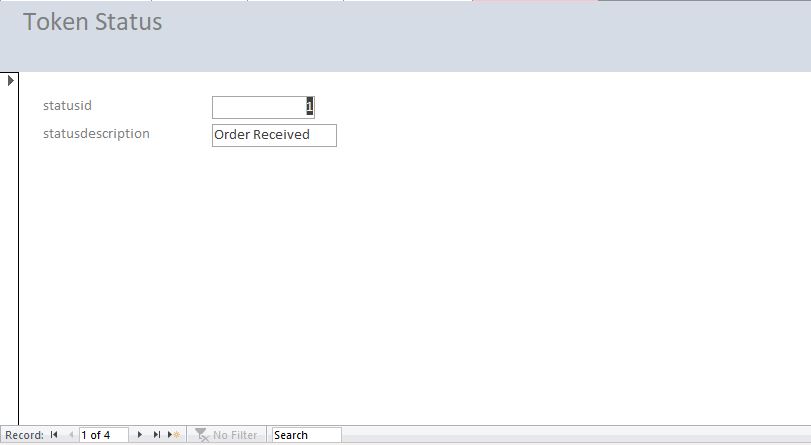
1. **Delivery status:** You can add new delivery statuses for online and takeaway orders

****

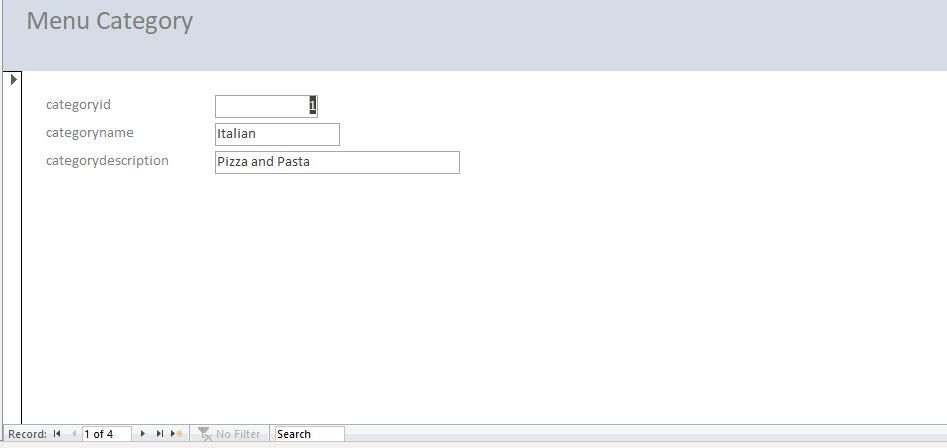
1. **Delivery Item:** Allows you to add recently ordered items for preparation

****

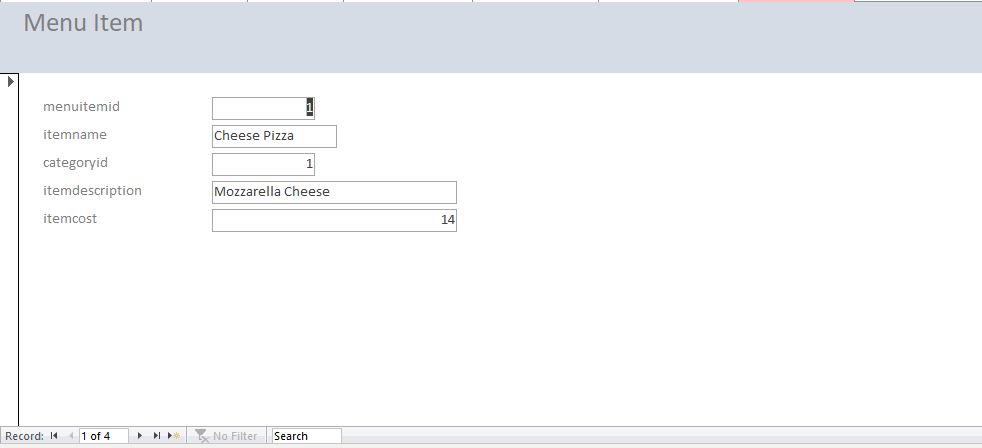
1. **Token Status**: You can add new token statuses for dine-in orders

****

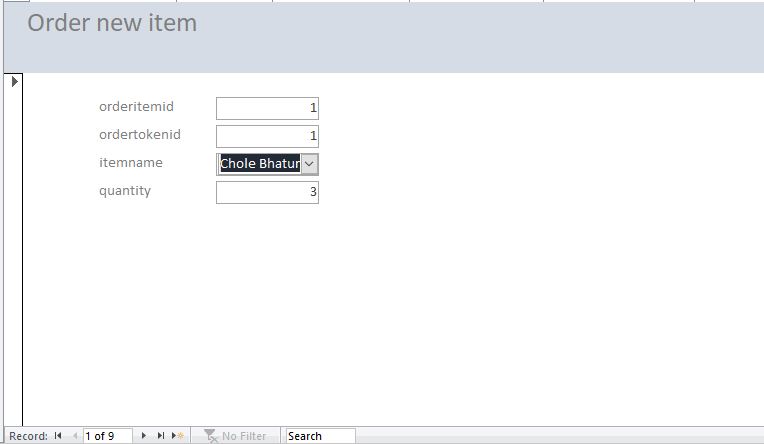
1. **Menu Category:** New categories of food can be added

****

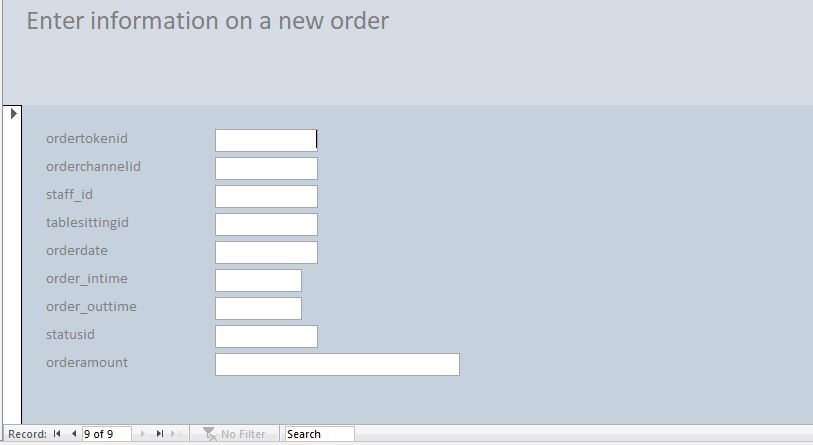
**5. Menu Item**: New Menu Items can be added

****

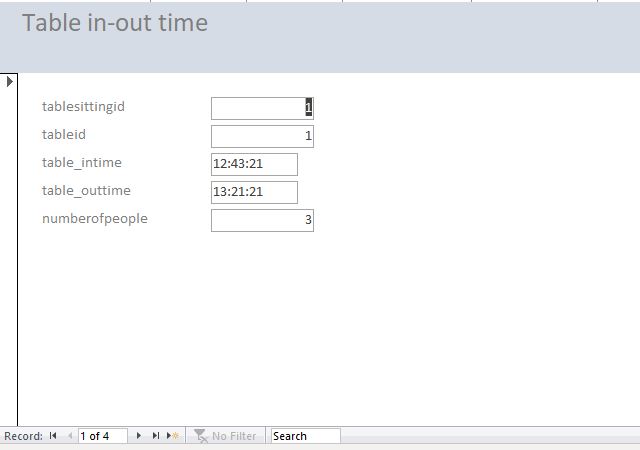
1. **Order Items:** Waiter takes down order items and their quantity

****

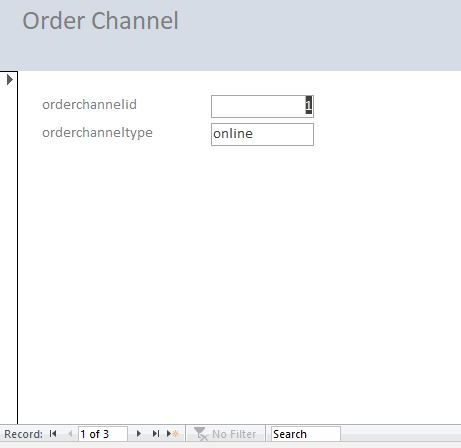
1. **Order Token:** Host enters information about guests here

****

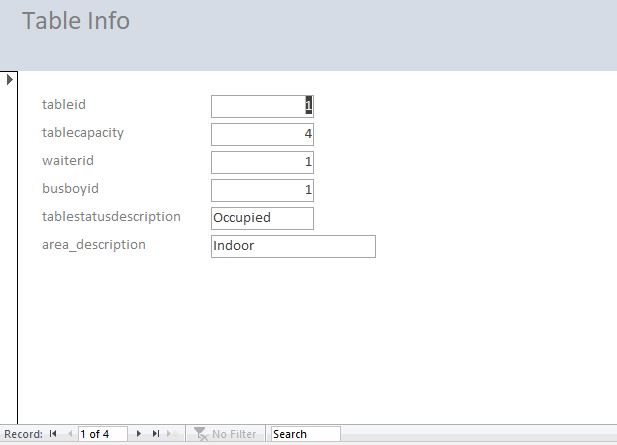
1. **Table in-out time:** Times when customers enter and leave are noted along with the number of guests

****

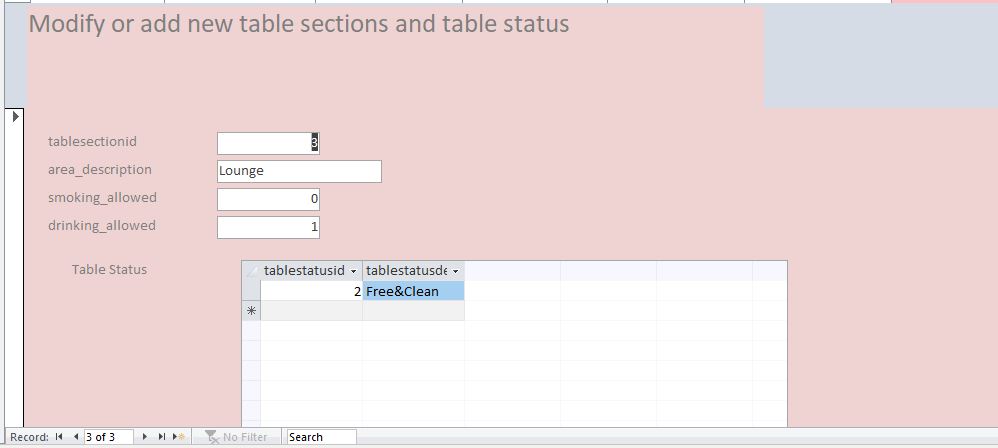
1. **Order Channel:** Can add new order channels

****

1. **Table Info:** You can add and update table id, waiter id, status and section of table, busboy id

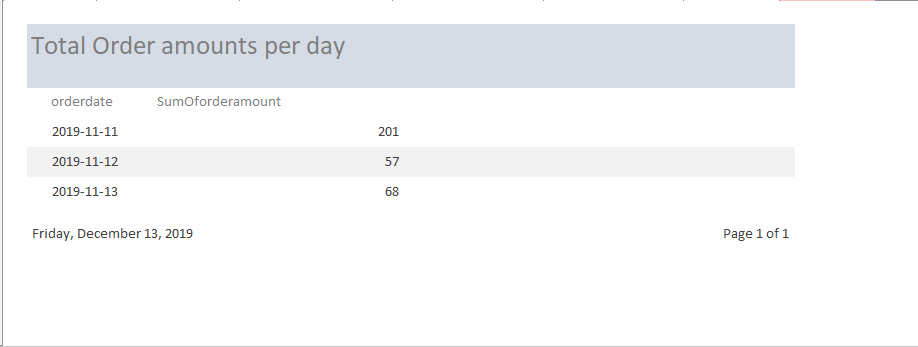
****

1. **Table Status (has a sub-form):** New sections as well as New statuses for readiness of a table to accommodate guests can be noted

****

**Reports:**

1. **Total order amounts per day : Total revenue generated per day**



1. **Quantity of item per day: For each date, reports the menu item ordered and it’s quantity(If an item hasn’t been ordered on that day, there is no corresponding entry)**



1. **Total orders per section per day: Reports the number of orders for a particular area per day(If a section hasn’t been used on that day, there is no corresponding entry)**



1. **Number of people eating in per day: Counts total guests by date.**

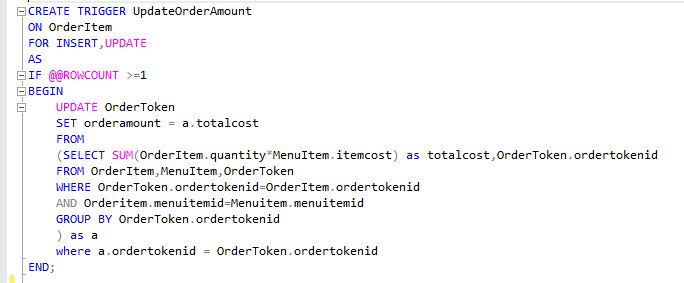


1. **Count of tables served by waiter per day: According to staff\_id, calculates number of tables served per day(useful for calculating paychecks for each waiter)**

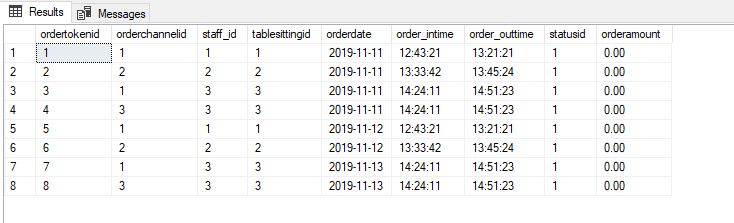


**Trigger**

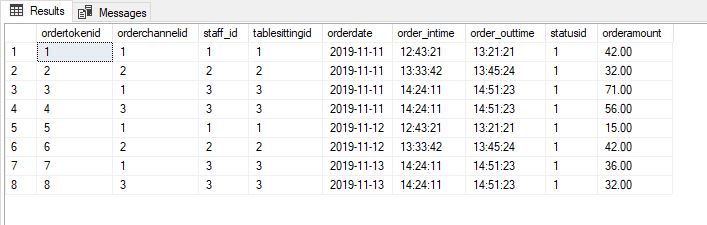
Updates orderamount in OrderToken according to items ordered in OrderItem(quantity is derived from this entity) by the logic orderamount = quantity\*itemcost for each item. Item cost is derived from MenuItem. Initially orderamount is set to zero. After the trigger is executed orderamount gets populated in OrderToken.

****

**Before trigger is executed:**

****

**After trigger is executed: (As soon as we insert values into OrderItem, orderamount will reflect in OrderToken)**

****