Assessment Front Sheet

Assessment Title	Database Application for Fast Burgers	
Qualification	Module Code and title	
HND in Computer Science	HP2J 48 Relational Database Management Systems	
Student ID	Assessor's Name	
01000552	Kanchana Senadheera	
Cohort	Date Issued	Submitted on
SP 20	20th of June 2021	10th of January 2022

No.	Learning Outcome	Task no
2	Design an RDBMS from a given scenario.	
3	Map the design model to the physical model.	
4	Create and run SQL statements/ queries on a RDBMS.	

I certify that the work submitted for this Assessment is my own and research sources are fully acknowledged.

Student Signature: Malinga Rajapaksha Date: 10/01/2022

DATABASE SOLUTION FOR FASTBURGERS INC

Malinga Rajapaksha

01000552

15/01/2022

ABSTRACT

A relational database management system (RDBMS) incorporates a relativity data model, which often includes an application programming interface for the Structured Query Language (SQL) (SQL). It is a relational database management system (RDBMS) that organizes, and accesses databases based on the connections between data items. In a relational database, tables represent the relationships between data elements. For interdependencies between these tables, data values rather than references are indicated. This allows for a high level of data independence.

ENTITIES AND ATTRIBUTES

Entity Name: Customer
Attribute
Customer ID
First name
Last name
Contact
email

Entity Name: order
Attribute
Order ID
Order Date
Time
total

Entity Name: staff
Attribute
Staff ID
First Name
Last name
role

Entity Name: shift
Attribute
Staff ID
Shift ID
Role

Entity Name: menu
Attribute
Menu ID
type

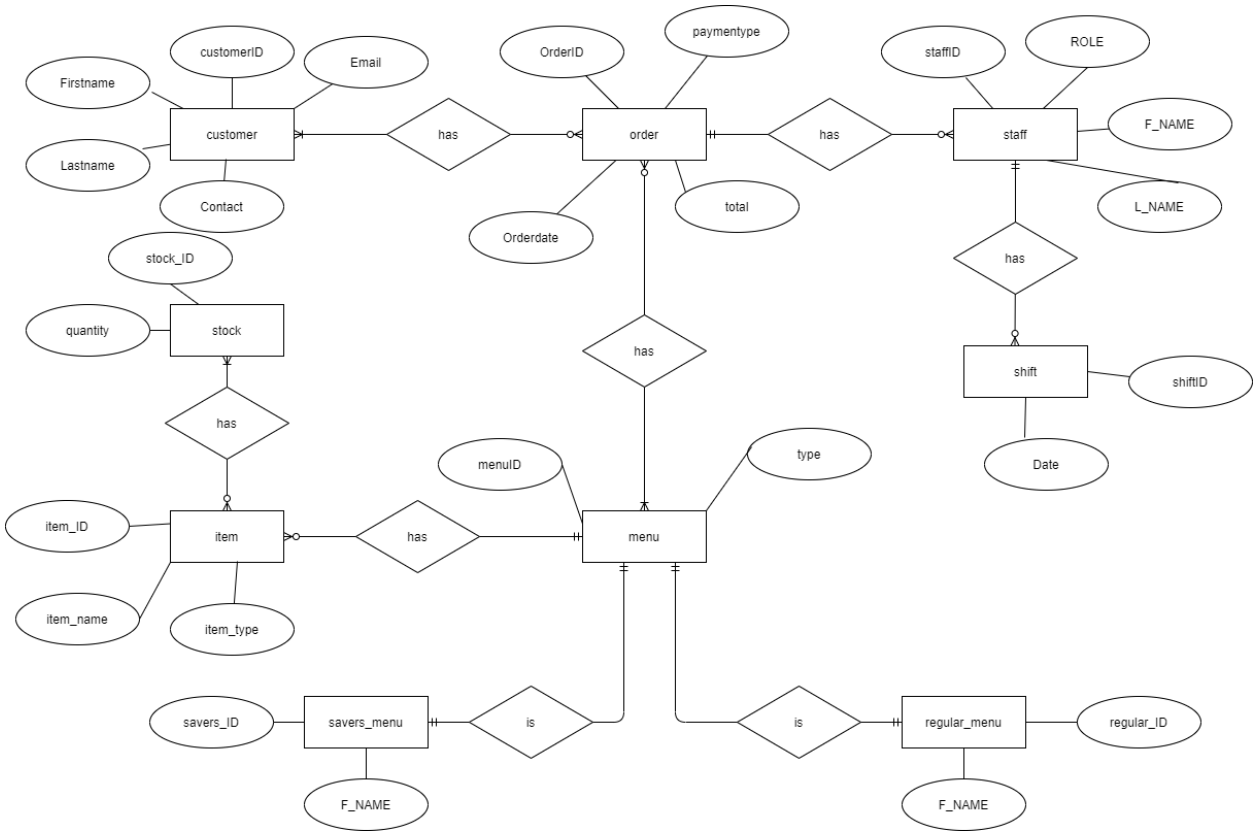
Entity Name: savers menu
Attribute
savers ID
Food name

Entity Name: regular menu
Attribute
Regular ID
Food name

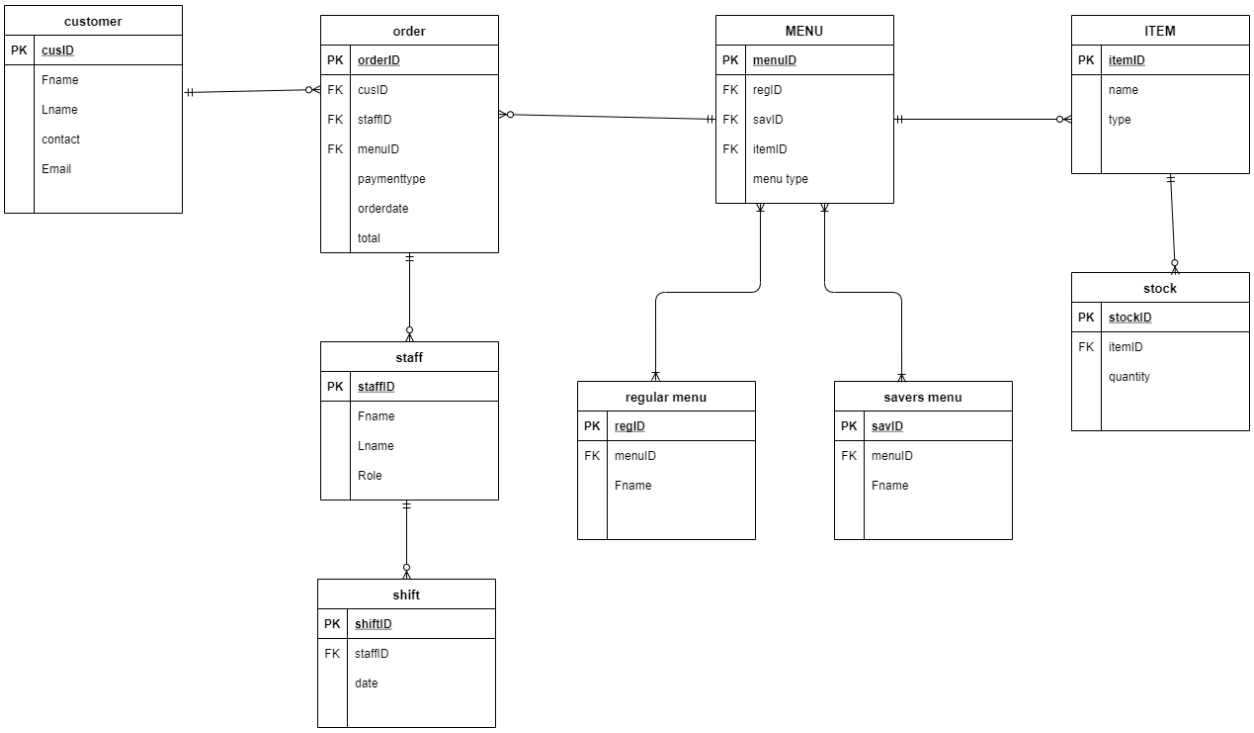
Entity Name: stock
Attribute
stock ID
quantity

Entity Name: item
Attribute
item ID
Item name
Item type

Entity Relationships



3NF ERD

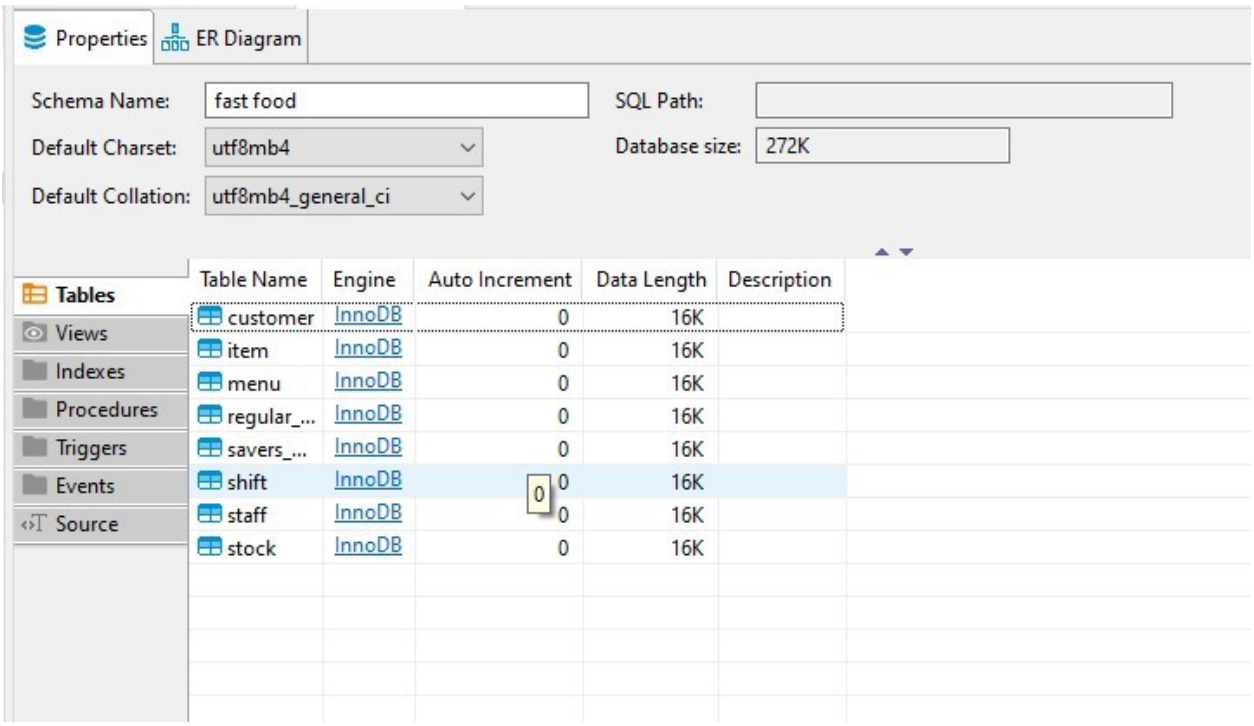

customer
PK cusID
Fname
Lname
contact
Email
order
PK orderID
FK cusID
FK staffID
FK menuID
paymenttype
orderdate
total
MENU
PK menuID
FK regID
FK savID
FK itemID
menu type
ITEM
PK itemID
name
type
stock
PK stockID
FK itemID
quantity
staff
PK staffID
Fname
Lname
Role
regular menu
PK regID
FK menuID
Fname
savers menu
PK savID
FK menuID
Fname
shift
PK shiftID
FK staffID
date

IMPLEMENTATION

```
CREATE TABLE Customer (
CusId INT NOT NULL,
FirstName VARCHAR(40) NOT NULL,
LastName VARCHAR(40) NOT NULL,
Contact VARCHAR(40) NOT NULL,
Email VARCHAR(40) NOT NULL,
PRIMARY KEY (CusId)
);

CREATE TABLE Staff (
StaffId INT NOT NULL,
FName VARCHAR(40) NOT NULL,
LName VARCHAR(40) NOT NULL,
Role VARCHAR(40) NOT NULL,
PRIMARY KEY (StaffId)
);

CREATE TABLE Shift (  ShiftId
INT NOT NULL,  StaffId INT
NOT NULL,
ShiftDate DATE,
PRIMARY KEY (ShiftId),
FOREIGN KEY (StaffId) REFERENCES Staff(StaffId)
);

CREATE TABLE Item (
ItemId INT NOT NULL,
```

```
Name VARCHAR(40) NOT NULL,
Type VARCHAR(40) NOT NULL,
PRIMARY KEY (ItemId)
);

CREATE TABLE Stock (
StockId INT NOT NULL,
ItemId INT NOT NULL,
Quantity INT NOT NULL,
PRIMARY KEY (StockId),
FOREIGN KEY (ItemId) REFERENCES Item(ItemId)
);

CREATE TABLE Savers_Menu (
S_Id INT NOT NULL, f_name VARCHAR(40)
NOT NULL,
PRIMARY KEY (S_Id)
);

CREATE TABLE Regular_Menu (
R_Id INT NOT NULL, f_name VARCHAR(40)
NOT NULL,
PRIMARY KEY (R_Id)
);


CREATE TABLE Menu (
MenuId INT NOT NULL,
ItemId INT NOT NULL,
S_Id INT  NULL,
```

```
R_Id INT  NULL,
Name VARCHAR(40) NOT NULL, menuType
VARCHAR(40) NOT NULL,
PRIMARY KEY (MenuId),
FOREIGN KEY(ItemId) REFERENCES Item(ItemId),
FOREIGN KEY (S_Id) REFERENCES Savers_Menu(S_Id),
FOREIGN KEY (R_Id) REFERENCES Regular_Menu(R_Id)
);

CREATE TABLE orders (
OrderId INT NOT NULL,
StaffId INT NOT NULL,
CusId INT NOT NULL,
MenuId INT NOT NULL,
DateOrder DATE NOT NULL,
PaymentType VARCHAR(5) NOT NULL,
Total DECIMAL(8,2) NOT NULL,
PRIMARY KEY (OrderId),
FOREIGN KEY (StaffId) REFERENCES Staff(StaffId),
FOREIGN KEY (CustId) REFERENCES Customer(CusId),
FOREIGN KEY (MenuId) REFERENCES Menu(MenuId)
);
```


Properties
ER Diagram
Schema Name:
fast food
SQL Path:
Default Charset:
utf8mb4
Database size:
272K
Default Collation:
utf8mb4_general_ci
Tables
Views
Indexes
Procedures
Triggers
Events
Source
Table Name
Engine
Auto Increment
Data Length
Description
customer
InnoDB
0
16K
item
InnoDB
0
16K
menu
InnoDB
0
16K
regular_...
InnoDB
0
16K
savers_...
InnoDB
0
16K
shift
InnoDB
0
16K
staff
InnoDB
0
16K
stock
InnoDB
0
16K

ASSUMPTIONS

For both managers and employees, Shift should have Date, StaffId, and ShiftId. Employees are not required to take orders, and some may have none.

BUSINESS RULES

The regular menu, which includes a breakfast area, closes at 11 a.m. every day; this event must be handled by application development because managing a daily time change is a manual procedure that requires updating the database every day. Because it is assumed that store managers are unfamiliar with database queries, this process can be handled during application development. Another guideline is that replenishment must be announced if the number of burger units reaches 500 and the 1 kilogram bag of chips reaches 200. The relational database system by itself cannot track and trigger a restock notification, but when paired with an application, it can. Furthermore, because this is a time-sensitive operation, the system must check for stock every time an order or an item is used. As a result, this is also a business rule.

JUSTIFICATION OF SELECTED DATA TYPES

A name and a data type must be assigned to each column in a database table. There are three forms of data in MySQL: textual, numeric, and date and time. In each field, the developer must determine the data type to utilize. The data types used in the table CUSTOMER are listed below.

- CusId INT – INT (integer) is a numeric data type. In most cases, identifying numbers take the form of integers. -2147483648 to 2147483647 is the signed range. From 0 to 4294967295 is the unsigned range. The maximum display width is specified by the size parameter (which is 255) (W3Schools, n.d.)

- VARCHAR is used for FirstName, LastName, Contact, and Email. This data type is a string. The VARCHAR type is commonly used to create names or any words. The size parameter specifies the maximum character length of a column, which can range from 0 to 65535.

SAMPLE DATA

```
INSERT INTO Customer (CusId, FirstName, LastName, Contact, Email)
VALUES('100' ,'anush', 'Hurley', '07234567159', 'anushnHush@Gmail.com'),
('101' ,'Josh', 'katar', '01123459929', 'Josh@Yahoo.com'),
('102' ,'Ricky', 'Martine', '02235465744', 'Rikyt@hotline.com'),
('103' ,'Markle', 'Shane', '05476234234', 'MseShane@Gmail.com'),
('104' ,'stepan', 'Watson', '03320484849', 'Stepanil@Gmail.com'),
('105' ,'Billy', 'Gates', '03456743782', 'BillyGat@Gmail.com'),
('106' ,'hovik', 'Warn', '07210304994', 'Warn@Yahoo.com'),
('107' ,'Sam', 'Patrick', '02393942233', 'Samjack@Yahoo.com'),
('108' ,'Jacky', 'Jonson', '03939495955', 'Jacky@Gmail.com'),
('109' ,'Ann', 'Raymond', '07220040504', 'AnnRay@Gmail.com');


INSERT INTO Staff(staffId,FName, LName,role)
VALUES ('1', 'Dawson', 'Craig', 'Manager'),
('2', 'Jesse', 'Mclaughlin', 'Sales'),
('3', 'Danny', 'Marry', 'Sales'),  ('4',
'Debra', 'Day', 'Chef'),
('5', 'Pugh', 'Stevens', 'Sales'),
('6', 'Berry', 'Cross', 'Sales'),
('7', 'Caroll', 'Cross', 'Chef'),
('8', 'Billy' , 'Jonson', 'Sales'),
('9', 'Jack' , 'Rodolfo', 'Sales'),
('10', 'Cross' , 'Swanson', 'Sales');


INSERT INTO Shift (ShiftId, StaffId, ShiftDate)
VALUES ('010', '1', '2020/02/12'),
('011', '2', '2020/02/14'),
```

```
('012', '3', '2020/02/14'),
('013', '4', '2020/02/15'),
('014', '5', '2020/02/14'),
('015', '6', '2020/02/15'),
('016', '7', '2020/02/17'),
('017', '8', '2020/02/17'),
('018', '9', '2020/02/18'),
('019', '10', '2020/02/19');


INSERT INTO Item (ItemId, Name, Type)
VALUES('1', 'Breads', 'Food'),
('2', 'Burgers', 'Food'),
('3', 'Pastry', 'Food'),
('4', 'coke', 'Drink'),
('5', 'donut', 'Food'),
('6', 'milk', 'Drink'),
('7', 'Coffee', 'Drink'),
('8', 'Fries', 'Food'),
('9', 'cake', 'Food'),
('10', 'cupcakes', 'Food');


INSERT INTO Stock (StockId, ItemId, Quantity) VALUES('001', '1', '1000'),
('002', '2', '670'),
('003', '3', '500'),
('004', '4', '150'),
('005', '5', '120'),
```

```
('006', '6', '150'),
('007', '7', '235'),
('008', '8',  '25'),
('009', '9', '42'),
('010', '10', '60');


INSERT INTO Savers_Menu(S_Id, f_name )
VALUES ('0100','Pastry'),
('0101', 'Burgers'),
('0102', 'cake'),
('0103', 'cupcakes'),
('0104', 'Sausage Bun'),
('0105', 'Chocolate Cake'),
('0106', 'Gingerbread Cake'),
('0107', 'Coffee Cake'),
('0108', 'Chocolate Chess Pie'),
('0109', 'Shortbread Tea Cakes');


INSERT INTO Regular_Menu(R_Id, f_name ) VALUES ('0000', 'Pastry'),
('0001', 'Burgers'),
('0002', 'cake'),
('0003', 'cupcakes'),
('0004', 'Sausage Bun'),
('0005', 'Chocolate Cake'),
```

('0006', 'Gingerbread Cake'),

('0007', 'Coffee Cake'),

('0008', 'Chocolate Chess Pie'),

('0009', 'Shortbread Tea Cakes');

INSERT INTO Menu (MenuId, ItemId, S_Id, R_Id, Name, menuType)

VALUES('201', '1', '0100' , '0000', 'Burgers' , 'Regular'),

('202', '2', '0101' , '0001' , 'Burgers', 'Regular'),

('203', '3', '0102' , '0002' , 'Cheesy', 'Regular'),

('204', '4', '0103' , '0003' , 'Crispy wrap', 'Savers'),

('205', '5', '0104' , '0004' , 'Pizza', 'Regular'),

('206', '6', '0105' , '0005' , 'Burgers', 'Savers'),

('207', '7', '0106' , '0006' , 'Soup', 'Savers'),

('208', '8', '0107' , '0007' , 'Burgers', 'Savers'),

('209', '9', '0108' , '0008' , 'Coffe', 'Regular'),

('210', '10', '0109' , '0009' , 'tea' , 'Regular');

INSERT INTO Orders (OrderId, StaffId, CusId, MenuId ,DateOrder, PaymentType, Total) VALUES
('1001', '1', '100', '201', '2020/02/13', 'Card', '1200.00'),

('1002', '2', '101', '202', '2020/02/13', 'Cash', '1300.00'),

('1003', '3', '102', '203', '2020/02/13', 'Cash', '1200.00'),

('1004', '4', '103', '204', '2020/02/14', 'Cash', '1250.00'),

('1005', '5', '104', '205', '2020/02/14', 'Card', '1400.00'),

('1006', '6', '105', '206', '2020/02/15', 'Card', '1385.00'),

('1007', '7', '106', '207', '2020/02/15', 'Cash', '1450.00'), ('1008',
'8', '107', '208', '2020/02/16', 'Card', '1400.00'),
('1009', '9', '108', '209', '2020/02/17', 'Cash', '1540.00'),
('1010', '10', '109', '210', '2020/02/18', 'Card', '1300.00');

DATA SELECTION AND PROJECTION

1. SELECT * FROM customer;

SELECT * FROM customer | Enter a SQL expression to filter results (use Ctrl+Space)

	CusId	FirstName	LastName	Contact	Email
1	100	anush	Hurley	07234567159	anushnHush@Gmail.com
2	101	Josh	katar	01123459929	Josh@Yahoo.com
3	102	Ricky	Martine	02235465744	Rikyt@hotline.com
4	103	Markle	Shane	05476234234	MseShane@Gmail.com
5	104	stepan	Watson	03320484849	Stepanil@Gmail.com
6	105	Billy	Gates	03456743782	BillyGat@Gmail.com
7	106	hovik	Warn	07210304994	Warn@Yahoo.com
8	107	Sam	Patrick	02393942233	Samjack@Yahoo.com
9	108	Jacky	Jonson	03939495955	Jacky@Gmail.com
10	109	Ann	Raymond	07220040504	AnnRay@Gmail.com

Save Cancel Script 200 10 Rows: 1

IST en Writable Smart Insert 1 : 1 : 0 Sel: 0 | 0

2. SELECT CONCAT (FName, " ", LName) AS Name , Role FROM Staff;

staff 1

SELECT CONCAT (FName, " ", LName) AS N | Enter a SQL expression to filter results (use Ctrl+Space)

	Name	Role
1	Dawson Craig	Manager
2	Jesse Mclaughlin	Sales
3	Danny Marry	Sales
4	Debra Day	Chef
5	Pugh Stevens	Sales
6	Berry Cross	Sales
7	Caroll Cross	Chef
8	Billy Jonson	Sales
9	Jack Rodolfo	Sales
10	Cross Swanson	Sales

Save Cancel Script 200 10 Rows: 1

3. SELECT * FROM orders WHERE staffId='1' ORDER BY DateOrder ;

orders 1

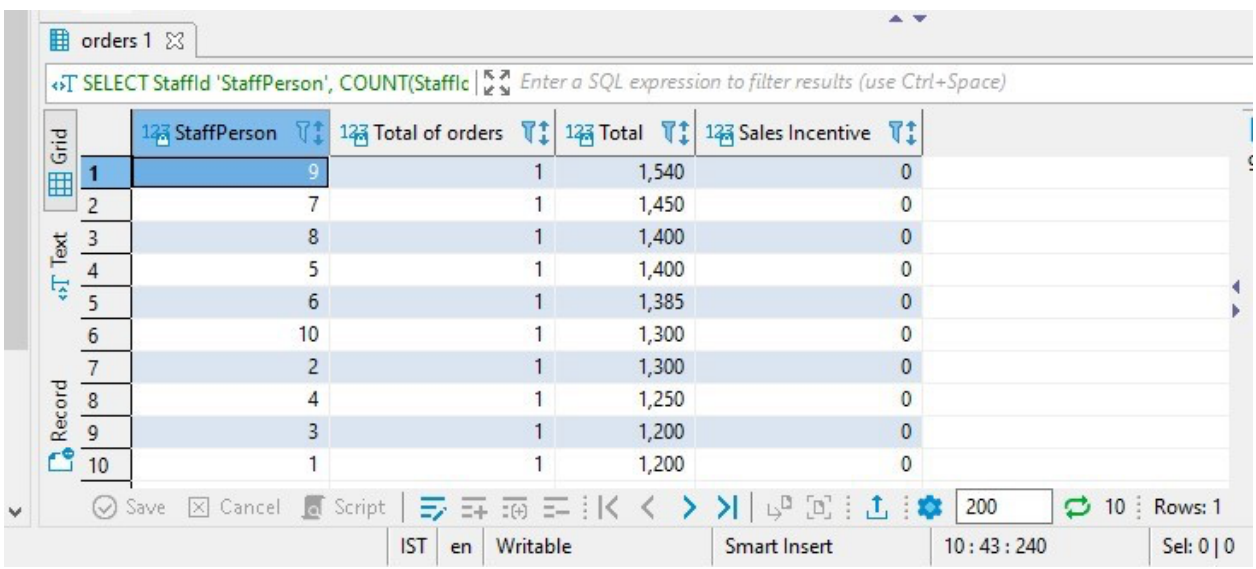
SELECT * FROM orders WHERE staffId='1' C | Enter a SQL expression to filter results (use Ctrl+Space)

	OrderId	StaffId	CusId	MenuId	DateOrder	PaymentType
1	1,001	1	100	201	2020-02-13	Card

Save Cancel Script 200 1 Rows: 1

4. SELECT

```
    StaffId 'StaffPerson',
    COUNT(StaffId) 'Total of orders',
    SUM(Total) 'Total',
    CASE
        WHEN SUM(Total) > 5000 THEN SUM(Total) * 0.1
        ELSE 0
    END 'Sales Incentive'
FROM orders o
GROUP BY StaffId ORDER BY SUM(Total) DESC;
```



	StaffPerson	Total of orders	Total	Sales Incentive
1	9	1	1,540	0
2	7	1	1,450	0
3	8	1	1,400	0
4	5	1	1,400	0
5	6	1	1,385	0
6	10	1	1,300	0
7	2	1	1,300	0
8	4	1	1,250	0
9	3	1	1,200	0
10	1	1	1,200	0

CONCLUSION

FastBurgers Inc. required the creation of a database application to handle special requirements. FastBurgers Inc. required a database application, so the above-mentioned queries and data were used to construct it. Furthermore, the program performs as expected, as evidenced by the results of various searches.