

ASSIGNMENT/ASSESSMENT ITEM COVER SHEET

Student Name:

FIRST NAME

FAMILY / LAST NAME

Student Number:

Email:

Course Code

Course Title

(Example)

A	B	C	D	1	2	3	4
---	---	---	---	---	---	---	---

(Example)

Intro to University

Campus of Study:

(eg Callaghan, Ourimbah, Port Macquarie)

Assessment Item Title:

Due Date/Time:

Tutorial Group (If applicable):

Word Count (If applicable):

Lecturer/Tutor Name:

Extension Granted:

Yes

No

Granted Until:

Please attach a copy of your extension approval

NB: STUDENTS MAY EXPECT THAT THIS ASSIGNMENT WILL BE RETURNED WITHIN 3 WEEKS OF THE DUE DATE OF SUBMISSION

Please tick box if applicable

Students within the Faculty of Business and Law, Faculty of Science and Information Technology, Faculty of Engineering and Built Environment and the School of Nursing and Midwifery:

I verify that I have completed the online Academic Q&A Module and adhered to its principles

Students within the School of Education:

"I understand that a minimum standard of correct referencing and academic literacy is required to pass all written assignments in the School of Education; and I have read and understood the School of Education Course Outline Policy Supplement, which includes important information related to assessment policies and procedures.

I declare that this assessment item is my own work unless otherwise acknowledged and is in accordance with the University's academic integrity policy available from the Policy Library on the web at <http://www.newcastle.edu.au/policylibrary/000608.html>
I certify that this assessment item has not been submitted previously for academic credit in this or any other course. I certify that I have not given a copy or have shown a copy of this assessment item to another student enrolled in the course.

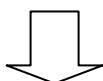
I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:

- Reproduce this assessment item and provide a copy to another member of the Faculty; and/or
- Communicate a copy of this assessment item to a plagiarism checking service (which may then retain a copy of the item on its database for the purpose of future plagiarism checking).
- Submit the assessment item to other forms of plagiarism checking.

I certify that any electronic version of this assessment item that I have submitted or will submit is identical to this paper version.

Turnitin ID:
(if applicable)

DATE
STAMP
HERE



Insert
this
way

Signature:

hoby

Date:

INFO6001 – Database Management 1

Assignment 1

Project: Database design of Superdeli Pizza

Requirement Analysis and Conceptual Design

Submitted by

Rosy Mohanty

Student id- c3411123

Email id- rosy.mohanty@uon.edu.au

Table of Contents

Preface	3
Part 1: Requirements	4
Data Requirements	4
Transaction Requirements	6
Business Rules	18
Part 2: EER Model with Data Dictionary	19
Data Dictionary.....	19
Entity Types.....	19
Relationships.....	21
Attributes:.....	22
Summary	35

Preface

This is the requirements analysis and conceptual design for the implementation of the database of Superdeli Pizzas.

The requirements analysis includes the data requirements, transaction requirements and business rules. This will ensure that all the data required to be stored is identified, can be manipulated and is managed according to the store's business policies.

The conceptual design includes an extended entity relationship diagram in UML which describes the required entities, their attributes and their relationships. It is further clarified with a data dictionary for the entities, attributes and also the relationships. This conceptual model was developed from the requirements analysis.

This document includes detail explanation of the above mentioned topic by dividing them into sub sections. The parts of requirements are order processing, menu items & ingredients and employees. Then there is transaction requirement section having all the queries. The last section is EER model which has EER model and data dictionary.

Part 1: Requirements

Data Requirements

➤ Order Data

An order records all the menu orders that have been made so far. Superdeli Pizza takes orders via phone as well as through walk-in customers. Superdeli Pizza provides both delivery and pickup services. For each order, there is an order number, order date and time, customer info, staff info, items ordered, type of order. Also, order can be of two types i.e. phone and walkin.

- Phone

The order that are made on phone calls. For each phone order, the time the call was answered as well as the time the call was terminated is recorded.

- TakeAway

When customer give order on phone call and come by himself to pick it up.

- Delivery

When customer give order on phone call and give his/her details to deliver the order. For each order, time of order, address of customer and driverLicence.

- Walkin

Order that customers come and give. For each walkin order, the time the customer walks in is recorded.

➤ Customer Data

When a customer orders, the customer's phone number is entered to the system along with the id of the staff taking the order. For each customer, there is a customer ID, customer name, address, phone, what has been ordered and status. By status it is meant if the customer is an existing customer or new.

- Existing Customer

If the customer has previously ordered, the name and address appear on the screen. The customer is then asked for his name and address and then the order is taken.

- **New Customer**

If the customer has not ordered before or if the name and address given do not correspond with what are recorded in the computer, a new customer record is created and the order is taken.

- **Employee**

Workers that work on pizza shop are employees. For each employee, there is an employee number, firstname, lastname, postal address, contact number, tax file number, bank details (bank code, bank name, account number), payment rate, status, and a description. Drivers also have a driver's license number. Employees at the store can be divided into two types: those who work in the shop are paid hourly and those who carry out deliveries are paid by the number of deliveries.

- **Staff Payment Data**

Payment rates for shop workers and drivers are maintained in the database. Employee payments are made for each shift to the employee's bank account. Employee payment record needs to be maintained in the database. It includes gross payment, tax withheld, total amount paid, payment date, payment period start date, payment period end date, and bank details of the employee.

- **Driver Payment Data**

Payment data for each driver is stored for how many deliveries he has made.

- **Shop Worker Payment Data**

Payment data of every worker is stored in this with how many hours an employee has work.

- **Shift Data**

Hours are not regular and a record is kept for each time an employee works- a shift (start date, start time, end date, end time). The orders a driver delivers during a shift is kept on the record which include delivery address and time for each delivery.

- **Driver Shift Data**

Shift data for drivers are stored separately with deliveryAddress and deliverTime for each delivery.

- **Shop Worker Data**

Shift data for workers who work in store are stored with hourlyRate.

➤ **Menu Item Data**

Each item in the menu has an item code (unique), name, size and a current selling price. An item in the menu is made up of a number of ingredients. The ingredients and quantities used for the item are recorded in the database.

➤ **Ingredients data**

Each ingredient has a code (unique), name, type, description, stock level at current stocktake period, date last stocktake was taken, suggested current stock level, reorder level, and a list of suppliers who supply the ingredients.

➤ **Suppliers Data**

A supplier can supply many ingredients. Each ingredient can be supplied by **many** suppliers. For each customer, there is a Supplier number, name, address, phone, email and contact person.

➤ **Ingredient Order Data**

A stocktake is taken each week, where the actual levels of ingredients in store (i.e., stock levels of ingredients at current stocktake period) are collected. The actual levels of ingredients in store, together with suggested current stock levels and reorder levels, are used by the store manager to order ingredients for the following week. Information about ingredient orders needs to be maintained in the database, including order number, date of the order, date received order, total amount, order status, description, quantity and price of all ingredients, supplier number, and ingredient code

Transaction Requirements

Data Manipulation

Pizza database

Creating Tables

```
DROP TABLE DeliveryOrder
DROP TABLE Employees
DROP TABLE Driver
DROP TABLE shopWorker
DROP TABLE Payment
DROP TABLE driverPaymentData
DROP TABLE shopWorkerPaymentData
DROP TABLE shiftData
DROP TABLE driverShiftData
DROP TABLE shopWorkershiftData
DROP TABLE ingredientData
DROP TABLE ingredientOrderData
DROP TABLE Suppliers
DROP TABLE PickupOrder
```

```

DROP TABLE WalkInOrder
DROP TABLE PhoneOrder
DROP TABLE Orders
DROP TABLE Customer
DROP TABLE newCustomer
DROP TABLE previousCustomer
DROP TABLE QOrderMenuItem
DROP TABLE menuIngredientData
DROP TABLE ingredientToIngredientOrderData
DROP TABLE ingredientSupplierData
DROP TABLE MenuItem
--DROP TABLE
--DROP TABLE
--DROP TABLE

CREATE TABLE Customer(
    customerId CHAR(10) PRIMARY KEY,
    firstName VARCHAR(20) NOT NULL,
    lastName VARCHAR(20) NOT NULL,
    Address VARCHAR(200) NOT NULL,
    phoneNumber VARCHAR(10) NOT NULL,
    isHoax VARCHAR(10) DEFAULT 'unverified',
    CHECK(isHoax IN ('verified', 'unverified'))
);

CREATE TABLE newCustomer(
    customerId CHAR(10) PRIMARY KEY,

```



```

        FOREIGN KEY(customerId) REFERENCES Customer(customerId) ON UPDATE
CASCADE ON DELETE CASCADE,
    );

CREATE TABLE previousCustomer(
    customerId CHAR(10) PRIMARY KEY,
    FOREIGN KEY(customerId) REFERENCES Customer(customerId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE Employees(
    employeeId CHAR(10) PRIMARY KEY,
    firstName VARCHAR(20) NOT NULL,
    lastName VARCHAR(20) NOT NULL,
    EmployeeType VARCHAR(20),
    Address VARCHAR(200) NOT NULL,
    phoneNumber VARCHAR(10) NOT NULL,
    TFN VARCHAR(10) NOT NULL,
    bankCode VARCHAR(10) NOT NULL,
    accNo VARCHAR(20) NOT NULL,
    bankName VARCHAR(20) NOT NULL,
);

CREATE TABLE Driver(
    employeeId CHAR(10) PRIMARY KEY,
    driverLicence VARCHAR(20) NOT NULL,
    noOfDeliveries CHAR(5),
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE shopWorker(
    employeeId CHAR(10) PRIMARY KEY,
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE Payment(
    paymentId CHAR(10) PRIMARY KEY,
    employeeId CHAR(10),
    grossPay FLOAT,
    taxWithHeld FLOAT,
    payEndDate DATE,
    totalAmountPaid FLOAT,
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE driverPaymentData(
    paymentId CHAR(10) PRIMARY KEY,
    FOREIGN KEY(paymentId) REFERENCES Payment(paymentId) ON UPDATE CASCADE
ON DELETE CASCADE,
);

CREATE TABLE shopWorkerPaymentData(
    paymentId CHAR(10) PRIMARY KEY,
    FOREIGN KEY(paymentId) REFERENCES Payment(paymentId) ON UPDATE CASCADE
ON DELETE CASCADE,
);

```

```

CREATE TABLE shiftData(
    shiftId CHAR(10) PRIMARY KEY,
    employeeId CHAR(10),
    startDate DATE,
    startTime TIME,
    endDate DATE,
    endTime TIME,
    shiftType VARCHAR(20),
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE driverShiftData(
    shiftId CHAR(10) PRIMARY KEY,
    employeeId CHAR(10),
    FOREIGN KEY(shiftId) REFERENCES shiftData(shiftId) ON UPDATE CASCADE ON
DELETE CASCADE,
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE NO
ACTION ON DELETE NO ACTION,
);

CREATE TABLE shopWorkerShiftData(
    shiftId CHAR(10),
    employeeId CHAR(10),
    hourlyRate FLOAT,
    FOREIGN KEY(shiftId) REFERENCES shiftData(shiftId) ON UPDATE CASCADE ON
DELETE CASCADE,
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE NO
ACTION ON DELETE NO ACTION,
);

CREATE TABLE ingredientData(
    ingredientId CHAR(10) PRIMARY KEY,
    ingredienName VARCHAR(20),
    type VARCHAR(20),
    stockLevel VARCHAR(10),
    lastStock VARCHAR(10),
    suggStock VARCHAR(10),
    reorderLevel VARCHAR(10),
);

CREATE TABLE ingredientOrderData(
    ingredientOrderId CHAR(10) PRIMARY KEY,
    employeeId CHAR(10),
    OrderDate DATE,
    dateRecievedOrder DATE,
    totalAmount FLOAT,
    orderStatus VARCHAR(10),
    Quantity CHAR(5),
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE Suppliers(
    supplierId CHAR(10) PRIMARY KEY,
    Name VARCHAR(20),
    Address VARCHAR(200) NOT NULL,

```

```

        phoneNumber VARCHAR(10) NOT NULL,
        Email VARCHAR(20) NOT NULL,
        contactPerson VARCHAR(20),
    );

CREATE TABLE Orders(
    OrderId CHAR(10) PRIMARY KEY,
    employeeId CHAR(10),
    OrderDateTime DATETIME2,
    TotalAmountDue FLOAT,
    PaymentMethod VARCHAR(20) NOT NULL,
    PaymentApprovalNo VARCHAR(20) NOT NULL,
    OrderStatus VARCHAR(20),
    customerId CHAR(10),
    Quantity CHAR(5),
    FOREIGN KEY(CustomerId) REFERENCES Customer (CustomerId) ON UPDATE
CASCADE ON DELETE CASCADE,
    -- not to implement here FOREIGN KEY(StaffId) REFERENCES
InStore(StaffId) ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY(employeeId) REFERENCES Employees(employeeId) ON UPDATE
CASCADE ON DELETE CASCADE,
);

CREATE TABLE WalkInOrder(
    OrderId CHAR(10) PRIMARY KEY,
    WalkInTime DATETIME2,
    FOREIGN KEY(OrderId) REFERENCES Orders(OrderId) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE PhoneOrder(
    OrderId CHAR(10) PRIMARY KEY,
    timeCallAnswered DATETIME2,
    TimeCallTerminated DATETIME2,
    FOREIGN KEY(OrderId) REFERENCES Orders(OrderId) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE PickupOrder(
    OrderId CHAR(10) PRIMARY KEY,
    PickupTime DATETIME2,
    FOREIGN KEY(OrderId) REFERENCES PhoneOrder(OrderId) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE DeliveryOrder(
    OrderId CHAR(10) PRIMARY KEY,
    shiftId CHAR(10),
    Address VARCHAR(200) NOT NULL,
    DeliveryTime DATETIME2,
    FOREIGN KEY(OrderId) REFERENCES PhoneOrder(OrderId) ON UPDATE CASCADE ON
DELETE CASCADE,
    -- not to implement here - Foreign Key ShiftId references DriverShift
(ShiftId) ON UPDATE CASCADE, ON DELETE CASCADE
    FOREIGN KEY(shiftId) REFERENCES shiftData(shiftId) ON UPDATE NO ACTION
ON DELETE NO ACTION,
)

```

```

CREATE TABLE MenuItem(
    ItemNo CHAR(10) PRIMARY KEY,
    Name VARCHAR(20) NOT NULL,
    Size VARCHAR(10) NOT NULL,
    Price FLOAT,
    CurrentSellingPrice FLOAT,
    Description VARCHAR(50)
);

CREATE TABLE QOrderMenuItem(
    ItemNo CHAR(10) NOT NULL,
    OrderId CHAR(10) NOT NULL,
    quantity int NOT NULL,
    PRIMARY KEY (ItemNo, OrderId, quantity),
    FOREIGN KEY (OrderId) REFERENCES Orders (OrderId) ON UPDATE CASCADE ON
DELETE CASCADE,
    FOREIGN KEY (ItemNo) REFERENCES MenuItem (ItemNo) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE menuIngredientData(
    ItemNo CHAR(10) NOT NULL,
    ingredientId CHAR(10) NOT NULL,
    PRIMARY KEY (ItemNo, ingredientId),
    FOREIGN KEY (ingredientId) REFERENCES ingredientData (ingredientId) ON
UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (ItemNo) REFERENCES MenuItem (ItemNo) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE ingredientToIngredientOrderData(
    orderId CHAR(10) NOT NULL,
    ingredientId CHAR(10) NOT NULL,
    PRIMARY KEY (orderId, ingredientId),
    FOREIGN KEY (ingredientId) REFERENCES ingredientData (ingredientId) ON
UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (orderId) REFERENCES Orders (orderId) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE ingredientSupplierData(
    supplierId CHAR(10) NOT NULL,
    ingredientId CHAR(10) NOT NULL,
    PRIMARY KEY (supplierId, ingredientId),
    FOREIGN KEY (ingredientId) REFERENCES ingredientData (ingredientId) ON
UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (supplierId) REFERENCES Suppliers (supplierId) ON UPDATE
CASCADE ON DELETE CASCADE
);

```

- Putting Values to table

```

-- LOADING SAMPLE DATA

-- CUSTOMER TABLE
INSERT INTO Customer VALUES ('C012345678', 'John', 'MacDonald', '55 Cliff
Street, Coins NSW, 2995', '0222223333', 'verified');

```

```

INSERT INTO Customer VALUES ('C012345679', 'Peter', 'Parker', '56 Fort Street,
Blacktown NSW, 2965', '022223334', 'unverified');
INSERT INTO Customer VALUES ('C012345677', 'Anna', 'Shelbey', '88 North
Street, Punchbowl NSW, 2005', '022223335', 'verified');

-- NEWCUSTOMER TABLE
INSERT INTO newCustomer VALUES ('C012345679');

-- PREVIOUSCUSTOMER TABLE
INSERT INTO previousCustomer VALUES ('C012345678');
INSERT INTO previousCustomer VALUES ('C012345677');

-- EMPLOYEES TABLE
INSERT INTO Employees VALUES ('E012233445', 'Najam', 'Hashx', 'Manager', '55
Dellwood Street, Coins NSW, 2995', '0233445555', '111222', '1234', '00112233',
'Common Wealth');
INSERT INTO Employees VALUES ('E012233446', 'Bob', 'Marley', 'InStore', '65
Pecker Street, Jesmond NSW, 2985', '0233445556', '222333', '4567', '00224466',
'NAB');
INSERT INTO Employees VALUES ('E012233447', 'Dwayne', 'Johnson', 'Driver', '75
London Street, Paramata NSW, 2955', '0233445557', '333444', '7899', '00113355',
'ANZ');

-- DRIVER TABLE
INSERT INTO driver VALUES ('E012233447', '1122-3344', '5');

-- SHOPWORKER TABLE
INSERT INTO shopWorker VALUES ('E012233445');
INSERT INTO shopWorker VALUES ('E012233446');

-- PAYMENT TABLE
INSERT INTO Payment VALUES ('P012233445', 'E012233445', 50000.0, 2.5, '2019-4-
12', 25000.0);
INSERT INTO Payment VALUES ('P012233446', 'E012233446', 40000.0, 3.5, '2019-5-
12', 15000.0);
INSERT INTO Payment VALUES ('P012233447', 'E012233447', 30000.0, 2.7, '2019-6-
12', 20000.0);

-- DRIVERPAYMENTDATA TABLE
INSERT INTO driverPaymentData VALUES ('P012233447');

--SHOPWORKERPAYMENTDATA TABLE
INSERT INTO shopWorkerPaymentData VALUES ('P012233446');
INSERT INTO shopWorkerPaymentData VALUES ('P012233445');

-- SHIFTDATA TABLE

```

```

INSERT INTO shiftData VALUES ('S012233445', 'E012233446', '2019-7-12',
'10:0:0', '2019-7-12', '20:0:0', 'Shop Worker');
INSERT INTO shiftData VALUES ('S012233446', 'E012233447', '2019-8-12',
'12:0:0', '2019-8-12', '18:0:0', 'Driver');
INSERT INTO shiftData VALUES ('S012233447', 'E012233445', '2019-9-12',
'11:0:0', '2019-9-12', '19:0:0', 'Shop Worker');

-- DRIVERSHIFTDATA TABLE
INSERT INTO driverShiftData VALUES ('S012233446', 'E012233447');

-- SHOPWORKERSHIFTDATA TABLE
INSERT INTO shopWorkerShiftData VALUES ('S012233445', 'E012233446', 20.0);
INSERT INTO shopWorkerShiftData VALUES ('S012233447', 'E012233445', 25.0);

-- INGREDIENTDATA TABLE
INSERT INTO ingredientData VALUES ('I012233445', 'Chilli', 'Spices',
'low', 'full', 'max', 'min');
INSERT INTO ingredientData VALUES ('I012233446', 'Italian Bread', 'Bread',
'medium', 'full', 'max', 'medium');
INSERT INTO ingredientData VALUES ('I012233447', 'Chicken', 'meat', 'very
low', 'full', 'max', 'min');

-- INGREDIENTORDERDATA TABLE
INSERT INTO ingredientOrderData VALUES ('I012233445', 'E012233445', '2019-9-
15', '2019-9-20', 55.0, 'Delivered', '10');
INSERT INTO ingredientOrderData VALUES ('I012233446', 'E012233445', '2019-10-
10', '2019-10-18', 85.0, 'Inprogress', '15');
INSERT INTO ingredientOrderData VALUES ('I012233447', 'E012233446', '2019-9-
05', '2019-9-11', 40.0, 'On The Way', '08');

-- SUPPLIERS TABLE
INSERT INTO Suppliers VALUES ('S012345678', 'Nick', '40 Buxon Street, Rodes
NSW, 2555', '1222224444', 'nick@gmail.com', 'Self');
INSERT INTO Suppliers VALUES ('S012345679', 'Sherjil', '50 End Street, Central
NSW, 2925', '1222225555', 'sherjil@yahoo.com', 'Manager');
INSERT INTO Suppliers VALUES ('S012345677', 'Joe', '80 West Street, Auburn
NSW, 2010', '1222229999', 'joe@gmail.com', 'Self');

-- ORDER TABLE
INSERT INTO Orders VALUES ('C012233445', 'E012233446', '2019-10-12 10:0:0',
45.0, 'Cash', '0233445566', 'picked', 'C012345678', '3');
INSERT INTO Orders VALUES ('C012233446', 'E012233446', '2019-11-20 10:0:0',
85.0, 'Card', '0233445577', 'phone', 'C012345679', '2');
INSERT INTO Orders VALUES ('C012233447', 'E012233445', '2019-12-10 10:0:0',
55.0, 'Card', '0233445588', 'phone', 'C012345677', '1');

-- WalkIn TABLE
INSERT INTO WalkInOrder VALUES ('C012233445', '2019-10-12 10:0:0');

-- PHONEORDER TABLE
INSERT INTO PhoneOrder VALUES ('C012233446', '2019-10-13 10:0:0', '2019-10-13
10:10:0');
INSERT INTO PhoneOrder VALUES ('C012233447', '2019-10-19 18:30:0', '2019-10-19
18:35:0');

-- DELIVERY TABLE
INSERT INTO DeliveryOrder VALUES ('C012233446', 'S012233446', '15 Coles Street,
Points NSW, 6555', '2019-10-13 11:0:0');

-- PICKUPORDER TABLE
INSERT INTO PickupOrder VALUES ('C012233447', '2019-10-19 19:15:0');

-- MenuItem TABLE

```

```

INSERT INTO MenuItem VALUES ('I01245678', 'beef', 'Large', 13.8, 10.0, 'Yummy Beef');
INSERT INTO MenuItem VALUES ('I01245679', 'chicken', 'Large', 12.8, 9.0, 'Tandoori Chicken');
INSERT INTO MenuItem VALUES ('I01245680', 'chicken', 'medium', 14.8, 12.0, 'Fajita');

-- OrderMenuItem TABLE
INSERT INTO QOrderMenuItem VALUES ('I01245678', 'C012233445', 4);

-- MENU-INGREDIENT-DATA TABLE
INSERT INTO menuIngredientData VALUES ('I01245679', 'I012233446');
INSERT INTO menuIngredientData VALUES ('I01245678', 'I012233445');
INSERT INTO menuIngredientData VALUES ('I01245680', 'I012233447');

-- INGREDIENT-TO-INGREDIENT-ORDER-DATA TABLE
INSERT INTO ingredientToIngredientOrderData VALUES ('C012233447', 'I012233447');

-- INGREDIENT-SUPPLIER-DATA TABLE
INSERT INTO ingredientSupplierData VALUES ('S012345678', 'I012233445');
INSERT INTO ingredientSupplierData VALUES ('S012345677', 'I012233447');
INSERT INTO ingredientSupplierData VALUES ('S012345679', 'I012233446');

```

Inserting Data

- Inserting details of new orders.

```

INSERT INTO Orders (OrderNo, OrderDateTime, CustomerInfo, StaffInfo, TypeOfOrder,
TotalAmountDue, PaymentMethod, PaymentApprovalNo, MenuItemInfo,
QuantityOfEachOrderedMenuItem, OrderStatus)
VALUES (1, '2022-02-17 10:30:00', 'John Smith', 'Jane Doe', 'phone', 25.99, 'credit card', '1234567890',
'pizza, soda', '1, 2', 'in progress');

```

- Inserting details of new Customers.

```

INSERT INTO Customers (CustomerNumber, FirstName, LastName, PhoneNo, Email, Address,
PostalCode)
VALUES (1, 'John', 'Smith', '555-1234', 'john.smith@example.com', '123 Main St', '12345');

```

- Inserting new employees' details.

```

INSERT INTO Employees (EmployeeNumber, FirstName, LastName, PhoneNo, Email, Address,
PostalCode, StartingDate, HourlyWage, Position)
VALUES (1, 'Jane', 'Doe', '555-5678', 'jane.doe@example.com', '456 Maple Ave', '67890', '2021-01-01',

```

15.

- Inserting details of new menu items.

```
INSERT INTO MenuItem (ItemNumber, ItemName, Description, Price, Ingredients)
VALUES (1, 'Pepperoni Pizza', 'Pizza with tomato sauce, cheese, and pepperoni', 12.99, 'tomato sauce, mozzarella cheese, pepperoni');
```

- Inserting new ingredients' data.

```
INSERT INTO Ingredient (IngredientNumber, IngredientName, SupplierInfo, CostPerUnit, StockUnit)
VALUES (1, 'Mozzarella Cheese', 'Acme Food Distributors', 2.99, 'lb');
```

- Inserting new payment data.

```
INSERT INTO Payment (PaymentNumber, PaymentDateTime, PaymentMethod, PaymentAmount, PaymentApprovalNo)
VALUES (1, '2022-02-17 11:00:00', 'credit card', 25.99, '1234567890');
```

- Inserting details of new stocks.

```
INSERT INTO Stock (WeekNumber, IngredientNumber, StockUnit, StockLevel, StockValue)
VALUES (7, 1, 'lb', 10.5, 31.39);
```

Updating and Deleting existing Data

- Updating and deleting details of existing orders.

```
UPDATE Orders
SET TotalAmountDue = 25.99, PaymentMethod = 'credit card', PaymentApprovalNo = '1234'
WHERE OrderNo = 1001;
```

- Updating and deleting Customers data.

```
DELETE FROM Customers
WHERE CustomerNumber = 101;
```

- Updating and deleting employees' details.


```
DELETE FROM Employees
WHERE EmployeeNumber = 201;
```

- Updating and deleting menu items details.

```
DELETE FROM MenuItem
WHERE ItemNumber = 301;
```

- Updating and deleting ingredients' information.

```
DELETE FROM Ingredients
WHERE IngredientNumber = 401;
```

- Updating and deleting stock data.

```
DELETE FROM Stock
WHERE IngredientNumber = 401 AND WeekNumber = 5;
```

- Updating and deleting payment data.

```
DELETE FROM Payments
WHERE OrderNo = 1001;
```

Queries

- Updating details of Employees using EmployeeNumber.

```
UPDATE Employees
SET FirstName = 'John', LastName = 'Doe', PostalAddress = '123 Main St', ContactNumber = '555-1234',
TaxFileNumber = '123-45-6789', BankDetails = 'ABC Bank, 123456789'
WHERE EmployeeNumber = 1;
```

- Updating details of customers using CustomerNumber.

```
UPDATE Customers
SET FirstName = 'Jane', LastName = 'Doe', PostalAddress = '456 Elm St', ContactNumber = '555-4321'
WHERE CustomerNumber = 1;
```

- Deleting the items which are no longer in the menu.

```
DELETE FROM MenuItem
WHERE ItemNumber NOT IN (SELECT ItemNumber FROM Orders);
```

- Deleting the ingredients which are not used anymore

```
DELETE FROM Ingredients
WHERE IngredientNumber NOT IN (SELECT IngredientNumber FROM MenuItemIngredients);
```

- Searching an order based on OrderNumber.

```
SELECT *
FROM Orders
WHERE OrderNumber = 1234;
```

- Searching a customer based on CustomerNumber.

```
SELECT *
FROM Customers
WHERE CustomerNumber = 5678;
```

- Searching an item in menu based on ItemNumber.

```
SELECT *
FROM MenuItem
WHERE ItemNumber = 7890;
```

- Searching an ingredient based on IngredientNumber.

```
SELECT *
FROM Ingredients
WHERE IngredientNumber = 2345;
```

- Listing the orders on the basis of TypeOfOrder.

```
SELECT *
FROM Orders
ORDER BY TypeOfOrder;
```

- Listing staff members which have worked for the lowest hours.

```
SELECT *
FROM Employees
WHERE Type = 'Instore' AND Status = 'Active'
ORDER BY PaymentRate
LIMIT 1;
```

- Searching an employee based on EmployeeNumber.

```
SELECT *
FROM Employees
WHERE EmployeeNumber = 2;
```

- Searching a stock based on WeekNumber and StockUnit.

```
SELECT *
FROM Stock
WHERE WeekNumber = 5 AND StockUnit = 'Tomatoes';
```

- Searching a payment based on PaymentNumber.

```
SELECT *
FROM Payments
WHERE PaymentNumber = 6789;
```

- Listing menu items which contains particular ingredients.

```
SELECT MenuItemName
FROM MenuItems
WHERE ItemNumber IN (SELECT ItemNumber FROM MenuItemIngredients WHERE IngredientNumber = 1234);
```

- Searching an order based on EmployeeNumber.

```
SELECT *
FROM Orders
WHERE EmployeeNumber = 3;
```

- Listing menu items which are low in quantity in the stock.

```
SELECT MenuItemName
FROM MenuItems
WHERE ItemNumber IN (SELECT ItemNumber FROM MenuItemIngredients WHERE IngredientNumber IN (SELECT IngredientNumber FROM Stock WHERE Quantity < 10));
```

- Listing staff members which have started working on same day on the basis of StartingDate.

```
SELECT *
FROM Employees
WHERE StartingDate = '2022-01-01';
```

- Updating the stock level of ingredients in the stock.

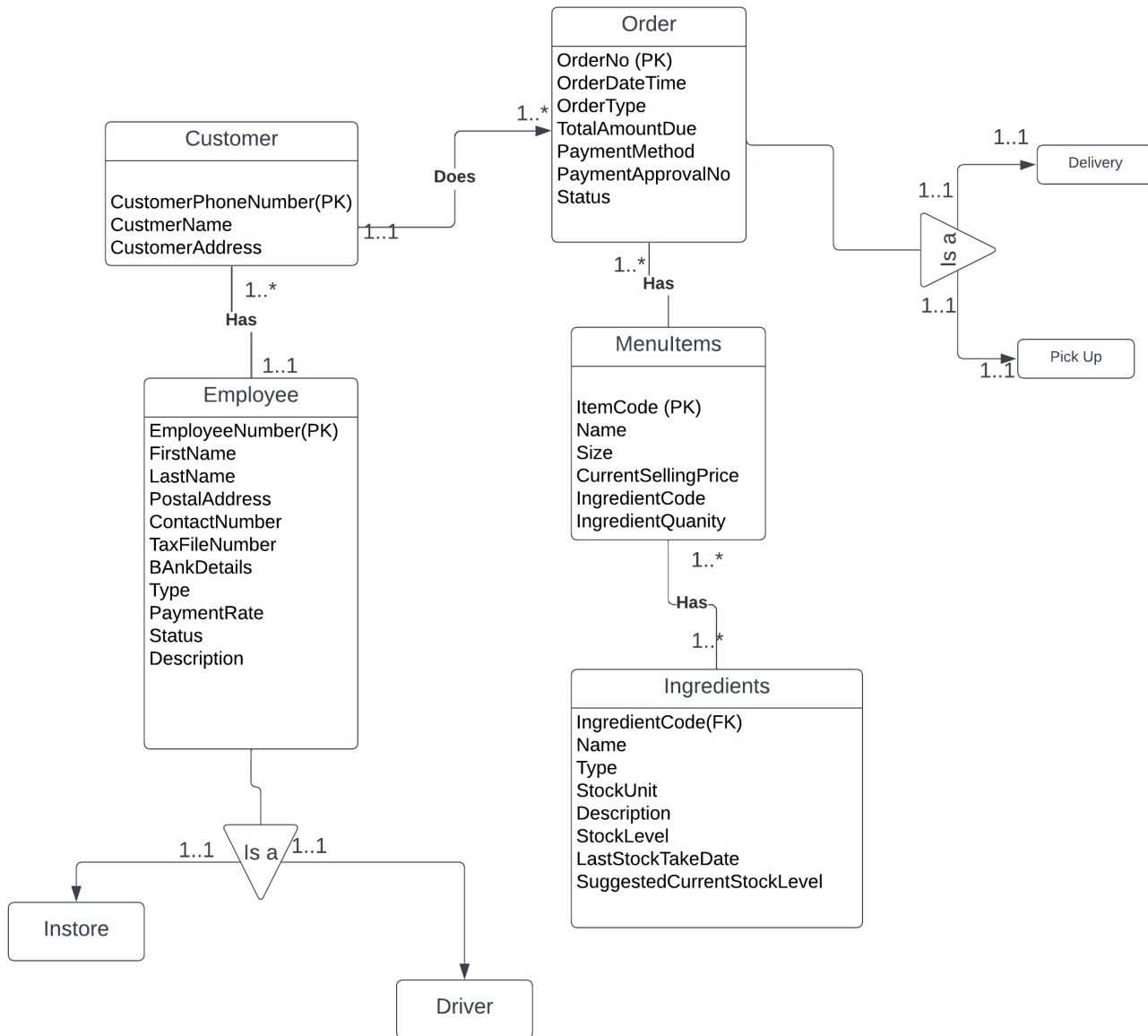
```
UPDATE Stock
SET Quantity = 15
WHERE IngredientNumber = 1234 AND WeekNumber = 6;
```

Business Rules

- A customer can place an order either in-store or over the phone.
- An order can be either for pick up or delivery.
- For phone orders, the customer must provide their delivery address, and the order must include the time the call was answered, the time the call was terminated, and whether it is for pick up or delivery.
- For pick-up orders, the order must include the pick-up time.
- For delivery orders, the order must include the delivery time, and the shift information.
- The staff can be divided into two groups, including instore staff and drivers.
- The orders can be listed based on the type of order (i.e., pick-up or delivery).
- A payment method and payment approval number must be recorded for each order.
- A menu item can consist of multiple ingredients.
- An order can consist of multiple menu items.
- Each order must have a unique order number.
- Each customer must have a unique customer number.
- Each employee must have a unique employee number.
- Each menu item must have a unique item number.
- Each ingredient must have a unique ingredient number.
- The stock level of each ingredient must be recorded.
- The payment method can be either cash, credit card, or debit card.

Part 2: EER Model With Data Dictionary

EER Model



Data Dictionary Entity Types

Entity Name	Description	Aliases	Occurrence
Customer	General term describing all the customers.	NA	Each customer can places one or many orders.

Staff	General term describing all the staff employed by Pizzeria.	NA	Each member of staff working in the Pizza place.
In Store staff	General term for the staff member that works in the store.	NA	In store staff member accepts and verifies orders placed by customers.
Delivery staff	General term for the staff member that works as delivery man for the order.	NA	Delivery driver staff member is responsible for delivering the order. Each driver has driving license number.
Order	term for the orders placed by customer.	Pizza order	Orders are placed by customer consisting one or more menu items. Orders are accepted and verified by in-store staff member. Delivery orders are delivered by driver staff members
Walk in	Term used when the customer self come to the store to place its order	NA	Orders are placed by customer consisting one or more menu items. Orders are accepted and verified by in-store staff member.
Phone order	Order placed by customer through phone	NA	Order taken & verified in store staff
Pickup order	This term used when the come to the store to pick its order. Pick up order is subset of phone order	NA	Order then delivered by in store staff.
Delivery order	This term used when the order is to delivered to customer address.	NA	Delivery orders are delivered by driver staff members.
Menu item	term describing all the items sold by Popular Pizza	NA	Menu item is made up of ingredients. Customer places order for one or more food items.
Ingredient	term describing all the	NA	Customer places order for one or more food items that are made up of different ingredients. Ingredients are

	ingredients used to make a menu (food) item.		supplied by different suppliers.
Suppliers	describing all the entities who supply the ingredients.	NA	When stock level of ingredients reach at order level, an order is placed to supplier to buy the ingredients.
Ingredient Order	describing all the orders placed to buy ingredients.	NA	An ingredient order is placed to buy ingredients from suppliers.
Shift	describing the time period of duty for staff members.	NA	Staff members work in shifts. They are paid for each of the shift they work. Number of deliveries made by delivery driver within a shift is noted.
Staff payment details	term describing the payment made to staff members for single shift	NA	Staff members are paid for the tasks performed for the shift.

Relationships

Entity Name	Multiplicity	Relationship	Multiplicity	Entity Name
Payment	1..*	Due	1..*	Employees
Employees	1..*	has	1..*	ShiftData
Manager	1..1	manages	1..*	IngredientOrder Data
Manager	1..1	manages	1..*	IngredientData
MenuData	1..*	uses	1..*	IngredientData
ShopWorker	1..1	takes	1..*	OrderData
Customer	1..1	give	1..*	OrderData
Employees	1..1	goes	1..*	InfredientOrderData

Payment	1..*	due	1..*	Employees
OrderData	0..*	has	1..*	MenuData
IngredientData	1..*	Supplied by	0..*	Suppliers
Suppliers	0..*	Supplies	1..*	IngredientOrderData
ingredientData	1..*	has	1..*	ingredientOrderData

Attributes:

Entity Name	Attributes	Description	Data Type & Length	Null	Multivalued	Derived	Default
Employees	EmployeeNo	Unique id given to every employee	Int(4)	N	N	N	
	name	Employee name	Varchar(25)	N	N	N	
	staffType	Whether they are instore or	Varchar(25)	N	N	N	

		delivery boys					
	paymentRate	Rate according to shifts	Int(10)	N	N	N	
	postalAddresses	Employees address	Varchar(50)	N	Y	N	
	contactNo	Contact info of employees	Int(10)	N	N	N	
	TFN	Employees Tax File Number	Varchar(15)	N	N	N	
	bankCode	Employees bank code	Int(10)	N	N	N	
	accNo	Employees account number	Int(10)	N	N	N	
	bankName	Employees bank name	Varchar(25)	N	N	N	
	status	Employees status	Varchar(25)	N	N	N	
	description	Employees Discription	Varchar(50)	N	N	N	

Enty Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
Payment	paymentNo	Unique payment number	Int(5)	N	N	N	
	grossPay	Payment without tax	Int(10)	N	N	N	
	taxWithHeld	Payment with tax included	Int(10)	N	N	N	

	payEndDate	Payment final date	Char(25)	N	N	N	
	totalAmountPaid	Total payment	Int(10)	N	N	N	
	accNo	Account number in which payment is deposit	Int(10)	N	N	N	
	accName	Name on which payment is issued	Varchar(25)	N	N	N	

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
ShiftData	shiftNo	Unique shift number is given to each shift	Int(5)	N	N	N	
	startDate	Shift start date	Char(25)	N	N	N	
	startTime	Shift start time	Char(10)	N	N	N	
	endDate	Shift end date	Char(25)	N	N	N	
	endTime	Shift end time	Char(10)	N	N	N	
	shiftType	Type of shift	Char(25)	N	N	N	

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
orderData	orderNo	A unique	Int(10)	N	N	N	2

		order number for every order					
	orderDate	Date of order placed	Char(25)	N	N	N	
	staffInfo	Staff that take the order	Char(25)	N	N	N	
	itemName	Name of the order	Char(25)	N	N	N	
	totalAmountDue	Total amount of the order	Int(10)	N	N	N	
	itemPrice	Price of each item	Int(10)	N	N	N	
	typeOfOrder	Order type	Char(25)	N	N	N	
	Quantity	Quantity of order	Int(10)	N	N	N	
	paymentMethod	Either cash or card	Char(25)	N	N	N	

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
menuData	itemNo	Unique item number to every menu item	Int(10)	N	N	N	
	itemName	Name of the item	Char(25)	N	N	N	
	itemSize	Size of the item	char(10)	N	N	N	
	ingredient	Ingredients used in the item	Char(25)	N	N	N	

	currentPrice	Price of item	Int(10)	N	N	N	
	quantity	Quantity of item	Char(10)	N	N	N	
	description	Description of item	Char(25)	N	N	N	

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
ingredientData	ingredientNo	Unique ingredient number to every ingredient	Int(10)	N	N	N	
	name	Name of the ingredient	Char(25)	N	N	N	
	type	Type of ingredient	Char(25)	N	N	N	
	description	Brief description about the ingredient	Char(25)	N	N	N	

	stocklevel	Level of ingredient left	Char(25)	N	N	N	
	lastStock	last stock of the ingredient	Char(25)	N	N	N	
	suggStock	Suggested stock of ingredient needed	Char(25)	N	N	N	
	recordLevel	Recoed level of the ingredient	Char(25)	N	N	N	
	listOfSuppliers	Suppliers who supplies ingredient	Char(25)	N	N	N	

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
ingredientOrderData	orderNo	Unique order number of ingredients	Int(10)	N	N	N	
	DateOfOrder	Date when the ingredients order placed	Char(25)	N	N	N	
	dateRecievedOrder	Date when order received	Char(25)	N	N	N	
	totalAmount	Total amount of the order	Int(10)	N	N	N	
	orderStatus	Status of the order	Char(25)	N	N	N	
	Description	Brief description of the order	Char(25)	N	N	N	
	quantity	Quantity of the order	Char(25)	N	N	N	2

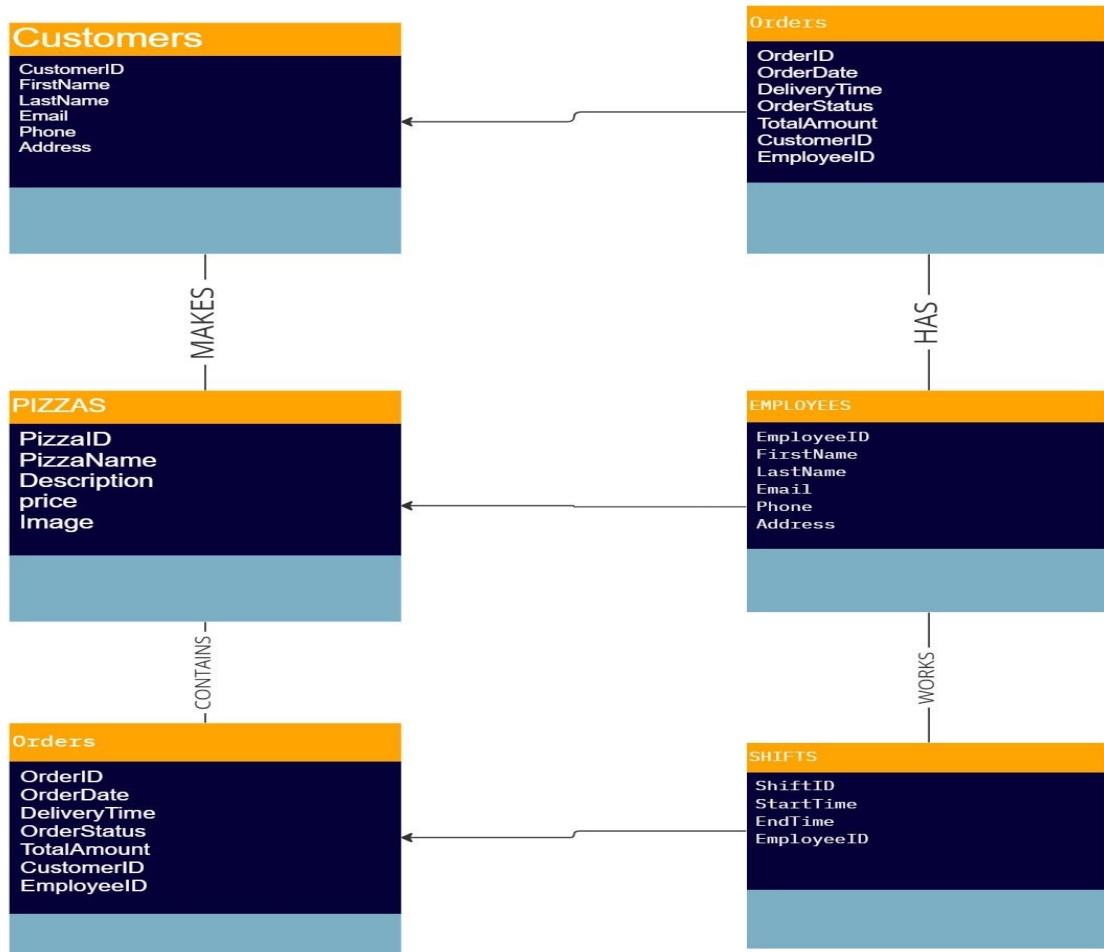
	price	Price of the order	Char(25)	N	N	N	
--	-------	--------------------	----------	---	---	---	--

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
suppliers	supplierNo	Unique supplier number every supplier	Int(10)	N	N	N	
	name	Name of the supplier	Char(25)	N	N	N	
	address	Address of the supplier	Varchar(50)	N	Y	N	
	phoneNo	Phone number of the supplier	Int(10)	N	N	N	
	email	Email of the supplier	Varchar(25)	N	N	N	

	contactPerson	Person tocontact	Char(10)	N	N	N	
--	---------------	------------------	----------	---	---	---	--

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multivalued	Derived	Default
customer	customerNo	Unique customer number to every customer	Int(10)	N	N	N	
	phoneNo	Phone number of customers	Char(10)	N	N	N	
	name	Name of the customers	Char(25)	N	N	N	
	address	Address of the customers	Varchar(50)	N	Y	N	
	hoax	Verification process until customer is validated	Char(25)	N	N	N	

EER ENTITIES



miro

In this example, there are four entities: Customers, Orders, Pizzas, and Employees. Customers can place one or many orders, and each order can contain one or many pizzas. Orders are associated with a customer and an employee who takes the order. Employees can work one or many shifts, and each shift is associated with an employee. The entities are connected through different types of relationships, such as "makes" between Customers and Orders, "contains" between Orders and Pizzas, and "works" between Employees and Shifts.

Queries

- Q.1 For an in-office staff with id number xxx, print his/her 1stname, lname, and hourly payment rate.

```
--Q1
select e.firstName, e.lastName, sw.hourlyRate
from employees e, shopWorker s, shopWorkerShiftData sw
where e.employeeId= s.employeeId and
sw.employeeId= e.employeeId and
s.employeeId= 'E012233446';
```

Result:

Results

Messages

	firstName	lastName	hourlyRate
1	Bob	Marley	20
2	Bob	Marley	20

- Q.2 List all the ingredient details of a menu item named xxx.

```
--Q2
select i.*
from ingredientData i, MenuItem m, menuIngredientData mi
where i.ingredientId= mi.ingredientId and
m.itemNo= mi.itemNo and
m.name = 'beef';
```

Result:

Results		Messages					
	ingredientId	ingredienName	type	stockLevel	lastStock	suggStock	reorderLevel
1	1012233445	Chilli	Spices	low	full	max	min

- Q.3 List all the shift details of a delivery staff with first name xxx and last name tttbetween date yyy and zzz

```
--Q3
select e.firstName, e.lastName, s.*
from shiftData s, driverShiftData d, Employees e
where s.startDate Between '2019-08-10' and '2019-08-28' and
s.employeeId = d.employeeId and
e.employeeId = s.employeeId;
```

Result:

Results		Messages							
	firstName	lastName	shiftId	employeeId	startDate	startTime	endDate	endTime	shiftType
1	Dwayne	Johnson	S012233446	E012233447	2019-08-12	12:00:00.0000000	2019-08-12	18:00:00.0000000	Driver

- Q.4 List all the order details of the orders that are made by a walk-in customer with firstname xxx and last name ttt between date yyy and zzz.

```
--Q4
select o.*
from Orders o, walkInOrder w, Customer c
where o.orderDateTime Between '2019-10-10' and '2019-10-28' and
o.orderID= w.orderID and
o.customerID = c.customerID and
c.firstName = 'John' and
c.lastName = 'MacDonald';
```

Result:

	OrderId	employeeId	OrderDateTime	TotalAmountDue	PaymentMethod	PaymentApprovalNo	OrderStatus	customerId	Quantity
1	C012233445	E012233446	2019-10-12 10:00:00.0000000	45	Cash	0233445566	picked	C012345678	3

- Q.5 List all the order details of the orders that are taken by an in-office staff with firstname xxx and last name ttt between date yyy and zzz.

```
--Q5
select o.*, e.firstName, e.lastName
from orders o, walkInOrder w, employees e
where o.orderId = w.orderId and
o.employeeId = e.employeeId and
o.orderDateTime Between '2019-10-10' and '2019-10-28'
```

Result:

	OrderId	employeeId	OrderDateTime	TotalAmountDue	PaymentMethod	PaymentApprovalNo	OrderStatus	customerId	Quantity	firstName	lastName
1	C012233445	E012233446	2019-10-12 10:00:00.0000000	45	Cash	0233445566	picked	C012345678	3	Bob	Marley

- Q.6 Print the salary paid to a delivery staff named xxx in current month. Note the currentmonth is the current month that is decided by the system.

```
--Q6
select p.totalAmountPaid, dr.noOfDeliveries
from payment p, driverPaymentData d, employees e, driver dr
where p.paymentId = d.paymentId and
p.employeeId = e.employeeId and
dr.employeeId = e.employeeId and
e.firstName = 'Dwayne' and
e.lastName = 'Johnson' and
DATEPART(YEAR, (p.payEndDate)) = YEAR(GETDATE());
```

Result:

	totalAmountPaid	noOfDeliveries
1	20000	5

- Q.7 List the name of the menu item that is mostly ordered in current year.

```
--Q7
SELECT m.ItemNo, m.Name, SUM(QOM.quantity) AS timesOrdered
FROM MenuItem m, QOrderMenuItem QOM, Orders o
WHERE m.ItemNo = QOM.ItemNo AND
o.OrderId = QOM.OrderId AND
DATEPART(YEAR, (o.OrderDateTime)) = YEAR(GETDATE())
GROUP BY m.ItemNo, m.Name
HAVING SUM(QOM.quantity) >= ALL
(SELECT SUM(QOM2.quantity)
FROM MenuItem m2, QOrderMenuItem QOM2, Orders o2
WHERE m2.ItemNo = QOM2.ItemNo AND
o2.OrderId = QOM2.OrderId AND
DATEPART(YEAR, (o2.OrderDateTime)) =
YEAR(GETDATE()))
GROUP BY m2.ItemNo)
```

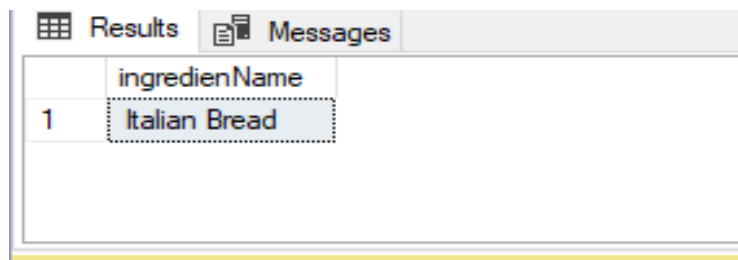
Result

	ItemNo	Name	timesOrdered
1	101245678	beef	4

- Q.8 List the name(s) of the ingredient(s) that was/were supplied by the supplier with supplier ID xxx on date yyy

```
--Q8
select i.ingredienName
from ingredientData i, suppliers s, ingredientSupplierData n where
i.ingredientId = n.ingredientId and
s.supplierId = n.supplierId and
s.name = 'sherjil';
```

Result:



The screenshot shows a database query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with one column labeled 'ingredienName' and one row containing the value 'Italian Bread'. The row is highlighted with a blue background.

	ingredienName
1	Italian Bread

Summary

Requirement analysis and conceptual design of Superdeli Pizza database has been documented in this report. This whole report has been designed to address a problem of making the manual ordering system of Superdeli Pizza online. For making any process computerized, data management is one of the main task. Starting from requirement analysis, a conceptual design of the database has been represented in this report.