|  |
| --- |
| COMP 1140 – Database And Information Management |
| Mega Pizza Project |
| Physical Database Design |

|  |
| --- |
| Sharjeel Sohail  10/31/2019 |

**TABLE OF CONTENTS:**

01 - Reflection Piece ……………………………………………………………………………………………………………………02

Part 02: Requirement Specification ………………………………………………………………………………….…….03-08

02A - Data Requirements …………………………………………………………………………………………....03-06

02B - Transaction Requirements ………………………………………………………………………………..…….07

02C - Business Rules …………………………………………………………………………………………………….07-08

Part 03: EER Diagram and Data dictionary………………………………………………………………………………….09-16

Part 04A - Relational Model Mapping ………………………………………………………………………………….……..17-18

Part 04B - Normalized Relation Schema ……………………………………………………………………………………...19-21

Part 05 – SQL Script ……………………………………………………………………………………………………………………..22-33

**01 - Reflection piece:**

The comments I got from my marker on assignment 02 were following:

The main issue I had with the Normalization process, I tried understanding that but messed it up the whole concept of it. I didn’t provide and function dependencies of any of the relation as well as which form the relation is in. Although, I tried doing the method up to BCNF but there’s no point of doing that unless you find functional dependencies. Some other issues I had was with in my relational mapping. The notifiable error I had made was that I repeated the attributed in the sub-classes as well, for which I lost six marks. Other issue was that my foreign keys were mixed up, and not written where they should be. I also had two minor errors in my EER, related to multiplicity.

After getting the feedback from the marker and the lecturer in the lecture, I tried my best to work on the issues I should. I fixed my EER immediately and understood the multiplicity mistakes I made. I gave a lot of time on Normalization, read slides from the lecture and a bit of help from YouTube made my concept clear about functional dependencies and a proper way to reduce them, as well as notes on BCNF as how to make a relation up to that normal form.

**02A - DATA REQUIREMENTS:**

**2.1 - Order Processing**

Mega Pizza takes orders by phone and walk-in customers and provides both delivery and pickup services. When a customer orders, the customer’s phone number is entered to the system along with the id of the staff taking the order. The customer is then asked for his/her name and address then order is taken. 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. For each phone order, the time the call was answered as well as the time the call was terminated is recorded. For each walk-in order, the time the customer walks in is recorded. For a phone order, after the order has been taken, a verification process occurs whereby the staff dials the number given and confirms the order with the customer. If it is not confirmed, the customer is recorded as a hoax and the order is kept on hold (if and until the customer calls back in which case the verification process takes place before hoax is removed and order goes through). Each order contains date of order, the items ordered, quantity of each item, price of each item, total amount due, payment method, order status, type (phone/walk-in), and description. If the payment is via card, a payment approval number is recorded. For a phone order, if the order is a pickup order, the pickup time is recorded; if the order is a delivery order, the delivery time and address and the driver who delivered the order are recorded.

Two entities could be derived from this data, given below:

1. **Order**
2. **Customer**

**Order**

The following are the data required for the “Order” entity.

* Customer details (Such as: Phone number, address etc)
* Order type (Phone order/Walk-in order)
* Order date and time
* In-and-out time (If phone order; Call pick/end time. If walk-in order; The time customer walks in)
* Total items and total amount of the order
* Payment method (Card or cash)
* Order status (Confirmed or not)
* Short description of the order
* Pickup time OR Delivery time

**Customer**

The following are the data required for the “Customer” entity.

* Customer’s ID
* Customer’s first name
* Customer’s last name
* Customer’s complete address
* Customer’s phone number
* Customer’s status (Phone customer/walk-in customer)
* Order details of the customer (What has been ordered

**2.2 - Menu items, Ingredients and suppliers**

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. 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. A supplier can supply many ingredients. Each ingredient can be supplied by many suppliers. A stocktake is taken each week, where the actual levels of ingredients in store are recorded. This is then compared with suggested levels (based on orders for the week). This report is 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.

Four entities could be derived from this data, given below;

1. **Menu Items**: the menu of available pizza
2. **Ingredients**: the ingredients of each pizza
3. **Suppliers**: details of suppliers supplying the ingredients
4. **Ingredient Orders**: details of ordering ingredients

**Menu Items**

The following are the data required for the “Menu items” entity.

* Item No of the pizza selected
* Name of the pizza
* Size of the pizza
* Current selling price of the pizza
* Description of the pizza

**Ingredients**

The following are the data required for the “Ingredients” entity.

* Unique code of the ingredient
* Ingredient name
* Ingredient type
* Description of the ingredient
* Stock level at current stock take period
* Stock level at last stocktake period
* Suggested current stocktake
* Reorder level of the ingredient

**Suppliers**

The following are the data required for the “Suppliers” entity.

* Supplier number (Unique no)
* Supplier’s name
* Supplier’s address
* Supplier’s phone number
* Supplier’s email
* Supplier’s contact person

**Ingredient Orders**

The following are the data required for the “Ingredient Orders” entity.

* Order number
* Date of the order placed
* Total amount of the ingredient order
* Status of the order
* Description
* Date of the order arrived

**2.3 - Employees**

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. For each employee, there is an employee number, first name, last name, postal address, contact number, tax file number, bank details (bank code, bank name, and account number), payment rate, status, and a description. Drivers also have a driver’s license number. 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 order a driver delivers during a shift is kept on the record. 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.

Three entities could be derived from this data, given below.

1. **Employee**
2. **Employee payment**
3. **Shift**

**Employee Data**

The following are the data required for the “Employee” entity.

* Employee number
* Employee status (In-shop employee/delivery employee)
* Employee’s first name
* Employee’s last name
* Employee’s postal address
* Contact number of the employee
* Tax file number
* Bank details (bank code, bank name, account number)
* Payment rate (in Au Dollars)
* Employee’s description

**Employee payment data**

The following are the data required for the “Employee payment” entity.

* Payment record ID
* Gross payment
* Tax withheld (on the pay)
* Total amount paid to the employee
* Bank details of the employee
* Payment period start date
* Payment period end date

**Shift data**

The following are the data required for the “Shift” entity.

* Start date of the shift
* Start time of the shift
* End date of the shift
* End time of the shift
* Shift type
* Orders a driver employee delivers during a shift, including delivery address and time for each delivery

**02B - Transaction Requirements**

**Data Manipulation**

* Insert, update and delete existing order.
* Insert, update and delete a customer.
* Update the current selling price of the menu item.
* Insert, update and delete the ingredients and quantities used for the item.
* Insert, update and delete the current stock level of an ingredient.
* Insert, update and delete the suppliers for each ingredient.
* Update the payment rate for the employee.
* Insert, update and delete an employee and their details.

**Queries**

* Search a payment based on an employee number, on a particular pay date.
* Search a delivery based on a customer order number.
* List ingredients and quantity used for menu items.
* Report of ingredients levels for current period and suggested stock levels.
* Search a customer data by using his phone number.
* Provide the current selling price and the suppliers for an item using item’s unique code.
* Identify an employee’s status and their payment rate.

**02C Business Rules**

* The amount of each ingredient remaining must be updated every time some is used.
* The results of the weekly stock take must be input into database.
* Each ingredient must have a unique code.
* A stock take must be taken each week.
* The report of the stock take must be sent to the store manager to order ingredients for the following week.
* When an ingredients stock level decreases below its reorder level, an order for the ingredient must be placed.
* A new customer must be marked as an un-verified until the verification process is successfully completed.
* The customer is recorded as a hoax and the order is kept on hold if the order is not confirmed through call.
* Phone orders must be verify by staff by dialing the number given and confirm the order with the customer.
* Shift hours may not be regular but a record is kept for each time an employee works.
* Employees must record each shift they work in the database.
* An employee can only be either in-store worker or delivery driver.
* Delivery drivers must have a driver’s license number in their info.
* Employees cannot delete data from the database.
* An Employee’s status can only be either full time or part time.
* Payments can only be added by accounting staff.
* An orders payment method can only have one of the following values:

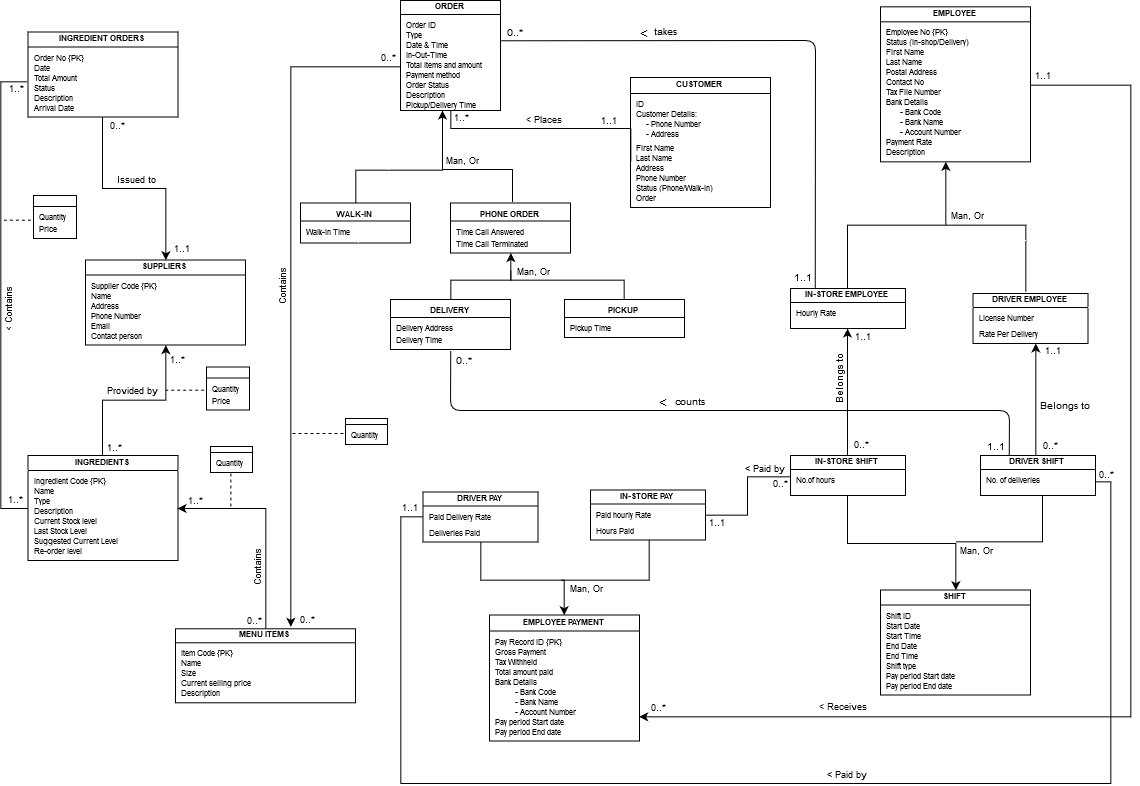
1. Credit card
2. Debit card
3. Cash

* An order’s type can only be either:

1. Pick up
2. Delivery

* If an order is paid for using a card, the approval number must be stored in the order’s payment Approval No.

**03 - EER DIAGRAM AND DATA DICTIONARY:**



***ENTITY TYPES:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Order | Describing orders customer have made | Pizza Order | When an order is made by a customer |
| Walk-in | Orders made by customers in store | Pizza Order | When an order is made in store |
| Phone Order | Orders made by customers over phone | Pizza Order | When an order is made over the phone |
| Customer | Describing who has made the order | Pizza Customer | When someone orders/buys a pizza |
| Menu Items | The Menu of the available pizza | Variety | When a customer has to make a choice while ordering |
| Ingredients | The ingredients of each item/pizza | x | When the menu item is being made |
| Suppliers | Details of suppliers supplying the ingredients | Providers | When ingredients needs more quantity or variety |
| Ingredient Orders | Details of ordering Ingredients | Order System | When placing the ingredients order to suppliers |
| Employee | Describing details of the employees/staff | Staff | When processing orders and getting paid |
| In-store employee | Employee who works in the store | Staff | The one who makes an order |
| Driver employee | Employee who delivers the order | Staff | The one who delivers the order |
| Employee Payment | Describing the details of the employee’s payment/wage | Salary | When employees are getting paid |
| Driver pay | Details of the delivery driver’s pay | Salary | When delivery driver is getting paid |
| In-store pay | Details of the in-store employee’s pay | Salary | When in-store employee is paid |
| Shift | Including the shift information of the employee | Work day | When processing pays, to check how much to pay |
| In-store shift | Shift information of the in-store employee | Work detail | When processing pays |
| Driver shift | Shift information of delivery driver | Work detail | When processing pays |

***RELATIONSHIP TYPES:***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name** | **Multiplicity** | **Relationship** | **Multiplicity** | **Entity Name** |
| Order | 0..\* | Contains | 0..\* | Menu Items |
| (Man, Or) | Generalisation | (Man, Or) | Walk-in |
| (Man, Or) | Generalisation | (Man, Or) | Phone Order |
| (Man, Or) | Generalisation | (Man, Or) | Delivery |
| (Man, Or) | Generalisation | (Man, Or) | Pickup |
| Delivery (Order) | 0..\* | counts | 1..1 | Driver Shift |
| Customer | 1..1 | places | 1..\* | Order |
| Menu Items | 0..\* | Contains | 1..\* | Ingredients |
| Ingredients | 1..\* | Provided by | 1..\* | Suppliers |
| Employee | 1..1 | Receives | 0..\* | Employee Payment |
| (Man, Or) | Generalisation | (Man, Or) | In-store Employee |
| (Man, Or) | Generalisation | (Man, OR) | Driver Employee |
| In Store Employee | 1..1 | takes | 0..\* | Order |
| Employee Payment | 1..1 | Contains | 1..\* | Shift |
| (Man, Or) | Generalisation | (Man, OR) | In-Store Pay |
| (Man, Or) | Generalisation | (Man, OR) | Driver Pay |
| Shift | (Man, Or) | Generalisation | (Man, OR) | In-Store Shift |
| (Man, Or) | Generalisation | (Man, OR) | Driver Shift |
| In-Store Shift | 0..\* | Paid by | 1..1 | In-Store Pay |
| 0..\* | Belongs to | 1..1 | In-Store Employee |
| Driver Shift | 0..\* | Paid by | 1..1 | Driver Pay |
| 0..\* | Belongs to | 1..1 | Driver Employee |

***ATTRIBUTES:***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Order | Order ID | Unique Order ID | Int | N | N | N |  |
| Customer details | Customer’s personal details | Char (100) | N | N | N |  |
| Type | Walk-in or Phone order | Char  (10) | Y | N | N |  |
| Date & Time | Date and time the order made | Date time | Y | N | N |  |
| In-Out-Time | When the customer walked in/out OR Call started/terminated | Date time | Y | N | N |  |
| Total items and amount | How much total menu Item and the total bill | Float | Y | N | N |  |
| Payment method | Card/cash | Char  (100) | Y | N | N |  |
| Order status | Confirmed or not | Char (10) | Y | N | N |  |
| Description | Any extra description of the order | Char (100) | Y | N | N |  |
| Pickup/Delivery Time | When order was picked up/delivered | Int | Y | N | N |  |
| Walk-in | Walk-in time | Time recorded when the customer walks in | Date time | Y | N | N |  |
| Phone Order | Time call answered | Time recorded when a call answered | Date time | Y | N | N |  |
| Time Call Terminated | Time recorded when a call is terminated | Date time | Y | N | N |  |
| Delivery | Delivery Address | Address recorded for the delivery | Char  (100) | Y | N | N |  |
| Delivery Time | Delivery time recorded | Date time | Y | N | N |  |
| Pickup | Pickup time | Time recorded when its picked up | Date time | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Customer | ID | Customer unique Id number | Int | N | N | N |  |
| First Name | Customers first name | Char (10) | Y | N | N |  |
| Last Name | Customers last name | Char (10) | Y | N | N |  |
| Address | Customer postal address | Var- Char (100) | N | N | N |  |
| Phone Number | Contact number of the customer | Int | N | N | N |  |
| Status | Phone Customer or Walk-in | Char (10) | Y | N | N |  |
| Order | What has been ordered | Var- Char (10) | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Menu Items | Item Code | Unique item code | Int | N | N | N |  |
| Name | Name of the item | Char (10) | Y | N | N |  |
| Size | Size (Large, Medium, Small) | Char (10) | Y | N | N |  |
| Current Selling price | Current prices of the item | Float | Y | N | N |  |
| Description | Any further description of the item | Char (100) | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Ingredients | Code | Ingredients unique code | Int | N | N | N |  |
| Name | What is the name of the ingredient | Char (10) | N | N | N |  |
| Type | What type | Char (10) | Y | N | N |  |
| Description | Any short description | Char (100) | Y | N | N |  |
| Current stock level | Stock taken at current level | Float | Y | N | N |  |
| Last stock level | Stock taken the last time | Float | Y | N | N |  |
| Suggested stock level | What could be the suggested stock take | Float | Y | N | N |  |
| Re-order level | How much stock to re order | Float | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Suppliers | Code | Unique code of the supplier | Int | N | N | N |  |
| Name | First and last name | Char (10) | N | N | N |  |
| Address | Full address of the supplier | Char (100) | Y | N | N |  |
| Phone number | Contact number | Int | N | N | N |  |
| Email | Email address to contact | Var-char  (20) | N | N | N |  |
| Contact person | Who should be the person to deal/contact with | Var-char  (20) | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Ingredient- Orders | Order No | Unique order number | Int | N | N | N |  |
| Date | Date of the order placed | Date time | Y | N | N |  |
| Total amount | Total amount of the order | Float | Y | N | N |  |
| Status | Confirmed or not | Var-char (10) | Y | N | N |  |
| Description | Any short description of the order | Char (100) | Y | N | N |  |
| Arrival date | When the order arrived | Date time | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Employee | Employee No | Unique employee number | Int | N | N | N |  |
| Status | In-store employee or delivery driver | Char (10) | Y | N | N |  |
| First name | First name of the employee | Char (10) | Y | N | N |  |
| Last name | Last name of the employee | Char (10) | Y | N | N |  |
| Postal address | Postal address of the employee | Char (100) | N | N | N |  |
| Contact no | Verified contact number | Int | N | N | N |  |
| Tax file number | TFN number | Int | N | N | N |  |
| Bank details | Bank details; Code, Name, Account number | Var-Char (10) | N | N | N |  |
| Payment rate | What’s the pay rate | Float | Y | N | N |  |
| Description | Any short description | Char (100) | Y | N | N |  |
| In-store Employee | Hourly rate | Hourly rate is fixed of the in-store employee | Float | Y | N | N |  |
| Driver Employee | License Number | License number is recorded of the driver | VARCHAR | Y | N | N |  |
| Rate per delivery | Rate each delivery delivered is noted | Float | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Employee Payment | Pay record ID | Pay slip ID | Int | N | n | N |  |
| Gross payment | Total gross amount paid | Float | Y | N | N |  |
| Tax withheld | Total tax to be deduct | Float | Y | N | N |  |
| Total amount paid | Total net amount paid | Float | Y | N | N |  |
| Bank details | Bank details; Code, Name, Account number | Char (100) | N | N | N |  |
| Pay period start date | Pay cycle starts | Date time | Y | N | N |  |
| Pay period end date | Pay cycle ends | Date time | Y | N | N |  |
| Driver Pay | Paid delivery rate | Payment paid each delivery | Float | Y | N | N |  |
| Deliveries paid | Total deliveries paid | Int | Y | N | N |  |
| In-Store Pay | Paid hourly rate | Payment per hour is fixed | Float | Y | N | N |  |
| Hours paid | Total hours paid | Int | Y | N | N |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type & Length** | **Nulls** | **Multi-Values** | **Derived** | **Default** |
| Shift | Start date | Starting date of the shift | Date time | Y | N | N |  |
| Start time | What time shift started | Date time | Y | N | N |  |
| End date | Ending date of the shift | Date time | Y | N | N |  |
| End time | What time shift ended | Date time | Y | N | N |  |
| Shift type | Part time / casual | Char (10) | Y | N | N |  |
| Pay period start date | Pay cycle start date | Date time | Y | N | N |  |
| Pay period end date | Pay cycle end date | Date time | Y | N | N |  |
| In-Store shift | No. Of hours | Pay based on how many hours worked | Float | Y | N | N |  |
| Driver shift | No. Of deliveries | Pay based on how many deliveries done | Float | Y | N | N |  |

**04A Relational Model Mapping:**

|  |
| --- |
| **1) Ingredient Orders** (OrderNo, Date, TotalAmount, Status, Description, ArrivalDate)  **Primary Key:** OrderNo  **Contains** (OrderNo, IngredientCode, Quantity, Price)  **Primary Key**: OrderNo, IngredientCode, Quantity)  **Foreign Key:** OrderNo **references** Ingredient Orders (orderNo)  **Foreign Key:** IngredientCode **references** Ingredients (ingredientCode)  **Foreign Key:** SupplierCode **references** Suppliers (SupplierCode) |
| **2) Suppliers** (SupplierCode, Name, Address, PhoneNumber, Email, ContactPerson)  **Primary Key:** Code  **Alternate Key**: PhoneNumber, Email |
| **3) Ingredients** (IngredientCode, Name, Type, Description, CurrentStockLevel, LastStockLevel, SuggestedCurrentLevel, Re-orderLevel)  **Primary Key:** Code  **Provided by** (IngredientCode, SuppliersCode, Quantity, Price)  **Primary Key:** IngredientCode, SuppliersCode, Quantity  **Foreign Key:** IngredientCode **references** Ingredients (IngredientCode)  **Foreign Key:** SupplierCode **references** Suppliers (SupplierCode) |
| **4) Order** (OrderID, PhoneNumber, Address, OrderType, Date&Time, In-Out-time, TotalItems&Amount, PaymentMethod, OrderStatus, Description, Pickup/DeliveryTime)  **Primary Key:** OrderID  **Alternate Key**: PhoneNumber, Address  **Order Contains** (OrderID, ItemCode, Quantity)  **Primary Key**: OrderID, ItemCode, Quantity  **Foreign Key**: OrderID **references** Order (OrderID)  **Foreign Key**: ItemCode **references** MenuItems (ItemCode)  **Foreign Key:** CustomerID **references** Customers (CustomerID)  **Foreign Key:** EmployeeNo **references** InStoreEmployee (EmployeeNo) |
| **5) Walk-In** (OrderID, Walk-inTime)  **Primary Key:** OrderID  **Foreign Key:** OrderID **references** Order (OrderID) |
| **6) Phone Order** (OrderID, PhoneNumber, Address, OrderType, Date&Time, In-Out-time, TotalItems&Amount, PaymentMethod, OrderStatus, Description, Pickup/DeliveryTime, TimeCallAnswered, TimeCallTerminated)  **Primary Key:** OrderID  **Foreign Key:** OrderID **references** Order (OrderID) |
| **7) Delivery** (OrderID, TimeCallAnswered, TimeCallTerminated, DeliveryAddress, DeliveryTime)  **Primary Key:** OrderID  **Foreign Key:** OrderID **references** PhoneOrder (OrderID) |
| **8) Pickup** (OrderID, PickupTime)  **Primary Key:** OrderID  **Foreign Key:** OrderID **references** PhoneOrder (OrderID) |
| **9) Employee Payment** (PayRecordID, GrossPayment, TaxWithheld, TotalAmountPaid, BankCode, BankName, AccountNumber, PayPeriodStartDate, PayPeriodEndDate)  **Primary Key:** PayRecordID  **Alternate Key**: AccountNumber |
| **10) Driver-Pay** (PayRecordID, PaidDeliveryRate, DeliveriesPaid)  **Primary Key:** PayRecordID  **Foreign Key**: PayRecordID **references** Employee Payment (PayRecordID) |
| **11) In-Store Pay** (PayRecordID, PaidHourlyRate, HoursPaid)  **Primary Key:** PayRecordID  **Foreign Key**: PayRecordID **references** Employee Payment (PayRecordID) |
| **12) Employee** (EmployeeNo, Status, FirstName, LastName, PostalAddress, ContactNo, TaxFileNumber, BankCode, BankName, AccountNumber, PaymentRate, Description)  **Primary Key:** EmployeeNo  **Alternate Key:** ContactNo, TaxFileNumber, AccountNumber |
| **13) In-Store Employee** (EmployeeNo, HourlyRate)  **Primary Key:** EmployeeNo  **Foreign Key**: EmployeeNo **references** Employee (EmployeeNo) |
| **14) Driver Employee** (EmployeeNo, LicenseNumber, RatePerDelivery)  **Primary Key:** EmployeeNo  **Foreign Key**: EmployeeNo **references** Employee (EmployeeNo) |
| **15) Shift** (ShiftID, StartDate, StartTime, EndDate, EndTime, ShiftType)  **Primary Key:** ShiftID |
| **16) In-Store Shift** (ShiftID, No.OfHours, PayRecordID, EmployeeNo)  **Primary Key:** ShiftID  **Foreign Key**: ShiftID **references** Shift (ShiftID)  **Foreign Key:** PayRecordID **references** In-Store Pay (PayRecordID)  **Foreign Key:** EmployeeNo **references** In-Store Employee (EmployeeNo) |
| **17) Driver Shift** (ShiftID, No.OfDeliveries, PayRecordID, EmployeeNo)  **Primary Key:** ShiftID  **Foreign Key**: ShiftID **references** Shift (ShiftID)  **Foreign Key:** PayRecordID **references** Driver Pay (PayRecordID)  **Foreign Key:** EmployeeNo **references** Driver Employee (EmployeeNo) |
| **18) Menu Items** (ItemCode, Name, Size, CurrentSellingPrice, Description)  **Primary Key:** ItemCode  **holds** (ItemCode, IngredientCode, Quantity)  **Primary Key:** ItemCode, IngredientCode, Quantity  **Foreign Key:** ItemCode **references** Menu Items (ItemCode)  **Foreign Key:** IngredientCode **references** ingredients (IngredientCode) |
| **19) Customer** (CustomerID, FirstName, LastName, Address, PhoneNumber, Status, Order)  **Primary Key:** CustomerID  **Alternate Key:** PhoneNumber, Address |

**04B Normalized Relation Schema:**

**1) Ingredient** **order** (OrderNo, orderDate, totalAmount, orderStatus, orderDescription, supplierCode, arrivalDate)

**Primary** **Key**: OrderNo

**FD1**: OrderNo > orderDate, totalAmount, orderStatus, orderDescription, supplierCode, arrivalDate

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**2) Supplier** (supplierCode, supplierName, supplierAddress, phoneNumber, emailAddress, contactPerson)

**Primary** **Key**: SupplierCode

**FD1**: supplierCode > supplierName, supplierAddress, phoneNumber, emailAddress, contactPerson

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**3) Ingredients** (IngCode, IngName, IngType, IngDescription, CurrentStockLevel, LastStockLevel, SuggestedCurrentLevel, ReOrderLevel)

**Primary** **Key**: IngCode

**FD1**: IngCode > IngName, IngType, IngDescription, CurrentStockLevel, SuggestedCurrentLevel

**FD2**: CurrentStockLevel > LastStockLevel, ReOrderLevel

Since two attributes IN FD2 depend on one non-prime attribute, here exists transitive dependency, Hence the relation is in 2nd Normal form and not in BCNF.

**The Normalization process:**

Here we now form a new relation out of transitive dependency by keeping the PK relation:

**i) IngredientsOnly** (IngCode, IngName, IngType, IngDescription, CurrentStockLevel, SuggestedCurrentLevel)

**Primary** **Key**: IngCode

**ii) IngredientsLevel** (CurrentStockLevel, LastStockLevel, ReOrderLevel)

**Primary** **Key**: CurrentStockLevel

**Foreign** **Key**: CurrentStockLevel **references** IngredientOnly (CurrentStockLevel) **ON UPDATE CASCADE, ON DELETE CASCADE**

**4) Orders** (OrderId, phoneNumber, orderAddress, orderType, dateAndTime, InAndOutTime, totalItemsAndAmount, paymentMethod, orderStatus, orderDescription, pickupOrDeliveryTime)

**Primary** **Key**: OrderId

**FD1**: OrderId > phoneNumber, orderAddress, orderType, dateAndTime, InAndOutTime, totalItemsAndAmount, paymentMethod, orderStatus, orderDescription, pickupOrDeliveryTime

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**5)** **EmployeePayment** (PayRecordId, GrossPayment, taxWithHeld, TotalAmountPaid, BankCode, bankName, accountNumber, payPeriodStartDate, payPeriodEndDate)

**Primary** **Key**: PayRecordId

**FD1**: PayRecordId > GrossPayment, taxWithHeld, TotalAmountPaid, accountNumber, payPeriodStartDate, payPeriodEndDate

**FD2**: accountNumber > backCode, bankName

Since two attributes in FD2 depend on one non-prime attribute, here exists transitive dependency, Hence the relation is in 2nd Normal form and not in BCNF.

**The** **Normalization** **Process**:

Here we now form a new relation out of transitive dependency by keeping the PK relation:

**i) EmployeePaydetails** (PayRecordId, GrossPayment, taxWithHeld, TotalAmountPaid, accountNumber, payPeriodStartDate, payPeriodEndDate)

**Primary** **Key**: PayRecordId

**ii) Employeebankdetails** (accountNumber, BankCode, bankName)

**Primary** **Key**: accountNumber

**Foreign** **Key**: accountNumber **references** EmployeePaydetails (accontNumber) **ON UPDATE CASCADE, ON DELETE CASCADE**

**6) Employee** (employeeNo, employeeStatus, firstName, lastName, postalAddress, contactNo, taxFileNumber, bankCode, bankName, accountNumber, paymentRate, employeeDescription)

**Primary** **Key**: employeeNo

**FD1**: employeeNo > employeeStatus, firstName, lastName, postalAddress, contactNo, accountNumber taxFileNumber, paymentRate, employeeDescription

**FD2**: accountNumber > backCode, bankName

Since two attributes in FD2 depend on one non-prime attribute, here exists transitive dependency, Hence the relation is in 2nd Normal form and not in BCNF.

**The Normalization Process:**

Here we now form a new relation out of transitive dependency by keeping the PK relation:

**i)** **employeeInfo** (employeeNo, employeeStatus, firstName, lastName, postalAddress, contactNo, accountNumber, taxFileNumber, paymentRate, employeeDescription)

**Primary** **Key**: employeeNo

**ii)** **employeeInfoBank** (accountNumber, backCode, bankName)

**Primary** **Key**: accountNumber

**Foreign** **Key**: accountNumber **references** employeeInfo (accountNumber) **ON UPDATE CASCADE, ON DELETE CASCADE**

**7) Shifts** (shiftId, startDate, StartTime, endDate, endTime, shiftType)

**Primary** **Key**: shiftId

**FD1**: shifId > startDate, StartTime, endDate, endTime, shiftType

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**8) MenuItems** (itemCode, itemName, size, currentSellingPrice, itemDescription)

**Primary** **Key**: itemCode

**FD1**: itemCode > itemName, size, currentSellingPrice, itemDescription

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**9) Customers** (customerId, fName, lName, custAddress, phoneNumber, custStatus, custOrder)

**Primary** **Key**: customerId

**FD1**: customerId > fName, lName, custAddress, phoneNumber, custStatus, custOrder

As there is no transitive dependency or partial dependency found. Hence, the relation is already in BCNF form.

**PART 05 – PHYSICAL DATABASE DESIGN**

-- Assignment 3 Physical database design

-- Created by Sharjeel Sohail c3316130

-- Due date: November 31st 2019

-- % DROP ALL THE TABLES %

DROP TABLE IngredientOrder;

DROP TABLE Suppliers;

DROP TABLE IngredientsOnly;

DROP TABLE IngredientsLevel;

DROP TABLE Orders;

DROP TABLE WalkInOrder;

DROP TABLE PhoneOrder;

DROP TABLE Delivery;

DROP TABLE Pickup;

DROP TABLE EmployeePayDetails;

DROP TABLE EmployeeBankDetails;

DROP TABLE DriverPay;

DROP TABLE InStorePay;

DROP TABLE EmployeeInfo;

DROP TABLE EmployeeInfoBank;

DROP TABLE InStoreEmployee;

DROP TABLE DriverEmployee;

DROP TABLE Shifts;

DROP TABLE InStoreShift;

DROP TABLE DriverShift;

DROP TABLE MenuItems;

DROP TABLE Customers;

-- % CREATING ALL THE TABLES %

--SELECT \* FROM IngredientOrder;

CREATE TABLE IngredientOrder(

orderNo CHAR(10) PRIMARY KEY,

orderDate DATETIME2,

totalAmount FLOAT NOT NULL,

orderStatus VARCHAR(20) DEFAULT 'Pending',

orderDescription CHAR(50),

supplierCode CHAR(10),

arrivalDate DATETIME2,

-- Foreign key added later

);

--SELECT \* FROM Suppliers;

CREATE TABLE Suppliers(

supplierCode CHAR(10) PRIMARY KEY,

supplierName CHAR(20) NOT NULL,

supplierAddress VARCHAR(30),

phoneNumber VARCHAR(10),

emailAddress VARCHAR(30) UNIQUE,

contactPerson CHAR(20) NOT NULL

);

--SELECT \* FROM IngredientsOnly;

CREATE TABLE IngredientsOnly(

IngCode CHAR(10),

IngName CHAR(20) NOT NULL,

IngType VARCHAR(10),

IngDescription CHAR(30),

CurrentStockLevel FLOAT,

SuggestedCurrentLevel FLOAT NOT NULL,

PRIMARY KEY (CurrentStockLevel)

);

--SELECT \* FROM IngredientsLevel;

CREATE TABLE IngredientsLevel(

CurrentStockLevel FLOAT PRIMARY KEY,

LastStockLevel FLOAT,

ReOrderLevel FLOAT

FOREIGN KEY(CurrentStockLevel) REFERENCES IngredientsOnly(CurrentStockLevel) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM Orders;

CREATE TABLE Orders(

OrderId CHAR(10) PRIMARY KEY,

customerId CHAR (10),

PhoneNumber VARCHAR(10) UNIQUE,

OrderAddress VARCHAR(30) NOT NULL,

OrderType VARCHAR(10),

dateAndTime DATETIME2,

InAndOutTime DATETIME2,

TotalItemsAndAmount VARCHAR(30) NOT NULL,

PaymentMethod VARCHAR(20) DEFAULT 'Bank card',

OrderStatus VARCHAR(20),

OrderDescription CHAR(40),

PickupOrDeliveryTime DATETIME2,

);

--SELECT \* FROM WalkInOrder;

CREATE TABLE WalkInOrder(

OrderId CHAR(10) PRIMARY KEY,

WalkInTime DATETIME2,

FOREIGN KEY (OrderId) REFERENCES Orders (OrderId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM PhoneOrder;

CREATE TABLE PhoneOrder(

OrderId CHAR(10) PRIMARY KEY,

TimeCallAnswered DATETIME2,

TimeCallTerminated DATETIME2,

FOREIGN KEY (OrderId) REFERENCES Orders (OrderId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM Delivery;

CREATE TABLE Delivery(

OrderId CHAR(10) PRIMARY KEY,

DeliveryAddress VARCHAR(40) NOT NULL,

DeliveryTime DATETIME2,

FOREIGN KEY (OrderId) REFERENCES PhoneOrder (OrderId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM Pickup;

CREATE TABLE Pickup(

OrderId CHAR(10) PRIMARY KEY,

PickupTime DATETIME2,

FOREIGN KEY (OrderId) REFERENCES PhoneOrder (OrderId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM EmployeePayDetails;

CREATE TABLE EmployeePayDetails(

PayRecordId VARCHAR(20) PRIMARY KEY,

GrossPayment FLOAT,

TaxWithheld FLOAT,

TotalAmountPaid FLOAT NOT NULL,

AccountNumber VARCHAR(20) UNIQUE,

PayPeriodStartDate DATETIME2,

PayPeriodEndDate DATETIME2,

);

--SELECT \* FROM EmployeeBankDetails;

CREATE TABLE EmployeeBankDetails(

AccountNumber VARCHAR(20) PRIMARY KEY,

BankCode VARCHAR(10) NOT NULL,

BankName CHAR(20),

FOREIGN KEY(AccountNumber) REFERENCES EmployeePayDetails(PayRecordId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM DriverPay;

CREATE TABLE DriverPay(

PayRecordId VARCHAR(20) PRIMARY KEY,

PaidDeliveryRate FLOAT DEFAULT 'Undecided',

DeliveriesPaid INT,

FOREIGN KEY (PayRecordId) REFERENCES EmployeePayDetails (PayRecordId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM InStorePay;

CREATE TABLE InStorePay(

PayRecordId VARCHAR(20) PRIMARY KEY,

PaidHourlyRate FLOAT,

HoursPaid FLOAT

FOREIGN KEY (PayRecordId) REFERENCES EmployeePayDetails (PayRecordId) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM EmployeeInfo;

CREATE TABLE EmployeeInfo(

EmployeeNo VARCHAR(20) PRIMARY KEY,

EmployeeStatus CHAR(10),

FirstName CHAR(15) NOT NULL,

LastName CHAR(15) NOT NULL,

PostalAddress VARCHAR(30),

ContactNo VARCHAR(10),

TaxFileNumber VARCHAR(15),

AccountNumber VARCHAR(20) UNIQUE,

PaymentRate FLOAT DEFAULT 'Undecided',

EmployeeDescription CHAR(30)

);

--SELECT \* FROM EmployeeInfoBank;

CREATE TABLE EmployeeInfoBank(

AccountNumber VARCHAR(20) PRIMARY KEY,

BankCode VARCHAR(10) NOT NULL,

BankName CHAR(20),

FOREIGN KEY(AccountNumber) REFERENCES EmployeeInfo(EmployeeNo) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM InStoreEmployee;

CREATE TABLE InStoreEmployee(

EmployeeNo VARCHAR(20) PRIMARY KEY,

HourlyRate FLOAT,

FOREIGN KEY(EmployeeNo) REFERENCES EmployeeInfo(EmployeeNo) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM DriverEmployee;

CREATE TABLE DriverEmployee(

EmployeeNo VARCHAR(20) PRIMARY KEY,

LicenseNo VARCHAR(10) NOT NULL,

RatePerDelivery FLOAT,

FOREIGN KEY(EmployeeNo) REFERENCES EmployeeInfo(EmployeeNo) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM Shifts;

CREATE TABLE Shifts(

ShiftID CHAR(10) PRIMARY KEY,

StartDate DATETIME2,

StartTime DATETIME2,

EndDate DATETIME2,

EndTime DATETIME2,

ShiftType VARCHAR(10)

);

--SELECT \* FROM InStoreShift;

CREATE TABLE InStoreShift(

ShiftId CHAR(10) PRIMARY KEY,

PayRecordId VARCHAR(20),

EmployeeNo VARCHAR(20),

NoOfHours FLOAT,

FOREIGN KEY (ShiftId) REFERENCES Shifts (ShiftId) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (PayRecordId) REFERENCES InStorePay(PayRecordId) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (EmployeeNo) REFERENCES InStoreEmployee (EmployeeNo) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM DriverShift;

CREATE TABLE DriverShift(

ShiftId CHAR(10) PRIMARY KEY,

PayRecordId VARCHAR(20),

EmployeeNo VARCHAR(20),

NoOfDeliveries INT,

FOREIGN KEY (ShiftId) REFERENCES Shifts (ShiftId) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (PayRecordId) REFERENCES DriverPay (PayRecordId) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (EmployeeNo) REFERENCES DriverEmployee (EmployeeNo) ON UPDATE CASCADE ON DELETE CASCADE

);

--SELECT \* FROM MenuItems;

CREATE TABLE MenuItems(

ItemCode CHAR(10) PRIMARY KEY,

ItemName CHAR(20) NOT NULL,

Size VARCHAR(10),

CurrentSellingPrice FLOAT,

ItemDescription CHAR(50),

);

--SELECT \* FROM Customers;

CREATE TABLE Customers(

CustomerId CHAR(10) PRIMARY KEY,

fName CHAR(15) NOT NULL,

lName CHAR(15) NOT NULL,

CustAdress VARCHAR(30),

phoneNumber INT UNIQUE,

CustStatus CHAR(15),

CustOrder VARCHAR(50)

);

-- % ADDING FOREIGN KEYS INTO THE TABLE %

ALTER TABLE IngredientOrder

ADD FOREIGN KEY(supplierCode) REFERENCES Suppliers (supplierCode) ON UPDATE CASCADE ON DELETE CASCADE;

-- % INSERING SAMPLE DATA INTO THE TABLES %

INSERT INTO Suppliers

VALUES ('SUPP22', 'Travis', '109 University Drive NSW 2299', '0232447845', 'travisscott@gmail.com', 'Travis Scott');

INSERT INTO Suppliers

VALUES ('SUPP23', 'Kylie', '25 Nothcut Lane NSW 2675', '0444786908', 'kyliejenner@gmail.com', 'Mr Scott');

INSERT INTO Suppliers

VALUES ('SUPP24', 'Stormi', 'Evatt House UON NSW 2390', '0212767554', 'stormiscott1@gmail.com', 'Kylie Jenner');

INSERT INTO IngredientOrder

VALUES ('IO22334400', '2019-10-27 10:0:0', 1021.50, 'Confirmed', 'Please add plastic bags too', 'SUPP22', '2019-10-29 11:0:0');

INSERT INTO IngredientOrder

VALUES ('IO22334411', '2019-10-28 10:0:0', 499.98, 'Pending', 'Order should be here in two days', 'SUPP23', '2019-10-30 11:0:0');

INSERT INTO IngredientOrder

VALUES ('IO22334422', '2019-10-29 10:0:0', 202.43, 'On the way', 'Add extra spices if easy', 'SUPP24', '2019-10-31 11:0:0');

INSERT INTO IngredientsOnly

VALUES ('ING00123', 'Bega Cheese', 'Dairy', 'Fresh Cheese Slices', 89.4,175.00);

INSERT INTO IngredientsOnly

VALUES ('ING00124', 'Capsicum', 'Vegetable', 'Red Capsicum with leaf', 88.5,50.00);

INSERT INTO IngredientsOnly

VALUES ('ING00125', 'Red beans', 'Beans', 'Red protein beans', 10.0, 70.00);

INSERT INTO IngredientsLevel

VALUES (89.4, 44.40, 120);

INSERT INTO IngredientsLevel

VALUES (88.5, 39.25, 105.50);

INSERT INTO IngredientsLevel

VALUES (10.0, 30.5, 50.0);

INSERT INTO Orders

VALUES ('ORDER01', 'CUS24', '0332567890', '12 Street Lakemba 2294', 'Takeaway', '2019-10-11 09:0:0', '2019-10-11 09:15:0', '2 Items for $23.75', 'Cash', 'Confirmed', 'Add extra cheese', '2019-10-11 09:15:00');

INSERT INTO Orders

VALUES ('ORDER02', 'CUS56', '0223556774', '55A Turntin Rd Lakemba 2294', 'Dine In', '2019-10-12 10:0:0', '2019-10-12 10:15:0', '4 Items for $43.55', 'Cash', 'Not Confirmed', 'Serve in plastic plates', '2019-10-12 10:15:00');

INSERT INTO Orders

VALUES ('ORDER03', 'CUS88', '0445776772', 'Huxley Squares Auburn 2244', 'Takeaway', '2019-10-11 14:0:0', '2019-10-11 14:25:0', '1 Item for $9.75', 'Card', 'Pending', 'Add extra chilli flakes', '2019-10-11 14:25:00');

INSERT INTO WalkInOrder

VALUES ('ORDER01', '2019-10-11 09:0:0');

INSERT INTO WalkInOrder

VALUES ('ORDER02', '2019-10-12 10:0:0');

INSERT INTO WalkInOrder

VALUES ('ORDER03', '2019-10-13 11:20:0');

INSERT INTO PhoneOrder

VALUES ('ORDER01', '2019-10-11 20:0:0', '2019-10-11 20:09:0');

INSERT INTO PhoneOrder

VALUES ('ORDER02', '2019-10-12 09:0:0', '2019-10-12 09:05:0');

INSERT INTO PhoneOrder

VALUES ('ORDER03', '2019-10-11 15:0:0', '2019-10-11 15:12:0');

INSERT INTO Delivery

VALUES ('ORDER01', '02A Naghton Ave Brookshill 2288', '2019-10-19 14:05:0');

INSERT INTO Delivery

VALUES ('ORDER02', '88 Flesh Lane Sydney 2278', '2019-10-20 19:10:0');

INSERT INTO Delivery

VALUES ('ORDER03', '02A Naghton Ave Brookshill 2288', '2019-10-20 16:55:0');

INSERT INTO Pickup

VALUES ('ORDER01', '2019-10-22 18:23:0');

INSERT INTO Pickup

VALUES ('ORDER02', '2019-10-03 09:20:0');

INSERT INTO Pickup

VALUES ('ORDER03', '2019-10-18 22:02:0');

INSERT INTO EmployeePayDetails

VALUES ('EMP87039', 1201.89, 179.85, 1032.20, '0554893', '2019-10-07 12:0:0', '2019-10-14 12:0:0');

INSERT INTO EmployeePayDetails

VALUES ('EMP87874', 1140.89, 150.87, 990.54, '2239842', '2019-10-07 12:0:0', '2019-10-14 12:0:0');

INSERT INTO EmployeePayDetails

VALUES ('EMP89002', 998.20, 95.22, 902.12, '0887334', '2019-10-14 12:0:0', '2019-10-21 12:0:0');

INSERT INTO EmployeeBankDetails

VALUES ('0554893', 'NEWY24', 'Commonwealth Bank');

INSERT INTO EmployeeBankDetails

VALUES ('2239842', 'ANZ01', 'ANZ Bank');

INSERT INTO EmployeeBankDetails

VALUES ('0887334', 'NEWY24', 'Commonwealth Bank');

INSERT INTO DriverPay

VALUES ('EMP87039', 9.75, 22);

INSERT INTO DriverPay

VALUES ('EMP87874', 9.33, 18);

INSERT INTO DriverPay

VALUES ('EMP89002', 9.75, 31);

INSERT INTO InStorePay

VALUES ('EMP87039', 27.03, 39.75);

INSERT INTO InStorePay

VALUES ('EMP87874', 27.03, 27.90);

INSERT INTO InStorePay

VALUES ('EMP89002', '20.21', 42.50);

INSERT INTO EmployeeInfo

VALUES ('EMP44871', 'Casual', 'Sharjeel', 'Sohail', '109 University Drive', '0451645687', '051-4463-90', '336887', 27.03, 'Long tall brown guy');

INSERT INTO EmployeeInfo

VALUES ('EMP55639', 'Permanent', 'Zain', 'Abedin', '41 Mayfield Newcastle', '0322876452', '032-3343-77', '992332', 17.50, 'Old loyal employee');

INSERT INTO EmployeeInfo

VALUES ('EMP12390', 'Casual', 'Shingi', 'Chikoto', '40A Englund St Birmingham', '0453886120', '776-2678-22', '886352', 35.95, 'Tall black guy');

INSERT INTO EmployeeInfoBank

VALUES ('EMP44871', 'NEWY24', 'Commonwealth Bank');

INSERT INTO EmployeeInfoBank

VALUES ('EMP55639', 'NEWY24', 'Commonwealth Bank');

INSERT INTO EmployeeInfoBank

VALUES ('EMP12390', 'ANZ01', 'ANZ Bank');

INSERT INTO InStoreEmployee

VALUES ('EMP44871', 27.03);

INSERT INTO InStoreEmployee

VALUES ('EMP55639', 17.50);

INSERT INTO InStoreEmployee

VALUES ('EMP12390', 35.95);

INSERT INTO DriverEmployee

VALUES ('EMP66785', 'N376YY98', 6.75);

INSERT INTO DriverEmployee

VALUES ('EMP111982', 'X99IO664', 6.75);

INSERT INTO DriverEmployee

VALUES ('EMP77690', 'YYU5561M', 6.75);

INSERT INTO Shifts

VALUES ('MORNING12', '2019-10-11', '12:0:0', '2019-10-11', '20:0:0', 'Dayshift');

INSERT INTO Shifts

VALUES ('MORNING09', '2019-10-20', '09:0:0', '2019-10-20', '15:0:0', 'Dayshift');

INSERT INTO Shifts

VALUES ('NIGHT11', '2019-10-27', '11:0:0', '2019-10-28', '05:0:0', 'Overnight');

INSERT INTO InStoreShift

VALUES ('MORNING12', 'EMP87039', 'EMP44871', 8.25);

INSERT INTO InStoreShift

VALUES ('MORNING09', 'EMP87874', 'EMP55639', 5.50);

INSERT INTO InStoreShift

VALUES ('NIGHT11', 'EMP89002', 'EMP12390', 7.00);

INSERT INTO DriverShift

VALUES ('MORNING12', 'EMP87039', 'EMP66785', 14);

INSERT INTO DriverShift

VALUES ('MORNING09', 'EMP87874', 'EMP111982', 09);

INSERT INTO DriverShift

VALUES ('NIGHT11', 'EMP89002', 'EMP77690', 21);

INSERT INTO MenuItems

VALUES ('ITEM001', 'Lasangna', 'large', 18.99, 'Beef cheese lasagna');

INSERT INTO MenuItems

VALUES ('ITEM002', 'Chicken Supreme', 'large', 21.95, 'Chicken Veg pizza');

INSERT INTO MenuItems

VALUES ('ITEM003', 'Meat lover', 'Small', 12.90, 'Beef pizza with buffalo sauce');

INSERT INTO Customers

VALUES ('CUS24', 'Mitchelle', 'Elf', '99 Ulav Street Lambton', 0887556231, 'PhCustomer', 'Chicken Supreme & Meat Lover');

INSERT INTO Customers

VALUES ('CUS56', 'Micheal', 'Shelby', '22 Garrison St Laks', 0456778923, 'DCustomer', 'Chicken lasagna & Garlic bread');

INSERT INTO Customers

VALUES ('CUS88', 'Robin', 'Billy', 'Red Rose Apartments Mumbai', 0562991663, 'PhCustomer', 'Chicken Supreme');

-- % CREATING QUERIES FROM HERE %

-- Q.3 List all the order details of the orders that are made by a walk-in cusotmer with

-- fist name Robin and last name billy between date 2019-10-08 and 2019-10-15.

SELECT WalkInOrder.\*, Orders.\*

FROM Orders, Customers, WalkInOrder

WHERE Orders.OrderId = WalkInOrder.OrderId AND

Orders.customerId = Customers.CustomerId AND

Customers.fName = 'Robin' AND

Customers.lName = 'Billy' AND

Orders.dateAndTime BETWEEN '2019-10-08' AND '2019-10-15';

-- Q.2 List all the shift details of a delivery staff with first name Sharjeel

--and last name Sohail between 2019-10-10 and 2019-10-28

SELECT EmployeeInfo.FirstName, EmployeeInfo.LastName, Shifts.\*

FROM EmployeeInfo, Shifts

WHERE EmployeeInfo.FirstName = 'Sharjeel' AND

EmployeeInfo.LastName = 'Sohail';

-- Q.1 For an in office staff with Id number xxx, print sharjeel first name

-- sohail as last name and pay rate.

SELECT EmployeeInfo.EmployeeNo, EmployeeInfo.FirstName,EmployeeInfo.LastName,

EmployeeInfo.PaymentRate

FROM EmployeeInfo

WHERE EmployeeInfo.EmployeeNo = 'EMP44871'

-- Q.6 List names of the ingredients that was/were supplied by the supplier id

--between 2019-10-28 and 2019-10-30

SELECT IngredientsOnly.IngName, IngredientOrder.arrivalDate, Suppliers.supplierCode

FROM IngredientsOnly, IngredientOrder, Suppliers

WHERE Suppliers.supplierCode = IngredientOrder.supplierCode AND

IngredientOrder.arrivalDate BETWEEN '2019-10-28' AND '2019-10-31';

-- Q.4 Print the salary paid to a delivery staff with first name and last name in

--in current month. Current month is decided by the system

**End of the Assignment!**