Data Requirements

**1.Menu Items**

Any items on Delitastes menu will be stored as “MenuItem” these items each store a unique code for identification as well as the name of the item, the size of the item and the price the item is sold for. The menu items will also be connected to the stored ingredients to show the ingredients used on the items as well as the quantity of those ingredients.

**2.Ingredients**

Every ingredient used within Delitastes will be stored as “Ingredients”, each Ingredient will store a unique identification code as well as the name of the ingredient, type of ingredient and a description of it. Along with this each ingredient will store the stock level recorded during the last stocktake, the date of the last stocktake, the suggested current stock level and the reorder level.

**3.Ingredient Order**

Each week a stocktake of the store is taken, in which the actual levels of ingredients in the store are determined, this information along with the suggested current stock levels and reorder level are used to generate an order for the following week. The ingredient orders will be stored in the database as “IngredientOrder”. Each order will use an order number for unique identification and store the date it was made, the date at which the order is received, and the status of the order will also be stored. Information on the ingredients within the order must also be stored which will be included as the quantity of ingredients, the price of the ingredients, individual ingredient as well as a total order price.

**4.Employees**

Every employee employed at Delitastes Pizza will be stored as an “EMPLOYEE” each employee will store their employee number to identify them and personal information on the worker, i.e. first name, last name, address, contact number, description and if employed as a driver, license details and their pay rate per delivery. Financial information must also be stored on each employee, this will be their bank details, pay rate and tax file number.

**5. Customers**

Anyone who orders from Delitaste Pizza is considered a “Customer” within the database. Each customer will be assigned a unique id for accessibility and will also store their phone number. Further information on the customer stored will be an address, first name and last name. When a new customer orders they will be required to submit this information to complete their order and upon doing so the information will be paired to a unique Customer id.

**6.Orders**

Every order from Delitaste Pizza will be stored as “Order”, each order will be connected to the “MenuItem” data allowing information about the products within the order i.e. the Items ordered, quantity of each item and the price of each item. Every order will also store the date of the order, the order status, total price and payment method as well as whether the order was done via phone or walk in. In the case that the order was walk in the order will store pickup time, where as if the order was done via phone it will store pick up time if picked up from store and if delivery will store the address of delivery, time and the driver who delivered it.

**7. Employee Shifts**

Each shift worked by a worker will be stored as “EmployeeShift” this will be uniquely identified by the start date for a worker shifts individually, this will also store the start time, end time and end date of the shift. On top of this if the shift worked is delivery, it will store the number of deliveries made and if the shift is worked in store it will store the total hours of the shift.

**8. Payment**

Employees pay records will be recorded individually as “Payment” it will use a unique payslip number for identification and will store the gross pay, total pay and tax withheld. The payment will also store the date of the payment, the date the payment period begins and the date the payment period ends.

Transaction Requirements

**Data Entry**

* Enter new customer details.
* Enter new employees’ details.
* Enter the specific order details for ingredient order.
* Enter the current stocktake levels and other details.
* Enter new ingredients or menu items.
* Enter details of a customer’s order.

**Data Update/Deletion**

* Update order status to indicate completion.
* Update Employees number/address if changed.
* Delete Employee entity if resigned.
* Update HourlyRate if changed.
* Update number of deliveries when a delivery is completed.
* Update/Delete MenuItem if taken off menu.

**Data Queries**

* List DeliverDriver pay per delivery.
* Identify the TotalPay a specific employee received last shift.
* List the items on a specific order number.
* List the ingredients on a specific menu item.
* Identify InStoreWorker hourly rate.
* Calculate the total pay of a shift based on deliveries made.
* Calculate price of a customers order.
* Calculate the price of an IngredientOrder.
* Identify a Customers address.
* Find an employees Tax File Number.
* Determine what method an order was places with.
* Find the time a customer walked into the store.
* Find the phone number of a specific employee.
* Find which employee delivered a specific order.
* Find which employees worked on a specific day.
* List all details about a specific employee.
* List all DeliveryDriver employees.
* List all InStoreWorkers.
* Find price of an IngredientOrder from a specific date.
* Find price of a specific ingredient.

Business Requirements

* When a mobile order is received employee must call to verify and once confirmed that the order is real it is begun processing. If it is not confirmed the customer is marked as a HAUX and kept on hold.
* A payment approval number must be sent if a card payment.
* If a customer has not ordered before their details must be entered into the system.
* If a customer has ordered before but their details don’t match must be recognised and a new customer record must be made.
* Employees will not delete data within the database.
* The store manager must be able to use data to order new ingredients.
* Employees must record each shift they work in the database
* The amount of each ingredient remaining must be updated every time some is used.
* The results of the weekly stocktake must be input into the database.
* The results of the weekly stocktake must be input into the database.
* An employee can only be either an in-store worker or a delivery driver.
* Employee status can only be either full time or part time.
* Payments can only be added by accounting staff.





Data Dictionary:

Entity Type:

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Order | Describes orders customers have made and their details. | Pizza Order | Happens when an order is made from DeliTastes. |
| PickUpOrder | Specifies that an order is for pickup. | NA | Happens whenever an order is to be picked up. |
| DeliveryOrder | Specifies that an order is for delivery. | NA | Happens whenever a delivery order is made. |
| WalkInOrder | Specifies that an order was ordered in store. | NA | Happens whenever an order is made within the store. |
| PhoneOrder | Specifies an order was made over the phone | NA | Happens whenever a customer calls the store for an order. |
| Customer | Contains details on DeliTastes customers used when processing orders. | Clients | Created when someone orders from Delitastes for the first time. |
| Employees | Contains personal, financial and workplace information on all employees at DeliTastes. | Workers  Staff | Occurs whenever payments are made or when orders are processed. |
| DeliveryDriver | Specifies that an employee is a delivery driver and stores relevant info. | NA | Happens whenever a delivery driver is payed or makes a delivery. |
| InStoreWorker | Specifies an employee works within the store and stores relevant info. | NA | Happens whenever a new employee is hired or an order is processed. |
| MenuItem | Describes the items on the menu at DeliTastes. | NA | The size of the menu has not been specified. |
| Ingredients | Describes the ingredients used on all items at DeliTastes | NA | The range of ingredients within the store has not been specified. |
| IngredientOrder | Details the ingredients being ordered for the next stocktake period as well as the price and time of arrival of order. | Stock Order  Stock Take | Occurs at the end of each stocktake period. |
| EmployeeShift | Describes date and time of shifts worked by each employee. | Roster | Occurs separately as a Weak entity for each employee. |
| InStoreShift | Specifies hours worked within a shift in store. | NA | Whenever as shift is worked by an InStore Empoyee. |
| DeliveryShift | Specifies number of deliveries made. | NA | Whenever a delivery shift is worked by a driver. |
| Payment | Describes the pay rates for specific employees based on their shifts. | NA | Occurs separately as a Weak entity for each employee’s shift. |
| InStorePayment | Determines the pay rates for in store shifts specifically for the Payment to use. | NA | Occurs when an in store employee works a shift. |
| DeliveryPay | Determines the total pay for delivery shifts being used within Payment. | NA | Occurs when a delivery driver works a shift. |

Relationships:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity Name | Multiplicity | Relationship | Multiplicity | Entity Name |
| Customer | 1..1 | Places | 0..\* | Order |
| Order | 0..\* | Has | 1..\* | MenuItem |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | PhoneOrder |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | WalkInOrder |
| WalkInOrder | {Mandatory} | Generalisation | {Mandatory} | PickUpOrder |
| PhoneOrder | {Mandatory,Or} | Generalisation | {Mandatory,Or} | PickUpOrder |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | DeliveryOrder |
| MenuItem | 1..\* | Includes | 0..\* | Ingredients |
| IngredientOrder | 1..\* | Contains | 0..\* | Ingredients |
| Employee | {Mandatory,Or} | Generalisation | {Mandatory,Or} | DeliveryDriver |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | InStoreWorker |
| 1..1 | Works | 0..\* | EmployeeShift |
| InStoreWorker | 1..1 | Processes | 0..\* | Order |
| DeliveryDriver | 1..1 | Does | 0..\* | DeliveryShift |
| EmployeeShift | {Mandatory,Or} | Generalisation | {Mandatory,Or} | DeliveryShift |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | InStoreShift |
| InStoreShift | 1..1 | Determines | 1..1 | InStorePay |
| DeliveryShift | 1..1 | Completes | 0..\* | DeliveryOrder |
| 1..1 | Determines | 1..1 | DeliveryPay |
| Payment | 0..\* | SentTo | 1..1 | Employee |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | InStorePay |
| {Mandatory,Or} | Generalisation | {Mandatory,Or} | DeliveryPay |

Attributes:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entity | Attributes | Description | Data Type & Length | Nulls | Multi-valued | Derived | Default |
| Customer | CustomerId{PK} | Unique Customer Identifier. | Char(10) | N | N | N | None |
| FirstName | Customers First Name. | VarChar(50) | N | N | N | None |
| LastName | Customers Last Name. | VarChar(50) | N | N | N | None |
| PhoneNumber | Customers Phone Number. | Int(10) | N | N | N | None |
| Address | Customers home address. | VarChar(99) | N | N | N | None |
| Orders | Customers order Id, shown through relationship. | Char(10) | N | N | N | None |
| Order | OrderNumber{PK} | Unique Order Identifier. | Char(10) | N | N | N | None |
| TotalPrice | Total Price of order. | VarChar(10) | N | N | Y | None |
| OrderDate | Date and time the order was placed. | Datetime | N | N | N | None |
| OrderStatus | Details how much of the order has been complete. | VarChar(99) | N | N | N | None |
| ItemsInOrder | Details the items within the order and their prices. Shown through relation with MenuItems. | Char(10) | N | Y | N | None |
| PaymentApproval | Used for card orders to approve payment. | Bool | Y | N | N | None |
| PhoneOrder | TimeCallAnswered | Indicates the time the call was answered. | Time | N | N | N | None |
| TimeCallEnded | Indicates the time the call was ended. | Time | N | N | N | None |
| WalkInOrder | WalkInTime | Time the customer walked in the store. | Time | N | N | N | None |
| PickUpOrder | PickupTime | Time the order was picked up from store. | Time | N | N | N | None |
| DeliveryOrder | DeliveryTime | Time the order arrived at the customer. | Time | N | N | N | None |
| DriverDelivered | Details the driver who drove the order. | Char(10) | N | N | N | None |
| MenuItem | ItemCode {PK} | PrimaryKey used to identify specific occurrence. | Char(10) | N | N | N | None |
| Name | Name of item on menu. | VarChar(20) | N | Y | N | None |
| CurrentPrice | Price of the item. | Decimal(4,2) | N | N | N | None |
| Size | Describes the size of the menu item. | VarChar() | N | N | N | None |
| IngredientsUsed | Is implied through relationship in EER, and shows the ingredients used and the amount of them. | Char(10) | N | Y | N | None |
| Ingredient  Order | IngredientOrder  Number {PK} | Used as a primary identification key for entity. | Char(10) | N | N | N | None |
| OrderDate | Records the date the order was done. | Date | N | N | N | None |
| OrderArrivalDate | Records the date the order arrives. | Date | Y | N | N | None |
| IngredientPrice | Price of Ingredient | Decimal(6,2) | N | Y | N | None |
| Status | Tracks Status of the ingredient Order. | VarChar(99) | N | N | N | None |
| TotalPrice | Total price of the order. | Decimal(8,2) | N | N | Y | None |
| IngredientsOrdered | Is a relationship attribute to ingredients that shows the ingredients used and the amount of each ingredient. | Char(10) | N | Y | N | None |
| Ingredients | IngredientCode {PK} | Used to identify specific ingredient. | Char(10) | N | N | N | None |
| Description | Describes ingredient. | VarChar(99) | Y | N | N | None |
| CurrentLevel | Stores current stock level of ingredient. | VarChar(10) | N | N | N | None |
| LastStocktake | Last date stocktake was taken at. | Date | N | N | N | None |
| SuggestedLevel | Suggested level of current ingredient. | VarChar(10) | N | N | N | None |
| RestockLevel | Indicates the level at restock. | VarChar(10) | Y | N | N | None |
| IngredientType | Describes the type of ingredient. | VarChar(99) | Y | N | N | None |
| Employee | EmployeeNumber {PK} | Unique identification for employees. | Char(10) | N | N | N | None |
| FirstName | Employees first name. | VarChar(20) | N | N | N | None |
| LastName | Employees last name. | VarChar(20) | N | N | N | None |
| Address | Employees address. | VarChar(99) | N | N | N | None |
| ContactNumber | Details an employees contact number. | Int(1) | N | N | N | None |
| TFN | Details employees tax file number. | Char(10) | N | N | N | None |
| BankDetails | Stores an employees Account Number and BSB, | VarChar(20) | N | Y | N | None |
| Decription | Describes employees physical details. | VarChar(99) | Y | N | N | None |
| DeliveryDriver | LicenseDetails | Employees license details for delivery drivers. | Char(16) | N | N | N | None |
| RatePerDelivery | Amount a driver is payed per delivery driven. | Decimal(4,2) | N | N | N | None |
| InStoreWorker | HourlyRate | Amount employee is payed per hour. | Decimal(4,2) | N | N | N | None |
| EmployeeShift | ShiftNumber {PK} | Unique Identifier for an individual shift based off of the date and employee Number | VarChar(99) | N | N | N | None |
| StartTime | Time the employees shift starts. | Time | N | N | N | None |
| EndTime | Time the employee shift ends. | Time | N | N | N | None |
| StartDate | Date the shift begins. | Date | N | N | N | None |
| EndDate | Date the shift ends. | Date | N | N | N | None |
| DeliveryShift | NumberOfDeliveries | The number of delivieries a driver did within a shift. | Int | N | N | N | None |
| InStoreShift | ShiftPaymentAmount | The total amount earned in a shift derived from hours worked and hourly rate. | Decimal(6,2) | N | N | Y | None |
| Payment | PayslipNumber | Unique identifier for each payslip. | Char(10) | N | N | N | None |
| GrossPayment | Total pay before tax. | Decimal(6,2) | N | N | N | None |
| TaxWitheld | Tax deducted from employees pay. | Decimal(6,2) | Y | N | N | None |
| TotalPaid | Amount sent to employees bank. | Decimal(6,2) | N | N | N | None |
| PaymentDate | Records the date at which the payment is made. | Date | N | N | N | None |
|  | PaymentPeriodStart | Records earliest date in payment period. | Date | N | N | N | None |
| PaymentPeriodEnd | Records final date in the payment period. | Date | N | N | N | None |
| DeliveryPay | PaymentForShift | Determinesthe pay for a delivery shift. | Decimal(6,2) | N | N | Y | None |
| InStorePay | ShiftPaymentAmount | Determines the pay for an shift instore. | Decimal(6,2) | N | N | Y | None |

**Relational Schema:**

I performed quite well in the previous assignment with my only marks being lost on having forgotten to write out the least of relations that had been normalized. So this time that has been provided at the end of that section.

**Pre Normalized Relational Model:**

**Customer(**CustomerID,FirstName,LastName,PhoneNumber,Address**)**

**Primary Key** CustomerID

**Order(**OrderNumber,CustomerID,EmployeeNumberTotalPrice,OrderDate,OrderStatus,PaymentApprovalNumber**)**

**Primary Key** OrderNumber

**Foreign Key** CustomerID **References Customer(**CustomerID**)**  **ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**PhoneOrder(**OrderNumber, TimeCallAnswered,TimeCallEnded**)**

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**WalkInOrder(**OrderNumber, TimeEntered**)**

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**PickUpOrder(**OrderNumber ,PaymentApprovalNumber,TimeCallAnswered,TimeCallEnded,PickupTime**)**

**PrimaryKey** OrderNumber

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**DeliveryOrder(**OrderNumber ,ShiftID,DeliveryTime,DriverDelivered**)**

**PrimaryKey** OrderNumber

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Has(**OrderNumber,ItemCode,Quantity**)**

**Primary Key** OrderNumber,ItemCode

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ItemCode **References MenuItem(**ItemCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**MenuItem(**ItemCode,Name,CurrentPrice,Size,Descriptions**)**

**Primary Key** ItemCode

**Includes(**ItemCode,IngredientCode,Quantity**)**

**Primary Key** ItemCode, IngredientCode

**Foreign Key** ItemCode **References MenuItem(**ItemCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Ingredients(**IngredientCode,Name,Description,CurrentLevel,LastStocktake,SuggestedLevel,RestockLevel,IngredientType**)**

**Primary Key** IngredientCode

**IngredientOrder(**IngredientOrderNumber,IngredientCode,OrderDate,OrderArrivalDate,IngredientPrice,Status,TotalPrice**)**

**Primary Key** IngredientOrderNumber

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Contains(**IngredientOrderNumber,IngredientCode,Quantity**)**

**Primary Key** IngredientOrderNumber,IngredientCode

**Foreign Key** IngredientOrderNumber **References IngredientOrder(**IngredientOrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Employee(**EmployeeNumber,FirstName,LastName,Address,ContactNumber,TaxFileNumber,BankDetails,Type,Description**)**

**Primary Key** EmployeeNumber

**InStoreWorker(**EmployeeNumber,HourlyRate**)**

**Primary Key** EmployeeNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**DeliveryDriver(**EmployeeNumber,LicenseDetails,RatePerDelivery**)**

**Primary Key** EmployeeNumber

**Foreign Key** EmployeeNumber **Refererences Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**EmployeeShift(**ShiftID,StartDate,StartTime,EndTime,EndDate**)**

**Primary Key** ShiftID

**InStoreShift(**EmployeeNumber,ShiftID,WorkedHours**)**

**Primary Key** EmployeeNumber,ShiftID

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References InStoreWorker(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References EmployeeShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**DeliveryShift(**EmployeeNumber,NumberOfDeliveries,ShiftID**)**

**Primary Key** EmployeeNumber,ShiftID

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References DeliveryDriver(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References EmployeeShift(**StartID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**InStorePay(**PayslipNumber, PaidHours,PaidRate,ShiftID**)**

**Primary Key** PayslipNumber

**Foreign Key**  PayslipNumber **References Payment(**PayslipNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**ForeignKey** ShiftID **References InStoreShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**DeliveryPay(**PayslipNumber,NumberOfDeliveries,RatePerDelivery,ShiftID**)**

**PrimaryKey** PayslipNumber

**Foreign Key** PayslipNumber **References Payment(**PayslipNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**ForeignKey** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Payment(**PayslipNumber,EmployeeNumber,GrossPayment,TaxWitheld,TotalPaid,PaymentDate,PaymentPeriodEnd**)**

**Primary Key** PayslipNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Normalized BCNF:**

**Customer(**CustomerID,FirstName,LastName,PhoneNumber,Address**)**

**Primary Key** CustomerID

FD1: CustomerID -> FirstName,LastName,PhoneNumber,Address

The only functional dependency that exists is from a key to non prime attributes meaning the relation is already in BCNF. (Assuming the business allows multiple registers of the same phone number address and name).

**Order(**OrderNumber,CustomerID,EmployeeNumber,TotalPrice,OrderDate,OrderStatus,PaymentApprovalNumber**)**

**Primary Key** OrderNumber

**Foreign Key** CustomerID **References Customer(**CustomerID**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: OrderNo -> CustomerID,EmployeeNumber,TotalPrice,OrderDate,OrderStatus,PaymentApprovalNumber

No Values are created as composites so 1st normal form.

Since every non-candidate key is fully functionally dependent on candiadate key OrderNo it is in 2nd Normal form

Since there is no transititve dependency it is in 3rd Normal Form.

And since OrderNo is a candidate Key it is in BCNF.

**PhoneOrder(**OrderNumber, TimeCallAnswered,TimeCallEnded**)**

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: OrderNumber -> TimeCallAnswered,TimeCallEnded

Only functional dependency that exists is from a super key so BCNF.

**WalkInOrder(**OrderNumber, TimeEntered**)**

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: OrderNumber -> Time Entered.

BCNF.

**DeliveryOrder(**OrderNumber ,ShiftID,DeliveryTime,DriverDelivered**)**

**PrimaryKey** OrderNumber

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: OrderNumber -> ShiftID,DeliveryTime,DriverDelivered

FD2: ShiftID-> DriverDelivered

FD3: DeliveryTime,DriverDelivered-> OrderNumber,ShiftID

Since ShiftID isn’t a super key this is only in 3rd Normal Form.

Normalization Process:

**DeliveryOrderOnly(**OrderNumber,DeliveryTime,DriverDelivered)

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** DriverDelivered **References OrderDriverDetails(**DriverDelivered**)**

**OrderDriverDetails(**DriverDelivered,ShiftID**)**

**Primary Key** DriverDelivered

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**Is now BCNF as all possible determinates are super keys.

**Has(**OrderNumber,ItemCode,Quantity**)**

**Primary Key** OrderNumber,ItemCode

**Foreign Key** OrderNumber **References Order(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ItemCode **References MenuItem(**ItemCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: OrderNumber -> ItemCode,Quantity.

The table is in BCNF already as there is only one Dependency that is dependent on a primary key.(Unsure as what if the only order ever placed under the new system was added because in that case you could determine ordernumber based off its non prime attributes.)

**MenuItem(**ItemCode,Name,CurrentPrice,Size,Descriptions**)**

**Primary Key** ItemCode

FD1: ItemCode -> Name,CurrentPrice,Size,Description

FD2: Name -> CurrentPrice,Size,Description,ItemCode

Determinates are all super keys meaning this is already in BCNF.

**Includes(**ItemCode,IngredientCode,Quantity**)**

**Primary Key** ItemCode, IngredientCode

**Foreign Key** ItemCode **References MenuItem(**ItemCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: ItemCode -> IngredientCode, Quantity.

Since there is only one functional dependency and the relation is made of all candidate keys it has to be in BCNF.

**Ingredients(**IngredientCode,Name,Description,CurrentLevel,LastStocktake,SuggestedLevel,RestockLevel,IngredientType**)**

**Primary Key** IngredientCode

FD1: IngredientCode -> Name,Description,CurrentLevel,LastStocktake,SuggestedLevel,RestockLevel,IngredientType

FD2:Name -> IngredientCode,Description,CurrentLevel,LastStocktake,SuggestedLevel,RestockLevel,IngredientType (Since ingredients are uniquely named.)

Since all determinates are super-keys this is already in BCNF.

**IngredientOrder(**IngredientOrderNumber,IngredientCode,OrderDate,OrderArrivalDate,IngredientPrice,Status,TotalPrice**)**

**Primary Key** IngredientOrderNumber

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: IngredientOrderNumber -> IngredientCode,OrderDate,OrderArrivalDate,IngredientPrice,Status,TotalPrice

FD2:OrderArrivalDate,TotalPrice -> IngredientOrderNumber.

FD3:IngredientCode->IngredientPrice

The entity is not in BCNF as there is a non super key as a determinate to other keys.

Normalization Process:

**IngredientOrderDetails(**IngredientOrderNumber,OrderDate,TotalPrice,IngredientCode**)**

**Primary Key** IngredientOrderNumber

**Foreign Key** OrderDate **References IngredientOrderDates(**OrderDate**)**

**IngredientOrderDates(**OrderDate,OrderArrivalDate**)**

**Primary Key** OrderDate,OrderArrivalDate

With this relational mapping the entity is in BCNF, as all determinates will be a super key.

**Contains(**IngredientOrderNumber,IngredientCode,Quantity**)**

**Primary Key** IngredientOrderNumber,IngredientCode

**Foreign Key** IngredientOrderNumber **References IngredientOrder(**IngredientOrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** IngredientCode **References Ingredient(**IngredientCode**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: Ingredient OrderNumber-> Ingredient Code,Quantity

Since the only functional dependency for the entity is from a candidate key it is in BCNF.

**Employee(**EmployeeNumber,FirstName,LastName,Address,ContactNumber,TaxFileNumber,BankDetails,Type,Description**)**

**Primary Key** EmployeeNumber

FD1: EmployeeNumber -> FirstName,LastName,Address,ContactNumber,TaxFileNumber,BankDetails,Type,Description

FD2: FirstName, LastName -> EmployeeNumber,Address,ContactNumber,TaxFileNumber,BankDetails,Type,Description

FD3: TaxFileNumber-> EmployeeNumber,FirstName,LastName,Address,ContactNumber, BankDetails,Type,Description

FD4:BankDetails -> EmployeeNumber,FirstName,LastName,Address,ContactNumber,TaxFileNumber,Type,Description

This entity is in 3rd form but not BCNF as the determinants aren’t all candidate keys.

FD5: ContactNumber-> EmployeeNumber,FirstName,LastName,Address,BankDetails,ContactNumber,TaxFileNumber,Type,Description

Since all these functional dependencies are able to determine all other attributes /columns in the entity each of their determinants are super-keys, this means that the entity is already in BCNF.

**InStoreWorker(**EmployeeNumber,HourlyRate**)**

**Primary Key** EmployeeNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: EmployeeNumber->HourlyRate

The entity is already in BCNF since its super-entity is normalized previously.

**DeliveryDriver(**EmployeeNumber,LicenseDetails,RatePerDelivery**)**

**Primary Key** EmployeeNumber

**Foreign Key** EmployeeNumber **Refererences Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1:EmployeeNumber-> LicenseDetails,RatePerDelivery

FD2:LicenseDetails->EmployeeNumber,RatePerDelivery

Since license details are unique they can be used to identify the employeenumber they belong to however since it identifies all the other attributes in the entity it is a super key meaning this is already BCNF.

**EmployeeShift(**ShiftID,StartDate,StartTime,EndTime,EndDate**)**

**Primary Key** ShiftID

FD1: ShiftID -> StartDate,StartTime,EndDate,EndTime

FD2:StartDate,EndDate -> ShiftID, StartTime,EndTime

All Determinates are Super keys meaning it is in BCNF.

**InStoreShift(**EmployeeNumber,ShiftID,WorkedHours**)**

**Primary Key** EmployeeNumber,ShiftID

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References InStoreWorker(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References EmployeeShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1:ShiftID->WorkedHours,EmployeeNumber

Already in BCNF as the only functional dependency is with a super key.

**DeliveryShift(**EmployeeNumber,NumberOfDeliveries,ShiftID**)**

**Primary Key** EmployeeNumber,ShiftID

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** EmployeeNumber **References DeliveryDriver(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References EmployeeShift(**StartID**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: ShiftID->NumberOfDeliveries,EmployeeNumber

Already in BCNF same as previous.

**InStorePay(**PayslipNumber, PaidHours,PaidRate,ShiftID**)**

**Primary Key** PayslipNumber

**Foreign Key**  PayslipNumber **References Payment(**PayslipNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**ForeignKey** ShiftID **References InStoreShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1:PayslipNumber -> PaidHours,PaidRate,ShiftID

FD2:ShiftID-> PayslipNumber,PaidHours,PaidRate

Already in BCNF as no other attribute can find unique values.

**DeliveryPay(**PayslipNumber,NumberOfDeliveries,RatePerDelivery,ShiftID**)**

**PrimaryKey** PayslipNumber

**Foreign Key** PayslipNumber **References Payment(**PayslipNumber**) ON UPDATE CASCADE, ON DELETE CASCADE  
ForeignKey** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1: PayslipNumber -> NumberOfDeliveries,RatePerDelivery,ShiftID

FD2:ShiftID->PayslipNumber,NumberOfDeliveries,RatePerDelivery

Since all determinates are super keys it is in BCNF.

**Payment(**PayslipNumber,EmployeeNumber,GrossPayment,TaxWitheld,TotalPaid,PaymentDate,PaymentPeriodEnd**)**

**Primary Key** PayslipNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

FD1:PayslipNumber-> EmployeeNumber,GrossPayment,TaxWitheld,TotalPaid,PaymentDate,PaymentPeriodEnd

FD2: GrossPayment,TaxWitheld -> TotalPaid

FD3:TotalPaid,TaxWithheld -> GrossPayment

FD4:EmployeeNumber,PaymentPeriodEnd -> PayslipNumber,GrossPayment,TaxWitheld,TotalPaid,PaymentDate

This relation is not in BCNF as there are non super entities that can be determinates for a subset of the attributes within the table.

This means it is in in 3rd Normal Form but not BCNF.

Normalization Process:

**Payment(**PayslipNumber,EmployeeNumber,TotalPaid,PaymentDate,PaymentPeriod,End**)**

**Primary Key** PayslipNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** TotalPaid **References PaymentDetails(**TotalPaid**) ON UPDATE CASCADE, ON DELETE CASCADE**

**PaymentDetails(**PayslipNumber,TotalPaid,GrossPayment,TaxWithheld**)**

**Primary Key** TotalPaid,PayslipNumber

**Foreign Key** PayslipNumber **References Payment(**PayslipNumber**)**

this normalization ensures the relation is in BCNF as all determinates are super keys.

**List Of Normalized BCNF:**

**Payment(**PayslipNumber,EmployeeNumber,TotalPaid,PaymentDate,PaymentPeriod,End**)**

**Primary Key** PayslipNumber

**Foreign Key** EmployeeNumber **References Employee(**EmployeeNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** TotalPaid **References PaymentDetails(**TotalPaid**) ON UPDATE CASCADE, ON DELETE CASCADE**

**PaymentDetails(**PayslipNumber,TotalPaid,GrossPayment,TaxWithheld**)**

**Primary Key** TotalPaid,PayslipNumber

**Foreign Key** PayslipNumber **References Payment(**PayslipNumber**)**

**IngredientOrderDetails(**IngredientOrderNumber,OrderDate,TotalPrice,IngredientCode**)**

**Primary Key** IngredientOrderNumber

**Foreign Key** OrderDate **References IngredientOrderDates(**OrderDate**)**

**IngredientOrderDates(**OrderDate,OrderArrivalDate**)**

**Primary Key** OrderDate,OrderArrivalDate

**DeliveryOrderOnly(**OrderNumber,DeliveryTime,DriverDelivered)

**Primary Key** OrderNumber

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** DriverDelivered **References OrderDriverDetails(**DriverDelivered**)**

**OrderDriverDetails(**DriverDelivered,ShiftID**)**

**Primary Key** DriverDelivered

**Foreign Key** OrderNumber **References PhoneOrder(**OrderNumber**) ON UPDATE CASCADE, ON DELETE CASCADE**

**Foreign Key** ShiftID **References DeliveryShift(**ShiftID**) ON UPDATE CASCADE, ON DELETE CASCADE**Is now BCNF as all possible determinates are super keys.

**Physical Database Design:**

DROP TABLE IngredientOrder;

DROP TABLE Ingredient;

DROP TABLE MenuItem;

DROP TABLE PhoneOrder;

DROP TABLE DeliveryOrder;

DROP TABLE WalkInOrder;

DROP TABLE [ORDER];

DROP TABLE InStoreShift;

DROP TABLE DeliveryShift;

DROP TABLE InStoreWorker;

DROP TABLE DeliveryDriver;

DROP TABLE EmployeeShift;

DROP TABLE InStorePay;

DROP TABLE DeliveryPay;

DROP TABLE Payment;

DROP TABLE Employee;

DROP TABLE Customer;

--DATABASE CREATION

--CREATE DATABASE DelitastePizza

--CUSTOMER TABLE

create table Customer(

CustomerID Char(10) primary key,

FirstName VarChar(50),

LastName VarChar(50),

PhoneNumber Int,

Address VarChar(99));

--EMPLOYEE TABLE

create table Employee (

EmployeeNumber Char(10) primary key,

FirstName VarChar(20),

LastName VarChar(20),

Address VarChar(20),

ContactNumber Int,

TaxFileNumber Char(10),

BankDetails VarChar(20),

Description VarChar(99));

--INSTOREWORKER TABLE

create table InStoreWorker (

EmployeeNumber Char(10) primary key,

HourlyRate DECIMAl(2)

FOREIGN KEY (EmployeeNumber) REFERENCES Employee(EmployeeNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--DELIVERYDRIVER TABLE

create table DeliveryDriver (

EmployeeNumber Char(10) primary key,

LicenseDetails Char(16),

RatePerDelivery DECIMAL(2),

FOREIGN KEY (EmployeeNumber) REFERENCES Employee(EmployeeNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--EMPLOYEESHIFT TABLE

create table EmployeeShift(

EmployeeNumber Char(10),

ShiftID Char(10) primary key,

StartDate Date,

StartTime Time,

EndDate Date,

EndTime Time,

FOREIGN KEY (EmployeeNumber) REFERENCES Employee(EmployeeNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--INSTORESHIFT TABLE

create table InStoreShift(

EmployeeNumber Char(10),

ShiftID Char(10),

WorkedHours Decimal(2),

Primary Key (EmployeeNumber,ShiftID),

FOREIGN KEY (EmployeeNumber) REFERENCES InStoreWorker(EmployeeNumber)

ON UPDATE NO ACTION ON DELETE NO ACTION,

FOREIGN KEY (ShiftID) REFERENCES EmployeeShift(ShiftID)

ON UPDATE NO ACTION ON DELETE NO ACTION );

--DELIVERYSHIFT TABLE

create table DeliveryShift (

EmployeeNumber Char(10),

ShiftID Char(10),

NumberOfDeliveries Int,

Primary Key (EmployeeNumber,ShiftID),

FOREIGN KEY (EmployeeNumber) REFERENCES DeliveryDriver(EmployeeNumber)

ON UPDATE NO ACTION ON DELETE NO ACTION,

FOREIGN KEY (ShiftID) REFERENCES EmployeeShift(ShiftID)

ON UPDATE NO ACTION ON DELETE NO ACTION );

--PAYMENT TABLE

create table Payment (

PayslipNumber Char(10) primary key,

EmployeeNumber Char(10),

GrossPayment Decimal(2),

TaxWithheld Decimal(2),

TotalPaid Decimal(2),

PaymentDate Date,

PaymentPeriodEnd Date NOT NULL,

FOREIGN KEY (EmployeeNumber) REFERENCES Employee(EmployeeNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--INSTOREPAY TABLE

create table InStorePay (

PayslipNumber Char(10) primary key,

PaidHours Decimal NOT NUll,

PaidRate Decimal NOT NULL,

ShiftID Char(10) NOT NULL,

FOREIGN KEY (PayslipNumber) REFERENCES Payment(PayslipNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--DELIVERYPAY TABLE

create table DeliveryPay (

PayslipNumber Char(10) primary key,

NumberOfDelivieries Int NOT NULl,

RatePerDelivery Decimal(2) NOT NULl,

ShiftID Char(10) NOT NULL,

FOREIGN KEY (PayslipNumber) REFERENCES Payment(PayslipNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--ORDER TABLE

create table [Order](

OrderNumber Char(10) primary key,

CustomerID Char(10) NOT NULL,

TotalPrice VarChar(10) NOT NULL,

OrderDate DateTime NOT NULL,

OrderStatus VarChar(99) DEFAULT 'Processing',

PaymentApprovalNumber Bit,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

ON UPDATE CASCADE ON DELETE CASCADE,

EmployeeNumber Char(10) FOREIGN KEY REFERENCES Employee(EmployeeNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--PHONEORDER TABLE

create table PhoneOrder (

OrderNumber Char(10) primary key,

TimeCallAnswered TIME NOT NULL,

TimeCallEnded TIME NOT NULL,

FOREIGN KEY(OrderNumber) REFERENCES [ORDER](OrderNumber));

--WalkInOrder

create table WalkInOrder (

OrderNumber Char(10) primary key,

TimeEntered TIME NOT NULL,

FOREIGN KEY(OrderNumber) REFERENCES [Order](OrderNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--DELIVERY ORDER TABLE

create table DeliveryOrder (

OrderNumber Char(10) primary key,

ShiftID Char(10) NOT NULL,

DeliveryTime TIME NOT NULL,

EmployeeNumber Char(10) NOT NULL,

FOREIGN KEY(OrderNumber) REFERENCES [Order](OrderNumber)

ON UPDATE CASCADE ON DELETE NO ACTION,

FOREIGN KEY (EmployeeNumber) REFERENCES InStoreWorker(EmployeeNumber)

ON UPDATE NO ACTION ON DELETE NO ACTION,

FOREIGN KEY (ShiftID) REFERENCES EmployeeShift(ShiftID)

ON UPDATE NO ACTION ON DELETE NO ACTION );

--MENUITEM TABLE

create table MenuItem (

ItemCode Char(10) PRIMARY KEY,

OrderNumber Char(10) NOT NULL,

[Name] VarChar(20) NOT NULL,

Size VarChar(20) NOT NULL,

CurrentPrice Decimal(4,2) NOT NULL,

Description VarChar(99) NOT NULL,

FOREIGN KEY (OrderNumber) REFERENCES [Order](OrderNumber)

ON UPDATE CASCADE ON DELETE NO ACTION);

--INGREDIENT TABLE

create table Ingredient (

IngredientCode Char(10) PRIMARY KEY,

Description VarChar(99) NOT NULL,

CurrentLevel VarChar(10) NOT NULL,

LastStocktake Date NOT NULL ,

SuggestedLevel VarChar(10) NOT NULL ,

RestockLevel VarChar(10) NOT NULL,

IngredientType VarChar(99) DEFAULT 'NA');

--INGREDIENTORDER TABLE

create table IngredientOrder (

IngredientOrderNumber Char(10) PRIMARY KEY,

OrderDate DATE NOT NULL,

OrderArrivalDate DATE,

IngredientPrice Decimal(6,2) NOT NULL,

Status VarChar(99) NOT NULL,

TotalPrice Decimal(8,2) NOT NULL,

);

--INGREDIENT IN ORDER JUNCTION TABLE

create table IngredientsInOrder(

IngredientCode Char(10) NOT NULL,

IngredientOrderNumber Char(10) NOT NULL,

Quantity Int NOT NULL,

PRIMARY KEY (IngredientCode,IngredientOrderNumber),

FOREIGN KEY (IngredientCode) REFERENCES Ingredient(IngredientCode)

ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (IngredientOrderNumber) REFERENCES IngredientOrder(IngredientOrderNumber)

ON UPDATE CASCADE ON DELETE CASCADE);

--INGREDIENT ON ITEM TABLE

create table IngredientsOnItem(

IngredientCode Char(10) NOT NULL,

ItemCode Char(10) NOT NULL,

Quantity Int NOT NULL,

PRIMARY KEY (IngredientCode,ItemCode),

FOREIGN KEY (IngredientCode) REFERENCES Ingredient(IngredientCode)

ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (ItemCode) REFERENCES MenuItem(ItemCode)

ON UPDATE CASCADE ON DELETE CASCADE);

**Physical Database Queries:**

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

SELECT E.FirstName,E.LastName,I.HourlyRate FROM Employee E Inner JOIN InstoreWorker I

on E.EmployeeNumber = I.EmployeeNumber;

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

SELECT F.IngredientCode, F.Description, CurrentLevel,LastStocktake,SuggestedLevel,RestockLevel,IngredientType FROM Ingredient F LEFT OUTER JOIN Includes I INNER JOIN MenuItem M

ON M.ItemCode = I.ItemCode

ON F.IngredientCode = I.IngredientCode

WHERE M.Name = 'Margerita';

--Q.3 List all the shift details of a delivery staff with first name xxx and last name ttt between date yyy and zzz. (5)

SELECT DISTINCT E.\*,I.NumberOfDeliveries FROM DeliveryShift I INNER JOIN EmployeeShift E

ON I.ShiftID = E.ShiftID

WHERE E.EmployeeNumber = (SELECT EmployeeNumber FROM Employee

Where FirstName = 'Michael' AND LastName = 'James' AND StartDate between '2021-10-10' and '2021-10-19')

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

SELECT DISTINCT O.\* , W.TimeEntered FROM [Order] O INNER JOIN WalkInOrder W

ON O.OrderNumber = W.OrderNumber

WHERE O.OrderDate between '2021-10-12' AND '2021-10-16' AND O.CustomerID = (SELECT CustomerID FROM Customer

WHERE FirstName = 'Ben' AND LastName = 'Wilson')

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

SELECT O.\*, W.TimeEntered, P.TimeCallAnswered,P.TimeCallAnswered,D.DeliveryTime,U.PickUpTime FROM [Order] O LEFT OUTER JOIN WalkInOrder W ON O.OrderNumber = W.OrderNumber

LEFT OUTER JOIN PhoneOrder P ON O.OrderNumber = P.OrderNumber LEFT OUTER JOIN DeliveryOrder D ON O.OrderNumber = D.OrderNumber

LEFT OUTER JOIN PickupOrder U

ON O.OrderNumber = U.OrderNumber

WHERE O.OrderDate between '2021-10-12' AND '2021-10-15' AND O.EmployeeNumber = (SELECT EmployeeNumber FROM Employee

WHERE FirstName = 'Charles' AND LastName = 'White')

--Q.6 Q.6 Print the salary paid to a delivery staff with first name xxx and last name ttt in the current month. Note the current month is the current month that is decided by the system. (10)

DECLARE @current\_month INT

SET @current\_month = MONTH(getDATE())

SELECT SUM(TotalPaid) AS TotalPaid FROM Payment P LEFT OUTER JOIN DeliveryPay D

ON P.PayslipNumber =D.PayslipNumber

WHERE MONTH(P.PaymentDate) = @current\_month

AND P.EmployeeNumber = (SELECT EmployeeNumber FROM Employee

WHERE FirstName = 'Michael' AND LastName = 'James')

-- Q.7 List the name of the menu item that is mostly ordered in the current year. Note the current year is the current year that is decided by the system. (10)

DECLARE @current\_year INT

SET @current\_year = YEAR(getDATE())

SELECT M.Name FROM MenuItem M

WHERE M.ItemCode = (SELECT top 1 ItemCode FROM Has inner join [Order] O

on Has.OrderNumber = O.OrderNumber

WHERE Year(O.OrderDate) = @current\_year

GROUP BY Has.ItemCode

ORDER BY SUM (Quantity) DESC);