



DELITASTE PIZZA

Assignment 2

COMP1140

Jacob Boyce – c3264527
C3264526@uon.edu.au

Contents

Data Requirements	2
Business Rules	4
Transactions:	5
Documentation of EER Model (Data Dictionary)	7
Entity Types	7
Relationships	8
Attributes	9
EER Diagram	15

Assignment 1 Revised

Below is a revision of assignment 1 based on the feedback received from the markers.

Summary of Comments and Changes

Comments

- Stock-tale Entity is irrelevant as the ingredient stock can be derived from the stock orders
- Although mode of order is valid it will cause some data redundancy and it suits the business rules better if it is simply a subclass of the relevant orders
- HumanAddress was a bit unconventional and unnecessary, as it is merely an attribute not an entity.
- Instore should be separated from employee as it is defined in the business rules as its own entity.
- Instore Pay should be separated from Payment Record. As they are paid by the number of hours worked in a shift. Number of deliveries is more relevant to the pay rather than the shift as they are paid on the number of deliveries.
- Many Attributes can have nulls

Relevant Changes

- Removed 'Stock Take' Entity completely
- Removed 'Mode of Order' Entity and split it into the 'Delivery' and the 'Pickup' Entities. These entities are now inherited by the 'Phone Order' and the 'Walk-In Order' in accordance with the business rules
- Removed the 'humanaddress' entity and simply made it a composite attribute in all the relevant tables
- Made Instore an official sub entity of employee rather than having it be the employee
- Removed Driver Shift, Created the Driver Pay and Instore Pay Entities. Moved relevant information about the pay to said entities.
- Allowed nulls on several attributes

Requirement Analysis

Data Requirements

Menu Item:

Delitaste Pizza has several menu items offered to customers. Menu items have a unique code identifying them, size, name and selling price. Each menu item is composed of several ingredients and for each ingredient particular quantity of said ingredients for a menu item.

Ingredient:

Each Delitaste Pizza Menu Item is composed of ingredients. An ingredient has a unique code, name, description, type, suggested stock level and a suggested re-order level. An ingredient has a 'current stock level as of the last stocktake date'. To record the quantity of a given ingredient a stocktake is

recorded once a week (i.e., all the actual stock of every ingredient in the physical store is recorded).

Stock Order

A stock-take occurs every week. In this event all the actual levels of ingredients in store and the date of the stocktake is recorded. A stocktake has a date, and many ingredients stock levels.

When stock levels are too low then new stock must be ordered, this is a stock order. Each stock order is kept and stored. This includes the order number, date of order, the date when the order was received, order status and total price. A stock order also contains for each of the ingredient ordered a quantity, price, description and ingredient code.

Customer

A customer is a person who uses Delitaste pizza. In the system a customer's phone number, name, address and whether they are or are not a hoax is recorded. A customer makes orders through Delitaste pizza.

Order

An order is made by a customer. An order is composed of menu items and for each menu item in an order there is a quantity. An order also has a total amount due, order status, price, staff id description, mode of reception and date. There are two different modes of making an order, one way is by phone and the other way is by walking in. There are 2 different modes of receiving an order, one being by pickup and the other being by delivery. An order also has a payment. If the payment type is via card, then the payment approval number is recorded.

Mode of Order:

This is how the order is taken.

Phone Order:

A phone order occurs when a customer calls Delitaste Pizza by phone to make an order. A phone order has a time when the call was made and terminated. (A phone order is a subclass of an order)

Walk-in order:

A walk-in order occurs when a customer walks into the physical store to make an order. A walk-in order records the time when the customer walked in. A walk-in order may only be received by pickup. (A walk-in order is a subclass of an order).

Mode of Reception

This is how the customer should receive the order.

Pickup order:

If an order is to be picked up by the customer in the store, then the pickup-time is recorded.

Delivery Order:

If an order is to be delivered to the customer at their desired address, then the delivery time, address and the driver to deliver the order is recorded.

Payment:

Each order has a payment. A payment is any kind of transaction that occurred in Delitaste Pizza where money is exchanged for menu items or for work. A payment consists of the amount due, and what type of payment was used (card, cash, visa, etc.) 6

Card Payment:

A card payment is a payment which occurred through a debit or credit card. For each card payment that occurs the payment approval number is recorded. (This is a subclass of payment).

Employee

An employee is a person who provides work for Delitaste Pizza in exchange for money. An employee that is not a driver is a shop worker. Shop workers are paid for the hours worked. Each employee has a number (unique code), first name, last name, postal address, contact number, tax file number, payment rate, status and description. An employee also has a set of bank details.

The bank details include a bank code, bank name and account number.

Driver:

A driver is an employee at Delitaste Pizza who delivers orders to customers. A driver delivers orders to customers. For each driver in addition to all the employee details kept their drivers license numbers are kept. A driver is paid for the number of deliveries they complete.

In-Store:

An instore employee is a person who works at Delitaste pizza and serves customer orders inside the shop. Each driver is paid on the amount of hours they work in a given shift.

Shift

Shifts are a pre-designated amount of time where an employee works for the store in exchange for money. Shifts are not regular and fixed beforehand, and employees are paid per shift. A shift has a start date, start time, end date, end time. A shift also has a payment record.

Payment Record:

A payment record is kept every time an employee is paid. This contains all the relevant details about payments to an employee's bank details. Employee payments include the gross payment, tax withheld, the total amount paid, payment date, payment start period, payment end date and the bank details of the employee. (This is a subclass of payment).

Driver Pay:

Drivers are paid on the number of orders they deliver in each shift.

In-store pay:

In store employees are paid based on the amount of the number of hours they work in store on a given shift.

Business Rules

- Drivers are paid for the number of deliveries they do, and shop workers are paid for the number of hours they complete.

- Shifts are not fixed or regular.
- Employees are paid per shift
- All employee payment records are kept.
- There are two distinct types of workers, shop workers and drivers.
- A stock-take is taken every week and is recorded in the system.
- All stock-orders are recorded in the system.
- Walk-in orders may not receive by delivery and only by pickup.
- Phone orders may be picked up or delivered.
- If a customer has not previously ordered at a store they must be recorded in the system and customers are uniquely identified by their phone number.
- All phone orders must be verified. If an order is not verified a customer is declared a hoax and the order status is changed to hold.

Transactions:

Data Manipulation Queries

- Insert, Update and Delete and Employee
- Insert, update and delete a shift
- Insert and update or delete a payment record
- Insert update and delete a customer
- Insert update and delete an order
- Insert and update or delete a stock-take
- Insert update and delete a stock order
- Insert update and delete an ingredient
- Insert update and delete a menu item (Although this was not explicitly stated in the briefing if this could not be done then there would be no way to change the menu and thus if something went wrong, for example a menu item no longer complies with regulation then the business could not change this).
- Insert update or delete bank details
- Insert update or delete a payment
- Insert update or delete a payment record

Queries

- Search all information about an employee from name, contact number or id.
- Search all shifts of an employee from name, contact number or id.
- Search all shifts past a certain date (Although this transaction is not explicitly stated in the briefing if they could not do this the business would not have the ability to contact employees to work or have any kind of pre-planning for a shift).
- Search all payment records for an employee by name, contact number or id.
- Search the details of a customer by phone number
- Search a customer's order by their phone number and vice versa (Although this query is not explicitly stated in the briefing, if they could not do this, they would not this the verification process would be impossible, and it would be impossible to work out what order a custom has whether they walk in or where an order should be delivered.
- Search all orders by a given order status (Although this query is not explicitly stated in the briefing, if they could not do this then they would not know which orders to prepare, serve, deliver and to dial back for verification and thus the business could not fundamentally operate as according to the briefing).

- Search all information about an ingredient such as stock as of last stock-take date, suggested re-order levels, suggested stock levels, costs and re-order status (if it is being re-ordered and if so, how much is being reordered, the total cost of the re-order) from the description, name or unique code of an ingredient.
- Search all information about a menu item such a price, and ingredient composition from the name or unique code.

Documentation of EER Model (Data Dictionary)

Entity Types

Entity Name	Description	Aliases	Occurrence
Menu Item	Any item that can be purchased by customers on the Delitaste Pizza Menu. Menu Items are identified by their code and casually by their name.	Pizza	In the store on the store menu. Items being cooked with ingredients. On the phone when customers make an order
Ingredient	Physical Ingredients which are used to make menu items. Ingredients can be identified by their code and casually by their name		All ingredients are stored inside the store usually away from the customers view.
StockOrder	An event where ingredients (stock) is ordered and delivered to the business. The stock order is based on the suggested re-order quantities of the ingredients		When stock is bought from external suppliers.
Order	An order is when an event where a customer purchases several items for money and then when the order is made out the customer can then receive their items.		When the customer makes an order.
Customer	A person who purchases items at the shop.		Any person who purchases item including people who walk in or order by phone.
PhoneOrder	An order which is taken by phone	Phone Order	When a customer orders something by phone
WalkInOrder	An order which is taken when a customer walks into the store	Walk In Order	When a customer enters the physical store and orders something.
Payment	An event where money is exchanged for goods or services	Transaction	When a customer gives money in exchange for menu items. When an employee is given money in exchange for their work
CardPayment	A payment where the transaction method is by card. I.e the payment is by a credit or debit card	Card Payment, Card Transaction	
Pickup	Represents an order that is being given to a customer by being picked up. That is the customer goes to the store to pick up the Pizza	Delivery Order, Pickup Order	When an order is delivered to a customer or picked up at the store by the customer
Delivery	When an order is delivered to the customer. A delivery has an address and a driver. Delivery	Delivery	When an order is delivered to the customer

	receptions can only be made by phone orders.		
Employee	A person who works in the business providing useful work in exchange for money	Staff, Worker, Shop worker	Any person who works for Delitaste pizza (i.e provides work in exchange for money)
Driver	An employee who works for Delitaste Pizza that can drive so they can deliver orders to customer. Drivers earn money on shifts based on the number of deliveries.	Delivery Person	Any employee who works for Delitaste Pizza that can drive.
Instore	An employee who works for Delitaste Pizza who works instore and is paid on an hourly basis.	Instore Worker	Any employee who works for Delitaste Pizza who works in the shop.
Shift	A span of time across two date-times which an employee provides work. Shifts can be delegated to employees whenever they are needed		When an employee turns up to work for a given amount of time.
PaymentRecord	A record that shows that an employee was paid. This includes other financial details about the payment such as the tax withheld and the bank details.	Payment Record, Employee Payment	When an employee is paid for the work that they do.
DriverPay	Drivers are paid on the number of deliveries they did in a shift. This is a payment record that includes this information	Driver Payment Record	When a driver is paid for the amount of deliveries, they did
InStorePay	InStore employees are paid on the number of hours they work. This is a payment record that includes this information	In Store Employee Payment Record	When an instore employee is paid for the number of hours they worked in a shift.

Relationships

Entity Name	Multiplicity	Relationship	Multiplicity	Entity Name
StockOrder	0..*	Orders	0..*	Ingredients
MenuItem	1..*	Contains	1..*	Ingredients
Order	0..*	Has	1..*	MenuItem
Order	0..*	Served by	1..1	Employee
Customer	1..1	Purchases	0..*	Order
Order	1..1	Has	1..1	Payment
Delivery	0..*	Delivered by	1..1	Driver
Employee	1..1	Works	0..*	Shift
Instore	1..1	Has	0..*	InstorePay
Driver	1..1	Has	0..*	DriverPay
PaymentRecord	0..1	Paid for a	1..1	Shift
Driver	Mandatory, Or	Inheritance	Mandatory, Or	Employee
InStore	Mandatory, Or	Inheritance	Mandatory, Or	Employee

DriverPay	Mandatory, Or	Inheritance	Mandatory, Or	Payment Record
DriverPay	Mandatory, Or	Inheritance	Mandatory, Or	Payment Record
Payment Record	Mandatory, And	Inheritance	Mandatory, And	Payment
CardPayment	Optional, Or	Inheritance	Optional, Or	Payment
WalkInOrder	Optional, Or	Inheritance	Optional, Or	Order
PhoneOrder	Optional, Or	Inheritance	Optional, Or	Order
Delivery	Mandatory, Or	Inheritance	Mandatory, Or	Phone Order
Pickup	Mandatory, Or	Inheritance	Mandatory, Or	Phone Order
PickUp	Mandatory, And	Inheritance	Mandatory, And	WalkInOrder

Attributes

Entity Name	Attributes	Description	Data Type & Length	Nulls	Multi-valued	Derived	Default
Ingredient	Ingredient Code	Unique code to identify an ingredient	Not enough information on the format of ingredient codes in the briefing!	No	No	No	None
	Name	Name of given ingredient	30-character varchar	No	No	No	None
	Type	The type of ingredient	30-character varchar	Yes	No	No	None
	Suggested Reorder Level	How much of an ingredient should be ordered in a stock order given it's below the suggested stock level	Positive integer	Yes	No	No	None
	Suggested Stock Level	How stock there should be of an ingredient after a stocktake	Positive Integer	Yes	No	No	None
	Quantity	How much of an ingredient there is in a store as of the last stock-take	Positive Integer	Yes	No	Yes, derived from the last stock order	
StockOrder	StockOrder Number	Unique number to identify a stock order	Not enough information on the format of a stock	No	No	No	None

			order number to answer!				
	Total cost	The total cost of a stock order	Integer	N/A	No	Yes, by summation of all of the prices of the ingredients in an order	None
	Date (Ordered)	The date when a stock order was made	Date (dd/mm/yyyy)	No	Yes	No	None
	Date (Received)	When an order is received	Date (dd/mm/yyyy)	Yes	Yes	No	None
	Status	The current status of a stock order i.e if it was purchased but not delivered, if it has been received etc	2-character char (Status can be represented in a code)	No	No	No	PR (Purchased but not received)
Menuitem	Menu Item Code	A unique identifier for a menu item	Not enough information on the format of menu codes to answer!	No	No	No	None
	Name	The name of a menu item	15-character varchar	No	No	No	None
	Size	The size of a menu item (L, M, S) etc.	2-character varchar	Yes	No	No	None
	Selling Price	The current selling price of a menu item	Float point number, size 3 with 2 decimal places	Yes	No	No	None
Order	Order Number	A unique identifier for an order	Not enough information on the order number format to answer	No	No	No	None
	Description	A small description or other information	150-character varchar	Yes	No	No	None

		about an order					
	Status	The status of a given order, for example its being prepared, ready to serve, being verified etc.	2-character char (Status can be represented in a code to save space).	No	No	No	PR (Purchased and being prepared)
	Date	The date when an order occurred	Date (dd/mm/yyyy)	No	No	No	None
	Type	The type of order (walk in, phone order)	N/A	No	No	Yes, Derived from the sub-sup structure	None
	Total Price due	The total cost of a given order	Integer	Yes	No	Yes, derived from summing the cost of all menu items	None
	Payment Method	The way a payment was done	Varchar 15 character	No	No	Yes, Derived from the payment type of a payment	None
Walk-in Order	Walk-in Time	The time when a customer walked in the store to make an order	Time Format (This is usually a fraction of a day).	Yes	No	Ye	
Phone Order	Call Time	The time when a customer called the store	Time Format (as above)	Yes	No	No	None
	Termination Time	The time when a customer ended its call to the store	Time Format (As above)	Yes	No	No	None
	Call Duration	The amount of time a call lasted	Time format (as above)	Yes	No	Yes, derived from the	None

						difference in termination and call time	
Payment	Payment ID	A unique code to identify a payment transaction that occurred.	Integer	No	No	No	None
	Amount Due	The total amount of money spent in a payment	Floating point, size 5 with 2 decimal places	No	No	No	None
	Payment Type	A code to identify a type of payment	1 character char	Yes	No	No	None
Card Payment	Payment Approval Number	A unique code sent by a bank or financial instution to prove that a payment was received from a payment that occurs via card	Not enough information on the approval number format to answer	No	No	No	None
Customer	Customer ID	A unique code to identify a customer	Auto-incrementing integer	No	No	No	None
	Phone Number	The phone number of a customer	15-character varchar, digits only	No	No	No	None
	Name (First)	The first name of a customer	30-character varchar	Yes	Yes	No	None
	Name (Last)	The last name of a customer	30-character varchar	Yes	Yes	No	None
	Is Hoax	Whether a customer is or is not declared a hoax (based on the verification procedure)	Boolean	Yes	No	No	None
	Address (Street)	A number which	30 Character Varchar	Yes	Yes	No	None

		identifies which house an address is on a given street					
	Address (City)	The city in which an address lies in	30-character varchar	Yes	Yes	No	None
	Address (Postcode)	The postcode of an address	4-character variable char	Yes	Yes	No	None
Employee	Employee Number	A unique number to identify an employee	Not enough information on the format of employee numbers to answer	No	No	No	None
	Name (First)	The first name of an employee	30-character varchar	No	Yes	No	None
	Name (Last)	The last name of an employee	30-character varchar	No	Yes	No	None
	Contact Number	The contact of the employee	15-character varchar (digits only)	No	No	No	None
	Tax File Number	A unique number which the government uses to identify taxpayers	9-character fixed char	No	No	No	None
	Description	A short description about an employee's role	500-character variable char.	Yes	No	No	None
	Payment Rate	How much an employee is paid	Floating point number size 3 2 decimal places	Yes	No	No	None
	BankDetails Account Number	The unique account number of a bank account.	16-character varchar (digits only)	Yes	Yes	No	None
	BankDetails Bank Code	The Bank code that identifies a bank and its branch, also known as BSB	6-character char, digits only	Yes	Yes	No	None

	BankDetails Bank name	The name of a bank	50-character varchar	Yes	Yes	No	None
	Address (Street)	A number which identifies which house an address is on a given street	30 Character Varchar	Yes	Yes	No	None
	Address (City)	The city in which an address lies in	30-character varchar	Yes	Yes	No	None
	Address (Postcode)	The postcode of an address	4-character variable char	Yes	Yes	No	None
	Date-time(start)	The date in which the shift started	Date (dd/mm/yyyy)	Yes	Yes	No	None
	Date-time (end)	The date in which the shift ended	Date (dd/mm/yyyy)	Yes	Yes	No	None
Payment Record	Total Payment	The gross amount on a payment without tax being removed.	5-digit floating point number with 2 decimal places	No	No	No	None
	Date (Start)	The date when a payment begins	Date (dd/mm/yyyy)	No	No	No	None
	Date (End)	The date when a payment ends	Date (dd/mm/yyyy)	Yes	No	No	None
	Tax withheld	The amount of tax paid on a payment record	Floating point number 3 digits 2 decimal places	Yes	No	No	None
	Payment Period (Start)	The date from when a debt is incurred on the business	Date (dd/mm/yyyy)	Yes	No	No	None
	Payment Period (End)	The date due on a payment	Date (dd/mm/yyyy)	Yes	No	No	None
Delivery	Delivery Time	The time when the delivery is to be received	Time	No	No	No	None

	Address (Street)	A number which identifies which house an address is on a given street	30 Character Varchar	Yes	Yes	No	None
	Address (City)	The city in which an address lies in	30-character varchar	Yes	Yes	No	None
	Address (Postcode)	The postcode of an address	4-character variable char	Yes	Yes	No	None
Pickup	Pickup Time	The time when the order is to be picked up by the customer	Time	No	No	No	None
DriverPay	Number of deliveries	The amount of deliveries completed in the shift. This is what they are paid for	Tiny Integer	No	No	No	None
InStorePay	Hours Worked	The number of hours worked in a given shift. This is what they are paid for	Tiny Integer	No	No	No	None

EER Diagram

- Please find the **completed EER inside** of the **Assignments Folder**. The file is named 'EER Diagram Revised.pdf'

Relational Model

StockOrder(StockOrderNumber, DateOrdered, DateReceived, Status)

Primary Key StockOrderNumber

Ingredient(IngredientCode, Name, Description, Type, SuggestedReorderLevel, SuggestedStockLevel)

Primary Key IngredientCode

IngredientOrders(StockOrderNumber, IngredientCode, Quantity, Price)

Primary Key StockOrderNumber, IngredientCode

Foreign Key StockOrderNumber **references** StockOrder(StockOrderNumber) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key IngredientCode **references** Ingredient(IngredientCode) ON UPDATE CASCADE ON DELETE SET NULL

MenuItem(MenuItemCode, Name, Size, SellingPrice)

Primary Key MenuItemCode

MenuItemComposition(MenuItemCode, IngredientCode, Quantity)

Primary Key MenuItemCode, IngredientCode

Foreign Key MenuItemCode **References** MenuItem(MenuItemCode) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key IngredientCode **references** Ingredient(IngredientCode) ON UPDATE CASCADE ON DELETE CASCADE

Order(OrderNumber, Description, OrderStatus, Date, CustomerID, EmployeeNumber, PaymentID)

Primary Key OrderNumber

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

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE CASCADE ON DELETE SET NULL

Foreign Key PaymentID **References** Payment(PaymentID) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key PaymentID

WalkInOrder(OrderNumber, WalkInTime, PickupTime)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

PhoneDeliveryOrder(OrderNumber, CallTime, TerminationTime, DeliveryTime, AddressStreet, AddressCity, AddressPostcode, Driver)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

Foreign Key Driver **references** Driver(EmployeeNumber) ON UPDATE CASCADE ON DELETE SET NULL

PhonePickupOrder(OrderNumber, CallTime, TerminationTime, PickupTime)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

OrderItems(OrderNumber, MenuItemCode, Quantity)

Primary Key OrderNumber, MenuItemCode

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

Foreign Key MenuItemCode **References** MenuItem(MenuItemCode) ON UPDATE CASCADE ON DELETE SET NULL

Customer(CustomerID, PhoneNumber, FirstName, LastName, isHoax, AddressStreet, AddressCity, AddressPostcode)

Primary Key CustomerID

Alternate Key Phone Number

Payment(PaymentID, Amount Due, Payment Type)

Primary Key PaymentID

CardPayment(PaymentID, PaymentApprovalNumber)

Primary Key PaymentID PaymentApprovalNumber

Foreign Key PaymentID **References** Payment(PaymentID) ON UPDATE NO ACTION ON DELETE NO ACTION

Instore(EmployeeNumber, FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName)

Primary Key EmployeeNumber

Alternate Key TaxFileNumber

Driver(EmployeeNumber, FirstName, LastName, ContactNumber, TaxFileNumber, Description,

PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName, DriversLicenseNumber)

Primary Key EmployeeNumber

Alternate Key TaxFileNumber

Alternate Key DriversLicenseNumber

Shift(ShiftID, EmployeeNumber, Date Start, Time Start, Time End, Date End)

Primary Key ShiftID

Foreign Key EmployeeNumber **References** Driver(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

DriverPay(PaymentID, EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, PaymentPeriodEnd, Amount Due, PaymentType, ShiftID, NumberOfDeliveries)

Primary Key PaymentID

Foreign Key ShiftID **References** Shift ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Driver(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key ShiftID

InStorePay(PaymentID, EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, PaymentPeriodEnd, Amount Due, PaymentType, ShiftID, HoursWorked)

Primary Key PaymentID

Foreign Key ShiftID **References** SHIFT ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key ShiftID

Normalisation Process

First Normal Form

First check the schema is in first normal form (1NF). A relational schema is in 1NF if and only if

- There are only single valued attributes
- The attribute domain does not change
- Each Attribute has a unique name
- The order of data entry does not matter.

These rules apply to every entity in the database schema (by observation of the data dictionary and the relational model schema).

Second Normal Form

Check the schema is in second normal form(2NF). A relational schema is in 2NF if and only if every non-prime attribute is fully functionally dependent on the primary key and the table is in 1NF. That is only the full primary key functionally determines all attributes in an entity and not a proper subset.

Check all functional dependencies for partial dependencies. The functional dependency set in the table below is where $X \rightarrow Y$ where X is prime or a candidate key or a proper subset of the prime key and Y is non-prime and the dependency is not trivial.

Entity Name	Functional Dependencies (prime to non-prime attributes only)	Reasoning (If applicable)	Entity in 2NF?
StockOrder	StockOrderNumber \rightarrow DateOrdered, DateReceived, Status		Yes
Ingredient	IngredientCode \rightarrow name, description, type, suggested		Yes
IngredientOrders	StockOrderNumber, IngredientCode \rightarrow Quantity, Price	Quantity and price are determined only by both keys because the quantity and price are for a particular ingredient on a given stock order	Yes
MenuItem	MenuItemCode \rightarrow name, size, selling price		Yes
Menu Item Composition	MenuItemCode, IngredientCode \rightarrow Quantity	MenuItemCode and IngredientCode fully determine the quantity as the quantity is for a given ingredient on a particular menu item.	Yes
Order	OrderNumber \rightarrow description, orderstatus, date, customerid, employeenumber, paymentID PaymentID \rightarrow description, orderstatus, date, customerid, employeeNumber	OrderNumber or PaymentId could be used to identify a tuple however both have been identified as candidate keys.	Yes
WalkInOrder	OrderNumber \rightarrow walkintime, pickuptime		Yes
PhoneDelivery Order	OrderNumber \rightarrow calltime, terminationtime, deliverytime, address street, address city, address postcode		Yes
PhonePickup Order	OrderNumber \rightarrow CallTime, TerminationTime, PickupTime		Yes
OrderItems	OrderNumber, MenuItemCode, \rightarrow Quantity	OrderNumber and menuitemcode fully determine quantity as the quantity is for a particular menu item for a given order	Yes
Customer	CustomerID \rightarrow Phone Number, First Name, Last Name, is Hoax, AddressStreet, AddressCity, AddressPostcode PhoneNumber \rightarrow First Name, Last Name, is hoax, Address Street, Address City, Address Postcode		Yes

	CustomerID, PhoneNumber -> First Name, Last Name, is hoax, Address Street, Address City, Address Postcode		
Payment	PaymentID -> amount due, paymentType		Yes
CardPayment	PaymentId -> paymentapprovalNumber		Yes
InStore	EmployeeNumber -> FirstName, LastName, ConactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name	None of the attributes are functionally dependent on any proper subset of all 4 candidate keys (from business rules). All 3 candidate keys were identified as either prime or an alternate key.	Yes
Driver	EmployeeNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, DriversLicenseNumber TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, Drivers License Number DriversLicenseNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name	None of the attributes are functionally dependent on any proper subset of all 4 candidate keys (from business rules). All 4 candidate keys were identified as either prime or an alternate key.	Yes
Shift	ShiftID -> EmployeeNumber, Date Start, Date End, Time Start, Time End		Yes
DriverPay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDatened, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, NumberOfDeliveries		Yes
InStorePay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDatened, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, Hours Worked		Yes

As there are no partial dependencies present and the schema is in 1NF the schema is also in 2NF.

Third Normal Form

Check the relational schema is in third normal form (3NF). A relational schema is in 3NF if and only if the schema is in 2NF and there are no transitive dependencies that is all non-prime attributes are solely and only dependent on the primary key and not on any non-prime attribute.

Check all the entities are in 3NF. The Functional dependencies F in the table is that where $X \rightarrow Y$ where X is prime or non-prime, Y is non-prime, and the dependency is not trivial.

Entity Name	Functional Dependencies F (prime to non-prime, and non-prime to non-prime)	Reasoning (If applicable)	Entity in BCNF?
StockOrder	StockOrderNumber \rightarrow DateOrdered, DateReceived, Status	Date Ordered, Date Received and Status are only determined by the given stock order. Status cannot be derived from the date.	Yes
Ingredient	IngredientCode \rightarrow name, description, type, suggested reorder level, suggested stock level	All attributes are only determined by the given ingredient.	Yes
IngredientOrders	StockOrderNumber, IngredientCode \rightarrow Quantity, Price	Quantity and price are determined only by both keys because the quantity and price are for a particular ingredient on a given stock order	Yes
MenuItem	MenuItemCode \rightarrow name, size, selling price	All attributes are only determined by the given menu item. Although size may affect the selling price no meaningful business rule is given to formally establish this.	Yes
Menu Item Composition	MenuItemCode, IngredientCode \rightarrow Quantity	MenuItemCode and IngredientCode fully determine the quantity as the quantity is for a given ingredient on a particular menu item.	Yes
Order	OrderNumber \rightarrow description, orderstatus, date, customerid, employeenumber, paymentID PaymentID \rightarrow description, orderstatus, date, employeenumber	All attributes are only determined by the order number or the payment ID. However, the paymentID is an alternate key.	Yes
WalkInOrder	OrderNumber \rightarrow walkintime, pickuptime	All attributes are only dependent on the order number	Yes
PhoneDelivery Order	OrderNumber \rightarrow calltime, terminationtime, deliverytime, address street, address city, address postcode	Address is not determined by the customer(super) as the customer does not	Yes

		necessarily have to deliver at their home.	
PhonePickup Order	OrderNumber -> CallTime, TerminationTime, PickupTime	All attributes are fully dependent on order number	Yes
OrderItems	OrderNumber, MenuItemCode, -> Quantity	OrderNumber and menuitemcode fully determine quantity as the quantity is for a particular menu item for a given order	Yes
Customer	CustomerID -> Phone Number, First Name, Last Name, is Hoax, AddressStreet, AddressCity, AddressPostcode PhoneNumber -> First Name, Last Name, is hoax, Address Street, Address City, Address Postcode	Phone Number identifies all attributes in the entity however it was nominated as an alternate key.	Yes
Payment	PaymentID -> amount due, paymentType	All attributes are fully dependency on the paymentID.	Yes
CardPayment	PaymentId -> paymentapprovalNumber	Payment approval number is fully dependent on paymentID	Yes
InStore	EmployeeNumber -> FirstName, LastName, ConactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name	Contact Number and Tax File number also uniquely identify all fields, but they have been nominated as alternate keys.	Yes
Driver	EmployeeNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, DriversLicenseNumber TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, Drivers License Number DriversLicenseNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name	Contact Number, Tax File Number and Drivers License Number all uniquely identify all other fields however they have already been nominated as alternate keys.	Yes

Shift	ShiftID -> EmployeeNumber, Date Start, Date End, Time Start, Time End	Date start, time start and employee number cant determine the key as shifts are not fixed or scheduled in any kind of way, that is there is no business rule that explicitly specifies there cant be too shifts at the same time. Thus all attributes are determined by the shiftID.	Yes
DriverPay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, NumberOfDeliveries ShiftID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, NumberOfDeliveries	As each payment is for a shift, the shift must be unique for each payment and thus a payment is functionally dependent on a shift. However, this has been nominated as an alternate key.	Yes
InStorePay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, Hours Worked ShiftID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Hours Worked	As each payment is for a shift, the shift must be unique for each payment and thus a payment is functionally dependent on a shift. However, this has been nominated as an alternate key.	Yes

As shown the table is already in 3NF as all non-prime attributes are solely functionally dependent on the primary key.

Boyce Codd Normal Form

Check that the relational schema is in Boyce Codd Normal Form (BCNF). A schema is in BCNF is and only if for all functional dependencies $X \rightarrow Y$

- The dependency is trivial
- X is a superkey in R

Check all the entities are in BCNF. The Functional dependencies F in the table is that where $X \rightarrow Y$ where $X \rightarrow Y$ is not trivial.

Entity Name	Functional Dependencies F (prime to non-prime, and non-prime to non-prime)	Reasoning (If applicable)	Entity in BCNF?
StockOrder	StockOrderNumber -> DateOrdered, DateReceived, Status	Date Ordered, Date Received and Status are	Yes

		only determined by the given stock order. Status cannot be derived from the date.	
Ingredient	IngredientCode -> name, description, type, suggested reorder level, suggested stock level	All attributes are only determined by the given ingredient.	Yes
IngredientOrders	StockOrderNumber, IngredientCode -> Quantity, Price	Quantity and price are determined only by both keys because the quantity and price are for a particular ingredient on a given stock order	Yes
MenuItem	MenuItemCode -> name, size, selling price	All attributes are only determined by the given menu item. Although size may affect the selling price no meaningful business rule is given to formally establish this.	Yes
Menu Item Composition	MenuItemCode, IngredientCode -> Quantity	MenuItemCode and IngredientCode fully determine the quantity as the quantity is for a given ingredient on a particular menu item.	Yes
Order	OrderNumber -> description, orderstatus, date, customerid, employeenumber, paymentID PaymentID -> description, orderstatus, date, employeenumber	All attributes are only determined by the order number	Yes
WalkInOrder	OrderNumber -> walkintime, pickuptime	All attributes are only dependent on the order number	Yes
PhoneDelivery Order	OrderNumber -> calltime, terminationtime, deliverytime, address street, address city, address postcode	Address is not determined by the customer(super) as the customer does not necessarily have to deliver at their home.	Yes
PhonePickup Order	OrderNumber -> CallTime, TerminationTime, PickupTime	All attributes are fully dependent on order number	Yes
OrderItems	OrderNumber, MenuItemCode, -> Quantity	OrderNumber and menuitemcode fully determine quantity as the quantity is for a particular menu item for a given order	Yes

Customer	<p>CustomerID -> Phone Number, First Name, Last Name, is Hoax, AddressStreet, AddressCity, AddressPostcode</p> <p>PhoneNumber -> First Name, Last Name, is hoax, Address Street, Address City, Address Postcode</p>	Phone Number identifies all attributes in the entity however it was nominated as an alternate key.	Yes
Payment	PaymentID -> amount due, paymentType	All attributes are fully dependency on the paymentID.	Yes
CardPayment	PaymentId -> paymentapprovalNumber	Payment approval number is fully dependent on paymentID	Yes
InStore	<p>EmployeeNumber -> FirstName, LastName, ConactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name</p> <p>TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name</p>	Tax File number also uniquely identify all fields, but they have been nominated as alternate keys.	Yes
Driver	<p>EmployeeNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, DriversLicenseNumber</p> <p>TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name, Drivers License Number</p> <p>DriversLicenseNumber -> FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, Address Street, Address City, Address Postcode, Account Number, Bank Code, Bank Name</p>	Tax File Number and Drivers License Number all uniquely identify all other fields however they have already been nominated as alternate keys.	Yes
Shift	ShiftID -> EmployeeNumber, Date Start, Date End, Time Start, Time End	Date start, time start and employee number cant determine the key as shifts are not fixed or scheduled in any kind of way, that is there is no business rule that explicitly specifies there cant be too shifts at the same time. Thus all attributes are	Yes

		determined by the shiftID.	
DriverPay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, NumberOfDeliveries ShiftID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, NumberOfDeliveries	As each payment is for a shift, the shift must be unique for each payment and thus a payment is functionally dependent on a shift. However, this has been nominated as an alternate key.	Yes
InStorePay	PaymentID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Shift ID, Hours Worked ShiftID -> EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, Payment Period End, Amount Due, Payment Type, Hours Worked	As each payment is for a shift, the shift must be unique for each payment and thus a payment is functionally dependent on a shift. However, this has been nominated as an alternate key.	Yes

The relational schema is already in BCNF. As all entities meet the conditions.

Normalisation Process.

The schema is already in BCNF. As specified in the Assignment 2 Q & A additional functional dependencies will be assumed for two tables.

Example 1, Normalising the Driver Table

Driver(EmployeeNumber, DriversLicenseNumber, TaxFileNumber, FirstName, LastName, ContactNumber, Description, PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName)

Primary Key EmployeeNumber, DriversLicenseNumber

Assume the following functional dependencies

EmployeeNumber, DriversLicenseNumber, TaxFileNumber -> FirstName, LastName, ContactNumber, Description, PaymentRate, Address Street, Address City, AddressPostcode, Account Number, BankCode, BankName

TaxFileNumber -> AccountNumber, BankCode, BankName

Description -> PaymentRate

EmployeeNumber -> FirstName, LastName, ContactNumber

Normalising

Entity is in 1NF as there are only single valued attributes, the attribute domain is constant, the data entry order is irrelevant and each attribute is atomic.

Entity is not in 2NF as

- EmployeeNumber -> FirstName, LastName, ContactNumber
- EmployeeNumber is a candidate key

Perform Lossless Join Decomposition

Let R = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, FirstName, LastName, ContactNumber, Description, paymentRate, AddressStreet, Address City, Address Postcode, Account Number, BankCode, Bank Name

Let X = Employee Number

Let Y = FirstName, LastName, ContactNumber

Thus R-Y = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName

XY = EmployeeNumber, FirstName, LastName, ContactNumber

This leaves with the two tables

R-Y = Driver(EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, Address Postcode, AccountNumber, BankCode, BankName)

Primary Key EmployeeNumber, DriversLicenseNumber, TaxFileNumber

Foreign Key EmployeeNumber **references** Employee(EmployeeNumber)

XY = Employee(EmployeeNumber, FirstName, LastName, ContactNumber)

Primary Key EmployeeNumber

Table is still not in 2NF as partial dependencies still exist

- TaxFileNumber -> AccountNumber, BankCode, BankName

Perform Lossless Join Decomposition

Let X = TaxFileNumber

Let Y = AccountNumber, BankCode, BankName

Let R = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, Address Postcode, AccountNumber, BankCode, BankName

R-Y = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, Address Postcode

XY = TaxFileNumber, AccountNumber, BankCode, BankName

This leaves with the three tables

R-Y = Driver(EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, Address Postcod)

Primary Key EmployeeNumber, DriversLicenseNumber, TaxFileNumber

Foreign Key EmployeeNumber **references** Employee(EmployeeNumber) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key TaxFileNumber **references** AccountInfo(TaxFileNumber) ON UPDATE CASCADE ON DELETE CASCADE

XY = AccountInfo(TaxFileNumber, AccountNumber, BankCode, BankName)

Primary Key TaxFileNumber

Employee(EmployeeNumber, FirstName, LastName, ContactNumber)

Primary Key EmployeeNumber

All 3 entities are now in 2NF as all attributes are functionally dependent on only the primary key and not a proper subset of the primary key.

Entity is not in 3NF as transitive dependencies exist

Description -> PaymentRate

Perform Lossless Join Decomposition

Let X = Description

Let Y = PaymentRate

R = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, Address Postcode, AccountNumber, BankCode, BankName

R-Y = EmployeeNumber, DriversLicenseNumber, TaxFileNumber, AddressStreet, AddressCity, Address Postcode

XY = Description, PaymentRate

This means there will be 4 tables

R-Y = Driver(EmployeeNumber, DriversLicenseNumber, TaxFileNumber, AddressStreet, AddressCity, Address Postcode, Description)

Primary Key EmployeeNumber, DriversLicenseNumber, TaxFileNumber

Foreign Key EmployeeNumber **references** Employee(EmployeeNumber) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key TaxFileNumber **references** AccountInfo(TaxFileNumber) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key Description **references** PaymentRates(Description) ON UPDATE CASCADE ON DELETE SET NULL

AccountInfo(TaxFileNumber, AccountNumber, BankCode, BankName)

Primary Key TaxFileNumber

Employee(EmployeeNumber, FirstName, LastName, ContactNumber)

Primary Key EmployeeNumber

PaymentRates(Description, PaymentRate)

Primary Key Description

The table is now fully in 3NF as there are no transitive dependencies across non-keyed attributes.

The Table is fully in BCNF as there are no transitive dependencies at all.

Example 2, Normalising the order table

Order(OrderNumber, Description, OrderStatus, Date, CustomerID, EmployeeNumber, PaymentID)

Assume the following functional dependencies exist

OrderNumber, CustomerID -> PaymentID, Date, OrderStatus, EmployeeNumber, CustomerID

OrderStatus -> CustomerID.

Entity is in 1NF as there are only single valued attributes, the attribute domain is constant, the data entry order is irrelevant and each attribute is atomic.

Entity is in 2NF as there are no partial dependencies in the functional dependency set F. That is a proper subset of the candidate key does not functionally depend on any non-prime attribute.

Entity is in 3NF. There are no transitive dependencies from the candidate keys to the non-prime attributes.

Entity is not in BCNF, a transitive dependency exists where OrderStatus is not a candidate key

- OrderStatus -> CustomerID

Perform lossless join decomposition.

Let X = OrderStatus

Let Y = CustomerID

Let R = OrderNumber, CustomerID, Description, orderStatus, Date, EmployeeNumber, PaymentID

R-Y = OrderNumber, Description, OrderStatus, Date, EmployeeNumber, PaymentID

XY = OrderStatus, CustomerID

R-Y = **Order**(OrderNumber, OrderStatus, Date, EmployeeNumber, PaymentID)

Primary Key OrderNumber

Foreign Key OrderStatus **References** statusCustomer(OrderStatus) ON UPDATE CASCADE ON DELETE NO ACTION.

XY = **statusCustomer**(OrderStatus, CustomerID)

Primary Key OrderStatus

Final Relational Schema in BCNF

Note this is identical to the original relational schema as it was already in BCNF.

StockOrder(StockOrderNumber, DateOrdered, DateReceived, Status)

Primary Key StockOrderNumber

Ingredient(IngredientCode, Name, Description, Type, SuggestedReorderLevel, SuggestedStockLevel)

Primary Key IngredientCode

IngredientOrders(StockOrderNumber, IngredientCode, Quantity, Price)

Primary Key StockOrderNumber, IngredientCode

Foreign Key StockOrderNumber **references** StockOrder(StockOrderNumber) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key IngredientCode **references** Ingredient(IngredientCode) ON UPDATE CASCADE ON DELETE SET NULL

MenuItem(MenuItemCode, Name, Size, SellingPrice)

Primary Key MenuItemCode

MenuItemComposition(MenuItemCode, IngredientCode, Quantity)

Primary Key MenuItemCode, IngredientCode

Foreign Key MenuItemCode **References** MenuItem(MenuItemCode) ON UPDATE CASCADE ON DELETE CASCADE

Foreign Key IngredientCode **references** Ingredient(IngredientCode) ON UPDATE CASCADE ON DELETE CASCADE

Order(OrderNumber, Description, OrderStatus, Date, CustomerID, EmployeeNumber, PaymentID)

Primary Key OrderNumber

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

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE CASCADE ON DELETE SET NULL

Foreign Key PaymentID **References** Payment(PaymentID) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key PaymentID

WalkInOrder(OrderNumber, WalkInTime, PickupTime)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

PhoneDeliveryOrder(OrderNumber, CallTime, TerminationTime, DeliveryTime, AddressStreet, AddressCity, AddressPostcode, Driver)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

Foreign Key Driver **references** Driver(EmployeeNumber) ON UPDATE CASCADE ON DELETE SET NULL

PhonePickupOrder(OrderNumber, CallTime, TerminationTime, PickupTime)

Primary Key OrderNumber

Foreign Key OrderNumber **References** Order(OrderNumber) ON UPDATE NO ACTION ON DELETE CASCADE

OrderItems(OrderNumber, MenuItemCode, Quantity)

Primary Key OrderNumber, MenuItemCode

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

Foreign Key MenuItemCode **References** MenuItem(MenuItemCode) ON UPDATE CASCADE ON DELETE SET NULL

Customer(CustomerID, PhoneNumber, FirstName, LastName, isHoax, AddressStreet, AddressCity, AddressPostcode)

Primary Key CustomerID

Alternate Key Phone Number

Payment(PaymentID, Amount Due, Payment Type)

Primary Key PaymentID

CardPayment(PaymentID, PaymentApprovalNumber)

Primary Key PaymentID PaymentApprovalNumber

Foreign Key PaymentID **References** Payment(PaymentID) ON UPDATE NO ACTION ON DELETE NO ACTION

Instore(EmployeeNumber, FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName)

Primary Key EmployeeNumber

Alternate Key TaxFileNumber

Driver(EmployeeNumber, FirstName, LastName, ContactNumber, TaxFileNumber, Description, PaymentRate, AddressStreet, AddressCity, AddressPostcode, AccountNumber, BankCode, BankName, DriversLicenseNumber)

Primary Key EmployeeNumber

Alternate Key TaxFileNumber

Alternate Key DriversLicenseNumber

Shift(ShiftID, EmployeeNumber, Date Start, Time Start, Time End, Date End)

Primary Key ShiftID

Foreign Key EmployeeNumber **References** Driver(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

DriverPay(PaymentID, EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, PaymentPeriodEnd, Amount Due, PaymentType, ShiftID, NumberOfDeliveries)

Primary Key PaymentID

Foreign Key ShiftID **References** Shift ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Driver(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key ShiftID

InStorePay(PaymentID, EmployeeNumber, TotalPayment, TaxWithheld, PaymentDateStart, PaymentDateEnd, PaymentPeriodStart, PaymentPeriodEnd, Amount Due, PaymentType, ShiftID, HoursWorked)

Primary Key PaymentID

Foreign Key ShiftID **References** SHIFT ON UPDATE NO ACTION ON DELETE NO ACTION

Foreign Key EmployeeNumber **References** Instore(EmployeeNumber) ON UPDATE NO ACTION ON DELETE NO ACTION

Alternate Key ShiftID