

**Informatics Institute of Technology**

**Department of Computing**

**(B.Eng.) in Software Engineering**

Module: Database Systems

5COSC008C

**Database Systems Coursework (2020/2021)**

# Coursework Part A: Conceptual ERD

## Part A Project: FOODTOOYOU

**Date of Submission: 26/10/2020**

Module Leader – Mr. Ragu Sivaraman

Name : Oshadha Malith Goonathilake

UoW ID - w1762649

Student ID - 2018402

Group - E

**Table of Contents**

[Introduction 3](#_Toc54463040)

1) [Produce a complete CONCEPTUAL](#_Toc54463041) [ERD for FOODTOOYOU. 4](#_Toc54463042)

[2) Create a data dictionary to document how you identified the entities for FOODTOOYOU. 6](#_Toc54463043)

[3) Create a data dictionary to document how you identified the relationships and multiplicities for FOODTOOYOU. 8](#_Toc54463044)

[4) Create a data dictionary to document how you identified the attributes and primary keys for each entity for FOODTOOYOU.](#_Toc54463045) 12

[Conclusion](#_Toc54463046) 18

[References](#_Toc54463047) 19

# Introduction

The main idea of doing this coursework is to create a Conceptual ER diagram individually. FOODTOOYOU is a company where delivers grocery items ordered by the registered members of the company from registered retailing stores. It will be delivered on the same day or on the following day. The idea was initiated during the COVID-19 lockdown that was put in UK from March to June 2020 when access to the groceries become challenging.

Part A first question is to create a database architecture to undertake a database project to support the needs of the company. It needs to include all the entities, relationships, multiplicities, attributes and primary keys that has been identified

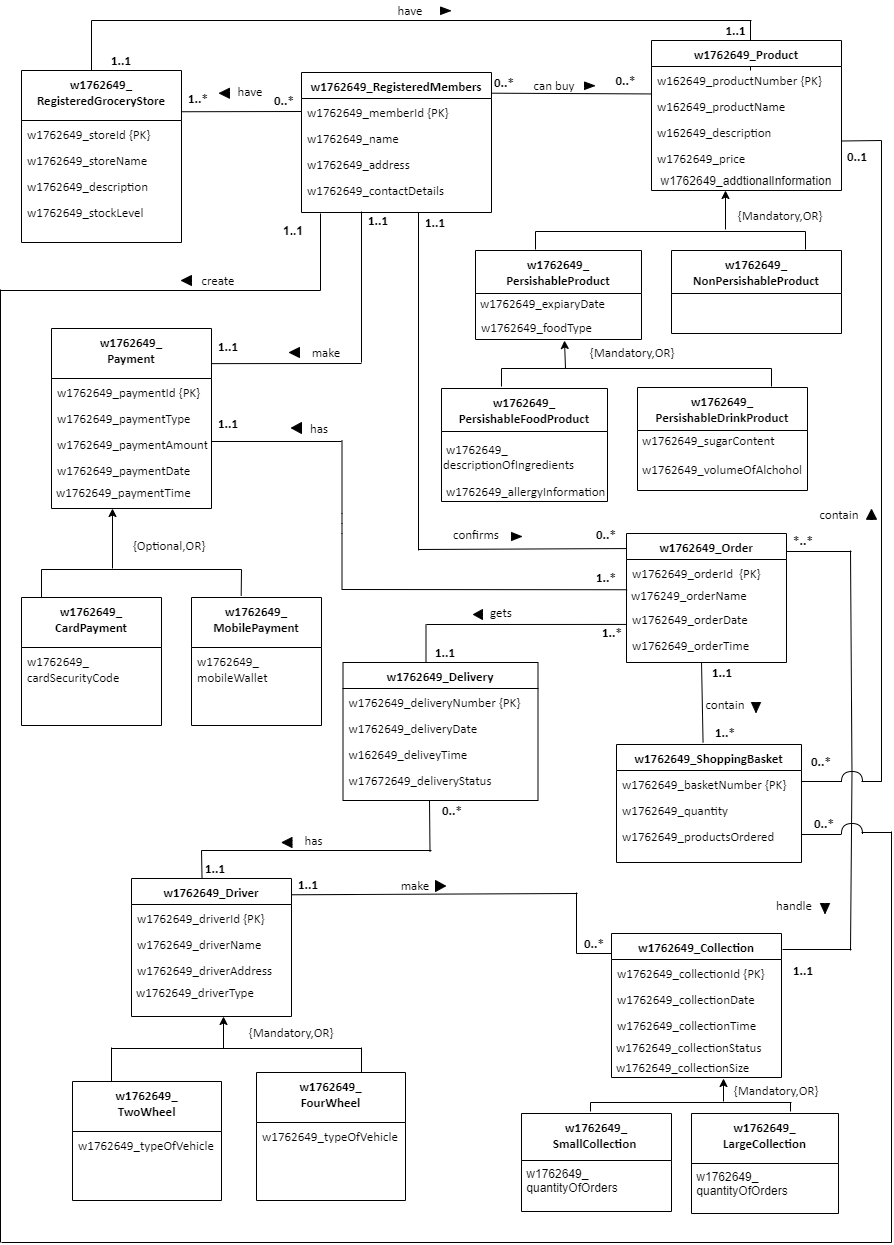
Part A second question is to create a data dictionaryto document how the entities have been identified for FOODTOOYOU.

Part A third question is to create a data dictionary to document how the relationshipsand multiplicities are identified for FOODTOOYOU.

Part A last question is to create a data dictionary to document how the attributesand primary keysfor each entity have been identified for FOODTOOYOU.

# Produce a complete CONCEPTUAL

# ERD for FOODTOOYOU.



# Create a data dictionary to document how you identified the entities for FOODTOOYOU.

|  |  |
| --- | --- |
| *Entity name* | *Description* |
| w1762649\_RegisteredGroceryStore | General term describing all the grocery stores in the FOODTOOYOU where the registered members buy their products. |
| w1762649\_RegisteredMembers | General term describing all the registered members in the FOODTOOYOU who buy products from the grocery stores, who confirm orders and make payments. |
| w1762649\_Product | General term describing all the products including perishable products and non-perishable products. Perishable products contain two types as perishable food products and perishable drink products in the grocery stores of FOODOOYOU where the registered members can access and put to the shopping baskets. |
| w1762649\_Payment | General term describing the payments including card payments and mobile payments done by the registered members of FOODTOOYOU for their orders. |
| w1762649\_Order | General term describing all the orders ordered by the registered members of FOODTOOYOU which makes a collection. |
| w1762649\_ShoppingBasket | General term describing all the shopping baskets which are contained in an order owned by the registered members of FOODTOOYOU. |
| w1762649\_Collection | General term describing all the collections including small collections and large collections which are formed by all the orders altogether which are done by the drivers of FOODTOOYOU and where the products ordered by the registered members contain. |
| w1762649\_Delivery | General term describing all the deliveries of the orders, done by the drivers of FOODTOOYOU. |
| w1762649\_Driver | General term describing all the drivers in FOODTOOYOU who delivers orders to the houses of the registered members and who do the collections in FOODTOOYOU. Drivers include two-wheel drivers and four-wheel drivers. |

|  |  |  |
| --- | --- | --- |
| *General entity* | *Specialized entity* | *Explanation* |
| w1762649\_Product | w1762649\_PerishableProduct | Product can be divided into two as perishable and non-perishable. Products must be either perishable or Non-Perishable products.  In the coursework specification it has mentioned that most products are perishable products.  Other than perishable products there are non-perishable products where the coursework specification mentioned that not all products are perishable. |
| w1762649\_NonPerishableProduct |
| w1762649\_PerishableProduct | w1762649\_PerishableFoodProduct | Perishable Products must be either divided as food products or drink products.  Food products and Drink products are identified as perishable products because they have attributes that a perishable product should have. |
| w1762649\_PerishableDrinkProduct |
| w1762649\_Payment | w1762649\_CardPayment | Payments can be either done by card or by mobile. Those are the main types of payments that FOODTOOYOU accept. Other than the main types of payments there can be payments through cash also. |
| w1762649\_MobilePayment |
| w1762649\_Driver | w1762649\_FourWheel | Drivers must use either four wheel or two wheel vehicles for their delivery services in FOODTOOYOU where there are no other vehicle types in FOODTOOYOU. |
| w1762649\_TwoWheel |
| w1762649\_Collection | w1762649\_SmallCollection | Collections are varying size as small collection and large collection in FOODTOOYOU delivery service. Collection must be divided as either small or large. By dividing the collection as large and small it is easy to track and manage the collection process. |
| w1762649\_LargerCollection |

# Create a data dictionary to document how you identified the relationships and multiplicities for FOODTOOYOU.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Entity name* | *Multiplicity* | *Relationship* | *Multiplicity* | *Entity name* | *Justifications for the multiplicity*  *(4 statements for each relationship)* |
| w1762649\_  RegisteredGroceryStore | 1..\* | have | 0..\* | w1762649\_  RegisteredMembers | One registered grocery store may not have any registered member. |
| One registered grocery store may have maximum of many registered members. |
| One registered member may have minimum of one registered grocery store. |
| One registered member may have maximum of many registered grocery stores. |
| w1762649\_  RegisteredGroceryStore | 1..1 | has | 1..1 | w1762649\_Poduct | One registered grocery store has at least of one unique product. |
| One registered grocery store has a maximum of one unique product. |
| One Product may have a minimum of one registered grocery store. |
| One product may have maximum of one registered grocery store. |
| w1762649\_  RegisteredMembers | 0..\* | can buy | 0..\* | w1762649\_Product | One registered member may not buy any product. |
| One registered member can buy maximum of many number of products. |
| One product may not be bought by any registered member. |
| One product can be bought by maximum of many registered members. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Entity name* | *Multiplicity* | *Relationship* | *Multiplicity* | *Entity name* | *Justifications for the multiplicity*  *(4 statements for each relationship)* |
| w1762649\_  RegisteredMembers | 1..1 | create | 0..\* | w1762649\_  ShoppingBasket | One registered member may not create any shopping basket. |
| One registered member create maximum of many shopping baskets.(Members have to create separate shopping baskets for each grocery store) |
| One shopping basket is created by minimum of one registered member. |
| One shopping basket is created by maximum of one registered member. |
| w1762649\_  ShoppingBasket | 0..\* | contain | 0..1 | w1762649\_  Product | One shopping basket may not contain any product. |
| One shopping basket contain maximum of one product (As products are unique to each store and registered members have to create separate shopping baskets to every store.) |
| One Product may not be contained in any shopping basket. (Because members may not put any product into a shopping basket.) |
| One product is contained in maximum of many shopping baskets. |
| w1762649\_  RegisteredMember | 1..1 | confirms | 0..\* | w1762649\_Order | One registered member may not confirm any order. |
| One registered member may confirm maximum of many orders. |
| One order is confirmed by at least one registered member. |
| One order is confirmed by maximum of one registered member. |
| *Entity name* | *Multiplicity* | *Relationship* | *Multiplicity* | *Entity name* | *Justifications for the multiplicity*  *(4 statements for each relationship)* |
| w1762649\_Order | 1..\* | has | 1..1 | w1762649\_  Payment | One order has a minimum of one payment. |
| One order has maximum of one payment. |
| One payment is had by a minimum of one order. |
| One payment is had by maximum of many orders. |
| w1762649\_  Order | 1..1 | contain | 1..\* | w1762649\_  ShoppingBasket | One order contains a minimum of one shopping basket. |
| One Order contains a maximum of many shopping baskets. |
| One shopping basket is contained in at least of one order. |
| One shopping basket is contained in maximum of one order. |
| w1762649\_Order | 1..\* | gets | 1..1 | w1762649\_  Delivery | One order gets minimum of one delivery. |
| One order gets maximum of one delivery. |
| One delivery has a minimum of one order. |
| One delivery has a maximum of many orders. |
| w1762649\_  Collection | 1..1 | handle | \*..\* | w1762649\_Order | One collection is handled by minimum of many orders. |
| One collection is handled by maximum of many orders. |
| One order handles minimum of one collection. |
| One order handles maximum of one collection. |
| w1762649\_Delivery | 0..\* | has | 1..1 | w1762649\_Driver | One delivery has a minimum of one driver. |
| One delivery has a maximum of one driver. |
| One driver may not have any deliveries. |
| One driver may have maximum of many deliveries. |
| *Entity name* | *Multiplicity* | *Relationship* | *Multiplicity* | *Entity name* | *Justifications for the multiplicity*  *(4 statements for each relationship)* |
| w1762649\_Driver | 1..1 | make | 0..\* | w1762649\_-Collection | One driver may not make any collections. |
| One driver make maximum of many collections. |
| One Collection is made by at least one driver. |
| One collection is made by maximum of one driver. |

# Create a data dictionary to document how you identified the attributes and primary keys for each entity for FOODTOOYOU.

|  |  |  |
| --- | --- | --- |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_  RegisteredGroceryStore | w1762649\_storeId {PK} | **storeID** is recognized as a primary key because each store can be uniquely identified.  **storeName** is used to store the name of each Store, **decription** stores the information of each store while the **stockLevel** stores the level of stock to maintain the stocks easily. |
| w1762649\_storeName |
| w1762649\_description |
| w1762649\_stockLevel |
| w1762649\_  RegisteredMembers | w1762649\_memberId {PK} | **memberID** is unique to each registered member because it is necessary when the member visits to buy products.  **name** attribute is used to store the name of each member, **address** stores the address of each member which helps when delivering the orders and the **contactDetails** store the telephone numbers etc. where the member can be contacted. |
| w1762649\_name |
| w1762649\_address |
| w1762649\_contactDetails |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_Product | w1762649\_productNumber {PK} | **productNumber** is use to identify each product uniquely when billing and buying products.(Helpful in member side and from the shop side.)  **productName** is used to save the name of each product, **description** attribute is used to store the product information, **price** attribute is used store the prices of each product.  **aditionalInformation** attribute is used to give some additional information such as discounts etc. |
| w1762649\_productName |
| w1762649\_description |
| w1762649\_price |
| W1762649\_additionalInformation |
| w1762649\_PerishableProduct | w1762649\_expiaryDate | **(expiaryDate** attribute stores the date of expiry of perishable products.  **foodType** attribute stores the type of perishable food. Example: Biscuit, Soft drinks) Useful for the members who buy the product. |
| w1762649\_foodType |
| w1762649\_PerishableFoodProduct | w1762649\_descriptionOfIngredients | **(descriptionOfIngredients** stores the information of the ingredients which made up that product.  **allergyInformation** attribute saves about the information, that if there is an allergy containing ingredients in the particular product) Useful for the members who buy the product. |
| W1762649\_allergyInformation |
| w1762649\_PerishableDrinkProduct | w1762649\_sugarContent | **sugarContent** stores the amount of sugar included in that particular drink product.  **volumeOfAlchohol** saves the information of the alcoholic percentage of that particular drink product. |
| w1762649\_volumeOfAlchohol |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_Payment | w1762649\_paymentId {PK} | **paymentID** is uniquely identified and it is useful when a member pay their amount and when in any case of trouble regarding the payment.  **paymentType** is used to store the type of payment namely card payment, mobile payment.  **paymentAmount** is used to store the amount paid by the member, **paymentDate** and **paymentTime** is used store date and time of the payment is done and to make convenient o the member. |
| w1762649\_paymentType |
| w1762649\_paymentAmount |
| w1762649\_paymentDate |
| w1762649\_paymentTime |
| w1762649\_CardPayment | w1762649\_cardSecurityCode | **cardSecurityCode** stores the security code of a card either credit or debit card. |
| w176264\_MobilePayment | w176264\_mobileWallet | **mobileWallet** stores the information about the payments which are done through the mobile device. |
| w1762649\_Order | w1762649\_orderId {PK} | **orderID** is use to identify each order uniquely and orderID is very useful when delivering the products and when doing collections.  **orderName** saves the name of each order, **orderDate** and **orderTime** stores the date and time of the order, which is confirmed and to make convenient to the member. |
| w1762649\_orderName |
| w1762649\_orderDate |
| w1762649\_orderTime |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_ShoppingBasket | w1762649\_basketNumber {PK} | **basketNumber** is uniquely identified because it is very useful when billing and when putting products to the basket by the member.  **quantity** stores the amount of products included in a basket and **productsOrdered** attribute store the names of each products in the shopping basket. |
| w1762649\_quantity |
| w1762649\_productsOrdered |
| w1762649\_Collection | w1762649\_collectionId {PK} | **collectionID** is identified as a primary key because each collection can be uniquely identified and it is very useful when doing collection by the drivers.  **CollectionDate** and **collectionTime** is used make an accuracy when doing collection(date and time that collection is done) and **collectionStatus** stores the status of the collection as pending, confirmed, in process, delivered, etc.  **collectionSize** saves the information of the size of the collection whether it is small or large. |
| w1762649\_collectionDate |
| w1762649\_collectionTime |
| w1762649\_collectionStatus |
| w1762649\_collectionSize |
| w1762649\_SmallCollection | w1762649\_quantityOfOrders | As mentioned in the coursework specification collection is divided into small and large. To identify a collection as a small collection there should be an amount that shows the quantity of orders should be contained in small collection should end.(Ex: Small collection –> 1-10 orders)  **quantityOfOrders** stores the quantity of orders that contained in small collection. |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_LargeCollection | w1762649\_quantityOfOrders | As mentioned in the coursework specification collection is divided into small and large. To identify a collection as a large collection there should be an amount that shows the quantity of orders should be contained in large collection. .(Ex: Large collection –> upward 10)  **quantityOfOrders** stores the quantity of orders that contained in large collection. |
| w1762649\_Delivery | w1762649\_deliveryNumber {PK} | **deliveryNumber** is uniquely identified because it is useful for the member who request a delivery as well as for the company side.  **deliveryDate** and **deliveryTime** stores the date and time of the delivery and **deliveryStatus** stores the status of the delivery as pending, confirmed, in process or delivered. |
| w1762649\_deliveryDate |
| w1762649\_deliveryTime |
| w1762649\_deliveryStatus |
| w1762649\_Driver | w1762649\_driverId {PK} | **driverID** is identified as a primary key because driver can be uniquely identified.  **driverName** stores the name of the drivers of FOODTOOYOU, **driverAddress** stores the address of each driver while **driverType** saves the type of the driver whether the driver is four wheel driver or two wheel driver. |
| w1762649\_driverName |
| w1762649\_driverAddress |
| w1762649\_driverType |
| *Entity name* | *Attributes for this entity (include PK)* | *Justification* |
| w1762649\_TwoWheel | w1762649\_typeOfVehicle | **typeOfVehicle** stores the information of the vehicle type whether it is a bike or moped when considering two wheel drivers. |
| w1762649\_FourWheel | w1762649\_typeOfVehicle | **typeOfVehicle** stores the information of the vehicle type whether it is a car or van when considering four wheel drivers. |

# Conclusion

As this is an individual coursework, it helps to clearly learn about the conceptual ER diagram in a further manner. Using sub classes and super classes helps to further elaborate the scenario given. Usefulness and how to apply data dictionaries in a workplace is learnt from this coursework. Data dictionaries help to clearly clarify about the entities, attributes, primary keys, subclasses and super classes and why those were identified as the entities, attributes, primary keys, subclasses and super classes.

Main areas learnt from this coursework,

* To produce a complete **CONCEPTUAL ERD** for FOODTOOYOU.
* To create a **data dictionary** to document how we identified the **entities** for FOODTOOYOU.
* To create a **data dictionary** to document how we identified the **relationships** and **multiplicities** for FOODTOOYOU.
* To create a **data dictionary** to document how we identified the **attributes** and **primary keys** for each entity for FOODTOOYOU.

# References

* W3schools.com. 2020. *Java Inheritance (Subclass And Superclass)*. [online] Available at: <https://www.w3schools.com/java/java\_inheritance.asp> [Accessed 14 October 2020].
* Tutorialspoint.com. 2020. *What Is Data Dictionary*. [online] Available at: <https://www.tutorialspoint.com/What-is-Data-Dictionary> [Accessed 14 October 2020].
* Dl.ebooksworld.ir. 2020. [online] Available at: <https://dl.ebooksworld.ir/motoman/Pearson.Database.Systems.A.Practical.Approach.to.Design.Implementation.and.Management.6th.Global.Edition.www.EBooksWorld.ir.pdf> [Accessed 14 October 2020].