Assignment - 2

Project Group: 15

Group Members: Aditi Jain, Prerana Urs Praveen Kumar Urs, Rachit Singh, Saurav Dasgupta, Suhani Shah, Vikrant Vaze

Database Specification: Purpose, Business Problems addressed, Business Rules & Design Decisions

Topic: Online order processing and delivery service

Database Purpose:

The purpose of the database model is to generate, process and maintain data to support an online order processing and delivery service. The database enables the organization to facilitate online deliveries for food, grocery, fresh produce, medical supplies and other general commodities from local restaurants and stores to the respective customers. The database supports the organization to receive an online order, fulfill it and assign it to the appropriate executive, ensuring an efficient and timely delivery.

Business Problems addressed:

The database model addresses the following business problems:

- The database model maintains the information about the registered stores and restaurants along with different products that are ready for delivery. It maintains inventory related details storing information about the product availability which thereby facilitates displaying the customer with the final catalog for all the available products.
- Enables the management and leadership teams to create descriptive reports in order to build marketing strategies and employee goals within the organisation.
- Permits the order status to be tracked in addition to the availability of delivery executives which further facilitates the orders assignments to the executives thereby ensuring a timely delivery.
- Provides customers with the information regarding the active promotions and offers

based on the membership type and products during the purchase.
- Allows customers to provide a detailed feedback based on the overall delivery experience.
- Provides the customers with information regarding customer care services to manage all possible queries and concerns raised by the customers.

Business Rules :

- Each Customer has a single delivery address and a single phone number.
- Each Customer must Login with unique login credentials to view the Catalog and place the order.
- Every Customer must mandatorily save their Card details during sign up.
- Each Customer is linked to a single active Cart.
- Each Customer is allowed to place a single order at a time from one Catalog that contains a single Store/restaurant.
- Each Customer may provide one Feedback for each Order.
- Each employee can either be a DeliveryExecutive or a back-end staff in the organisation
- Each DeliveryExecutive is assigned to zero or more orders to complete at a time. Each Order is assigned to one deliveryExecutive.

Design Decisions :

Entity Name	Entity Role	Relationship with other Entities
Customer	The database aims at serving customers with the timely delivery of the ordered services. This Entity maintains basic information of all the customers using the service/tool, it has attributes to store information regarding the customer's name, contact details, DOB etc. It also maintains data about the customers address/location and stores information regarding the type of	This acts as one of the core entities with the primary key as **CustomerID.** The relationship with corresponding entities are as follows :. ● **Cart** : Each customer must have a cart assigned in order to store the selected products from the catalog, and hence a one to one relationship. ● **Login** : Every customer has to login in order to access the catalog and proceed further. Every customer has a one-to-one relationship with the Login entity; one Customer must have one userID. ● **Membership** : A customer can either have a single membership

	membership owned by the customer, if any.	or none, this could be used during different promotions and offers. ● **Location** : Every customer must update their delivery location every time they login. ● **Card** : Every customer should mandatorily save their card details for delivery transactions. A customer can have one or more card details added.
Catalog	Catalog entity is used to display the online menu/list of products provided by the nearby stores/restaurants based on their availability.	The entity has a primary key ‘**CatalogID**’ and finds relationship with other entities as follows: ● **Product** : A Catalog consists of all the products that are available in each store/restaurant for delivery. It shares a many-to-many relationship with the products, using ‘ProductID’ as FK in the Catalog entity.. ● **Store** : A list of stores or restaurants available for delivery are maintained in the catalog. It also maintains an unique relationship with the ‘CatalogID’ & ‘StoreID’ respectively. A catalog is mapped to a single store. ● **Inventory** : A catalog allows the inventory to have a constant check on availability of the products, using ‘ItemID’ as FK in Catalog entity. A product in catalog is mapped to one or more items from inventory. ● **Promotions** : Different products/memberships may either have many or no active promotions and offers acting on them which are automatically visible for the customer on the Catalog.
Employee	The Employees entity stores the details of all the employees (Admin,	**EmployeeID** acts as the primary key in this entity and has the below mentioned relationships with other entities -

	Manager, Customer Care Representative, Delivery Executive, etc) performing various actions either on the user interface or the back-end process of the organisation to maintain ease in allocating executives for the opted delivery service.	● **DeliveryExecutive** : 'One to many' relationship exists with Delivery Executive Entity wherein, every employee as an admin assigns delivery executives for each order placed by the customer, 'EmployeeID'acts as PK,FK. ● **CustomerCareOffice** : None or many employees can be playing the role of a customer care representative based on the customer care services. ● **Login** : the login details of all the employees except for delivery executives in the organisation are stored in the Login entity.
Cart	The Cart is used to maintain and display the list of all the products that are selected by the customers from the catalog. It shows the selected product along with the desired quantity, price and other key details.	The primary key for this entity is **CartID** which is autogenerated. This entity shares relationship with the below mentioned entities : ● **OrderHeader** :As soon as the customer places the order,the Cart entity provides an unique CartID in addition to customer details to the OrderHeader entity to create an order record. ● **OrderItems** : Cart entity provides a cartID and the list of corresponding products that are included in the corresponding order. A cart contains one or more products for entry in orderItems under one orderID. ● **Inventory** : Cart checks for the availability of the selected product when the product is added to the cart. ● **Customer** : A cart is linked to a customer with a one-to-one relationship. One cart is created by default for each customer for them to add the products. ● **Catalog** : A customer can have a single order from a single store/restaurant at once. Hence,

		the Cart entity can have one or more products from a catalog.
Membership	Membership entity maintains different types of memberships that are offered by the organisation. It includes information about the membership related offers that can be redeemed by the customer while placing an order. It also stores information regarding the membership validity.	The primary key maintained in this entity as **MembershipID** is auto-generated. The relationships with other entities are as follows : ● **Promotions** : This entity is directly linked to the Promotions entity through an associative entity in order to avoid a many-to-many relationship. Every membership can have multiple promotions and each promotion can be valid for more than one membership.. ● **Customer** : Unique membershipID is generated for each customer, thus having an optional one to many relationship between Membership and Customer entities respectively.
MembershipPromo	This entity acts as an <u>associative entity</u> for handling many-to-many relationship between 'Membership' and 'Promotions' entities.	This associative entity allows each membership to have multiple promotions and each promotion can be valid for more than one membership.
Product	This entity holds complete details of the product that are available to the Customer on the catalog. Product details include Product ID, name, description, weight, unit of measure,etc.	**ProductID** being the primary key for this entity, it finds relationship with the below mentioned entity : ● **Catalog** : Product participates in a many-to-many relationship with Catalog which is denoted using an associative entity ProductCatalog through which the product and catalog mapping is done.
ProductCatalog	This entity acts as an <u>associative entity</u> maintaining the unique	It comprises ProductID and **CatalogID** acting a**s** composite PK,FK keys. The

	value of **Products - Catalog** combination.	relationship type shared with Product and Catalog entities is 'many-to-many'.
Inventory	This entity stores the list of all the products and their respective stock information corresponding to different stores and restaurants that are available in the organization's network.	Every item stored in the Inventory has a unique **ItemID** and the entity holds the below relationships - ● **Cart**: Available quantity of item is validated from Inventory entity when a product is added to the cart by the Customer. Inventory is linked to the cart through a many-to-one relationship as multiple products can be added to a single cart. ● **Store :** Inventory is linked to the Store to fetch any information regarding the product's availability. ● **Catalog**: Inventory updates on the stock of products from a store in catalog.
OrderHeader	This entity presents a cumulative list of active orders, comprising all the header level details of the sales order such as Total amount, Order Status, Date of Order, Date of order completion, membership and store details.	Each time a customer submits an order, an **OrderID** is system generated as the primary key in the OrderHeader which holds the following relationships - ● **Cart** : OrderHeader is linked to the Cart entity through a one-to-one relationship. An unique order id will be generated for every cart id and the header level details will be fetched corresponding to the cart id. ● **OrderItems** : OrderHeader has a one-to-many relationship with the OrderItems entity as one sales order can have multiple items ordered in it. ● **OrderAssignment :** OrderHeader is linked to the OrderAssignment entity for assignment of the delivery executive to all the orders pending for delivery. ● **Invoice** : OrderHeader will be directly linked to Invoice through a one-to-one relationship. An

		unique invoice number will be generated for every order id and will be initiated for the customer in the form of an invoice receipt. ● **Feedback :** OrderHeader will be linked to the Feedback entity through an optional one-to-one relationship. This will store the customer feedback provided for the corresponding order.
OrderItems	This entity stores the list of products ordered in the corresponding order id.	**OrderItems** holds the below relationships - ● **OrderHeader**: OrderItems will fetch the order id and the cart id through a many-to-one relationship as multiple products from the same store/restaurant can be ordered in a single order. ● **Cart:** Based on the order id and cart id received from OrderHeader, OrderItems will fetch the product details from the Cart. Every OrderID is linked to only one CartID.
OrderAssignment	This entity is used to store the mapping of orders to delivery executives.	This will be linked to OrderHeader and DeliveryExecutive entities to map the delivery executive with the orderID. One DeliveryExecutive can be assigned to multiple orders while a single order is assigned to a single DeliveryExecutive.
Payment	Payment entity retrieves the payment related details from the Invoice entity will have the details related to payment like the payment status, date, unique payment ID, thereby keeping a record of the payment made by the customer based on the invoice that has been generated.	**PaymentID** is the primary key which finds relationship with the following entities : ● **Invoice** : Payment retrieves the invoice details from the Invoice entity on the basis of 'InvoiceID'in a one-to-one relationship. Every order payment initiates an invoice. ● **Card** : Payment retrieves the details of the card from the Card entity using 'CardID' for payment transaction. Every saved

		card of the customer is not always used for payment unless the order is finalised and further processed to the OrderHeader from the Cart.
Invoice	The invoice entity will display the relevant invoice details, comprising the recent orders placed by the customer, it also includes the additional tax charges, promotional offers and discounts, if applicable. The Invoicedate lets us track the details about when the invoice was generated.	**InvoiceID** being the primary key of this entity, would have following mentioned relationship with other entities : ● **OrderHeader** : An invoice is generated based on the order details provided by the above mentioned entities using 'OrderID' as the FK. There is a one-to-one relationship. ● **Payment** : Invoice has one-to-one mapping with the Payment entity.
Card	The Card entity is used to store the details of the card/s that have been stored for making payments by the customer.	It has **CardID** as the primary key. It relates to the Customer entity as follows: ● **Customer** : stores'CustomerID' as PK,FK thereby ensuring a unique connection between the card/s and the respective customer in a many-to-one relationship respectively. ● **Payment** : One card is used to make one payment.
Store	This entity holds information of different products that are supplied by each store/restaurant based on their availability and location.	**StoreId** is defined as the primary key for this entity, which has relationship with the following mentioned entities : ● **Location** : Store's location is stored in the Location entity. Everystore has a single location. ● **Inventory:** Store maintains inventory of products. ● **Catalog**: A store has a catalog to display the products to sell.

CustomerCareOffice	This entity stores the data related to all the employees working as customer care representatives in the organisation.	**CCOfficeID** is the primary key used to connect with the **Employee** entity to store the employee's office details.
Feedback	The Feedback Entity allows the customers to provide feedback about their delivery experience. There exists only one feedback for each Order.	**FeedbackID** acts as the Primary key along with CustomerID, OrderID as the foreign keys. Relationship with other entity is as follows: ● **OrderHeader** : Feedback can be given by the customer corresponding to a completed order. There can be one feedback for every order.
DeliveryExecutive	This entity stores the data of the employees working as delivery executives in the organisation. It allows the model to facilitate efficient delivery of orders based on their availability also taking into consideration the current location of particular delivery executives.	**EmployeeID** acts as both Primary and Foreign key. It finds a relationship with the OrderAssignment Entity to determine the current available employees for delivery. Relationship with other entity is as follows: ● **Location**: Retrieves current location of DeliveryExecutive. Based on the current location of the delivery executive the order delivery duty is assigned. ● **OrderAssignment :** DeliveryExecutive is assigned one or more orders to complete. ● **Employee** : DeliveryExceutive is an employee.
Location	The entity consists of master values defining and maintaining the locations of Customer, Store, and Restaurants. Each Customer, Store or Restaurant has it's unique location.	**LocationID** being the Primary key. It stores the location of deliveryExecutive, Customer and store. Store has a location. Customer has a location. DeliveryExecutive has a location.
Login	This entity stores the basic login related information of the	It uses **LoginID** as the primary key to maintain unique login entries of both customers and employees.

	customers and the employees. For each user, being a Customer or an Employee, there is only one login associated uniquely.	**Customer, Employees :** Different customer and employee id's can be verified to each of their categories by linking CustomerID being the primary key in Customer entity and EmployeeID in the Employee entity and storing it in the BusinessEntityId attribute in the Login entity.
Promotions	Promotions entity stores the details of all the active promotions and offers that could be used by the customer on different products available on the catalog.	**PromotionID** is the primary key for this entity and it shares the below relationships with other entities - ● **Catalog** : Any active promotions or offers that could act on the available products or stores is reflected on the catalog directly. Each catalog may either have none or many promotions. ● **Membership** : Promotions entity is entitled with Membership entity using the MembershipPromo as an associative entity in a many-to-many relationship.