Contents

[1st Normalization 2](#_Toc176292142)

[2nd Normalization form 5](#_Toc176292143)

[3rd Normalization form 8](#_Toc176292144)

[ERD 13](#_Toc176292145)

# 1st Normalization

Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. The first normalization form (1NF) requires that all tables in a database meet the following criteria:

1. Every table needs to have a primary key which will be a single value for every record on the table.
2. Each one of them should be atomic that is, every column should be characterized by only one attribute.
3. All the rows in a column have to be of the same or similar type and in the same way all the values in a column have to be of the same type.

Based on the AWS migration project for Ozmart Retail Group, here is a list of potential database tables normalized to the first normal form (1NF):

**1. Customers Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Raised as a primary key, it is a unique key used for identifying each customer. |
| first\_name | VARCHAR | Customer's first name |
| last\_name | VARCHAR | Customer's last name |
| email | VARCHAR | Customer's email address |
| phone\_number | VARCHAR | Customer's phone number |
| address\_line1 | VARCHAR | First line of customer’s address |
| address\_line2 | VARCHAR | If they have any second line to the customer’s address they are also entered. |
| city | VARCHAR | Customer's city |
| state | VARCHAR | Customer's state |
| postal\_code | VARCHAR | Customer's postal code |
| country | VARCHAR | Customer's country |

**2. Products Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Product ID, the identification number of every item in the list. |
| product\_name | VARCHAR | Name of the product |
| product\_description | TEXT | Description of the product |
| price | DECIMAL | Price of the product |
| stock\_quantity | INT | Amount of the product in stock |
| category\_id | INT | Foreign key, references product category |

**3. Orders Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Primary key, that is the number unique for each order. |
| order\_date | DATETIME | Order ID and date and time when the order was placed |
| customer\_id | INT | Foreign key, refers to customers table |
| total\_amount | DECIMAL | Tab Total amount of the order |
| status | VARCHAR | Order status or the current position of the order on the flow for instance pending, shipped or delivered. |

**4. Order\_Items Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_item\_id | INT | Primary key, many products can be put in one order hence an order item has a unique order identifier. |
| order\_id | INT | Foreign key, references orders table This is because the option ‘Generate foreign key, references orders table’ is used and, as it has been explained before, this type of foreign key can reference any table in the database. |
| product\_id | INT | Foreign key, refers to products table |
| quantity | INT | Number of the product that has been purchased |
| price | DECIMAL | The price of the product that was ordered by the customer at the time he ordered. |

**5. Categories Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| category\_id | INT | Foreign key, code of the category of the product that the coupon applies to. |
| category\_name | VARCHAR | Name of the category |

**6. Payments Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| payment\_id | INT | Primary key, It will act as an identifier that will uniquely identify each payment that is made. |
| order\_id | INT | Foreign key, references orders table |
| payment\_date | DATETIME | The time and date for the payment was made |
| payment\_method | VARCHAR | Type of payment accepted, for example, credit card, pay-pal. |
| payment\_amount | DECIMAL | Amount of the payment |

**7. Employees Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | The primary key which allows tracking unique identity of each employee |
| first\_name | VARCHAR | Employee's first name |
| last\_name | VARCHAR | Employee's last name |
| email | VARCHAR | Employee's email address |
| phone\_number | VARCHAR | Employee's phone number |
| hire\_date | DATE | Date the employee was hired |
| job\_title | VARCHAR | Information as to the rank of the employee |
| department\_id | INT | It is a foreign key which refers the departments table. |

**8. Departments Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| department\_id | INT | Primary key, it refers to unique identification code of the department. |
| department\_name | VARCHAR | Name of the department |

**9. Inventory Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| inventory\_id | INT | Primary key, A specific identification number for each record being kept in an inventory. |
| product\_id | INT | Foreign key / references products table |
| warehouse\_id | INT | Warehouse\_id is foreign key which has a reference to the warehouses table. |
| stock\_quantity | INT | The amount of product in store |

**10. Warehouses Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| warehouse\_id | INT | Prim key, Each record in this table has one id that is unique to each warehouse. |
| warehouse\_name | VARCHAR | Name of the warehouse |
| location | VARCHAR | Location of the warehouse |

It is very important to have these tables normalized up to the 1NF, which means that each field should store atomic values, and each record is uniquely specified. It will help in the development of a scalable, secure and strong database environment of Ozmart Retail Group for AWS cloud migration strategy.

# 2nd Normalization form

To make the database tables to be in second normal form, it is mandatory that they are in the first normal form and all the non key fields are fully functionally dependent on the key. This means that there must be elimination of partial dependencies, which refer to a situation whereby one or more non-key attributes are dependent on a portion of the composite primary key.

Given the database schema provided earlier, here’s how we can modify it to achieve 2NF:

**1. Customers Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Pkey, unique Numeric value for each entry of the customers table |
| first\_name | VARCHAR | Customer's first name |
| last\_name | VARCHAR | Customer's last name |
| email | VARCHAR | Customer's email address |
| phone\_number | VARCHAR | Customer's phone number |
| address\_line1 | VARCHAR | Customer’s first line of physical residence address |
| address\_line2 | VARCHAR | Second line of the customer’s addresses if any |
| city | VARCHAR | Customer's city |
| state | VARCHAR | Customer's state |
| postal\_code | VARCHAR | Customer's postal code |
| country | VARCHAR | Customer's country |

*There are no modifications required as all the attributes apart from the primary key, that is, customer\_id is totally contingent on the primary key.*

**2. Products Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Primary key, unique id, it is used to uniquely identify each product. |
| product\_name | VARCHAR | Name of the product |
| product\_description | TEXT | Description of the product |
| price | DECIMAL | Price of the product |
| stock\_quantity | INT | Amount of product in quantity on the stock |
| category\_id | INT | Category, foreign key, references Categories table |

*None are possible because all non key attributes are completely determined by the primary key Product\_id.*

**3. Orders Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Primary key; Refers to number which is unique to each order placed. |
| order\_date | DATETIME | A few of the order parameters that are required includes; |
| customer\_id | INT | Foreign key, refers to Customers table |
| total\_amount | DECIMAL | Order: Total Amount |
| status | VARCHAR | Shipment details: Order status (for instance pending, shipped or delivered) |

*None as all the extra columns are complete dependent of the primary key order\_id.*

**4. Order\_Items Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_item\_id | INT | Primary key, identification number of the order item |
| order\_id | INT | Orders refers to Foreign key, which belong to the Orders table. |
| product\_id | INT | Foreign key, refers Products table |
| quantity | INT | Amount of the product ordered |
| price | DECIMAL | Ordering price of the product that has been set for that particular product. |

*No modifications are required as all the remaining attributes are totally contingent on order\_item\_id.*

**5. Categories Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| category\_id | INT | Primary key, every category is given an identification number. |
| category\_name | VARCHAR | Name of the category |

*No changes are needed in this case all the categories’ attributes are shown to be fully dependent on the primary key which is category\_id.*

**6. Payments Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| payment\_id | INT | Primary key is an identifier an individual payment made and thus can only be unique. |
| order\_id | INT | Foreign key ref – Orders table |
| payment\_date | DATETIME | On this option, candidates have to indicate the date and time the payment was made |
| payment\_method | VARCHAR | Type of acceptance (e. g. , credit card, PayPal) |
| payment\_amount | DECIMAL | Amount of the payment |

*There are no changes to be made because all the non-key attributes totally rely on the primary key payment\_id.*

**7. Employees Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | PrimaryKey, Each employee record has to be unique with a unique identification no. |
| first\_name | VARCHAR | Employee's first name |
| last\_name | VARCHAR | Employee's last name |
| email | VARCHAR | Employee's email address |
| phone\_number | VARCHAR | Employee's phone number |
| hire\_date | DATE | Every employee, date the employee was hired |
| job\_title | VARCHAR | Employees present in the form of the employee holder of the particular job title |
| department\_id | INT | Foreign key, refer to the Department table |

*There is no necessity for change because all attributes which are not key are fully depend on the primary key employee\_id.*

**8. Departments Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| department\_id | INT | Primary key, this is an unique ID for every department |
| department\_name | VARCHAR | Name of the department |

*The foreign key is department\_id which is the primary key of this table hence no modification is require as all of the non key attributes are totally dependent on the key field department\_id.*

**9. Inventory Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| inventory\_id | INT | Primary key, that is, unique identification number for every record contained in the inventory table |
| product\_id | INT | Foreign key, refers to Products table |
| warehouse\_id | INT | Foreign key of this column refers to Warehouses table. |
| stock\_quantity | INT | Inventory of stock or inventory of available product |

*The relationships should not be altered as every other attributes in the table are completely dependent on the primary key, inventory\_id.*

**10. Warehouses Table (Already in 2NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| warehouse\_id | INT | Identifying fields, unique number assigned to the warehouse to differentiate it from all other warehouses within the locality. |
| warehouse\_name | VARCHAR | Name of the warehouse |
| location | VARCHAR | Location of the warehouse |

*None as all the attributes in the current model are non key and are fully dependent on the primary key warehouse\_id.*

**Summary of Changes for 2NF**

For this particular schema, the database table did conform to 1NF and 2NF as none of the tables had partial dependency meaning that all non-key attributes were fully functionally dependent on the whole of the primary key. If there where tables with more than one field in the PK and where there were some other fields with a partial dependency on that PK, then there would be a need to further subdivide the tables. Nevertheless, in this case, each of these tables is already in the 2NF compliance.

# 3rd Normalization form

Now to bring the structures of the database tables from the present state to the third normal form (3NF), the tables have to be firstly in second normal form (2NF). After that, we remove all the transitive dependencies from the graph as they do not represent important relations. A table that satisfies the conditions for the existence of 2NF and all the attributes that are not a member of one key are not transitively dependent on the primary key. This means that none of the non-key attribute can have their values dependent on any other non-key attribute.

Based on the 2NF tables provided earlier, here is how we can modify them to achieve 3NF:

**1. Customers Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Primary key, also it is unique identity card number for each customer. |
| first\_name | VARCHAR | Customer's first name |
| last\_name | VARCHAR | Customer's last name |
| email | VARCHAR | Customer's email address |
| phone\_number | VARCHAR | Customer's phone number |
| address\_line1 | VARCHAR | First line of the customer’s address |
| address\_line2 | VARCHAR | The second line of the customer’s address if any |
| city | VARCHAR | Customer's city |
| state | VARCHAR | Customer's state |
| postal\_code | VARCHAR | Customer's postal code |
| country | VARCHAR | Customer's country |

*Not a single attribute has been changed; all the attributes are fully functionally dependent on the primary key there is no transitive dependency.*

**2. Products Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | The primary key which will be the unique identifier for every single product. |
| product\_name | VARCHAR | Name of the product |
| product\_description | TEXT | Description of the product |
| price | DECIMAL | Price of the product |
| stock\_quantity | INT | Amount of the product available in the stores |
| category\_id | INT | Foreign key references Categories table |

*It does not require any modification; All the attributes which are other than key attributes are fully functionally dependent on primary key and there are no transitive dependencies.*

**3. Orders Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Product ID, This field is auto generated and contains unique identifier for each orders. |
| order\_date | DATETIME | SWOT assessment of opening new stores Date and time the order was placed |
| customer\_id | INT | Foreign key, references Customers table |
| total\_amount | DECIMAL | Sum of the order means the total cost of the whole order. |
| status | VARCHAR | Delivery status of the order (e. g. , new, shipped, delivered) |

*None to make; All the attributes except the primary key attributes are fully functionally dependent on the primary key and there are no transitive dependencies.*

**4. Order\_Items Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_item\_id | INT | Foreign key, represents the unique identification number for each item of the order |
| order\_id | INT | Foreign key references Orders table |
| product\_id | INT | An important part of the construction of this table is the fact that foreign key references Products table. |
| quantity | INT | Number of units that the customer wants the product to be shipped |
| price | DECIMAL | As ordered for the product the price of the product was as follows. |

*No changes required; all the attributes which are not key are fully functionally dependent to the primary key attributes and there is no transitive dependency.*

**5. Categories Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| category\_id | INT | Primary key, unique identification of each category |
| category\_name | VARCHAR | Name of the category |

*It does not require changes as all the attributes that are not part of key are fully functionally dependent on the primary key and there are no transitive dependencies.*

**6. Payments Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| payment\_id | INT | Primary key, it is an identification number that is given to every payment made. |
| order\_id | INT | Foreign key, refer to the Orders table |
| payment\_date | DATETIME | When they made the payment so in relation to the date and time the repayment was effected. |
| payment\_method | VARCHAR | Mode of payment ( e. g. Credit card, PayPal) |
| payment\_amount | DECIMAL | Amount of the payment |

*There is no need to make any changes: All attributes, which are not key, are fully functionally dependent on the primary key There are no transitive dependencies.*

**7. Employees Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Primary key, it will be an identification number unique for each employee |
| first\_name | VARCHAR | Employee's first name |
| last\_name | VARCHAR | Employee's last name |
| email | VARCHAR | Employee's email address |
| phone\_number | VARCHAR | Employee's phone number |
| hire\_date | DATE | Number of years that the employee has served for the company |
| job\_title | VARCHAR | What Spartan Productions employee a document is about |
| department\_id | INT | Foreign key, referencing to the Departments table |

*I do not find it necessary to make any modifications; all the attributes are functionally dependent on the PK; there are no transitive dependencies.*

**8. Departments Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| department\_id | INT | Primary key, unique identifier which also means a key to create each department. |
| department\_name | VARCHAR | Name of the department |

*No alteration is required; all attributes are totally functionally dependent on the primary attribute and there are no RCA.*

**9. Inventory Table (Already in 3NF)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| inventory\_id | INT | Primary key that refers to unique identification number for every record that belongs to inventory. |
| product\_id | INT | Foreign key, refers the products table |
| warehouse\_id | INT | Foreign key, refers to Warehouses table |
| stock\_quantity | INT | Amount of product available |

*No changes required; any attribute that is not an element of the primary key is fully functionally dependent and there are no transitive dependencies.*

**10. Warehouses Table (Already in 3NF)**

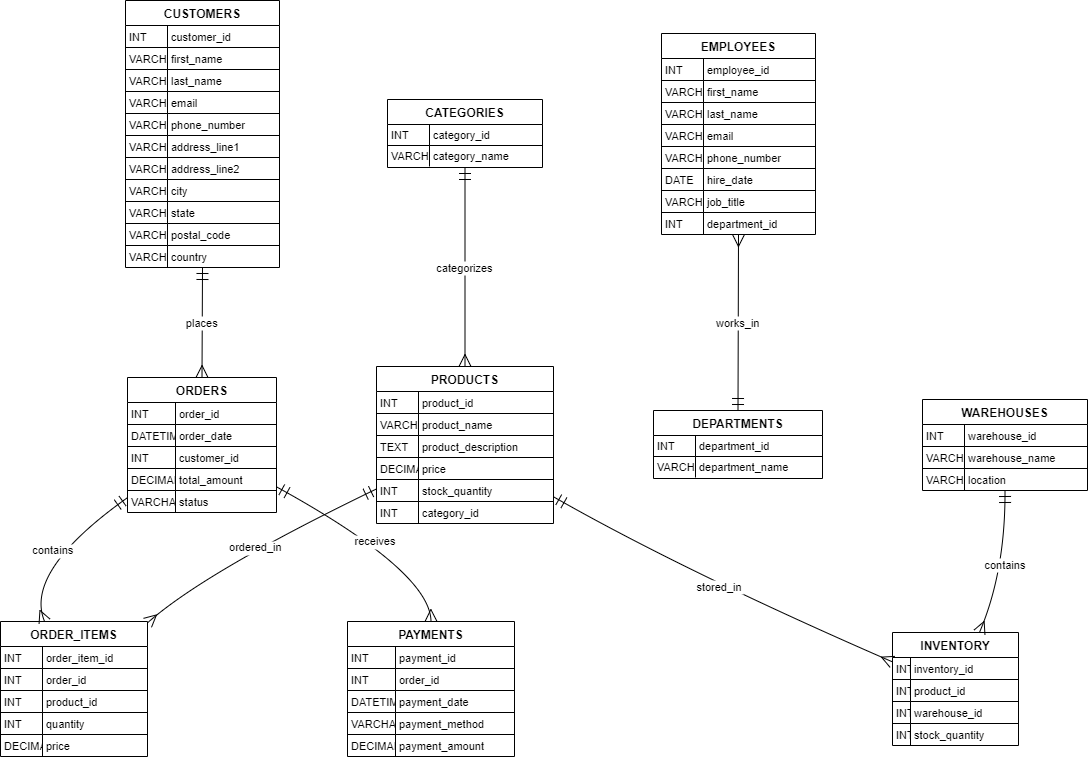
|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| warehouse\_id | INT | Key, can be used to uniquely identify the warehouses |
| warehouse\_name | VARCHAR | Name of the warehouse |
| location | VARCHAR | Location of the warehouse |

*No changes required; All the other attributes are fully functionally dependent on the primary key and there is no transitive dependencies are present.*

**Summary of Changes for 3NF**

It has to be noted that the tables were already in 2NF and 3NF within this schema. No table has any transitive dependency; this means that all the attributes that are non-key are dependent on the primary key of the table. If there were transitive dependencies that is where a non-key attribute has dependency on another non-key attribute then we would need to decompose the table again. However, as designed all of them meet the requirements of 3NF, though it is designed this way purposely.

# ERD



Crow’s foot notation Entity Relationship Diagram is a graphical representation of a database organization with relation to the particular entities. Here's a detailed explanation of the ERD for the Ozmart Retail Group database schema, based on the Mermaid code provided: Here's a detailed explanation of the ERD for the Ozmart Retail Group database schema, based on the Mermaid code provided:

**Entities and Their Attributes**

1. **CUSTOMERS**
   * **Attributes:** First\_Name, Last\_Name, E-Mail, Phone\_Number, AddressLine1, AddressLine2, City, State, Postal\_Code, Country, CustomerID (Primary Key)
   * **Description:** This table is to store details for the customers. Hence, customer\_id is an identifier which distinguishes one customer from another.
2. **PRODUCTS**
   * **Attributes:** TABLE PRODUCT\_DETAILS (product\_id (PK), product\_name, product\_description, price, stock\_quantity, category\_id (FK))
   * **Description:** This table holds information on products which are within the market for sale. In each product an identification code in the form of product\_id is given to each product. category\_id represents the link to the CATEGORIES table to ay out the category of a given product.
3. **ORDERS**
   * **Attributes:** order\_id (Primary Key), order\_date, customer\_id (Foreign Key), total\_amount and status
   * **Description:** This table stores data of the customer orders. Every order has a name which is called order\_id. ‘customer\_id’ is a foreign key to the ‘CUSTOMERS’ table, which signals which customer made the order.
4. **ORDER\_ITEMS**
   * **Attributes:** order\_item\_id (primary key), order\_id(GFK), product\_id (GFK), quantity, price
   * **Description:** The table below identifies the items likely to feature in the order. Every one of the items in an order has got their own number, which is referred to as order\_item\_id. order\_id is a foreign key in the ORDERS table, which shows which product was ordered, and product\_id is another foreign key in PRODUCTS table.
5. **CATEGORIES**
   * **Attributes:** fields including, category\_id (PK), category\_name**.**
   * **Description:** The products are divided into these categories in this table. In each of them, there is category\_id through which this category is distinguished from the others.
6. **PAYMENTS**
   * **Attributes:** payment\_id (primary key ), ord\_id (foreign key), payment date, payment type, payment amount
   * **Description:** This table captures payments for order. Payment information is described in Table 2 below, each of which is distinct by payment\_id. order\_id is a foreign key that suggests that it links to the ORDERS table of the database.
7. **EMPLOYEES**
   * **Attributes:** employee\_id (PK), first\_name, last\_name, e\_mail, telephone\_number, date\_of\_joining, position-holder, department\_id (FK)
   * **Description:** This table is used to store data pertaining to the employees. Every employee has his/her own identity number which is used as a primary key in the table, called employee\_id. The ‘department\_id’ field connects the table to another table of DEPARTMENTS.
8. **DEPARTMENTS**
   * **Attributes:** department\_id — primary key, department\_name
   * **Description:** This table keeps information of departments that exist in the company. Every department has its unique identification number which is referred to as department\_id.
9. **INVENTORY**
   * **Attributes:** inventory\_id –Primary key, product\_id –Foreign key, warehouse\_id –Foreign keys, Stock Quantity
   * **Description:** This table is for management of products’ stock in different warehouses. Every record in the inventory table has a unique identification number which is inventory\_id. product\_id is a foreign key in the table which refers to the PRODUCTS table while the warehouse\_id is also a foreign key in the table, which allows it to refer to the WAREHOUSES table.
10. **WAREHOUSES**
    * **Attributes:** 11warehouse\_id (Primary key), warehouse\_name, location
    * **Description:** This table records details concerning warehouse entities. Every single warehouse has a warehouse\_id which makes it distinct from other warehouses.

**Relationships Between Entities**

1. **CUSTOMERS to ORDERS** (CUSTOMERS ||--o{ ORDERS)
   * **Type:** One-to-Many
   * **Description:** An order can be placed by a single valued customer, but an order can be placed only once by a specific customer.
2. **ORDERS to ORDER\_ITEMS** (ORDERS ||--o{ ORDER\_ITEMS)
   * **Type:** One-to-Many
   * **Description:** An order has many order items but each of the order items is related to just one order.
3. **PRODUCTS to ORDER\_ITEMS** (PRODUCTS ||--o{ ORDER\_ITEMS)
   * **Type:** One-to-Many
   * **Description:** An order item includes one or more products that are most desirable by the customer But a product may belong to one or many order items.
4. **CATEGORIES to PRODUCTS** (CATEGORIES ||--o{ PRODUCTS)
   * **Type:** One-to-Many
   * **Description:** A category can be the product family of multiple products but every product can only belong to a specific category.
5. **ORDERS to PAYMENTS** (ORDERS ||--o{ PAYMENTS)
   * **Type:** One-to-Many
   * **Description:** An order & can be linked to several payments (e. g. , installment payments), but a payment can only be linked to one order.
6. **EMPLOYEES to DEPARTMENTS** (EMPLOYEES }o--|| DEPARTMENTS)
   * **Type:** Many-to-One
   * **Description:** Several employees can be assigned to a specific department whereas individual employee can be assigned to only one department.
7. **PRODUCTS to INVENTORY** (PRODUCTS ||--o{ INVENTORY)
   * **Type:** One-to-Many
   * **Description:** A given product can exist in several inventory records, for example, different locations, while each inventory record belongs to a single product only.
8. **WAREHOUSES to INVENTORY** (WAREHOUSES ||--o{ INVENTORY)
   * **Type:** One-to-Many
   * **Description:** A warehouse can contain various products (which are presented in different records of inventory), but each record of inventory is united only by one warehouse.

**Summary**

The ERD below depicts the architecture of the Ozmart Retail Group’s database incorporating the entities as well as attributes incorporated into the database in addition to how these entities are connected. Every relationship is depicted by using a crow’s foot notation which easily depicts the type of relationship involved; many to one, one to many, or one to one. It aids in determining the structure of the data within the database put into usage as well as how various entities within the database relate with each other which plays a vital role in database design and implementation, query formulation as well as ensuring rightful data integrity.