



**CC5051NI Databases**

**50% Individual Coursework**

**Autumn 2023**

**Student Name: Pratik Karanjit**

**London Met ID:** 22067545

**Assignment Submission Date: Sunday, January 15, 2024**

**Word Count: 4061**

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

**Table Of Contents**

[1.Introduction 1](#_Toc156159875)

[1.1 Introduction to the business 1](#_Toc156159876)

[1.2 Business Rules 2](#_Toc156159877)

[1.3 Identification of entities and attributes 3](#_Toc156159878)

[2. Initial ERD 6](#_Toc156159879)

[3. Normalization 11](#_Toc156159880)

[4. Final ERD 18](#_Toc156159881)

[5. Implementation 26](#_Toc156159882)

[5.1 Creation of tables and results 26](#_Toc156159883)

[5.2 Populate data and its results 34](#_Toc156159884)

[6. Database queries 42](#_Toc156159885)

[6.1 Information Queries 42](#_Toc156159886)

[6.2 Transaction queries 47](#_Toc156159887)

[7. Critical evaluation 52](#_Toc156159888)

[8. Database dump file and drop query 53](#_Toc156159889)

[8.1 Database dump file creation 53](#_Toc156159890)

[8.2 Drop queries 55](#_Toc156159891)

[9. References 59](#_Toc156159892)

**Table Of Figures**

[Figure 1: Initial ERD 6](#_Toc156159814)

[Figure 2: Final ERD. 18](file:///C:\Users\prati\Downloads\Database%20coursewoprkl.docx#_Toc156159815)

[Figure 3: Create user. 26](#_Toc156159816)

[Figure 4: Grant access to the user. 26](#_Toc156159817)

[Figure 5: Connect to user. 26](#_Toc156159818)

[Figure 6:Create vendor table. 27](#_Toc156159819)

[Figure 7: Vendor table details. 27](#_Toc156159820)

[Figure 8: Create discount table. 28](#_Toc156159821)

[Figure 9: Discount table details. 28](#_Toc156159822)

[Figure 10: Transaction table details. 29](#_Toc156159823)

[Figure 11: Transaction table details. 29](#_Toc156159824)

[Figure 12: Create customer table. 30](#_Toc156159825)

[Figure 13: View customer table details. 30](#_Toc156159826)

[Figure 14: Create product table. 31](#_Toc156159827)

[Figure 15: View product details. 31](#_Toc156159828)

[Figure 16: Create order table. 32](#_Toc156159829)

[Figure 17: View orders table. 32](#_Toc156159830)

[Figure 18: Create order details table. 33](#_Toc156159831)

[Figure 19: View order details table. 33](#_Toc156159832)

[Figure 20: Insert data in vendor table and display them. 34](#_Toc156159833)

[Figure 21: Insert values in discount table and display them. 35](#_Toc156159834)

[Figure 22: Insert values in product table. 36](#_Toc156159835)

[Figure 23: Display product table values. 36](#_Toc156159836)

[Figure 24: Insert values in customer table. 37](#_Toc156159837)

[Figure 25: Display values of customer table. 37](#_Toc156159838)

[Figure 26:Insert values in transaction table and display them. 38](#_Toc156159839)

[Figure 27: Insert values in orders table. 39](#_Toc156159840)

[Figure 28: Display values of orders table. 40](#_Toc156159841)

[Figure 29: Insert values in order details table and display them. 41](#_Toc156159842)

[Figure 30: Information queries Question 1 42](#_Toc156159843)

[Figure 31: Information query question 2 43](#_Toc156159844)

[Figure 32: Information query question 3 44](#_Toc156159845)

[Figure 33: Information query question 4 45](#_Toc156159846)

[Figure 34: Information query question 5 46](#_Toc156159847)

[Figure 35: Transaction query question 1 47](#_Toc156159848)

[Figure 36: Transaction query question 2 48](#_Toc156159849)

[Figure 37: Transaction query question 3 49](#_Toc156159850)

[Figure 38: Transaction query question 4 50](#_Toc156159851)

[Figure 39: Transaction query question 5 51](#_Toc156159852)

[Figure 40: Creation of dmp file. 53](#_Toc156159853)

[Figure 41: Process of dmp file creation. 54](#_Toc156159854)

[Figure 42: Drop table OrderDetails. 55](#_Toc156159855)

[Figure 43: Drop table Orders. 55](#_Toc156159856)

[Figure 44: Drop table Product. 56](#_Toc156159857)

[Figure 45: Drop table Customer. 56](#_Toc156159858)

[Figure 46: Drop table Transaction. 57](#_Toc156159859)

[Figure 47: Drop table Discount. 57](#_Toc156159860)

[Figure 48: Drop table vendor. 58](#_Toc156159861)

**Table Of Tables**

[Table 1: Entities and attributes of Customer. 3](#_Toc156159862)

[Table 2: Entities and attributes of product. 4](#_Toc156159863)

[Table 3: Entities and attributes of orders. 5](#_Toc156159864)

[Table 4: Data dictionary of orders. 8](#_Toc156159865)

[Table 5: Data dictionary for product. 9](#_Toc156159866)

[Table 6: Data dictionary for customer. 10](#_Toc156159867)

[Table 7: Final ERD data dictionary for discount. 20](#_Toc156159868)

[Table 8: Final ERD data dictionary for transaction. 20](#_Toc156159869)

[Table 9: Final ERD data dictionary for vendor. 21](#_Toc156159870)

[Table 10: Final ERD data dictionary for orders. 22](#_Toc156159871)

[Table 11: Final ERD data dictionary for product. 23](#_Toc156159872)

[Table 12: Final ERD data dictionary for customer. 24](#_Toc156159873)

[Table 13: Final ERD data dictionary for OrderDetails. 25](#_Toc156159874)

**Acknowledgement**

First and foremost, I would like to thank our module leader Yunisha Bajracharya Ma’am and our lecturer and tutor Aadesh Tandukar Sir for guiding me throughout this coursework. I am genuinely thankful for providing us with the expertise and invaluable insights that helped me completing this coursework.

I am also thankful to my respective colleagues who helped us and encouraged us to make the project successful. Last but not the least, I express my sincere gratitude to all those who contributed to this project directly and indirectly.

# 1.Introduction

## 1.1 Introduction to the business

The upcoming online marketplace Gadget Emporium is an online platform that provides a wide range of electronic products to private customers and business organizations. Mr. John, the founder of Gadget Emporium is a passionate entrepreneur and electronic enthusiast. The store currently needs a database system to support Mr. John’s Gadget Emporium that can manage products, orders, and customers. It should also provide transaction details to the buyer and maintain the customer's database side by side.

All these processes require a strong database system that should be manageable, maintainable, and sustainable. The database shall also have multiple other entities like vendor details, discount options according to the customer category, detailed info about customer’s transactions, and so on. This very system needs to be developed using SQL PLUS using Oracle database. It can be easily done by going through ERDs and normalizations.

**Current business activities and operations**

Gadget Emporium primarily focuses on selling their electronic devices through e-commerce. The business has not yet started as Mr. John is still on the planning phase of the project. The store plans on linking with different vendors that would fulfill the customer’s demands. This seamless delivery, customer’s wide range of choices and further purchase processes are to be maintained by the database.

## 1.2 Business Rules

Gadget emporium follows a set of business rules that affects the implementation of its database schema.

* One customer can have multiple orders whereas one order should be linked to only one customer.
* A customer ordering for products could be optional whereas if an order is placed, it must be associated with the customer.
* Customer getting a discount is optional as regular customers are assigned to a discount of nil.
* For vendor to have products is optional as vendors might run out of stock occasionally.
* One invoice should be generated for each customer.
* Customers are divided into three groups, i.e., Regular, Staff and VIP. Each of which is entitled to different discount rates. Regular gets a discount of 0%, Staff 5% and VIPs 10%.
* The system should maintain information about electronic gadgets like product name, description, category, price, and stock levels.
* Each product should belong to a particular category while each category may hold one or more products.
* Customers should be able to know the availability of products by providing availability status or stating the total quantity left itself.
* Each product should be associated with a single vendor. Each vendor can supply one or more products.
* An order can have multiple products and any one type of product might be included in multiple orders.
* Each order detail must have one payment option.

## 1.3 Identification of entities and attributes

Identifying entities and attributes is a crucial step in creation of further ERDs and its normalization. List of various entities and attributes are mentioned below:

1. **Customer**

Customer is a major entity of this coursework as every product and order revolves around it. Customer has few attributes which can be viewed below:

|  |  |
| --- | --- |
| Entity | Attributes |
| Customer | CustomerID (PK) |
| CustomerFirstName |
| CustomerLastName |
| CustomerPhoneNumber |
| CustomerEmail |
| CustomerAddress |
| CustomerCategory |
| Discount |

Table : Entities and attributes of Customer.

1. **Product**

Product entity holds all the product details that would be required for the customers to view and purchase them. Its attributes can be viewed below:

|  |  |
| --- | --- |
| Entity | Attributes |
| Product | ProductID (PK) |
| ProductName |
| ProductPrice |
| ProductDescription |
| StockQuantity |
| ProductCategory |
| VendorID |
| VendorName |

Table : Entities and attributes of product.

1. **Orders**

Orders entity holds all the order details of the customer. This entity will help in sorting customer’s purchase dates and totals. Its attributes can be viewed below:

|  |  |
| --- | --- |
| Entity | Attributes |
| Orders | OrderID (PK) |
| OrderDate |
| OrderTotal |
| InvoiceID |
| InvoiceTotal |
| PaymentOption |
| CustomerID (FK) |
| ProductID (FK) |
| OrderQuantity |

Table : Entities and attributes of orders.

# 2. Initial ERD

An Entity Relation Diagram (ERD) is a graphical representation that displays the relationship between objects and attributes. An ERD uses data modeling techniques that can help define business processes and serve as a foundation to the system. (TechTarget, 2024)

A diagram of a data flow

Description automatically generated with medium confidence

Figure : Initial ERD

**Assumptions**

1. One customer can make multiple orders whereas multiple orders can be made by a single customer.
2. A customer to have placed an order is optional whereas if an order is placed, it is mandatory to have a customer related to it.
3. One product can have multiple orders of it and multiple orders can be of multiple products as well.
4. A product to have an order placed is optional whereas if an order is placed, it is mandatory to have a product linked to it.

Data dictionary for initial ERD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Orders | OrderID | VARCHAR (20) | Primary Key | Uniquely identifies each order with an id. |
| OrderDate | DATE | Not Null | Gives detail about the date of order. |
| OrderTotal | NUMBER | Not Null | Gives the sum of order prices. |
| InvoiceID | NUMBER | Not Null | Helps in unique identification of invoices. |
| InvoiceTotal | NUMBER | Not Null | Gives the total after discount. |
| PaymentOption | VARCHAR (30) | Not Null | Gives choices of payment option. |
| CustomerID | VARCHAR (20) | Foreign Key | Stays as a foreign key for customer details. |
| ProductID | VARCHAR (10) | Foreign Key | Stays as a foreign key for product details. |

Table : Data dictionary of orders.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Product | ProductID | VARCHAR (10) | Primary Key | Uniquely identifies each product with an id. |
| ProductName | VARCHAR (50) | Not Null | Gives name of products. |
| ProductPrice | NUMBER | Not Null | Gives the prices of products. |
| ProductDescription | VARCHAR (200) | Not Null | Gives description of product. |
| ProductCategory | VARCHAR (50) | Not Null | Gives detail about type of product. |
| StockQuantity | NUMBER | Not Null | Gives availability knowledge of products in stock. |
| VendorID | NUMBER | Not Null | Gives unique id to each vendor. |
| VendorName | VARCHAR (30) | Not Null | Gives the names of vendors. |

Table : Data dictionary for product.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Customer | CustomerID | VARCHAR (15) | Primary Key | Uniquely identifies each customer with an id. |
| CustomerFirstName | VARCHAR (30) | Not Null | Gives first name of customer. |
| CustomerLastName | VARCHAR (30) | Not Null | Gives last name of customer. |
| CustomerAddress | VARCHAR (20) | Not Null | Gives address of customer. |
| CustomerPhoneNumber | NUMBER | Not Null, Unique | Gives phone number of customer. |
| CustomerEmail | VARCHAR (50) | Not Null, Unique | Gives email of customer. |
| CustomerCategory | VARCHAR (4) | Not Null | Gives category of customer. |
| DiscountRate | NUMBER | Not Null | Gives discount rate of customers. |

Table : Data dictionary for customer.

# Normalization

Normalization is the process of organizing data in database. It starts off with redundant data in a table which are not properly separated. This data redundancy is removed by normalization processes. Normalization divides larger tables into smaller tables and links them with relationships. This helps in creating tables with non-redundant data and maintains a healthy database. (JavaTPoint, 2021)

Normalization processes from UNF to 3NF is done below:

**UNF**

**CommerceRecord** (OrderID, OrderDate, OrderTotal, CustomerID, CustomerFirstName, CustomerEmail, CustomerPhoneNumber, CustomerLastName, CustomerAddress, CustomerCategory, DiscountRate, PaymentOption, InvoiceID, InvoiceTotal, {ProductID, ProductName, ProductPrice, StockQuantity, ProductDescription, ProductCategory, OrderQuantity, VendorID, VendorName})

In the above UNF, I have listed down all the attributes and given it an entity name. Orders has been taken as a reference for solving this normalization process as it can have customer’s and product’s information.

A suitable unique identifier, OrderID, has been given as a primary key to represent the entire attributes.

All the multi valued attributes (repeating groups) are enclosed in curly brackets ({}). This sums up the UNF process for this normalization.

**1NF**

**Order-customer-1** (OrderID, OrderDate, OrderTotal, CustomerID, CustomerFirstName, CustomerLastName, CustomerAddress, CustomerEmail, CustomerPhoneNumber, CustomerCategory, DiscountRate, PaymentOption, InvoiceID, InvoiceTotal)

**Product-order-1** (ProductID, OrderID\*, ProductName, ProductPrice, StockQuantity, ProductDescription, ProductCategory, OrderQuantity, VendorID, VendorName)

In the above 1NF process, repeating groups in the UNF table has been removed and a new table is formed which holds all the repeating groups. A new entity has been named for it.

The unique identifier from UNF has been carried forward to the new table. Also, ProductID has been chosen as the unique identifier (Primary Key) for the new table. The multi valued attributes and non multi valued attributes are now separated in the 1NF.

The two tables formed in 1NF are named as Order-customer-1 and Product-order-1.

**2NF**

**From product-order-1,**

ProductID, OrderID → OrderQuantity

ProductID → ProductName, ProductPrice, ProductDescription, ProductCategory, VendorID, VendorName

OrderID → X

In the above 2NF, the partial dependencies and fully dependent attributes are separated. The composite keys are linked with OrderQuantity. This makes the rise of a new table with only the composite keys and OrderQuantity.

Another transitive dependency is checked for ProductID which was the unique identifier for 1NF table. All the attributes linked with ProductID are kept together and another table is formed for this.

As no attributes are linked with OrderID, it is a single attribute which does not form a new table.

This sums up 2NF process with creation of one new table. The total number of tables now are three.

Tables so formed are:

**Order-customer-2** (OrderID, OrderDate, OrderTotal, CustomerID, CustomerFirstName, CustomerEmail, CustomerPhoneNumber, CustomerLastName, CustomerAddress, CustomerCategory, DiscountRate, PaymentOption, InvoiceID, InvoiceTotal)

**Product-order-2** (ProductID\*, OrderID\*, OrderQuantity)

**Product-vendor-2** (ProductID, VendorID, VendorName, ProductName, ProductPrice, ProductDescription, ProductCategory, StockQuantity)

**3NF**

**In product-vendor-2,**

ProductID → VendorID → VendorName

ProductID → VendorID

VendorID → VendorName

ProductID gives information about VendorID and VendorID gives information about VendorName. With this the VendorID and VendorName can be extracted from product-vendor-2 table.

This helps in formation of a new table “Vendor” which holds attributes that are only linked with vendor.

**In order-customer-2,**

OrderID → CustomerID → CustomerFirstName, CustomerLastName, CustomerAddress, CustomerEmail, CustomerPhoneNumber, CustomerCategory, DiscountRate

OrderID → CustomerID

CustomerID → CustomerFirstName, CustomerLastName, CustomerAddress, CustomerEmail, CustomerPhoneNumber, CustomerCategory, Discount

In the above transitive dependency, OrderID is linked with CustomerID which helps in separating all the customer details along with DiscountRate. This DiscountRate is again separated later.

**In order-customer-2,**

OrderID → InvoiceID → InvoiceTotal, paymentOption

OrderID → InvoiceID

InvoiceID → InvoiceTotal, PaymentOption

The invoiceTotal and paymentOption are separated with InvoiceID which helps in creation of a new table “Transaction”.

**Again, in order-customer-2,**

CustomerID → CustomerCategory → Discount

CustomerID → CustomerCategory

CustomerCategory → DiscountRate

This transitive dependency helps in separating DiscountRate from customer details table which now allows in creation of Customer as well as Discount table simultaneously.

Tables formed after 3NF are as follows:

**Orders -3** (OrderID, InvoiceID\*, CustomerID\*, OrderDate, OrderTotal)

**Customer-3** (CustomerID, CustomerCategory\*, CustomerFirstName, CustomerLastName, CustomerAddress, CustomerPhoneNumber, CustomerEmail)

**Product-3** (ProductID, VendorID\*, ProductName, ProductPrice, StockQuantity, ProductDescription, ProductCategory)

**Product-order-3** (ProductID\*, OrderID\*, OrderQuantity)

**Vendor-3** (VendorID, VendorName)

**Transaction-3** (InvoiceID, InvoiceTotal, PaymentOption)

**Discount-3** (CustomerCategory, DiscountRate)

This sums up the normalization processes as all the attributes are separated with its corresponding attributes. The data redundancy that previously prevailed has now been successfully removed and a clear and healthy database has been created with the help of normalization processes.

# Final ERD

A close-up of a document

Description automatically generatedAn ERD is a type of flowchart that displays how entities and attributes line up with each other and how they can be related to each other while being on different tables. These attributes are given keys to uniquely identify themselves. Also, these keys help in connecting tables with primary key acting as a foreign in another table. (LucidChart, 2024)

Figure : Final ERD.

**Assumptions**

1. Each vendor can have multiple products whereas each product should be associated with one vendor.
2. A vendor to have products is optional whereas products to be associated with a vendor is mandatory.
3. A customer to have discount is optional as Regular customers are entitled to a discount of zero. If discount is available, its mandatory to be linked with a customer.
4. A customer can only have one category of discount. Similarly, discount is given to a particular customer.
5. An order should be associated with a single transaction and a transaction should be associated with one order.
6. It is mandatory for both orders and transactions to take place if one or the other occurs.
7. One customer can have multiple orders, but multiple orders should be linked with one customer.
8. A customer to have placed an order is mandatory whereas if an order is placed it must be linked with a customer.
9. As orders and products table form many to many relations, a new table is established that acts as a bridge entity between each other. Multiple side is placed on the bridge entity which helps in removing the existing many to many relations.

Data dictionary for the final ERD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attributes | Data type | Constraints | Description |
| Discount | CustomerCategory | VARCHAR (10) | Primary Key | Uniquely identifies each category. |
| DiscountRate | NUMBER | Not Null | Gives discount rate of customers. |

Table : Final ERD data dictionary for discount.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attributes | Data type | Constraints | Description |
| Transaction | InvoiceID | NUMBER | Primary Key | Uniquely identifies each invoice. |
| InvoiceTotal | NUMBER | Not Null | Gives the total after discount. |
| PaymentOption | VARCHAR (30) | Not Null | Gives the payment options. |

Table : Final ERD data dictionary for transaction.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attributes | Data type | Constraints | Description |
| Vendor | VendorID | NUMBER | Primary Key | Uniquely identifies each vendor. |
| VendorName | VARCHAR (30) | Not Null | Gives vendor names. |

Table : Final ERD data dictionary for vendor.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Orders | OrderID | VARCHAR (20) | Primary Key | Uniquely identifies each order with an id. |
| OrderDate | DATE | Not Null | Gives detail about the date of order. |
| OrderTotal | NUMBER | Not Null | Gives the sum of order prices. |
| InvoiceID | NUMBER | Foreign Key | Stays as foreign key for invoice details. |
| CustomerID | VARCHAR (20) | Foreign Key | Stays as a foreign key for customer details. |

Table : Final ERD data dictionary for orders.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Product | ProductID | VARCHAR (10) | Primary Key | Uniquely identifies each product with an id. |
| ProductName | VARCHAR (50) | Not Null | Gives name of products. |
| ProductPrice | NUMBER | Not Null | Gives the prices of products. |
| ProductDescription | VARCHAR (200) | Not Null | Gives description of product. |
| ProductCategory | VARCHAR (50) | Not Null | Gives detail about type of product. |
| StockQuantity | NUMBER | Not Null | Gives availability knowledge of products in stock. |
| VendorID | NUMBER | Foreign Key | Stays as foreign key for vendor details. |

Table : Final ERD data dictionary for product.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Data type | Constraints | Description |
| Customer | CustomerID | VARCHAR (15) | Primary Key | Uniquely identifies each customer with an id. |
| CustomerFirstName | VARCHAR (30) | Not Null | Gives first name of customer. |
| CustomerLastName | VARCHAR (30) | Not Null | Gives last name of customer. |
| CustomerAddress | VARCHAR (20) | Not Null | Gives address of customer. |
| CustomerPhoneNumber | NUMBER | Not Null, Unique | Gives phone number of customer. |
| CustomerEmail | VARCHAR (50) | Not Null, Unique | Gives email of customer. |
| CustomerCategory | VARCHAR (4) | Foreign Key | Stays as foreign key for customer category details. |

Table : Final ERD data dictionary for customer.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attributes | Data type | Constraints | Description |
| OrderDetails | ProductID | VARCHAR (20) | Primary Key and Foreign Key | Stays as a composite key. |
| OrderID | VARCHAR (10) | Primary Key and Foreign Key | Stays as a composite key. |
| OrderQuantity | NUMBER | Not Null | Gives the order quantity. |

Table : Final ERD data dictionary for OrderDetails.

# Implementation

## 5.1 Creation of tables and results

**Creating user and granting permission**

A black screen with white text

Description automatically generated

Figure : Create user.

A black screen with white text

Description automatically generated

Figure : Grant access to the user.

A black screen with white text

Description automatically generated

Figure : Connect to user.

**Vendor Table**

A black screen with white text

Description automatically generated

Figure :Create vendor table.

**Resulting Output**

A screen shot of a computer

Description automatically generated

Figure : Vendor table details.

**Discount Table**

A black screen with white text

Description automatically generated

Figure : Create discount table.

**Resulting Output**

A screen shot of a computer

Description automatically generated

Figure : Discount table details.

**Transaction table**

A screen shot of a computer

Description automatically generated

Figure : Transaction table details.

**Resulting Output**

**A black screen with white text

Description automatically generated**

Figure : Transaction table details.

**Customer table**

A computer screen shot of white text

Description automatically generated

Figure : Create customer table.

**Resulting Output**

**A screen shot of a computer

Description automatically generated**

Figure : View customer table details.

**Product table**

A computer screen with white text

Description automatically generated

Figure : Create product table.

**Resulting Output**

**A screenshot of a computer

Description automatically generated**

Figure : View product details.

**Order Table**

A computer screen shot of white text

Description automatically generated

Figure : Create order table.

**Resulting Output**

**A screen shot of a computer

Description automatically generated**

Figure : View orders table.

**OrderDetails table**

A black screen with white text

Description automatically generated

Figure : Create order details table.

**Resulting output**

**A screen shot of a computer

Description automatically generated**

Figure : View order details table.

## Populate data and its results

Inserting values in vendor table and displaying them

A screenshot of a computer

Description automatically generated

Figure : Insert data in vendor table and display them.

Inserting values in Discount table and displaying them

A screen shot of a computer

Description automatically generated

Figure : Insert values in discount table and display them.

Inserting values in product table and displaying them

A screenshot of a computer

Description automatically generated

Figure : Insert values in product table.

A screen shot of a computer

Description automatically generated

Figure : Display product table values.

Inserting values into Customer table and displaying them

A black screen with white text

Description automatically generated

Figure : Insert values in customer table.

A screen shot of a computer

Description automatically generated

Figure : Display values of customer table.

Inserting values into Transaction table and displaying them

A screenshot of a computer screen

Description automatically generated

Figure :Insert values in transaction table and display them.

Inserting values into Orders table and displaying them

A computer screen with white text

Description automatically generated

Figure : Insert values in orders table.

A computer screen shot of a black screen

Description automatically generated

Figure : Display values of orders table.

Inserting values into OrderDetails table and displaying them

A screenshot of a computer screen

Description automatically generated

Figure : Insert values in order details table and display them.

# Database queries

## 6.1 Information Queries

Q1. List all the customers that are also staff of the company.

**SQL** **script**

**SELECT** \*

**FROM** Customer

**WHERE** CustomerCategory = 'Staff';

A black screen with white text

Description automatically generated

Figure : Information queries Question 1

**Purpose** **and explanation**

Get all the data from customer table whose customerCategory is “Staff”. This outputs CustomerID, CustomerFirstName, CustomerLastName, CustomerAddress, CustomerPhoneNumber, CustomerEmail along with CustomerCategory.

Q2. List all the orders made for any product between the dates 01-05-2023 till 28-05-2023.

**SQL** **Script**:

**SELECT** o.ORDERID, o.ORDERDATE, o.ORDERTOTAL, o.CUSTOMERID, o.INVOICEID

**FROM** orders o

**JOIN** orderdetails od ON o.ORDERID = od.ORDERID

**JOIN** product p ON od.PRODUCTID = p.PRODUCTID

**WHERE** p.PRODUCTID = 'J1'

**AND** o.ORDERDATE BETWEEN TO\_DATE('01-MAY-2023', 'DD-MON-YYYY') AND TO\_DATE('28-MAY-2023', 'DD-MON-YYYY');

A black screen with white text

Description automatically generated

Figure : Information query question 2

**Purpose and explanation**

To find those orders that were made during the duration of May 1 to May 28. First, OrderID, OrderDate, OrderTotal, CustomerID and InvoiceID are selected to display. Then, order details, orders and product table are joined to extract data from respective tables. Lastly, a particular product is chosen, here, J1 is taken and order date is taken out from it.

Q3. List all the customers with their order details and the customers who have not ordered any products yet.

**SQL** **Script**:

**SELECT** C. CUSTOMERID, C. CUSTOMERFIRSTNAME, C.CUSTOMERLASTNAME, C. CUSTOMERADDRESS, C. CUSTOMERPHONENUMBER, C. CUSTOMEREMAIL, C. CUSTOMERCATEGORY, OD.ORDERID, OD. PRODUCTID, OD.ORDERQUANTITY FROM CUSTOMER C

**LEFT** JOIN ORDERS O ON C.CUSTOMERID = O.CUSTOMERID

**LEFT** JOIN ORDERDETAILS OD ON O.ORDERID = OD.ORDERID;

A black screen with white text

Description automatically generated

Figure : Information query question 3

**Purpose and explanation**  
The required attributes are selected for display. Left join is then used for orders and customer table. Similarly, left join is applied on orderDetails to join orders and orderDetails table. This allows to list all customers with their order details and the customers who have not ordered any products yet.

Q4. List all product details that have the second letter ‘a’ in their product name and have a stock quantity more than 50.

**SQL** **Script**

**SELECT** \*

**FROM** Product

**WHERE** ProductName LIKE '\_a%'

**AND** StockQuantity > 50;

A black screen with white text

Description automatically generated

Figure : Information query question 4

**Purpose and explanation**

All attributes of product table are selected whose product name has second letter ‘a’ and stock quantity more than 50. This displayed the result Samsung H213 whose second letter is indeed a and stock quantity is greater than 50.

Q5. Find out the customer who has ordered recently.

**SQL** **Script**

**SELECT** C. CustomerFirstName || ' ' || C.CustomerLastName AS Customer\_name, O.OrderID, O.OrderDate, C. CustomerEmail

**FROM** Customer C

**JOIN** Orders O ON C. CustomerID = O.CustomerID

**WHERE** O.OrderDate = (SELECT MAX(OrderDate) FROM Orders);

A screen shot of a computer

Description automatically generated

Figure : Information query question 5

**Purpose and explanation**

To find the latest order customer, first name and last name has been concatenated. Orders and customers tables are joined, and order date is extracted by using MAX function. This resulted in outputting only the customer that had the recent most order.

## 6.2 Transaction queries

Q1. Show the total revenue of the company for each month.

**SQL** **Script**

**SELECT** TO\_CHAR(OrderDate, 'MM-YYYY') AS Month, SUM(T.InvoiceTotal) AS TotalRevenue

**FROM** ORDERS O

**JOIN** Transaction T ON O.InvoiceID = T. InvoiceID GROUP BY TO\_CHAR(OrderDate, 'MM-YYYY');

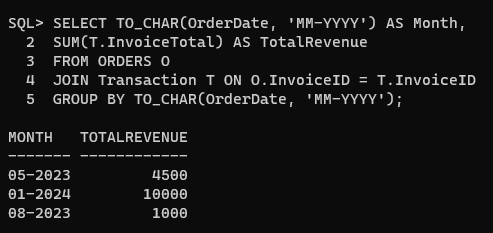


Figure : Transaction query question 1

**Purpose and explanation**

Firstly, orderDate has been arranged in an easier format. SUM function is used to add up the total revenue of each month. JOIN is used on transaction and orders. Group BY is used to group together months and year.

Q2. Find those orders that are equal or higher than the average order total value.

**SQL** **SCRIPT**

**SELECT** OrderID, OrderTotal

**FROM** Orders

**WHERE** OrderTotal >= (SELECT AVG(OrderTotal) FROM Orders);

A screen shot of a computer

Description automatically generated

Figure : Transaction query question 2

**Purpose and explanation**

WHERE is used to filter those order totals that are equal to or above the average order total from orders table. This allowed to display OrderID and OrderTotal of those orders that are equal or higher than the average order total value.

Q3. List the details of vendors who have supplied more than 3 products to the company.

**SQL** **Script**

**SELECT** V.VendorID, V.VendorName

**FROM** Vendor V

**JOIN** Product P ON V.VendorID = P.VendorID

**GROUP** **BY** V.VendorID, V.VendorName

**HAVING** COUNT(P.ProductID) > 3;

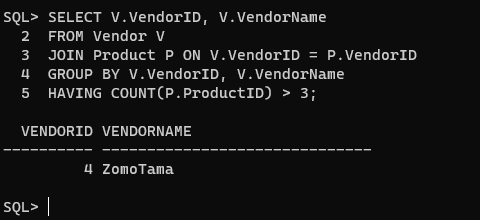


Figure : Transaction query question 3

**Purpose and explanation**

From vendor table, VendorID and Name are selected to join product and vendor tables. Using GROUP BY, rows with the same combination of VendorID and VendorName will be grouped together.

Q4. Show the top 3 product details that have been ordered the most.

**SQL** **Script**

**SELECT** ROWNUM AS RANK, PRODUCTID, PRODUCTNAME, ORDERQUANTITY **FROM** (

**SELECT** P.PRODUCTID, P.PRODUCTNAME, SUM(OD.ORDERQUANTITY) AS ORDERQUANTITY FROM PRODUCT P

**JOIN** ORDERDETAILS OD ON P.PRODUCTID = OD. PRODUCTID

**GROUP** **BY** P.PRODUCTID, P. PRODUCTNAME

**ORDER** **BY** ORDERQUANTITY DESC

)

**WHERE** ROWNUM <= 3;

A computer screen shot of a black screen

Description automatically generated

Figure : Transaction query question 4

**Purpose and explanation**

OrderDetails and Product table are joined and required attributes are taken. OrderQuantity is set to descending order for it to display the highest ordered product name to be at the top. ROWNUM <= 3 displays the top 3 most ordered product details.

Q5. Find out the customer who has ordered the most in August with his/her total spending on that month.

**SQL** **Script**

**SELECT** \* FROM (

**SELECT** C.CUSTOMERID, C.CUSTOMERFIRSTNAME, C.CUSTOMERLASTNAME, **SUM**(O.ORDERTOTAL) AS TOTAL\_SPENDING FROM CUSTOMER C

**JOIN** ORDERS O ON C.CUSTOMERID = O.CUSTOMERID

**WHERE** TO\_CHAR(O.ORDERDATE, 'MM') = '08'

**GROUP** BY C. CUSTOMERID, C.CUSTOMERFIRSTNAME, C.CUSTOMERLASTNAME

**ORDER BY** TOTAL\_SPENDING DESC

) **WHERE** ROWNUM = 1;

A screen shot of a computer

Description automatically generated

Figure : Transaction query question 5

**Purpose and explanation**

SUM of order total is taken and orders with customer table is joined. Month of August is selected and grouped together with customer details. Finally, ROWNUM =1 displays the customer who has ordered the most in August with his/her total spending on that month.

# Critical evaluation

The coursework based on developing a database system for Gadget Emporium has expanded my knowledge in this module. Having completed only a handful of such projects, this coursework was really a test of my intellect and how well I could bring things together to design a working database. This module has brought insights about creating something out of a given scenario which is what a real-world project is all about. It has taught many things along its way for example, in the process of creating ERDs, I spent valuable time creating these diagrams in draw.io which has enhanced my skills. Talking outside of module domain, in companies where there is so much to develop and create, this has helped a lot in covering the basics of Oracle and SQL PLUS. With creating databases comes many things that underlie the topic. This coursework has created a strong base of how normalization is done to remove data redundancy from tables. Also, designing of ERDs has been learnt which is very important for other subjects as well. Similarly, data dictionary can also be utilized in subjects like Software Engineering where designing is so crucial.

This coursework has not only provided knowledge about database systems but also practically use it for data modeling. With a given scenario, the task of creating database system has been successfully completed. The proposed system is now able to keep track of customers, products, and orders that the customers may place. The coursework has been completed solely with the use of Oracle SQL PLUS. Along the way, I faced some challenges that were a roadblock to my completion but eventually with constant practice and determination I have been able to overcome these challenges. Also, mentioning about my strengths that I had over this period of completion is my skill in using draw.io and documentation. It may not seem a big of a difference, but it has been helpful as I have used these tools quite often in the past and using it repeatedly has assisted me in completing these tasks faster and with more confidence. In conclusion, the assessment has been a successful task which taught abundance of things with practical knowledge.

# Database dump file and drop query

## 8.1 Database dump file creation

A black screen with white text

Description automatically generated

Figure : Creation of dmp file.

A screen shot of a computer

Description automatically generated

Figure : Process of dmp file creation.

## 8.2 Drop queries

A screen shot of a computer

Description automatically generated

Figure : Drop table OrderDetails.

A screen shot of a computer

Description automatically generated

Figure : Drop table Orders.

A screen shot of a computer

Description automatically generated

Figure : Drop table Product.

A screenshot of a computer

Description automatically generated

Figure : Drop table Customer.

A screen shot of a computer

Description automatically generated

Figure : Drop table Transaction.

A black screen with white text

Description automatically generated

Figure : Drop table Discount.

A screen shot of a computer

Description automatically generated

Figure : Drop table vendor.

# References

JavaTPoint, 2021. *Normalization.* [Online]   
Available at: https://www.javatpoint.com/dbms-normalization  
[Accessed 12 1 2024].

LucidChart, 2024. *What is an entity relationship diagram?.* [Online]   
Available at: https://www.lucidchart.com/pages/er-diagrams  
[Accessed 12 1 2024].

LucidChart, 2024. *What is an Entity Relationship Diagram?.* [Online].

TechTarget, 2024. *Entity Relationship Diagram.* [Online]   
Available at: https://www.techtarget.com/searchdatamanagement/definition/entity-relationship-diagram-ERD  
[Accessed 12 1 2024].