DBD 281 PROJECT

Malcolm Mudehwe 577564, Nosipho donkrag 577354, Phogole Mamogobo 577526

INDEX:

1. Introduction:

The Pretoria North Supermarket, located in the heart of Pretoria North, has been serving customers for over a decade. Despite its small size, it offers a variety of products ranging from fresh produce, meats, dairy products, and dry goods. The store's aim is to provide a convenient shopping experience for customers with personalized customer service and a wide range of products.

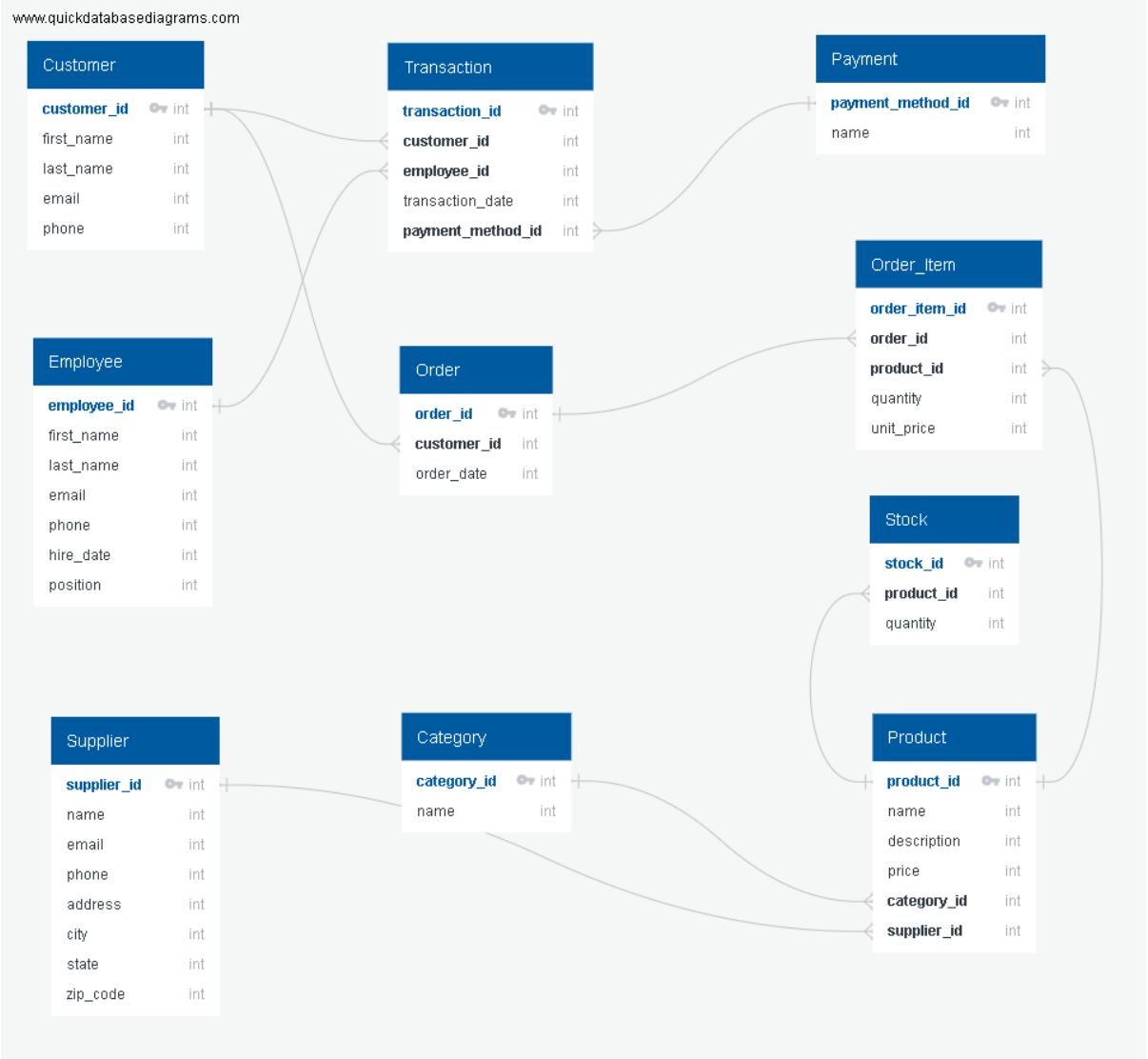
2. Background:

The Pretoria North Supermarket was established in 2010 by a local entrepreneur with a passion for providing quality products and services to the community. The store has since become a staple in the small town, attracting loyal customers who appreciate its personalized service and wide selection of products.

Over the years, the store has adapted to changing customer needs and preferences by expanding its product offerings and incorporating new technologies to streamline operations. For example, the store now offers online ordering and delivery services to make shopping more convenient for customers. They require a robust database to help with their transition. Thus they have in listed the services of a team of Database professionals to help with the development of the database.

Despite competition from larger chain supermarkets, the supermarket has remained successful by prioritizing customer service and building strong relationships with its community. The store is committed to continuing to provide quality products and services to its customers for many years to come.

3. Entity relationship diagram



4. Normalisation process

The schema satisfies the requirements for 1NF because each table has a primary key, and each field in a table contains only atomic values. The primary key uniquely identifies each record in the table, and the atomic values in each field ensure that there is no repeating data. For example, the Customer table has a primary key customer_id, which ensures that each customer record in the table is unique. The first_name, last_name, email, and phone fields are all atomic, meaning they contain only a single value.

Second Normal Form (2NF):

The schema satisfies the requirements for 2NF because each non-primary key field in a table is dependent on the primary key. There are no partial dependencies on the primary key. For example, the Order_Item table has a composite primary key that includes both order_id and product_id. The quantity and unit_price fields are both dependent on the composite primary key, which ensures that there are no partial dependencies on the primary key.

Third Normal Form (3NF):
The schema satisfies the requirements for 3NF because there are no transitive dependencies in the tables. That is, a non-primary key field is dependent only on the primary key, and not on any other non-primary key field in the same table. For example, the Product table has a category_id field that is dependent only on the product_id field, and not on any other non-primary key field in the table. This ensures that there is no redundancy or duplication of data in the table.

Fourth Normal Form (4NF):
The schema satisfies the requirements for 4NF because there are no multi-valued dependencies in the tables. That is, a non-primary key field is dependent only on a subset of the primary key, and not on the entire primary key. There are no such dependencies in the given schema.

5. Physical database implementation

Database creation
The TSQL code for the creation of the database of 'Pretoria North Supermarket' is given below:

```
CREATE DATABASE Pretoria_North_SuperMarket
ON PRIMARY
(
    NAME = 'Pretoria_North_SuperMarket',
    FILENAME = 'c:\DBD281_Project\SQLDATA\Pretoria_North_SuperMarket.mdf',
    SIZE = 10MB,
    MAXSIZE = 15MB,
    FILEGROWTH = 10%
)

LOG ON(

NAME = 'LOG_Pretoria_North_SuperMarket',
FILENAME = 'c:\DBD281_Project\LOGFILE\LOG_Pretoria_North_SuperMarket.Ldf',
SIZE = 5MB,
MAXSIZE = 10MB,
FILEGROWTH = 10%

)
```

There were no additional secondary files created for the database, because the database itself does not exceed the maximum size of a single window; thus due to the size of the database, no additional secondary files were required.
The maximum size of the database was estimated in relation to the size of an existing database for a small unicorn shop (Northwind). This database provided a guide for a small business' database size and an additional 10% precision was accommodated for.

Tables' creation:
A list of tables that exist within the database as well as a short description of each is given below. This section will discuss the implementation process of these said tables.

Table Name	Description
Customers	The customers table was created to hold the information of the customers; and contains the following columns: ➢ customer identification number; ➢ first and last name;

	➢ Email and phone number. To describe each customer.
Employee	Lists of employees that work for the supermarket are given in this table. The table has the following columns: ➢ Employee identification number; ➢ First and last name; ➢ Email and phone; ➢ Hire date; and ➢ Position. Each row holds the information of one of the 5 employees.
Category	Each product sold in the supermarket can be categorised in one of the following categories: ➢ Dairy; ➢ Meat; ➢ Produce; ➢ Preserves; ➢ Canned/dried Products; ➢ Bread/Pasta/grains. Each of the above categories has a corresponding category identification number and a description in the category table. Thus the category table has 3 columns namely: ✓ Category identification number; ✓ Name; and ✓ Description.
Orders	The orders table hold the orders' id number corresponding to each customer. This table allows the viewing of the orders id and its corresponding customer id, of the customer who placed the order. This table has three columns: ➢ Order id; ➢ Customer id; and ➢ Order date. Each row has a unique order id for every order placed.
Order-Item	This table hold the details about the order placed. The order-item table acts as a bridge entity between orders and products. Because one order can be made up of multiple products and a product type can be ordered by multiple orders. The columns in this table are: ➢ Order item id; ➢ Order id; ➢ Product id; ➢ Quantity and unit Price. The same order id can appear many times and the same product id can appear multiple times; however only the order item id is unique.
Products	The product table is one of the largest tables in the database; it holds the list of all different the products sold at the supermarket. Each product type has a unique id. The supplier id for each product is also given in this table. The columns present are: ➢ Product id; ➢ Product name; ➢ Description; ➢ Price; ➢ Category id; and ➢ Supplier id.
Suppliers	The supermarket has a total of 10 suppliers that supply the different products sold. This table holds the information of the suppliers. The

	columns present are: ➢ Supplier id, ➢ Name of the supplier, ➢ Email, phone and address, ➢ City and the postal code.
Transactions	The transaction table is the link between the customers, employees and the payment method. It holds the ids of all of these entities and the date of the transaction. It has the following columns: ➢ Transaction id, ➢ Customer id, ➢ Employee id, ➢ Date of the transaction, ➢ Payment method. To access the payment method for each order; the user will have to go through the transactions.
Payment	The different payment methods accepted by the supermarket are given in this table. Each row corresponds to a different payment method. The payment methods that are accepted by the supermarket are: ✓ Cash, ✓ EFT (electronic fund transfer), ✓ Mobile payment, or ✓ Debit and credit card payment. The table is made up of these two columns: ➢ Payment method id; and ➢ Description.
Stock	The stock table holds the stock id, which identifies the stock being dealt with, the product id and the quantity on hand for a specific stock.

Considerations placed when creating the above tables:

The TSQL code for creating the customers, employee and supplier table is given below.

- The user of the database must be able to easily identify which table the identity key being used belongs to. Hence each table has its own sequence of numbers to assist in this identification.

  For example, the customer identification in the customers table starts at 1001; while the employee ids in the employee table start at 20001.

```
use Pretoria_North_SuperMarket;
GO
CREATE TABLE Customers(
    customer_id int Primary key identity(1001,1) not null,
    FirstName nvarchar(255),
    LastName nvarchar(255),
    email nvarchar(255),
    phone nvarchar(255)
)
GO
CREATE TABLE Employee(
    employee_id int Primary key identity(20001,1) not null,
    FirstName nvarchar(40),
    LastName nvarchar(40),
    email nvarchar(100),
    phone nvarchar(12),
    HireDate DATETIME,
    Position char(20)
    )
GO
CREATE TABLE Supplier(
    supplier_id int Primary key identity(1,1) not null,
    Name nvarchar(40),
    email nvarchar(100),
    phone nvarchar(12),
    address nvarchar(70),
    city nvarchar(15),
    PostalCode nvarchar(10)
    )
```

In doing this, the manger will be able to easily identify if they are working with an employee or a customer based on the key at hand. Each table has a set of its own unique key sequence.

- ## Indexes and constraints:
  - **Constraints**

Every table has an identifying key. The constraints placed on this key are primary key, which encompasses a 'NOT NULL' as well as a 'UNIQUE' constraint.
Foreign keys are placed on tables that reference a primary key; the following tables have foreign key constraints:

- Transactions,
- Order;
- Order Item;
- Stock; and
- Product table.

Non-clustered indexes:

The non-clustered indexes are placed on the most frequently used tables for queries.

- The employees will view the customers' information table more frequently in order to retrieve their contact information, so they can notify the customers when their orders are ready. Hence a clustered index was placed on the customers table. The TSQL code used to place this additional constraint on the customers table is given below:

```
use Pretoria_North_SuperMarket;
go
create nonclustered index IX_Customers
on Customers(customer_id);
```

- The order item table will also have a non-clustered index, because the employees will frequently use the order item table in the queries to view the information about a specific order.

- **Clustered**

  The manager would like the ease of viewing transactions that occurred for the specific month; in the order in which they occurred, thus a clustered index was created on the transactions table.
  The clustered index was created when the primary key constraint was added to the transaction table. The primary key of a table is a clustered index. This means that majority of the tables that exist in the Pretoria North database have clustered indexes.

6. Sample Data

The sample data for the database comes from excel files; these files will be attached to the project deliverables. The supermarkets kept its information in excel files over the years. An example of such a file for customers is shown below:

CustomerID	FirstName	LastName	Email	Phone
1001	Charlotte	Kelley	c.kelley@randatmai	855-4497-18
1002	Adam	Armstrong	a.armstrong@randa	914-0326-68
1003	Cherry	Parker	c.parker@randatma	065-2416-32
1004	Edith	Miller	e.miller@randatmai	877-5093-79
1005	Sawyer	Thompson	s.thompson@randa	934-7001-15
1006	Kelsey	Thomas	k.thomas@randatm	924-7377-00
1007	Mary	Brooks	m.brooks@randatm	331-8684-83
1008	Dominik	Hill	d.hill@randatmail.c	033-3169-48
1009	Lana	Harrison	l.harrison@randatm	202-2500-41
1010	Chester	Robinson	c.robinson@randat	395-3133-20
1011	Kate	Hunt	k.hunt@randatmail	039-4925-35
1012	Derek	Spencer	d.spencer@randatm	090-5120-49
1013	Preston	Payne	p.payne@randatma	863-5541-63
1014	Thomas	Armstrong	t.armstrong@randa	892-9629-07
1015	Hailey	Mason	h.mason@randatm	091-1832-00
1016	Emma	Allen	e.allen@randatmail	196-4186-45
1017	Fenton	Owens	f.owens@randatma	307-1395-69
1018	Vanessa	Andrews	v.andrews@randat	504-7794-00
1019	Daryl	Riley	d.riley@randatmail	447-2932-99

In the creation of the database the file growth for both the log and the primary file were explicitly defined. This placed limitations on how the data in the excel files will be inputted into SQL Server management. Had the file growth not been explicitly defined; the 'BULK INSERT' would've been utilised to insert the data into SQL Server.

An alternative method using the SQL Server import and export wizard was used. Using the wizard the

- Data source was selected as excel sheets,
- Data types and the range for each column was stated using Mapping,
- Preview of the data was available before the import was complete, and
- A direct import into the desired table (that respects the constraints placed on the table) was achieved.

The wizard allowed for constraint checking and it will not allow data imports that violate the integrity constraints placed on the tables.

The sample data for the following tables were provided in excel form:

- Customer information,
- Order Information (order item),
- Orders,
- Products,
- Stock,
- Suppliers, and
- Transactions.

7. Entity Identification and explanation

Triggers:

- **`check product stock:`** Trigger to prevent insertion of a new Order_Item record if the product is out of stock
- **`check product orders:`** Trigger to prevent deletion of a Product record if there are still order items associated with the product
- **`update Order date:`** Trigger that updates the Order_date column of the Product table whenever an order is placed
- **`check supplier products:`** Trigger to prevent deletion of a Supplier record if there are still products associated with the supplier

Dynamic stored procedures:

- **`get products by category:`** stored procedure that takes in a category ID parameter and dynamically generates a SELECT statement to retrieve all products belonging to that category
- **`update product price:`** Stored procedure that updates the product price
- **`get order details:`** stored procedure to get full order details
- **`insert customer:`** stored procedure to insert a new customer and return the ID of the newly inserted record
- **`spSearchProductsByCategoryAndPrice:`** dynamic stored procedure that retrieves data from multiple tables based on user input

Table:

- **Category**: table of product categories containing category name, ID and description.
- **Customers**: This table lists the company's customers, it includes their first name, last name, Phone number, email and customer ID.
- **Employee**: contains employee details including first name, last name, emails, hire date, and position.
- **Order Item**: contains details of an order such as the quantity and the price
- **Orders**: contains the details of an order such as the customer ID and order date
- **Payment**: contains payment method id and description, this table contains different types of payment methods
- **Prods**: contains product details like product name, description, price, category ID, and supplier ID
- **Products**: contains product details like product name, description, price, category ID, and supplier ID
- **Stock**: Shows the quantity of a product that the store has on hand, including product ID and Units on hand
- **Supplier**: contains information about suppliers including, supplier name, email, phone, address, city, postal code
- **Transaction**: This table mainly records transaction dates and includes the customer ID, transaction date, employee ID and payment method

8. List of Questions or Queries

1. Due to the impact of the pandemic, the supermarket has to make cuts to its employees. The supermarket manager would like to see the transactions of all its employees and the duration of these transactions. In the hope of letting go its least utilised employee.

2. The Supermarket would like to reward its employees for all the hard work placed in the first quarter of the year. This is a tradition the manager would like to start. The manager has requested the ability to easily see the number of transactions each employee has successfully carried out during the first quarter.

3. The supermarket is thinking of introducing a rewards program for its top customers. The manager has requested the ability to view the total amount spent by each customer. This will help reward its top customers, and maintain customer relations.

4. The lack of a centralised data source has created an inventory problem over the years. The manager would like to be able to view the quantity of products on hand; at any time this will assist in identifying which product require restocking.

5. The manager would like to see the top 3 selling products in the supermarket to ensure that there is always stock easily available for these products.

Stored Procedures

6. With the new online ordering system; the owners have requested the ability to automatically reject an order if the product is out of stock. Which means that the order item must not be inserted of the quantity on hand is zero.

7. Using the new ordering system, it has become important to not lose information about an order, in case it is cancelled or changed. The manager has requested the prevention of the deletion of a product's record if there are orders related to the product.

8. When a customer places an order, the date of which the order was placed must reflect accurately in the database; this is to assist in the retrieval of order information and prioritizing orders if stock of a certain product is low.

9. Currently, there are multiple suppliers that the supermarket uses to order food of the same category; this was an attempt to cut costs as different suppliers charge less; however the owners would like to formalise the supermarket's supply chain. They will be reducing the number of suppliers, however the manager requests the notification of a removal of a supplier before it is formalised.

Dynamic Stored Procedures

10. In the efforts to formalise the supply chain, the manager has requested the ability to enter a category identification number and view all the products in the category. This is to help spread out the category with the smallest products to other categories. So the supermarket can let the said supplier go.

11. Each year inflation causes the product prices to rise; the employees would like the ease of updating the products' prices; by just entering the new inflation rate and producing new updated prices.

12. The online ordering system must be able to show the entire order details to the customer over their phone. The database must assist with retrieving the entire order details for the customers.
13. The granularity level of the database must be able to span across all tables in order to give the most detailed of results. The stored procedure must span multiple tables.

14. The supermarket has seen a rise in the number of new customers. To assist in the managing of the number of new customers, the employees have requested the ability to insert a new customer and receive the customer identification number of the customer. This is ideal for in-store orders; where customers are queuing for their orders, once the order is ready, the customer identification number can just be read out loud and the customer can retrieve their order.