

Case Study Article on **FITFLEX DATABASE** **DESIGN**

A comprehensive Approach

Written By:
Praveen Chaudhary

[LinkedIn Profile](#)

Table of Contents

1. Introduction.....	(Pg. 4)
1.1. Overview of FITFLEX e-commerce store.....	(Pg. 4)
2. Mission and Objective.....	(Pg. 4)
2.1. Mission Statement.....	(Pg. 4)
2.2. Business Objectives.....	(Pg. 5)
3. Database Development.....	(Pg. 5)
3.1. Predefined Entity Tables.....	(Pg. 5)
3.1.1. Core Entity Tables Description.....	(Pg. 6)
3.2. Predefined Fields List.....	(Pg. 7)
4. List of Attributes.....	(Pg. 8-10)
4.1. Customer Table.....	(Pg. 8)
4.2. Employee Table.....	(Pg. 8)
4.3. Product Table.....	(Pg. 9)
4.4. Sales Order Table.....	(Pg. 9)
4.5. Store Table.....	(Pg. 9)
4.6. Product Category Table.....	(Pg. 10)
4.7. Order Status Table.....	(Pg. 10)
4.8. Payment Mode Table.....	(Pg. 10)
5. Business Rules	(Pg.11-12)
5.1. Customer Data Management.....	(Pg.11)
5.2. Employee Data Management.....	(Pg.11)
5.3. Product Inventory.....	(Pg.11)
5.4. Order Processing.....	(Pg.12)
5.5. Store Operations.....	(Pg.12)
5.6. Data Integrity and Consistency.....	(Pg.12)
6. Key Challenges and Solutions.....	(Pg.13)
6.1. Stockouts and Overstocking.....	(Pg.13)
6.2. Centralized Data Management.....	(Pg.13)
6.3. Enhancing Record Accuracy.....	(Pg.13)
6.4. Data Driven Decision Making.....	(Pg.13)

[LinkedIn Profile](#)

7. ENTITY – RELATIONSHIP (ER) DIAGRAM.....	(Pg. 14)
7.1. Overview	(Pg. 14)
7.2. Entity – Relationship Diagram.....	(Pg. 14)
8. Relationships and Database Structure.....	(Pg. 15)
8.1. One-to-Many Relationships.....	(Pg. 15)
8.2. Many-to-Many Relationships.....	(Pg. 15)
9. Conclusion.....	(Pg. 15)
10. Appendix.....	(Pg. 16)
10.1. Database Creation.....	(Pg.16-17)
10.2. Key Features of the Database Design.....	(Pg. 17)
10.3. Data Dictionary.....	(Pg.18-20)
10.3.1.Customer Table.....	(Pg. 18)
10.3.2.Employee Table.....	(Pg. 18)
10.3.3.Product Table.....	(Pg.19)
10.3.4. Sales Order.....	(Pg.19)
10.3.5. Store Table.....	(Pg.19)
10.3.6.Product Category Table.....	(Pg.20)
10.3.7.Order Status Table.....	(Pg.20)
10.3.8.Payment Mode Table.....	(Pg.20)
10.4. Testing Database (Views and Operational Queries)	(Pg.21-23)
10.4.1.(View-1) – Order Details/Customer Receipt.....	(Pg.21)
10.4.2.(Query-1) - Order Details/Customer Receipt.....	(Pg.21)
10.4.3.(View-2) – Hot-Selling Products.....	(Pg.22)
10.4.4.(Query-2) – Hot-Selling Products.....	(Pg.22)
10.4.5.(View-3) – Best Performing Employee.....	(Pg.23)
10.4.6.(Query-3) - Best Performing Employee.....	(Pg.23)

[LinkedIn Profile](#)

1.INTRODUCTION

1.1

Fitflex is an e-commerce store specializing in selling fitness-related products such as gym equipment, activewear, supplements, and accessories. The platform serves as a one-stop shop for customers, offering a seamless shopping experience with features like personalized recommendations, order tracking, and customer support.

2. Mission and Objectives

2.1 Mission Statement

The primary mission of FitFlex's database system is to enable seamless data management, ensuring accurate record-keeping, improving business intelligence, and facilitating strategic planning based on real-time and historical data.

A well-structured database will support informed decision-making, leading to enhanced customer experience and optimized business operations.

[LinkedIn Profile](#)

2.2 Business Objectives

- Design and implement a normalized database schema.
- Generate reports and dashboards to support business decisions.
- Ensure reliable, structured, and secure storage of customer, product, and transaction data.

3. Database Development

Effective database design is crucial for maintaining organized and user-friendly data management. It ensures smooth operations, such as tracking customer and order information, while minimizing errors. This is essential for businesses to function efficiently and achieve their goals.

3.1 Predefined Entity Tables

Based on the Fitflex e-commerce store operations, these core tables were identified for the database as mentioned below.

3.11 The FitFlex database consists of eight core tables:

- **Customer Table** - Stores customer details, including name, contact information, address, and postal code.
- **Employee Table** - Records employee details such as position, store assignment, and contact information.
- **Product Table** - Manages product details including category, price, and available stock.

[LinkedIn Profile](#)

- **Product Category Table** - Defines categories of products with attributes CategoryID and CategoryName.
- **Sales Order Table** - Tracks customer orders, including product purchases, order date, and store information.
- **Store Table** - Maintains store locations and associated employees.
- **Order Status Table** - Provides order status descriptions with StatusID and OrderStatus.
- **Payment Mode Table** - Details payment methods with attributes PaymentID and Payment Type.

Predefined Tables

- **Product**
- **Customer**
- **Employee**
- **Sales Order**
- **Product Category**
- **Order Status**
- **Payment Mode**
- **Store**

[LinkedIn Profile](#)

3.2 Predefined Field lists

Predefined Field List

- ProductID (PK)
 - Product_Name
 - CategoryID
 - Product_Category
 - Price
 - Stock Quantity
 - CustomerID
 - First_Name
 - Last_Name
 - E-mail
 - Phone
 - Address
 - Employee Job_Role
 - StoreID
 - OrderID
 - Order_Date
 - StatusID
 - Order_Status
 - PaymentID
 - Payment_Mode
- } *'Same for Employee and Customer Entity Tables'*

4. **List of Attributes**

4.1 **Customer Table**

- **Customer ID** – A unique identifier assigned to each customer in the FitFlex database, serving as the primary key to facilitate data retrieval and management.
- **Customer Name** – The full name of the customer, aiding in personalized interactions and user identification.
- **Customer Number** – The customer's contact phone number, which may include a country code and is used for communication purposes.
- **Customer Address** – The complete residential address, including street, city, and state, utilized for billing, deliveries, and verification.
- **Customer Zipcode** – The postal code associated with the customer's address, assisting in regional classification and accurate address identification.

4.2 **Employee Table**

- **Employee ID** – A unique identifier assigned to each employee, functioning as the primary key to ensure distinct employee records.
- **Employee Name** – The full name of the employee, used for identification and internal communication.
- **Employee Position** – The job title or designation of the employee within FitFlex, helping to classify roles and responsibilities.
- **Employee Email** – The professional email address used for internal communication and system authentication.
- **Employee StoreID** – A unique identifier linking the employee to a specific store location, typically serving as a foreign key referencing the Store Table.

4.3 **Product Table**

- Product ID – A unique identifier assigned to each product to maintain distinct records in the inventory system.
- Product Name – The official name of the product, providing a clear description.
- Product Category – The classification or type of product, helping to group similar items for better inventory management.
- Product Price – The cost of the product, stored in the currency used by FitFlex.
- Product Stock Quantity – The number of available product units in inventory, essential for stock management.

4.4 **Sales Order Table**

- Order ID – A unique identifier assigned to each order, ensuring efficient order tracking.
- Customer ID – A reference to the unique identifier of the customer who placed the order, establishing a relationship between customers and orders.
- Product ID – A reference to the unique identifier of the purchased product, linking orders to specific items.
- Order Date – The date when the order was placed, helping track order history and timelines.
- Store ID – A reference to the unique identifier of the store where the order was processed, establishing a connection between orders and store locations.

4.5 **Store Table**

- Store ID – A unique identifier assigned to each FitFlex store, ensuring distinct store records.
- Store Name – The official name of the store, used for identification purposes.
- Store Location – The physical address or geographical area where the store operates.

[LinkedIn Profile](#)

4.6 **Product Category Table**

- Category ID – A unique identifier assigned to each product category, serving as the primary key for categorization.
- Category Name – The name of the product category, describing the type or classification of products.

4.7 **Order Status Table**

- Status ID – A unique identifier assigned to each order status, serving as the primary key.
- Order Status – A detailed explanation of what the status represents, providing clarity for users.

4.8 **Payment Mode Table**

- Payment ID – A unique identifier for each payment mode, acting as the primary key.
- Payment Type – The name of the payment method, such as "Credit Card," "Debit Card," "Cash," or "Online Wallet."

5. Business Rules

5.1. Customer Data Management

- Each customer must have a unique identifier to ensure distinct customer records.
- Customer details should include essential information such as full name, contact details, residential address, and zip code.
- Contact details must contain either a valid phone number or an email address for communication and verification purposes.

5.2. Employee Data Management

- Every employee must be assigned a unique identifier to distinguish individual records within the database.
- Employee records should include key attributes such as full name, job position, email address, store ID, and residential address.
- Each employee must have a valid email address and an associated store ID to ensure proper role assignment and access control.

5.3. Product Inventory

- Each product must have a unique identifier to maintain accurate inventory records and prevent duplication.
- Product information should encompass details such as product name, category, pricing, and available stock quantity.
- Every product entry must be categorized appropriately and include a clearly defined product name.

[LinkedIn Profile](#)

5.4. Order Processing

- Every order must be assigned a unique identifier to facilitate efficient tracking and management.
- Order records should contain critical details such as customer ID, product ID, order date, and store ID for accurate transaction processing.
- A valid customer ID, product ID, and store ID must be linked to every order to maintain data integrity.

5.5. Store Operations

- Each store must have a unique identifier to differentiate its records and locations.
- Store records should include key information such as the store's name and geographical location.

Beyond the standard entity-specific rules for customers, employees, products, orders, and stores, the Fitflex database requires additional business rules to ensure efficient data management, system integrity, and operational excellence. These rules focus on overall database design, security, performance, and decision-making.

5.6. Data Integrity and Consistency

- All foreign key relationships must be enforced using constraints to maintain referential integrity.
- Data should be normalized up to at least the third normal form (3NF) to minimize redundancy and improve efficiency.
- Historical data (such as old order records) must be archived periodically to optimize database performance.

[LinkedIn Profile](#)

6. Key Challenges and Solutions

6.1 Stockouts and Overstocking

Managing inventory is a significant challenge for FitFlex. Stockouts lead to lost sales, while overstocking increases operational costs. The database system ensures real-time inventory tracking, allowing for demand forecasting and reducing inefficiencies in stock management.

6.2 Centralized Data Management

FitFlex aims to establish a centralized database to consolidate all business operations, including inventory, customer data, and sales. This reduces redundancy, enhances data accuracy, and streamlines business workflows.

6.3 Enhancing Record Accuracy

Manual data entry often results in errors, impacting decision-making. Implementing an automated database minimizes human errors, ensuring data accuracy and reliability.

6.4 Data-Driven Decision-Making

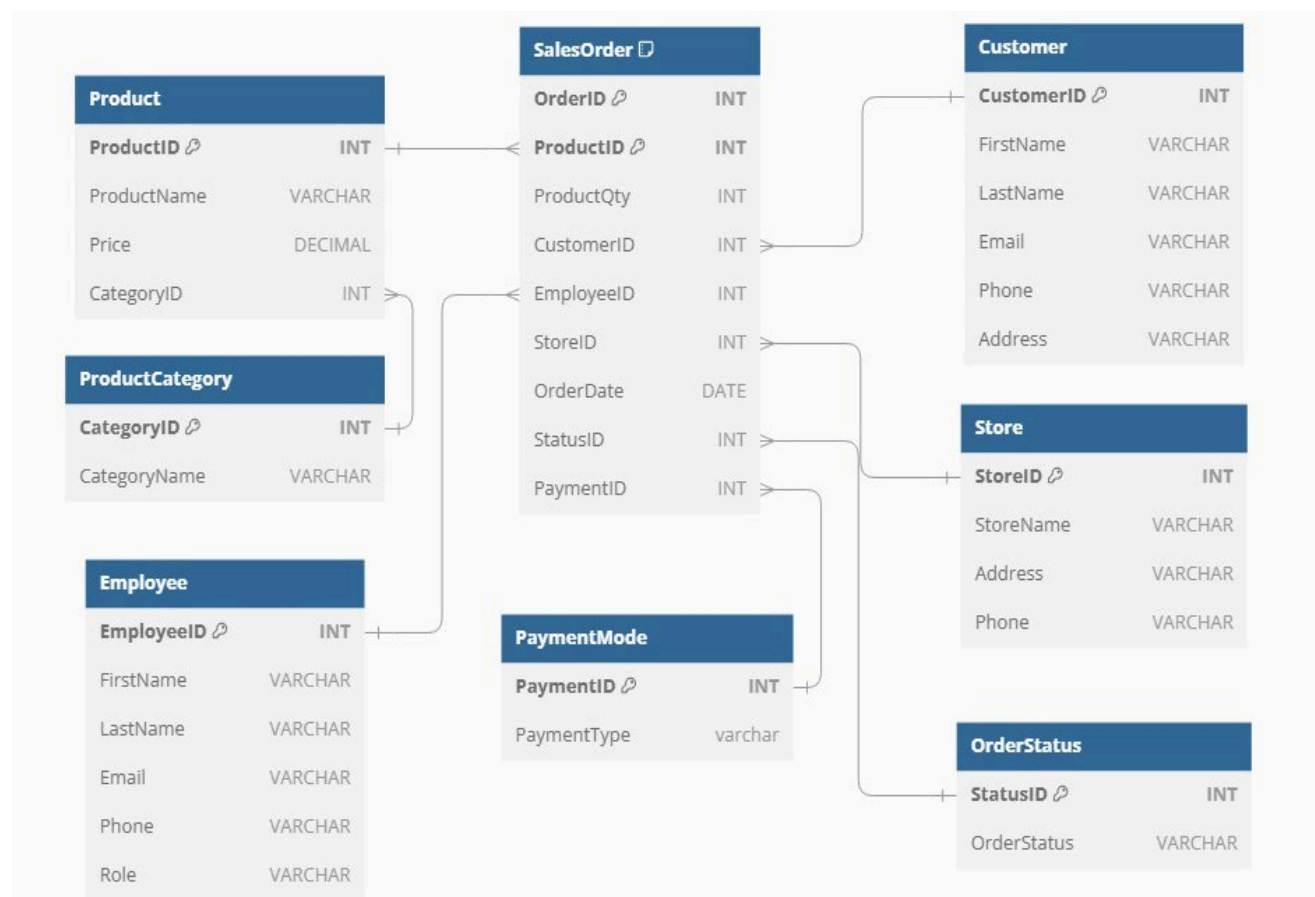
By integrating real-time feedback and historical data analysis, FitFlex can make strategic business decisions, optimize pricing, improve customer engagement, and predict market trends.

7. ENTITY – RELATIONSHIP (ER) DIAGRAM

7.1 Overview of the (ER)- Diagram

The ERD provides a visual representation of the database structure, showing how different tables are related and represented for a retail or sales management system. It includes several interconnected entities as mentioned in the predefined list of entity tables to manage various aspects of the business:

7.2 Entity – Relationship (ER) Diagram



[LinkedIn Profile](#)

8. Relationships and Database Structure

8.1 One-to-Many Relationships:

- ✓ **Customers** → **SalesOrders** (One customer can place multiple orders)
- ✓ **SalesOrders** → **Products** (One order can have multiple products)
- ✓ **SalesOrders** → **Store** (One order is placed at a single store, but a store can process multiple orders)
- ✓ **ProductCategory** → **Products** (One category can have multiple products)
- ✓ **PaymentMode** → **SalesOrders** (One payment method can be used in multiple orders)
- ✓ **Store** → **SalesOrders** (One store can have multiple orders)
- ✓ **OrderStatus** → **SalesOrders** (One order status can be assigned to multiple orders)

8.2 Many-to-Many Relationship & Solution:

- ✓ **Products & SalesOrders** have a many-to-many relationship (one order can contain multiple products, and a product can appear in multiple orders).
- ✓ To handle this, we introduce a **SalesOrders-Products** linking table between **SalesOrders** and **Products**.

9. “Conclusion”:

The FitFlex database is a robust, relational system designed to efficiently manage key business operations like customer management, order processing, and product tracking. With modular tables and unique identifiers, it ensures data accuracy, minimizes redundancy, and supports detailed reporting for scalable growth and operational optimization.

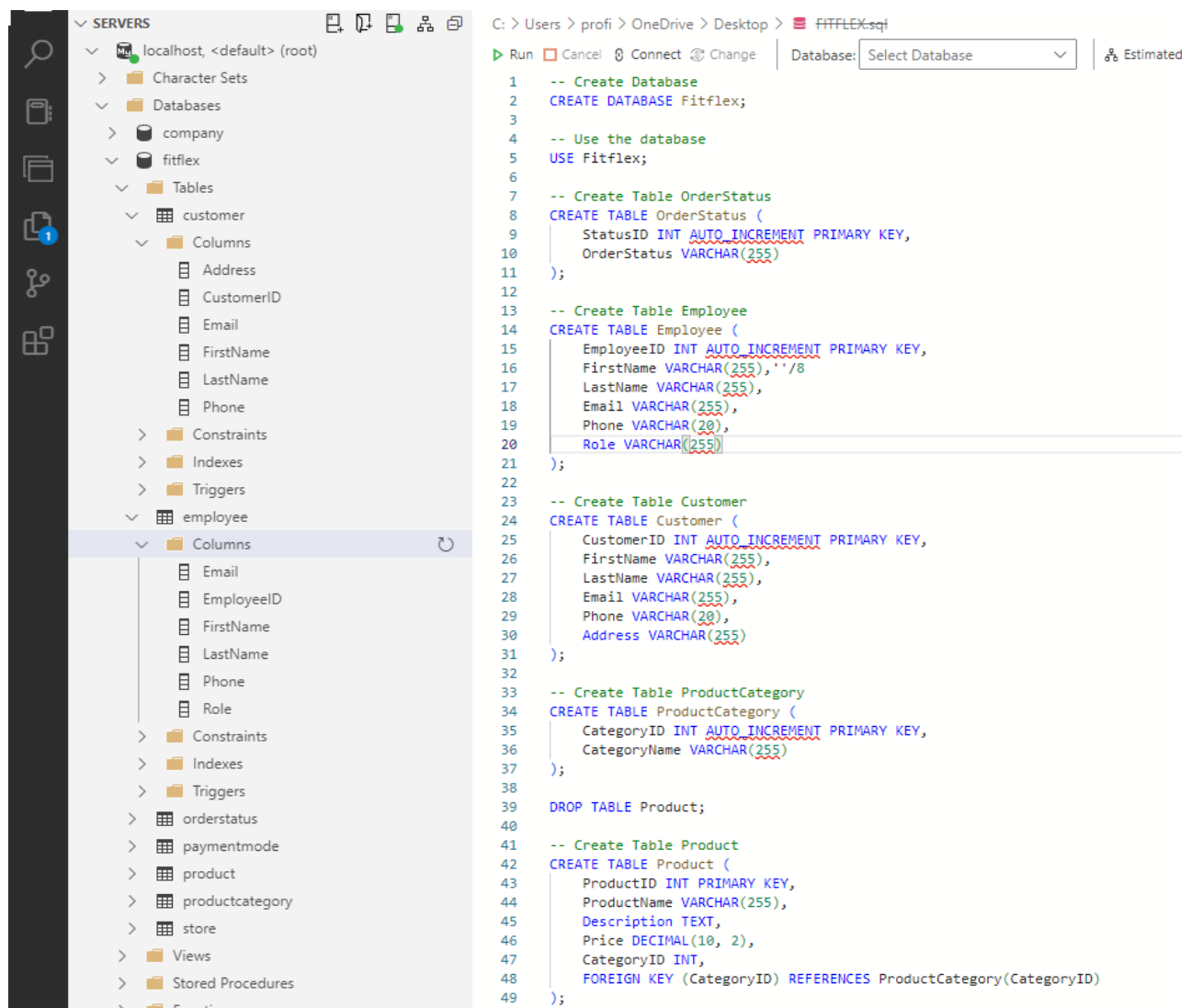
[LinkedIn Profile](#)

10. Appendix

10.1 DATABASE CREATION

Content:

- The goal is to design a database for managing customer orders, employees, products, and transactions efficiently.
- The database is structured to maintain data integrity, ensure scalability, and optimize querying.

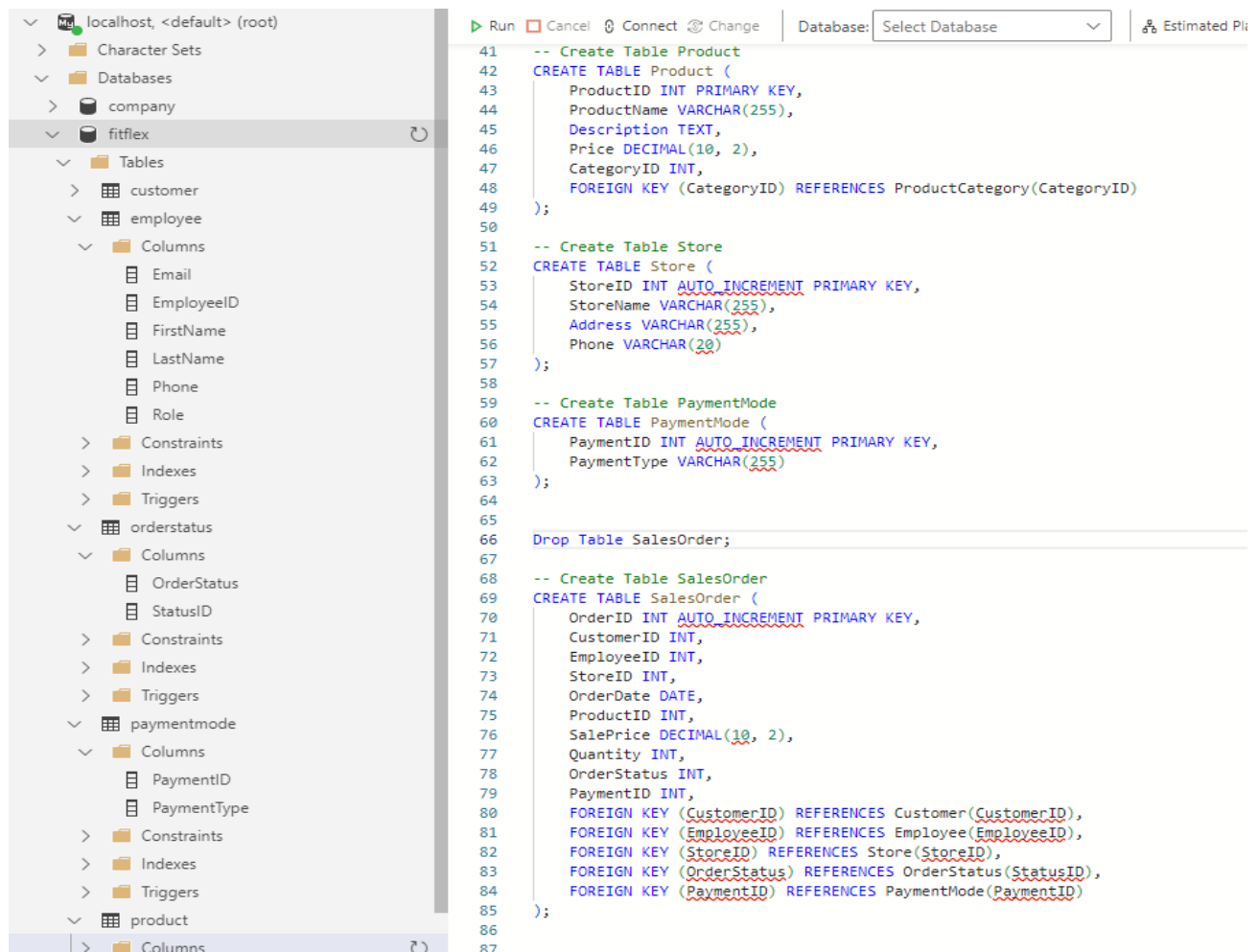


The screenshot displays the SQL Server Enterprise Manager (left pane) and the SQL Server Enterprise Edition (right pane). The left pane shows the server hierarchy: **SERVERS** > **localhost, <default> (root)** > **Databases** > **fitflex** > **Tables** > **customer** > **Columns**. The right pane shows the SQL script for creating the database and tables.

```
C: > Users > profi > OneDrive > Desktop > FITFLEX.sql
Run Cancel Connect Change Database: Select Database Estimated

1  -- Create Database
2  CREATE DATABASE Fitflex;
3
4  -- Use the database
5  USE Fitflex;
6
7  -- Create Table OrderStatus
8  CREATE TABLE OrderStatus (
9      StatusID INT AUTO_INCREMENT PRIMARY KEY,
10     OrderStatus VARCHAR(255)
11 );
12
13 -- Create Table Employee
14 CREATE TABLE Employee (
15     EmployeeID INT AUTO_INCREMENT PRIMARY KEY,
16     FirstName VARCHAR(255),
17     LastName VARCHAR(255),
18     Email VARCHAR(255),
19     Phone VARCHAR(20),
20     Role VARCHAR(255)
21 );
22
23 -- Create Table Customer
24 CREATE TABLE Customer (
25     CustomerID INT AUTO_INCREMENT PRIMARY KEY,
26     FirstName VARCHAR(255),
27     LastName VARCHAR(255),
28     Email VARCHAR(255),
29     Phone VARCHAR(20),
30     Address VARCHAR(255)
31 );
32
33 -- Create Table ProductCategory
34 CREATE TABLE ProductCategory (
35     CategoryID INT AUTO_INCREMENT PRIMARY KEY,
36     CategoryName VARCHAR(255)
37 );
38
39 DROP TABLE Product;
40
41 -- Create Table Product
42 CREATE TABLE Product (
43     ProductID INT PRIMARY KEY,
44     ProductName VARCHAR(255),
45     Description TEXT,
46     Price DECIMAL(10, 2),
47     CategoryID INT,
48     FOREIGN KEY (CategoryID) REFERENCES ProductCategory(CategoryID)
49 );
50
```

[LinkedIn Profile](#)



10.2 Key Features of the Design

- **Normalization:**
 - Data is split across multiple tables to eliminate redundancy.
 - Ensures efficient storage and easier maintenance.
- **Relationships:**
 - Foreign keys enforce data integrity between tables.
- **Primary Keys:**
 - Each table uses unique identifiers like ID columns for efficient querying.
- **Scalability:**
 - The design allows for the addition of new products, categories, and order statuses without altering the structure.

[LinkedIn Profile](#)

10.3 Data Dictionary

Each entity table contains a list of attributes or fields that store essential information for smooth and efficient operations of the database structure.

10.3.1 Customers Table

Field Name	Data Type	Description
CustomerID	INT (PK)	Unique identifier for each customer.
First_Name	VARCHAR	First name of the customer.
Last_Name	VARCHAR	Last name of the customer.
E-mail	VARCHAR	Email address of the customer.
Address	VARCHAR	Residential address of the customer.
Contact No.	INT	Contact number of the customer.

10.3.2 Employees Table

Field Name	Data Type	Description
EmployeeID	INT (PK)	Unique identifier for each employee.
First_Name	VARCHAR	First name of the employee.
Last_Name	VARCHAR	Last name of the employee.
E-mail	VARCHAR	Email address of the employee.
Job_Role	VARCHAR	Employee's job role (e.g., Manager...)
Contact No.	INT	Contact number of the employee.

[LinkedIn Profile](#)

10.3.3 Product Table

Field Name	Data Type	Description
ProductID	INT (PK)	Unique identifier for each product.
Product_Name	VARCHAR	Name of the product.
Price	DECIMAL	Price of the product.
CategoryID	INT (FK)	Unique Identifier for each category.

10.3.4 Sales Order Table

Field Name	Data Type	Description
OrderID	INT (CPK)	Unique identifier for each employee.
CustomerID	INT (FK)	Associated CustomerID
EmployeeID	INT (FK)	Associated EmployeeID
StoreID	INT (FK)	Associated StoreID
Order_Date	DATE	Date associated with the sales order.
ProductID	INT (CPK)	Associated ProductID
Product Quantity	INT	Quantity associated with the sale.
StatusID	INT (FK)	Associated Order StatusID
PaymentID	INT (FK)	Associated PaymentID

10.3.5 Store Table

Field Name	Data Type	Description
StoreID	INT (PK)	Unique identifier for each store.
Store_Name	VARCHAR	Name of the store.
Address	VARCHAR	Address of the Store
Contact_No	INT	Contact number of the store.

[LinkedIn Profile](#)

10.3.6 Product Category Table

Field Name	Data Type	Description
CategoryID	INT (PK)	Unique identifier for each category
Category_Name	VARCHAR	Associated Category name for each product

10.3.7 Order Status Table

Field Name	Data Type	Description
StatusID	INT (PK)	Unique identifier for Order Status
Order_Status	VARCHAR	Status of the Order

10.3.8 Payment Mode Table

Field Name	Data Type	Description
PaymentID	INT (PK)	Unique identifier for each payment.
Payment_Type	VARCHAR	Mode of Payment associated.

[LinkedIn Profile](#)

10.4 Testing Database (Views and Operational Queries)

10.4.1 (View 1) – Order Details / Customer Receipt

Purpose: want to show the customer the details of their order

Tables used: SalesOrder, Customer, Employee, Product, Store, OrderStatus, PaymentMode

View Name: OrderDetails

Fields Used: CustomerName, EmployeeName, StoreName, OrderID, OrderStatus, PaymentType, ProductName, ProductQty, OrderTotal

Calculated Fields: ProductTotal, OrderTotal, Customer Name, EmployeeName

10.4.2 (Query-1) - Order Details / Customer Receipt

```
352 CREATE VIEW OrderDetails AS
353 SELECT
354     so.OrderID,
355     CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName,
356     CONCAT(e.FirstName, ' ', e.LastName) AS EmployeeName,
357     s.StoreName,
358     os.OrderStatus,
359     pm.PaymentType,
360     p.ProductName,
361     so.ProductQty,
362     (p.Price * so.ProductQty) AS ProductTotal,
363     (SELECT SUM(p2.Price * so2.ProductQty)
364      FROM SalesOrder so2
365      JOIN Product p2 ON so2.ProductID = p2.ProductID
366      WHERE so2.OrderID = so.OrderID) AS OrderTotal
367 FROM
368     SalesOrder so
369 JOIN
370     Customer c ON so.CustomerID = c.CustomerID
371 JOIN
372     Employee e ON so.EmployeeID = e.EmployeeID
373 JOIN
374     Store s ON so.StoreID = s.StoreID
375 JOIN
376     OrderStatus os ON so.StatusID = os.StatusID
377 JOIN
378     PaymentMode pm ON so.PaymentID = pm.PaymentID
379 JOIN
380     Product p ON so.ProductID = p.ProductID;
381
382 SELECT * FROM OrderDetails WHERE OrderID = 2;
```

Results		Messages								
	OrderID	CustomerName	EmployeeName	StoreName	OrderStatus	PaymentType	ProductName	ProductQty	ProductTotal	OrderTotal
1	2	Dwight Schrute	Jane Smith	Uptown Fitness	Completed	Credit Card	Resistance Bands	3	59.97	2609.76
2	2	Dwight Schrute	Jane Smith	Uptown Fitness	Completed	Credit Card	Running Shoes	7	629.93	2609.76
3	2	Dwight Schrute	Jane Smith	Uptown Fitness	Completed	Credit Card	Hiking Backpack	9	1169.91	2609.76
4	2	Dwight Schrute	Jane Smith	Uptown Fitness	Completed	Credit Card	Fitness Tracker	5	749.95	2609.76

[LinkedIn Profile](#)

10.4.3 (View-2) – Top Hot-Selling Products

Purpose: Identify the top 5 selling products

Tables used: SalesOrder, Product

View Name: Hot Selling Products

Fields Used: ProductID, ProductName

Calculated Fields: TotalQuantitySold, TotalSales

10.4.4 (Query-2) – Top Hot-Selling Products

```
384 CREATE VIEW HotProducts AS
385 SELECT
386     p.ProductID,
387     p.ProductName,
388     SUM(so.ProductQty) AS TotalQuantitySold,
389     SUM(so.ProductQty * p.Price) AS TotalSales
390 FROM
391     SalesOrder so
392 JOIN
393     Product p ON so.ProductID = p.ProductID
394 GROUP BY
395     p.ProductID, p.ProductName
396 ORDER BY
397     TotalSales DESC
398 LIMIT 5;
399
400 SELECT * FROM HotProducts
```

Results Messages

	ProductID	ProductName	TotalQuantitySold	TotalSales
1	1	Treadmill	28	27999.72
2	10	Elliptical Machine	15	8999.85
3	7	Hiking Backpack	28	3639.72
4	8	Fitness Tracker	21	3149.79
5	2	Dumbbell Set	15	2999.85

[LinkedIn Profile](#)

10.4.5 (View-3) – Best Performing Employee

Purpose: Identify the top 5 performing employees

Tables used: SalesOrder, Product, Employee

Fields Used: EmployeeID, eFirstName, eLastName

Calculated Fields: EmployeeName, TotalSales

10.4.6 (Query-3) - Best Performing Employee

```
402  SELECT
403      e.EmployeeID,
404      CONCAT(e.FirstName, ' ', e.LastName) AS EmployeeName,
405      SUM(so.ProductQty * p.Price) AS TotalSales
406  FROM
407      SalesOrder so
408  JOIN
409      Employee e ON so.EmployeeID = e.EmployeeID
410  JOIN
411      Product p ON so.ProductID = p.ProductID
412  GROUP BY
413      e.EmployeeID, EmployeeName
414  ORDER BY
415      TotalSales DESC
416  LIMIT 5;
```

Results Messages

	EmployeeID	EmployeeName	TotalSales
1	6	David Brown	13289.71
2	3	Mike Taylor	10319.78
3	8	Linda Garcia	8599.75
4	1	John Doe	5819.83
5	5	Sarah Lee	4489.71

[LinkedIn Profile](#)