Text

Description automatically generated with medium confidence

**DBMS PROJECT**

FOXCORE RETAIL

DESIGNING A DATABASE

**MAX 506 – DATABASE DESIGN AND SQL**

**GROUP 1**

**ADARSH SABRAD**

**ADITYARAJ SHARMA**

**KARAN RANA**

**PRAVALIKA DASARI**

**Table of Content**

1. ABSTRACT

1.1 COMPANY BACKGROUND

1.2 BUSINESS PROBLEM

1.3 GOAL

1. ER DIAGRAM
2. DB RELATIONAL SCHEMA
3. NORMALIZATION
4. DATABASE CREATION

5.1 DDL - SQL COMMANDS

5.1.1 VENUE

5.1.2 EVENT

5.1.3 BOOTH

5.1.4 EVENT\_TYPE

5.1.5 SHIFT\_MASTER

5.1.6 SHIFT

5.1.7 SALESPERSON

5.1.8 SALES

5.1.9 PRODUCT

6. DML

6.1 INSERT

6.2 UPDATE & DELETE

7. REPORTS

8. CONCLUSION

**Abstract**

**1.1 Company Background**

Foxcore Retail was founded by Liam Corrigan and Mitchell Fox after they graduated from the University of Western Ontario. The company started as a small retail business selling inexpensive novelty items at music festivals and trade shows. Inspired by a hand-held electric toy and a cooling towel that stayed cool for hours, the two entrepreneurs named their business Foxcore Retail, using a combination of their last names.

As sales grew, the partners hired salespeople to help them handle high demand, paying commissions to incentivize performance. They also tested a variety of new products tailored to specific kinds of shows. By its second year, Foxcore had expanded and was managing up to three shows per weekend, with multiple booths at some venues.

**1.2 Business Problem**

The business problem faced by Corrigan and Fox of Foxcore Retail is the inefficiency and inaccuracy of their manual sales-tracking system. The partners had been using a paper-based system to record sales, which led to errors, lost commissions, and unreliable inventory estimates. As the business grew, the partners realized that they needed to implement a more disciplined database system to collect and analyze valuable data that could provide insight into the performance of employees, events, and products.

The inability to access and analyze the information resulted in sub-optimal decision-making for both short- and long-term planning. The partners recognized that they needed to track exact details about which products were sold, by whom, when, and where. The business problem was to find a solution that would allow them to efficiently and accurately track sales data to improve their operations and decision-making.

**1.3 Goal**

To address the business problem of inefficiency and inaccuracy in Foxcore Retail's manual sales-tracking system, a custom relational database system will be designed. This system will enable the efficient tracking of events, sales consultants, and individual sales of various products. By accessing and analyzing this database, valuable data can be collected to provide strategic insights that will improve decision-making for both short- and long-term planning. With this database, Foxcore Retail can accurately track their sales, identify which products are selling well, and optimize their operations accordingly.

**2. ER Diagram**

**Diagram

Description automatically generated**

**The relationships that we defined from the ER Diagram are as follows:**

**VENUE --- EVENT**

A venue can host many events, but each event can only take place at one venue.

**EVENT --- EVENT\_TYPE**

An event can have only one event type, but each event type can be assigned to multiple events.

**EVENT --- BOOTH**

Each event can have multiple booths, but each booth can belong to only one event.

**BOOTH --- SHIFT**

Each booth can have multiple shifts, but each shift belongs to a booth.

**SHIFT --- SALESPERSON**

A shift can contain multiple salespersons, but each salesperson can have zero to multiple shifts.

**SALESPERSON --- SALES**

Each sale is made by one salesperson, but each salesperson can make multiple sales.

**SALES --- PRODUCT**

Each sale is associated with only one product, but each product can be associated with multiple sales.

**3. Foxcore Retail DB Relational Schema**

**Diagram

Description automatically generated**

**4. Normalization**

A picture containing table

Description automatically generated

**5.** **DATABASE CREATION**: **DDL-SQL Command**

|  |  |  |
| --- | --- | --- |
| **Entity** | **Attributes** | **Datatype** |
| VENUE | **Venue\_ID (PK)** | VARCHAR(10) |
| Venue\_Name | VARCHAR(50) |
| Venue\_Address | VARCHAR(100) |
| Venue\_Description | VARCHAR(200) |
| EVENT | **EVENT\_ID (PK)** | VARCHAR(10) |
| Event\_Name | VARCHAR(50) |
| Start\_Date | DATE |
| End\_Date | DATE |
| Event\_Description | VARCHAR(200) |
| BOOTH | **Booth\_ID (PK)** | VARCHAR(10) |
| Booth\_Location | VARCHAR(200) |
| EVENT\_TYPE | **Event\_Type\_ID (PK)** | VARCHAR(10) |
| Event\_Type\_Name | VARCHAR(50) |
| SHIFT\_MASTER | **Master\_ID (PK)** | VARCHAR(10) |
| Start\_DateTime | DATETIME |
| End\_DateTime | DATETIME |
| SHIFT | **Shift\_ID (PK)** | VARCHAR(10) |
| SALESPERSON | Salesperson\_ID | VARCHAR(10) |
| First\_Name | VARCHAR(50) |
| Last\_Name | VARCHAR(50) |
| Address | VARCHAR(200) |
| Phone | VARCHAR(20) |
| Sales\_Incentive | DECIMAL(5,2) |
| SALES | **Sales\_ID (PK)** | VARCHAR(10) |
| Quantity | INTEGER |
| Final\_Price | DECIMAL(10,2) |
| PRODUCT | **Product\_ID (PK)** | VARCHAR(10) |
| Product\_Name | VARCHAR(50) |
| Wholesale\_Cost | DECIMAL(10,2) |
| Maximum\_SellingPrice | DECIMAL(10,2) |
| Minimum\_SellingPrice | DECIMAL(10,2) |
| FOC | BIT |

CREATE SCHEMA [FoxcoreDatabase | AUTHORIZATION FoxcoreOwner]

**VENUE**

CREATE TABLE VENUE (

    Venue\_ID VARCHAR(10) NOT NULL,

    Venue\_Name VARCHAR(50) NOT NULL,

    Venue\_Address VARCHAR(100) NOT NULL,

    Venue\_Description VARCHAR(200),

    PRIMARY KEY (Venue\_ID)

);

**EVENT**

CREATE TABLE EVENT (

    Event\_ID VARCHAR(10) NOT NULL ,

    Event\_Name VARCHAR(50) NOT NULL,

    Start\_Date DATE NOT NULL,

    End\_Date DATE NOT NULL,

    Event\_Description VARCHAR(200),

    Event\_Type\_ID VARCHAR(10) NOT NULL,

    Venue\_ID VARCHAR(10) NOT NULL,

    PRIMARY KEY (Event\_ID),

    FOREIGN KEY (Event\_Type\_ID) REFERENCES EVENT\_TYPE(Event\_Type\_ID),

    FOREIGN KEY (Venue\_ID) REFERENCES VENUE(Venue\_ID)

);

**BOOTH**

CREATE TABLE BOOTH (

    Booth\_ID VARCHAR(10) NOT NULL,

    Booth\_Location VARCHAR(20) NOT NULL,

    Event\_ID VARCHAR(10) NOT NULL,

    PRIMARY KEY (Booth\_ID),

    FOREIGN KEY (Event\_ID) REFERENCES EVENT(Event\_ID)

);

**EVENT TYPE**

CREATE TABLE EVENT\_TYPE (

    Event\_Type\_ID VARCHAR(10) NOT NULL,

    Event\_Type\_Name VARCHAR(50) NOT NULL,

    PRIMARY KEY (Event\_Type\_ID)

);

**SHIFT MASTER**

CREATE TABLE SHIFT\_MASTER (

    Master\_ID VARCHAR(10) NOT NULL,

    Booth\_ID VARCHAR(10) NOT NULL,

    Start\_DateTime DATETIME NOT NULL,

    End\_DateTime DATETIME NOT NULL,

    PRIMARY KEY (Master\_ID),

    FOREIGN KEY (Booth\_ID) REFERENCES BOOTH(Booth\_ID)

);

**SHIFT**

CREATE TABLE SHIFT (

    Shift\_ID VARCHAR(10) NOT NULL,

    Master\_ID VARCHAR(10) NOT NULL,

    Salesperson\_ID VARCHAR(10) NOT NULL,

    PRIMARY KEY (Shift\_ID),

    FOREIGN KEY (Salesperson\_ID) REFERENCES SALESPERSON(Salesperson\_ID),

    FOREIGN KEY (Master\_ID) REFERENCES SHIFT\_MASTER(Master\_ID)

);

**SALESPERSON**

CREATE TABLE SALESPERSON (

    Salesperson\_ID VARCHAR(10) NOT NULL,

    First\_Name VARCHAR(50) NOT NULL,

    Last\_Name VARCHAR(50) NOT NULL,

    Address VARCHAR(200) NOT NULL,

    Phone VARCHAR(20) NOT NULL

CHECK (Phone LIKE ‘\_\_\_-\_\_\_\_’),

    Sales\_Incentive DECIMAL(5,2) ,

    CHECK (Sales\_Incentive BETWEEN 0 AND 100),

    PRIMARY KEY (Salesperson\_ID)

);

**SALES**

CREATE TABLE SALES (

    Sales\_ID VARCHAR(10) NOT NULL,

    Quantity INTEGER NOT NULL,

    Final\_Price DECIMAL(10, 2) NOT NULL,

    Product\_ID INTEGER NOT NULL,

    Salesperson\_ID INTEGER NOT NULL,

    PRIMARY KEY (Sales\_ID),

    FOREIGN KEY (Product\_ID) REFERENCES PRODUCT(Product\_ID),

    FOREIGN KEY (Salesperson\_ID) REFERENCES SALESPERSON(Salesperson\_ID)

);

**PRODUCT**

CREATE TABLE PRODUCT (

    Product\_ID VARCHAR(10) NOT NULL,

    Product\_Name VARCHAR(50) NOT NULL,

    Wholesale\_Cost DECIMAL(10, 2) NOT NULL,

    Maximum\_SellingPrice DECIMAL(10, 2) NOT NULL,

    Minimum\_SellingPrice DECIMAL(10, 2) NOT NULL,

    FOC BIT NOT NULL Default 0,

    PRIMARY KEY (Product\_ID)

);

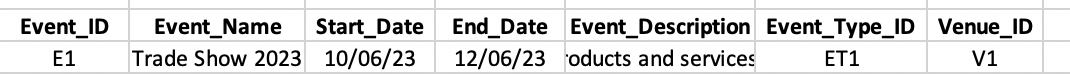
**6. DML**

We would use the SQL INSERT, UPDATE, and DELETE commands to populateand modify the current database. The following are the sample statements that could be used to help Foxcore Retail ensure its database is up-to-date and kept clear of unnecessary or outdated information.

The following are the sample statements that could be used to help Foxcore Retail ensure its database is up-to-date and kept clear of unnecessary or outdated information.

1. **To Insert a new row about all the details information on Event.**

INSERT INTO EVENT (Event\_ID, Event\_Name, Start\_Date, End\_Date, Event\_Description, Event\_Type\_ID, Venue\_ID) VALUES (‘E1’, ‘Trade Show 2023’, ‘2023-06-10’, ‘2023-06-12’, ‘Come and see the latest products and services from the industry leaders.’, ‘ET1’, ‘V1’);

****

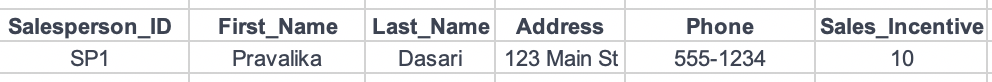
1. **To Insert a new row containing information on Sales from the Event.**

INSERT INTO SALES (Sales\_ID, Quantity, Final\_Price, Product\_ID, Salesperson\_ID) VALUES (‘SA1’, 10, 250.00, ‘P1’, ‘SP1’);

****

1. **To Insert information about the Salesperson working at the Event.**

INSERT INTO SALESPERSON (Salesperson\_ID, First\_Name, Last\_Name, Address, Phone, Sales\_Incentive) VALUES (‘SP1’,‘Pravalika’, ‘Dasari’, ‘123 Main St’,‘555-1234’, 10);

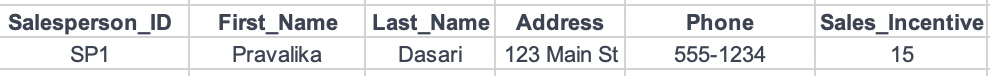
****

1. **How can you update the Sales\_Incentive in Salesperson table from 10.00 to 15.00?**

UPDATE SALESPERSON

SET Sales\_Incentive = 15.00

WHERE Salesperson\_ID = ‘SP1’;

****

1. **How can you update the Sales\_Incentive in Salesperson table who has sales greater than 200?**

UPDATE Salesperson

SET Sales\_Incentive = 20.00

WHERE Salesperson.ID IN (

    SELECT Salesperson\_ID

    FROM Sales

    WHERE Final\_Price > 200

);



1. **How to Delete Product whose Product\_ID is P1?**

DELETE FROM Product WHERE Product\_ID = ‘P1’;

**REPORTS**

1. Create a report that displays information about each salesperson, the percentage of incentive they earned, and the quantity of each product they sold. Additionally, include the name and selling price of each product, and the total sales for each product.

SELECT S.FIRST\_NAME, S.LAST\_NAME, S.SALES\_INCENTIVE, ST.QUANTITY, ST.FINAL\_PRICE, P.PRODUCT\_NAME, (ST.QUANTITY \* ST.FINAL\_PRICE) AS TOTAL\_COST

FROM SALESPERSON S

INNER JOIN SALES ST ON S.SALESPERSON\_ID = ST.SALESPERSON\_ID

INNER JOIN PRODUCT P ON S.PRODUCT\_ID = P. PRODUCT \_ID

****

1. Display a list of all events along with their corresponding details.

Select Venue\_Name, Event\_Name, Start\_Date,End\_Date,Event\_Type\_Name, Booth\_Location

From event as E

Inner join Event\_Type ET on E.Event\_Type\_id = ET.EVENT\_TYPE\_ID

INNER JOIN VENUE V ON E.VENUE\_ID = V.VENUE\_ID

INNER JOIN BOOTH B ON E.EVENT\_ID = B.EVENT\_ID

Table

Description automatically generated

1. List of Products sold by Foxcore.

Select    Product\_ID, Product\_Name, Wholesale\_Cost, Maximum\_SellingPrice,     Minimum\_SellingPrice

From Product

Table

Description automatically generated

**Conclusion**

There was inefficiency and inaccuracy in Foxcore Retails manual sales-tracking system. The solution to this problem was to design a custom relational database system that would enable the efficient tracking of events, sales consultants, and individual sales of various products. By accessing and analysing this database, valuable data could be collected to provide strategic insights that would improve decision-making for both short- and long-term planning. With this database, Foxcore Retail can accurately track its sales, identify which products are selling well, and optimize its operations accordingly.

In conclusion, the report outlines a comprehensive solution to the business problem faced by Foxcore Retail and provides a detailed plan for implementing a custom relational database system. The proposed database system is expected to improve the efficiency and accuracy of sales tracking, thereby enabling Foxcore Retail to make informed decisions about their operations and optimize their performance.