INFO 430: NIKE DATABASE DESIGN

Group 4
Jessi Zeng, Junna Cao, Hui Xie, Nicole Han

Information School
UNIVERSITY of WASHINGTON

Business Problem Space

Problem

As a global company, Nike **accumulates** a large sum of data from its consumers and partners making it **challenging** to **manage** and **extract** meaningful insights

Proposed Solution

To design an **efficient** inventory management system using a **secure** and **scalable** database

Stakeholders

Nike Employees

Customers

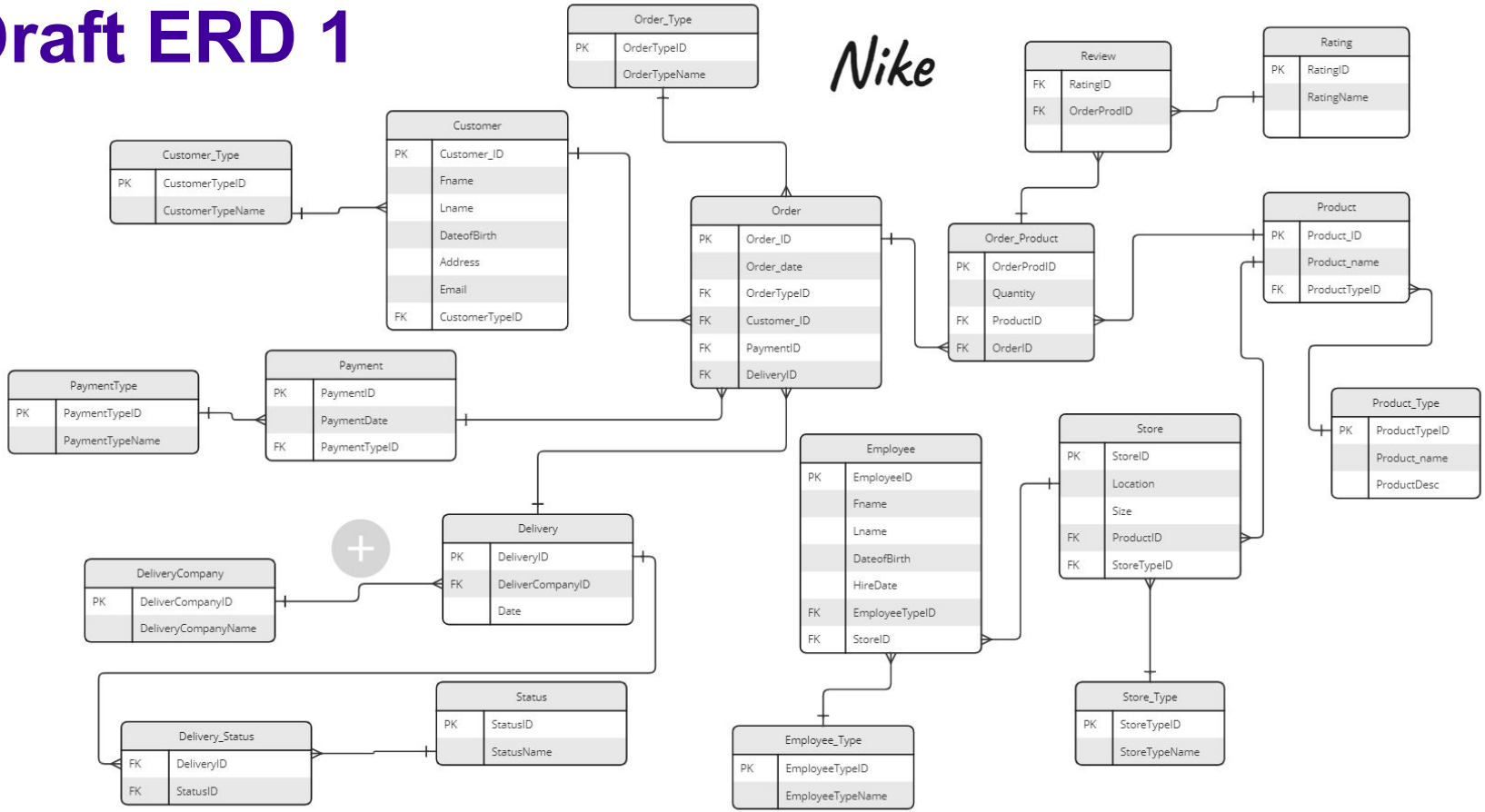
Suppliers/Manufacturers

Carriers

Designing database for…

- Product & Service Management
  - Inventory tracking
  - Product information
  - Order processing
- Customer Relationship Management
- Employee Management
- Sales & Marketing
  - Online & in store

Draft ERD 1


Nike
Order_Type
PK OrderTypeID
OrderTypeName
Review
FK RatingID
FK OrderProdID
Rating
PK RatingID
RatingName
Customer_Type
PK CustomerTypeID
CustomerTypeName
Customer
PK Customer_ID
Fname
Lname
DateofBirth
Address
Email
FK CustomerTypeID
Order
PK Order_ID
Order_date
FK OrderTypeID
FK Customer_ID
FK PaymentID
FK DeliveryID
Order_Product
PK OrderProdID
Quantity
FK ProductID
FK OrderID
Product
PK Product_ID
Product_name
FK ProductTypeID
Product_Type
PK ProductTypeID
Product_name
ProductDesc
PaymentType
PK PaymentTypeID
PaymentTypeName
Payment
PK PaymentID
PaymentDate
FK PaymentTypeID
Employee
PK EmployeeID
Fname
Lname
DateofBirth
HireDate
FK EmployeeTypeID
FK StoreID
Store
PK StoreID
Location
Size
FK ProductID
FK StoreTypeID
Delivery
PK DeliveryID
FK DeliverCompanyID
Date
DeliveryCompany
PK DeliverCompanyID
DeliveryCompanyName
Status
PK StatusID
StatusName
Delivery_Status
FK DeliveryID
FK StatusID
Employee_Type
PK EmployeeTypeID
EmployeeTypeName
Store_Type
PK StoreTypeID
StoreTypeName

Draft ERD 1

Entity Tables

- tblCustomer
- tblEmployee
- tblStore
- tblProduct
- tblOrder
- tblReview
- tblDelivery
- tblPayment

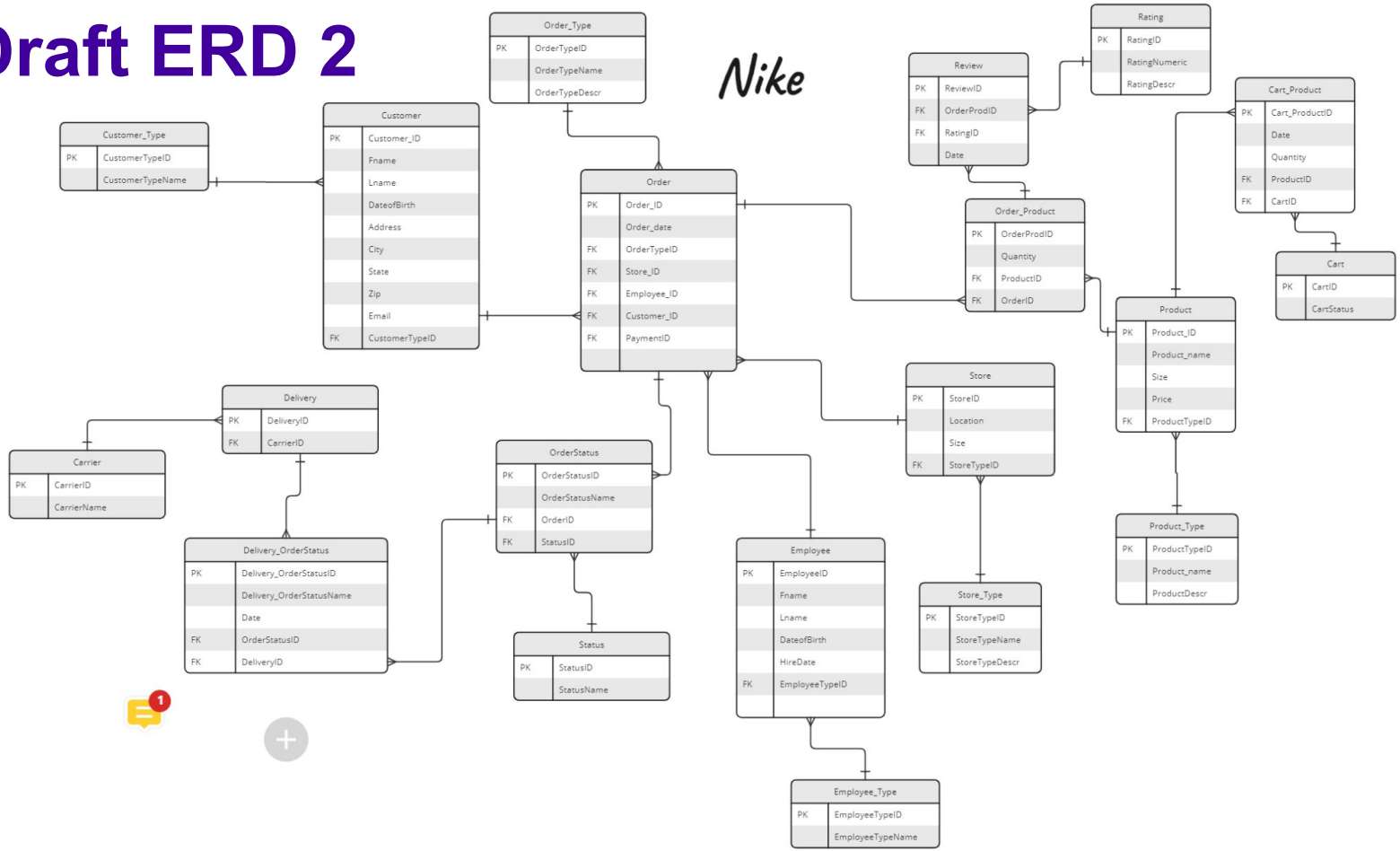
Type Tables

- tblCustomerType
- tblEmployeeType
- tblStoreType
- tblProductType
- tblOrderType
- tblRating
- tblDeliveryCompany
- tblPaymentType
- tblStatus

Transactional Tables

- tblOrder_Product
- tblDelivery_Status

Draft ERD 2


Nike
Order_Type
PK OrderTypeID
OrderTypeName
OrderTypeDescr
Customer_Type
PK CustomerTypeID
CustomerTypeName
Customer
PK Customer_ID
Fname
Lname
DateofBirth
Address
City
State
Zip
Email
FK CustomerTypeID
Order
PK Order_ID
Order_date
FK OrderTypeID
FK Store_ID
FK Employee_ID
FK Customer_ID
FK PaymentID
Review
PK ReviewID
FK OrderProdID
FK RatingID
Date
Rating
PK RatingID
RatingNumeric
RatingDescr
Cart_Product
PK Cart_ProductID
Date
Quantity
FK ProductID
FK CartID
Cart
PK CartID
CartStatus
Order_Product
PK OrderProdID
Quantity
FK ProductID
FK OrderID
Product
PK Product_ID
Product_name
Size
Price
FK ProductTypeID
Product_Type
PK ProductTypeID
Product_name
ProductDescr
Store
PK StoreID
Location
Size
FK StoreTypeID
Store_Type
PK StoreTypeID
StoreTypeName
StoreTypeDescr
Delivery
PK DeliveryID
FK CarrierID
Carrier
PK CarrierID
CarrierName
Delivery_OrderStatus
PK Delivery_OrderStatusID
Delivery_OrderStatusName
Date
FK OrderStatusID
FK DeliveryID
OrderStatus
PK OrderStatusID
OrderStatusName
FK OrderID
FK StatusID
Status
PK StatusID
StatusName
Employee
PK EmployeeID
Fname
Lname
DateofBirth
HireDate
FK EmployeeTypeID
Employee_Type
PK EmployeeTypeID
EmployeeTypeName

Draft ERD 2

Tables Added

- tblCart
- tblCart_product
- tblOrder_Status

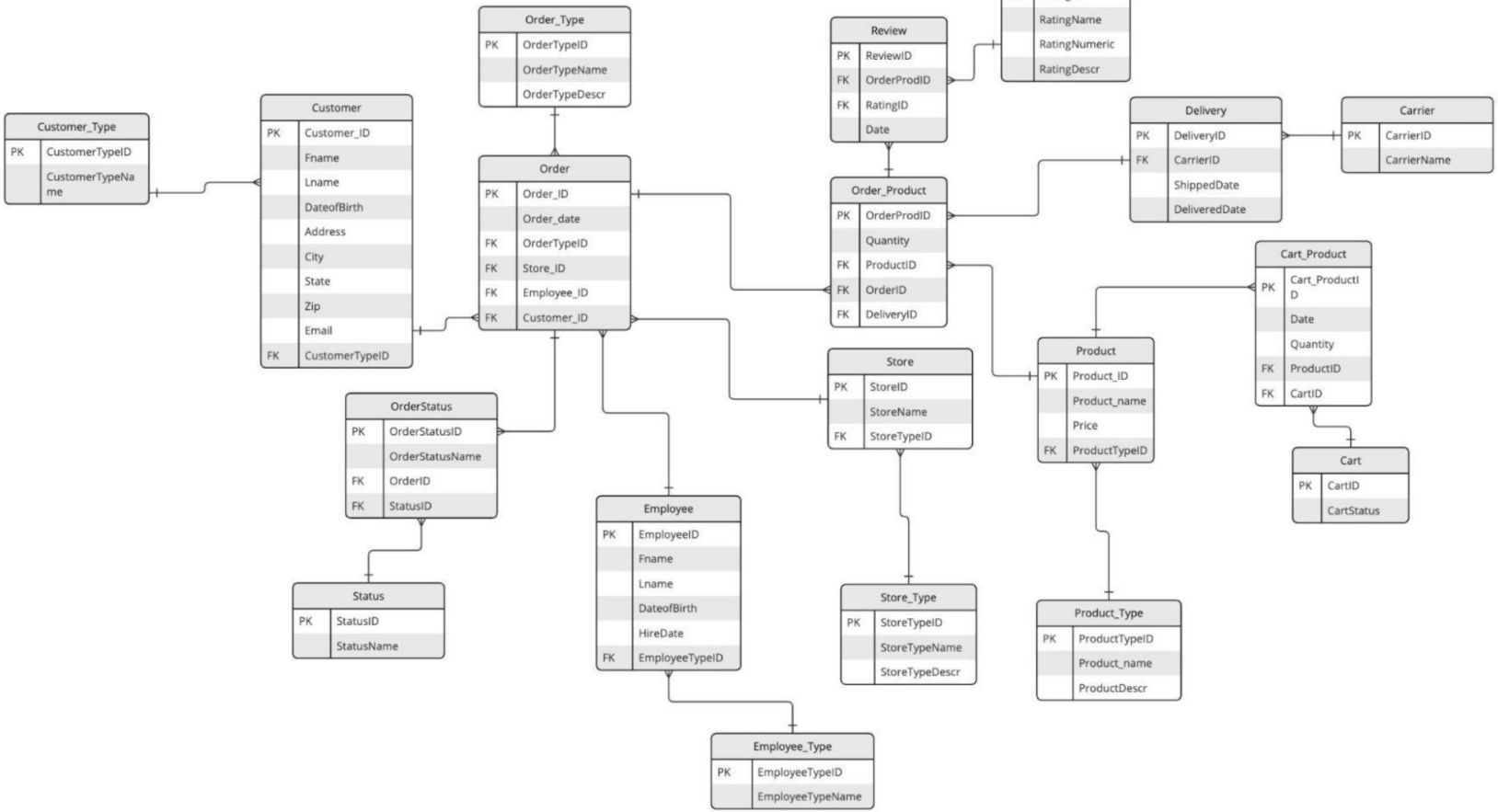
Tables Deleted

- tblPayment
- tblPaymentType

Tables Changed

- tblDelivery_Status -> tblDelivery_OrderStatus
- ProductID in tblStore -> tblOrder_Product & tblCart_Product
- StoreID in tblEmployee -> tblOrder

Nike


Rating
PK RatingID
RatingName
RatingNumeric
RatingDescr
Order_Type
PK OrderTypeID
OrderTypeName
OrderTypeDescr
Review
PK ReviewID
FK OrderProdID
FK RatingID
Date
Delivery
PK DeliveryID
FK CarrierID
ShippedDate
DeliveredDate
Carrier
PK CarrierID
CarrierName
Customer_Type
PK CustomerTypeID
CustomerTypeName
Customer
PK Customer_ID
Fname
Lname
DateofBirth
Address
City
State
Zip
Email
FK CustomerTypeID
Order
PK Order_ID
Order_date
FK OrderTypeID
FK Store_ID
FK Employee_ID
FK Customer_ID
Order_Product
PK OrderProdID
Quantity
FK ProductID
FK OrderID
FK DeliveryID
Cart_Product
PK Cart_ProductID
Date
Quantity
FK ProductID
FK CartID
Store
PK StoreID
StoreName
FK StoreTypeID
Product
PK Product_ID
Product_name
Price
FK ProductTypeID
Cart
PK CartID
CartStatus
OrderStatus
PK OrderStatusID
OrderStatusName
FK OrderID
FK StatusID
Employee
PK EmployeeID
Fname
Lname
DateofBirth
HireDate
FK EmployeeTypeID
Status
PK StatusID
StatusName
Store_Type
PK StoreTypeID
StoreTypeName
StoreTypeDescr
Product_Type
PK ProductTypeID
Product_name
ProductDescr
Employee_Type
PK EmployeeTypeID
EmployeeTypeName

Stored Procedure

- insert_product_type_product
- insert_cart_product_cart
- insert_employee_type_employee
- insert_store_type_store
- insert_rating_review
- insert_carrier_delivery
- insert_cusomer_type_customer
- insert_order_status_status

Stored Procedure - insert_order_status

```
CREATE OR ALTER PROCEDURE insert_OrderStatus
@StoreName VARCHAR(50),
@EFname VARCHAR(50),
@ELname VARCHAR(50),
@CEmail VARCHAR(75),
@StatusName VARCHAR(50)
AS
-- declare fk
DECLARE @OrderID INT, @StatusID INT

-- get fks
SET @OrderID = (
    SELECT TOP 1 OrderID
    FROM tblOrder O
        JOIN tblStore S on O.StoreID = S.storeID
        JOIN tblEmployee E ON O.EmployeeID = E.EmployeeID
        JOIN tblCustomer C ON O.CustomerID = C.CustomerID
    WHERE S.StoreName = @StoreName
    AND E.Fname = @EFname
    AND E.Lname = @ELname
    AND C.Email = @CEmail
)

IF @OrderID IS NULL
BEGIN
    PRINT '@OrderID is empty...check spelling';
    THROW 65451, '@OrderID cannot be NULL',1;
END

SET @StatusID = (
    SELECT StatusID
    FROM tblStatus
    WHERE StatusName = @StatusName
)

IF @StatusID IS NULL
BEGIN
    PRINT '@StatusID is empty...check spelling';
    THROW 65451, '@StatusID cannot be NULL',1;
END

BEGIN TRANSACTION T1
    INSERT INTO tblOrderStatus(OrderID, StatusID)
    VALUES(@OrderID, @StatusID)

    IF @@ERROR <> 0
        BEGIN
            ROLLBACK TRANSACTION T1
        END
    ELSE
        COMMIT TRANSACTION T1
GO
```

Stored Procedure - wrapper_insert_OrderStatus

```
CREATE OR ALTER PROCEDURE wrapper_insert_OrderStatus
@run INT
AS
DECLARE
@StoreName VARCHAR(50),
@EFname VARCHAR(50),
@ELname VARCHAR(50),
@CEmail VARCHAR(75),
@StatusName VARCHAR(50)
DECLARE @OrderPK INT, @StatusPK INT
DECLARE @OrderCount INT = (SELECT COUNT(*) FROM tblOrder)
DECLARE @StatusCount INT = (SELECT COUNT(*) FROM tblStatus)


WHILE @run > 0
BEGIN
    SET @OrderPK = (SELECT RAND() * @OrderCount + 1)
    SET @StatusPK = (SELECT RAND() * @StatusCount + 1)

    SET @StoreName = (
        SELECT StoreName
        FROM tblStore S
        JOIN tblOrder O ON S.StoreID = O.StoreID
        WHERE O.OrderID = @OrderPK
    )

    SET @EFname = (
        SELECT Fname
        FROM tblEmployee E
        JOIN tblOrder O on E.EmployeeID = O.EmployeeID
        WHERE O.OrderID = @OrderPK
    )

    SET @ELname = (
        SELECT Lname
        FROM tblEmployee E
        JOIN tblOrder O on E.EmployeeID = O.EmployeeID
        WHERE O.OrderID = @OrderPK
    )

    SET @CEmail = (
        SELECT Email
        FROM tblCustomer C
        JOIN tblOrder O on C.CustomerID = O.CustomerID
        WHERE O.OrderID = @OrderPK
    )

    SET @StatusName = (
        SELECT StatusName
        FROM tblStatus S
        JOIN tblOrder O on S.StatusID = O.OrderID
        WHERE O.OrderID = @OrderPK
    )

    EXEC insert_OrderStatus
    @StoreName = @StoreName,
    @EFname = @EFname,
    @ELname = @ELname,
    @CEmail = @CEmail,
    @StatusName = @StatusName

    SET @run = @run -1
END
```

Business Rule

- **Product & Service Management**
  - No order can take more than two days to ship once it's placed
  - No order can have the pending status more than one day
- **Customer Relationship Management**
  - No customer can be younger than 14 years old as a member, and get help from store manager who is younger than 25 years old
  - No customer can order more than 10 products labeled "limited edition"
  - Only customers with membership can order more than five products within 1 order in the Christmas shopping season in 2022 (11/24/2022 - 12/31/2022)
- **Employee Management**
  - No employee in WA with a hire date of less than one year can get a discount on items.
  - No employee under the age of 14 may work in washington state
- **Sales & Marketing**
  - No store can have less than 7 employees working at the same time.

Business Rule

- No order can have the pending status more than one day

```
CREATE FUNCTION fn_NoPendOrder()
RETURNS INTEGER
AS
BEGIN
    DECLARE @RET INTEGER = 0
    IF EXISTS (SELECT *
    FROM tblOrder O
        JOIN tblOrderStatus OS ON O.OrderID = OS.OrderID
        JOIN tblStatus S ON OS.StatusID = S.StatusID
    WHERE DATEDIFF(DAY, O.OrderDate, GETDATE()) > 1
    AND S.StatusName = 'Pending')
    BEGIN
        SET @RET = 1
    END
    RETURN @RET
END
GO
ALTER TABLE tblOrder
ADD CONSTRAINT CK_NoPendOrder
CHECK (dbo.fn_NoPendOrder() = 0)
```

Computed columns

- **Customer Relationship Management**
  - Measure the total shoes over $50 each customer has purchased
  - Measure the total customers are older than 50 years and joined as “Nike member” in each state
  - Measure the total amount of items worth more than $100 each customer has purchased
  - CASE statement labeling customer loyalty based on the number of products they have purchased
  - What’s the total number of ‘accessories’ brought by non members
- **Sales & Marketing**
  - Measure the total number of customers who have spent more than $2000 at each store
  - Measure the total product type ‘Hoodie’ under $20 for each store
  - Measure the total employees that are 20 years old currently in 2023 for each store

```
CREATE FUNCTION fn_totalCustomer50Shoes(@PK INT )
RETURNS INT
AS
BEGIN
DECLARE @RET INT = (SELECT COUNT(*)
                    FROM tblCustomer C
                    JOIN tblOrder O ON C.CustomerID = O.CustomerID
                    JOIN tblOrderProduct OP On O.OrderID = OP.OrderID
                    JOIN tblProduct P ON OP.ProductID = P.ProductID
                    JOIN tblProductType PT ON P.ProductTypeID = PT.ProductTypeID
                    WHERE P.Price > 50
                    AND PT.ProductTypeName = 'Shoes'
                    AND C.CustomerID = @PK
)
RETURN @RET
END
GO

ALTER TABLE tblCustomer
ADD TotalShoesOver50
AS (dbo.fn_totalCustomer50Shoes(CustomerID))
GO
```

	CustomerID	Fname	Lname	DOB	TotalShoesOver50
1	1	Sonya	Caffery	1960-01-12	0
2	2	Aatu	Rippey	1962-05-08	0

Views

- **Customer Relationship Management**
  - Select the customer that has purchased the most of product (monetary)
  - The customer has the highest cost for Nike shoes in Washington state in 2020
  - Select the 99 percentile users with the highest purchase
  - Which customers are between the top 20% and 30% in the money spent on women's clothing in the year 2022
- **Employee Management**
  - Select the employee that has worked the longest
  - Rank the employees onboarding length and age is between 30 to 40, return with job title
- **Sales & Marketing**
  - Select the top 10 stores that have the highest revenue over the years 2010 - 2015
  - Which state has the highest cumulative dollar spent on `shoes` (product type) in the year 2021

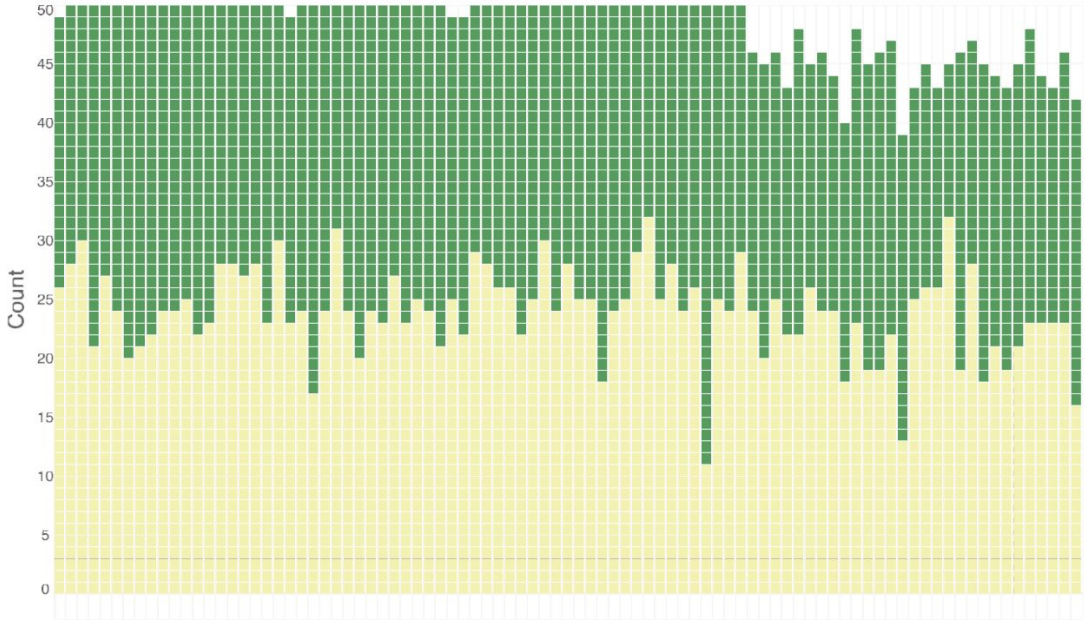
```sql
--Select the top 10 stores that have the highest revenue over the years 2010 - 2015


CREATE VIEW vw_top10storeRev AS
   SELECT S.StoreName, SUM(P.price) AS totalRevenue
   FROM tblStore S
      JOIN tblOrder O ON S.StoreID = O.StoreID
      JOIN tblOrderProduct OP ON O.OrderID = OP.OrderID
      JOIN tblProduct P ON OP.ProductID = P.ProductID
   WHERE O.Orderdate BETWEEN '2010-01-01' AND '2015-12-31'
   GROUP BY S.StoreName

GO
SELECT TOP 10 *
FROM vw_top10storeRev
ORDER BY totalRevenue DESC
```

```sql
-- The customer has the highest cost for Nike shoes in Washington state in 2020
CREATE VIEW vw_CustomerHighestCostNikeWashington2020 AS
SELECT C.CustomerID, C.Fname, O.OrderID, P.ProductID, OP.Quantity, P.Price, C.State, O.OrderDate,
    RANK() OVER (ORDER BY SUM(OP.Quantity * P.Price) DESC) AS CostRank
FROM tblCustomer C
    JOIN tblOrder O ON C.CustomerID = O.CustomerID
    JOIN tblOrderProduct OP ON O.OrderID = OP.OrderID
    JOIN tblProduct P ON OP.ProductID = P.ProductID
    JOIN tblProductType PT ON P.ProductTypeID = PT.ProductTypeID
WHERE PT.ProductTypeName = 'Shoes'
AND C.State = 'Washington'
AND YEAR(O.OrderDate) = 2020
GROUP BY C.CustomerID, C.Fname, O.OrderID, P.ProductID, OP.Quantity, P.Price, C.State, O.OrderDate;
GO
SELECT *
FROM vw_CustomerHighestCostNikeWashington2020
WHERE CostRank = 1;
```

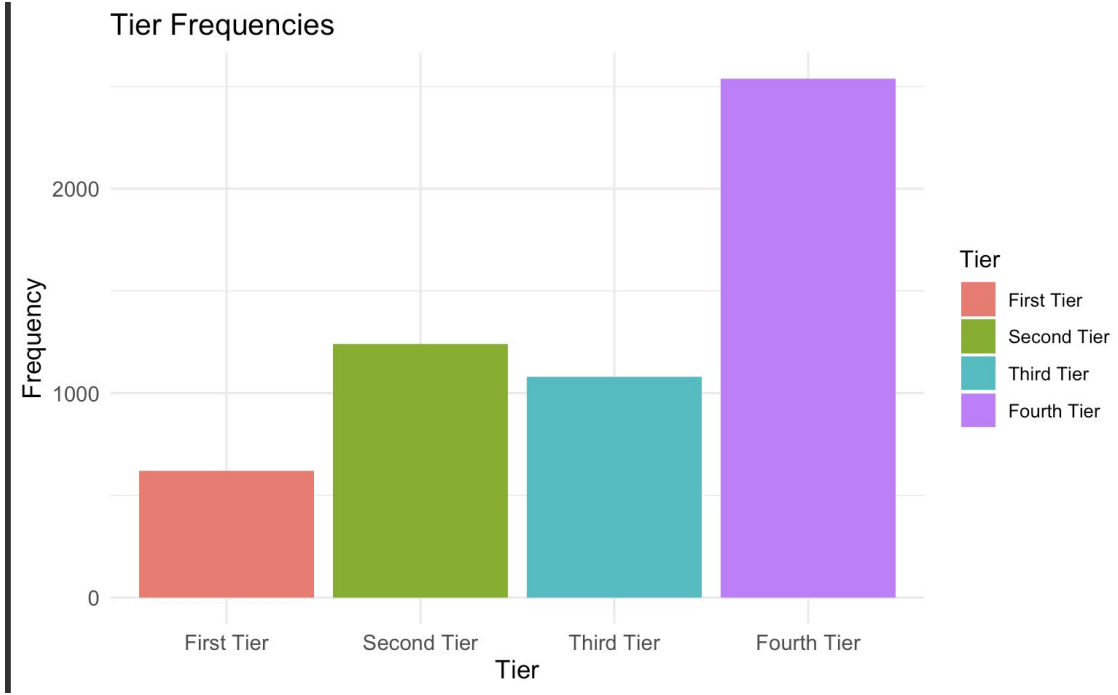
Data Visualization - Customer Demographics



Customer Type

- **Member**
- **Non-member**

Data Visualization - Customer Demographics



First Tier

- purchased more than 25 products

Second Tier

- purchased between 10 and 15 products

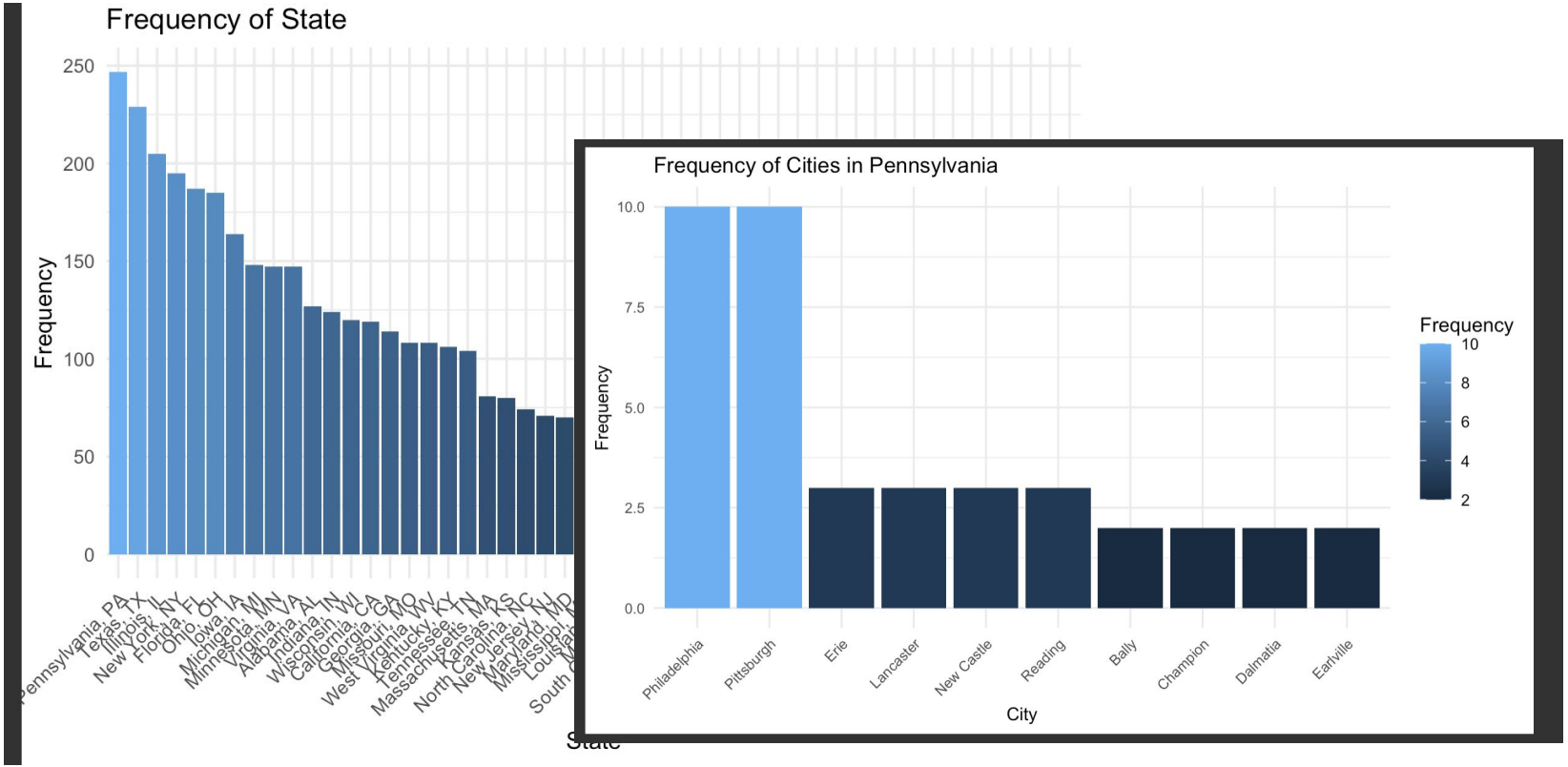
Third Tier

- purchased between 3 and 10 products

Fourth tier

- Purchased fewer than 3 products, mark them as fourth tier

Data Visualization - Customer Demographics




The End