Eric Harris
Nabil Baalbaki
Ryan O'Leary
Nour-Aldine Dabbagh

ISYS 464: Managing Enterprise Data
December 20, 2018

Project: Part 3

1. Intro Paragraph

A. Company Name: Nike, Inc
B. Specific business sector: Sports Good
C. Short description: Nike, Inc. is an American multinational corporation that is engaged in the design, development, manufacturing, and worldwide marketing and sales of footwear, apparel, equipment, accessories, and services. The company is headquartered near Beaverton, Oregon, in the Portland metropolitan area. We will focus on the shipment of footwear for the purposes of this assignment. Shipment being, everything from where the shoe is manufactured/shipped from to the customer getting the shoe in the mail

2. Entities & Attributes

A. Full List (Note: **Bold** = foreign Key)

Customer:

- CustomerID: NUMBER NOT NULL,
    i. Is a composite key in Order entity and primary key of Customer
- CustomerName VARCHAR(30)
- CustomerEmail VARCHAR(50)
- CustomerAddress VARCHAR(60)
    i. All attributes under this are composite to Address.
- CustomerCity VARCHAR(40)
    i. CustomerState VARCHAR(30)
    ii. CustomerZip NUMBER
    iii. CustomerCountry VARCHAR (30)

2. **Order**:

- OrderID NUMBER NOT NULL
    - Primary key in order entity
- **CustomerID** NUMBER NOT NULL
    - Foreign key in Order entity
- OrderDate DATE
- ShipDate DATE
- Paid BOOLEAN
- **ShippingID** NUMBER NOT NULL
    - Foreign key in Order entity to shipping
- **PaymentID** NUMBER NOT NULL
    - Foreign key in Order entity to Payment

3. **Payment:**
    - PaymentID NUMBER NOT NULL
        - Foreign key in Payment entity
    - PaymentAmount CURRENCY
    - PaymentMethod TEXT
    - PaymentDate DATE

4. **OrderDetail:**
    - OrderDetailID NUMBER NOT NULL
        - Primary key in Order details entity
    - **OrderID** NUMBER NOT NULL
        - Foreign key in OrderDetail entity to Order entity
    - **ProductID** NUMBER NOT NULL
        - Foreign key in OrderDetail entity to Product entity
    - OrderPrice CURRENCY
    - Quantity NUMBER
    - Size VARCHAR(50)
        - Characters to state whether an item is male or female
    - Color CHAR

5. **Product**
    - ProductID NUMBER NOT NULL
    - **ManufacturerID** NUMBER NOT NULL
        - Foreign key in Product entity to manufacturer entity
    - ProductName VARCHAR(100)
    - ProductDescription VARCHAR(100)
    - ModelID NUMBER NOT NULL
    - ProductColor VARCHAR(30)
    - Size NUMBER
    - ProductPrice CURRENCY
6. **Shipping**:

- ShippingID NUMBER NOT NULL
    - Primary key in Shipping entity
- OrderID
    - Foreign key in order entity
- CompanyName CHAR(45)
- TrackingID NUMBER NOT NULL (35)
- ShipDate DATE
- Shipped BOOLEAN

7.**Manufacturer**:

- ManufactuerID NUMBER NOT NULL
    - Primary key in manufacturer entity
- ProductID NUMBER NOT NULL
    - Foreign key comto product entity
- ManufactuerPhone NUMBER
- ManufacturerEmail VARCHAR(50)
- ManufacturerAddress VARCHAR(60)
    - All attributes under this are composite to Address
- ManufacturerCity, CHAR(40)
- ManufacturerState CHAR(35)
- ManufacturerZip NUMBER
- ManufacturerCountry CHAR(30)

B. Short Description of Each

Customer:

Customer places the order, the order will connect to a order detail entity. One customer may have many orders.

Order:

The order connects the order to the customer table and also to the OrderDetails, PaymentID, and Shipping.

Payment:

PaymentID connects the order to the payment.

OrderDetails:

This orderdetail shows a more in depth detail of the order showing size, color and quantity. It is connected to the Order entity by the OrderID and connected to the product by ProductID.

Product:

This would have information on the product details and connect the entity to orderID. It is also cohesive with the Manufacturer.

Shipping:

The shipping entity will be a foreign key to the order. This shipping order will be able to tell the customer who (UPS, FEDEX, DHL) is shipping the order, ship date and estimate the delivery date.

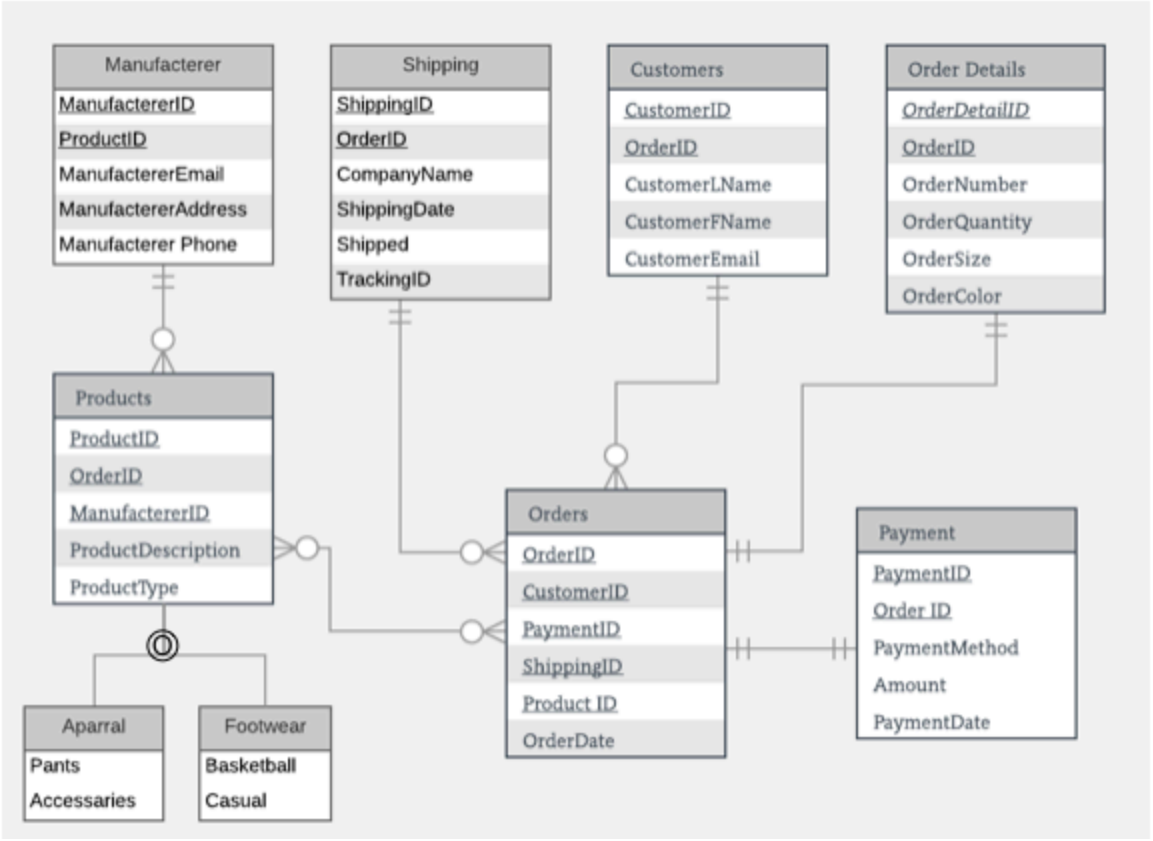
Manufacturer:

The manufacturer entity will be connected to product. Details about where the product came from.

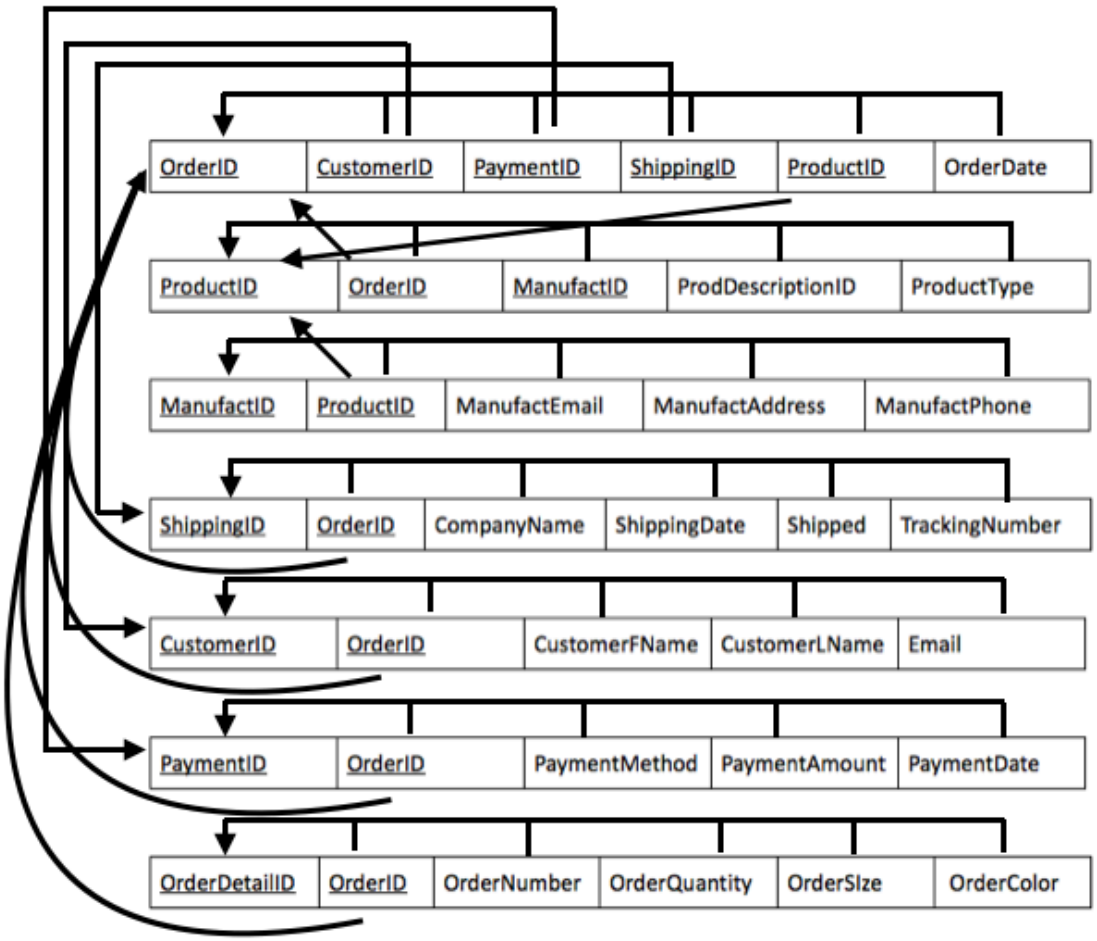
3. Business Rules:

1. A **Manufacturer** can supply multiple **Products**.
2. A **Product** is supplied by only one **Manufacturer**.
3. A **Product** can have multiple **Orders**.
4. **Order** can have multiple **Products**.
5. An **Order Detail** is only part of one **Order**.
6. Each **Order** has only one **Order Detail**.
7. Each **Order** has one **Shipper**.
8. Each **Shipper** can have multiple **Orders**.
9. Each **Order** has only one **Customer**.
10. A **Customer** can have multiple **Orders**.
11. Each **Order** has only one **Payment**.
12. Each **Payment** has only one **Order**
13. **Products** can be both **Apparel and Footwear - Overlay**
14. Subclasses **Apparel** and **Footwear** have **partial specialization**

4. ER Diagram:


Manufacterer
ManufactererID
ProductID
ManufactererEmail
ManufactererAddress
Manufacterer Phone
Shipping
ShippingID
OrderID
CompanyName
ShippingDate
Shipped
TrackingID
Customers
CustomerID
OrderID
CustomerLName
CustomerFName
CustomerEmail
Order Details
OrderDetailID
OrderID
OrderNumber
OrderQuantity
OrderSize
OrderColor
Products
ProductID
OrderID
ManufactererID
ProductDescription
ProductType
Orders
OrderID
CustomerID
PaymentID
ShippingID
Product ID
OrderDate
Payment
PaymentID
Order ID
PaymentMethod
Amount
PaymentDate
Aparral
Pants
Accessaries
Footwear
Basketball
Casual

5. Relation Model



6. SQL Statements

A. Create Tables:

```
CREATE TABLE ORDER
        (
        ORDERID         NUMBER          NOT NULL,
        CUSTOMERID      NUMBER          NOT NULL,
        PAYMENTID       NUMBER          NOT NULL,
        SHIPPINGID      NUMBER          NOT NULL,
```

```
	PRODUCTID	NUMBER		NOT NULL,
	ORDER_DATE	NUMBER,

	CONSTRAINT Order_PK PRIMARY KEY (ORDERID),

	CONSTRAINT Order_FK FOREIGN KEY (CUSTOMERID) REFERENCES
CUSTOMER(CUSTOMERID)
	CONSTRAINT Order_FK FOREIGN KEY (PAYMENTID) REFERENCES
PAYMENT(PAYMENTID)
	CONSTRAINT Order_FK FOREIGN KEY (SHIPPINGID) REFERENCES
SHIPPING(SHIPPINGID)
	CONSTRAINT Order_FK FOREIGN KEY (PRODUCTID) REFERENCES
PRODUCT(PRODUCTID)
	);

CREATE TABLE PRODUCT
	(
	PRODUCTID	NUMBER		NOT NULL,
	ORDERID		NUMBER		NOT NULL,
	MANUFACTID		NUMBER		NOT NULL,
	PRODUCTDESCRIPTION	VARCHAR(50),
	PRODUCTTYPE		VARCHAR(50),

	CONSTRAINT Product_PK PRIMARY KEY (PRODUCTID),

CONSTRAINT Product_FK Foreign Key (ORDERID) REFERENCES
ORDER(ORDERID)
CONSTRAINT Product_FK Foreign Key (MANUFACTID)
	);

CREATE TABLE MANUFACTURER
	(
	MANUFACTID		NUMBER		NOT NULL,
	PRODUCTID		NUMBER		NOT NULL,
	MANUFACT_EMAIL	VARCHAR(50),
	MANUFACT_PHONE	CHAR(13),

	CONSTRAINT MANUFACTURER_PK PRIMARY KEY (MANUFACTID),
```

```
	CONSTRAINT MANUFACTURER_FK Foreign Key (PRODUCTID)
REFERENCES PRODUCT(PRODUCTID)
	);

CREATE TABLE SHIPPING
	(
	SHIPPINGID		NUMBER		NOT NULL,
	ORDERID			NUMBER		NOT NULL,
	COMPANY_NAME		VARCHAR(40),
	SHIPPING_DATE		DATE		NOTNULL,
	SHIPPED			BOOLEAN		NOTNULL,
	TRACKING_NUMBER		VARCHAR(30),

	CONSTRAINT SHIPPING_PK PRIMARY KEY (SHIPPINGID),

	CONSTRAINT SHIPPING_FK FOREIGN KEY (ORDERID) REFERENCES
ORDER(ORDERID)
	);

CREATE TABLE ORDERDETAIL
	(
	ORDERDETAILID		NUMBER	NOT NULL,
	ORDERID			NUMBER	NOT NULL,
	ORDER_NUMBER		CHAR(25)	NOT NULL,
	ORDER_QUANTITY		NUMBER	NOT NULL,
	ORDER_COLOR		VARCHAR,

	CONSTRAINT ORDERDETAIL_PK PRIMARY KEY (ORDERDETAILID),

	CONSTRAINT ORDERDETAIL_FK FOREIGN KEY (ORDERID) REFERENCES
ORDER(ORDERID)
	);


CREATE TABLE CUSTOMER
	(
	CUSTOMERID	NUMBER	NOT NULL,
	ORDERID		NUMBER	NOT NULL,
	CUSTOMERFNAME		VARCHAR(30)		NOT NULL,
```

```
	CUSTOMERLNAME		VARCHAR(40)		NOT NULL,
	CUSTOMEREMAIL		VARCHAR(50)		NOT NULL,

	CONSTRAINT CUSTOMER_PK PRIMARY KEY (CUSTOMERID),

	CONSTRAINT CUSTOMER_FK FOREIGN KEY (ORDERID) REFERENCES
ORDER(ORDERID)
	);

CREATE TABLE PAYMENT
	(
	PAYMENTID		NUMBER		NOT NULL,
	ORDERID			NUMBER		NOT NULL,
	PAYMENT_METHOD
	PAYMENT_AMOUNT				CURRENCY,
	PAYMENT_DATE	DEFAULT

	CONSTRAINT PAYMENT_PK PRIMARY KEY (PAYMENTID),

	CONSTRAINT PAYMENT_FK FOREIGN KEY (ORDERID) REFERENCES
ORDER(ORDERID)
	);
```

B. Load Data:

```
INSERT INTO Customer VALUES
(001, "Smith', 'Alex', "needajob@gmail.com', '123 Sesame Street');

INSERT INTO Customer VALUES
(002, 'Obama', 'Barack', 'potus@gmail.com', '1600 Pennsylvania Ave');

INSERT INTO Customer VALUES
(003, 'Simpson', 'OJ', thejuiceman@gmail.com', '4500 Penitentiary Ave');


INSERT INTO Product VALUES
(7569, 7, 100, 'working style boots', 'Footwear');

INSERT INTO Product VALUES
```

(7570, 8, 101, 'coldgear long-sleeve shirt','Apparel');

INSERT INTO Product VALUES
(7575, 9, 102, 'dry-fit shorts', "Apparel');

INSERT INTO Order VALUES
(10, 004, 222, 43, 7576, “12/05/2018”);

INSERT INTO Order VALUES
(11, 005, 223, 44, 7577, “12/09/2018”);

INSERT INTO Order VALUES
(12, 006, 224, 45, 7578, “12/11/2018”);

INSERT INTO Manufacturer VALUES
(103, 7588, ‘ItsAllYours, Inc’, ‘metoyou@gmail.com', ‘455 Ugandy St’, 4156572345);

INSERT INTO Manufacturer VALUES
(104, 7589, ‘FoxCon’, ‘foxcon@gmail.com’, ‘601 Morage St’, 4153232345);

INSERT INTO Manufacturer VALUES
(105, 7590, ‘Acme’, acme@gmail.com, ‘55 Lombard St’, 4158792345);

INSERT INTO Shipping VALUES
(46, 13, ‘Chase’, “12/05/2018”, 78668, Yes);

INSERT INTO Shipping VALUES
(47, 14, ‘DHL’, “12/25/2018”, 67876, No);

INSERT INTO Shipping VALUES
(48, 15, ‘HDSCominThru’, “12/10/2018”, 76876, Yes);

INSERT INTO Payment VALUES
(225, 16, ‘cash’, “12/01/2018”, $45.99);

INSERT INTO Payment VALUES
(226, 17, ‘credit’, “11/13/2018”, $99.99);

INSERT INTO Payment VALUES
(227, 18, ‘credit’, “12/06/2018”, $9.99);

INSERT INTO OrderDetails VALUES
(991, 19, SM, 3, ‘blue’);

INSERT INTO OrderDetails VALUES
(992, 20, LG, 1, ‘red’);

INSERT INTO OrderDetails VALUES
(993, 21, XXXL, ‘pink’);

C. Retrievals:

SELECT OrderID FROM Shipping WHERE ShippingDate < ‘19-DEC-2018’;

This shows all the orders that shipped before December 19th which guarantees delivery by Christmas Holiday.

SELECT ProductID FROM Order WHERE ProductID LIKE ‘757*’ GROUP BY OrderDate WHERE OrderDate < ‘15-DEC-2018’;

This shows all products ordered by December 15th, 2018 with a similar product ID number. This will make it easier to place similar orders together from the manufacturer for shipping.

SELECT ALL OrderID FROM Customer WHERE CustomerID = ‘002’;

This will show all orders placed by the customer Barack Obama. This will make shipping more organized since all of his orders will be placed together for shipping.

SELECT OrderSize, OrderColor FROM OrderDetails WHERE OrderSize LIKE ‘*XXXL’ AND OrderColor LIKE ‘*pink’;
This statement will display orders that have items in the XXXL size and color pink. This would narrow down the search for a product that was ordered but is not currently available in that combination. Customer service will get the order details and contact the customer.

OrderDetailI	OrderID	OrderSize	OrderQuant	OrderColor
993	21	XXXL	1	pink

SELECT ProductID FROM Manufacturer WHERE ManufacturerID (SELECT ManufacturerID FROM Product WHERE ManufacturerID = 105);
Using subcategories, this statement will show us what product this manufacturer makes based on the ProductID and ManufacturerID.

ManufacturerID	ProductID	ManufacturerName	ManufacturerEmail	ManufacturerAddress	ManufacturerPhone
105	7590	Acme	acme@gmail.com	55 Lombard St	4158792345

7. Concluding Paragraph

A. The Benefits

Throughout this experience, the project gave us a much clearer understanding of how a database management system works (DBMS). It also helped to fortify concepts such as the ER diagrams and normalization. This allowed us see some aspects of how an e-commerce organization works with online orders with all the various entities that are involved. With this project, it also enabled us to practice our SQL skills more. We understand that it is much more complex but the project gave us the groundwork and painted a picture for us.

B. Challenges

Some of the challenges that we encountered during the course of the project include making changes to the scope of the database and running our SQL statements. During the project there were some instances where we became unsure about certain aspects about our database. We would deal with this by revisiting our entities/attributes and our ER diagram. From there we would make critical decisions about revising or keeping certain entities and attributes. We feel as though our biggest challenge may come from syntax errors with our SQL queries. We understand that we must carefully review and proofread our work and that we will get better with practice and time. The more queries we write, the better we will get.