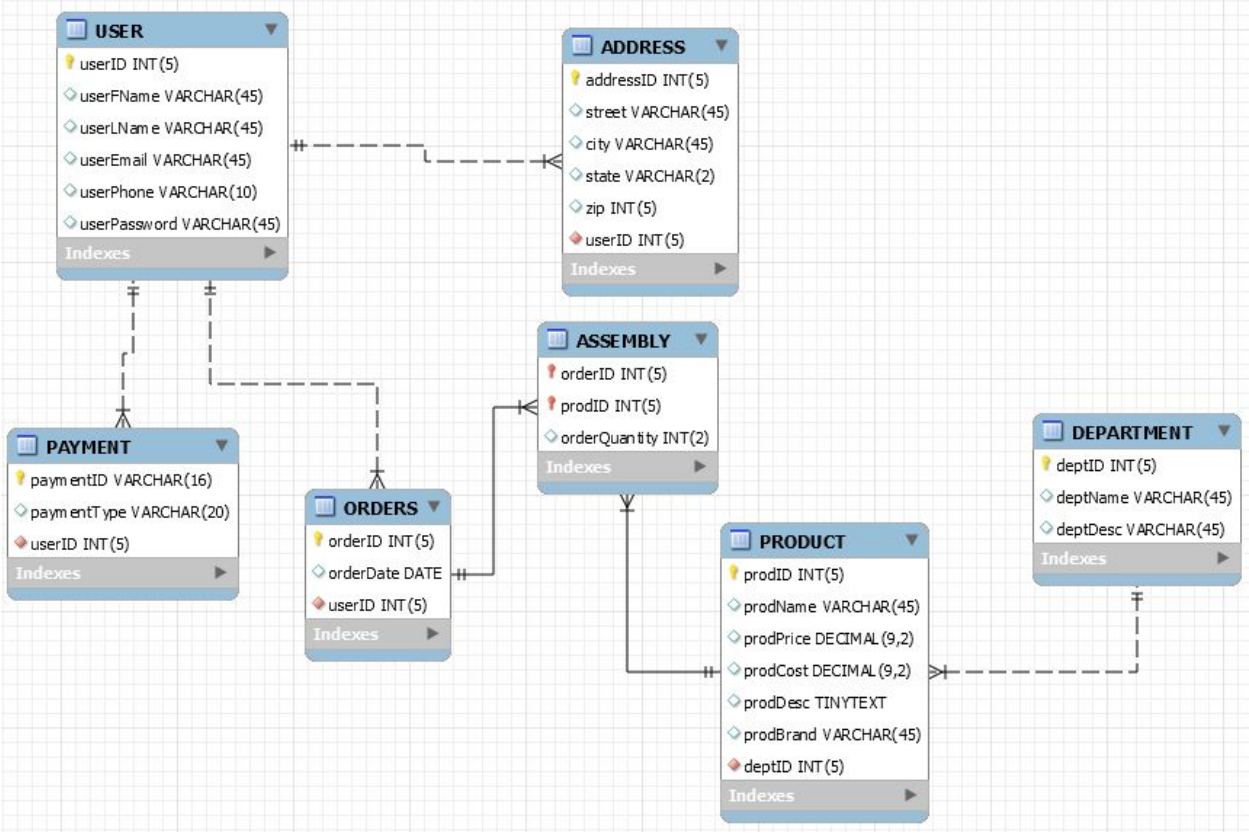
GROUP PROJECT 1 | GROUP 3 (Team MISTIC)

He, Dullabh, Ear, Granski, Alejos

Data Model



Data Dictionary

Table: **ADDRESS**

Column Name	Description	Data Type	Size	Format	Key
addressID	Unique sequential number identifying each address	Text	5		PK
street	Street name of the address	Text	45		
city	City name of the address	Text	45	Athens	
state	State name of the address	Text	2	GA	
zip	Zip code of the address	Text	5	30542	

userID	Indicates the user who has the listed address	Text	5	12553	FK

Table: **ASSEMBLY**

Column Name	Description	Data Type	Size	Format	Key
orderID	Unique sequential number indicating an order	Text	5		PK
prodID	Unique sequential number which identifies a specific product	Text	5		PK
orderQuantity	Number of orders a user has	Text	2	8	

Table: **DEPARTMENT**

Column Name	Description	Data Type	Size	Format	Key
deptID	Unique sequential number indicating a department	Text	5		PK
deptName	Name of the department	Text	45		
deptDesc	Description of the department	Text	45		

Table: **ORDERS**

Column Name	Description	Data Type	Size	Format	Key
orderID	Unique sequential number indicating an order	Text	5		PK
orderDate	The date the order was taken	Text	10	02/21/2017	
userID	Indicates which user has what order	Text	5		FK

Table: **PAYMENT**

Column Name	Description	Data Type	Size	Format	Key
paymentID	Unique sequential number which is the credit/debit card number	Text	16	4021304290433987	PK
paymentType	The type of payment such as credit or debit	Text	20	American Express	
userID	Indicates the user using which payment	Text	5		FK

Table: **PRODUCT**

Column Name	Description	Data Type	Size	Format	Key
prodID	Unique sequential number which identifies a specific product	Text	5		PK
prodName	Name of the product	Text	45		
prodPrice	Price of the product	Numeric	9	$199.99	
prodCost	Cost of the product	Numeric	9	$185	
prodDesc	Description of the product	Text	45		
prodBrand	The product's brand	Text	45		
deptID	The category the product falls under	Text	5		FK

Table: **USER**

Column Name	Description	Data Type	Size	Format	Key
userID	Unique sequential number which identifies a specific user	Text	5		PK
userFName	The user's first name	Text	45	Jalen	
userLName	The user's last name	Text	45	Granski	

userEmail	The user's email address	Text	45	mail@gmail.com	
userPhone	The user's phone number	Text	10	(888)888-8888	
userPassword	The user's password	Text	45	jalengranski2012	

Description of Queries

Query 1

- *Description:* This query reports the most expensive product ordered in states with more than three orders.
- *Justification:* This information helps Nordstrom determine which expensive product is worth selling more in which state.

Query 2

- *Description:* This query reports the products along with its brand that have order quantities less than 10 in order.
- *Justification:* This information allows Nordstrom to compare which products are being ordered more than the other. Nordstrom can use this information to look further into advertising extensively for certain products.

Query 3

- *Description:* This query reports the total amount of orders spent by people with first names that start with the letter 'A'.
- *Justification:* This information specifically looks into the customer base whose first name starts with an 'A'. Nordstrom can use this information to tailor the amount of advertising, email subscriptions, and mail subscriptions by looking into people who don't order many products.

Query 4

- *Description:* This query reports the products whose average price is greater than the average cost and by how much.
- *Justification:* This information determines which products on average give us a profit margin and by how much. Nordstrom can use this information to better allocate resources/labor to lower costs or raise prices to generate more profit.

Query 5

- *Description:* This query reports the users who live in Georgia, Florida, Tennessee, and South Carolina along with the products they ordered and the order dates.
- *Justification:* This information determines which products were ordered in a specific time frame for users who live in the Southeast region. Nordstrom can use this information to figure out if a product is in season. For example, we see a trend of boots being bought in the month of December.

Query 6

- *Description:* This query reports the order date and the order quantity of brands whose order quantity is greater than the average order quantity of that brand.
- *Justification:* This information determines when the brand was ordered more often than usual. Nordstrom can use this information to release a product during times when it'll be predictably popular.

Query 7

- *Description:* This query reports the method of payment for users who purchased Steve Madden products only.
- *Justification:* This information can be used to determine which type of method of payment is used more when users buy Steve Madden products. Nordstrom can decide which credit card company to partner with to create a rewards/incentive program to encourage the credit card user to buy more Steve Madden products.

Query 8

- *Description:* This query reports users who bought Kate Spade products and reports the users' email address.
- *Justification:* This information is relevant because Nordstrom can find all the users who purchased any Kate Spade products and send them more information about their products so that they can purchase more in the future.

Query 9

- *Description:* This query reports the users' cities and the number of orders in each city.
- *Justification:* This query is helpful to Nordstrom because it provides information on which city their customers are located and how many orders they are purchasing. It can be used to depict why their customers are purchasing a specific item based on location. For example, Nordstrom may need to know why customers from CT are purchasing jackets or men's boots.

Query 10

- *Description:* This query reports the product names and the number of users who bought each product.
- *Justification:* This information allows Nordstrom to see how many people purchased each product, and they can use this information to determine whether or not they need to restock.

SQL Queries

Query 1

```
# Report the most expensive product ordered in states with more than three orders.

SELECT state AS 'State', prodName AS 'Product Name',
MAX(prodPrice) AS 'Product Price'

FROM ADDRESS

JOIN USER ON ADDRESS.userID = USER.userID

JOIN ORDERS ON USER.userID = ORDERS.userID

JOIN ASSEMBLY ON ORDERS.orderID = ASSEMBLY.orderID

JOIN PRODUCT ON ASSEMBLY.prodID = PRODUCT.prodID

GROUP BY state

HAVING COUNT(ASSEMBLY.orderQuantity) > 3;
```

Query 2

```
# Report the products along with its brand that have order quantities less than 10 in order.

SELECT prodBrand AS 'Product Brand', prodName AS 'Product Name',
orderQuantity AS 'Order Quantity'

FROM PRODUCT

JOIN ASSEMBLY ON PRODUCT.prodID = ASSEMBLY.prodID

JOIN ORDERS ON ASSEMBLY.orderID = ORDERS.orderID
```

```
WHERE orderQuantity < 10

ORDER BY orderQuantity;
```

Query 3

```
# Report the total amount of orders spent by people with first
names that start with the letter ‘A’.

SELECT userFname AS 'User First Name', userLName 'User Last
Name', userEmail AS 'Email', street AS 'Street', city AS 'City',
state AS 'State', zip AS 'Zip Code',
COUNT(ASSEMBLY.orderQuantity) AS 'Order Quantity'

FROM USER

JOIN ADDRESS ON ADDRESS.userID = USER.userID

JOIN ORDERS ON ORDERS.userID = USER.userID

JOIN ASSEMBLY ON ASSEMBLY.orderID = ORDERS.orderID

JOIN PRODUCT ON PRODUCT.prodID = ASSEMBLY.prodID

GROUP BY USER.userID

HAVING USER.userFname REGEXP '^A';
```

Query 4

```
# Report the products whose average price is greater than the
average cost and by how much.

SELECT prodName AS 'Product Name', ROUND(AVG(prodPrice),2) AS
'Average Price', ROUND(AVG(prodCost),2) AS 'Average Cost',
(ROUND(AVG(prodPrice),2)-ROUND(AVG(prodCost),2)) AS 'Profit
Margin'

FROM PRODUCT

GROUP BY prodID

HAVING AVG(prodPrice) > (SELECT AVG(prodCost) FROM PRODUCT);
```

Query 5

```
# Report the users who live in Georgia, Florida, Tennessee, and South Carolina along with the products they ordered and the order dates.

SELECT userFName AS 'User First Name', userLName AS 'User Last Name', prodName AS 'Product Name', orderDate AS 'Order Date', state AS 'State'

FROM ADDRESS

JOIN USER ON USER.userID = ADDRESS.userID

JOIN ORDERS ON ORDERS.userID = USER.userID

JOIN ASSEMBLY ON ASSEMBLY.orderID = ORDERS.orderID

JOIN PRODUCT ON PRODUCT.prodID = ASSEMBLY.prodID

WHERE state IN('GA','FL','TN','SC');
```

Query 6

```
# Report the order date and the order quantity of brands whose order quantity is greater than the average order quantity of that brand.

SELECT orderQuantity, prodBrand, orderDate

FROM ASSEMBLY

JOIN PRODUCT ON ASSEMBLY.prodID = PRODUCT.prodID

JOIN ORDERS ON ASSEMBLY.orderID = ORDERS.orderID

WHERE orderQuantity > (SELECT AVG(orderQuantity) FROM ASSEMBLY
WHERE ASSEMBLY.prodID = PRODUCT.prodID);
```

Query 7

```
# Report the method of payment for users who purchased Steve Madden products only.

SELECT paymentType AS 'Payment Type', prodName AS 'Product Name', prodBrand AS 'Product Brand'
```

```
FROM PAYMENT

JOIN USER ON PAYMENT.userID = USER.userID

JOIN ORDERS ON USER.userID = ORDERS.userID

JOIN ASSEMBLY ON ORDERS.orderID = ASSEMBLY.orderID

JOIN PRODUCT ON ASSEMBLY.prodID = PRODUCT.prodID

WHERE prodBrand = 'Steve Madden'

GROUP BY paymentType;
```

Query 8

```
# Report users who bought Kate Spade products and reports the
users' email address.

SELECT userFName AS 'User First Name', userLName AS 'User Last
Name', userEmail AS 'User Email'

FROM USER

JOIN ORDERS ON USER.userID = ORDERS.userID

JOIN ASSEMBLY ON ASSEMBLY.orderID = ORDERS.orderID

JOIN PRODUCT ON PRODUCT.prodID = ASSEMBLY.prodID

WHERE prodBrand = 'Kate Spade' AND EXISTS(SELECT * FROM PRODUCT
WHERE PRODUCT.prodID = ASSEMBLY.prodID);
```

Query 9

```
# Report the users' cities and the number of orders in each
city.

SELECT city AS 'City', COUNT(ASSEMBLY.orderID) AS 'Number of
Orders'

FROM ADDRESS

JOIN USER ON USER.userID = ADDRESS.userID

JOIN ORDERS ON ORDERS.userID = USER.userID

JOIN ASSEMBLY ON ASSEMBLY.orderID = ORDERS.orderID
```

```
GROUP BY city;
```

Query 10

```
# Report the product names and the number of users who bought
each product.

 SELECT prodName AS 'Product Name', COUNT(USER.userID) AS
'Number of Users'

 FROM PRODUCT

 JOIN ASSEMBLY ON ASSEMBLY.prodID = PRODUCT.prodID

 JOIN ORDERS ON ORDERS.orderID = ASSEMBLY.orderID

 JOIN USER ON USER.userID = ORDERS.userID

 GROUP BY PRODUCT.prodID;
```

MySQL Account Information

Account name: ns_Group3

Password: Group3_ns