



- A user can tweet (read purchase) how many ever shoes he/she wants and add them to the shopping cart. Hence the 'Shopping\_Cart' has an 'order\_id' which is the primary key of the table (since it uniquely distinguishes each order) and multiple 't\_order\_id' for an order. Note that each order in the shopping cart can have more than one Twitter order.

### **SQL Statements for the conceptual model:**

#### **User Table:**

```
CREATE TABLE `User` (
  `Twitter_handle` VARCHAR(10),
  `name` VARCHAR(20),
  `profile_image_url` VARCHAR(200),
  `description` VARCHAR(100),
  `followers_count` INT,
  `following_count` INT,
  PRIMARY KEY (`Twitter_handle`)
);
```

#### **Tweets Table:**

```
CREATE TABLE `Tweets` (
  `tweet_id` INT NOT NULL AUTO_INCREMENT,
  `Twitter_handle` VARCHAR(10),
  `tweet_text` VARCHAR(140),
  `profile_image_url` VARCHAR(200),
  `created_at` DATETIME,
  PRIMARY KEY (`tweet_id`)
);
```

#### **Tweet Tags Table:**

```
CREATE TABLE `Tweet_Tags` (
  `tweet_id` INT NOT NULL,
  `tags` VARCHAR(20),
  PRIMARY KEY (`tweet_id`)
);
```

#### **Tweet Mentions Table:**

```
CREATE TABLE `Tweet_Mentions` (
  `tweet_id` INT NOT NULL,
  `source_user` VARCHAR(10),
```

```
`target_user` VARCHAR(10),  
PRIMARY KEY (`tweet_id`)  
);
```

#### **Tweet\_Url Table:**

```
CREATE TABLE `Tweet_Url` (  
  `tweet_id` INT NOT NULL,  
  `product_url` VARCHAR(200)  
  PRIMARY KEY (`tweet_id`)  
);
```

#### **DSW\_Account Table:**

```
CREATE TABLE `DSW_Account` (  
  `Twitter_handle` VARCHAR(10) NOT NULL,  
  `password` VARCHAR(10),  
  `dsw_points` INT  
  PRIMARY KEY (`Twitter_handle`)  
);
```

#### **Womens\_Shoes Table:**

```
CREATE TABLE `Womens_Shoes` (  
  `product_id` INT NOT NULL AUTO_INCREMENT,  
  `product_name` VARCHAR(20),  
  `product_url` VARCHAR(200),  
  `price` FLOAT  
  PRIMARY KEY (`product_id`)  
);
```

#### **Twitter\_Order Table:**

```
CREATE TABLE `Twitter_Order` (  
  `t_order_id` INT NOT NULL AUTO_INCREMENT,  
  `tweet_id` INT,  
  `Twitter_handle` VARCHAR,  
  `product_id` INT,  
  `product_url` VARCHAR(200)  
  `price` FLOAT  
  PRIMARY KEY (`t_order_id`)  
);
```

### **Shopping\_Cart Table:**

```
CREATE TABLE `Shopping_Cart` (  
  `order_id` INT NOT NULL AUTO_INCREMENT,  
  `total_amount` FLOAT,  
  `Twitter_handle` VARCHAR(10),  
  `no_of_items` INT,  
  PRIMARY KEY (`order_id`)  
);
```

### **Payment Table:**

```
CREATE TABLE `Payment` (  
  `payment_id` INT NOT NULL AUTO_INCREMENT,  
  `order_id` INT,  
  `Twitter_handle` VARCHAR(10),  
  `paid_on` DATETIME,  
  `total_amount` FLOAT,  
  `payment_details` VARCHAR,  
  PRIMARY KEY (`payment_id`)  
);
```

### **Adding Foreign Key Constraint:**

#### **Constraint for Tweet table:**

```
ALTER TABLE `Tweets`  
ADD CONSTRAINT `Tweets_fk1` FOREIGN KEY (`Twitter_handle`)  
REFERENCES User(`Twitter_handle`);
```

```
ALTER TABLE `Tweets`  
ADD CONSTRAINT `Tweets_fk2` FOREIGN KEY (`profile_image_url`)  
REFERENCES User(`profile_image_url`);
```

#### **Constraint for Tweet\_Tags table:**

```
ALTER TABLE `Tweet_Tags`  
ADD CONSTRAINT `Tweet_Tags_fk1` FOREIGN KEY (`tweet_id`)  
REFERENCES Tweets(`tweet_id`);
```

### **Constraint for Tweet\_Mentions table:**

```
ALTER TABLE `Tweet_Mentions`  
ADD CONSTRAINT `Tweet_Mentions_fk1` FOREIGN KEY (`tweet_id`)  
REFERENCES Tweets(`tweet_id`);  
  
ALTER TABLE `Tweet_Mentions`  
ADD CONSTRAINT `Tweet_Mentions_fk2` FOREIGN KEY (`source_user`)  
REFERENCES User(`Twitter_handle`);  
  
ALTER TABLE `Tweet_Mentions`  
ADD CONSTRAINT `Tweet_Mentions_fk3` FOREIGN KEY (`target_user`)  
REFERENCES User(`Twitter_handle`);
```

### **Constraint for Tweet\_Url table:**

```
ALTER TABLE `Tweet_Url`  
ADD CONSTRAINT `Tweet_Url_fk1` FOREIGN KEY (`tweet_id`)  
REFERENCES Tweets(`tweet_id`);
```

### **Constraint for DSW\_Account table:**

```
ALTER TABLE `DSW_Account`  
ADD CONSTRAINT `DSW_Account_fk1` FOREIGN KEY (`Twitter_handle`)  
REFERENCES User(`Twitter_handle`);
```

### **Constraint for Twitter\_Order table:**

```
ALTER TABLE `Twitter_Order`  
ADD CONSTRAINT `Twitter_Order_fk1` FOREIGN KEY (`tweet_id`)  
REFERENCES Tweet_Url(`tweet_id`);  
  
ALTER TABLE `Twitter_Order`  
ADD CONSTRAINT `Twitter_Order_fk2` FOREIGN KEY (`Twitter_handle`)  
REFERENCES Tweets(`Twitter_handle`);
```

```

ALTER TABLE `Twitter_Order`
ADD CONSTRAINT `Twitter_Order_fk3` FOREIGN KEY (`product_id`)
REFERENCES Womens_Shoes(`product_id`);

ALTER TABLE `Twitter_Order`
ADD CONSTRAINT `Twitter_Order_fk4` FOREIGN KEY (`product_url`)
REFERENCES Tweet_url(`product_url`);

ALTER TABLE `Twitter_Order`
ADD CONSTRAINT `Twitter_Order_fk5` FOREIGN KEY (`price`)
REFERENCES Womens_Shoes(`price`);

```

### **Constraint for Shopping\_Cart table:**

```

ALTER TABLE `Shopping_Cart`
ADD CONSTRAINT `Shopping_Cart_fk1` FOREIGN KEY (`Twitter_handle`)
REFERENCES Twitter_Order(`Twitter_handle`);

ALTER TABLE `Shopping_Cart`
ADD CONSTRAINT `Shopping_Cart_fk2` FOREIGN KEY (`t_order_id`)
REFERENCES Twitter_Order(`t_order_id`);

ALTER TABLE `Shopping_Cart`
ADD CONSTRAINT `Shopping_Cart_fk3` FOREIGN KEY (`Twitter_handle`)
REFERENCES Tweet_Order(`Twitter_handle`);

```

### **Constraint for Payment table:**

```

ALTER TABLE `Payment`
ADD CONSTRAINT `Payment_fk1` FOREIGN KEY (`order_id`)
REFERENCES Shopping_Cart(`order_id`);

ALTER TABLE `Payment`
ADD CONSTRAINT `Payment_fk2` FOREIGN KEY (`Twitter_handle`)
REFERENCES Shopping_Cart(`Twitter_handle`);

```

## USE-CASE

1. **Use Case:** Register for an account in DSW  
**Description:** User registers for an account in DSW  
**Actor:** User  
**Precondition:** When a customer wants to buy something from shop, firstly he will be registered  
**Steps:**  
**Actor action:** User request for registration  
**System Responses:** If customer information is correct then customer is registered and use case ends.  
**Post Condition:** Customer successfully registered  
**Alternate Path:** The customer request is not correct and system throws an error  
**Error:** User information is incorrect
2. **Use Case:** Make an order in DSW  
**Description:** User makes an order of a product in DSW store  
**Actors:** User  
**Precondition:** User must have a unique Twitter handle to tweet  
**Steps:**  
**Actor action** – User tweets about a product to order along with the product URL  
**System Responses** – An order is made for the product that matches the product URL  
**Post Condition:** An order is added to Twitter\_Order table for the product the user tweeted.  
**Alternate Path:** The product not currently available in the store  
**Error:** Product Not Available
3. **Use Case:** View a product already ordered through Twitter by a user  
**Description:** User views a product already ordered  
**Actors:** User  
**Precondition:** User must have made an order  
**Steps:**  
**Actor action** – User views a product from its URL  
**System Responses** – product URL would be displayed  
**Post Condition:** system displays product URL
4. **Use Case:** View the products above a particular price (say \$100)  
**Description:** Use views the products above a particular price  
**Actor:** User  
**Precondition:**  
**Steps:**  
**Actor action:** User views the products above a particular price  
**System Responses:** the list of products above a price are displayed  
**Post Condition:** system displays the list of products for the condition
5. **Use Case:** View the orders made by a user  
**Description:** User views the orders made by him/her  
**Actor:** User  
**Precondition:** User must have made at least one order to view an order  
**Steps:**  
**Actor action:** User views the history of orders  
**System Responses:** Displays all the orders made by a user  
**Alternate Path:** There are no orders made by a user  
**Error:** No history of orders available.

## RELATIONAL-ALGEBRA EXPRESSIONS FOR THE USE CASES

1. Use Case: View a product already ordered through Twitter

$$\Pi_{\{w.product\_url\}}(\sigma_{\{w.product\_id = t.product\_id \wedge t.Twitter\_handle = '@alice'\}}(\rho_{\{w\}}(Womens\_Shoes) \times \rho_{\{t\}}(Twitter\_Order)))$$

2. Use Case: View the products above a particular price (say \$100)

$$\Pi_{\{w.product\_url, w.product\_url\}}(\sigma_{\{w.price > 100\}}(Womens\_Shoes))$$

3. Use Case: View the orders made by a user

$$\Pi_{\{s.Twitter\_handle, s.order\_id\}}(\sigma_{\{s.Twitter\_handle = '@bob'\}}(Shopping\_Cart))$$

## SQL STATEMENTS

1. Use Case: Register for an account in DSW

```
INSERT INTO DSW_Account
(Twitter_handle, password, dsw_points)
VALUES (@john, john123, 0)
```

```
INSERT INTO DSW_Account
(Twitter_handle, password, dsw_points)
VALUES (@alice, alice123, 0);
```

```
INSERT INTO DSW_Account
(Twitter_handle, password, dsw_points)
VALUES (@bob, bob123, 0);
```

2. Use Case: Make an order in DSW

```
INSERT INTO Tweet
(tweet_id, Twitter_handle, tweet_text, profile_image_url, created_at )
VALUES (12321, @john, 'I would like to purchase
www.dsw.com/shoe/product_id=2341' , 'www.facebook.com/john.smith/
photo.php?fbid=10205' , 02-02-2015 );
```

```
INSERT INTO Tweet_url
(tweet_id, tweet_url )
VALUES (12321, 'www.dsw.com/shoe/product_id=2341');
```

```
INSERT INTO Twitter_Order
(t_order_id, tweet_id, Twitter_handle, product_id, product_url, price)
VALUES (4532, 12321, @john, 2341, , 'www.dsw.com/shoe/product_id=2341', 26.4 )
```

```
INSERT INTO Shopping_Cart
(order_id, t_order_id, no_of_items, total_amount, Twitter_handle)
VALUES ( 9876, 4532, 1, 26.4, @john )
```



3. Use Case: View a product already ordered through Twitter

```
SELECT w.product_url
FROM Womens_Shoes w, Twitter_Order t
WHERE
t.product_id = w.product_id AND
t.Twitter_handle = '@alice'
```

4. Use Case: View the products above a particular price (say \$100)

```
SELECT w.product_name, w.product_url
FROM Womens_Shoes w
WHERE
w.price > 100;
```

5. Use Case: View the orders made by a user

```
SELECT s.Twitter_handle, s.order_id
FROM Shopping_Cart s
WHERE
s.Twitter_handle = '@bob';
```