

# Term Project – Deliverable 5

Team Members: Dhiraj Pakhrin, Satkrit Bhattarai

1.

There is one change in our current business rule. The updated business rules are mentioned below:

- One seller can sell multiple products, but one product is associated with only one seller.
- One product can have multiple inventory records, but each inventory record corresponds to only one product.
- One consumer can place zero or multiple orders, but one order is associated with only one consumer.
- One order can contain multiple products, and each product can be present in multiple orders.
- One order can have one shipment, and one shipment is associated with one order.
- One Amazon warehouse contains multiple products (**Changed**)
- One shipment originates from one Amazon warehouse, and one Amazon warehouse can have multiple shipments.

2.

The changes that I have made to the business rule are as follows:

1. Added " product\_catagory " as an attribute in the Product table
2. Added "product\_price" as an attribute in the Product table
3. Added "warehouse\_location" as an attribute in the warehouse table
4. Added "w\_qty" as an attribute to the Warehouse table
5. Added "Status" as an attribute to the Warehouse table
6. Added "Customer\_first\_name" as an attribute to the customer table
7. Added "Customer\_last\_name" as an attribute to the customer table
8. Added "Customer\_email" as an attribute to the customer table

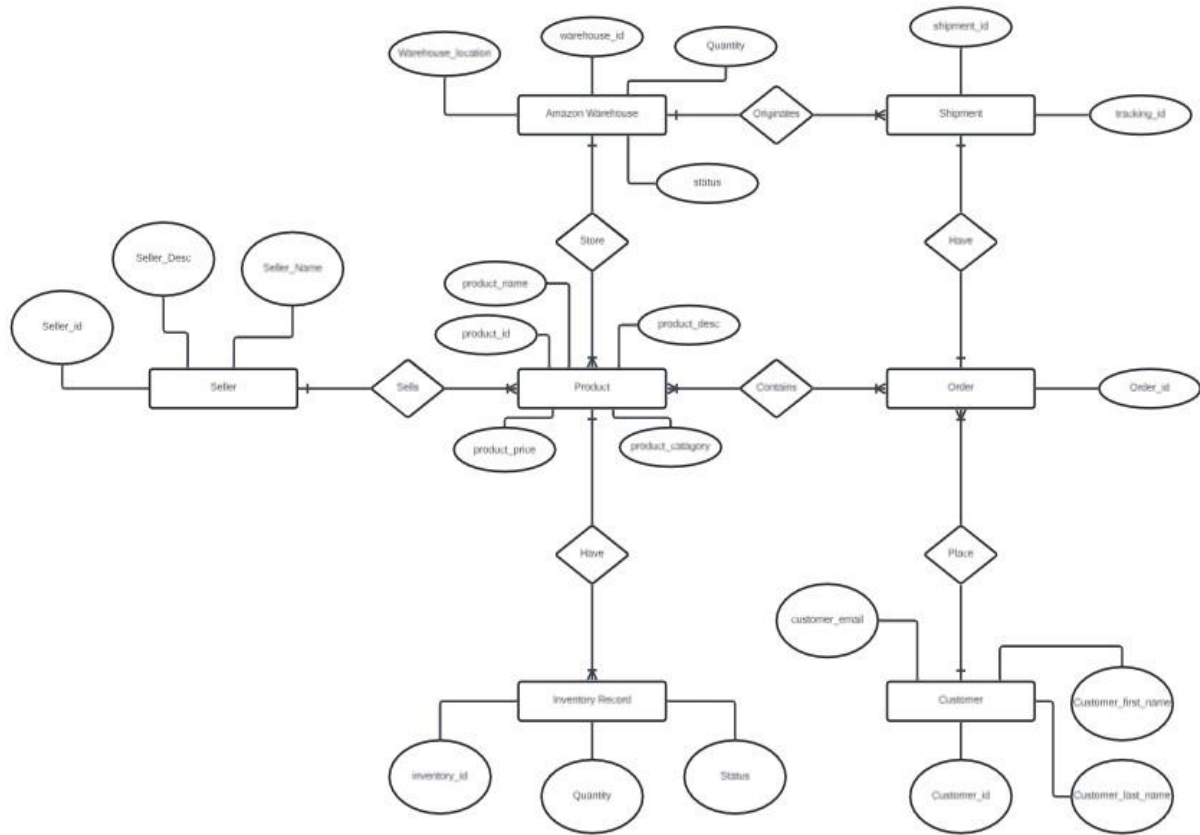


Fig: Conceptual ER-Diagram of our new business rule

3. Make any enhancements and corrections to your logical ERD and provide the updated ERD below.

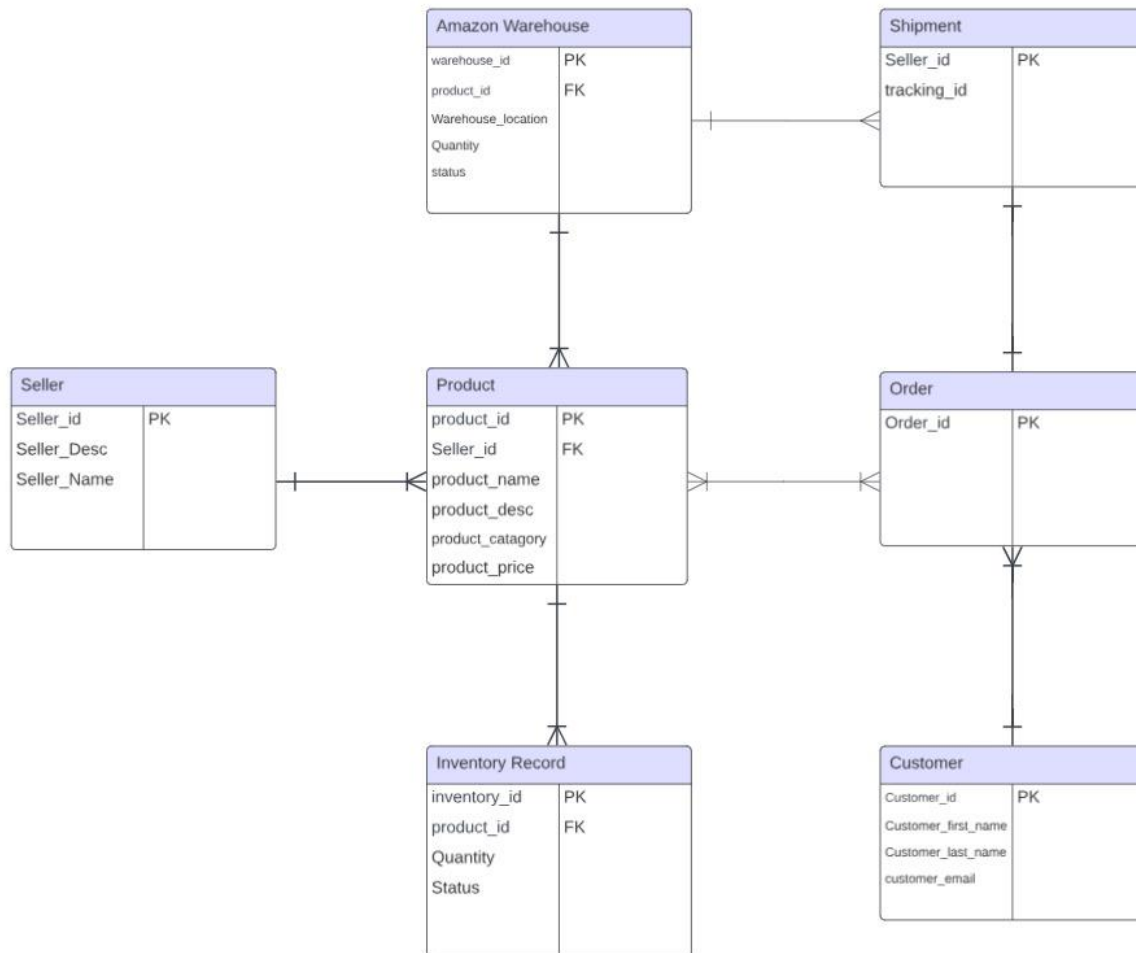



Fig: Conceptual ER-Diagram of New Business Rule

4.

## Aspect 2

### a. Creation of tables and constraints



```
16
17 CREATE TABLE Warehouse (
18   warehouse_id NUMBER(10) PRIMARY KEY,
19   warehouse_location VARCHAR2(255),
20   product_id NUMBER(10),
21   status VARCHAR2(255),
22   w_qty NUMBER(10),
23   FOREIGN KEY (product_id) REFERENCES Product(product_id)
24 );
25
26 CREATE TABLE Inventory (
27   inventory_id NUMBER(10) PRIMARY KEY,
28   product_id NUMBER(10),
29   qty NUMBER(10),
30   status VARCHAR2(255),
31   FOREIGN KEY (product_id) REFERENCES Product(product_id)
32 );
33
```

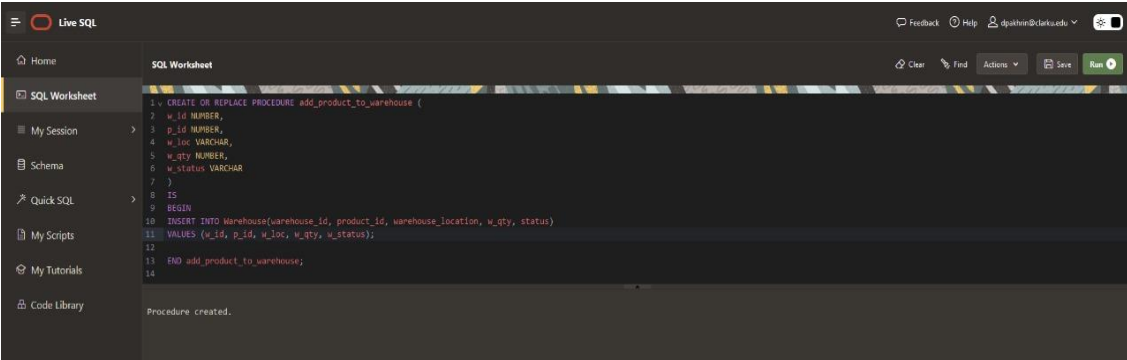
Table created.

Table created.

Table created.

Table created.

### b. Creation of reusable stored procedure



```
1 CREATE OR REPLACE PROCEDURE add_product_to_warehouse (
2   w_id NUMBER,
3   p_id NUMBER,
4   w_loc VARCHAR,
5   w_qty NUMBER,
6   w_status VARCHAR
7 )
8 IS
9 BEGIN
10  INSERT INTO Warehouse(warehouse_id, product_id, warehouse_location, w_qty, status)
11  VALUES (w_id, p_id, w_loc, w_qty, w_status);
12
13 END add_product_to_warehouse;
14
```

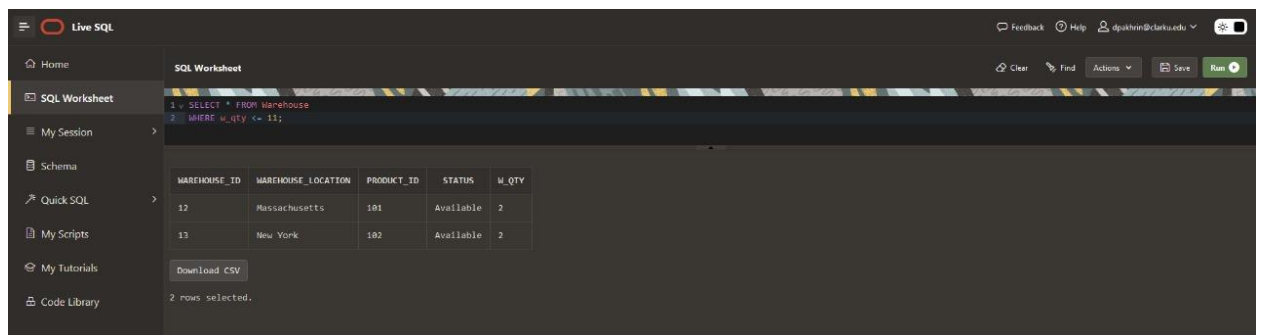
Procedure created.

c. Use of the stored procedure



The screenshot shows the 'Live SQL' interface. The left sidebar contains a menu with 'Home', 'SQL Worksheet', 'My Session', 'Schema', 'Quick SQL', and 'My Scripts'. The 'SQL Worksheet' is selected. The main area shows two SQL statements:   
1. `CALL add_product_to_warehouse(12, 101, 'Massachusetts', 2, 'Available');`  
2. `CALL add_product_to_warehouse(13, 102, 'New York', 2, 'Available');`  
Below the statements, it says 'Statement processed.' and 'Statement processed.'.

d. SQL query



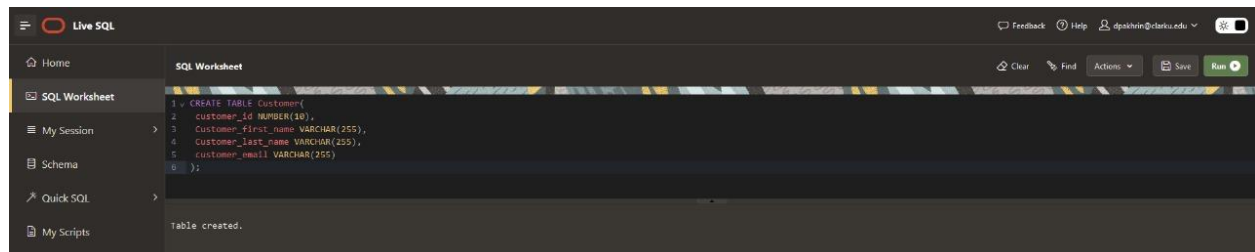
The screenshot shows the 'Live SQL' interface. The left sidebar is the same as in the previous screenshot. The main area shows a SQL query:   
1. `SELECT * FROM Warehouse`  
2. `WHERE w_qty <= 11;`  
Below the query, a table of results is displayed:

WAREHOUSE_ID	WAREHOUSE_LOCATION	PRODUCT_ID	STATUS	W_QTY
12	Massachusetts	101	Available	2
13	New York	102	Available	2

Below the table, there is a 'Download CSV' button and the text '2 rows selected.'

## Aspect 3

### a. Creation of tables and constraints

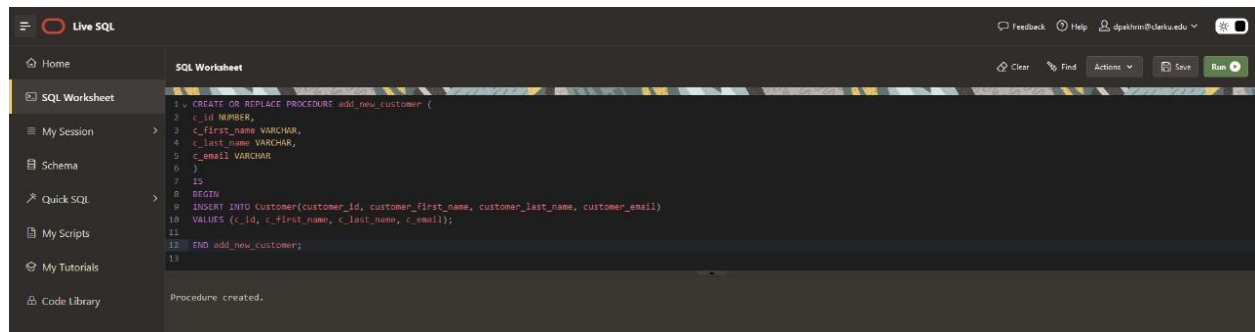


The screenshot shows the 'Live SQL' interface. On the left sidebar, 'SQL Worksheet' is selected. The main editor area contains the following SQL code:

```
1. CREATE TABLE Customer(  
2.   customer_id NUMBER(10),  
3.   customer_first_name VARCHAR(255),  
4.   customer_last_name VARCHAR(255),  
5.   customer_email VARCHAR(255)  
6. );
```

Below the code editor, the status bar indicates 'Table created.'

### b. Creation of reusable stored procedure

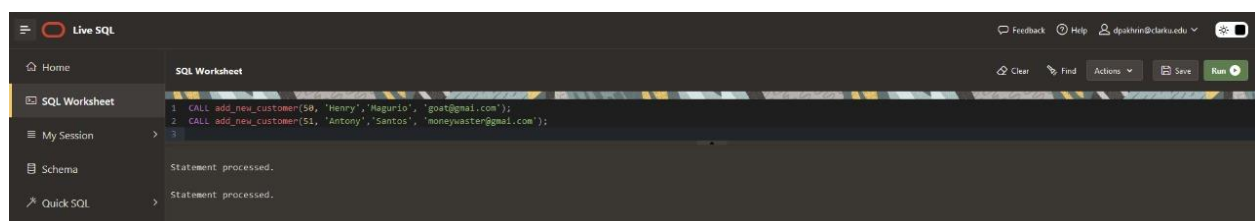


The screenshot shows the 'Live SQL' interface. On the left sidebar, 'SQL Worksheet' is selected. The main editor area contains the following SQL code:

```
1. CREATE OR REPLACE PROCEDURE add_new_customer (  
2.   c_id NUMBER,  
3.   c_first_name VARCHAR,  
4.   c_last_name VARCHAR,  
5.   c_email VARCHAR  
6. )  
7. IS  
8. BEGIN  
9.   INSERT INTO Customer(customer_id, customer_first_name, customer_last_name, customer_email)  
10.  VALUES (c_id, c_first_name, c_last_name, c_email);  
11.  
12. END add_new_customer;  
13.
```

Below the code editor, the status bar indicates 'Procedure created.'

### c. Use of the stored procedure



The screenshot shows the 'Live SQL' interface. On the left sidebar, 'SQL Worksheet' is selected. The main editor area contains the following SQL code:

```
1. CALL add_new_customer(58, 'Henry', 'Maurio', 'goat@gmail.com');  
2. CALL add_new_customer(51, 'Antony', 'Santos', 'moneywaster@gmail.com');  
3.
```

Below the code editor, the status bar indicates 'Statement processed.'

## d. SQL query

Live SQL

Feedback Help dpashin@clarku.edu

Home SQL Worksheet Clear Find Actions Save Run

```
1. select * from customer
2. where customer_last_name = 'Thapa';
3
4
```

CUSTOMER_ID	CUSTOMER_FIRST_NAME	CUSTOMER_LAST_NAME	CUSTOMER_EMAIL
50	Alex	Thapa	alex@gmail.com
50	Ferry	Thapa	ferry@gmail.com
50	Shiro	Thapa	shiro@gmail.com
50	Sergio	Thapa	sergio@gmail.com

Download CSV

4 rows selected.