SQL MINI PROJECT

E-Commercial System (Orcal SQL + Java)

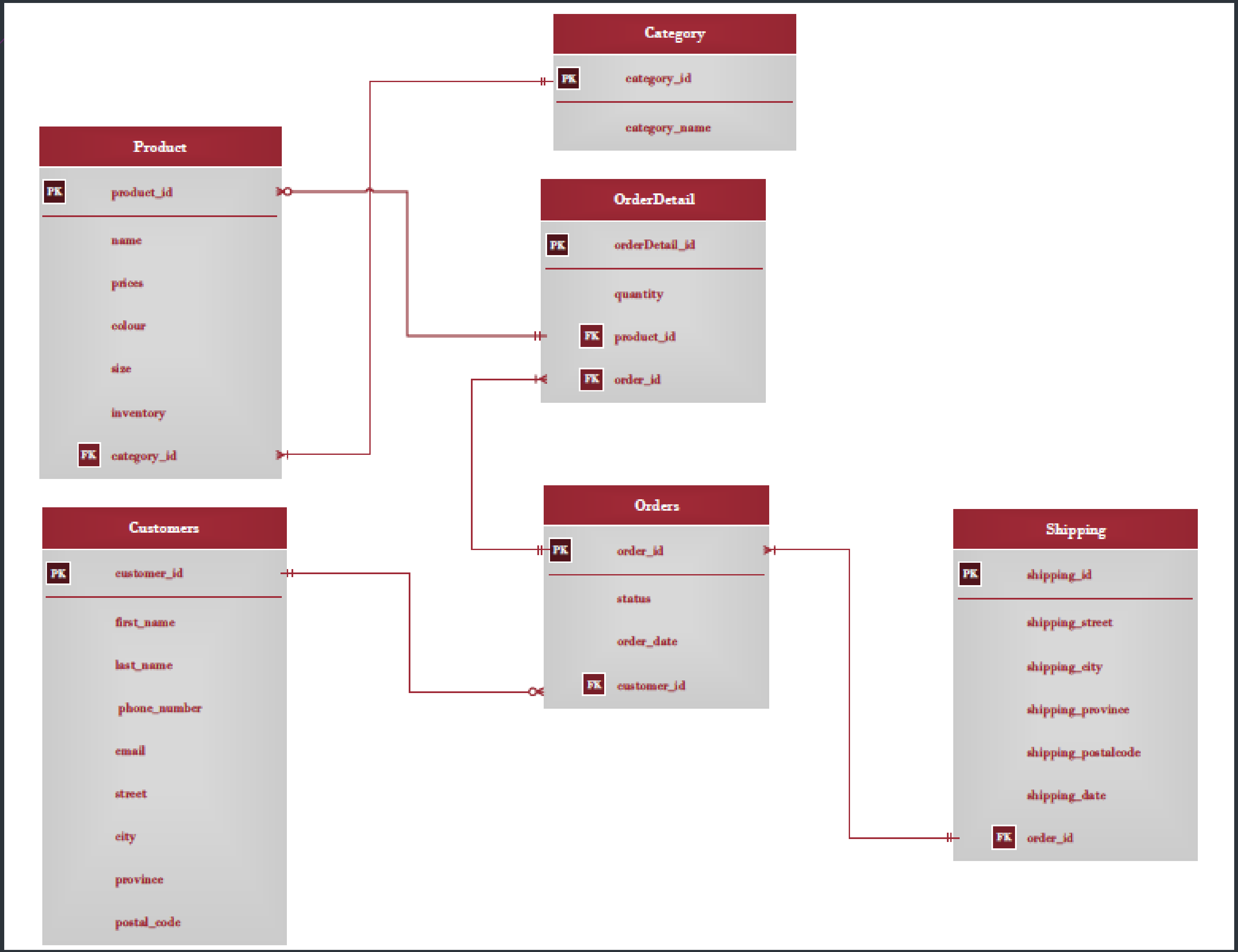
Author: Jason(Jabin) Choi

CONTENTS

PROBLEM DOMAIN DESCRIPTION

- **Problem Description**
  - E-commercial company selling clothes without databases
  - No accurate data analysis
  - Decrease in company profits
  - Outdated company image that can not catch up with trends
- **Potential Approach**
  - Building well-organized databases through PL/SQL
  - A total of six tables (Each has its own attributes)
- **Related Work**
  - Sequence
  - Trigger
  - Index
  - Procedure / Function

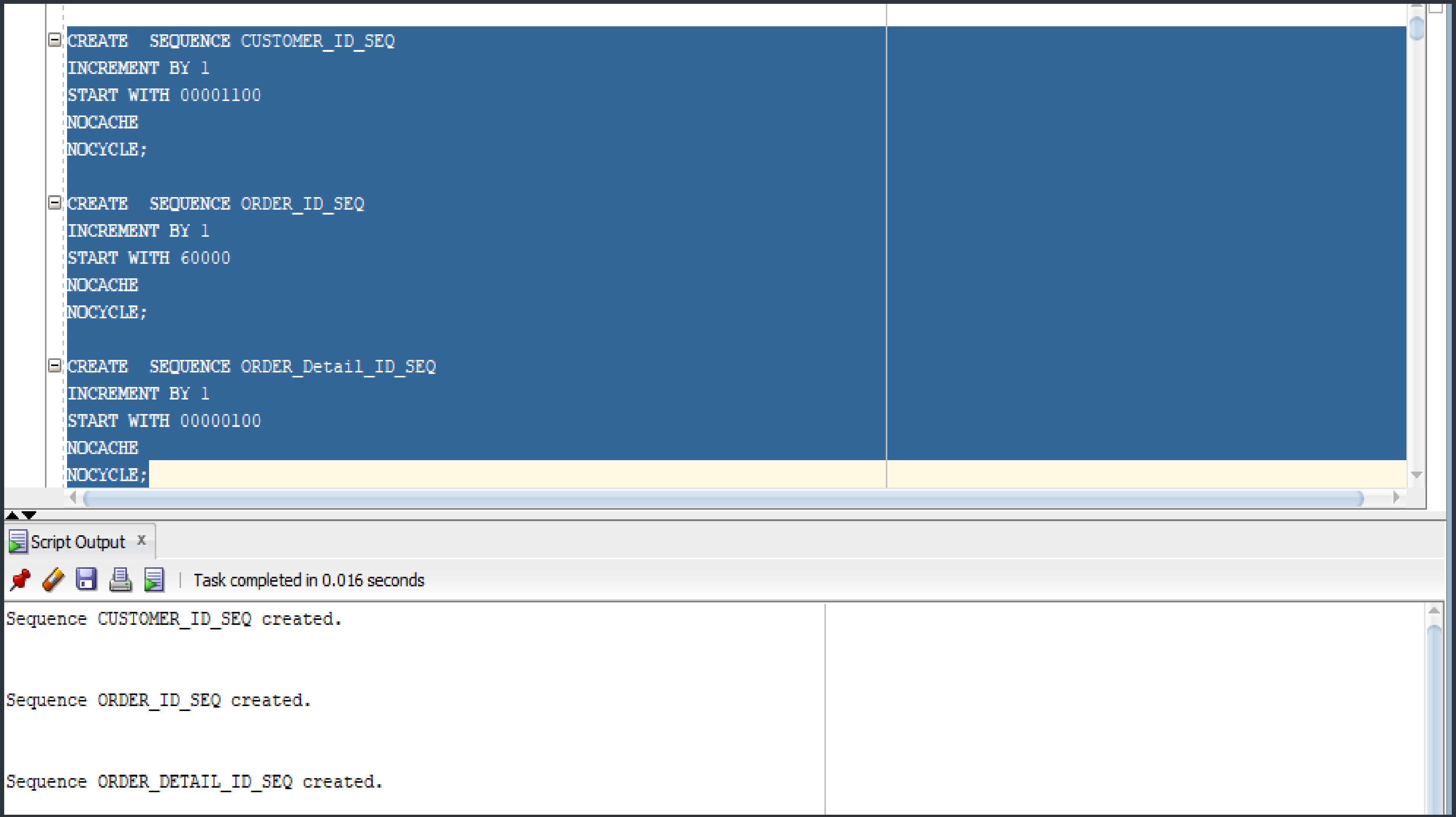
ENTITY-RELATIONSHIP DIAGRAM (ERD)



SEQUENCE

```
CREATE  SEQUENCE CUSTOMER_ID_SEQ
INCREMENT BY 1
START WITH 00001100
NOCACHE
NOCYCLE;

CREATE  SEQUENCE ORDER_ID_SEQ
INCREMENT BY 1
START WITH 60000
NOCACHE
NOCYCLE;

CREATE  SEQUENCE ORDER_Detail_ID_SEQ
INCREMENT BY 1
START WITH 00000100
NOCACHE
NOCYCLE;
```

SEQUENCE



```
CREATE  SEQUENCE CUSTOMER_ID_SEQ
INCREMENT BY 1
START WITH 00001100
NOCACHE
NOCYCLE;

CREATE  SEQUENCE ORDER_ID_SEQ
INCREMENT BY 1
START WITH 60000
NOCACHE
NOCYCLE;

CREATE  SEQUENCE ORDER_Detail_ID_SEQ
INCREMENT BY 1
START WITH 00000100
NOCACHE
NOCYCLE;
```

Task completed in 0.016 seconds

```
Sequence CUSTOMER_ID_SEQ created.


Sequence ORDER_ID_SEQ created.


Sequence ORDER_DETAIL_ID_SEQ created.
```

CREATE A TABLE

Customer Table

```
--/Customer/

CREATE TABLE Customer
( customer_id  NUMBER(8) Default CUSTOMER_ID_SEQ.NEXTVAL,
  first_name VARCHAR2(20),
  last_name VARCHAR2(20),
  phone_number VARCHAR2 (15),
  email VARCHAR2(50),
  street VARCHAR2(50),
  city VARCHAR2(20),
  province CHAR(2),
  postalCode CHAR(7),
  CONSTRAINT customer_customerID_pk PRIMARY KEY ( customer_id )
);
```

CREATE A TABLE

Category Table

```
--/Category/

CREATE TABLE Category
( category_id NUMBER(8) PRIMARY KEY,
  category_name VARCHAR(20)
);
```

CREATE A TABLE

Product Table

```
--/Product/

CREATE TABLE Product
( product_id NUMBER(8) PRIMARY KEY,
  product_name VARCHAR2(20),
  prices NUMBER(8, 2),
  colour VARCHAR2(20),
  product_size VARCHAR2(10),
  inventory VARCHAR2(10),
  category_id NUMBER(8) REFERENCES Category ( category_id )
);
```

CREATE A TABLE

Orders Table

```
--/Orders/

CREATE TABLE Orders
( order_id NUMBER(8) Default ORDER_ID_SEQ.NEXTVAL PRIMARY KEY,
  status VARCHAR2(10),
  order_date DATE,
  customer_id NUMBER(8) REFERENCES Customer ( customer_id )
  );
```

CREATE A TABLE

OrdersDetail Table

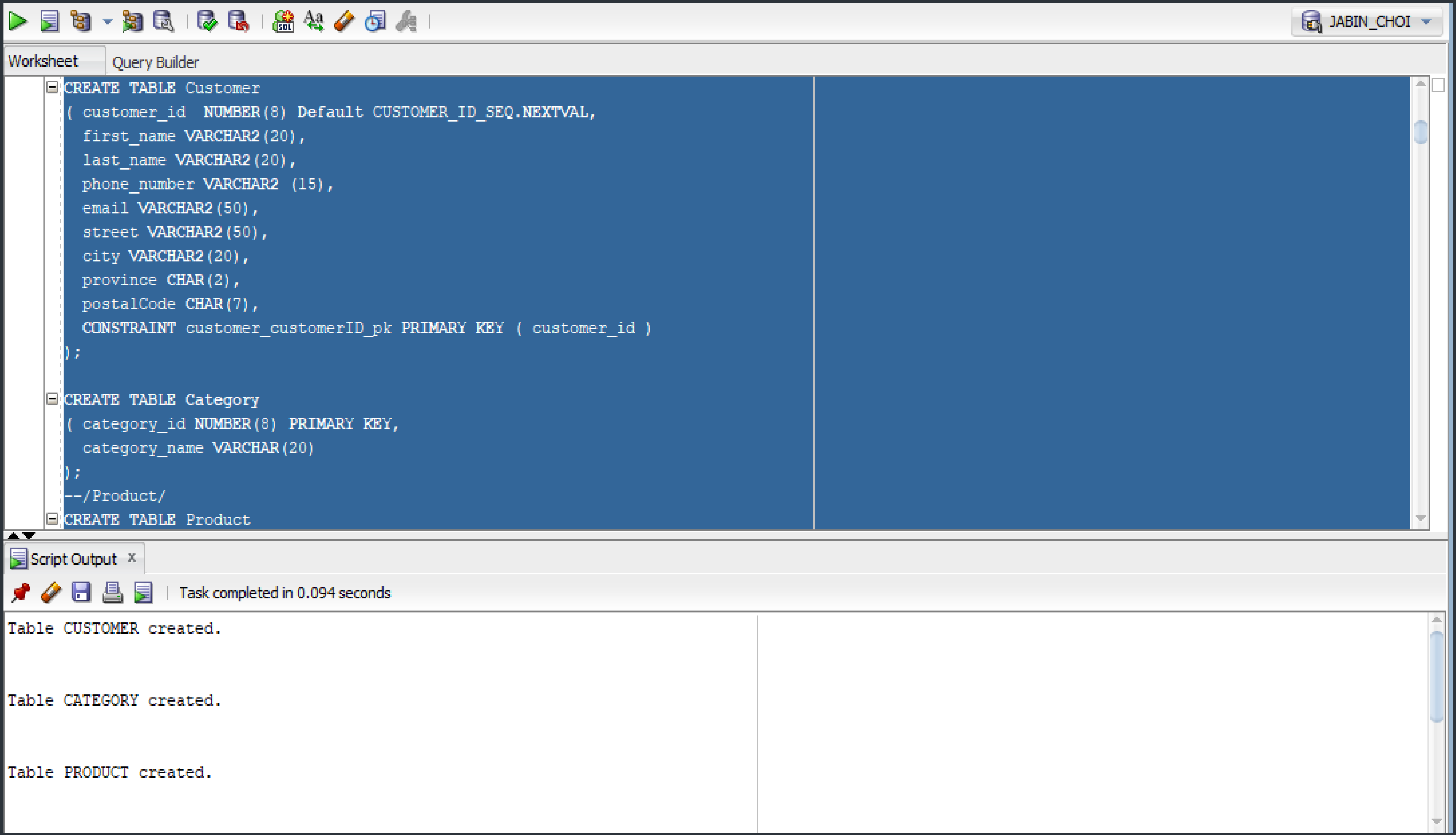
```
--/OrdersDetail/

CREATE TABLE OrderDetail
( orderDetail_id NUMBER(8) Default ORDER_Detail_ID_SEQ.NEXTVAL PRIMARY KEY,
  quantity NUMBER(3),
  order_id NUMBER(8) REFERENCES orders ( order_id ),
  product_id NUMBER(8) REFERENCES Product ( product_id )
);
```

CREATE A TABLE

Shipping Table

```
--/Shipping/

CREATE TABLE Shipping
( shipping_id NUMBER(8)PRIMARY KEY,
  shipping_street VARCHAR2(50),
  shipping_city VARCHAR2(20),
  shipping_province CHAR(2),
  shipping_postalcode CHAR(7),
  shipping_date DATE,
  order_id NUMBER(8) REFERENCES Orders (order_id)
);
```

CREATE A TABLE



```sql
CREATE TABLE Customer
( customer_id  NUMBER(8) Default CUSTOMER_ID_SEQ.NEXTVAL,
  first_name VARCHAR2(20),
  last_name VARCHAR2(20),
  phone_number VARCHAR2 (15),
  email VARCHAR2(50),
  street VARCHAR2(50),
  city VARCHAR2(20),
  province CHAR(2),
  postalCode CHAR(7),
  CONSTRAINT customer_customerID_pk PRIMARY KEY ( customer_id )
);

CREATE TABLE Category
( category_id NUMBER(8) PRIMARY KEY,
  category_name VARCHAR(20)
);
--/Product/
CREATE TABLE Product
```

```
Table CUSTOMER created.


Table CATEGORY created.


Table PRODUCT created.
```

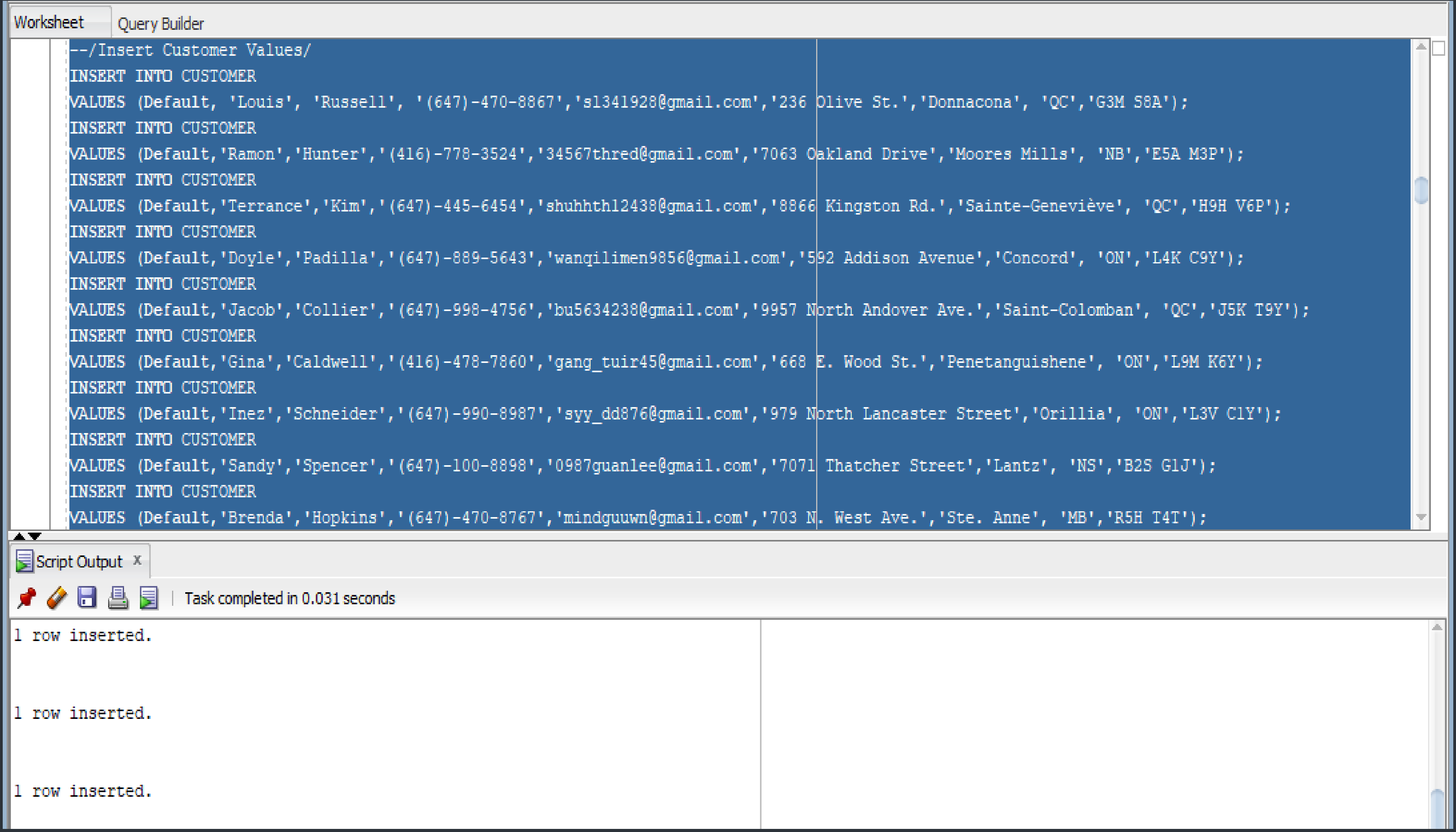
INSERT A VALUE

```
--/Insert Customer Values/
INSERT INTO CUSTOMER
VALUES (Default, 'Louis', 'Russell', '(647)-470-8867','sl341928@gmail.com','236 Olive
St.','Donnacona', 'QC','G3M S8A');
INSERT INTO CUSTOMER
VALUES (Default,'Ramon','Hunter','(416)-778-3524','34567thred@gmail.com','7063 Oakland
Drive','Moores Mills', 'NB','E5A M3P');
INSERT INTO CUSTOMER
VALUES (Default,'Terrance','Kim','(647)-445-6454','shuhhth12438@gmail.com','8866 Kingston
Rd.','Sainte-Geneviève', 'QC','H9H V6P');
INSERT INTO CUSTOMER
VALUES (Default,'Doyle','Padilla','(647)-889-5643','wanqilimen9856@gmail.com','592 Addison
Avenue','Concord', 'ON','L4K C9Y');
INSERT INTO CUSTOMER
VALUES (Default,'Jacob','Collier','(647)-998-4756','bu5634238@gmail.com','9957 North
Andover Ave.','Saint-Colomban', 'QC','J5K T9Y');
INSERT INTO CUSTOMER
VALUES (Default,'Gina','Caldwell','(416)-478-7860','gang_tuir45@gmail.com','668 E. Wood
St.','Penetanguishene', 'ON','L9M K6Y’);
.
.
.
```

INSERT A VALUE

```
--/Insert Product Values/
INSERT INTO product
VALUES (20001, 'Talence Blouse','87.00','white','XXS', '13', 00001003);
INSERT INTO product
VALUES (20002, 'Best Primrose Skirt','67.00','Ashen','S','11', 00001006);
INSERT INTO product
VALUES (20003, 'Best Weller Dress','120.00','Oak','M','10', 00001001);
INSERT INTO product
VALUES (20004, 'Howley Jeans','78.00','Reddish','XXS','8', 00001008);
INSERT INTO product
VALUES (20005, 'Fitted Mini Dress','120.00','Cordovan','S','24', 00001001);
INSERT INTO product
VALUES (20006, 'Coleridge T-Shirt','35.00','Grey','S','32', 00001005);
INSERT INTO product
VALUES (20007, 'Free Subah Pant','140.00','Fatigue','L','11', 00001002);
INSERT INTO product
VALUES (20008,'Best Carly Dress','127.00','Cardamon','M', '8',00001001);
INSERT INTO product
VALUES (20009, 'Free Jamilla T-Shirt','30.00','Dew Blue','XS','13', 00001003);
INSERT INTO product
VALUES (20010, 'Lance Cardigan','225.00','White','XXS','2', 00001004);
INSERT INTO product
VALUES (20011, 'Gap hoody', '15.00', 'Dark Gray', 'S', '0', 00001005);
```

INSERT A VALUE



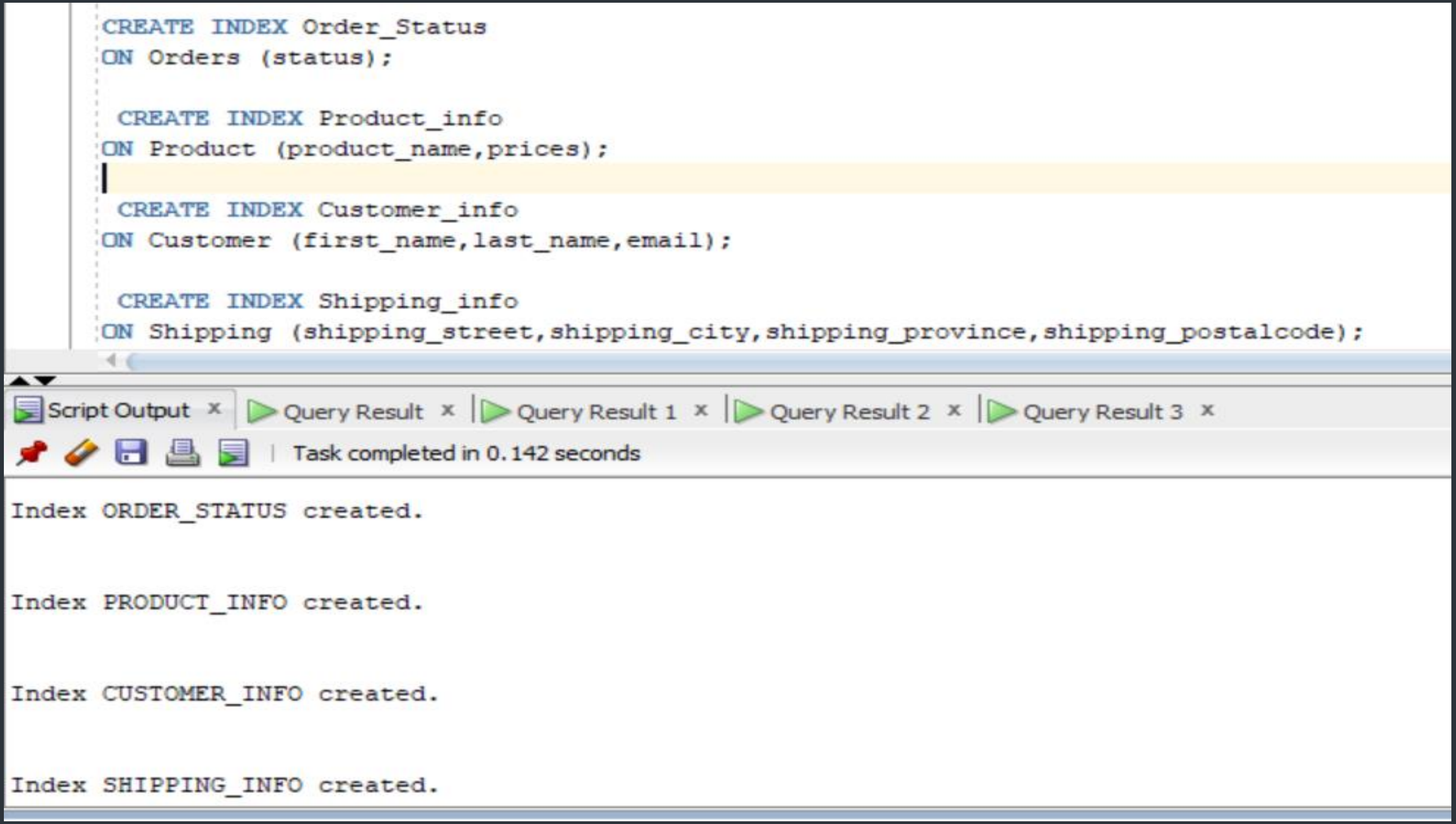
```
--/Insert Customer Values/
INSERT INTO CUSTOMER
VALUES (Default, 'Louis', 'Russell', '(647)-470-8867','s1341928@gmail.com','236 Olive St.','Donnacona', 'QC','G3M S8A');
INSERT INTO CUSTOMER
VALUES (Default,'Ramon','Hunter','(416)-778-3524','34567thred@gmail.com','7063 Oakland Drive','Moores Mills', 'NB','E5A M3P');
INSERT INTO CUSTOMER
VALUES (Default,'Terrance','Kim','(647)-445-6454','shuhhth12438@gmail.com','8866 Kingston Rd.','Sainte-Geneviève', 'QC','H9H V6P');
INSERT INTO CUSTOMER
VALUES (Default,'Doyle','Padilla','(647)-889-5643','wanqilimen9856@gmail.com','592 Addison Avenue','Concord', 'ON','L4K C9Y');
INSERT INTO CUSTOMER
VALUES (Default,'Jacob','Collier','(647)-998-4756','bu5634238@gmail.com','9957 North Andover Ave.','Saint-Colomban', 'QC','J5K T9Y');
INSERT INTO CUSTOMER
VALUES (Default,'Gina','Caldwell','(416)-478-7860','gang_tuir45@gmail.com','668 E. Wood St.','Penetanguishene', 'ON','L9M K6Y');
INSERT INTO CUSTOMER
VALUES (Default,'Inez','Schneider','(647)-990-8987','syy_dd876@gmail.com','979 North Lancaster Street','Orillia', 'ON','L3V C1Y');
INSERT INTO CUSTOMER
VALUES (Default,'Sandy','Spencer','(647)-100-8898','0987guanlee@gmail.com','7071 Thatcher Street','Lantz', 'NS','B2S G1J');
INSERT INTO CUSTOMER
VALUES (Default,'Brenda','Hopkins','(647)-470-8767','mindguuwn@gmail.com','703 N. West Ave.','Ste. Anne', 'MB','R5H T4T');
```

```
1 row inserted.


1 row inserted.


1 row inserted.
```

INDEX

```
CREATE INDEX Order_Status
ON Orders (status);

 CREATE INDEX Product_info
ON Product (product_name,prices);

 CREATE INDEX Customer_info
ON Customer (first_name,last_name,email);

 CREATE INDEX Shipping_info
ON Shipping (shipping_street,shipping_city,shipping_province,shipping_postalcode);
```

INDEX



```
CREATE INDEX Order_Status
ON Orders (status);

 CREATE INDEX Product_info
ON Product (product_name,prices);

 CREATE INDEX Customer_info
ON Customer (first_name,last_name,email);

 CREATE INDEX Shipping_info
ON Shipping (shipping_street,shipping_city,shipping_province,shipping_postalcode);
```

Task completed in 0.142 seconds

```
Index ORDER_STATUS created.


Index PRODUCT_INFO created.


Index CUSTOMER_INFO created.


Index SHIPPING_INFO created.
```

TRIGGER

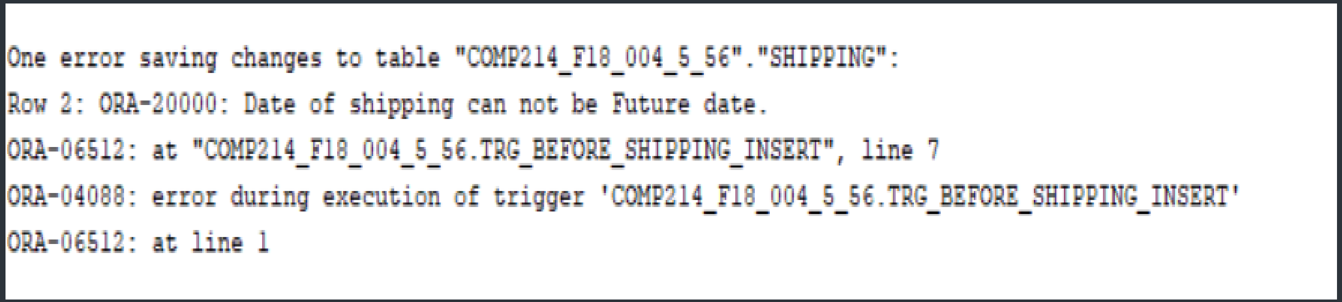
First Trigger

```
--/Trigger 01/

CREATE OR REPLACE TRIGGER
trg_before_order_insert
BEFORE INSERT
        on orders
        FOR EACH ROW
DECLARE
BEGIN
        -- Allow only past date of shipping
        IF(:new.order_date > sysdate) THEN
                RAISE_APPLICATION_ERROR(-20000,'Date of order can not be Future date.');
        END IF;
END;
```

TRIGGER

First Trigger

```
--/Trigger 01/

CREATE OR REPLACE TRIGGER
trg_before_order_insert
BEFORE INSERT
      on orders
      FOR EACH ROW
DECLARE
BEGIN
      -- Allow only past date of shipping
      IF(:new.order_date > sysdate) THEN
            RAISE_APPLICATION_ERROR(-20000,'Date of order can not be Future date.');
      END IF;
END;
```

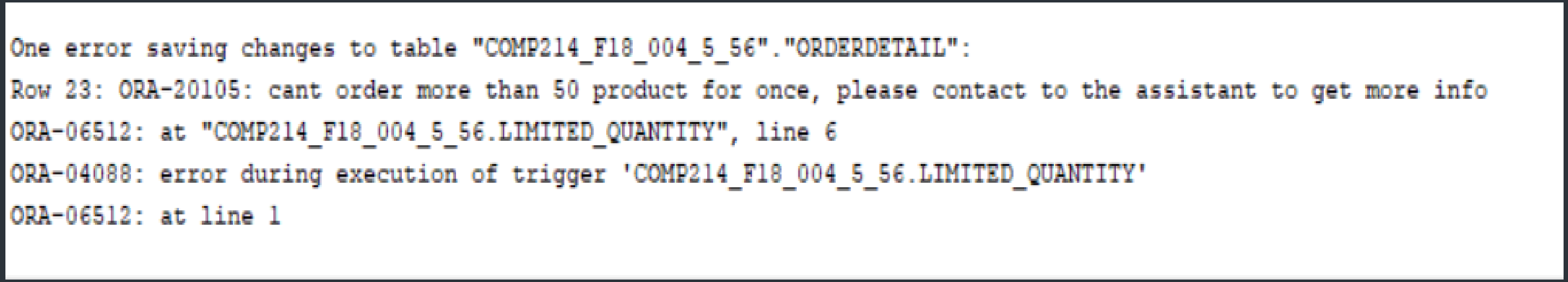


```
One error saving changes to table "COMP214_F18_004_5_56"."SHIPPING":
Row 2: ORA-20000: Date of shipping can not be Future date.
ORA-06512: at "COMP214_F18_004_5_56.TRG_BEFORE_SHIPPING_INSERT", line 7
ORA-04088: error during execution of trigger 'COMP214_F18_004_5_56.TRG_BEFORE_SHIPPING_INSERT'
ORA-06512: at line 1
```

TRIGGER

Second Trigger

```
--/Trigger 02/

CREATE OR REPLACE TRIGGER limited_quantity
BEFORE INSERT ON orderdetail
for each row
DECLARE
BEGIN
	IF :new.quantity > 50 THEN
		RAISE_APPLICATION_ERROR(-20105,'cant order more than 50 product for once, please contact to the assistant to get more info');
	END IF;
END;
```

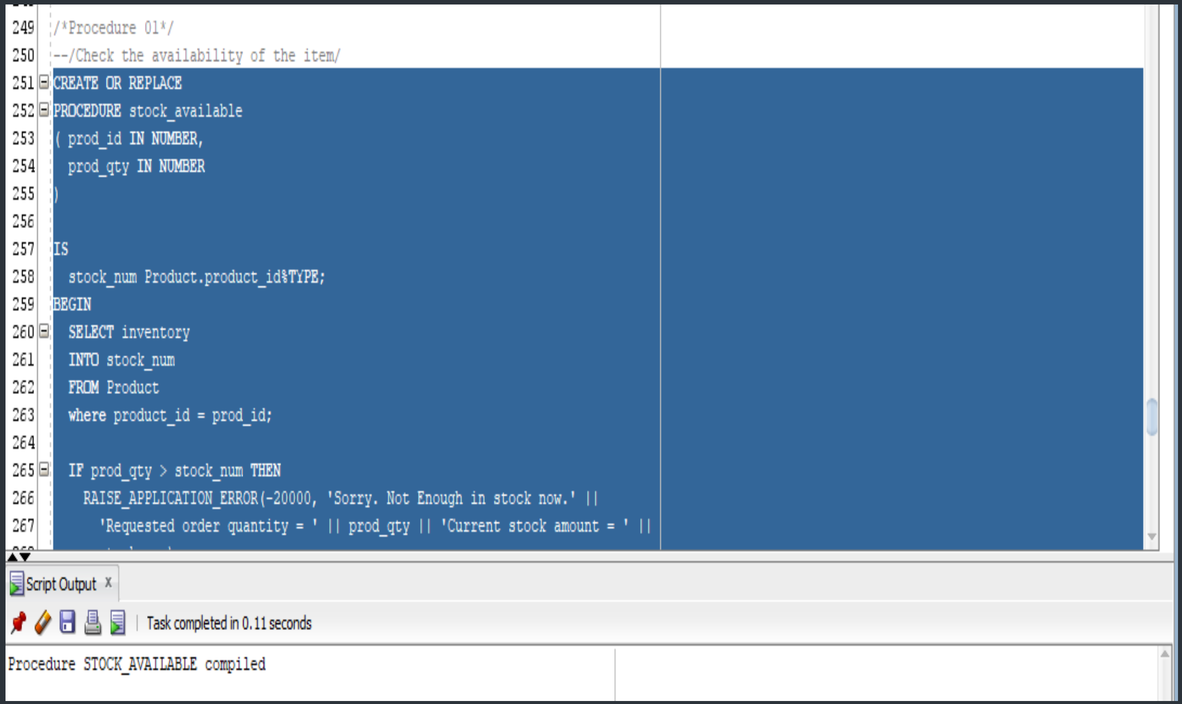


```
One error saving changes to table "COMP214_F18_004_5_56"."ORDERDETAIL":
Row 23: ORA-20105: cant order more than 50 product for once, please contact to the assistant to get more info
ORA-06512: at "COMP214_F18_004_5_56.LIMITED_QUANTITY", line 6
ORA-04088: error during execution of trigger 'COMP214_F18_004_5_56.LIMITED_QUANTITY'
ORA-06512: at line 1
```

PROCEDURE

```
/*Procedure 01*/
--/Check the availability of the item/
CREATE OR REPLACE
PROCEDURE stock_available
( prod_id IN NUMBER,
  prod_qty IN NUMBER
)
IS
        stock_num Product.product_id%TYPE;
BEGIN
        SELECT inventory
        INTO stock_num
        FROM Product
        where product_id = prod_id;

        IF prod_qty > stock_num THEN
                RAISE_APPLICATION_ERROR(-20000, 'Sorry. Not Enough in stock now.' ||
                        'Requested order quantity = ' || prod_qty || 'Current stock amount = ‘ || stock_num);
        END IF;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
                DBMS_OUTPUT.PUT_LINE('No Stock Found.');
END;
```

First Procedure



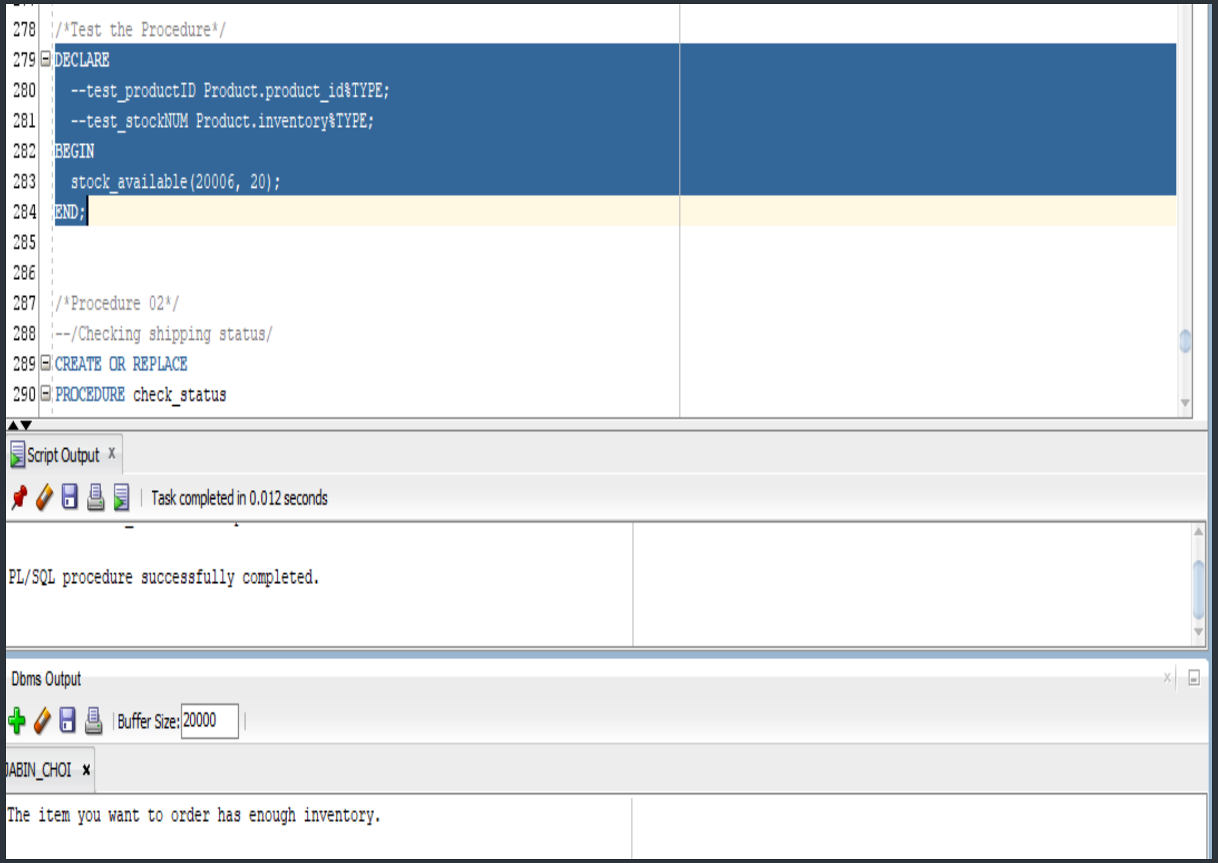
```
/*Procedure 01*/
--/Check the availability of the item/
CREATE OR REPLACE
PROCEDURE stock_available
( prod_id IN NUMBER,
  prod_qty IN NUMBER
)

IS
  stock_num Product.product_id%TYPE;
BEGIN
  SELECT inventory
  INTO stock_num
  FROM Product
  where product_id = prod_id;

  IF prod_qty > stock_num THEN
    RAISE_APPLICATION_ERROR(-20000, 'Sorry. Not Enough in stock now.' ||
      'Requested order quantity = ' || prod_qty || 'Current stock amount = ' ||
```

```
Task completed in 0.11 seconds

Procedure STOCK_AVAILABLE compiled
```

PROCEDURE

Test First Procedure

```
/*Test the Procedure 01*/
BEGIN
	stock_available(20006, 20);
END;
```



```
/*Test the Procedure*/
DECLARE
  --test_productID Product.product_id%TYPE;
  --test_stockNUM Product.inventory%TYPE;
BEGIN
  stock_available(20006, 20);
END;


/*Procedure 02*/
--/Checking shipping status/
CREATE OR REPLACE
PROCEDURE check_status
```

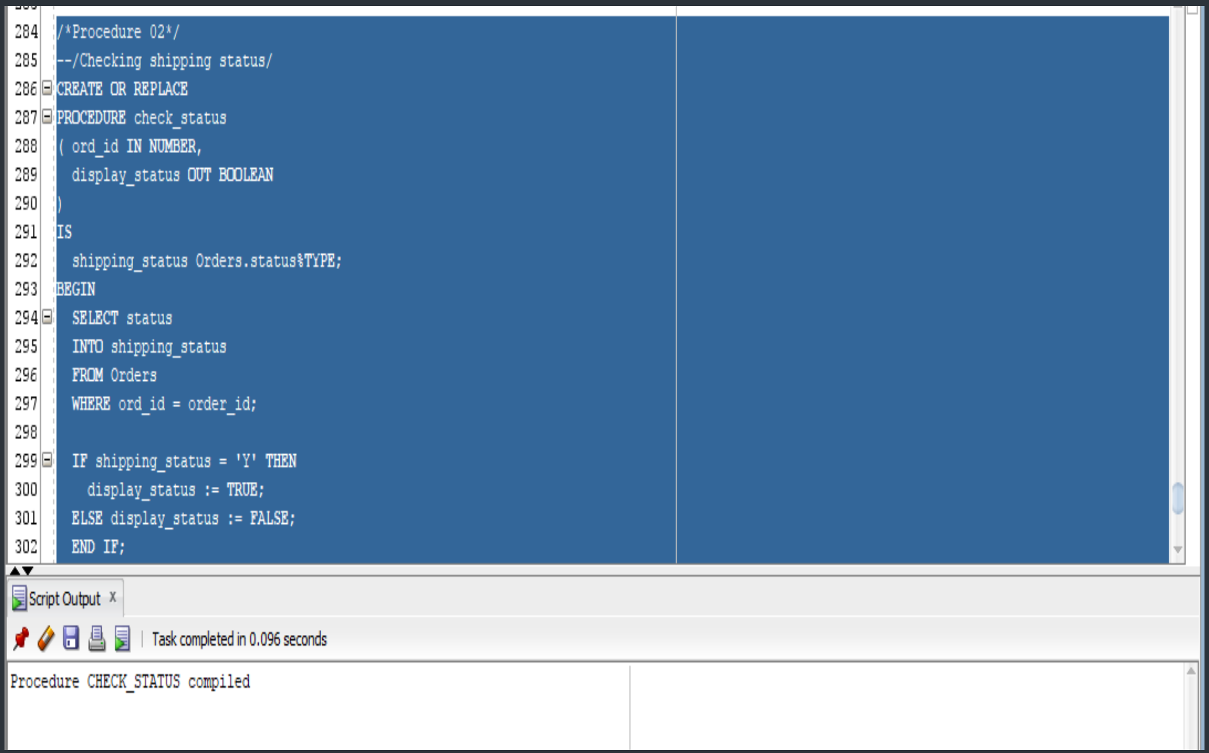
```
PL/SQL procedure successfully completed.
```

```
The item you want to order has enough inventory.
```

PROCEDURE

```
/*Procedure 02*/
--/Checking shipping status/
CREATE OR REPLACE
PROCEDURE check_status
( ord_id IN NUMBER,
  display_status OUT BOOLEAN
)
IS
        shipping_status Orders.status%TYPE;
BEGIN
        SELECT status
        INTO shipping_status
        FROM Orders
        WHERE ord_id = order_id;

        IF shipping_status = 'Y' THEN
                display_status := TRUE;
        ELSE display_status := FALSE;
        END IF;
        DBMS_OUTPUT.PUT_LINE('Is the selected item in transit ?' | | ' ' | |
                                CASE display_status
                                WHEN TRUE THEN 'Yes. In transit.'
                                ELSE 'Not Yet.'
                                END);
END;
```

Second Procedure



```
/*Procedure 02*/
--/Checking shipping status/
CREATE OR REPLACE
PROCEDURE check_status
( ord_id IN NUMBER,
  display_status OUT BOOLEAN
)
IS
  shipping_status Orders.status%TYPE;
BEGIN
  SELECT status
  INTO shipping_status
  FROM Orders
  WHERE ord_id = order_id;

  IF shipping_status = 'Y' THEN
    display_status := TRUE;
  ELSE display_status := FALSE;
  END IF;
```

Script Output ×

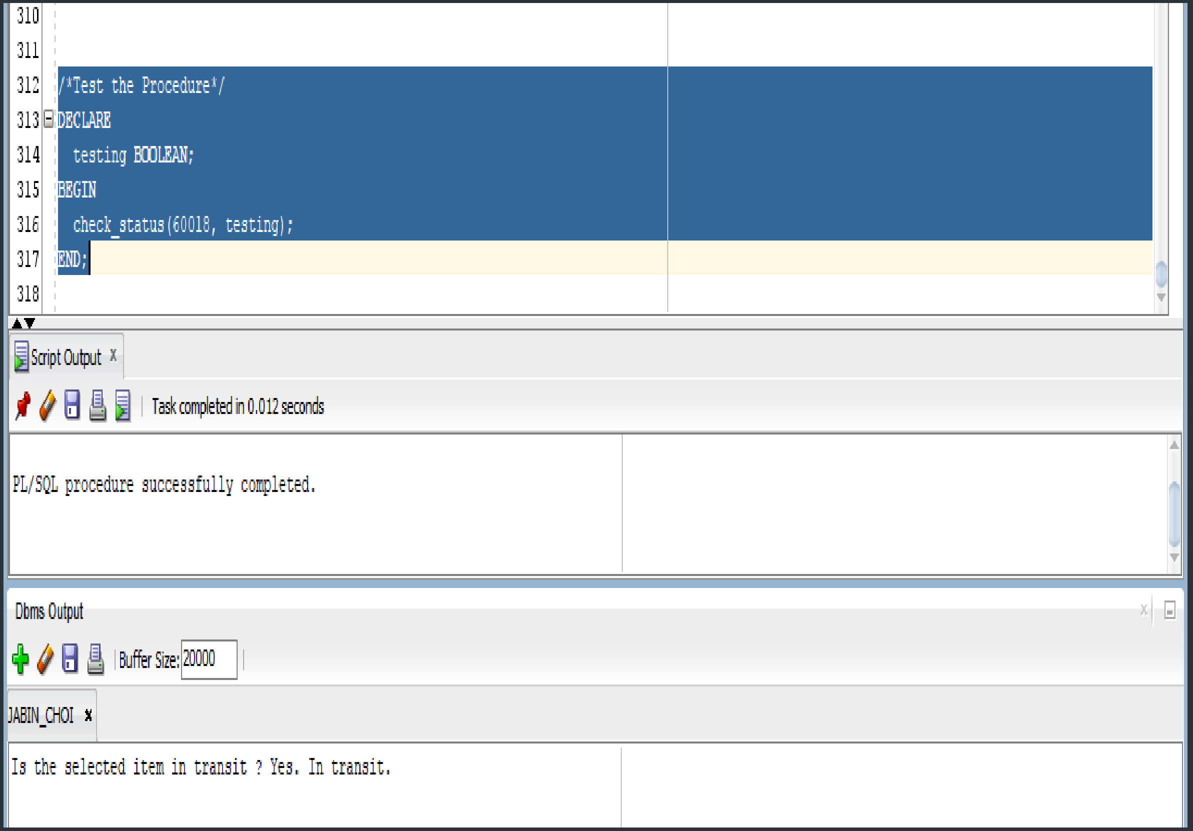
Task completed in 0.096 seconds

```
Procedure CHECK_STATUS compiled
```

PROCEDURE

Test Second Procedure

```
/*Test the Procedure 02*/
DECLARE
	testing BOOLEAN;
BEGIN
	check_status(60018, testing);
END;
```



```
/*Test the Procedure*/
DECLARE
  testing BOOLEAN;
BEGIN
  check_status(60018, testing);
END;
```

Script Output

Task completed in 0.012 seconds

```
PL/SQL procedure successfully completed.
```

Dbms Output

Buffer Size: 20000

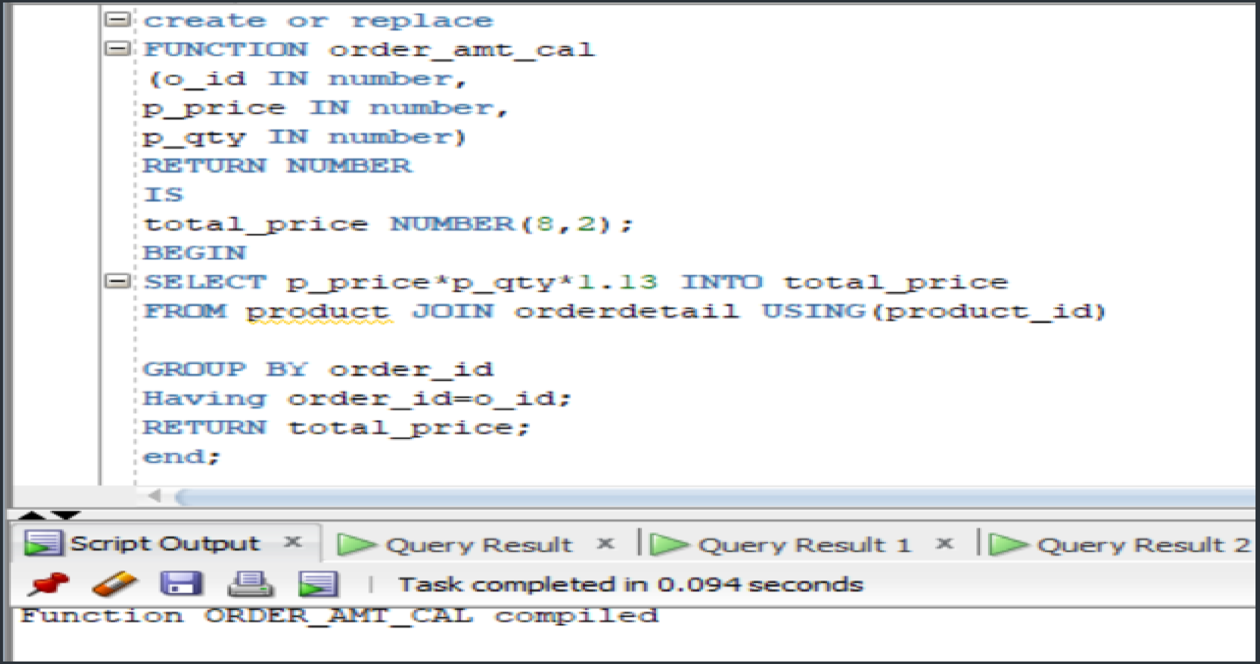
JABIN_CHOI

```
Is the selected item in transit ? Yes. In transit.
```

FUNCTION

First Function

```
--/Function 01/
CREATE OR REPLACE
FUNCTION order_amt_cal
(o_id IN number,
p_price IN number,
p_qty IN number
)
RETURN NUMBER
IS
	total_price NUMBER(8,2);
BEGIN
	SELECT p_price*p_qty*1.13 INTO total_price
	FROM product JOIN orderdetail USING(product_id)
	GROUP BY order_id
	Having order_id=o_id;
	RETURN total_price;
END;
```



```
create or replace
FUNCTION order_amt_cal
 (o_id IN number,
p_price IN number,
p_qty IN number)
RETURN NUMBER
IS
total_price NUMBER(8,2);
BEGIN
SELECT p_price*p_qty*1.13 INTO total_price
FROM product JOIN orderdetail USING(product_id)

GROUP BY order_id
Having order_id=o_id;
RETURN total_price;
end;
```

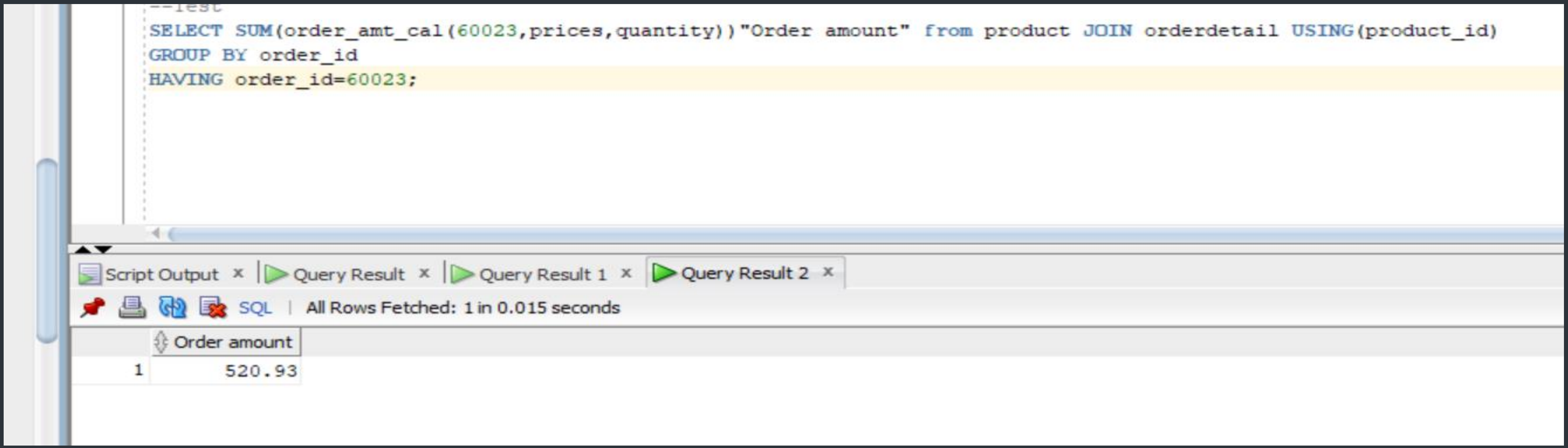
```
Task completed in 0.094 seconds
Function ORDER_AMT_CAL compiled
```

FUNCTION

Test First Function

/*Test Function 01*/

SELECT SUM(order_amt_cal(60023,prices,quantity))"Order amount" from product JOIN orderdetail USING(product_id)
GROUP BY order_id
HAVING order_id=60023;



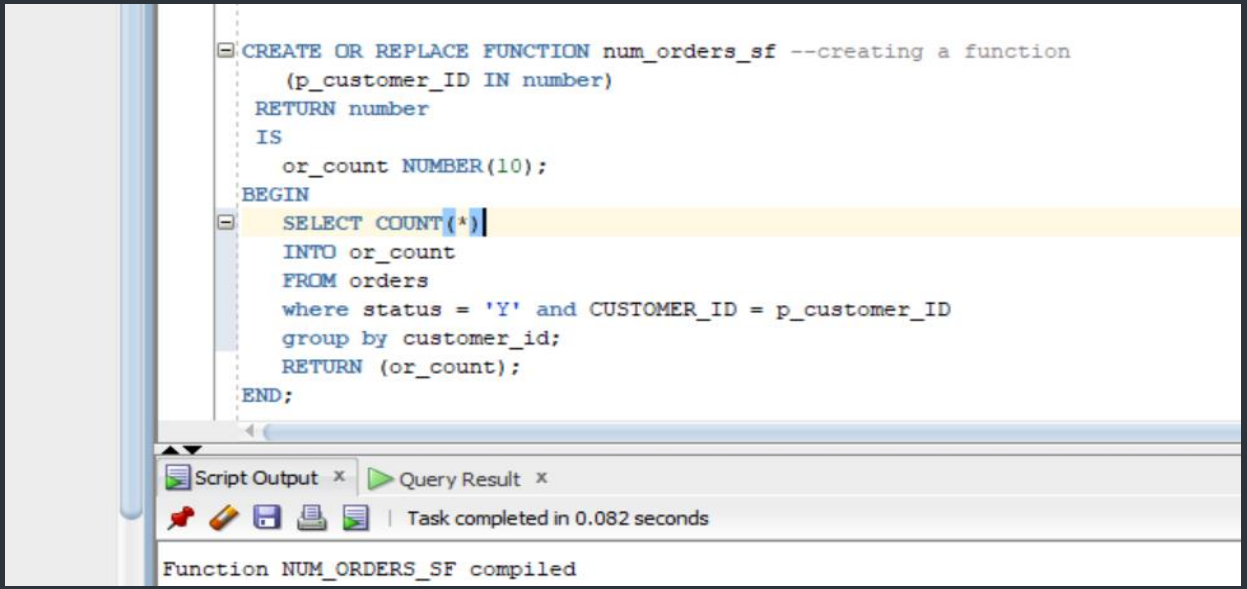
```
--Test
SELECT SUM(order_amt_cal(60023,prices,quantity))"Order amount" from product JOIN orderdetail USING(product_id)
GROUP BY order_id
HAVING order_id=60023;
```

```
All Rows Fetched: 1 in 0.015 seconds

   Order amount
1        520.93
```

FUNCTION

Second Function

```
--/Function 02/
CREATE OR REPLACE FUNCTION num_orders_sf
(p_customer_ID IN number
)
RETURN number

IS
      or_count NUMBER(10);
BEGIN
      SELECT COUNT(*)
      INTO or_count
      FROM orders
      WHERE status = 'Y'
              AND CUSTOMER_ID = p_customer_ID
      GROUP BY customer_id;
      RETURN (or_count);
END;
```



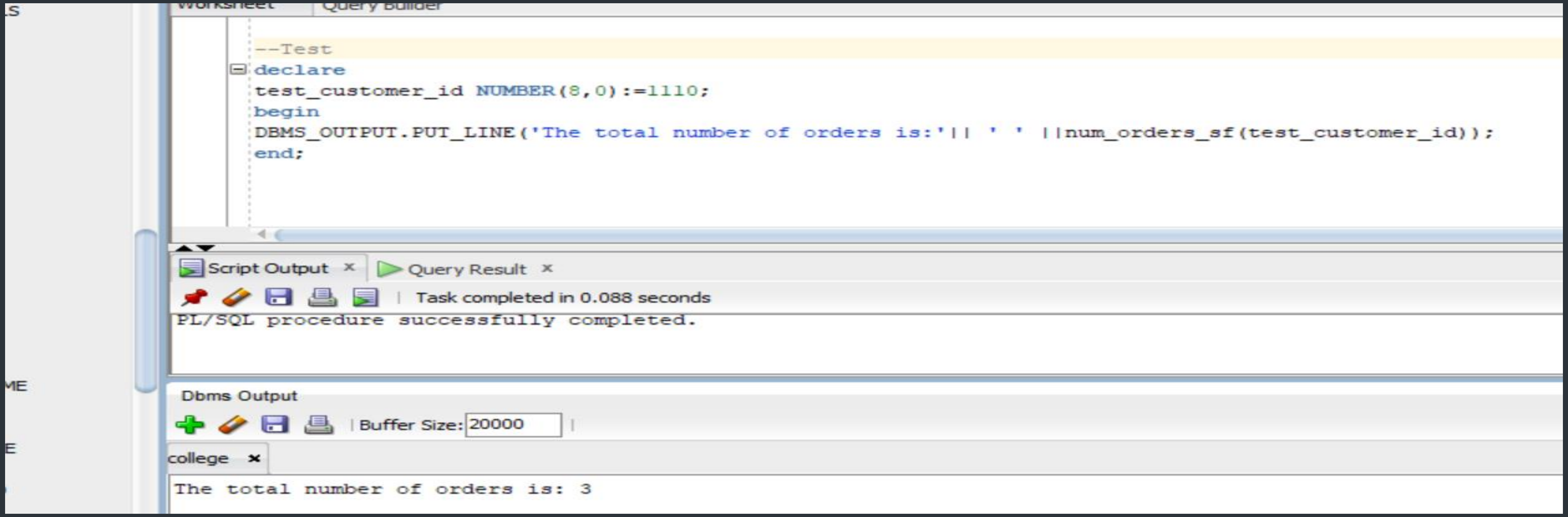
```
CREATE OR REPLACE FUNCTION num_orders_sf --creating a function
   (p_customer_ID IN number)
 RETURN number
 IS
   or_count NUMBER(10);
BEGIN
   SELECT COUNT(*)
   INTO or_count
   FROM orders
   where status = 'Y' and CUSTOMER_ID = p_customer_ID
   group by customer_id;
   RETURN (or_count);
END;
```

```
Task completed in 0.082 seconds

Function NUM_ORDERS_SF compiled
```

FUNCTION

Test Second Function

```
/*Test Function 02*/
DECLARE
	test_customer_id NUMBER(8,0):=1110;
BEGIN
	DBMS_OUTPUT.PUT_LINE('The total number of orders is:' | | ' ‘
					| |num_orders_sf(test_customer_id));
END;
```



```
--Test
declare
test_customer_id NUMBER(8,0):=1110;
begin
DBMS_OUTPUT.PUT_LINE('The total number of orders is:'|| ' ' ||num_orders_sf(test_customer_id));
end;
```

Script Output × Query Result ×

Task completed in 0.088 seconds

```
PL/SQL procedure successfully completed.
```

Dbms Output

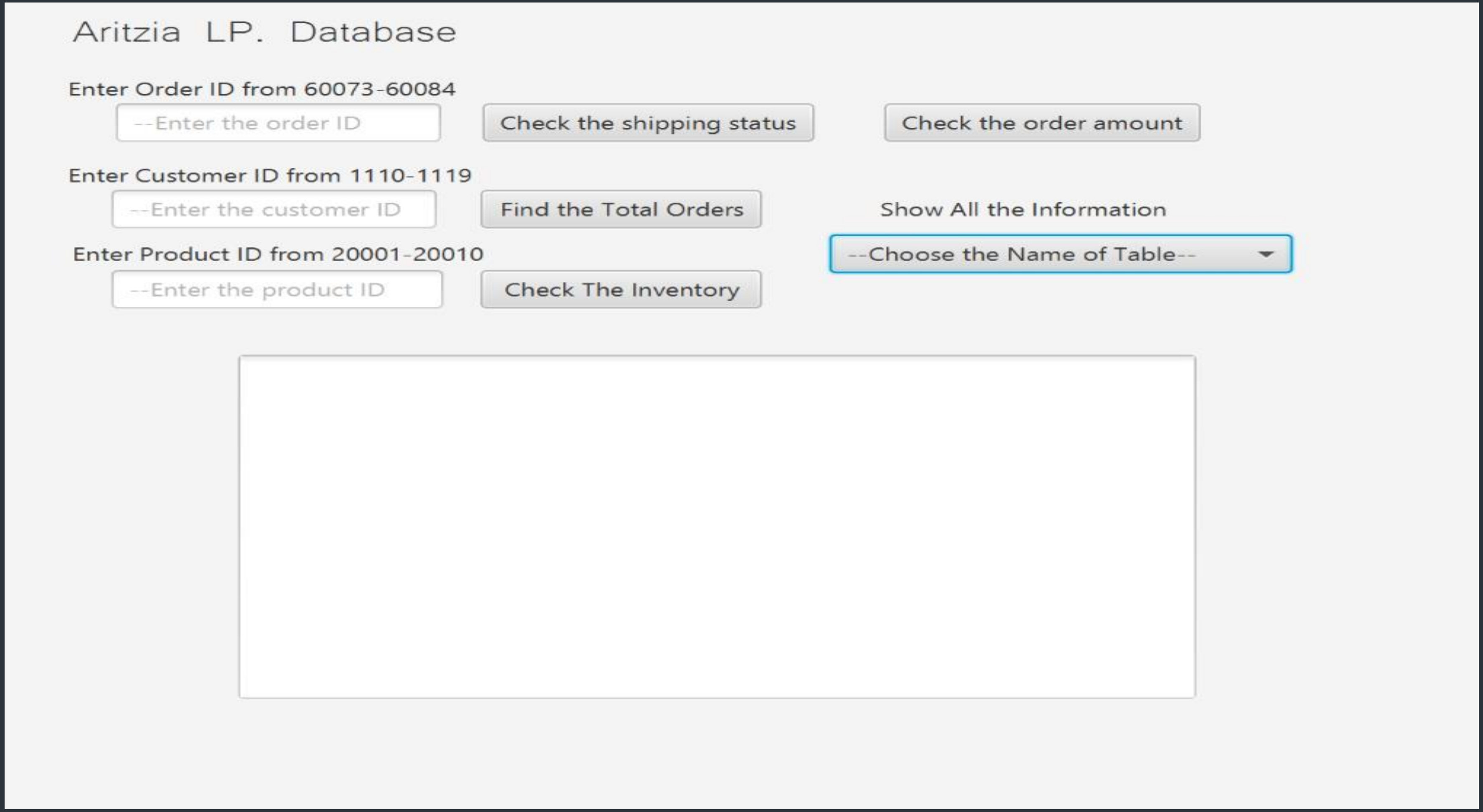
Buffer Size: 20000

college ×

```
The total number of orders is: 3
```

USER INTERFACE (UI) DESIGN

Overall UI Design


Aritzia LP. Database

Enter Order ID from 60073-60084

--Enter the order ID

Enter Customer ID from 1110-1119

--Enter the customer ID

Enter Product ID from 20001-20010

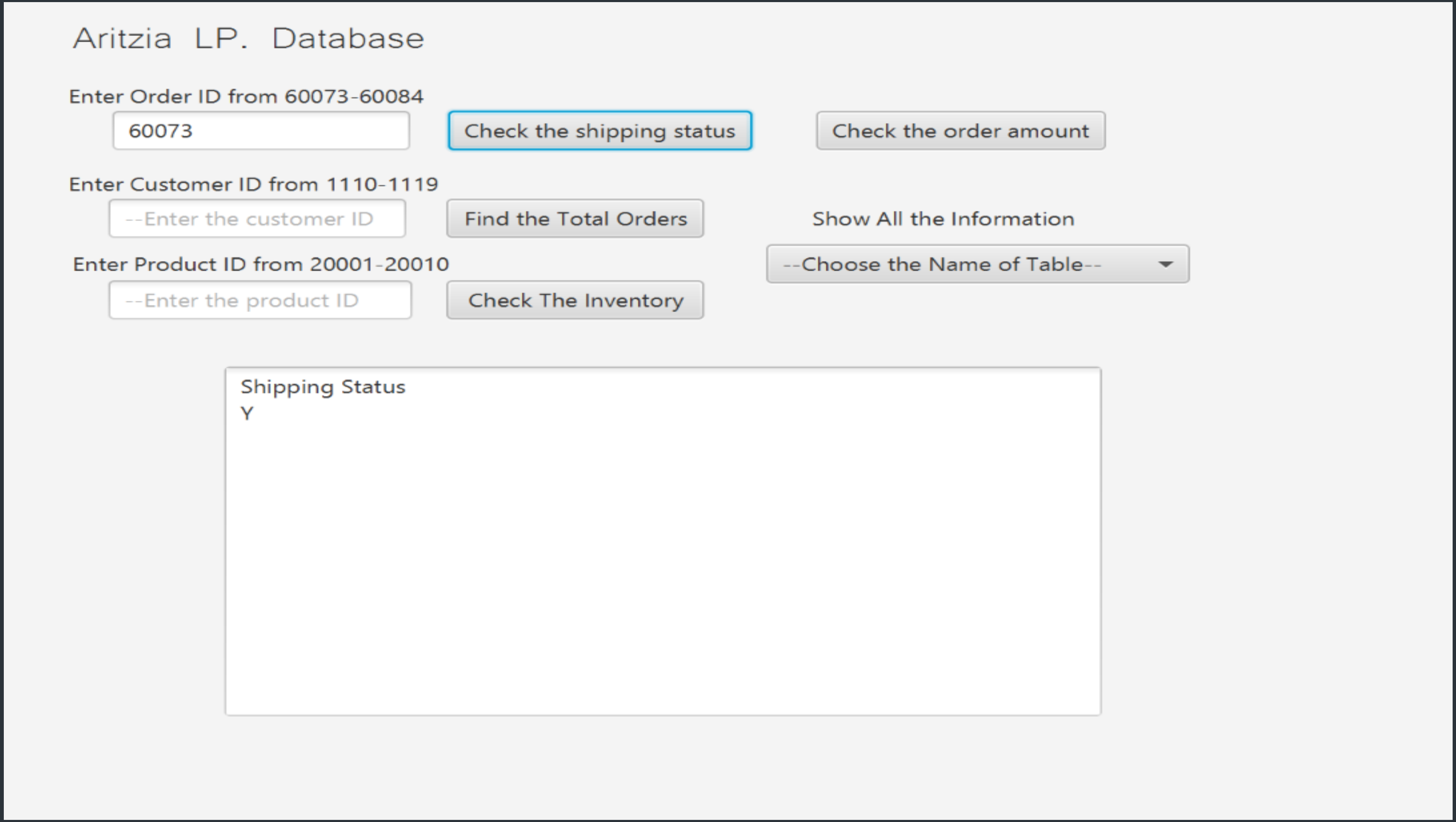
--Enter the product ID

Show All the Information

--Choose the Name of Table--

USER INTERFACE (UI) DESIGN

1.Check Shipping Status by Order ID


Aritzia LP. Database

Enter Order ID from 60073-60084

60073

Check the shipping status

Check the order amount

Enter Customer ID from 1110-1119

--Enter the customer ID

Find the Total Orders

Show All the Information

--Choose the Name of Table--

Enter Product ID from 20001-20010

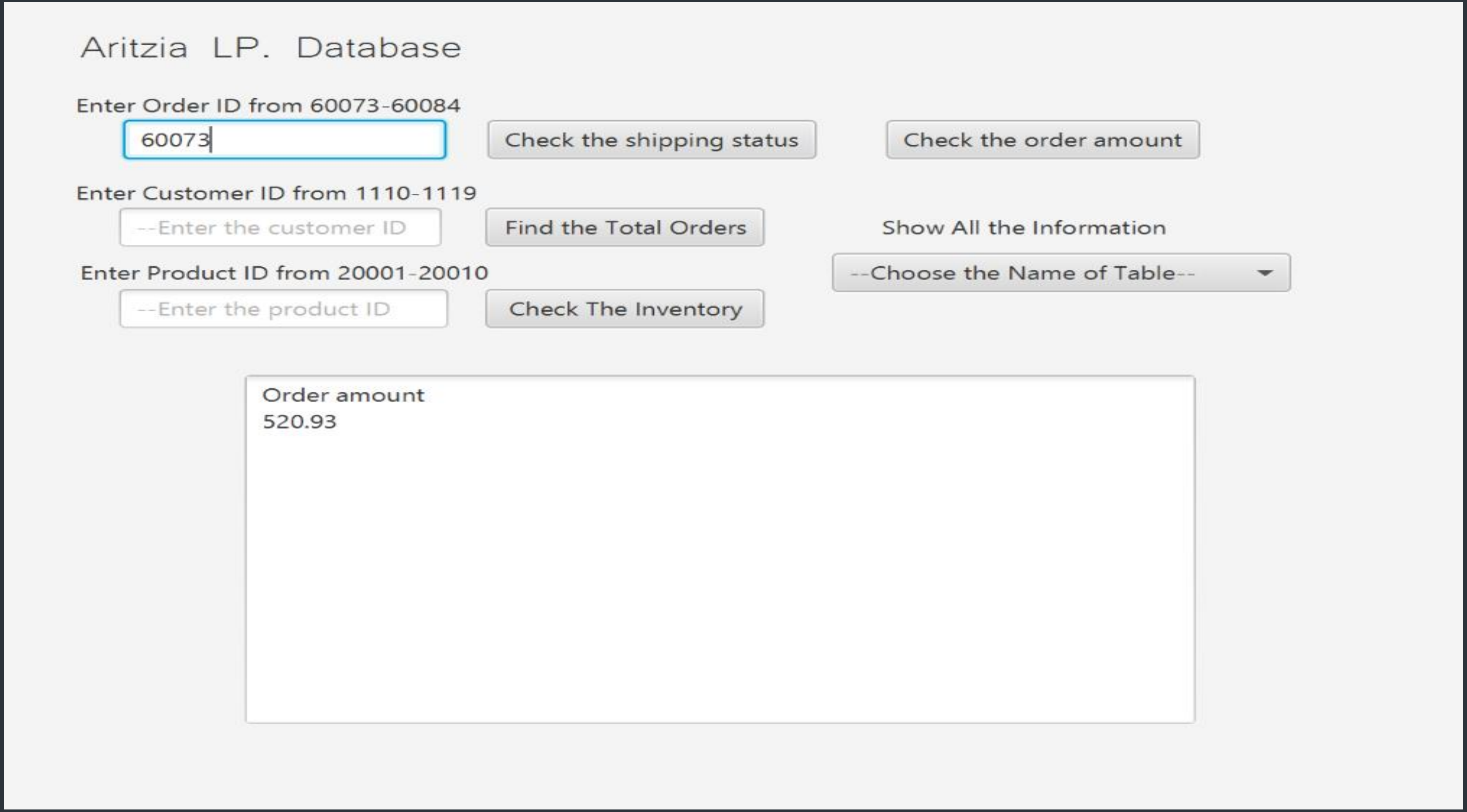
--Enter the product ID

Check The Inventory

```
Shipping Status
Y
```

USER INTERFACE (UI) DESIGN

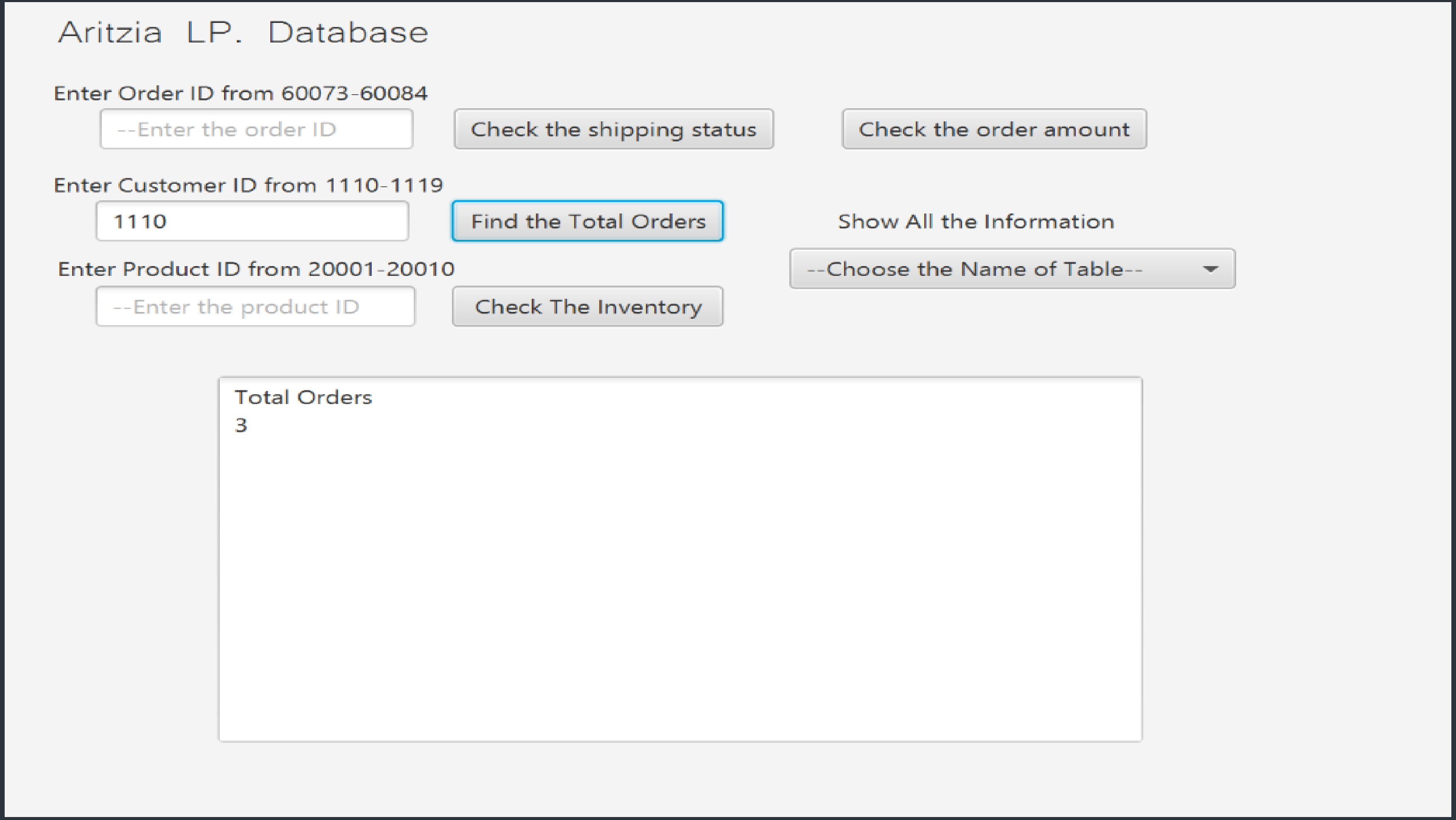
2. Calculate the Total Amount of an Order



```
Order amount
520.93
```

USER INTERFACE (UI) DESIGN

3. Calculate the User's the Total Number of Orders

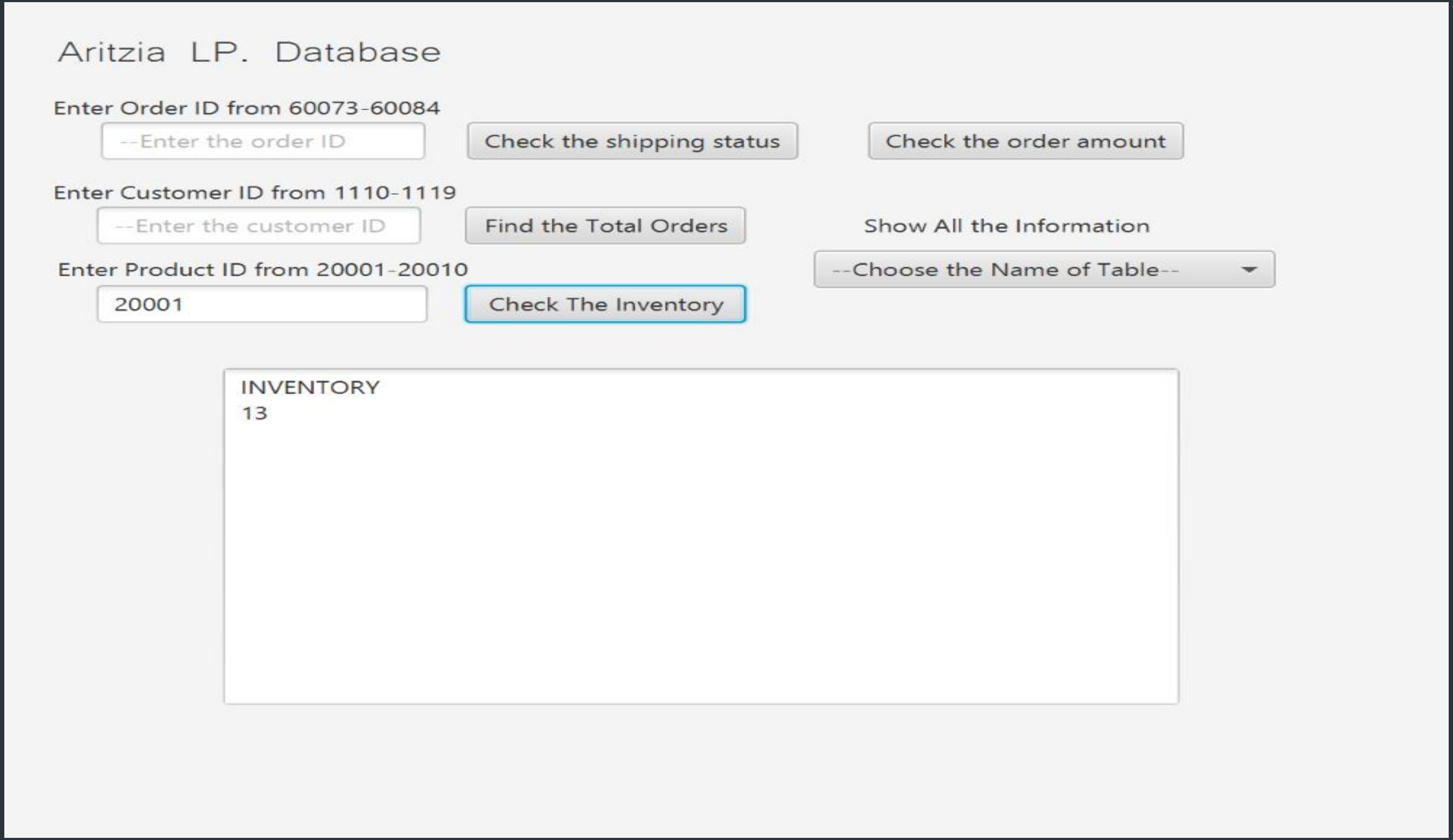

Aritzia LP. Database
Enter Order ID from 60073-60084
--Enter the order ID
Check the shipping status
Check the order amount
Enter Customer ID from 1110-1119
Find the Total Orders
Show All the Information
Enter Product ID from 20001-20010
--Choose the Name of Table--
--Enter the product ID
Check The Inventory

Customer ID: 1110

```
Total Orders
3
```

USER INTERFACE (UI) DESIGN

4. Check the Inventory by Product ID

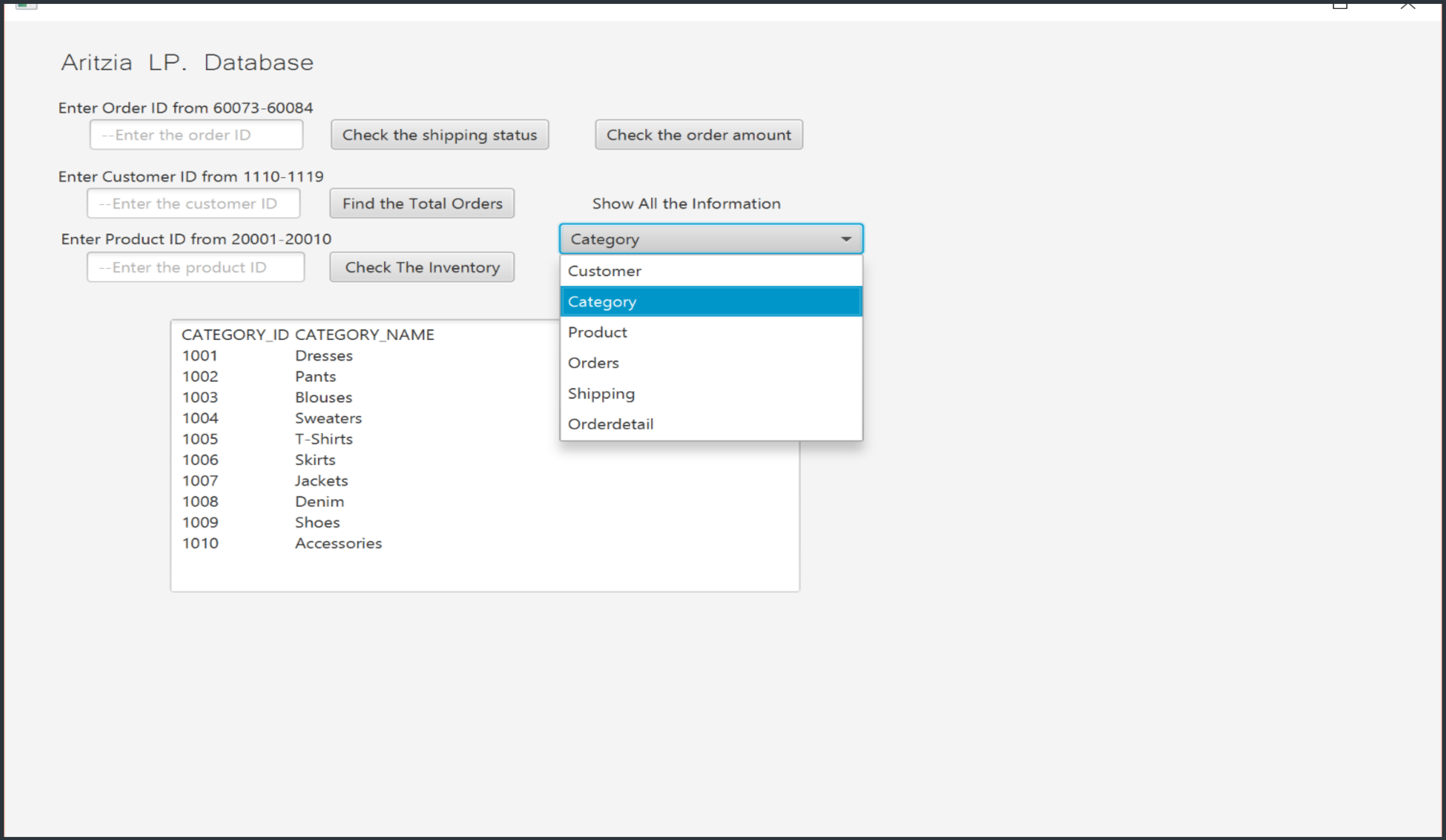

Aritzia LP. Database
Enter Order ID from 60073-60084
--Enter the order ID
Check the shipping status
Check the order amount
Enter Customer ID from 1110-1119
--Enter the customer ID
Find the Total Orders
Show All the Information
Enter Product ID from 20001-20010
--Choose the Name of Table--
Check The Inventory

Enter Product ID from 20001-20010: 20001

```
INVENTORY
13
```

USER INTERFACE (UI) DESIGN

5. Select the Table to Show all the Information



Aritzia LP. Database

Enter Order ID from 60073-60084

--Enter the order ID

Check the shipping status

Check the order amount

Enter Customer ID from 1110-1119

--Enter the customer ID

Find the Total Orders

Show All the Information

Enter Product ID from 20001-20010

--Enter the product ID

Check The Inventory

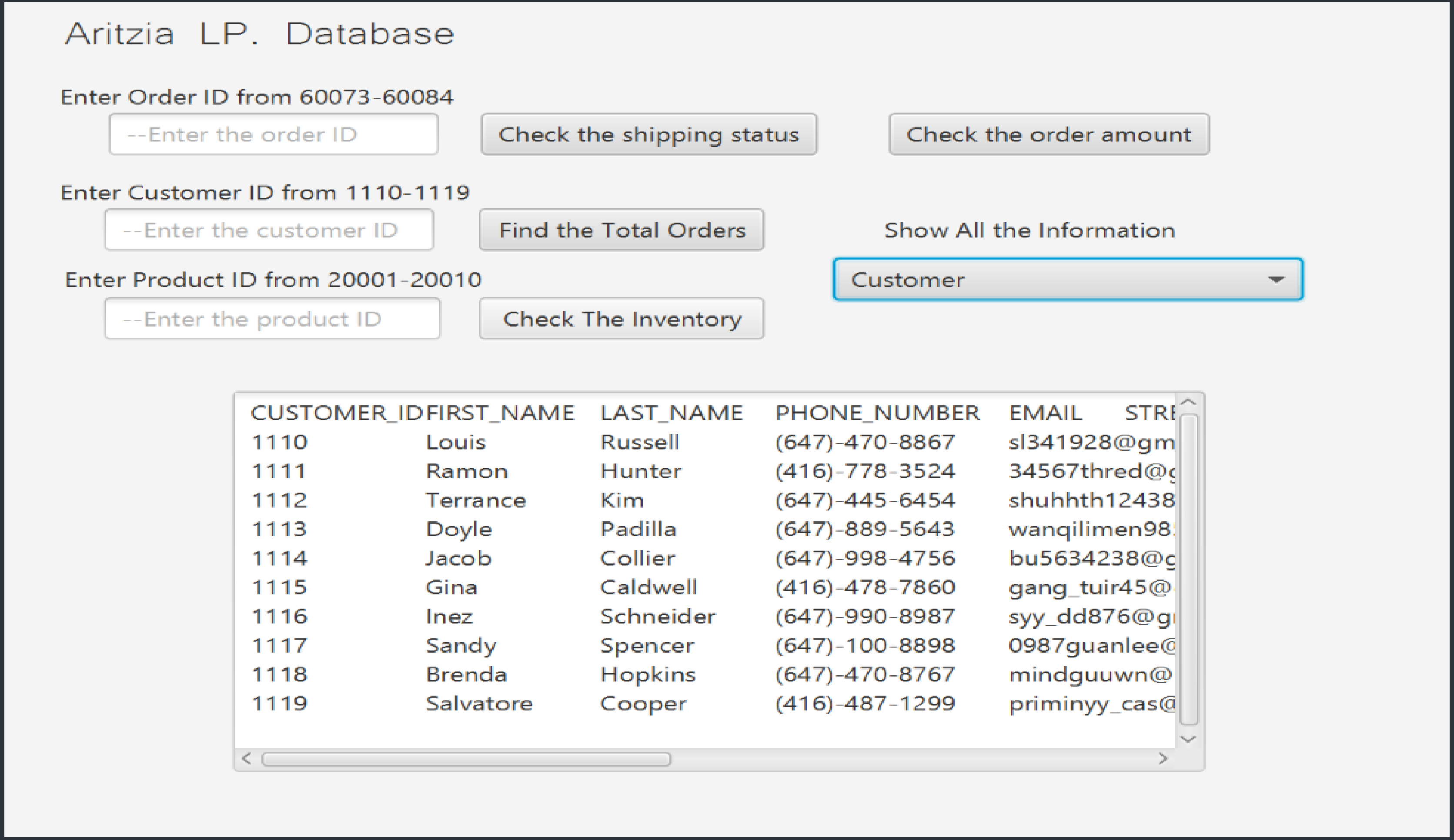
Category

- Customer
- Category
- Product
- Orders
- Shipping
- Orderdetail

```
CATEGORY_ID CATEGORY_NAME
1001        Dresses
1002        Pants
1003        Blouses
1004        Sweaters
1005        T-Shirts
1006        Skirts
1007        Jackets
1008        Denim
1009        Shoes
1010        Accessories
```

USER INTERFACE (UI) DESIGN

5. Select the Table to Show all the Information



Aritzia LP. Database

Enter Order ID from 60073-60084

Enter Customer ID from 1110-1119

Show All the Information

Customer

Enter Product ID from 20001-20010

CUSTOMER_ID	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	STRE
1110	Louis	Russell	(647)-470-8867	sl341928@gm	
1111	Ramon	Hunter	(416)-778-3524	34567thred@	
1112	Terrance	Kim	(647)-445-6454	shuhhth12438	
1113	Doyle	Padilla	(647)-889-5643	wanqilimen98	
1114	Jacob	Collier	(647)-998-4756	bu5634238@	
1115	Gina	Caldwell	(416)-478-7860	gang_tuir45@	
1116	Inez	Schneider	(647)-990-8987	syy_dd876@g	
1117	Sandy	Spencer	(647)-100-8898	0987guanlee@	
1118	Brenda	Hopkins	(647)-470-8767	mindguuwn@	
1119	Salvatore	Cooper	(416)-487-1299	priminyy_cas@	

Q/A