A. Structural Business Rules

New Product

- One listing has one to many products and one product is sold on one listing.
- One product is linked to one category and one category may have many products.

Product Delivery

- One seller can provide many products and one product is provided by one seller.

New Customer Account

- One customer has one to many accounts and one account belongs to one customer.

Product Purchase

- One customer may create many orders and one order is created by one customer.
- One product is linked to one order and one order can have one to many products.

Product Shipment

- One package is linked to one order and one order has one to many packages.
- One package has one to many products and one product is packed in one package.

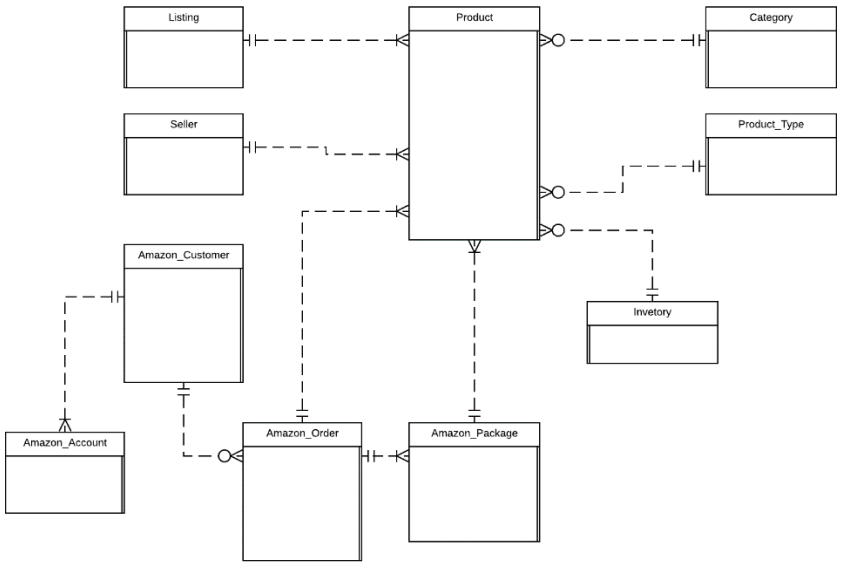
Product Type

- One product belongs to one type and one type may have many products.

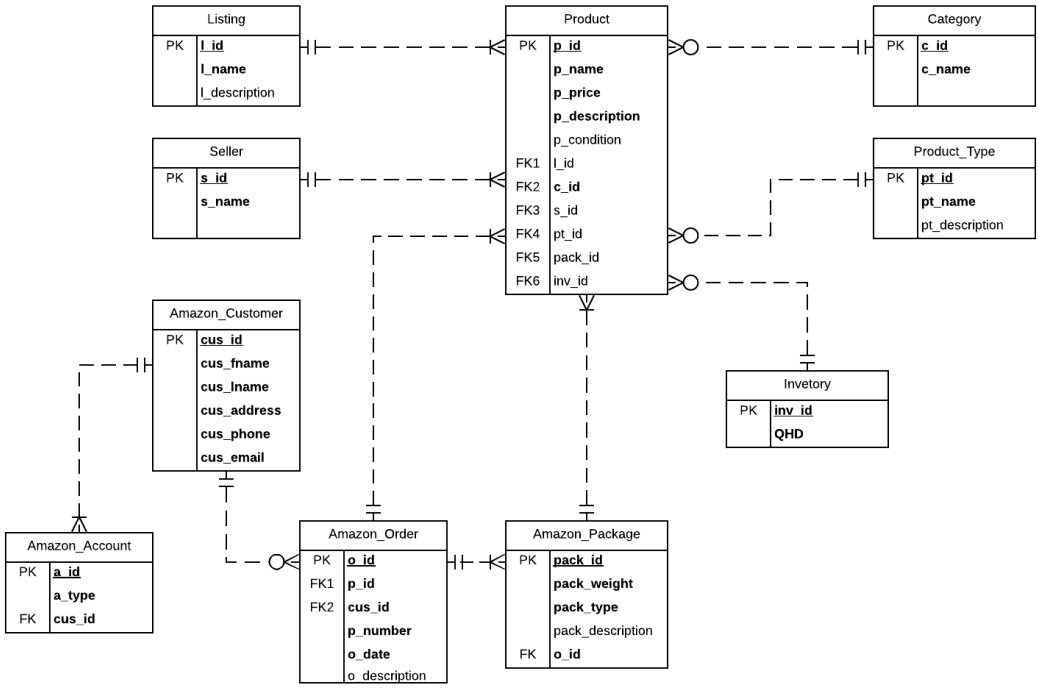
Product Inventory

- One product is stored in one inventory and one inventory may store many products

B. Conceptual ERD



C. Logical ERD



D. Aspects

Aspect1:

a. Reusable stored procedure.

Welcome Page | myz

0.50199997 seconds

Worksheet | Query Builder

```
CREATE OR REPLACE PROCEDURE ADD_PRODUCT( -- Add a new product
p_id_arg IN DECIMAL,
p_name_arg IN VARCHAR,
p_price_arg IN DECIMAL,
p_description_arg IN VARCHAR,
c_id_arg IN DECIMAL)
IS
BEGIN
INSERT INTO Product
(p_id, p_name, p_price, p_description, c_id)
VALUES(p_id_arg, p_name_arg, p_price_arg, p_description_arg, c_id_arg);
END;
```

Script Output

Task completed in 0.502 seconds

```
Procedure ADD_PRODUCT compiled
```

b. Use of the stored procedure.

Welcome Page | myz | 0.028 seconds

Worksheet | Query Builder

```
INSERT INTO Category VALUES(1,'Computers');
INSERT INTO Category VALUES(2,'Electronics');
```

Script Output | Task completed in 0.028 seconds

```
1 row inserted.


1 row inserted.
```

Welcome Page | myz | 0.03 seconds

Worksheet | Query Builder

```
BEGIN
    ADD_PRODUCT(1, 'Self-driving Video Camera', 28, 'Automatically follows a subject that is being recorded.', 2);
END;
/

BEGIN
    ADD_PRODUCT(2, 'Holographic Keyboard', 28, 'Emits a three-dimensional projection of a keyboard and recognizes virtual key presses from the typist.', 1);
END;
/
```

Script Output | Query Result | Task completed in 0.03 seconds

```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

c. SQL query.

Welcome Page | myz

0.028 seconds

Worksheet | Query Builder

```
SELECT Product.p_name, Product.p_price, Category.c_name
FROM Product
INNER JOIN Category ON Product.c_id = Category.c_id
WHERE Product.p_price <= 30
AND (Category.c_name = 'Computers' OR Category.c_name = 'Electronics');
```

Script Output | Query Result

Task completed in 0.028 seconds

```
P_NAME                           P_PRICE C_NAME
------------------------------ ---------- ------------------------------
Self-driving Video Camera              28 Electronics
Holographic Keyboard                   28 Computers
```

Change to the procedure:

Welcome Page | myz

0.212 seconds

Worksheet | Query Builder

```
CREATE OR REPLACE PROCEDURE ADD_PRODUCT( -- Add a new product
p_id_arg IN DECIMAL,
p_name_arg IN VARCHAR,
p_price_arg IN DECIMAL,
p_description_arg IN VARCHAR,
c_id_arg IN DECIMAL,
pt_id_arg IN DECIMAL,
pt_name_arg IN VARCHAR)
IS
BEGIN

DECLARE
    x NUMBER:=0;
BEGIN
    SELECT nvl((SELECT 1 FROM Product_Type WHERE pt_id = pt_id_arg) , 0) INTO x FROM dual;

    IF (X = 1) THEN

    INSERT INTO Product
    (p_id,p_name,p_price,p_description,c_id,pt_id)
    VALUES(p_id_arg,p_name_arg,p_price_arg,p_description_arg,c_id_arg,pt_id_arg);


    ELSE

    INSERT INTO Product_type(pt_id, pt_name) VALUES
    (pt_id_arg, pt_name_arg);
    INSERT INTO Product
    (p_id,p_name,p_price,p_description,c_id,pt_id)
    VALUES(p_id_arg,p_name_arg,p_price_arg,p_description_arg,c_id_arg,pt_id_arg);

    END IF;
END;
END;
```

Script Output

Task completed in 0.212 seconds

```
Procedure ADD_PRODUCT compiled
```

Welcome Page | myz

0.028 seconds

Worksheet | Query Builder

```
BEGIN
    ADD_PRODUCT(4, 'Self-driving Video Camera', 28, 'Automatically follows a subject that is being recorded.', 2, 1, 'Video Camera');
END;
/
```

Script Output

Task completed in 0.028 seconds

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

Welcome Page | myz

0.025 seconds

Worksheet | Query Builder

```
BEGIN
    ADD_PRODUCT(5, 'Self-driving Video Camera', 28, 'Automatically follows a subject that is being recorded.', 2, 1, 'Video Camera');
END;
/
```

Script Output

Task completed in 0.025 seconds

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

Welcome Page | myz

0.175 seconds

Worksheet | Query Builder

```
SELECT * FROM product_type;
```

Script Output

Task completed in 0.175 seconds

```
     PT_ID PT_NAME                        PT_DESCRIPTION
---------- ------------------------------ ------------------------------------
         1 Video Camera
```

Welcome Page | myz

0.025 seconds

Worksheet | Query Builder

```
BEGIN
    ADD_PRODUCT(3, 'Holographic Keyboard', 28, 'Emits a three-dimensional projection of a keyboard and recognizes virtual key presses from the typist.', 1, 2, 'Vitual Keyboard');
END;
/
```

Script Output

Task completed in 0.025 seconds

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

Welcome Page | myz

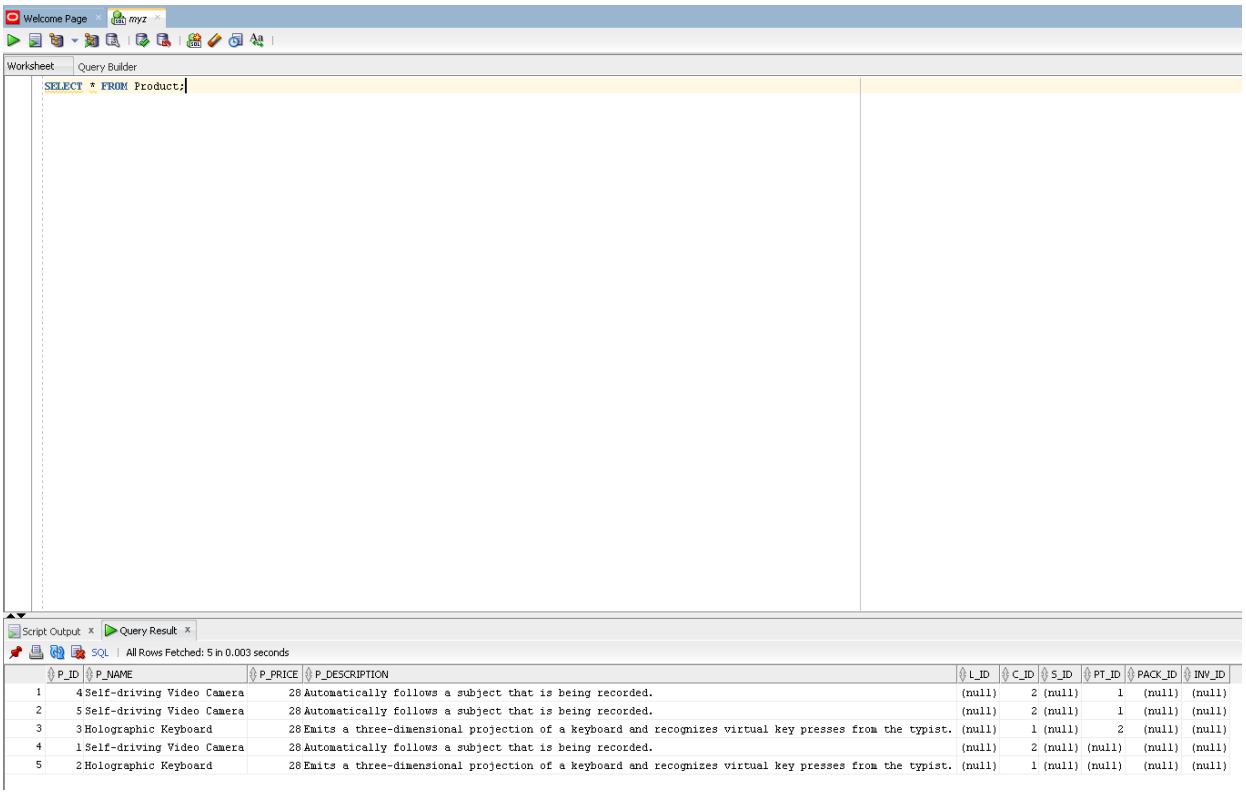
Worksheet | Query Builder

```
SELECT * FROM Product_type;
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.013 seconds

	PT_ID	PT_NAME	PT_DESCRIPTION
1	1	Video Camera	(null)
2	2	Vitual Keyboard	(null)



	P_ID	P_NAME	P_PRICE	P_DESCRIPTION	L_ID	C_ID	S_ID	PT_ID	PACK_ID	INV_ID
1	4	Self-driving Video Camera	28	Automatically follows a subject that is being recorded.	(null)	2	(null)	1	(null)	(null)
2	5	Self-driving Video Camera	28	Automatically follows a subject that is being recorded.	(null)	2	(null)	1	(null)	(null)
3	3	Holographic Keyboard	28	Emits a three-dimensional projection of a keyboard and recognizes virtual key presses from the typist.	(null)	1	(null)	2	(null)	(null)
4	1	Self-driving Video Camera	28	Automatically follows a subject that is being recorded.	(null)	2	(null)	(null)	(null)	(null)
5	2	Holographic Keyboard	28	Emits a three-dimensional projection of a keyboard and recognizes virtual key presses from the typist.	(null)	1	(null)	(null)	(null)	(null)

Aspect2:

a. Reusable stored procedure.

Welcome Page | myz

0.018 seconds

Worksheet | Query Builder

```
CREATE OR REPLACE PROCEDURE ADD_INVETORY( -- Add product from seller
p_id_arg IN DECIMAL,
inv_id_arg IN DECIMAL,
QHD_arg IN DECIMAL,
p_condition_arg IN VARCHAR)
IS
BEGIN


UPDATE Invetory
SET QHD = QHD_arg
WHERE inv_id = inv_id_arg;


UPDATE Product
SET p_condition = p_condition_arg,
    inv_id = inv_id_arg
WHERE p_id = p_id_arg;

END;
```

Script Output

Task completed in 0.018 seconds

```
Procedure ADD_INVETORY compiled
```

b. Use of the stored procedure.

Welcome Page | myz

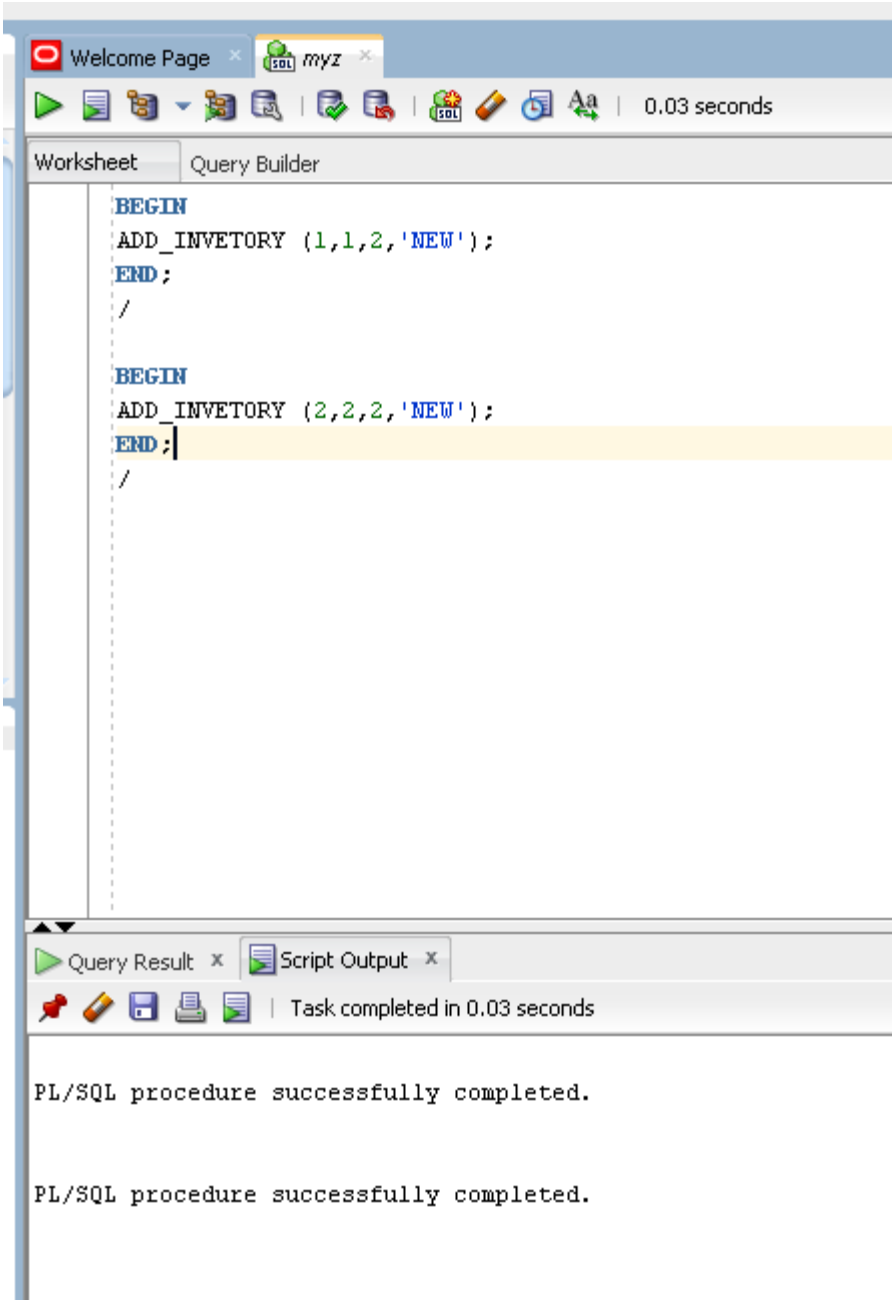
Worksheet | Query Builder

```
INSERT INTO Invetory(inv_id,QHD)
VALUES (1,0);

INSERT INTO Invetory(inv_id,QHD)
VALUES (2,0);
```

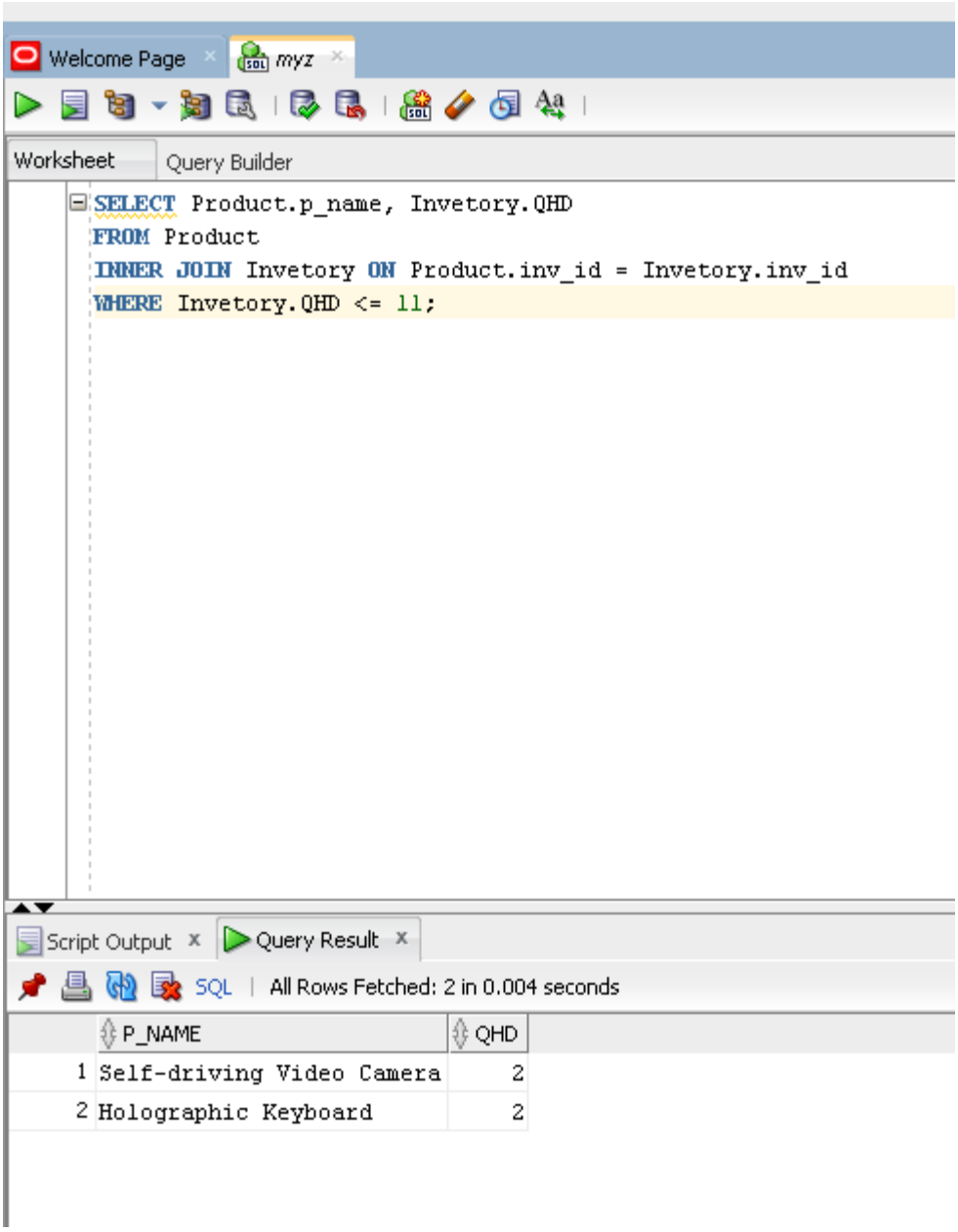
Query Result | Script Output

Task completed in 0.131 seconds

```
1 row inserted.


1 row inserted.
```


Welcome Page
myz
0.03 seconds
Worksheet
Query Builder
BEGIN
ADD_INVETORY (1,1,2,'NEW');
END;
/
BEGIN
ADD_INVETORY (2,2,2,'NEW');
END;
/
Query Result
Script Output
Task completed in 0.03 seconds
PL/SQL procedure successfully completed.
PL/SQL procedure successfully completed.

c. SQL query.



Aspect3:

a. Reusable stored procedure.

Welcome Page | myz | CUSTOMER

0.244 seconds

Worksheet | Query Builder

```
CREATE OR REPLACE PROCEDURE ADD_AMAZON_CUSTOMER( -- Add product from seller
cus_id_arg DECIMAL,
cus_fname_arg VARCHAR,
cus_lname_arg VARCHAR,
cus_address_arg VARCHAR,
cus_phone_arg DECIMAL,
cus_email_arg VARCHAR,
a_id_arg DECIMAL,
a_type_arg VARCHAR)
IS
BEGIN


INSERT INTO Amazon_Customer
VALUES(cus_id_arg,cus_fname_arg,cus_lname_arg,cus_address_arg,cus_phone_arg,cus_email_arg);


INSERT INTO Amazon_Account
VALUES(a_id_arg,a_type_arg,cus_id_arg);

END;
```

Script Output

Task completed in 0.244 seconds

```
Procedure ADD_AMAZON_CUSTOMER compiled
```

b. Use of the stored procedure.

Welcome Page | myz | CUSTOMER

0.18000001 seconds

Worksheet | Query Builder

```
BEGIN
ADD_AMAZON_CUSTOMER (1,'Yazhuo','Ma','106 Chester Street',6177854713,'ma.yazhuo.2013@gmail.com',1,'Normal');
END;
/

BEGIN
ADD_AMAZON_CUSTOMER (2,'George','Maiewski','500 Commonwealth Ave',1111111111,'maiewski@bu.edu',2,'Platinum');
END;
/
```

Script Output

Task completed in 0.18 seconds

```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

c. SQL query.

Welcome Pa... | ... | CUSTOM...

Worksheet | Query Builder

```
INSERT INTO Amazon_Account
VALUES (3,'Normal',1);
INSERT INTO Amazon_Account
VALUES (4,'Platinum',1);
INSERT INTO Amazon_Account
VALUES (5,'Normal',1);
```

Script Output | Query Result

Task completed in 0.074 seconds

```
1 row inserted.

1 row inserted.

1 row inserted.
```

Welcome Pa... | ... | CUSTOM...

Worksheet | Query Builder

```
1 SELECT Amazon_Customer.cus_lname,COUNT(*)
2 FROM Amazon_Customer
3 INNER JOIN Amazon_Account ON Amazon_Customer.cus_id = Amazon_Account.cus_id
4 GROUP BY Amazon_Customer.cus_lname
5 HAVING COUNT(*)>=4;
```

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.004 seconds

	CUS_LNAME	COUNT(*)
1	Ma	4

Aspect4:

a. Reusable stored procedure.

Welcome Pa... | ... | AMAZON_ORD...

0.13699999 seconds

Worksheet | Query Builder

```
CREATE OR REPLACE PROCEDURE ADD_Amazon_Order(
o_id_arg IN DECIMAL,
p_id_arg IN DECIMAL,
cus_id_arg IN DECIMAL,
p_number_arg IN DECIMAL,
o_date_arg IN DATE,
o_description_arg IN VARCHAR)
IS
BEGIN


INSERT INTO Amazon_Order(o_id,p_id,cus_id,p_number,o_date,o_description)
VALUES(o_id_arg,p_id_arg,cus_id_arg,p_number_arg,o_date_arg,o_description_arg);

END;
```

Script Output

Task completed in 0.137 seconds

```
Procedure ADD_AMAZON_ORDER compiled
```

b. Use of the stored procedure

Welcome Pa... | ... | AMAZON_CUSTOM...

0.13500001 seconds

Worksheet | Query Builder

```
BEGIN
ADD_Amazon_Order (1,1,1,1,CAST('04-DEC-2018' AS DATE),'None');
END;
/

BEGIN
ADD_Amazon_Order (2,2,2,3,CAST('04-DEC-2018' AS DATE),'None');
END;
/
```

Script Output

Task completed in 0.135 seconds

```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

Welcome Page | myz | AMAZON_ACCOUNT

0.034 seconds

Worksheet | Query Builder

```
BEGIN
ADD_AMAZON_CUSTOMER (3,'Chenyang','Wang','107 Chester Street',6177854713,'chengyang.wang@gmail.com',6,'Golden');
END;
/

BEGIN
ADD_AMAZON_CUSTOMER (4,'James','Yu','108 Chester Street',6666666666,'james.yu@gmail.com',7,'Golden');
END;
/
```

Script Output

Task completed in 0.034 seconds

```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

Welcome Page | myz | AMAZON_ACCOUNT

0.199 seconds

Worksheet | Query Builder

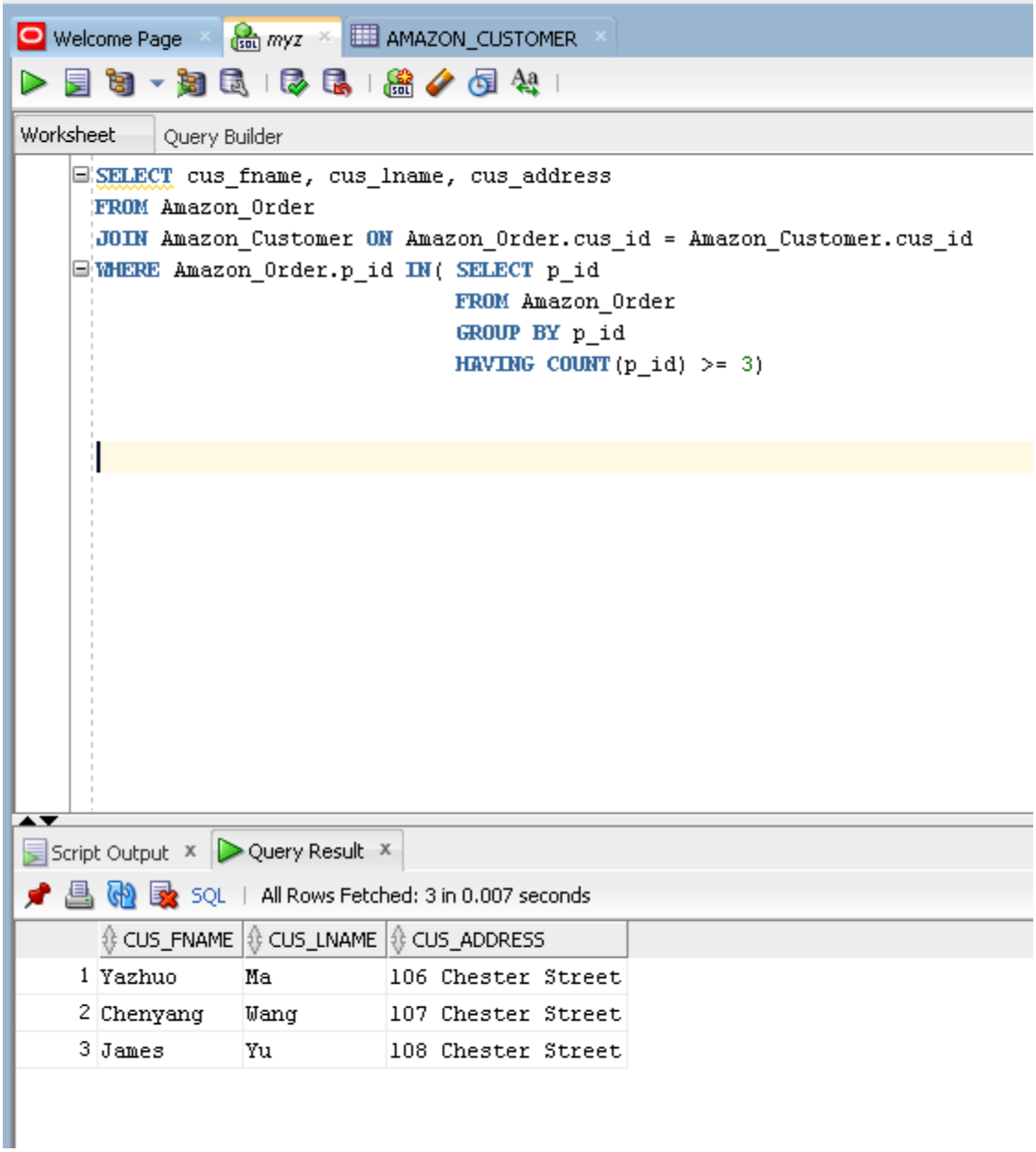
```
BEGIN
ADD_Amazon_Order (3,1,3,2,CAST('05-DEC-2018' AS DATE),'None');
END;
/

BEGIN
ADD_Amazon_Order (4,1,4,1,CAST('05-DEC-2018' AS DATE),'None');
END;
/
```

Script Output

Task completed in 0.199 seconds

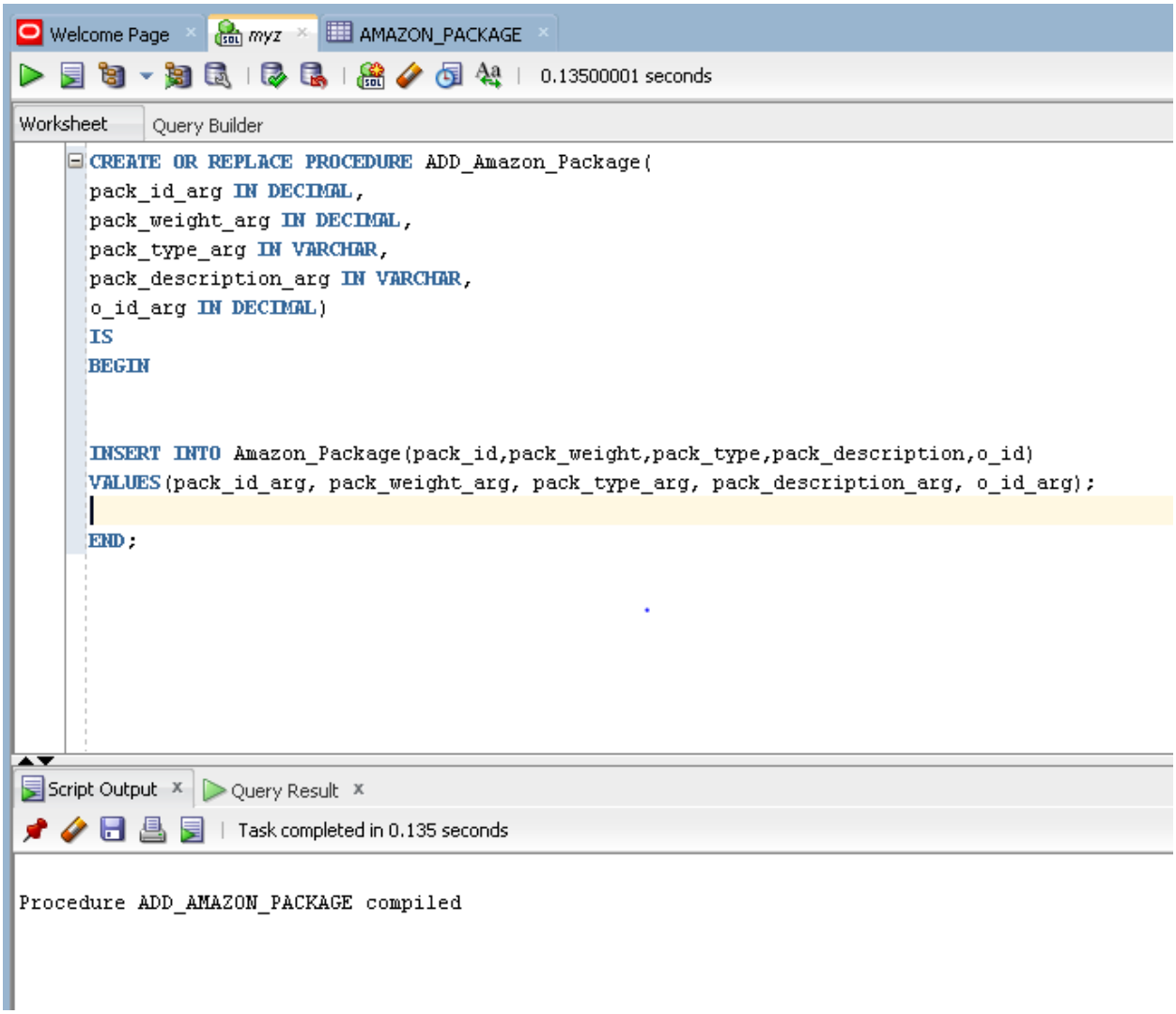
```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

c. SQL query.



Aspect5:

a. Creation of the stored procedure



.

b. Use of the stored procedure

Welcome Page | myz | AMAZON_PACKAGE

0.054 seconds

Worksheet | Query Builder

```
BEGIN
ADD_Amazon_Package (1,10,'Two Day','Tracking Number: 10010',1);
END;
/


BEGIN
ADD_Amazon_Package (2,20,'One Day','Tracking Number: 20020',2);
END;
/
```

Script Output | Query Result

Task completed in 0.054 seconds

```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.
```

c. SQL Query

Welcome Page | myz | AMAZON_PACKAGE

0.024 seconds

Worksheet | Query Builder

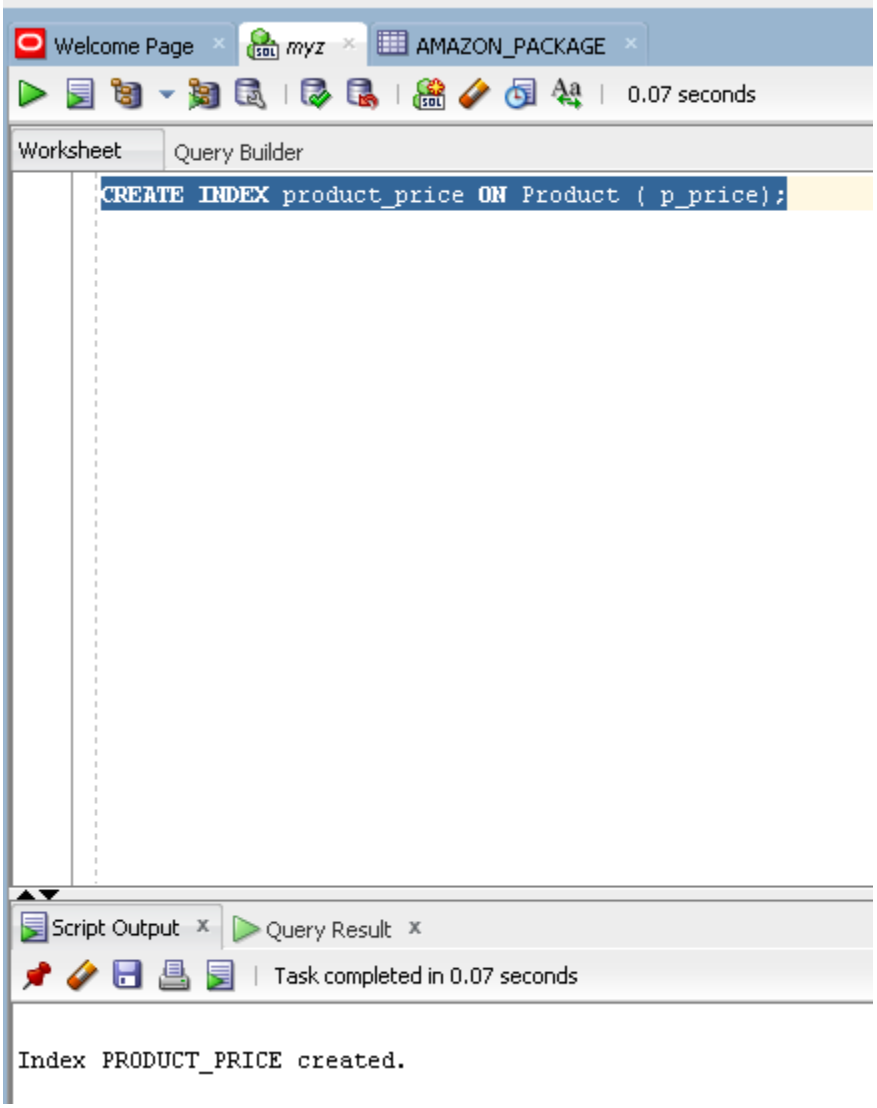
```
SELECT pack_id
FROM(SELECT pack_id,pack_weight
     FROM Amazon_Package
     WHERE pack_type = 'Two Day')
WHERE pack_weight <= 10;
```

Script Output | Query Result

Task completed in 0.024 seconds

```
   PACK_ID
----------
         1
```

INDEX:



This Index can speed up the query speed of aspect 1, which will reduce the spending time to find out p_price less than or equal to 30.