Relational Database for an Online Jersey eStore

CIS 5430 - Databases and Data Warehousing

Dr. Wang

Crystal Wu (Team Lead & Database Developer)

Aishwarya Gupta (Database Developer)

Rahul Bhogale (Database Developer)

Table of Contents

Project 1

Scenario

The e-Store database needs to keep track of orders for its jerseys. The purpose is to store information about customers, jersey orders, and employees. The database is designed to hold a large collection of customer, employee, order, and jersey data to support multiple users who will simultaneously access and query the data. The primary users will be employees and customers.

When a customer places an order, the system must record that order and the corresponding jersey items. The system must update the available quantity on hand to reflect that the jersey(s) has been sold. When an employee processes orders, the system must confirm that the ordered items are in stock.

The online store needs to keep track of customers and employees too. Each customer and employee have a unique ID. We keep track of each employee's name, start date, and salary as well as each customer's name and address.

The store sells two types of jerseys: customized and traditional. Customized jerseys have the name specified and must be manufactured to the customer's specifications. We keep track of the status of manufactured jerseys. Purchased jerseys do not have a customized name from the customer; instead, they are the traditional player jersey purchased off the shelf. We keep track of whether purchased player jerseys are protected by assurance. We keep track of all jerseys' available size, price, color style, available quantity, material, number, name, and team name.

Business Rules

One customer may or may not place many orders.
One order must be placed by one and only one customer.

One order must contain one or more product.
One product may or may not be in many orders.

One employee may process one or more orders.
One order must be processed by one and only one employee.

One product must be either manufactured or purchased.

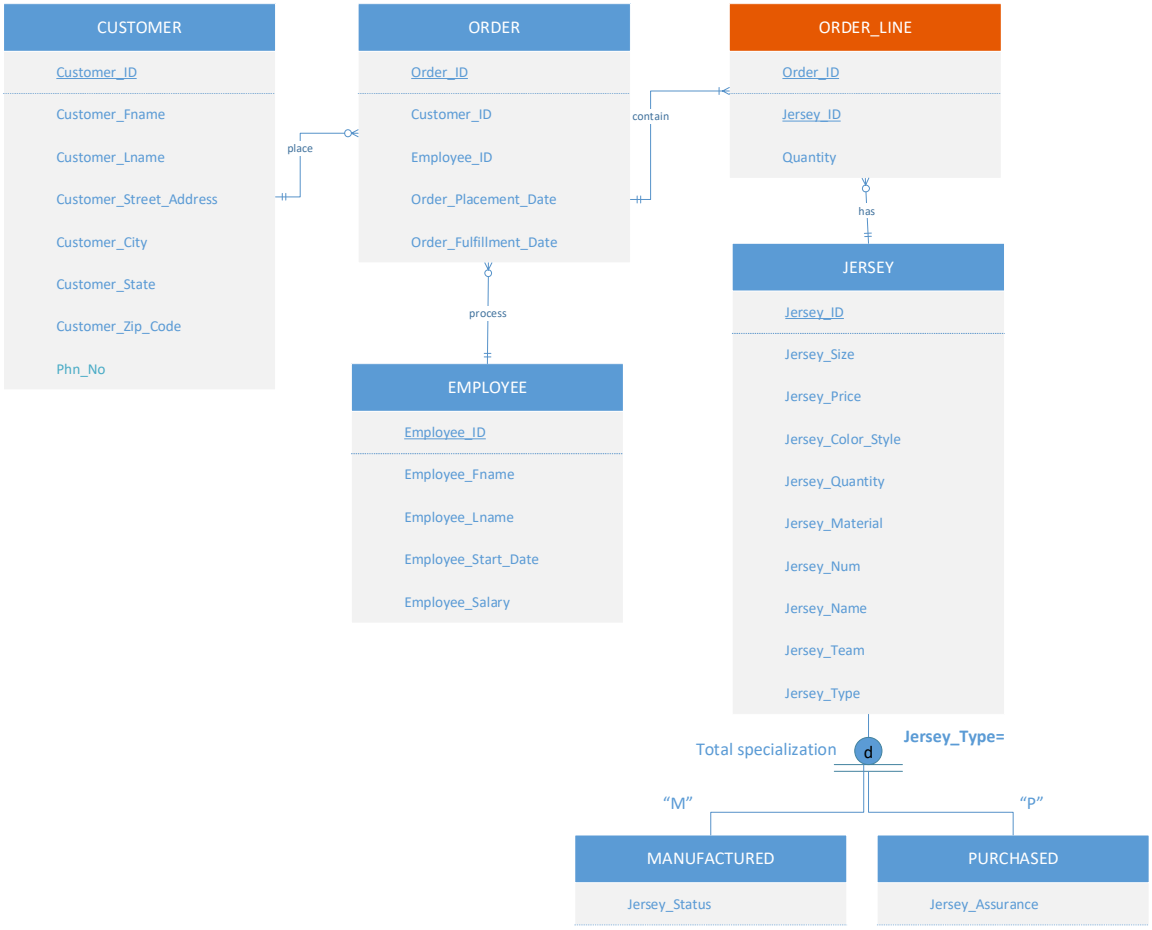
Relationship Matrix

1. Identify entity types and relationship types. Fill out the following relationship matrix.

	Customer	Order	Product	Employee
Customer		Placed by		
Order	Place		Contained in	Process
Product		Contain		
Employee		Processed by		

ER Diagram

2.Draw an ER/EER diagram using software tools includes 1) entity types, 2) relationship types, 3) keys, 4) attributes, and cardinality constraints (must show participation).



Tables & Attributes

3. **Database Logical Design:** Map the ER diagram to a relational database schema indicating **the relation name**, **primary key and foreign key.** Add appropriate additional attributes by yourself.

CUSTOMER

Customer_ID (PK)	Customer_Fname	Customer_Lname	Customer_Street_ Address	Customer_City
Customer_State	Customer_Zip_Code	Phn_no		

ORDER

Order_ID (PK)	Cust_ID (FK)	Emp_ID (FK)	Order_Placement_Date	Order_Fulfillment_Date

ORDER_LINE

Order_ID (FK)	Jersey_ID (FK)	Quantity		

EMPLOYEE

Employee_ID (PK)	Employee_Fname	Employee_Lname	Employee_Start_Date	Employee_Salary

JERSEY

Jersey_ID (PK)	Jersey_Size	Jersey_Price	Jersey_Color_Style	Jersey_Quantity
Jersey_Material	Jersey_Num	Jersey_Name	Jersey_Tname	Jersey_Type

MANUFACTURED

Jersey_ID (FK)	Jersey_Status			

PURCHASED

Jersey_ID (FK)	Jersey_Assurance			

Referential Integrity

4. Establish join paths for the above relational database using the referential integrity by drawing arrow lines between the above tables. Indicate all the foreign keys (FK).

F.K. -> P.K. (Foreign Key refers to Primary Key)

F.K. MANUFACTURED.Jersey_ID → P.K. JERSEY.Jersey_ID

F.K. PURCHASED.Jersey_ID → P.K. JERSEY.Jersey_ID

F.K. ORDER.Cust_ID → P.K. CUSTOMER.Customer_ID

F.K. ORDER.Emp_ID → P.K. EMPLOYEE.Employee_ID

F.K. ORDER_LINE.Ord_ID → P.K. ORDER.Order_ID

F.K. ORDER_LINE.Jersey_ID → P.K. JERSEY.Jersey_ID

CUSTOMER

Customer_ID (PK)	Customer_Fname	Customer_Lname	Customer_Street_Address	Customer_City
Customer_State	Customer_Zip_Code	Phn_no		

ORDER

Order_ID (PK)	Cust_ID (FK)	Emp_ID (FK)	Order_Placement_Date	Order_Fulfillment_Date

ORDER_LINE

Order_ID (FK)	Jersey_ID (FK)	Quantity		

Order_ID, Jersey_ID: composite PK

JERSEY

Jersey_ID (PK)	Jersey_Size	Jersey_Price	Jersey_Color_Style	Jersey_Quantity
Jersey_Material	Jersey_Num	Jersey_Name	Jersey_Tname	Jersey_Type

EMPLOYEE

Employee_ID (PK)	Employee_Fname	Employee_Lname	Employee_Start_Date	Employee_Salary

MANUFACTURED

Jersey_ID (FK)	Jersey_Status			

PURCHASED

Jersey_ID (FK)	Jersey_Assurance			

Functional Dependency Analysis

5. Do function analysis for each of your tables
   Attribute A -> Attribute B (Determinant attribute(s) Determines Dependent Attribute(s))

CUSTOMER
Customer_ID → Customer_Fname, Customer_Lname, Customer_Street_Address, Phn_no (Full dependency)
Customer_Zip_Code → Customer_City, Customer_State (Transitive dependency)

ORDER
Order_ID → Cust_ID, Emp_ID, Order_Placement_Date, Order_Fulfillment_Date (Full dependency)

ORDER_LINE
Ord_ID, Prod_ID → Quantity (Full dependency)

EMPLOYEE
Employee_ID → Employee_Fname, Employee_Lname, Employee_Start_Date, Employee_Salary (Full dependency)

JERSEY
Jersey_ID → Jersey_Size, Jersey_Price, Jersey_Color_Style, Jersey_Quantity, Jersey_Material, Jersey_Num, Jersey_Name, Jersey_Tname, Jersey_Type (Full dependency)

MANUFACTURED
Jersey_ID → Jersey_Status (Full dependency)

PURCHASED
Jersey_ID → Jersey_Assurance (Full dependency)

Normalization

6. Show all the normalized tables and indicate the normalization form for each of your tables.

Table Name	1NF	2NF	3NF
CUSTOMER	✓	✓	
ORDER	✓	✓	✓
ORDER_LINE	✓	✓	✓
EMPLOYEE	✓	✓	✓
PRODUCT	✓	✓	✓
MANUFACTURED	✓	✓	✓
PURCHASED	✓	✓	✓

The CUSTOMER table is in 2NF in the normalization table because the customer zip code determines the corresponding city and state.

Project II

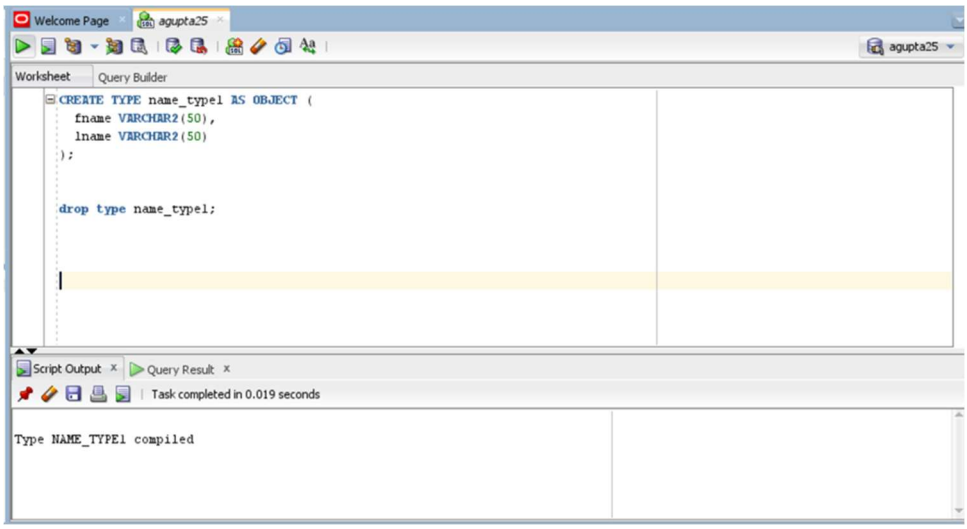
DDL Script

Aishwarya

```
drop table customer1 cascade constraints;
```

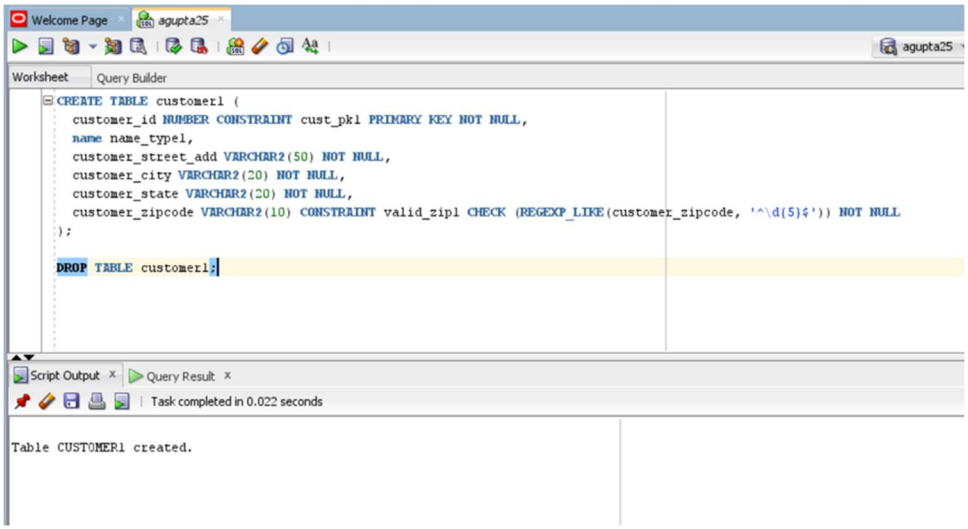
Create an object type as name for first and last name of customers

```
CREATE TYPE name_type1 AS OBJECT (
  fname VARCHAR2(50),
  lname VARCHAR2(50)
);
```



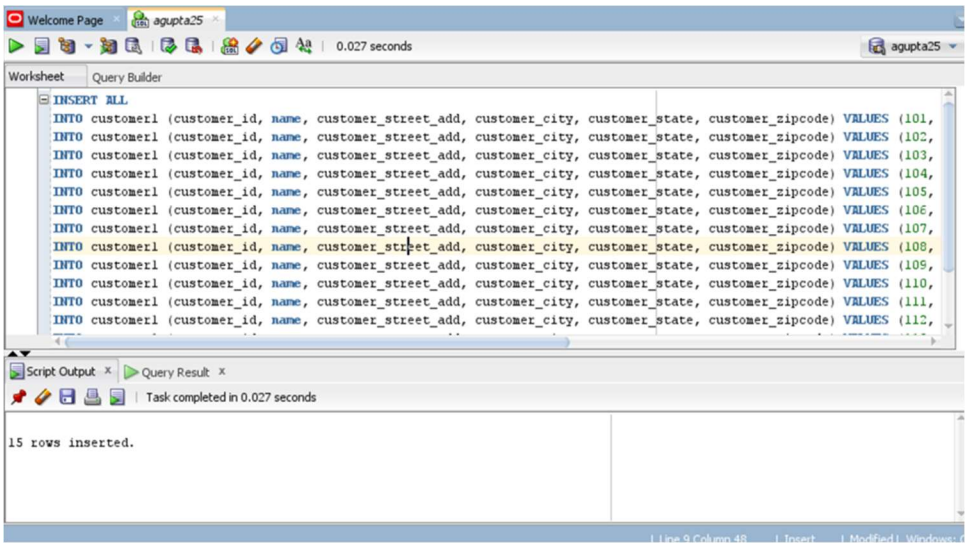
Then we use the above name_type object in customer table:

```
CREATE TABLE customer1 (
  customer_id NUMBER CONSTRAINT cust_pk1 PRIMARY KEY NOT NULL,
  name name_type1,
  customer_street_add VARCHAR2(50) NOT NULL,
  customer_city VARCHAR2(20) NOT NULL,
  customer_state VARCHAR2(20) NOT NULL,
  customer_zipcode VARCHAR2(10) CONSTRAINT valid_zip1 CHECK
(REGEXP_LIKE(customer_zipcode, '^\d{5}$')) NOT NULL
);
```



INSERT ALL
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (101, name_type1('Ash', 'Gupta'), 'Atlantic and Garvey', 'Alhambra', 'California', 91803)
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (102, name_type1('John', 'Doe'), '123 Main St', 'Anytown', 'CA', '12345')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (103, name_type1('Jane', 'Smith'), '456 Oak Ave', 'Otherville', 'TX', '67890')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (104, name_type1('Bob', 'Johnson'), '789 Maple St', 'Sometown', 'NY', '23456')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (105, name_type1('Samantha', 'Williams'), '234 Elm St', 'Anycity', 'FL', '34567')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (106, name_type1('Mike', 'Brown'), '567 Pine Ave', 'Othercity', 'IL', '45678')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (107, name_type1('Sarah', 'Taylor'), '890 Cedar Blvd', 'Somewhere', 'NY', '56789')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (108, name_type1('David', 'Lee'), '111 Oak St', 'Anyville', 'GA', '67890')
INTO customer1 (customer_id, name, customer_street_add, customer_city, customer_state, customer_zipcode) VALUES (109, name_type1('Karen', 'Davis'), '222 Maple Ave', 'Sometown', 'CA', '78901')

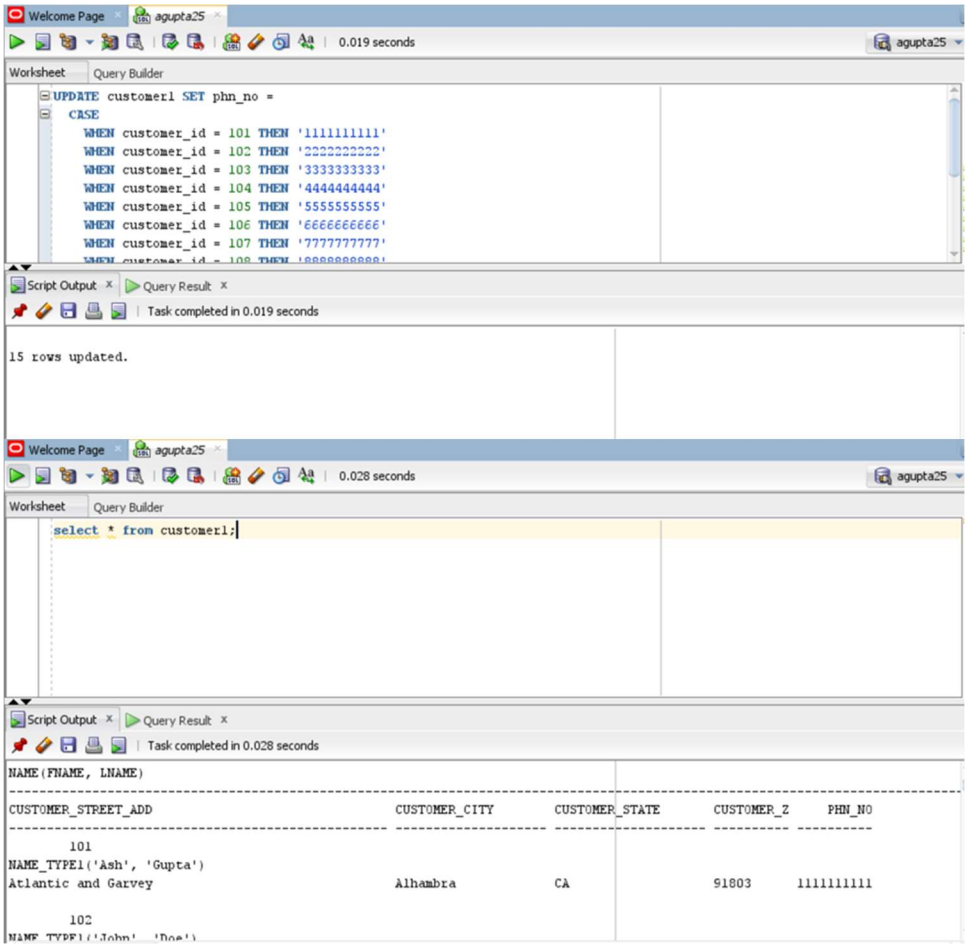
```
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (110, name_type1('Tom', 'Smith'), '333
Elm St', 'Othertown', 'NC', '89012')
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (111, name_type1('Ashley', 'Martin'), '444
Pine St', 'Somewhere', 'GA', '90123')
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (112, name_type1('Chris', 'Wilson'), '555
Cedar Ave', 'Anycity', 'OK', '23456')
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (113, name_type1('Lisa', 'Lee'), '666 Oak
Blvd', 'Othercity', 'NY', '34567')
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (114, name_type1('Mike','Davis'),'777
Maple St', 'Anytown', 'TX', '45678')
INTO customer1 (customer_id, name, customer_street_add, customer_city,
customer_state, customer_zipcode) VALUES (115, name_type1('Kim','Lee'), '55 Cesar
Ave', 'Othercity','WA','98765')
SELECT 1 FROM DUAL;
```



Updated the above table with unique phn no as an identifier mentioned in update delete statements below:

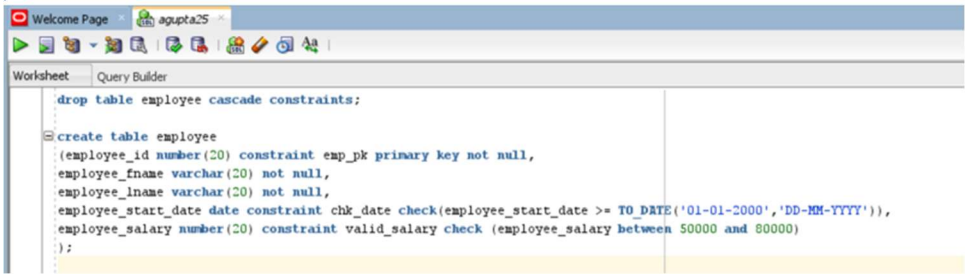
```
alter table customer1 add phn_no number(10) unique;
alter table customer1 drop column phn_no;


UPDATE customer1 SET phn_no =
  CASE
    WHEN customer_id = 101 THEN '1111111111'
    WHEN customer_id = 102 THEN '2222222222'
```

```
    WHEN customer_id = 103 THEN '3333333333'
    WHEN customer_id = 104 THEN '4444444444'
    WHEN customer_id = 105 THEN '5555555555'
    WHEN customer_id = 106 THEN '6666666666'
    WHEN customer_id = 107 THEN '7777777777'
    WHEN customer_id = 108 THEN '8888888888'
    WHEN customer_id = 109 THEN '9999999999'
    WHEN customer_id = 110 THEN '1112223333'
    WHEN customer_id = 111 THEN '2223334444'
    WHEN customer_id = 112 THEN '3334445555'
    WHEN customer_id = 113 THEN '4445556666'
    WHEN customer_id = 114 THEN '5556667777'
    WHEN customer_id = 115 THEN '6667778888'
  END;
```



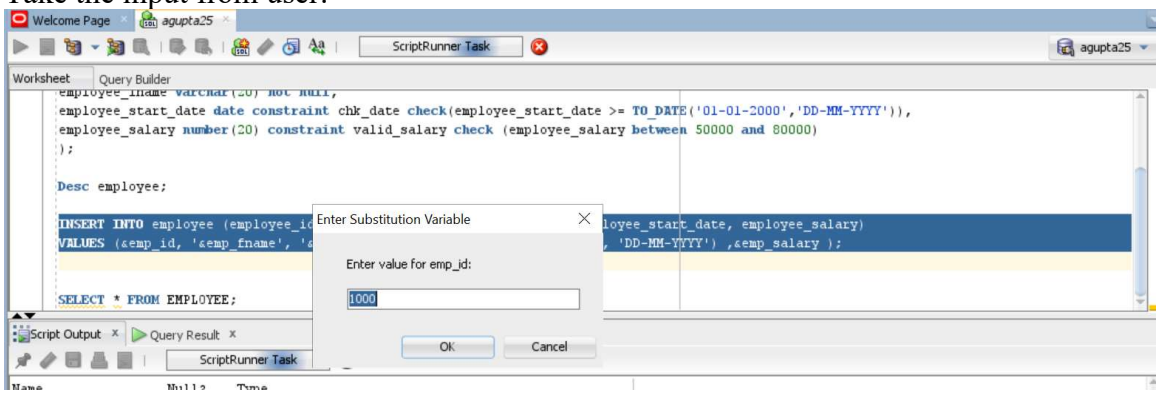

```
drop table employee cascade constraints;
create table employee
(employee_id number(20) constraint emp_pk primary key not null,
employee_fname varchar(20) not null,
employee_lname varchar(20) not null,
```

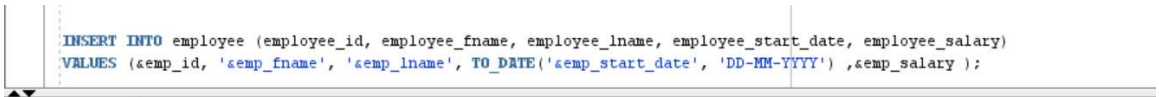
employee_start_date date constraint chk_date check(employee_start_date >= TO_DATE('01-01-2000','DD-MM-YYYY')),
employee_salary number(20) constraint valid_salary check (employee_salary between 50000 and 80000)
);

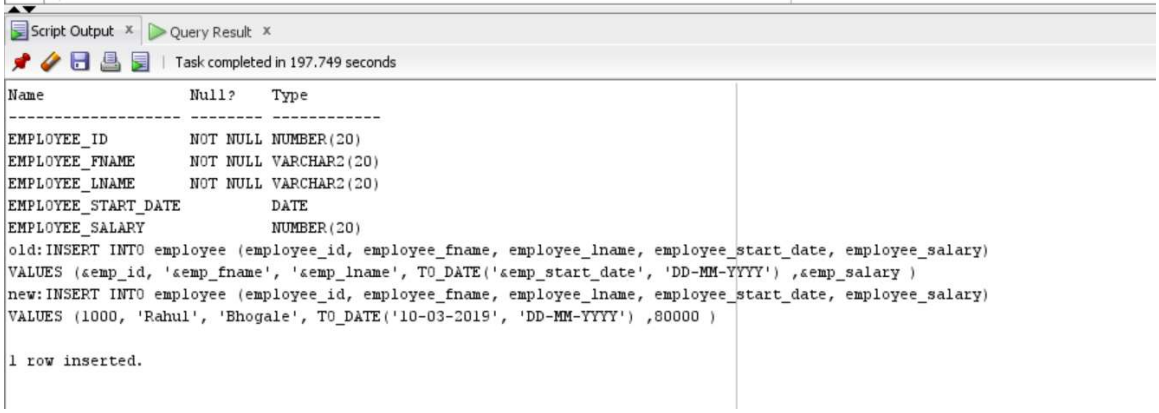


INSERT INTO employee (employee_id, employee_fname, employee_lname, employee_start_date, employee_salary)
VALUES (&emp_id, '&emp_fname', '&emp_lname', TO_DATE('&emp_start_date', 'DD-MM-YYYY') ,&emp_salary);

Take the input from user:



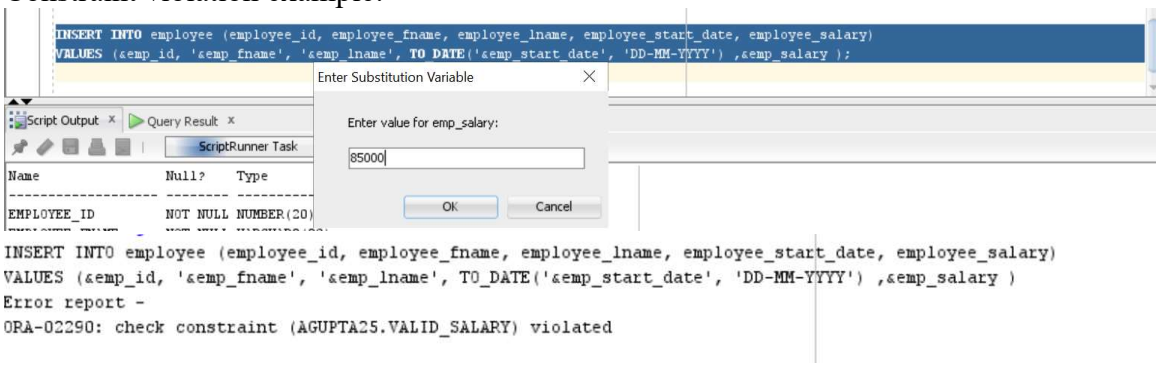




```
INSERT ALL
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1002, 'John', 'Doe', TO_DATE('01-
02-2001', 'DD-MM-YYYY'), 60000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1003, 'Jane', 'Smith', TO_DATE('15-
06-2002', 'DD-MM-YYYY'), 70000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1004, 'Michael', 'Johnson',
TO_DATE('22-11-2001', 'DD-MM-YYYY'), 55000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1005, 'Emily', 'Brown',
TO_DATE('03-05-2003', 'DD-MM-YYYY'), 75000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1006, 'William', 'Taylor',
TO_DATE('14-08-2002', 'DD-MM-YYYY'), 65000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1007, 'Olivia', 'Clark', TO_DATE('10-
03-2001', 'DD-MM-YYYY'), 50000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1008, 'James', 'Garcia',
TO_DATE('27-07-2003', 'DD-MM-YYYY'), 80000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1009, 'Sophia', 'Davis',
TO_DATE('19-01-2002', 'DD-MM-YYYY'), 55000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1010, 'Benjamin', 'Rodriguez',
TO_DATE('07-09-2001', 'DD-MM-YYYY'), 70000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1011, 'Isabella', 'Martinez',
TO_DATE('23-12-2002', 'DD-MM-YYYY'), 60000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1012, 'Daniel', 'Hernandez',
TO_DATE('18-04-2003', 'DD-MM-YYYY'), 75000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1013, 'Mia', 'Lopez', TO_DATE('06-
10-2001', 'DD-MM-YYYY'), 55000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1014, 'Jacob', 'Gonzalez',
TO_DATE('09-02-2002', 'DD-MM-YYYY'), 70000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1015, 'Ava', 'Perez', TO_DATE('28-
05-2003', 'DD-MM-YYYY'), 77000)
  INTO employee (employee_id, employee_fname, employee_lname,
employee_start_date, employee_salary) VALUES (1016, 'Brian', 'Rodriguez',
TO_DATE('12-09-2006', 'DD-MM-YYYY'), 63000)
```

INTO employee (employee_id, employee_fname, employee_lname, employee_start_date, employee_salary) VALUES (1017, 'Megan', 'Gonzalez', TO_DATE('01-11-2004', 'DD-MM-YYYY'), 55000)
INTO employee (employee_id, employee_fname, employee_lname, employee_start_date, employee_salary) VALUES (1018, 'Brandon', 'Wilson', TO_DATE('26-08-2002', 'DD-MM-YYYY'), 72000)
INTO employee (employee_id, employee_fname, employee_lname, employee_start_date, employee_salary) VALUES (1019, 'Rachel', 'Davis', TO_DATE('09-07-2007', 'DD-MM-YYYY'), 51000)
INTO employee (employee_id, employee_fname, employee_lname, employee_start_date, employee_salary) VALUES (1020, 'Justin', 'Moore', TO_DATE('14-03-2005', 'DD-MM-YYYY'), 68000)
SELECT 1 FROM DUAL;

Constraint violation example:



ALTER TABLE employee
DROP CONSTRAINT valid_salary;

Alter table employee add constraint valid_salary check (employee_salary between 50000 and 150000);

```
ALTER TABLE employee
DROP CONSTRAINT valid_salary;

Alter table employee add constraint valid_salary check (employee_salary between 50000 and 150000);
```

Script Output | Query Result

Task completed in 0.023 seconds

```
*Action:

Table EMPLOYEE altered.


Table EMPLOYEE altered.
```

Crystal

CREATE OR REPLACE TYPE team_name_ty AS OBJECT(
f_tname VARCHAR(15),
l_tname VARCHAR(15));

```
CREATE OR REPLACE TYPE team_name_ty AS OBJECT(
f_tname VARCHAR(15),
l_tname VARCHAR(15));
```

```
Type TEAM_NAME_TY compiled
```

CREATE TABLE jersey(
jersey_size VARCHAR(10),
jersey_price VARCHAR(10),
jersey_color_style VARCHAR(10),
jersey_quantity NUMBER,
jersey_material VARCHAR(10),
jersey_num NUMBER(2),
jersey_name VARCHAR(20),
jersey_type VARCHAR(10),
jersey_tname team_name_ty);

```
CREATE TABLE jersey(
jersey_size         VARCHAR(10),
jersey_price        VARCHAR(10),
jersey_color_style  VARCHAR(10),
jersey_quantity     NUMBER,
jersey_material     VARCHAR(10),
jersey_num          NUMBER(2),
jersey_name         VARCHAR(20),
jersey_type         VARCHAR(10),
jersey_tname        team_name_ty);
```

```
Table JERSEY created.
```

ALTER TABLE jersey
ADD jersey_id NUMBER;

```
ALTER TABLE jersey
ADD jersey_id NUMBER;
```

```
Table JERSEY altered.
```

INSERT INTO jersey VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 00, 'Lil Peanut', 'M', team_name_ty('Baltimore', 'Ravens'), 10001);
INSERT INTO jersey VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 23, 'chickenwhopper', 'M', team_name_ty('Kansas City', 'Chiefs'), 10002);

INSERT INTO jersey VALUES('L', '$169.99', 'Alternate', 1, 'Polyester', 23, 'King James', 'M', team_name_ty('Los Angeles', 'Chargers'), 10003);
INSERT INTO jersey VALUES('M', '$169.99', 'Alternate', 1, 'Polyester', 99, 'babymochi','M', team_name_ty('Los Angeles', 'Rams'), 10004);

```
INSERT INTO jersey VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 00, 'Lil Peanut', 'M', team_name_ty('Baltimore', 'Ravens'), 10001);
INSERT INTO jersey VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 23, 'chickenwhopper', 'M', team_name_ty('Kansas City', 'Chiefs'), 10002);
INSERT INTO jersey VALUES('L', '$169.99', 'Alternate', 1, 'Polyester', 23, 'King James', 'M', team_name_ty('Los Angeles', 'Chargers'), 10003);
INSERT INTO jersey VALUES('M', '$169.99', 'Alternate', 1, 'Polyester', 99, 'babymochi','M', team_name_ty('Los Angeles', 'Rams'), 10004);
```

ALTER TABLE jersey
ADD CONSTRAINT jerseyPK PRIMARY KEY (jersey_ID);

```
ALTER TABLE jersey
ADD CONSTRAINT jerseyPK PRIMARY KEY (jersey_ID);

Table JERSEY altered.
```

Below we verify the constraint:

```
INSERT INTO jersey VALUES('S', '$129.99', 'Team', 10, 'Polyester', 00, 'Beckham Jr', 'P', team_name_ty('Baltimore', 'Ravens'), 10004);

Error starting at line : 151 in command -
INSERT INTO jersey VALUES('S', '$129.99', 'Team', 10, 'Polyester', 00, 'Beckham Jr', 'P', team_name_ty('Baltimore', 'Ravens'), 10004)
Error report -
ORA-00001: unique constraint (CWU11.JERSEYPK) violated
```

INSERT INTO jersey VALUES('S', '$129.99', 'Team', 10, 'Polyester', 00, 'Beckham Jr', 'P', team_name_ty('Baltimore', 'Ravens'), 10005);
INSERT INTO jersey VALUES('S', '$149.99', 'Team', 10, 'Polyester', 15, 'Mahomes', 'P', team_name_ty('Kansas City', 'Chiefs'), 10006);
INSERT INTO jersey VALUES('M', '$129.99', 'Primary', 10, 'Polyester', 9, 'Burrow', 'P', team_name_ty('Cincinnati', 'Bengals'), 10007);
INSERT INTO jersey VALUES('XL', '$129.99', 'Primary', 10, 'Polyester', 0, 'Hurts', 'P', team_name_ty('Philadelphia', 'Eagles'), 10008);

```
INSERT INTO jersey VALUES('S', '$129.99', 'Team', 10, 'Polyester', 00, 'Beckham Jr', 'P', team_name_ty('Baltimore', 'Ravens'), 10005);
INSERT INTO jersey VALUES('S', '$149.99', 'Team', 10, 'Polyester', 15, 'Mahomes', 'P', team_name_ty('Kansas City', 'Chiefs'), 10006);
INSERT INTO jersey VALUES('M', '$129.99', 'Primary', 10, 'Polyester', 9, 'Burrow', 'P', team_name_ty('Cincinnati', 'Bengals'), 10007);
INSERT INTO jersey VALUES('XL', '$129.99', 'Primary', 10, 'Polyester', 0,  'Hurts', 'P', team_name_ty('Philadelphia', 'Eagles'), 10008);
```

ALTER TYPE team_name_ty
ADD MEMBER FUNCTION full_team_name RETURN VARCHAR2 CASCADE;

CREATE OR REPLACE TYPE BODY team_name_ty AS
MEMBER FUNCTION full_team_name
RETURN VARCHAR2 IS
  BEGIN
    RETURN(f_tname || ' ' || l_tname);
  END full_team_name;
END;

```
ALTER TYPE team_name_ty
ADD MEMBER FUNCTION full_team_name RETURN VARCHAR2 CASCADE;
```

```
Type TEAM_NAME_TY altered.
```

```
CREATE OR REPLACE TYPE BODY team_name_ty AS
MEMBER FUNCTION full_team_name
RETURN VARCHAR2 IS
    BEGIN
        RETURN(f_tname || ' ' || l_tname);
    END full_team_name;
END;
```

```
Type Body TEAM_NAME_TY compiled
```

CREATE TABLE manufactured_t(
jersey_id NUMBER,
jersey_status VARCHAR(2));

ALTER TABLE manufactured_t
ADD CONSTRAINT manufacturedFK FOREIGN KEY (jersey_id) REFERENCES
jersey(jersey_id);

ALTER TABLE manufactured_t
MODIFY jersey_status VARCHAR(20);

CREATE TABLE purchased_t(
jersey_id NUMBER,
jersey_assurance VARCHAR(10));

ALTER TABLE purchased_t
ADD CONSTRAINT purchasedFK FOREIGN KEY (jersey_id) REFERENCES
jersey(jersey_id);

```
CREATE TABLE manufactured_t(
jersey_id       NUMBER,
jersey_status   VARCHAR(2));
```

```
ALTER TABLE manufactured_t
ADD CONSTRAINT manufacturedFK FOREIGN KEY (jersey_id) REFERENCES jersey(jersey_id);
```

```
ALTER TABLE manufactured_t
MODIFY jersey_status VARCHAR(20);
```

```
CREATE TABLE purchased_t(
jersey_id           NUMBER,
jersey_assurance    VARCHAR(10));

ALTER TABLE purchased_t
ADD CONSTRAINT purchasedFK FOREIGN KEY (jersey_id) REFERENCES jersey(jersey_id);
```

```
Table MANUFACTURED_T altered.
```

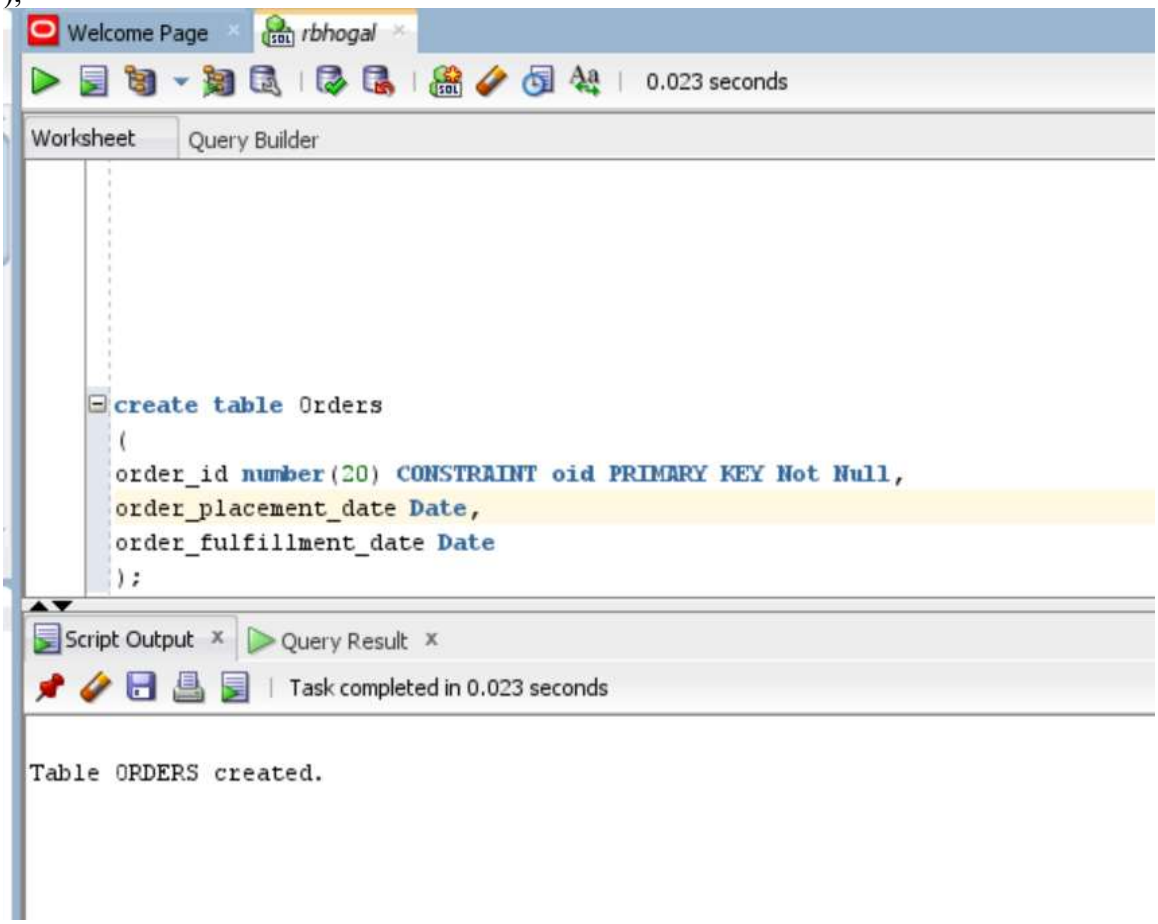
```
Table PURCHASED_T created.


Table PURCHASED_T altered.
```

INSERT INTO manufactured_t VALUES (10001, 'Production');
INSERT INTO manufactured_t VALUES (10002, 'Delivery');
INSERT INTO manufactured_t VALUES (10003, 'Delivery');
INSERT INTO manufactured_t VALUES (10004, 'Processed');
INSERT INTO purchased_t VALUES (10005, 'Free');
INSERT INTO purchased_t VALUES (10006, 'Free');
INSERT INTO purchased_t VALUES (10007, 'Free');
INSERT INTO purchased_t VALUES (10008, 'Free');

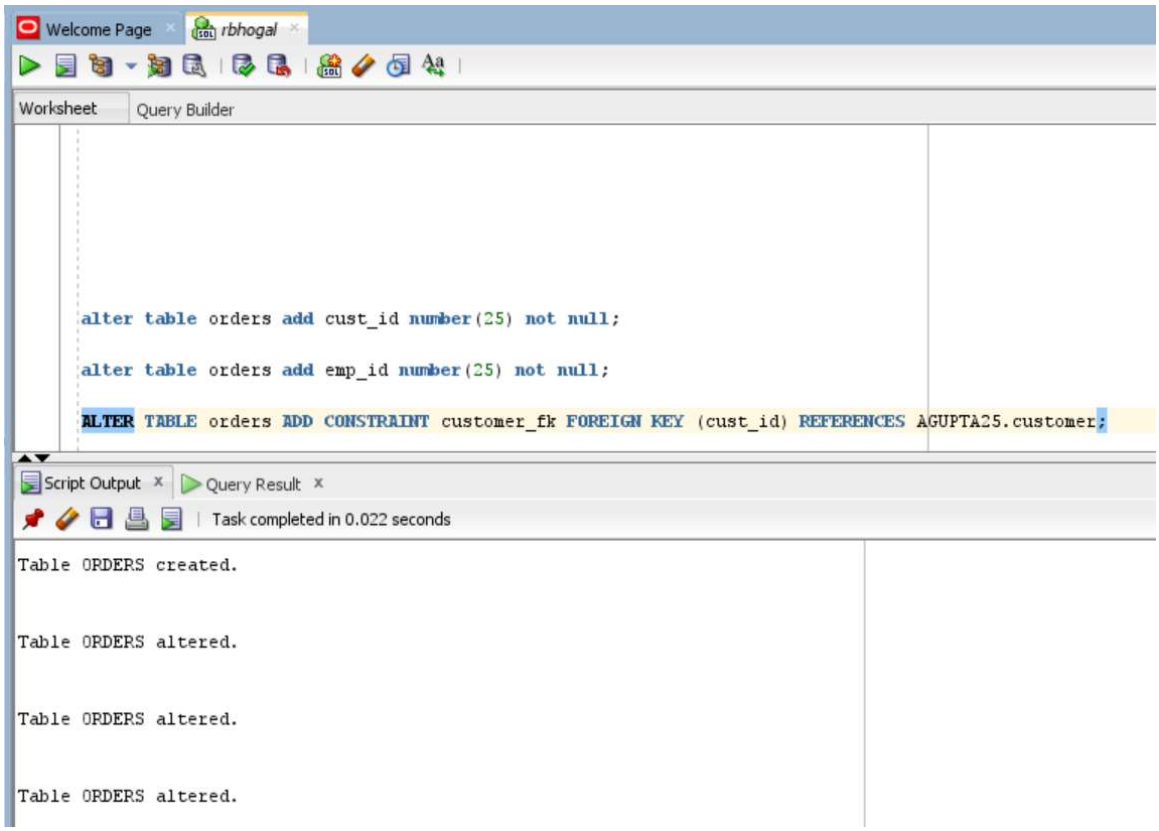
```
INSERT INTO manufactured_t VALUES (10001, 'Production');
INSERT INTO manufactured_t VALUES (10002, 'Delivery');
INSERT INTO manufactured_t VALUES (10003, 'Delivery');
INSERT INTO manufactured_t VALUES (10004, 'Processed');
INSERT INTO purchased_t VALUES (10005, 'Free');
INSERT INTO purchased_t VALUES (10006, 'Free');
INSERT INTO purchased_t VALUES (10007, 'Free');
INSERT INTO purchased_t VALUES (10008, 'Free');
```

Rahul

```
create table Orders
(
order_id number(20) CONSTRAINT oid PRIMARY KEY Not Null,
order_placement_date Date,
order_fulfillment_date Date
);
```




```
alter table orders add cust_id number(25) not null;

alter table orders add emp_id number(25) not null;

ALTER TABLE orders ADD CONSTRAINT customer_fk FOREIGN KEY (cust_id)
REFERENCES AGUPTA25.customer1;

ALTER TABLE orders ADD CONSTRAINT emp_fk FOREIGN KEY (emp_id)
REFERENCES AGUPTA25.employee;
```



insert into Orders values
(
1, '10-APR-23', '15-APR-23',101,1000
);

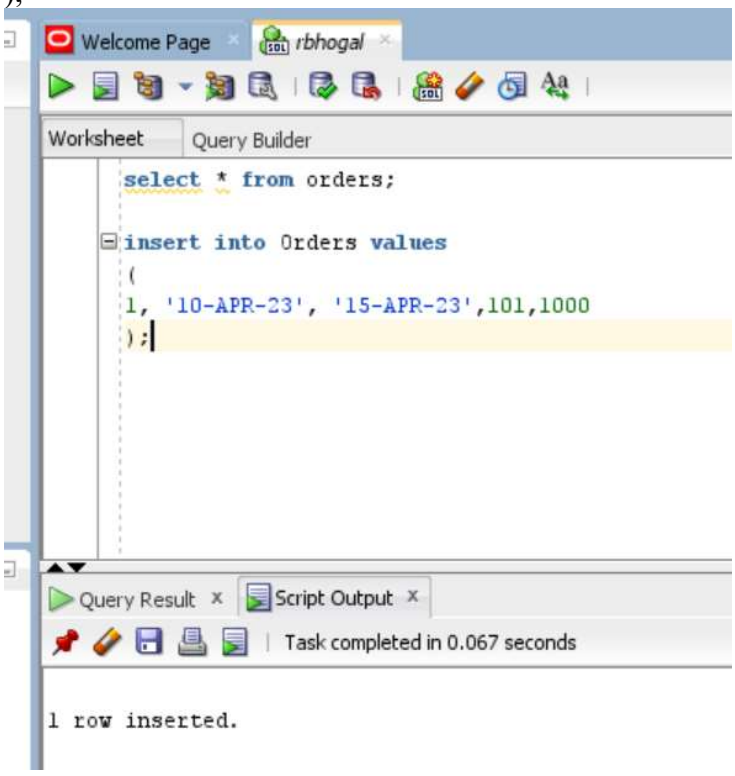
insert into Orders values
(
2, '11-APR-23', '16-APR-23',102,1001
);

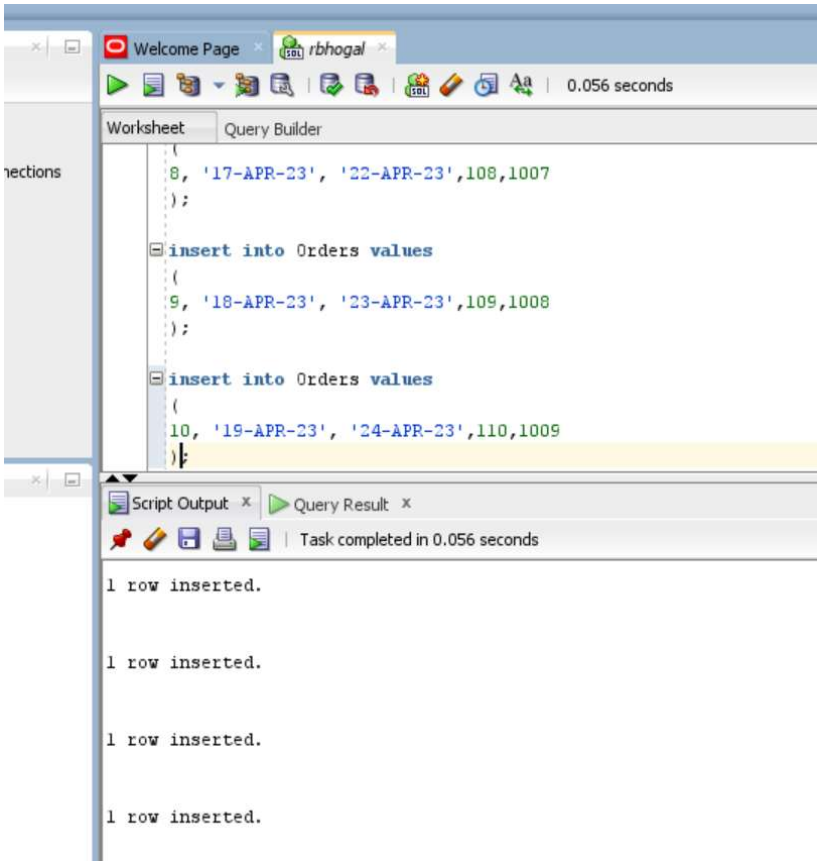
insert into Orders values
(
3, '12-APR-23', '17-APR-23',103,1002
);

insert into Orders values
(
4, '13-APR-23', '18-APR-23',104,1003
);

insert into Orders values
(
5, '14-APR-23', '19-APR-23',105,1004

```
);

insert into Orders values
(
6, '15-APR-23', '20-APR-23',106,1005
);

insert into Orders values
(
7, '16-APR-23', '21-APR-23',107,1006
);

insert into Orders values
(
8, '17-APR-23', '22-APR-23',108,1007
);

insert into Orders values
(
9, '18-APR-23', '23-APR-23',109,1008
);

insert into Orders values
(
10, '19-APR-23', '24-APR-23',110,1009
);
```

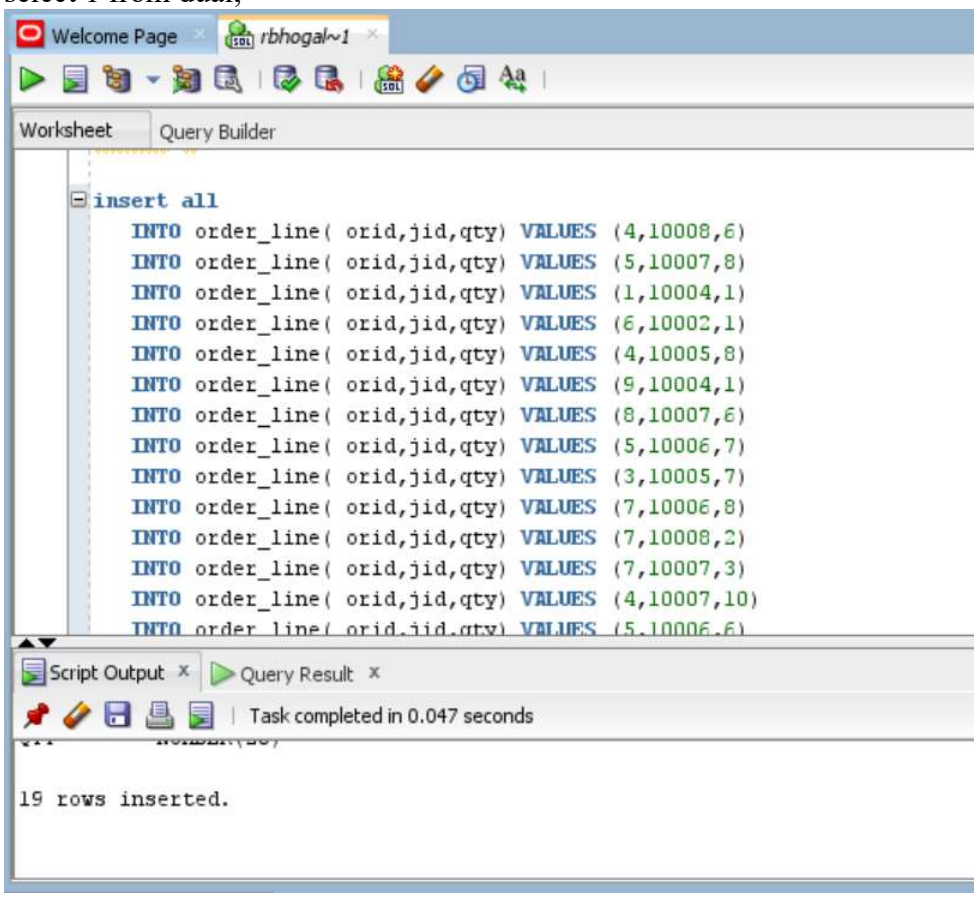




drop table order_line cascade constraints;

CREATE TABLE order_line (
orid NUMBER(20) ,
jid NUMBER(20) ,
qty NUMBER(20) DEFAULT 1);
ALter table order_line add CONSTRAINT fk_oid FOREIGN KEY (orid)
REFERENCES orders(order_id);
Alter table order_line add CONSTRAINT fk_jid FOREIGN KEY (jid) REFERENCES
CWU11.jersey;
insert all
  INTO order_line(orid,jid,qty) VALUES (4,10008,6)
  INTO order_line(orid,jid,qty) VALUES (5,10007,8)
  INTO order_line(orid,jid,qty) VALUES (1,10004,1)
  INTO order_line(orid,jid,qty) VALUES (6,10002,1)
  INTO order_line(orid,jid,qty) VALUES (4,10005,8)
  INTO order_line(orid,jid,qty) VALUES (9,10004,1)
  INTO order_line(orid,jid,qty) VALUES (8,10007,6)
  INTO order_line(orid,jid,qty) VALUES (5,10006,7)
  INTO order_line(orid,jid,qty) VALUES (3,10005,7)
  INTO order_line(orid,jid,qty) VALUES (7,10006,8)

INTO order_line(orid,jid,qty) VALUES (7,10008,2)
INTO order_line(orid,jid,qty) VALUES (7,10007,3)
INTO order_line(orid,jid,qty) VALUES (4,10007,10)
INTO order_line(orid,jid,qty) VALUES (5,10006,6)
INTO order_line(orid,jid,qty) VALUES (9,10001,1)
INTO order_line(orid,jid,qty) VALUES (10,10003,1)
INTO order_line(orid,jid,qty) VALUES (1,10005,10)
INTO order_line(orid,jid,qty) VALUES (7,10002,1)
INTO order_line(orid,jid,qty) VALUES (6,10001,1)
select 1 from dual;



Database Structure

Aishwarya

desc customer1;

```
desc customer1;
```

Script Output | Query Result

Task completed in 0.055 seconds

```
Name                 Null?    Type
-------------------- -------- ------------
CUSTOMER_ID          NOT NULL NUMBER
NAME                          NAME_TYPE1
CUSTOMER_STREET_ADD  NOT NULL VARCHAR2(50)
CUSTOMER_CITY        NOT NULL VARCHAR2(20)
CUSTOMER_STATE       NOT NULL VARCHAR2(20)
CUSTOMER_ZIPCODE     NOT NULL VARCHAR2(10)
PHN_NO                        NUMBER(10)
```

Desc employee;

```
Desc employee;
```

Script Output | Query Result

Task completed in 0.02 seconds

```
Name                 Null?    Type
-------------------- -------- ------------
EMPLOYEE_ID          NOT NULL NUMBER(20)
EMPLOYEE_FNAME       NOT NULL VARCHAR2(20)
EMPLOYEE_LNAME       NOT NULL VARCHAR2(20)
EMPLOYEE_START_DATE           DATE
EMPLOYEE_SALARY               NUMBER(20)
```

Crystal

Desc jersey;

```
Name                 Null?    Type
-------------------- -------- ------------
JERSEY_SIZE                   VARCHAR2(10)
JERSEY_PRICE                  VARCHAR2(10)
JERSEY_COLOR_STYLE            VARCHAR2(10)
JERSEY_QUANTITY               NUMBER
JERSEY_MATERIAL               VARCHAR2(10)
JERSEY_NUM                    NUMBER(2)
JERSEY_NAME                   VARCHAR2(20)
JERSEY_TYPE                   VARCHAR2(10)
JERSEY_TNAME                  TEAM_NAME_TY
JERSEY_ID            NOT NULL NUMBER
```

Desc manufactured_t;

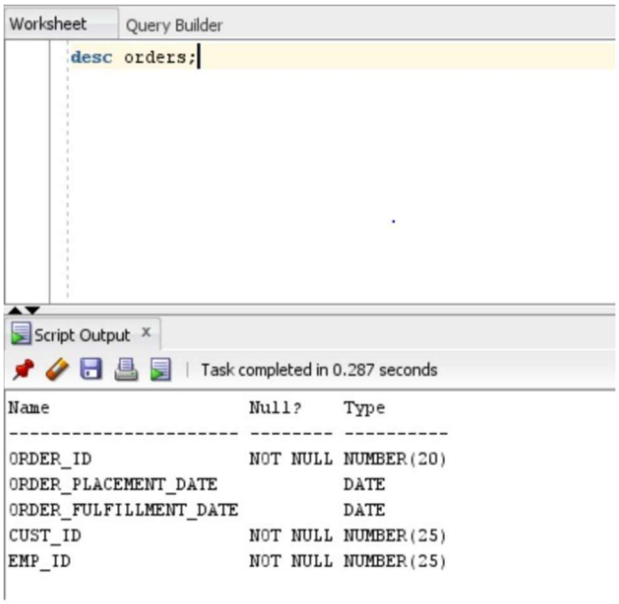
```
Name           Null? Type
-------------- ----- ------------
JERSEY_ID            NUMBER
JERSEY_STATUS        VARCHAR2(20)
```

Desc purchased_t;

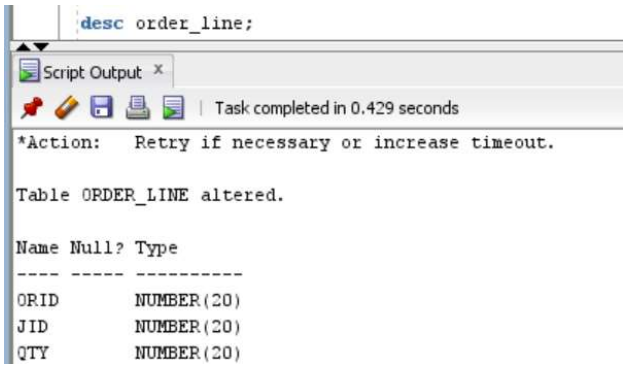
```
Name              Null? Type
----------------- ----- ------------
JERSEY_ID               NUMBER
JERSEY_ASSURANCE        VARCHAR2(10)
```

Rahul

Desc orders;



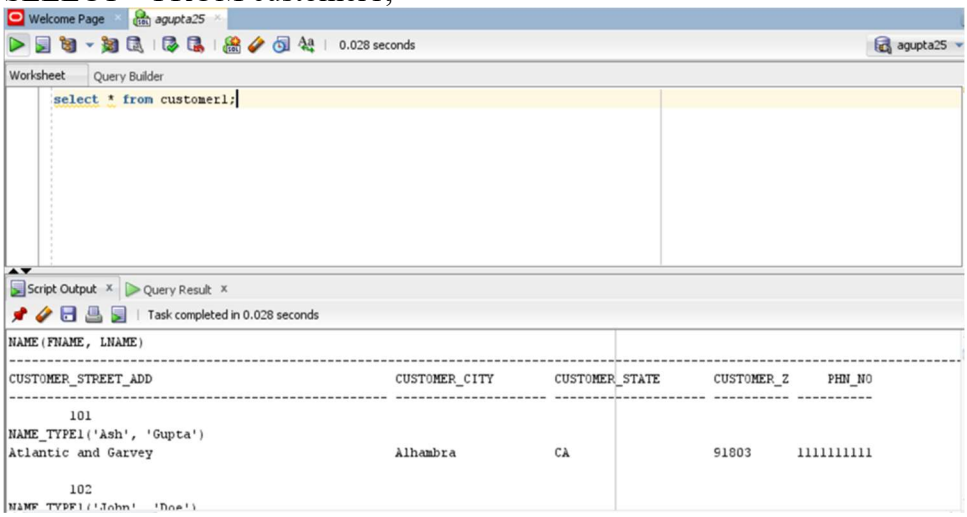
Desc order_line;



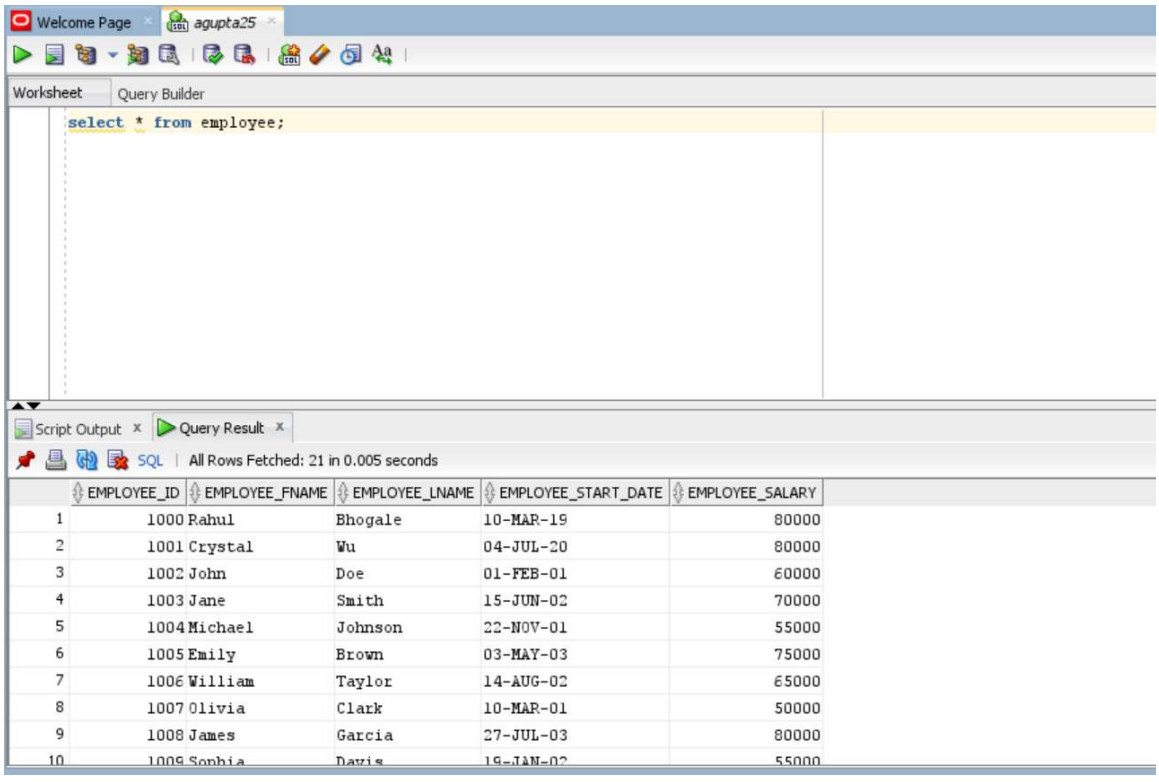
Database Instance

Aishwarya

SELECT * FROM customer1;



SELECT * FROM employee;



	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SALARY
1	1000	Rahul	Bhogale	10-MAR-19	80000
2	1001	Crystal	Wu	04-JUL-20	80000
3	1002	John	Doe	01-FEB-01	60000
4	1003	Jane	Smith	15-JUN-02	70000
5	1004	Michael	Johnson	22-NOV-01	55000
6	1005	Emily	Brown	03-MAY-03	75000
7	1006	William	Taylor	14-AUG-02	65000
8	1007	Olivia	Clark	10-MAR-01	50000
9	1008	James	Garcia	27-JUL-03	80000
10	1009	Sophia	Davis	19-JAN-02	55000

Crystal
SELECT * FROM jersey;

	JERSEY_SIZE	JERSEY_PRICE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID
1	S	$169.99	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEA...	10001
2	S	$169.99	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEA...	10002
3	L	$169.99	Alternate	1	Polyester	23	King James	M	[CWU11.TEA...	10003
4	M	$169.99	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEA...	10004
5	S	$149.99	Team	10	Polyester	15	Mahomes	P	[CWU11.TEA...	10006
6	M	$129.99	Primary	10	Polyester	9	Burrow	P	[CWU11.TEA...	10007
7	XL	$129.99	Primary	10	Polyester	0	Hurts	P	[CWU11.TEA...	10008
8	S	$129.99	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEA...	10005

```
JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------
 JERSEY_ID
----------
S          $169.99    Alternate               1 Polyester           0 Lil Peanut           M
TEAM_NAME_TY('Baltimore', 'Ravens')
     10001

S          $169.99    Alternate               1 Polyester          23 chickenwhopper       M
TEAM_NAME_TY('Kansas City', 'Chiefs')
     10002

JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------
 JERSEY_ID
----------

L          $169.99    Alternate               1 Polyester          23 King James          M
TEAM_NAME_TY('Los Angeles', 'Chargers')
     10003

M          $169.99    Alternate               1 Polyester          99 babymochi           M
TEAM_NAME_TY('Los Angeles', 'Rams')
```

```
JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------
 JERSEY_ID
----------
     10004

S          $149.99    Team                    10 Polyester          15 Mahomes              P
TEAM_NAME_TY('Kansas City', 'Chiefs')
     10006

M          $129.99    Primary                 10 Polyester           9 Burrow               P

JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------
 JERSEY_ID
----------
TEAM_NAME_TY('Cincinnati', 'Bengals')
     10007

XL         $129.99    Primary                 10 Polyester           0 Hurts                P
TEAM_NAME_TY('Philadelphia', 'Eagles')
     10008

JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------
 JERSEY_ID
----------
S          $129.99    Team                    10 Polyester           0 Beckham Jr           P
TEAM_NAME_TY('Baltimore', 'Ravens')
     10005


8 rows selected.
```

SELECT * FROM manufactured_t;

	JERSEY_ID	JERSEY_STATUS
1	10001	Production
2	10002	Delivery
3	10003	Delivery
4	10004	Processed

SELECT * FROM purchased_t;

	JERSEY_ID	JERSEY_ASSURANCE
1	10005	Free
2	10006	Free
3	10007	Free
4	10008	Free

Rahul

select * from orders;

```
select * from orders;
```

Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.005 seconds

	ORDER_ID	ORDER_PLACEMENT_DATE	ORDER_FULFILLMENT_DATE	CUST_ID	EMP_ID
1	1	10-APR-23	15-APR-23	101	1000
2	2	11-APR-23	16-APR-23	102	1001
3	3	12-APR-23	17-APR-23	103	1002
4	4	13-APR-23	18-APR-23	104	1003
5	5	14-APR-23	19-APR-23	105	1004
6	6	15-APR-23	20-APR-23	106	1005
7	7	16-APR-23	21-APR-23	107	1006
8	8	17-APR-23	22-APR-23	108	1007
9	9	18-APR-23	23-APR-23	109	1008
10	10	19-APR-23	24-APR-23	110	1009

select * from order_line;

```
select * from order_line;


commit;

delete from order_line where qty>=5;
```

Script Output | Query Result

SQL | All Rows Fetched: 9 in 0.003 seconds

	ORID	JID	...
1	7	10007	3
2	7	10008	2
3	6	10002	1
4	9	10004	1
5	6	10001	1
6	9	10001	1

erform "Go to Declaration"

Update, Delete, and Create View

Crystal

Change jersey_price data type from VARCHAR to NUMBER so we can perform arithmetic functions

```
ALTER TABLE jersey
DROP COLUMN jersey_price;

ALTER TABLE jersey
ADD jersey_price NUMBER;

UPDATE jersey
SET jersey_price = 169.99
WHERE jersey_ID IN (10001, 10002, 10003, 10004);

UPDATE jersey
SET jersey_price = 129.99
WHERE jersey_ID IN (10005, 10006);

UPDATE jersey
SET jersey_price = 139.99
WHERE jersey_ID IN (10007, 10008);
```

	JERSEY_...	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID	JERSEY_PRICE
1	S	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEAM_NAME_TY]	10001	169.99
2	S	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEAM_NAME_TY]	10002	169.99
3	L	Alternate	1	Polyester	23	King James	M	[CWU11.TEAM_NAME_TY]	10003	169.99
4	M	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEAM_NAME_TY]	10004	169.99
5	S	Team	10	Polyester	15	Mahomes	P	[CWU11.TEAM_NAME_TY]	10006	129.99
6	M	Primary	10	Polyester	9	Burrow	P	[CWU11.TEAM_NAME_TY]	10007	139.99
7	XL	Primary	10	Polyester	0	Hurts	P	[CWU11.TEAM_NAME_TY]	10008	139.99
8	S	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEAM_NAME_TY]	10005	129.99

```
JERSEY_SIZ JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------
 JERSEY_ID JERSEY_PRICE
---------- ------------
S          Alternate               1 Polyester           0 Lil Peanut           M
TEAM_NAME_TY('Baltimore', 'Ravens')
     10001       169.99

S          Alternate               1 Polyester          23 chickenwhopper       M
TEAM_NAME_TY('Kansas City', 'Chiefs')
     10002       169.99

JERSEY_SIZ JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------
 JERSEY_ID JERSEY_PRICE
---------- ------------

L          Alternate               1 Polyester          23 King James          M
TEAM_NAME_TY('Los Angeles', 'Chargers')
     10003       169.99

M          Alternate               1 Polyester          99 babymochi           M
TEAM_NAME_TY('Los Angeles', 'Rams')
```

```
JERSEY_SIZ JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------
 JERSEY_ID JERSEY_PRICE
---------- ------------
     10004       169.99

S          Team                    10 Polyester          15 Mahomes              P
TEAM_NAME_TY('Kansas City', 'Chiefs')
     10006       129.99

M          Primary                 10 Polyester           9 Burrow               P

JERSEY_SIZ JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------
 JERSEY_ID JERSEY_PRICE
---------- ------------
TEAM_NAME_TY('Cincinnati', 'Bengals')
     10007       139.99

XL         Primary                 10 Polyester           0 Hurts                P
TEAM_NAME_TY('Philadelphia', 'Eagles')
     10008       139.99
```

```
JERSEY_SIZ JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_NAME          JERSEY_TYP
---------- ---------- --------------- ---------- ---------- -------------------- ----------
JERSEY_TNAME(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------
 JERSEY_ID JERSEY_PRICE
---------- ------------
S          Team                      10 Polyester           0 Beckham Jr           P
TEAM_NAME_TY('Baltimore', 'Ravens')
     10005       129.99


8 rows selected.
```

```
Name                 Null?    Type
-------------------- -------- ------------
JERSEY_SIZE                   VARCHAR2(10)
JERSEY_COLOR_STYLE            VARCHAR2(10)
JERSEY_QUANTITY               NUMBER
JERSEY_MATERIAL               VARCHAR2(10)
JERSEY_NUM                    NUMBER(2)
JERSEY_NAME                   VARCHAR2(20)
JERSEY_TYPE                   VARCHAR2(10)
JERSEY_TNAME                  TEAM_NAME_TY
JERSEY_ID            NOT NULL NUMBER
JERSEY_PRICE                  NUMBER
```

Create a unique constraint for signed jerseys

```
ALTER TABLE jersey
ADD Jersey_Sign varchar(20);

ALTER TABLE jersey
ADD CONSTRAINT U_Jersey UNIQUE (Jersey_Sign);
```

```
Name                 Null?    Type
-------------------- -------- ------------
JERSEY_SIZE                   VARCHAR2(10)
JERSEY_COLOR_STYLE            VARCHAR2(10)
JERSEY_QUANTITY               NUMBER
JERSEY_MATERIAL               VARCHAR2(10)
JERSEY_NUM                    NUMBER(2)
JERSEY_NAME                   VARCHAR2(20)
JERSEY_TYPE                   VARCHAR2(10)
JERSEY_TNAME                  TEAM_NAME_TY
JERSEY_ID            NOT NULL NUMBER
JERSEY_PRICE                  NUMBER
JERSEY_SIGN                   VARCHAR2(20)
```

	JERSEY_SIZE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID	JERSEY_PRICE	JERSEY_SIGN
1	S	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEAM_NAME_TY]	10001	169.99	(null)
2	S	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEAM_NAME_TY]	10002	169.99	(null)
3	L	Alternate	1	Polyester	23	King James	M	[CWU11.TEAM_NAME_TY]	10003	169.99	(null)
4	M	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEAM_NAME_TY]	10004	169.99	(null)
5	S	Team	10	Polyester	15	Mahomes	P	[CWU11.TEAM_NAME_TY]	10006	129.99	(null)
6	M	Primary	10	Polyester	9	Burrow	P	[CWU11.TEAM_NAME_TY]	10007	139.99	(null)
7	XL	Primary	10	Polyester	0	Hurts	P	[CWU11.TEAM_NAME_TY]	10008	139.99	(null)
8	S	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEAM_NAME_TY]	10005	129.99	(null)

Below we test the unique constraint we just created:

```
UPDATE jersey
SET Jersey_Sign = 'PM04302023'
WHERE Jersey_ID IN (10006, 10007);
```

```
UPDATE jersey
SET Jersey_Sign = 'PM04302023'
WHERE Jersey_ID IN (10006, 10007)
Error report -
ORA-00001: unique constraint (CWU11.U_JERSEY) violated
```

We see that the unique constraint is violated because we entered the same value for jersey sign for two separate jersey products.

```
ALTER TABLE jersey
DROP COLUMN Jersey_Sign;
```

	JERSEY_SIZE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID	JERSEY_PRICE
1	S	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEAM_NAME_TY]	10001	169.99
2	S	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEAM_NAME_TY]	10002	169.99
3	L	Alternate	1	Polyester	23	King James	M	[CWU11.TEAM_NAME_TY]	10003	169.99
4	M	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEAM_NAME_TY]	10004	169.99
5	S	Team	10	Polyester	15	Mahomes	P	[CWU11.TEAM_NAME_TY]	10006	129.99
6	M	Primary	10	Polyester	9	Burrow	P	[CWU11.TEAM_NAME_TY]	10007	139.99
7	XL	Primary	10	Polyester	0	Hurts	P	[CWU11.TEAM_NAME_TY]	10008	139.99
8	S	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEAM_NAME_TY]	10005	129.99

Create a view for customers in New York and California

CREATE OR REPLACE VIEW NY_CA
AS SELECT c.name.fname AS Customer_Fname, c.name.lname AS Customer_Lname, c.customer_state
FROM agupta25.customer1 c
WHERE c.customer_state IN ('NY', 'CA');

```
--Create a view for customers in NY and CA
CREATE OR REPLACE VIEW NY_CA
AS SELECT c.name.fname AS Customer_Fname, c.name.lname AS Customer_Lname, c.customer_state
FROM agupta25.customer1 c
WHERE c.customer_state IN ('NY', 'CA');
```

SELECT * FROM NY_CA;

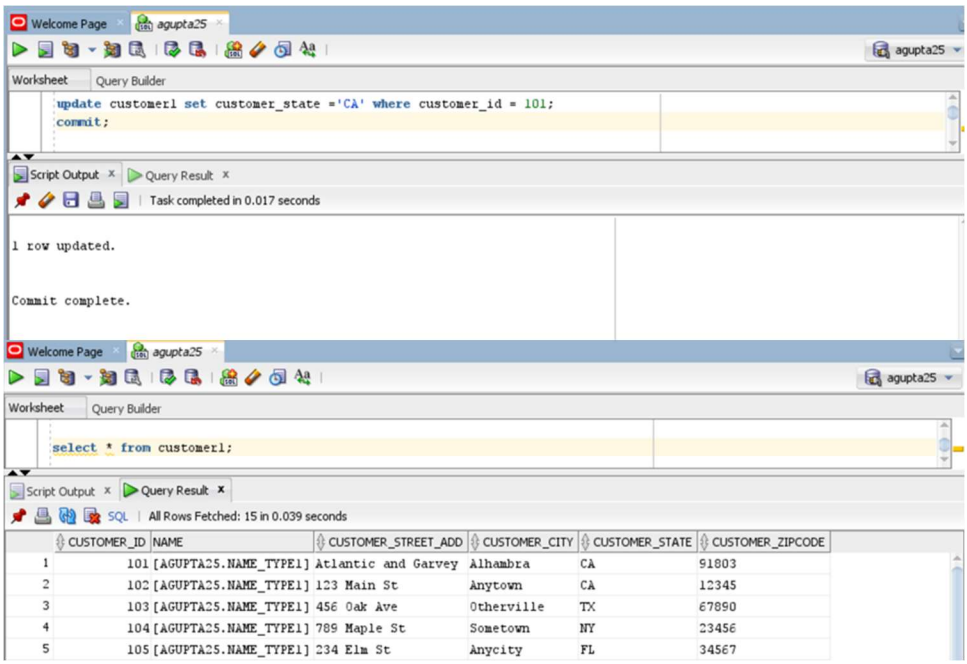
```
SELECT * FROM NY_CA;
```

	CUSTOMER_FNAME	CUSTOMER_LNAME	CUSTOMER_STATE
1	Ash	Gupta	CA
2	John	Doe	CA
3	Bob	Johnson	NY
4	Sarah	Taylor	NY
5	Karen	Davis	CA
6	Lisa	Lee	NY

Aishwarya

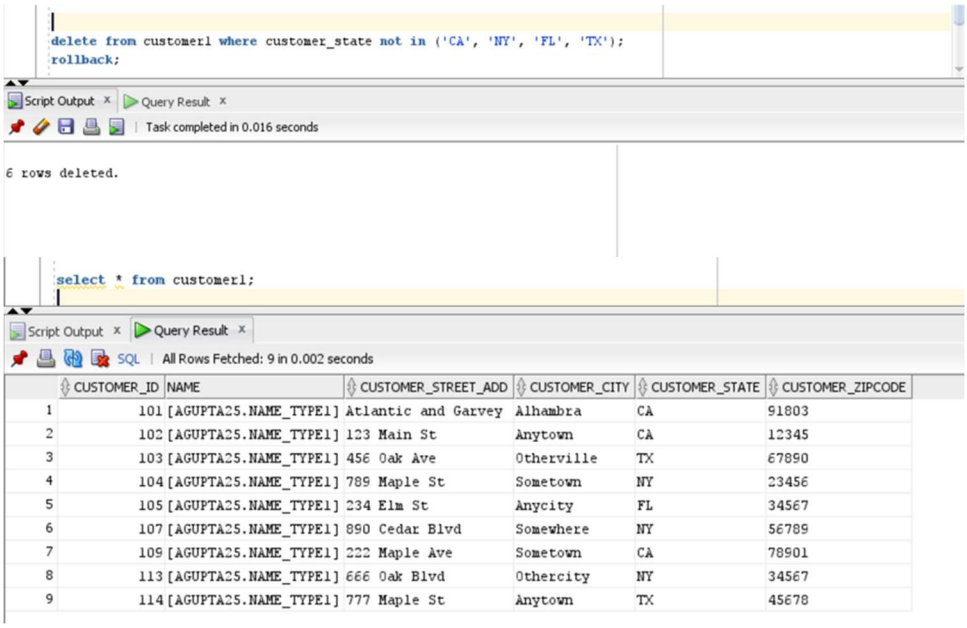
Update statement for customer ID#101 to ‘CA’

```
update customer1 set customer_state ='CA' where customer_id = 101;
commit;
```



Delete customers who are not located in CA, NY, FL, or TX

```
delete from customer1 where customer_state not in ('CA', 'NY', 'FL', 'TX');
rollback;
```



```
delete from customer1 where customer_state not in ('CA', 'NY', 'FL', 'TX');
rollback;
```

Script Output | Query Result | Task completed in 0.016 seconds

6 rows deleted.

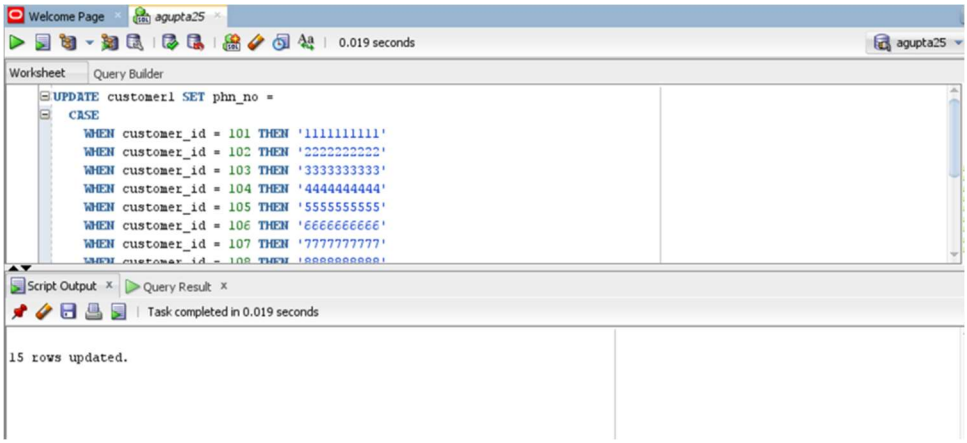
```
select * from customer1;
```

Script Output | Query Result | SQL | All Rows Fetched: 9 in 0.002 seconds

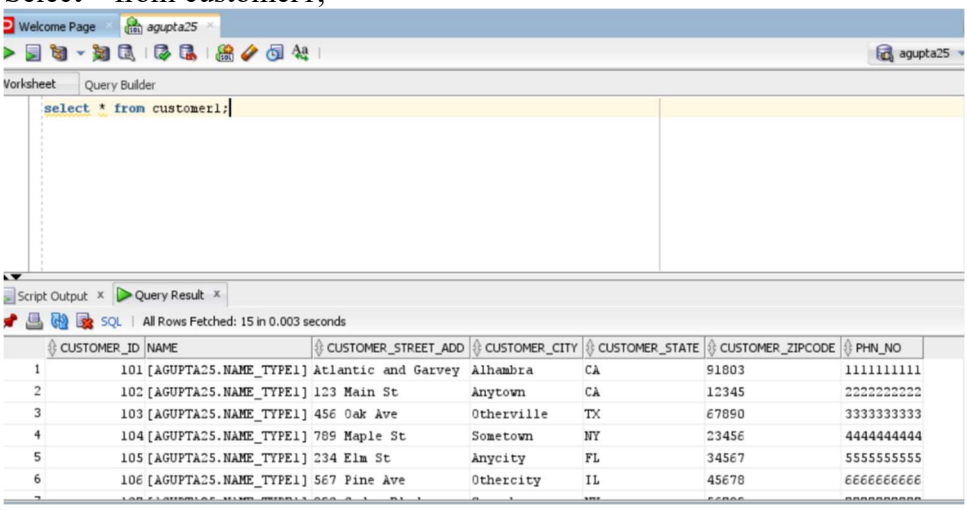
	CUSTOMER_ID	NAME	CUSTOMER_STREET_ADD	CUSTOMER_CITY	CUSTOMER_STATE	CUSTOMER_ZIPCODE
1	101	[AGUPTA25.NAME_TYPE1]	Atlantic and Garvey	Alhambra	CA	91803
2	102	[AGUPTA25.NAME_TYPE1]	123 Main St	Anytown	CA	12345
3	103	[AGUPTA25.NAME_TYPE1]	456 Oak Ave	Otherville	TX	67890
4	104	[AGUPTA25.NAME_TYPE1]	789 Maple St	Sometown	NY	23456
5	105	[AGUPTA25.NAME_TYPE1]	234 Elm St	Anycity	FL	34567
6	107	[AGUPTA25.NAME_TYPE1]	890 Cedar Blvd	Somewhere	NY	56789
7	109	[AGUPTA25.NAME_TYPE1]	222 Maple Ave	Sometown	CA	78901
8	113	[AGUPTA25.NAME_TYPE1]	666 Oak Blvd	Othercity	NY	34567
9	114	[AGUPTA25.NAME_TYPE1]	777 Maple St	Anytown	TX	45678

Update customer table with unique phone number values as shown below:

```
alter table customer1 add phn_no number(10) unique;
alter table customer1 drop column phn_no;


UPDATE customer1 SET phn_no =
  CASE
    WHEN customer_id = 101 THEN '1111111111'
    WHEN customer_id = 102 THEN '2222222222'
    WHEN customer_id = 103 THEN '3333333333'
    WHEN customer_id = 104 THEN '4444444444'
    WHEN customer_id = 105 THEN '5555555555'
    WHEN customer_id = 106 THEN '6666666666'
    WHEN customer_id = 107 THEN '7777777777'
    WHEN customer_id = 108 THEN '8888888888'
    WHEN customer_id = 109 THEN '9999999999'
    WHEN customer_id = 110 THEN '1112223333'
    WHEN customer_id = 111 THEN '2223334444'
    WHEN customer_id = 112 THEN '3334445555'
    WHEN customer_id = 113 THEN '4445556666'
    WHEN customer_id = 114 THEN '5556667777'
    WHEN customer_id = 115 THEN '6667778888'
  END;
```

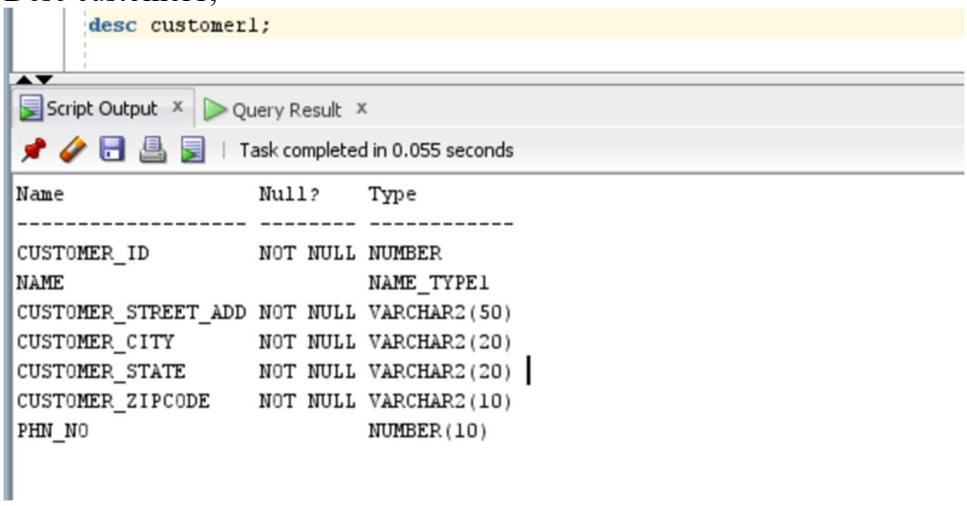


Select * from customer1;



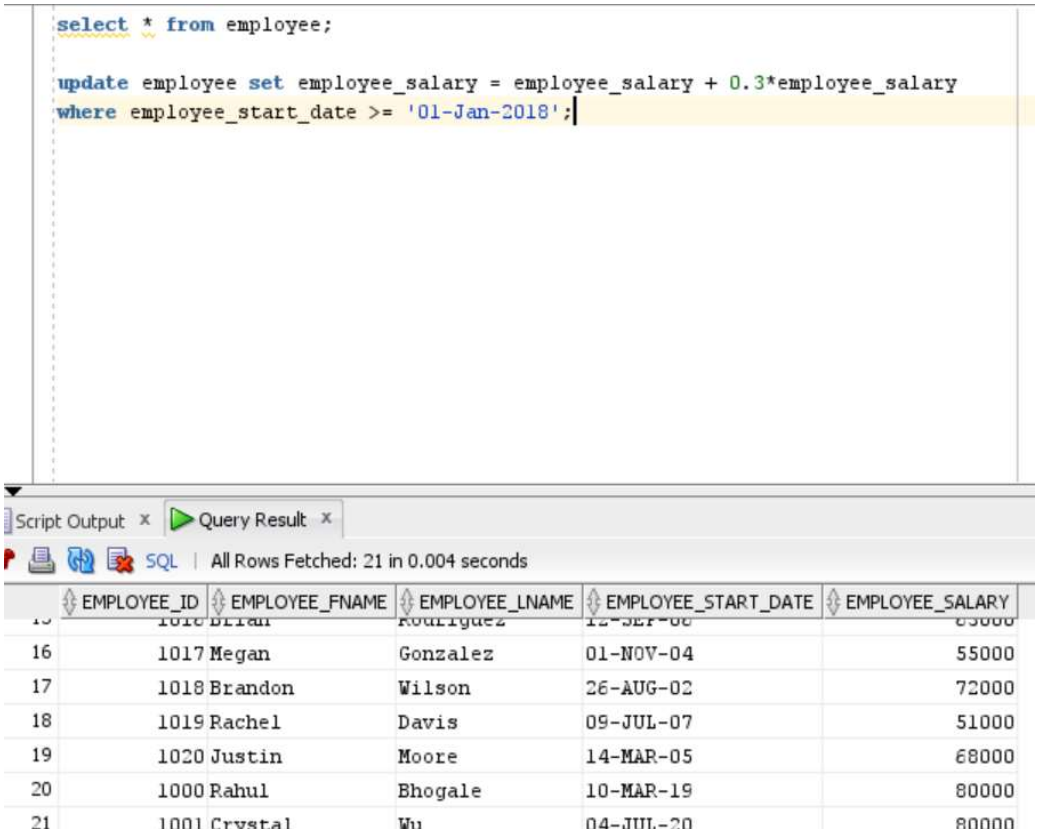
New Structure for Customer table after update of new column ie phn_no:

Desc customer1;



Update employee salary after start date of 01/01/2018

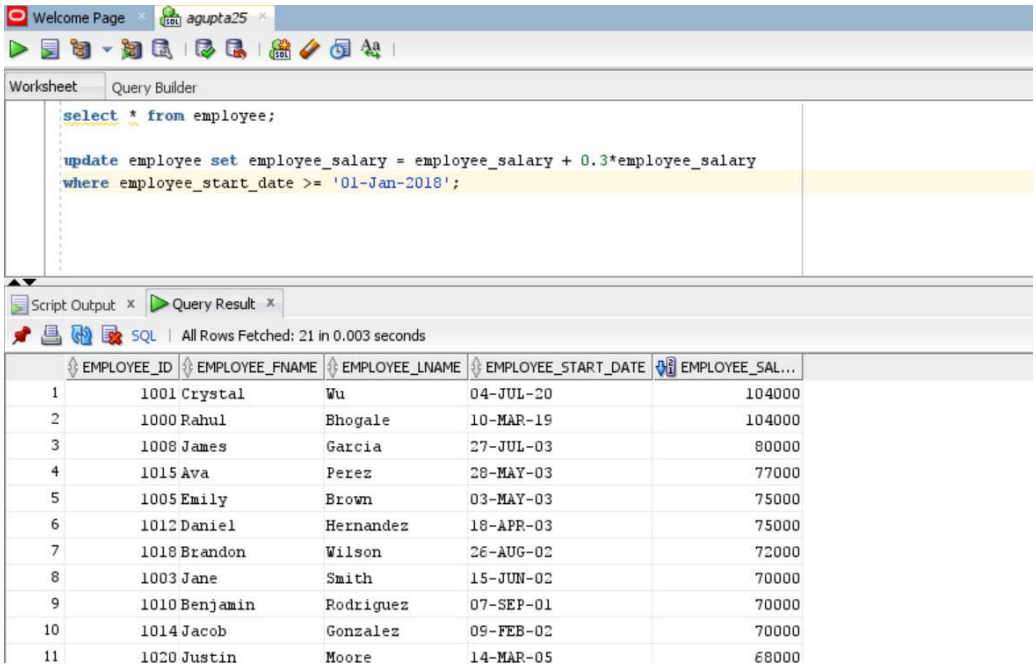
update employee set employee_salary = employee_salary + 0.3*employee_salary
where employee_start_date>='01-Jan-2018' ;



```
select * from employee;

update employee set employee_salary = employee_salary + 0.3*employee_salary
where employee_start_date >= '01-Jan-2018';
```

Script Output | Query Result | SQL | All Rows Fetched: 21 in 0.004 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SALARY
16	1017	Megan	Gonzalez	01-NOV-04	55000
17	1018	Brandon	Wilson	26-AUG-02	72000
18	1019	Rachel	Davis	09-JUL-07	51000
19	1020	Justin	Moore	14-MAR-05	68000
20	1000	Rahul	Bhogale	10-MAR-19	80000
21	1001	Crystal	Wu	04-JUL-20	80000



Welcome Page | agupta25

Worksheet | Query Builder

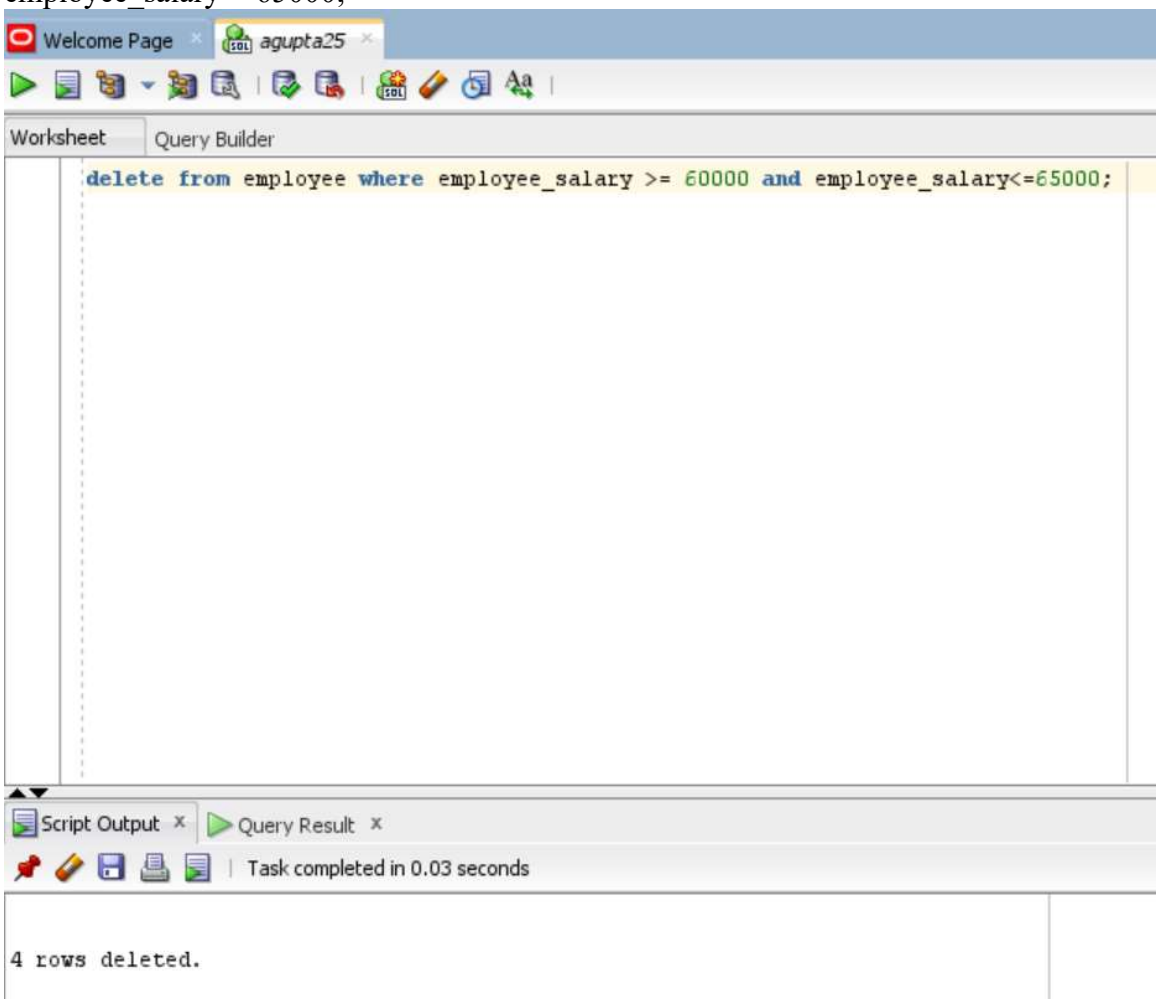
```
select * from employee;

update employee set employee_salary = employee_salary + 0.3*employee_salary
where employee_start_date >= '01-Jan-2018';
```

Script Output | Query Result | SQL | All Rows Fetched: 21 in 0.003 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SAL...
1	1001	Crystal	Wu	04-JUL-20	104000
2	1000	Rahul	Bhogale	10-MAR-19	104000
3	1008	James	Garcia	27-JUL-03	80000
4	1015	Ava	Perez	28-MAY-03	77000
5	1005	Emily	Brown	03-MAY-03	75000
6	1012	Daniel	Hernandez	18-APR-03	75000
7	1018	Brandon	Wilson	26-AUG-02	72000
8	1003	Jane	Smith	15-JUN-02	70000
9	1010	Benjamin	Rodriguez	07-SEP-01	70000
10	1014	Jacob	Gonzalez	09-FEB-02	70000
11	1020	Justin	Moore	14-MAR-05	68000

Delete employees whose salary is between 60000 and 65000

delete from employee where employee_salary >= 60000 and employee_salary<=65000;



Worksheet | Query Builder

```
delete from employee where employee_salary >= 60000 and employee_salary<=65000;

select * from employee;
```

Script Output | Query Result

SQL | All Rows Fetched: 17 in 0.004 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SAL...
1	1007	Olivia	Clark	10-MAR-01	50000
2	1019	Rachel	Davis	09-JUL-07	51000
3	1004	Michael	Johnson	22-NOV-01	55000
4	1017	Megan	Gonzalez	01-NOV-04	55000
5	1013	Mia	Lopez	06-OCT-01	55000
6	1009	Sophia	Davis	19-JAN-02	55000
7	1020	Justin	Moore	14-MAR-05	68000
8	1010	Benjamin	Rodriguez	07-SEP-01	70000
9	1014	Jacob	Gonzalez	09-FEB-02	70000
10	1003	Jane	Smith	15-JUN-02	70000
11	1018	Brandon	Wilson	26-AUG-02	72000
12	1012	Daniel	Hernandez	18-APR-03	75000

```
select * from employee;

rollback;

commit;
```

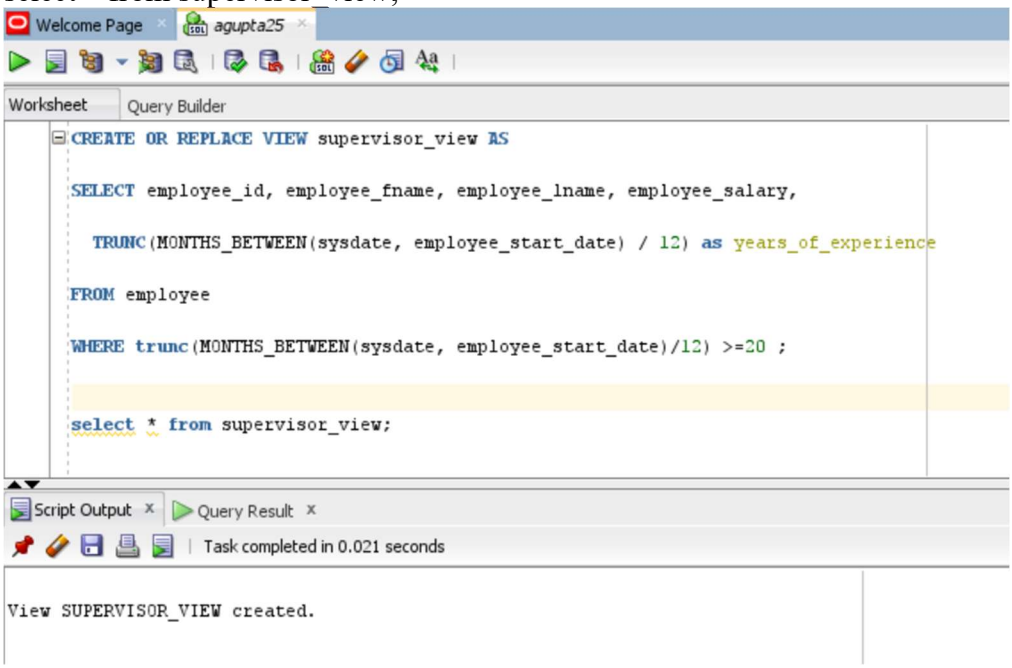
Script Output | Query Result

SQL | All Rows Fetched: 21 in 0.003 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SAL...
1	1007	Olivia	Clark	10-MAR-01	50000
2	1019	Rachel	Davis	09-JUL-07	51000
3	1004	Michael	Johnson	22-NOV-01	55000
4	1013	Mia	Lopez	06-OCT-01	55000
5	1009	Sophia	Davis	19-JAN-02	55000
6	1017	Megan	Gonzalez	01-NOV-04	55000
7	1002	John	Doe	01-FEB-01	60000
8	1011	Isabella	Martinez	23-DEC-02	60000
9	1016	Brian	Rodriguez	12-SEP-06	63000
10	1006	William	Taylor	14-AUG-02	65000
11	1020	Justin	Moore	14-MAR-05	68000
12	1014	Jacob	Gonzalez	09-FEB-02	70000
13	1010	Benjamin	Rodriguez	07-SEP-01	70000

Create views (supervisor view) for years of experience

```
CREATE OR REPLACE VIEW supervisor_view AS
SELECT employee_id, employee_fname, employee_lname, employee_salary,
  TRUNC(MONTHS_BETWEEN(sysdate, employee_start_date) / 12) as
years_of_experience
FROM employee
WHERE trunc(MONTHS_BETWEEN(sysdate, employee_start_date)/12) >=20 ;
drop view supervisor_view;
select * from supervisor_view;
```



```
select * from supervisor_view;
```

Script Output | Query Result

SQL | All Rows Fetched: 11 in 0.004 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_SALARY	YEARS_OF_EXPERIENCE
1	1002	John	Doe	66000	22
2	1003	Jane	Smith	70000	20
3	1004	Michael	Johnson	55000	21
4	1006	William	Taylor	65000	20
5	1007	Olivia	Clark	50000	22
6	1009	Sophia	Davis	55000	21

Rahul

Delete orders where quantity ordered for line item is greater than or equal to 5

delete from order_line where qty>=5;

rollback;

```
select * from order_line;

commit;

delete from order_line where qty>=5;
```

Script Output | Query Result

SQL | All Rows Fetched: 9 in 0.003 seconds

	ORID	JID	...
1	7	10007	3
2	7	10008	2
3	6	10002	1
4	9	10004	1
5	6	10001	1
6	9	10001	1

erform "Go to Declaration"

Update order ID to 1 where Jersey ID is not 10001, 10002, 10003, or 10004
update order_line set orid = 1 where jid not in (10001,10002,10003,10004);

```
select * from order_line;

update order_line set orid = 1 where jid not in (10001,10002,10003,10004);
```

Script Output | Query Result

SQL | All Rows Fetched: 19 in 0.004 seconds

	ORID	JID	QTY
1	1	10008	6
2	1	10008	2
3	1	10007	8
4	1	10007	10
5	1	10007	3
6	1	10007	6

Update order_fulfillment_date ahead for 10 days from order_placement date
update orders set order_fulfillment_date =order_placement_date + 10;
commit;

Welcome Page | rbhogal

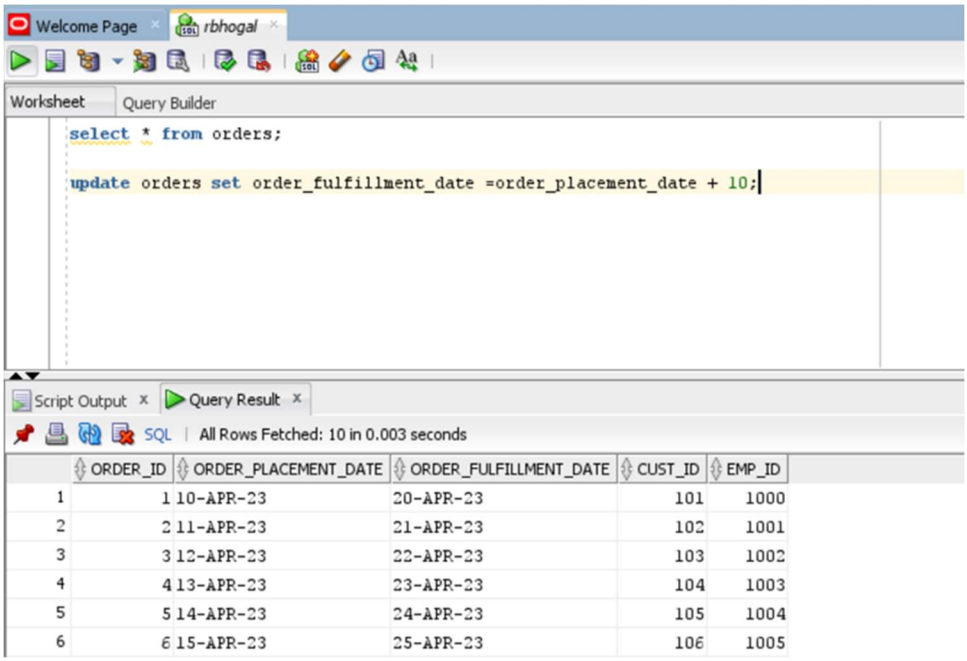
Worksheet | Query Builder

```
select * from orders;

update orders set order_fulfillment_date =order_placement_date + 10;
```

Query Result

SQL | All Rows Fetched: 10 in 0.035 seconds

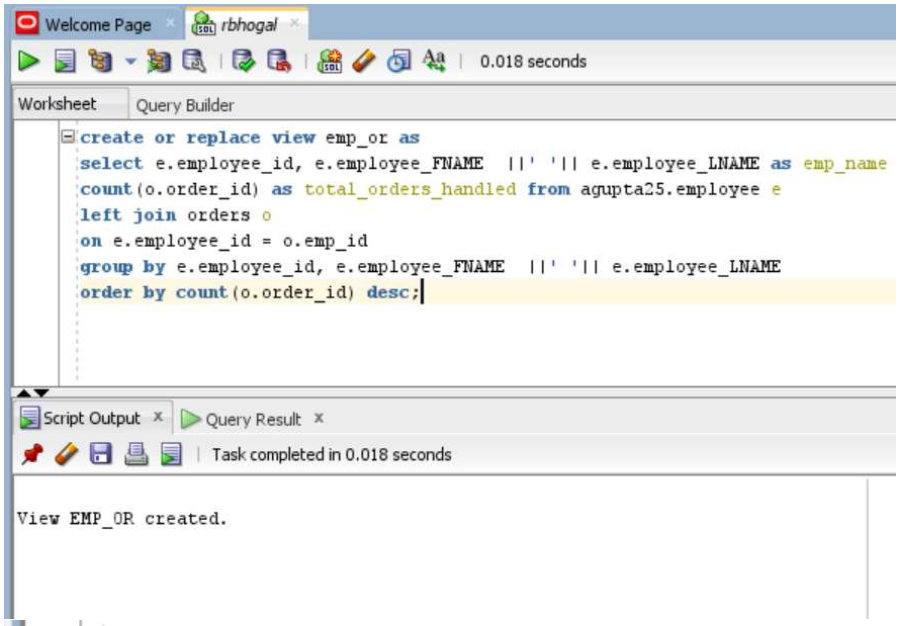
	ORDER_ID	ORDER_PLACEMENT_DATE	ORDER_FULFILLMENT_DATE	CUST_ID	EMP_ID
3	3	12-APR-23	17-APR-23	103	1002
4	4	13-APR-23	18-APR-23	104	1003
5	5	14-APR-23	19-APR-23	105	1004
6	6	15-APR-23	20-APR-23	106	1005
7	7	16-APR-23	21-APR-23	107	1006
8	8	17-APR-23	22-APR-23	108	1007
9	9	18-APR-23	23-APR-23	109	1008

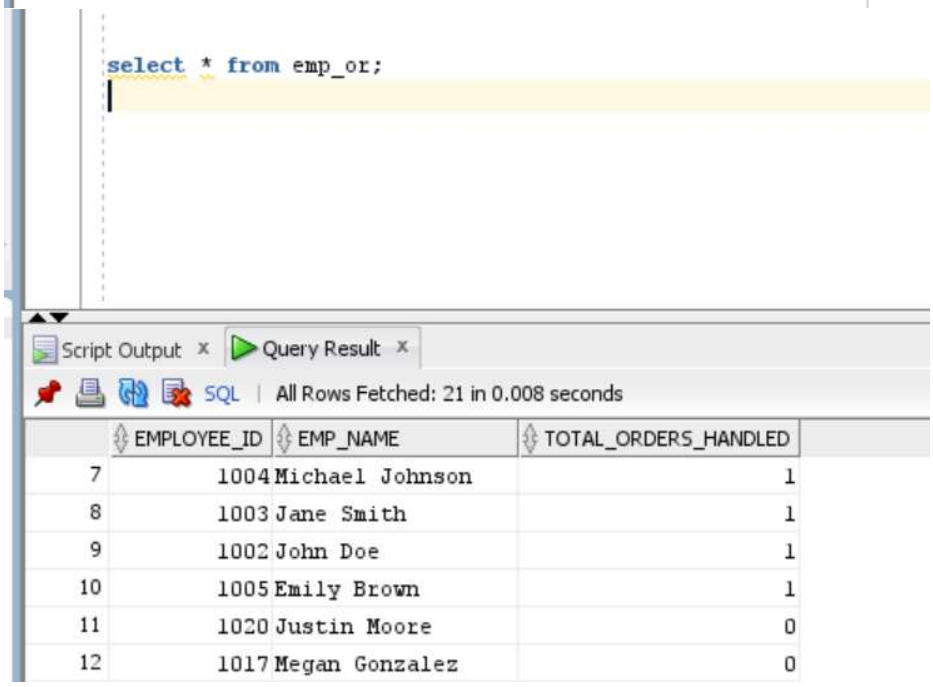


	ORDER_ID	ORDER_PLACEMENT_DATE	ORDER_FULFILLMENT_DATE	CUST_ID	EMP_ID
1	1	10-APR-23	20-APR-23	101	1000
2	2	11-APR-23	21-APR-23	102	1001
3	3	12-APR-23	22-APR-23	103	1002
4	4	13-APR-23	23-APR-23	104	1003
5	5	14-APR-23	24-APR-23	105	1004
6	6	15-APR-23	25-APR-23	106	1005

Create supervisor view showing how many orders handled by each employee in a day

```
create or replace view emp_or as
select e.employee_id, e.employee_FNAME  ||' '|| e.employee_LNAME as emp_name ,
count(o.order_id) as total_orders_handled from agupta25.employee e
left join orders o
on e.employee_id = o.emp_id
group by e.employee_id, e.employee_FNAME  ||' '|| e.employee_LNAME
order by count(o.order_id) desc;
```





	EMPLOYEE_ID	EMP_NAME	TOTAL_ORDERS_HANDLED
7	1004	Michael Johnson	1
8	1003	Jane Smith	1
9	1002	John Doe	1
10	1005	Emily Brown	1
11	1020	Justin Moore	0
12	1017	Megan Gonzalez	0

SQL Test Statements

Crystal

SELECT statement that calls the method defined in jersey table

```
SELECT j.jersey_tname.full_team_name(), j.jersey_name, j.jersey_num
FROM jersey j;
```

	J.JERSEY_TNAME.FULL_TEAM_NAME()	JERSEY_NAME	JERSEY_NUM
1	Baltimore Ravens	Lil Peanut	0
2	Kansas City Chiefs	chickenwhopper	23
3	Los Angeles Chargers	King James	23
4	Los Angeles Rams	babymochi	99

```
select * from supervisor_view;
```

Script Output | Query Result

SQL | All Rows Fetched: 11 in 0.004 seconds

	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_SALARY	YEARS_OF_EXPERIENCE
1	1002	John	Doe	66000	22
2	1003	Jane	Smith	70000	20
3	1004	Michael	Johnson	55000	21
4	1006	William	Taylor	65000	20
5	1007	Olivia	Clark	50000	22
6	1009	Sophia	Davis	55000	21

List employee names who have not processed any order

```
SELECT employee_fname, employee_lname
FROM agupta25.employee
WHERE employee_id NOT IN
(SELECT emp_id
FROM orders);
```

	EMPLOYEE_FNAME	EMPLOYEE_LNAME
1	Rachel	Davis
2	Mia	Lopez
3	Jacob	Gonzalez
4	Megan	Gonzalez
5	Isabella	Martinez
6	Brandon	Wilson
7	Benjamin	Rodriguez
8	Ava	Perez
9	Justin	Moore
10	Brian	Rodriguez
11	Daniel	Hernandez

List all the jerseys that have 'whopper' in the name

```
SELECT jersey_name, jersey_price
FROM jersey
WHERE jersey_name LIKE '%whopper';
```

	JERSEY_NAME	JERSEY_PRICE
1	chickenwhopper	169.99

Retrieve the names of customers who are in California and Florida

```
SELECT c.name.fname AS Customer_Fname, c.name.lname AS Customer_Lname,
c.customer_state
FROM agupta25.customer1 c
WHERE customer_state in ('CA', 'FL');
```

	CUSTOMER_FNAME	CUSTOMER_LNAME	CUSTOMER_STATE
1	Ash	Gupta	CA
2	John	Doe	CA
3	Samantha	Williams	FL
4	Karen	Davis	CA

Display the total cost for each order that has been placed

```
SELECT orid AS Order_ID, SUM(qty * jersey_price) AS Total_Cost
FROM rbhogal.order_line
JOIN jersey
ON order_line.jid = jersey.jersey_id
GROUP BY orid
ORDER BY orid;
```

	ORDER_ID	TOTAL_COST
1	1	1599.88
2	3	1000.923
3	4	3607.736
4	5	3090.769
5	6	339.98
6	7	2083.847
7	8	923.934
8	9	339.98
9	10	169.99

Retrieve the average, highest, and lowest price of each jersey type

```
SELECT jersey_type, MAX(jersey_price) AS Highest_Price, MIN(jersey_price) AS
Lowest_price, AVG(jersey_price) AS Average_Price
FROM jersey
GROUP BY jersey_type;
```

	JERSEY_TYPE	HIGHEST_PRICE	LOWEST_PRICE	AVERAGE_PRICE
1	M	169.99	169.99	169.99
2	P	153.989	142.989	148.489

Display the order IDs and the number of jersey types placed for each order

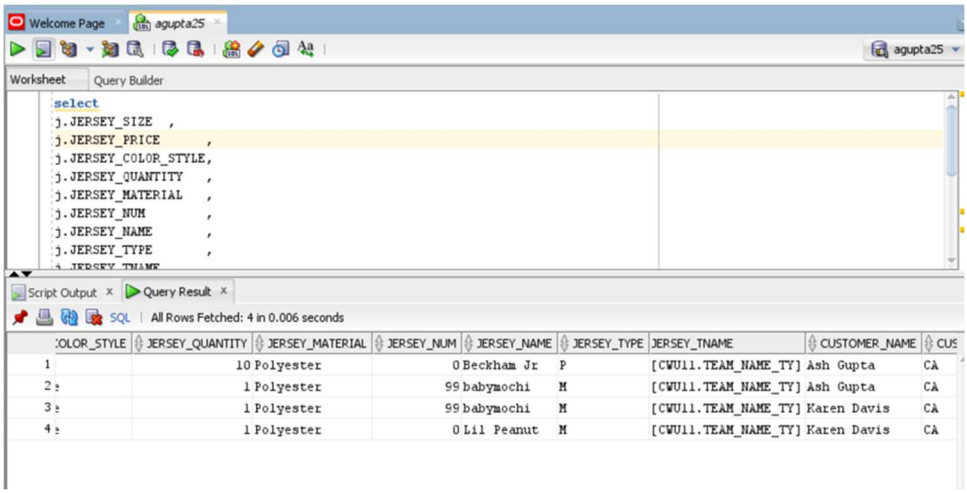
```
SELECT orid AS ORDER_ID, COUNT(orid) AS COUNT
FROM rbhogal.order_line
GROUP BY orid
ORDER BY orid;
```

	ORDER_ID	COUNT
1	1	2
2	3	1
3	4	3
4	5	3
5	6	2
6	7	4
7	8	1
8	9	2
9	10	1

Aishwarya

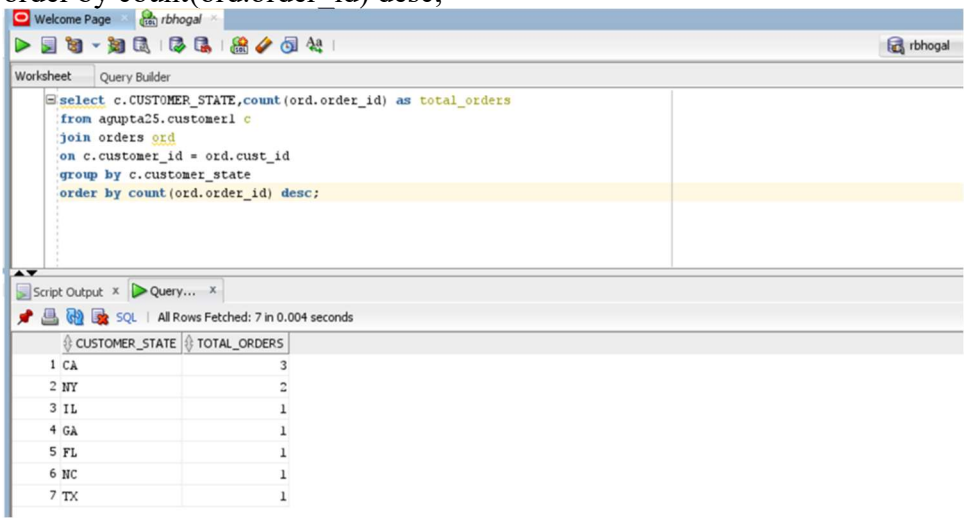
What jerseys did customers from California order?

```
select
j.JERSEY_SIZE  ,
j.JERSEY_PRICE     ,
j.JERSEY_COLOR_STYLE,
j.JERSEY_QUANTITY   ,
j.JERSEY_MATERIAL   ,
j.JERSEY_NUM       ,
j.JERSEY_NAME      ,
j.JERSEY_TYPE      ,
j.JERSEY_TNAME  ,
c.name.fname  ||' '|| c.name.lname as customer_name ,
c.CUSTOMER_STATE
from cwu11.jersey j
join rbhogal.order_line ol
on j.jersey_id = ol.jid
join rbhogal.orders ord
on ol.orid = ord.order_id
join customer1 c
on ord.cust_id = c.customer_id
where c.customer_state ='CA';
```



Rahul
How many orders did customers from each state place?

```
select c.CUSTOMER_STATE,count(ord.order_id) as total_orders
from agupta25.customer1 c
join orders ord
on c.customer_id = ord.cust_id
group by c.customer_state
order by count(ord.order_id) desc;
```



	CUSTOMER_STATE	TOTAL_ORDERS
1	CA	3
2	NY	2
3	IL	1
4	GA	1
5	FL	1
6	NC	1
7	TX	1

PL/SQL Statements

Crystal

Procedure with in mode parameter to increase jersey price by a given jersey type

```
CREATE OR REPLACE PROCEDURE raise_price
(v_id in jersey.jersey_type%type)
IS
BEGIN
   UPDATE jersey
   SET jersey_price = jersey_price * 1.1
   WHERE jersey_type = 'P';
END raise_price;
```

```
Procedure RAISE_PRICE compiled
```

```
EXECUTE raise_price('P');
```

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

Before SELCT statement:

	JERSEY_...	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID	JERSEY_PRICE
1	S	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEAM_NAME_TY]	10001	169.99
2	S	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEAM_NAME_TY]	10002	169.99
3	L	Alternate	1	Polyester	23	King James	M	[CWU11.TEAM_NAME_TY]	10003	169.99
4	M	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEAM_NAME_TY]	10004	169.99
5	S	Team	10	Polyester	15	Mahomes	P	[CWU11.TEAM_NAME_TY]	10006	129.99
6	M	Primary	10	Polyester	9	Burrow	P	[CWU11.TEAM_NAME_TY]	10007	139.99
7	XL	Primary	10	Polyester	0	Hurts	P	[CWU11.TEAM_NAME_TY]	10008	139.99
8	S	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEAM_NAME_TY]	10005	129.99

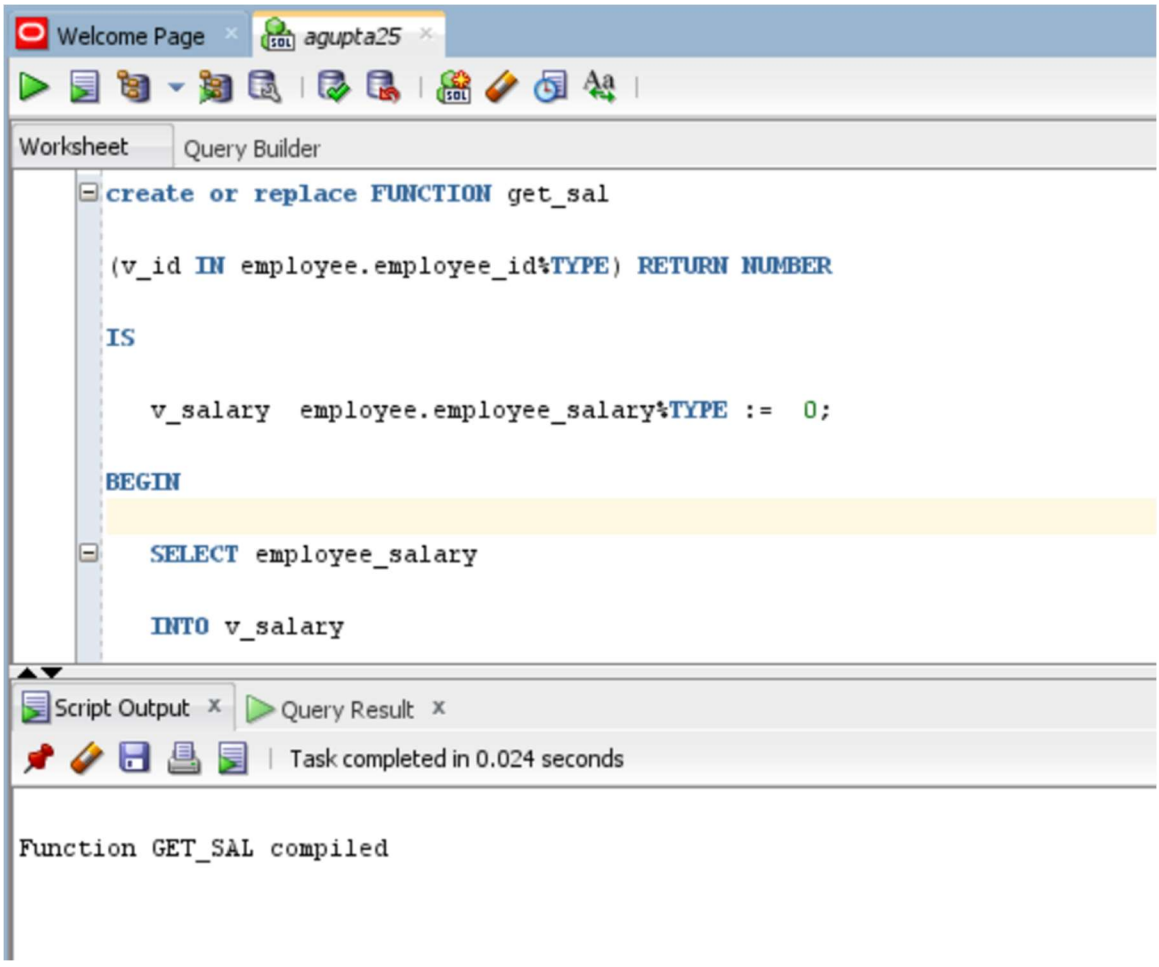
After SELECT statement:

SELECT * FROM jersey;

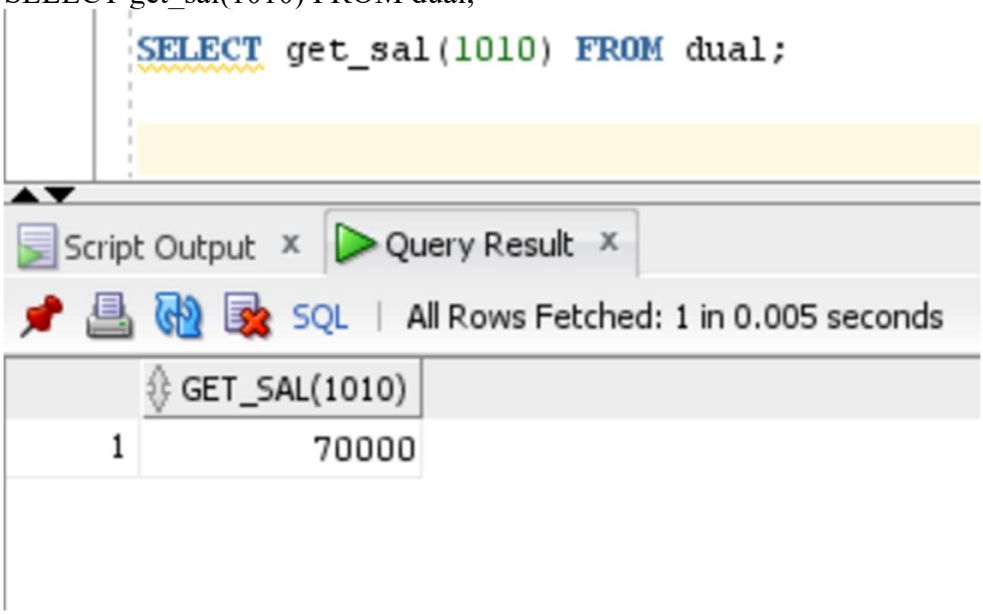
	JERSEY_SIZE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_NAME	JERSEY_TYPE	JERSEY_TNAME	JERSEY_ID	JERSEY_PRICE
1	S	Alternate	1	Polyester	0	Lil Peanut	M	[CWU11.TEAM_NAME_TY]	10001	169.99
2	S	Alternate	1	Polyester	23	chickenwhopper	M	[CWU11.TEAM_NAME_TY]	10002	169.99
3	L	Alternate	1	Polyester	23	King James	M	[CWU11.TEAM_NAME_TY]	10003	169.99
4	M	Alternate	1	Polyester	99	babymochi	M	[CWU11.TEAM_NAME_TY]	10004	169.99
5	S	Team	10	Polyester	15	Mahomes	P	[CWU11.TEAM_NAME_TY]	10006	142.989
6	M	Primary	10	Polyester	9	Burrow	P	[CWU11.TEAM_NAME_TY]	10007	153.989
7	XL	Primary	10	Polyester	0	Hurts	P	[CWU11.TEAM_NAME_TY]	10008	153.989
8	S	Team	10	Polyester	0	Beckham Jr	P	[CWU11.TEAM_NAME_TY]	10005	142.989

Aishwarya
Get salary function

```
create or replace FUNCTION get_sal
(v_id IN employee.employee_id%TYPE) RETURN NUMBER
IS
  v_salary  employee.employee_salary%TYPE :=  0;
BEGIN
  SELECT employee_salary
  INTO v_salary
  FROM employee
  WHERE employee_id =v_id;
  RETURN (v_salary);
END get_sal;
  Drop function get_sal;
```

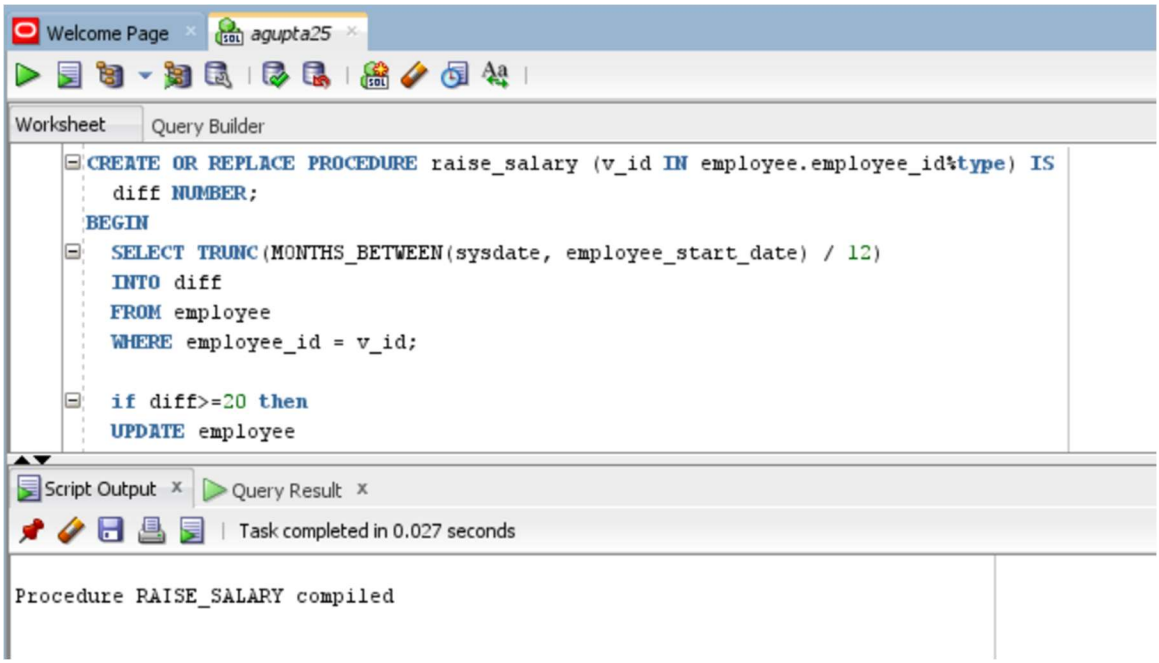


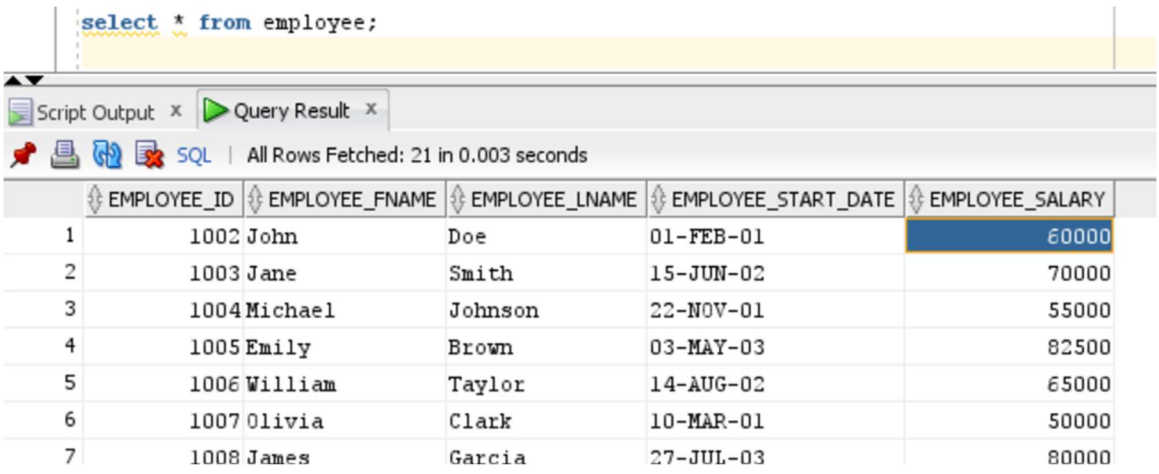
SELECT get_sal(1010) FROM dual;



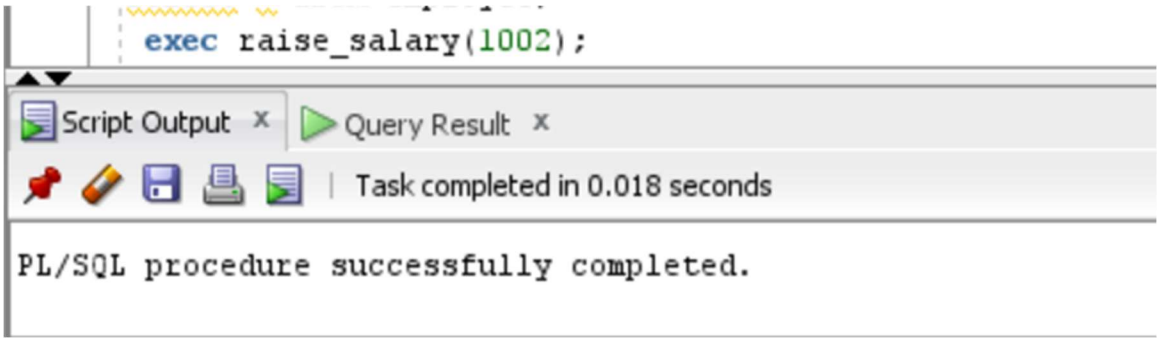
If employees' years of services >= 20, execute the raise salary procedure

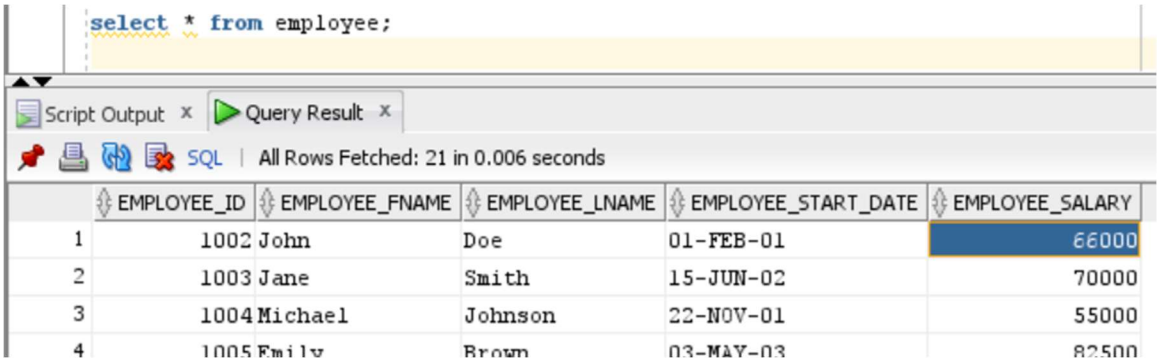
```
CREATE OR REPLACE PROCEDURE raise_salary (v_id IN
employee.employee_id%type) IS
  diff NUMBER;
BEGIN
  SELECT TRUNC(MONTHS_BETWEEN(sysdate, employee_start_date) / 12)
  INTO diff
  FROM employee
  WHERE employee_id = v_id;
  if diff>=20 then
  UPDATE employee
  SET employee_salary = employee_salary * 1.10
  WHERE employee_id = v_id;
  end if;
END raise_salary;

drop procedure raise_salary;
   exec raise_salary(1002);
```





	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SALARY
1	1002	John	Doe	01-FEB-01	60000
2	1003	Jane	Smith	15-JUN-02	70000
3	1004	Michael	Johnson	22-NOV-01	55000
4	1005	Emily	Brown	03-MAY-03	82500
5	1006	William	Taylor	14-AUG-02	65000
6	1007	Olivia	Clark	10-MAR-01	50000
7	1008	James	Garcia	27-JUL-03	80000



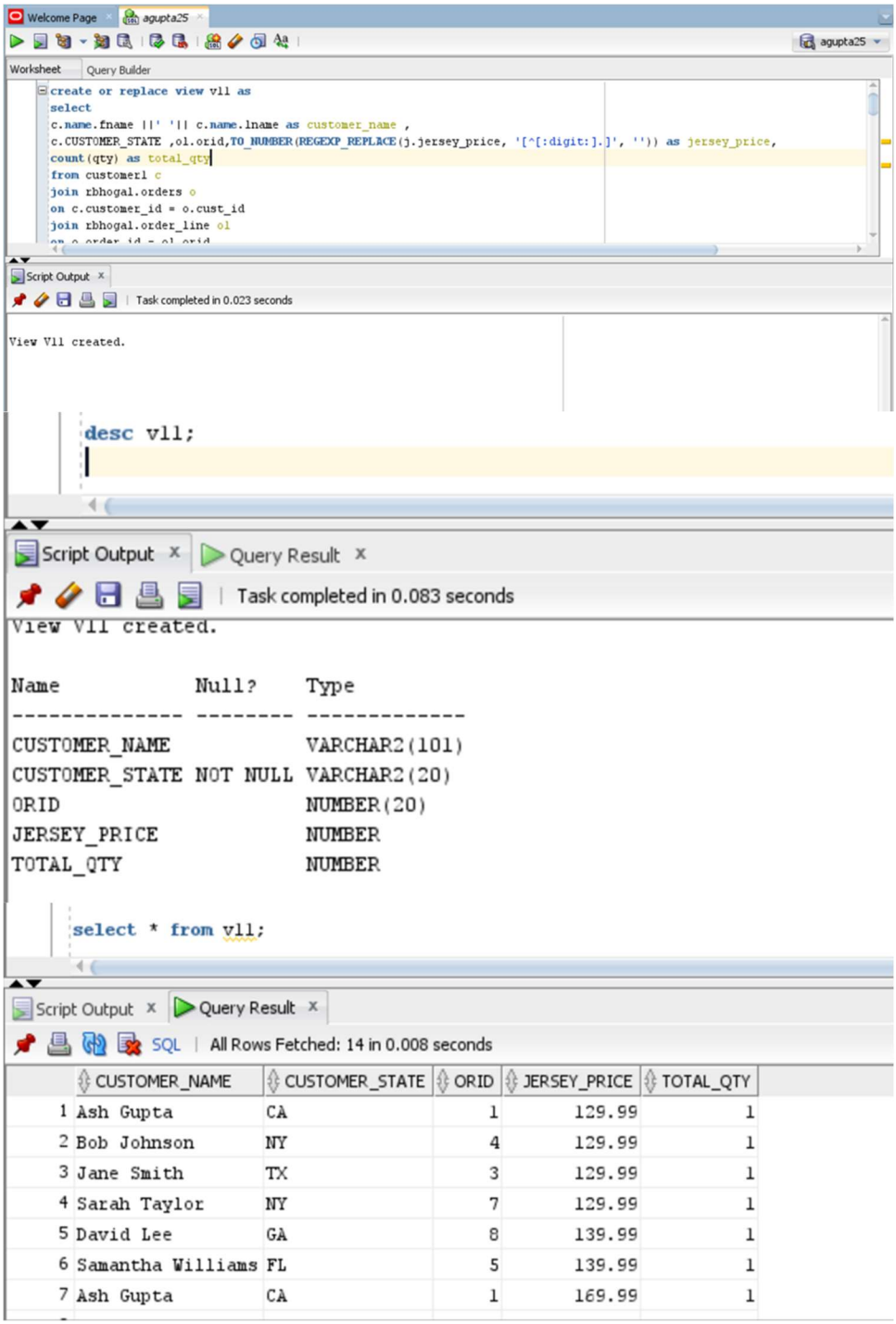


	EMPLOYEE_ID	EMPLOYEE_FNAME	EMPLOYEE_LNAME	EMPLOYEE_START_DATE	EMPLOYEE_SALARY
1	1002	John	Doe	01-FEB-01	66000
2	1003	Jane	Smith	15-JUN-02	70000
3	1004	Michael	Johnson	22-NOV-01	55000
4	1005	Emily	Brown	03-MAY-03	82500

Procedure to get discount 30% off if purchase of order >= $300

Step1:

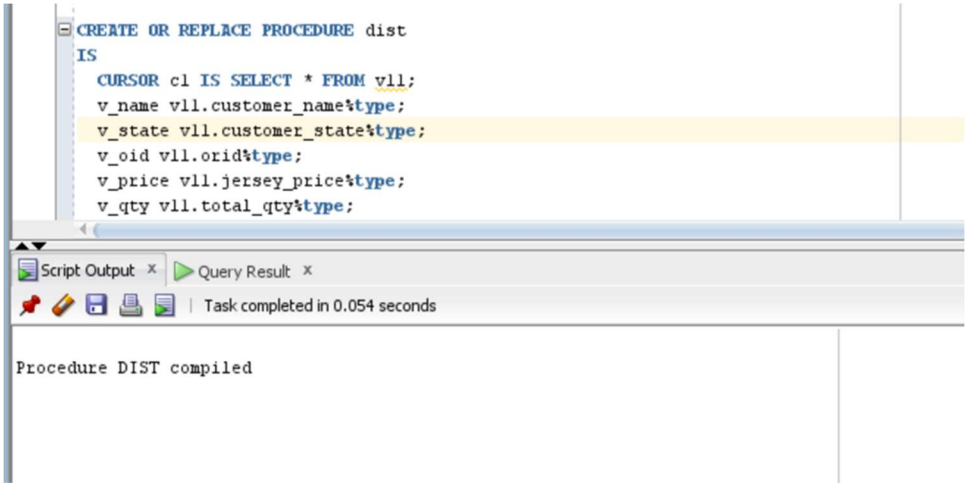
```
create or replace view v11 as
select
c.name.fname ||' '|| c.name.lname as customer_name ,
c.CUSTOMER_STATE ,ol.orid,TO_NUMBER(REGEXP_REPLACE(j.jersey_price,
'[^[:digit:].]', '')) as jersey_price,
count(qty) as total_qty
from customer1 c
join rbhogal.orders o
on c.customer_id = o.cust_id
join rbhogal.order_line ol
on o.order_id = ol.orid
join cwu11.jersey j
on ol.jid = j.jersey_id
group by c.name.fname  ||' '|| c.name.lname ,
c.CUSTOMER_STATE ,ol.orid,TO_NUMBER(REGEXP_REPLACE(jersey_price,
'[^[:digit:].]', ''))
order by count(qty),jersey_price, c.name.fname  ||' '|| c.name.lname ,
c.CUSTOMER_STATE ,ol.orid desc;

drop view v11;


select * from v11;

desc v11;
```



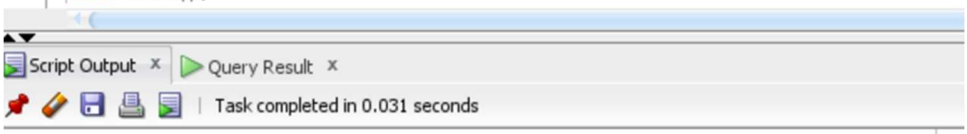
Step2: Used the above view in procedure

```
CREATE OR REPLACE PROCEDURE dist
IS
  CURSOR c1 IS SELECT * FROM v11;
  v_name v11.customer_name%type;
  v_state v11.customer_state%type;
```

```
  v_oid v11.orid%type;
  v_price v11.jersey_price%type;
  v_qty v11.total_qty%type;
  v_discount decimal(20,2);
  v_newprice decimal(20, 2); -- added scale to decimal datatype
BEGIN
  OPEN c1;
  LOOP
    FETCH c1 INTO v_name, v_state, v_oid, v_price, v_qty;
    EXIT WHEN c1%NOTFOUND;

    -- Do something with the fetched values, for example:
    IF to_number(v_price) * v_qty >= 300 THEN
      v_discount := 0.3;
      v_newprice := 0.7 * to_number(v_price) * v_qty;
    ELSE
      v_newprice := to_number(v_price) * v_qty;
    END IF;

    dbms_output.put_line('Customer Name: ' || v_name || ', State: ' || v_state || ', Order_id: '
|| v_oid || ', Final price: ' || v_newprice);
  END LOOP;
  CLOSE c1;
END;

SET SERVEROUTPUT ON;

EXEC dist();
```



```
EXEC dist();
```

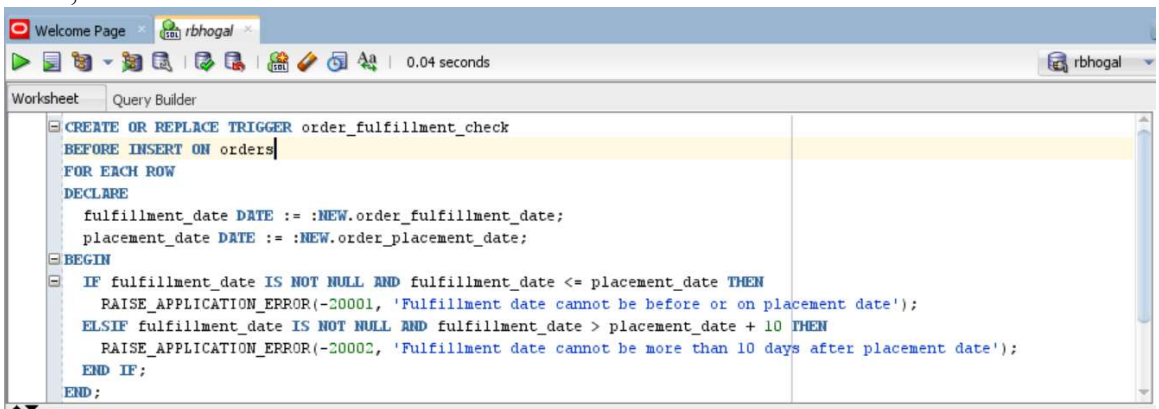


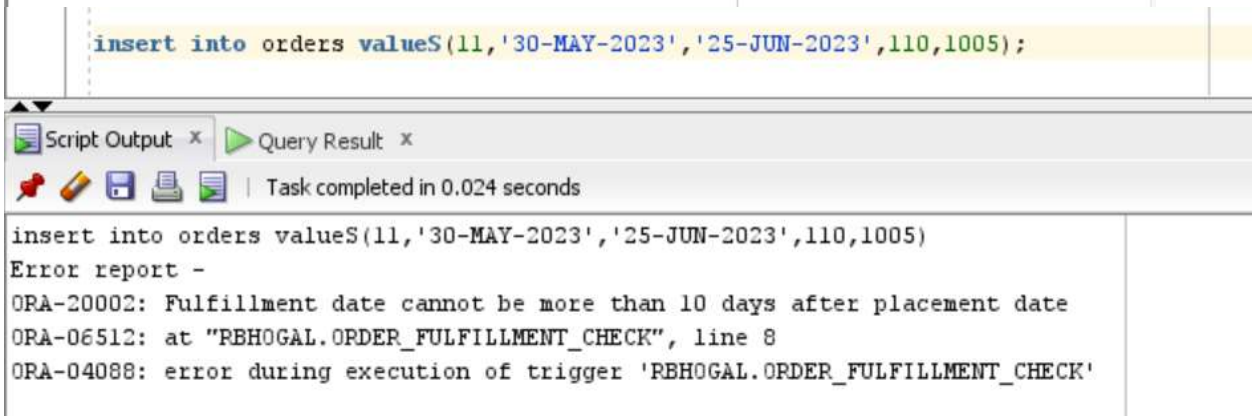

```
Procedure DIST compiled

Customer Name: Ash Gupta, State: CA, Order_id: 1, Final price: 129.99
Customer Name: Bob Johnson, State: NY, Order_id: 4, Final price: 129.99
Customer Name: Jane Smith, State: TX, Order_id: 3, Final price: 129.99
Customer Name: Sarah Taylor, State: NY, Order_id: 7, Final price: 129.99
Customer Name: David Lee, State: GA, Order_id: 8, Final price: 139.99
Customer Name: Samantha Williams, State: FL, Order_id: 5, Final price: 139.99
Customer Name: Ash Gupta, State: CA, Order_id: 1, Final price: 169.99
```

Rahul

Create order fulfillment trigger

```
CREATE OR REPLACE TRIGGER order_fulfillment_check
BEFORE INSERT ON orders
FOR EACH ROW
DECLARE
  fulfillment_date DATE := :NEW.order_fulfillment_date;
  placement_date DATE := :NEW.order_placement_date;
BEGIN
  IF fulfillment_date IS NOT NULL AND fulfillment_date <= placement_date THEN
    RAISE_APPLICATION_ERROR(-20001, 'Fulfillment date cannot be before or on
placement date');
  ELSIF fulfillment_date IS NOT NULL AND fulfillment_date > placement_date + 10
THEN
    RAISE_APPLICATION_ERROR(-20002, 'Fulfillment date cannot be more than 10
days after placement date');
  END IF;
END;
```





```
insert into orders valueS(11,'30-MAY-2023','25-MAY-2023',110,1005);
```

Script Output | Query Result

Task completed in 0.017 seconds

```
insert into orders valueS(11,'30-MAY-2023','25-MAY-2023',110,1005)
Error report -
ORA-20001: Fulfillment date cannot be before or on placement date
ORA-06512: at "RBHOGAL.ORDER_FULFILLMENT_CHECK", line 6
ORA-04088: error during execution of trigger 'RBHOGAL.ORDER_FULFILLMENT_CHECK'
```

```
drop trigger order_fulfillment_check;
```

Script Output | Query Result

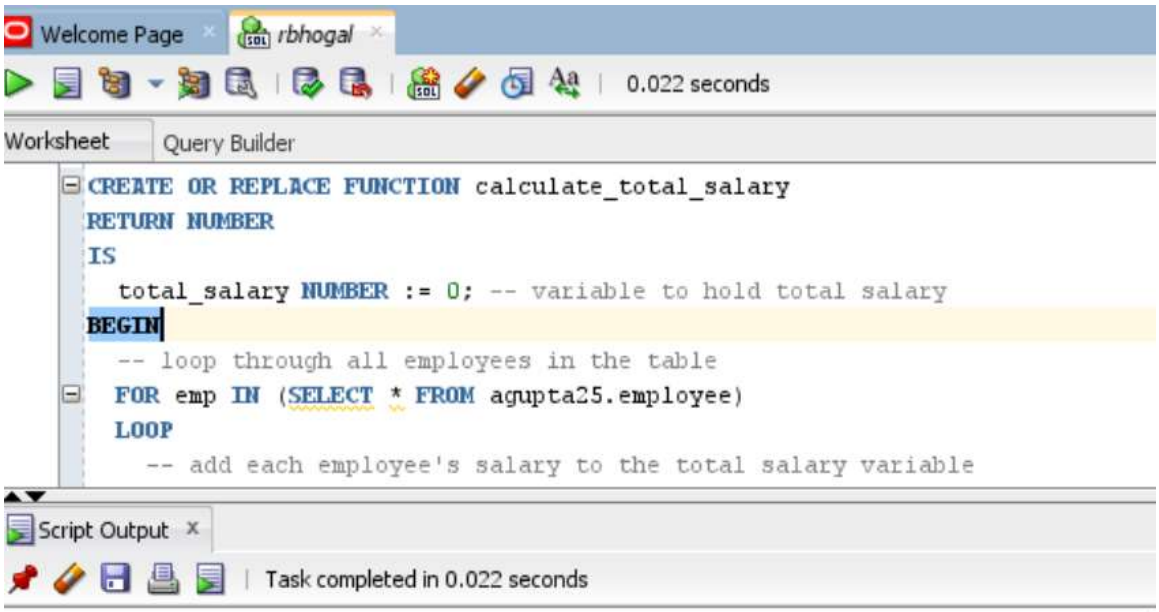
Task completed in 0.025 seconds

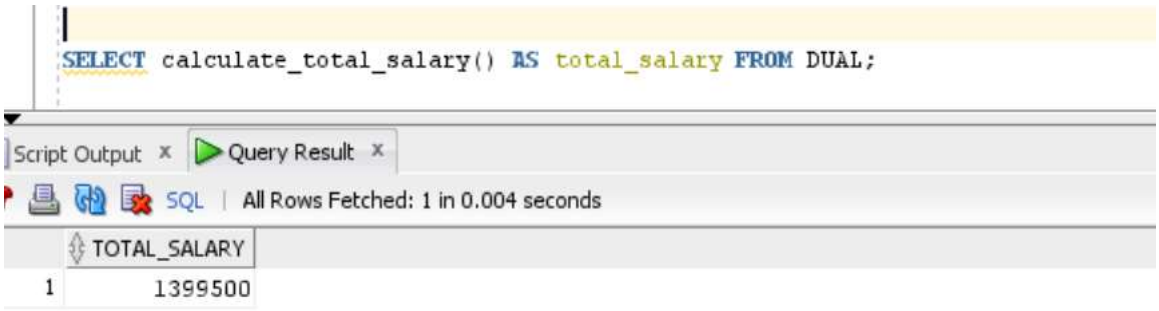
```
ORA-06512: at "RBHOGAL.ORDER_FULFILLMENT_CHECK", line 6
ORA-04088: error during execution of trigger 'RBHOGAL.ORDER_FULFILLMENT_CHECK'


Trigger ORDER_FULFILLMENT_CHECK dropped.
```

Create function to calculate total salary

```
CREATE OR REPLACE FUNCTION calculate_total_salary
RETURN NUMBER
IS
  total_salary NUMBER := 0; -- variable to hold total salary
BEGIN
  -- loop through all employees in the table
  FOR emp IN (SELECT * FROM agupta25.employee)
  LOOP
    -- add each employee's salary to the total salary variable
    total_salary := total_salary + emp.employee_salary;
  END LOOP;

  -- return the total salary
  RETURN total_salary;

END;
```

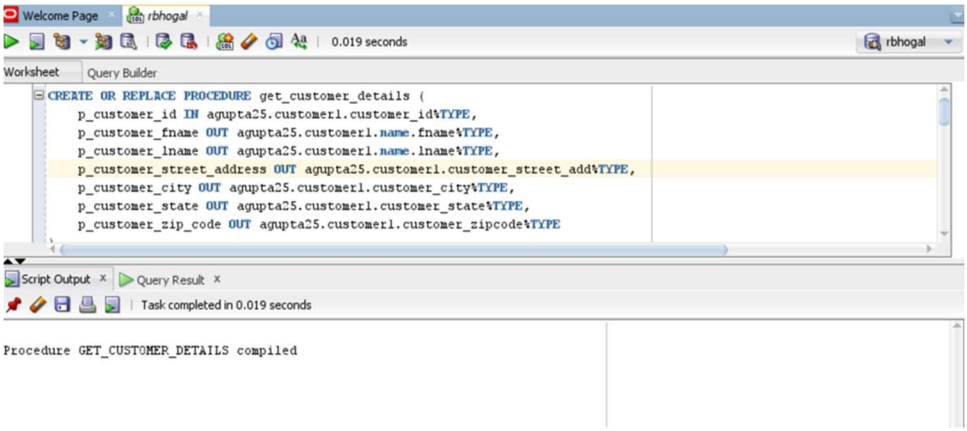
```
SELECT calculate_total_salary() AS total_salary FROM DUAL;
```





Procedure to view customer details

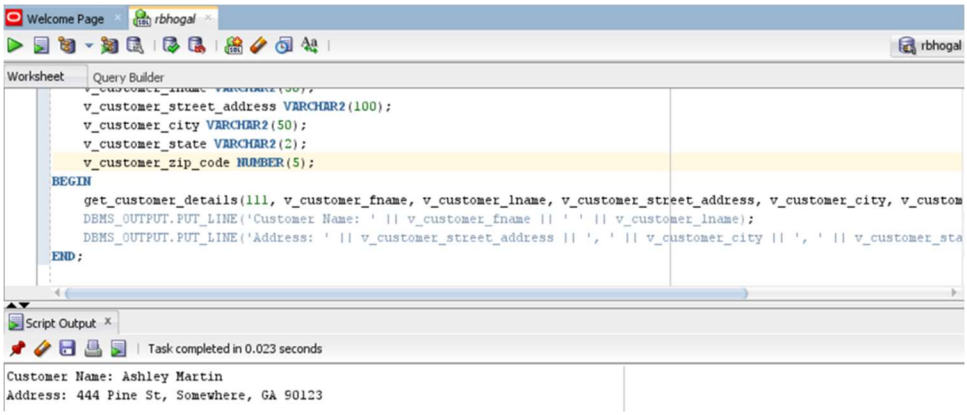
```
CREATE OR REPLACE PROCEDURE get_customer_details (
   p_customer_id IN agupta25.customer1.customer_id%TYPE,
   p_customer_fname OUT agupta25.customer1.name.fname%TYPE,
   p_customer_lname OUT agupta25.customer1.name.lname%TYPE,
   p_customer_street_address OUT agupta25.customer1.customer_street_add%TYPE,
   p_customer_city OUT agupta25.customer1.customer_city%TYPE,
   p_customer_state OUT agupta25.customer1.customer_state%TYPE,
   p_customer_zip_code OUT agupta25.customer1.customer_zipcode%TYPE
)
IS
BEGIN
  SELECT c.name.fname,c.name.lname, c.customer_street_add, c.customer_city,
c.customer_state, c.customer_zipcode
INTO p_customer_fname, p_customer_lname, p_customer_street_address,
p_customer_city, p_customer_state, p_customer_zip_code
FROM agupta25.customer1 c
WHERE c.customer_id = p_customer_id;

END;

drop procedure get_customer_details;
```



To obtain result of above created procedure, create another PLSQL Block

set SERVEROUTPUT ON; **--Use this command to see the results of DBMS_OUTPUT.PUT_LINE**

```
DECLARE
   v_customer_fname VARCHAR2(50);
   v_customer_lname VARCHAR2(50);
   v_customer_street_address VARCHAR2(100);
   v_customer_city VARCHAR2(50);
   v_customer_state VARCHAR2(2);
   v_customer_zip_code NUMBER(5);
BEGIN
   get_customer_details(111, v_customer_fname, v_customer_lname, v_customer_street_address, v_customer_city, v_customer_state, v_customer_zip_code);
   DBMS_OUTPUT.PUT_LINE('Customer Name: ' || v_customer_fname || ' ' || v_customer_lname);
   DBMS_OUTPUT.PUT_LINE('Address: ' || v_customer_street_address || ', ' || v_customer_city || ', ' || v_customer_state || ' ' || v_customer_zip_code);
END;
```


Welcome Page
rbhogal
rbhogal
Worksheet
Query Builder
v_customer_street_address VARCHAR2(100);
v_customer_city VARCHAR2(50);
v_customer_state VARCHAR2(2);
v_customer_zip_code NUMBER(5);
BEGIN
get_customer_details(111, v_customer_fname, v_customer_lname, v_customer_street_address, v_customer_city, v_custom
DBMS_OUTPUT.PUT_LINE('Customer Name: ' || v_customer_fname || ' ' || v_customer_lname);
DBMS_OUTPUT.PUT_LINE('Address: ' || v_customer_street_address || ', ' || v_customer_city || ', ' || v_customer_sta
END;
Script Output
Task completed in 0.023 seconds
Customer Name: Ashley Martin
Address: 444 Pine St, Somewhere, GA 90123

More Object Types - Bonus

Crystal

Object type inheritance

Besides the object data types we created above, we also defined hierarchies of object data types. Specifically, we built the purchased_ty and manufactured_ty subtypes in the hierarchy of the jersey_ty object data type.

```
CREATE TYPE jersey_ty AS OBJECT(
jersey_size         VARCHAR(10),
jersey_price        VARCHAR(10),
jersey_color_style  VARCHAR(10),
jersey_quantity     NUMBER,
jersey_material     VARCHAR(10),
jersey_num          NUMBER(2))
NOT FINAL NOT INSTANTIABLE;


CREATE TYPE manufactured_ty UNDER jersey_ty (jersey_status VARCHAR(10));

CREATE TYPE purchased_ty UNDER jersey_ty (jersey_assurance VARCHAR(10));

CREATE TABLE Manufactured of manufactured_ty;

ALTER TYPE jersey_ty ADD ATTRIBUTE(jersey_id NUMBER)
CASCADE NOT INCLUDING TABLE DATA;

ALTER TYPE jersey_ty ADD ATTRIBUTE(jersey_name varchar(20))
CASCADE NOT INCLUDING TABLE DATA;

INSERT INTO Manufactured VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 00,
'Production', 10001, 'Lil Peanut');
INSERT INTO Manufactured VALUES('S', '$169.99', 'Alternate', 1, 'Polyester', 23,
'Delivery', 10002, 'chickenwhopper');
INSERT INTO Manufactured VALUES('L', '$169.99', 'Alternate', 1, 'Polyester', 23,
'Delivery', 10003, 'King James');
INSERT INTO Manufactured VALUES('M', '$169.99', 'Alternate', 1, 'Polyester', 99,
'Processed', 10004, 'babymochi');

SELECT * FROM Manufactured;
```

	JERSEY_SIZE	JERSEY_PRICE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_STATUS	JERSEY_ID	JERSEY_NAME
1	S	$169.99	Alternate	1	Polyester	0	Production	10001	Lil Peanut
2	S	$169.99	Alternate	1	Polyester	23	Delivery	10002	chickenwhopper
3	L	$169.99	Alternate	1	Polyester	23	Delivery	10003	King James
4	M	$169.99	Alternate	1	Polyester	99	Processed	10004	babymochi

```
CREATE TABLE Purchased of purchased_ty;

INSERT INTO Purchased VALUES('S', '$129.99', 'Team', 1, 'Polyester', 00, 'Free',
10005, 'Beckham Jr');
INSERT INTO Purchased VALUES('S', '$149.99', 'Team', 1, 'Polyester', 15, 'Free',
10006, 'Mahomes');
INSERT INTO Purchased VALUES('M', '$129.99', 'Primary', 1, 'Polyester', 9, 'Free',
10007, 'Burrow');
INSERT INTO Purchased VALUES('XL, '$129.99', 'Primary', 1, 'Polyester', 0, 'Free',
10008, 'Hurts);

SELECT * FROM Purchased;
```

	JERSEY_SIZE	JERSEY_PRICE	JERSEY_COLOR_STYLE	JERSEY_QUANTITY	JERSEY_MATERIAL	JERSEY_NUM	JERSEY_ID	JERSEY_NAME	JERSEY_ASSURANCE
1	S	$129.99	Team	10	Polyester	0	10005	Beckham Jr	Free
2	S	$149.99	Team	10	Polyester	15	10006	Mahomes	Free
3	M	$129.99	Primary	10	Polyester	9	10007	Burrow	Free
4	XL	$129.99	Primary	10	Polyester	0	10008	Hurts	Free

```
CREATE OR REPLACE TYPE team_ty AS OBJECT(
f_tname VARCHAR(15),
l_tname VARCHAR(15));

ALTER TYPE team_ty
ADD MEMBER FUNCTION full_team_name RETUREN VARCHAR2 CASCADE;

CREATE OR REPLACE TYPE BODY team_ty AS
MEMBER FUNCTION full_team_name
RETURN VARCHAR2 IS
   BEGIN
      RETURN(f_tname || ' ' || l_tname);
   END full_team_name;
END;

ALTER TYPE jersey_ty ADD ATTRIBUTE(jersey_gender VARCHAR(10),
jersey_team team_ty)
CASCADE NOT INCLUDING TABLE DATA;

UPDATE Manufactured SET jersey_gender = "Women's"
AND jersey_team = team_ty('Kansas City', 'Chiefs')
WHERE jersey_id = 10006;
```

DESC jersey_ty;

Name	Null?	Type
JERSEY_SIZE		VARCHAR2(10)
JERSEY_PRICE		VARCHAR2(10)
JERSEY_COLOR_STYLE		VARCHAR2(10)
JERSEY_QUANTITY		NUMBER
JERSEY_MATERIAL		VARCHAR2(10)
JERSEY_NUM		NUMBER(2)
JERSEY_ID		NUMBER
JERSEY_NAME		VARCHAR2(20)
JERSEY_GENDER		VARCHAR2(10)
JERSEY_TEAM		CWU11.TEAM_TY()

DESC Manufactured;

Name	Null?	Type
JERSEY_SIZE		VARCHAR2(10)
JERSEY_PRICE		VARCHAR2(10)
JERSEY_COLOR_STYLE		VARCHAR2(10)
JERSEY_QUANTITY		NUMBER
JERSEY_MATERIAL		VARCHAR2(10)
JERSEY_NUM		NUMBER(2)
JERSEY_STATUS		VARCHAR2(10)
JERSEY_ID		NUMBER
JERSEY_NAME		VARCHAR2(20)
JERSEY_GENDER		VARCHAR2(10)
JERSEY_TEAM		TEAM_TY

DESC Purchased;

Name	Null?	Type
JERSEY_SIZE		VARCHAR2(10)
JERSEY_PRICE		VARCHAR2(10)
JERSEY_COLOR_STYLE		VARCHAR2(10)
JERSEY_QUANTITY		NUMBER
JERSEY_MATERIAL		VARCHAR2(10)
JERSEY_NUM		NUMBER(2)
JERSEY_ID		NUMBER
JERSEY_NAME		VARCHAR2(20)
JERSEY_ASSURANCE		VARCHAR2(10)
JERSEY_GENDER		VARCHAR2(10)
JERSEY_TEAM		TEAM_TY

SELECT * FROM Manufactured;

```
JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_STA  JERSEY_ID JERSEY_NAME          JERSEY_GEN
---------- ---------- ---------- --------------- ---------- ---------- ---------- ---------- -------------------- ----------
JERSEY_TEAM(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------------------------
S          $169.99    Alternate               1 Polyester           0 Production      10001 Lil Peanut           Women
TEAM_TY('San Francisco', 'Giants')

S          $169.99    Alternate               1 Polyester          23 Delivery        10002 chickenwhopper       Men
TEAM_TY('Kansas City', 'Chiefs')

L          $169.99    Alternate               1 Polyester          23 Delivery        10003 King James           Women
TEAM_TY('Los Angeles', 'Chargers')


JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM JERSEY_STA  JERSEY_ID JERSEY_NAME          JERSEY_GEN
---------- ---------- ---------- --------------- ---------- ---------- ---------- ---------- -------------------- ----------
JERSEY_TEAM(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------------------------
M          $169.99    Alternate               1 Polyester          99 Processed       10004 babymochi            Women
TEAM_TY('Los Angeles', 'Rams')
```

SELECT * FROM Purchased;

```
JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM  JERSEY_ID JERSEY_NAME          JERSEY_ASS JERSEY_GEN
---------- ---------- ---------- --------------- ---------- ---------- ---------- -------------------- ---------- ----------
JERSEY_TEAM(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------------------------
S          $129.99    Team                   10 Polyester           0      10005 Beckham Jr           Free       Women
TEAM_TY('Baltimore', 'Ravens')

S          $149.99    Team                   10 Polyester          15      10006 Mahomes              Free       Women
TEAM_TY('Kansas City', 'Chiefs')

M          $129.99    Primary                10 Polyester           9      10007 Burrow               Free       Men
TEAM_TY('Cincinnati', 'Bengals')


JERSEY_SIZ JERSEY_PRI JERSEY_COL JERSEY_QUANTITY JERSEY_MAT JERSEY_NUM  JERSEY_ID JERSEY_NAME          JERSEY_ASS JERSEY_GEN
---------- ---------- ---------- --------------- ---------- ---------- ---------- -------------------- ---------- ----------
JERSEY_TEAM(F_TNAME, L_TNAME)
--------------------------------------------------------------------------------------------------------------------------------
XL         $129.99    Primary                10 Polyester           0      10008 Hurts                Free       Women
TEAM_TY('Philadelphia', 'Eagles')
```

```
GRANT ALL on jersey_ty to rbhogal;
GRANT ALL on jersey_ty to agupta25;
GRANT ALL on Manufactured to rbhogal;
GRANT ALL on Manufactured to agupta25;
Grant succeeded.


Grant succeeded.


Grant succeeded.


Grant succeeded.
```