

# Online Vet Appointment System

## Table of Content -

Project Scenario (Intro)	3
Entity Relational Diagram (ER Diagram)	3
Data Dictionary	4
Create all the required tables	6
Insert data into tables	8
SQL Statements	13
Procedures	19
Functions	34
Web Application Screenshots and link	43
Members of Contribution	49

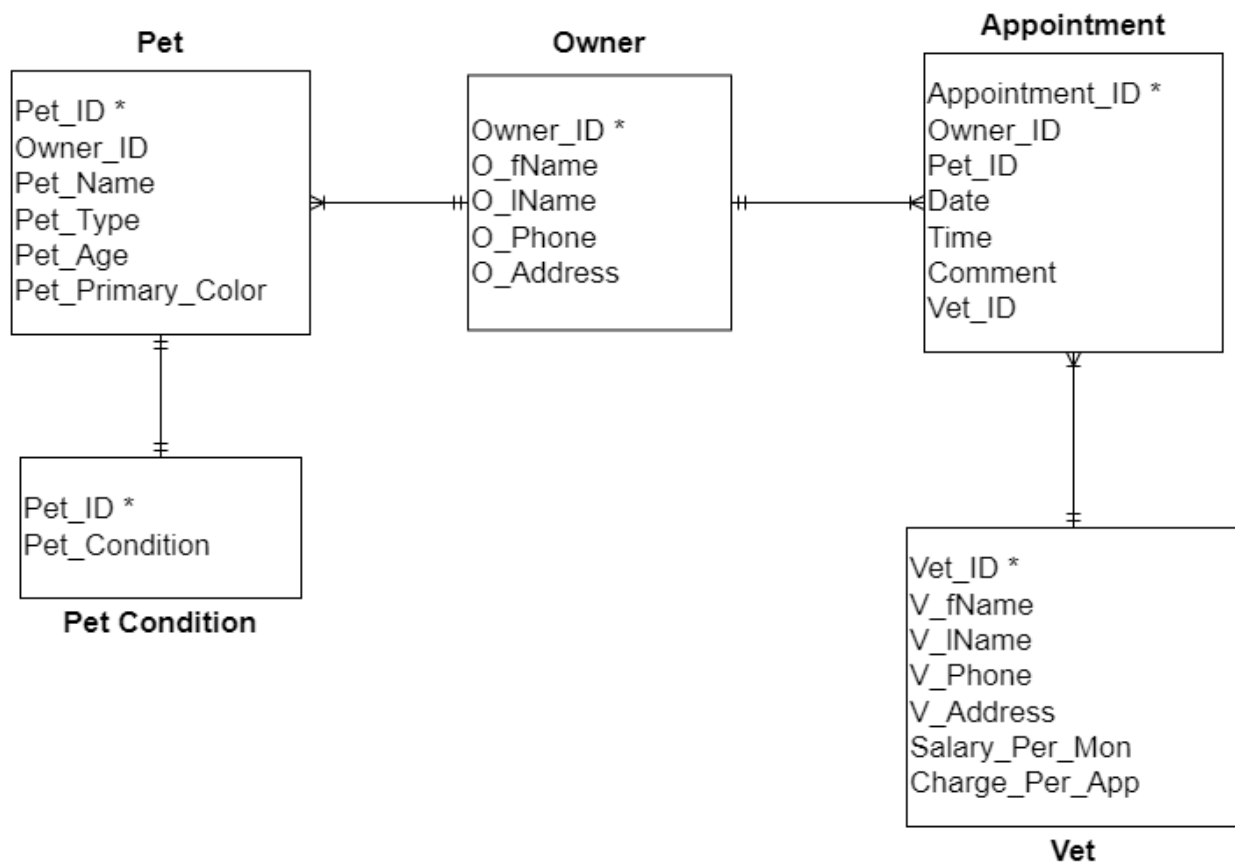
## Project Scenario (Intro)

Our plan is to develop a database system for a pet clinic to give online appointments. The clinic will have vets / staff, pet owners and their pets.

Moreover, we develop a database which can store vets/staff information, pet owners information, their pets information, and appointments information. We also create functionality to our database application so that it can perform complex tasks using functions and procedures. Moreover, our database application also can add, modify, or remove entries from the system.

Finally, we also focus on our developing database system/application and its blueprint to perform tasks with real life applications / websites.

## Entity Relational Diagram (ER Diagram)



## Data Dictionary

### Pet

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
Pet_ID	NUMBER	4	NO (PK)
Owner_ID	NUMBER	4	NO (FK)
Pet_Name	VARCHAR2	25	NO
Pet_Type	VARCHAR2	10	NO (CHECK)
Pet_Age	NUMBER	2	NO
Pet_Primary_Color	VARCHAR2	15	Yes

### Pet\_Condition

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
Pet_ID	NUMBER	4	NO (PK, FK)
Pet_Condition	VARCHAR2	50	YES

### Owner

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
Owner_ID	NUMBER	4	NO (PK)
O_fName	VARCHAR2	20	NO
O_lName	VARCHAR2	20	NO
O_Phone	VARCHAR2	12	UNIQUE
O_Address	VARCHAR2	50	NO

### Appointment

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
Appointment_ID	NUMBER	5	NO (PK)
Owner_ID	NUMBER	4	NO (FK)
Pet_ID	NUMBER	4	NO (FK)
Date	DATE		NO
Time	VARCHAR2	20	NO
Comment	VARCHAR2	40	Yes
Vet_ID	NUMBER	3	NO (FK)

### Vet

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
Vet_ID	NUMBER	3	NO (PK)
V_fName	NUMBER	20	NO
V_IName	VARCHAR2	20	NO
V_Phone	VARCHAR2	12	UNIQUE
V_Address	NUMBER	50	YES
Salary_Per_Mon	NUMBER	6	NO
Charge_Per_App	NUMBER	3	NO

## Create all the required tables

Drop Statement for All Table -

```
DROP TABLE Vet CASCADE CONSTRAINTS;  
DROP TABLE Owner CASCADE CONSTRAINTS;  
DROP TABLE Pet CASCADE CONSTRAINTS;  
DROP TABLE Pet_Condition CASCADE CONSTRAINTS;  
DROP TABLE Appointment CASCADE CONSTRAINTS;
```

Create **Vet** Table -

```
CREATE TABLE Vet (  
    Vet_ID NUMBER(3),  
    V_fName VARCHAR2(20) NOT NULL,  
    V_iName VARCHAR2(20) NOT NULL,  
    V_Phone VARCHAR2(12) UNIQUE,  
    V_Address VARCHAR2(50),  
    Salary_Per_Mon NUMBER(6) NOT NULL,  
    Charge_Per_App NUMBER(3) NOT NULL,  
    CONSTRAINT Vet_VetID_PK PRIMARY KEY (Vet_ID)  
);
```

Create **Owner** Table-

```
CREATE TABLE Owner (  
    Owner_ID NUMBER(4),  
    O_fName VARCHAR2(20) NOT NULL,  
    O_iName VARCHAR2(20) NOT NULL,  
    O_Phone VARCHAR2(12) UNIQUE,  
    O_Address VARCHAR2(50) NOT NULL,  
    CONSTRAINT Owner_OwnerID_PK PRIMARY KEY (Owner_ID)  
);
```

### Create **Pet** Table -

```
CREATE TABLE Pet (  
    Pet_ID NUMBER(4),  
    Owner_ID NUMBER(4),  
    Pet_Name VARCHAR2(25) NOT NULL,  
    Pet_Type VARCHAR2(10) CHECK(lower(Pet_Type) IN ('dog', 'cat', 'bird', 'rabbit', 'fish',  
'Cow', 'Goat', 'reptile')),  
    Pet_Age NUMBER(2),  
    Pet_Primary_Color VARCHAR2(15),  
    CONSTRAINT Pet_PetID_PK PRIMARY KEY (Pet_ID),  
    CONSTRAINT Pet_OwnerID_FK FOREIGN KEY (Owner_ID) REFERENCES Owner  
);
```

### Create **Pet\_Condition** Table -

```
CREATE TABLE Pet_Condition (  
    Pet_ID NUMBER(4),  
    Pet_Condition VARCHAR2(50),  
    CONSTRAINT PetCondition_PetID_PK PRIMARY KEY (Pet_ID),  
    CONSTRAINT PetCondition_PetID_FK FOREIGN KEY (Pet_ID) REFERENCES Pet  
);
```

### Create **Appointment** Table -

```
CREATE TABLE Appointment (  
    Appointment_ID NUMBER(5),  
    Owner_ID NUMBER(4),  
    Pet_ID NUMBER(4),  
    "Date" DATE NOT NULL,  
    "Time" VARCHAR2(20) NOT NULL,  
    "Comment" VARCHAR2(40),  
    Vet_ID NUMBER(3),  
    CONSTRAINT Appointment_AppID_PK PRIMARY KEY (Appointment_ID),  
    CONSTRAINT Appointment_OwnerID_FK FOREIGN KEY (Owner_ID) REFERENCES  
Owner,  
    CONSTRAINT Appointment_PetID_FK FOREIGN KEY (Pet_ID) REFERENCES Pet,  
    CONSTRAINT Appointment_VetID_PK FOREIGN KEY (Vet_ID) REFERENCES  
);
```

## Insert data into tables

Insert data into **Vet** table -

```
INSERT INTO Vet
VALUES(111, 'Tom', 'Harij', '01112223334', 'Gombak, Selangor', '6000', '80');

INSERT INTO Vet
VALUES(112, 'Mohammad', 'Imran', '01112663334', 'Bukit Damansara, Kuala Lumpur', '6500', '90');

INSERT INTO Vet
VALUES(113, 'Nurul', 'Hushni', '01166623334', 'Medan Idaman, Selangor', '8000', '100');

INSERT INTO Vet
VALUES(114, 'Yu', 'Yan', '01112223434', 'Bukit Bintang, Kuala Lumpur', '9000', '110');

INSERT INTO Vet
VALUES(115, 'Ying', 'Yue', '01122223434', 'Bukit Bintang, Kuala Lumpur', '9000', '120');

INSERT INTO Vet
VALUES(116, 'Safa', 'Kabir', '01212223434', 'Batu Caves, Selangor', '8500', '100');

INSERT INTO Vet
VALUES(117, 'Mohammad', 'Musfiq', '01212923434', 'Batu Caves, Selangor', '7000', '70');

INSERT INTO Vet
VALUES(118, 'Mohammad', 'Latif', '01912923434', 'Ampang, Kuala Lumpur', '7500', '85');

INSERT INTO Vet
VALUES(119, 'Mohammad', 'Razak', '01219923434', 'Batu Caves, Selangor', '10000', '150');

INSERT INTO Vet
VALUES(120, 'Mohammad', 'Mubarak', '01999923434', 'Batu Caves, Selangor', '10000', '140');
```

Insert data into **Owner** table -

```
INSERT INTO Owner
VALUES(1001, 'Esmee', 'Benitez', '01100000111', 'Gombak, Selangor');

INSERT INTO Owner
```

```
VALUES(1002, 'Freyja', 'Paul', '01100000112', 'Ampang, Selangor');

INSERT INTO Owner
VALUES(1003, 'Mohammad', 'Iman', '01100000113', 'Bandar Baru Bangi, Selangor');

INSERT INTO Owner
VALUES(1004, 'Mohammad', 'Asraf', '01100000114', 'Banting, Selangor');

INSERT INTO Owner
VALUES(1005, 'Mohammad', 'Arif', '01100001114', 'Gombak, Selangor');

INSERT INTO Owner
VALUES(1006, 'Danial', 'Hakim', '01100000115', 'Medan Idaman, Selangor');

INSERT INTO Owner
VALUES(1007, 'Raihan', 'Kabir', '01100000116', 'Batu Caves, Selangor');

INSERT INTO Owner
VALUES(1008, 'Mohammad', 'Hahmid', '01100001116', 'Gombak, Selangor');

INSERT INTO Owner
VALUES(1009, 'Mohammad', 'Rakib', '01100001117', 'Bukit Nanas, Kuala Lumpur');

INSERT INTO Owner
VALUES(1010, 'Abdur', 'Rahman', '01100000118', 'Gombak, Selangor');
```

Insert data into **Owner** table -

```
INSERT INTO Owner
VALUES(1001, 'Esmee', 'Benitez', '01100000111', 'Gombak, Selangor');

INSERT INTO Owner
VALUES(1002, 'Freyja', 'Paul', '01100000112', 'Ampang, Selangor');

INSERT INTO Owner
VALUES(1003, 'Mohammad', 'Iman', '01100000113', 'Bandar Baru Bangi, Selangor');

INSERT INTO Owner
VALUES(1004, 'Mohammad', 'Asraf', '01100000114', 'Banting, Selangor');

INSERT INTO Owner
VALUES(1005, 'Mohammad', 'Arif', '01100001114', 'Gombak, Selangor');

INSERT INTO Owner
```



```
VALUES(1006, 'Danial', 'Hakim', '01100000115', 'Medan Idaman, Selangor');

INSERT INTO Owner
VALUES(1007, 'Raihan', 'Kabir', '01100000116', 'Batu Caves, Selangor');

INSERT INTO Owner
VALUES(1008, 'Mohammad', 'Hahmid', '01100001116', 'Gombak, Selangor');

INSERT INTO Owner
VALUES(1009, 'Mohammad', 'Rakib', '01100001117', 'Bukit Nanas, Kuala Lumpur');

INSERT INTO Owner
VALUES(1010, 'Abdur', 'Rahman', '01100000118', 'Gombak, Selangor');

INSERT INTO Owner
VALUES(1011, 'Abdur', 'Salem', '01100010118', '');
```

Insert data into **Pet** table -

```
INSERT INTO Pet
VALUES(1101, 1001, 'Mini', 'cat', 2, 'white');

INSERT INTO Pet
VALUES(1102, 1002, 'Luna', 'cat', 5, 'brown');

INSERT INTO Pet
VALUES(1103, 1002, 'Teddy', 'dog', 7, 'orange');

INSERT INTO Pet
VALUES(1104, 1003, 'Bill', 'dog', 4, 'black');

INSERT INTO Pet
VALUES(1105, 1004, 'Bella', 'dog', 5, 'brown');

INSERT INTO Pet
VALUES(1106, 1004, 'Buddy', 'dog', 2, 'white');

INSERT INTO Pet
VALUES(1107, 1008, 'Milo', 'cat', 1, 'orange');

INSERT INTO Pet
VALUES(1108, 1005, 'Loki', 'cat', 2, 'black');

INSERT INTO Pet
```

```
VALUES(1109, 1006, 'Oliver', 'cat', 5, 'white');

INSERT INTO Pet
VALUES(1110, 1007, 'Bubba', 'bird', 1, 'green');

INSERT INTO Pet
VALUES(1111, 1009, 'Baldy', 'cat', 3, 'black');

INSERT INTO Pet
VALUES(1112, 1010, 'Putih', 'cat', 5, 'black');
```

Insert data into **Pet\_Condition** table -

```
INSERT INTO Pet_Condition
VALUES(1101, 'Good health');

INSERT INTO Pet_Condition
VALUES(1102, 'Not good');

INSERT INTO Pet_Condition
VALUES(1104, 'Ringworm');

INSERT INTO Pet_Condition
VALUES(1107, 'Psittacosis');

INSERT INTO Pet_Condition
VALUES(1112, 'Diabetes');

INSERT INTO Pet_Condition
VALUES(1111, 'Heartworm');
```

Insert data into **Appointment** table -

```
INSERT INTO Appointment
VALUES(10001, 1001, 1101, to_date('11/07/22', 'dd/mm/yy'), '11.00 am', null, 111);

INSERT INTO Appointment
VALUES(10002, 1002, 1102, to_date('14/07/22', 'dd/mm/yy'), '10.00 am', 'Rash on Skin', 111);
```

```
INSERT INTO Appointment
VALUES(10003, 1003, 1104, to_date('09/07/22', 'dd/mm/yy'), '12.00 pm', null, 112);

INSERT INTO Appointment
VALUES(10004, 1004, 1105, to_date('14/07/22', 'dd/mm/yy'), '05.00 pm', null, 114);

INSERT INTO Appointment
VALUES(10005, 1006, 1109, to_date('18/07/22', 'dd/mm/yy'), '10.00 am', 'Rash on Skin', 119);

INSERT INTO Appointment
VALUES(10006, 1010, 1112, to_date('20/07/22', 'dd/mm/yy'), '11.00 am', null, 120);

INSERT INTO Appointment
VALUES(10007, 1009, 1111, to_date('15/07/22', 'dd/mm/yy'), '09.00 am', null, 118);

INSERT INTO Appointment
VALUES(10008, 1008, 1107, to_date('13/07/22', 'dd/mm/yy'), '01.00 pm', null, 112);

INSERT INTO Appointment
VALUES(10009, 1002, 1103, to_date('14/07/22', 'dd/mm/yy'), '10.00 am', null, 111);

INSERT INTO Appointment
VALUES(10010, 1005, 1108, to_date('10/07/22', 'dd/mm/yy'), '11.00 am', 'Diabetes', 114);
```

## SQL Statements

- List all pet owner orders by their id.

```
SELECT *  
FROM owner  
ORDER BY owner_id;
```

The screenshot shows a database query tool interface. The top toolbar includes icons for running queries, saving, and other functions, along with a timer showing 0.078 seconds. Below the toolbar, there are two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab is active, displaying the following SQL query:

```
1 -- List all pet owner order by their id  
2 SELECT *  
3 FROM owner  
4 ORDER BY owner_id;
```

Below the query editor, there is a 'Script Output' tab. It shows the results of the query execution, indicating that the task was completed in 0.078 seconds. The results are displayed in a table with the following columns: OWNER\_ID, O\_FNAME, O\_LNAME, O\_PHONE, and O\_ADDRESS. The table contains 11 rows of data, representing pet owners and their orders.

OWNER_ID	O_FNAME	O_LNAME	O_PHONE	O_ADDRESS
1001	Esmee	Benitez	01100000111	Gombak, Selangor
1002	Freyja	Paul	01100000112	Ampang, Selangor
1003	Mohammad	Iman	01100000113	Bandar Baru Bangi, Selangor
1004	Mohammad	Asraf	01100000114	Banting, Selangor
1005	Mohammad	Arif	01100001114	Gombak, Selangor
1006	Danial	Hakim	01100000115	Medan Idaman, Selangor
1007	Raihan	Kabir	01100000116	Batu Caves, Selangor
1008	Mohammad	Hahmid	01100001116	Gombak, Selangor
1009	Mohammad	Rakib	01100001117	Bukit Nasa, Kuala Lumpur
1010	Abdur	Rahman	01100000118	Gombak, Selangor
1011	Abdur	Salem	01100010118	

11 rows selected.

- **List all pet owners with their pet information.**

```
SELECT o.owner_id, o.o_fname, o.o_lname, p.pet_id, p.pet_name, p.pet_type
FROM owner o
JOIN pet p
ON o.owner_id = p.owner_id
ORDER BY o.owner_id;
```

0.078 seconds

Worksheet Query Builder

```
1 -- Select all pet owner with their pet information
2 SELECT o.owner_id, o.o_fname, o.o_lname, p.pet_id, p.pet_name, p.pet_type
3 FROM owner o
4 JOIN pet p
5 ON o.owner_id = p.owner_id
6 ORDER BY o.owner_id;
```

Script Output x

Task completed in 0.078 seconds

OWNER_ID	O_FNAME	O_LNAME	PET_ID	PET_NAME	PET_TYPE
1001	Esmee	Benitez	1101	Mini	cat
1002	Freyja	Paul	1102	Luna	cat
1002	Freyja	Paul	1103	Teddy	dog
1003	Mohammad	Iman	1104	Bill	dog
1004	Mohammad	Asraf	1105	Bella	dog
1004	Mohammad	Asraf	1106	Buddy	dog
1005	Mohammad	Arif	1108	Loki	cat
1006	Danial	Hakim	1109	Oliver	cat
1007	Raihan	Kabir	1110	Bubba	bird
1008	Mohammad	Hahmid	1107	Milo	cat
1009	Mohammad	Rakib	1111	Baldy	cat
1010	Abdur	Rahman	1112	Putih	cat

12 rows selected.

- List all the vet id, first name, last name and salary from the Vet table who have a salary per month greater than 8000.

```
SELECT vet_id, v_fname, v_lname, salary_per_mon
FROM VET
WHERE salary_per_mon > 8000;
```

The screenshot shows a database query tool interface. At the top, there's a toolbar with various icons and a timer showing '0.078 seconds'. Below the toolbar, there's a 'Worksheet' tab and a 'Query Builder' tab. The 'Query Builder' tab is active, showing a SQL query in a text area. The query is as follows:

```
-- List all the vet id, first name, last name and salary from the Vet table who have a salary per month greater than 8000
SELECT vet_id, v_fname, v_lname, salary_per_mon
FROM VET
WHERE salary_per_mon > 8000;
```

Below the query editor, there's a 'Script Output' tab. It shows the results of the query in a table format. The table has four columns: VET\_ID, V\_FNAME, V\_LNAME, and SALARY\_PER\_MON. The results are as follows:

VET_ID	V_FNAME	V_LNAME	SALARY_PER_MON
114	Yu	Yan	9000
115	Ying	Yue	9000
116	Safa	Kabir	8500
119	Mohammad	Razak	10000
120	Mohammad	Mubarak	10000

- List all the pet's owner information from the Owner table who have more than one pet.

```
SELECT *
FROM owner
WHERE owner_id IN (SELECT UNIQUE(p.owner_id)
                   FROM pet p, (SELECT owner_id, COUNT(pet_id) AS cn FROM pet GROUP
                                BY owner_id) s
                   WHERE p.owner_id = s.owner_id AND s.cn > 1);
```

0.095 seconds

Worksheet Query Builder

```

1 -- List all the pet's owner information from the Owner table who have more than one pet.
2 SELECT *
3 FROM owner
4 WHERE owner_id IN (SELECT UNIQUE(p.owner_id)
5                     FROM pet p, (SELECT owner_id, COUNT(pet_id) AS cn FROM pet GROUP BY owner_id) s
6                     WHERE p.owner_id = s.owner_id AND s.cn > 1);

```

Script Output x

Task completed in 0.095 seconds

OWNER_ID	O_FNAME	O_LNAME	O_PHONE	O_ADDRESS
1002	Freyja	Paul	01100000112	Ampang, Selangor
1004	Mohammad	Asraf	01100000114	Banting, Selangor

- Change appointment date to 25/07/2022 where the pet's name is 'Putih' and it is a cat.

```

UPDATE appointment
SET appointment."Date" = to_date('25/07/2022', 'dd/mm/yyyy')
WHERE pet_id = (SELECT pet_id FROM pet WHERE pet_name = 'Putih' AND pet_type = 'cat');

```

0.063 seconds

Worksheet Query Builder

```

1 -- Change appointment date to 25/07/2022 where the pet's name is 'Putih' and it is a cat.
2 UPDATE appointment
3 SET appointment."Date" = to_date('25/07/2022', 'dd/mm/yyyy')
4 WHERE pet_id = (SELECT pet_id FROM pet WHERE pet_name = 'Putih' AND pet_type = 'cat');

```

Script Output x

Task completed in 0.063 seconds

1 row updated.

- List all appointments from the appointment table with owners name and their pet names ordered by appointment date and time.

```
SELECT a.appointment_id, a."Date", a."Time", a.owner_id, a.pet_id, o.o_fname || ' ' ||
o.o_lname AS owner_name, p.pet_name
FROM appointment a
LEFT JOIN owner o ON a.owner_id = o.owner_id
LEFT JOIN pet p ON a.pet_id = p.pet_id
ORDER BY a."Date", a."Time";
```

0.078 seconds

Worksheet Query Builder

```
-- List all appointment from appointment table with owners name and their pet names order by appointment date and time
1 SELECT a.appointment_id, a."Date", a."Time", a.owner_id, a.pet_id, o.o_fname || ' ' || o.o_lname AS owner_name, p.pet_name
2 FROM appointment a
3 LEFT JOIN owner o ON a.owner_id = o.owner_id
4 LEFT JOIN pet p ON a.pet_id = p.pet_id
5 ORDER BY a."Date", a."Time";
```

Script Output x Task completed in 0.078 seconds

APPOINTMENT_ID	Date	Time	OWNER_ID	PET_ID	OWNER_NAME	PET_NAME
10003	09/07/2022	12.00 pm	1003	1104	Mohammad Iman	Bill
10010	10/07/2022	11.00 am	1005	1108	Mohammad Arif	Loki
10001	11/07/2022	11.00 am	1001	1101	Esmee Benitez	Mini
10008	13/07/2022	01.00 pm	1008	1107	Mohammad Hahmid	Milo
10004	14/07/2022	05.00 pm	1004	1105	Mohammad Asraf	Bella
10009	14/07/2022	10.00 am	1002	1103	Freyja Paul	Teddy
10002	14/07/2022	10.00 am	1002	1102	Freyja Paul	Luna
10007	15/07/2022	09.00 am	1009	1111	Mohammad Rakib	Baldy
10005	18/07/2022	10.00 am	1006	1109	Danial Hakim	Oliver
10006	25/07/2022	11.00 am	1010	1112	Abdur Rahman	Putih

10 rows selected.

- List the appointment id, date, owner name, pet name, pet type order by appointment date.

```
SELECT a.appointment_id, a."Date", CONCAT(CONCAT(o.o_fname, ' '), o.o_lname),
p.pet_name, p.pet_type
FROM appointment a
LEFT JOIN owner o USING (owner_id)
LEFT JOIN pet p USING (pet_id);
```



0.062 seconds

Worksheet Query Builder

```

1 -- List the appointment id, date, owner name, pet name, pet type order by appointment date
2 SELECT a.appointment_id, a."Date", CONCAT(CONCAT(o.o_fname, ' '), o.o_lname), p.pet_name, p.pet_type
3 FROM appointment a
4 LEFT JOIN owner o USING (owner_id)
5 LEFT JOIN pet p USING (pet_id);

```

Script Output x

Task completed in 0.062 seconds

APPOINTMENT_ID	Date	CONCAT (CONCAT (O.O_FNAME, ' '), O.O_LNAME)	PET_NAME	PET_TYPE
10001	11/07/2022	Esmee Benitez	Mini	cat
10002	14/07/2022	Freyja Paul	Luna	cat
10009	14/07/2022	Freyja Paul	Teddy	dog
10003	09/07/2022	Mohammad Iman	Bill	dog
10004	14/07/2022	Mohammad Asraf	Bella	dog
10008	13/07/2022	Mohammad Hahmid	Milo	cat
10010	10/07/2022	Mohammad Arif	Loki	cat
10005	18/07/2022	Danial Hakim	Oliver	cat
10007	15/07/2022	Mohammad Rakib	Baldy	cat
10006	25/07/2022	Abdur Rahman	Putih	cat

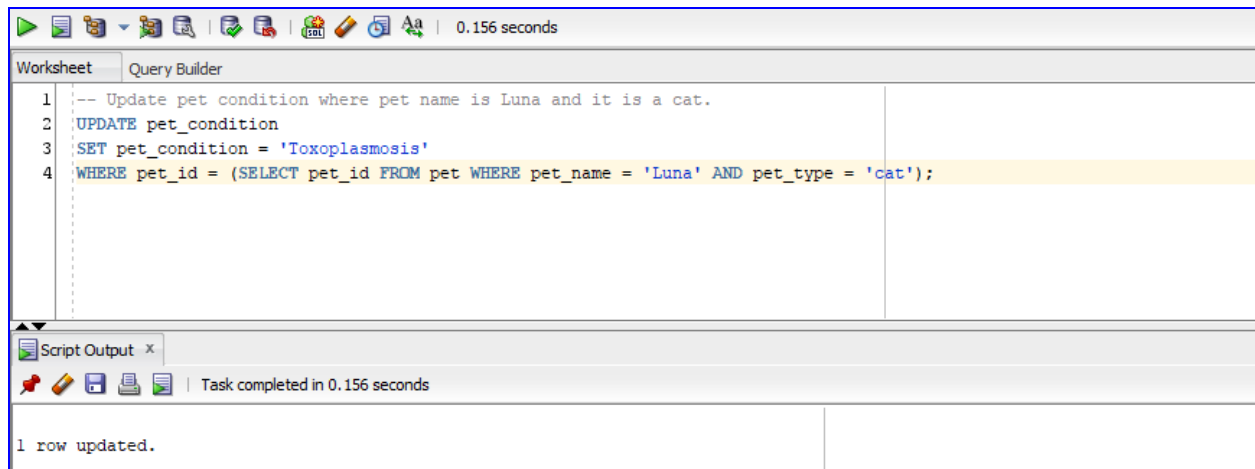
10 rows selected.

- Update pet condition where pet name is *Luna* and it is a *cat*.

```

UPDATE pet_condition
SET pet_condition = 'Toxoplasmosis'
WHERE pet_id = (SELECT pet_id FROM pet WHERE pet_name = 'Luna' AND pet_type =
'cat');

```



## Procedures

- **Create a procedure which creates a new appointment. (It takes input of owner id, pet id, appointment date, appointment time, and vet id.)**

```
SET SERVEROUTPUT ON;  
SET VERIFY OFF;  
  
CREATE OR REPLACE PROCEDURE makeAppointment  
(  
    o_id IN appointment.owner_id%TYPE,  
    p_id IN appointment.pet_id%TYPE,  
    a_date IN appointment."Date"%TYPE,  
    a_time IN appointment."Time"%TYPE,  
    v_id IN appointment.vet_id %TYPE  
)  
IS  
    po_id owner.owner_id%TYPE;  
    is_v_exist vet.vet_id%TYPE;  
    is_p_exist pet.pet_id%TYPE;  
    is_o_exist owner.owner_id%TYPE;  
    next_appointment_id appointment.appointment_id%TYPE;  
BEGIN  
    SELECT MAX(appointment_id)  
    INTO next_appointment_id  
    FROM appointment;
```

```

next_appointment_id := next_appointment_id + 1;

SELECT owner_id
INTO po_id
FROM pet
WHERE pet_id = p_id;

SELECT COUNT(*)
INTO is_o_exist
FROM owner
WHERE owner_id = o_id;

SELECT COUNT(*)
INTO is_p_exist
FROM pet
WHERE pet_id = p_id;

SELECT COUNT(*)
INTO is_v_exist
FROM vet
WHERE vet_id = v_id;

IF po_id = o_id AND is_o_exist = 1 AND is_p_exist = 1 AND is_v_exist = 1 THEN
    INSERT INTO appointment(appointment_id, owner_id, pet_id, "Date", "Time", vet_id)
    VALUES(next_appointment_id, o_id, p_id, a_date, a_time, v_id);

    DBMS_OUTPUT.PUT_LINE('New Appointment Created Successfully. ');
ELSE
    DBMS_OUTPUT.PUT_LINE('Your given Owner ID does not belong to the given Pet
ID. ');
END IF;
END;
/

```

```

ACCEPT o_id PROMPT 'Enter Owner ID: ';
ACCEPT p_id PROMPT 'Enter Pet ID: ';
ACCEPT a_date PROMPT 'Enter Appointment date (dd/mm/yyyy): ';
ACCEPT a_time PROMPT 'Enter appointment time: ';
ACCEPT v_id PROMPT 'Enter Vet / doctor ID: ';

BEGIN
    makeAppointment(&o_id, &p_id, to_date('&a_date', 'dd/mm/yyyy'), '&a_time', &v_id);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Your input is wrong. ');
END;

```

WorksheetQuery Builder

```

1  -- Procedure 1: Create a procedure which creates a new appointment. (It takes input of owner id, pet id, appoint
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE PROCEDURE makeAppointment
5  (
6      o_id IN appointment.owner_id%TYPE,
7      p_id IN appointment.pet_id%TYPE,
8      a_date IN appointment."Date"%TYPE,
9      a_time IN appointment."Time"%TYPE,
10     v_id IN appointment.vet_id %TYPE
11 )
12 IS
13     po_id owner.owner_id%TYPE;
14     is_v_exist vet.vet_id%TYPE;
15     is_p_exist pet.pet_id%TYPE;
16     is_o_exist owner.owner_id%TYPE;
17     next_appointment_id appointment.appointment_id%TYPE;
18 BEGIN
19     SELECT MAX(appointment_id)
20     INTO next_appointment_id
21     FROM appointment;
22
23     next_appointment_id := next_appointment_id + 1;

```

Script Output x

Task completed in 0.116 seconds

Procedure MAKEAPPOINTMENT compiled

WorksheetQuery Builder

```

47     VALUES(next_appointment_id, o_id, p_id, a_date, a_time, v_id);
48
49     DBMS_OUTPUT.PUT_LINE('New Appointment Created Successfully. ');
50
51     ELSE
52         DBMS_OUTPUT.PUT_LINE('Your given Owner ID does not belong to the given Pet ID. ');
53     END IF;
54
55 END;
56
57 ACCEPT o_id PROMPT 'Enter Owner ID: ':
58 ACCEPT p_id PROMPT 'Enter Pet ID: ':
59 ACCEPT a_date PROMPT 'Enter Appointment date (dd/mm/yyyy): ':
60 ACCEPT a_time PROMPT 'Enter appointment time: ':
61 ACCEPT v_id PROMPT 'Enter Vet / doctor ID: ':
62
63 BEGIN
64     makeAppointment(so_id, ap_id, to_date('sa_date', 'dd
65 EXCEPTION
66 WHEN OTHERS THEN
67     DBMS_OUTPUT.PUT_LINE('Your given input is wrong.
68 END;
69

```

Script Output x

ScriptRunner Task

Enter Value

Enter Owner ID:

1007

OKCancel

Procedure MAKEAPPOINTMENT compiled

WorksheetQuery Builder

```

47     VALUES(next_appointment_id, o_id, p_id, a_date, a_time, v_id);
48
49     DBMS_OUTPUT.PUT_LINE('New Appointment Created Successfully. ');
50
51     ELSE
52         DBMS_OUTPUT.PUT_LINE('Your given Owner ID does not belong to the given Pet ID. ');
53     END IF;
54
55 END;
56
57 ACCEPT o_id PROMPT 'Enter Owner ID: ':
58 ACCEPT p_id PROMPT 'Enter Pet ID: ':
59 ACCEPT a_date PROMPT 'Enter Appointment date (dd/mm/yyyy): ':
60 ACCEPT a_time PROMPT 'Enter appointment time: ':
61 ACCEPT v_id PROMPT 'Enter Vet / doctor ID: ':
62
63 BEGIN
64     makeAppointment(so_id, ap_id, to_date('sa_date', 'dd
65 EXCEPTION
66 WHEN OTHERS THEN
67     DBMS_OUTPUT.PUT_LINE('Your given input is wrong.
68 END;
69

```

Script Output x

ScriptRunner Task

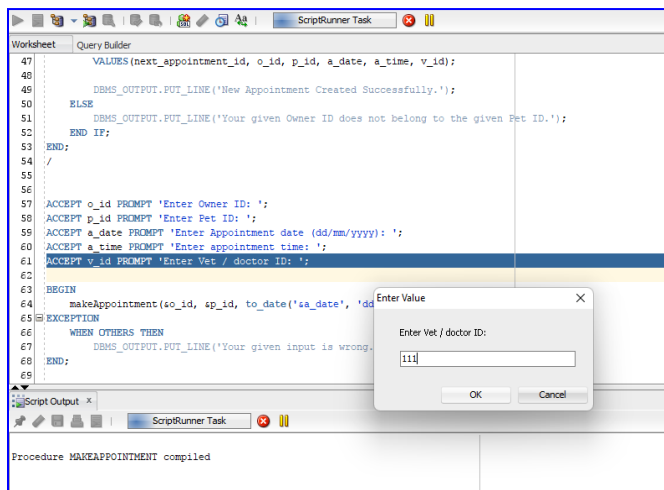
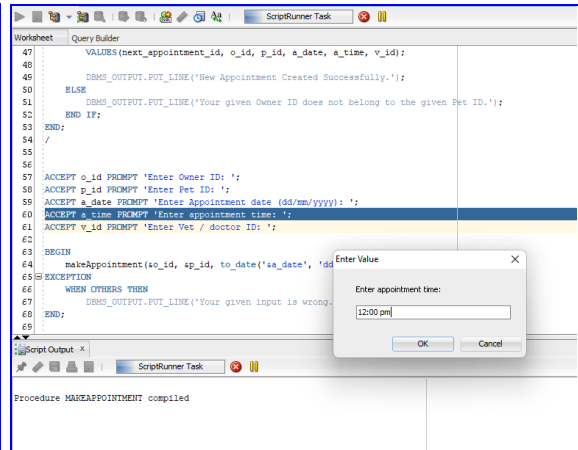
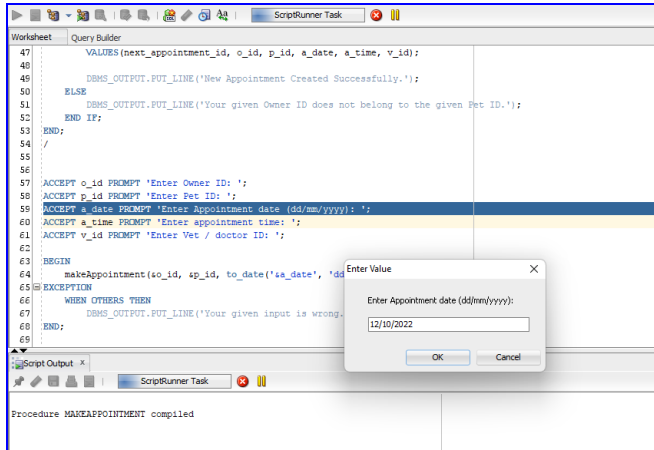
Enter Value

Enter Pet ID:

1110


OKCancel

Procedure MAKEAPPOINTMENT compiled



Worksheet	Query Builder
<pre> 47      VALUES(next_appointment_id, o_id, p_id, a_date, a_time, v_id); 48 49      DBMS_OUTPUT.PUT_LINE('New Appointment Created Successfully.');</pre>	
<pre> 50  ELSE 51      DBMS_OUTPUT.PUT_LINE('Your given Owner ID does not belong to the given Pet ID.');</pre>	
<pre> 52  END IF; 53  END; 54  / 55 56</pre>	
<pre> 57  ACCEPT o_id PROMPT 'Enter Owner ID: '; 58  ACCEPT p_id PROMPT 'Enter Pet ID: '; 59  ACCEPT a_date PROMPT 'Enter Appointment date (dd/mm/yyyy): '; 60  ACCEPT a_time PROMPT 'Enter appointment time: '; 61  ACCEPT v_id PROMPT 'Enter Vet / doctor ID: '; 62 63  BEGIN 64      makeAppointment(so_id, sp_id, to_date('sa_date', 'dd/mm/yyyy'), 'sa_time', sv_id); 65  EXCEPTION 66      WHEN OTHERS THEN 67          DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');</pre>	
<pre> 68  END; 69</pre>	

Script Output x
   Task completed in 112.076 seconds
<pre> Procedure MAKEAPPOINTMENT compiled  New Appointment Created Successfully.  PL/SQL procedure successfully completed.</pre>

- **Create a procedure that can change the appointment date. (It takes input of owner id, pet id, new appointment date, new appointment time.)**

```

SET SERVEROUTPUT ON;
SET VERIFY OFF;

CREATE OR REPLACE PROCEDURE changeAppointment
(
    o_id IN appointment.owner_id%TYPE,
    p_id IN appointment.pet_id%TYPE,
    u_a_date IN appointment."Date"%TYPE,
    u_a_time IN appointment."Time"%TYPE
)
IS
BEGIN
    UPDATE appointment
```

```
SET "Date" = u_a_date, "Time" = u_a_time  
WHERE owner_id = o_id AND pet_id = p_id;
```

```
DBMS_OUTPUT.PUT_LINE('Update Appointment date and time Successfully.');
```

```
END;  
/
```

```
ACCEPT o_id PROMPT 'Enter Owner ID: ';  
ACCEPT p_id PROMPT 'Enter Pet ID: ';  
ACCEPT u_a_date PROMPT 'Enter Appointment date (dd/mm/yyyy): ';  
ACCEPT u_a_time PROMPT 'Enter appointment time: ';
```

```
BEGIN  
    changeAppointment(&o_id, &p_id, to_date('&u_a_date', 'dd/mm/yyyy'), '&u_a_time');  
EXCEPTION  
    WHEN OTHERS THEN  
        DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');
```

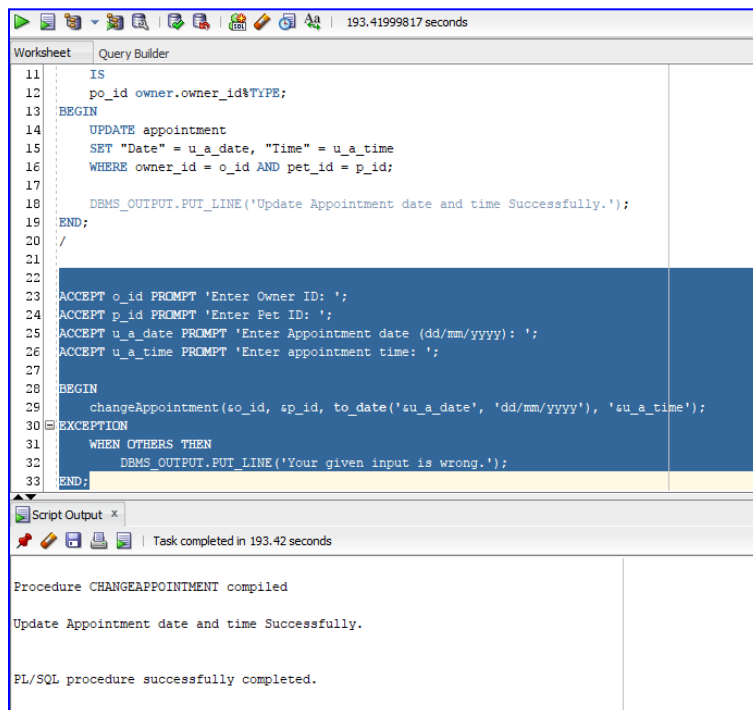
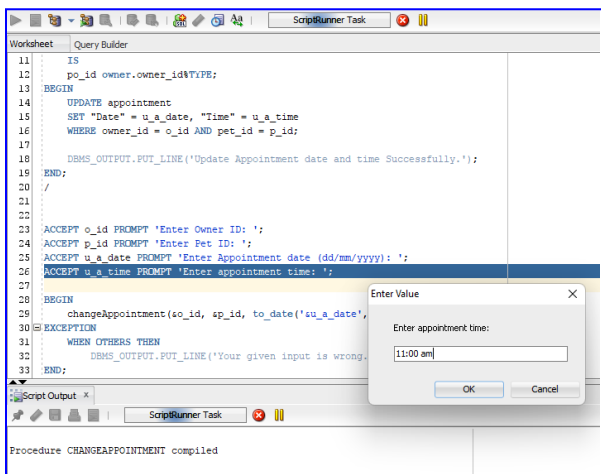
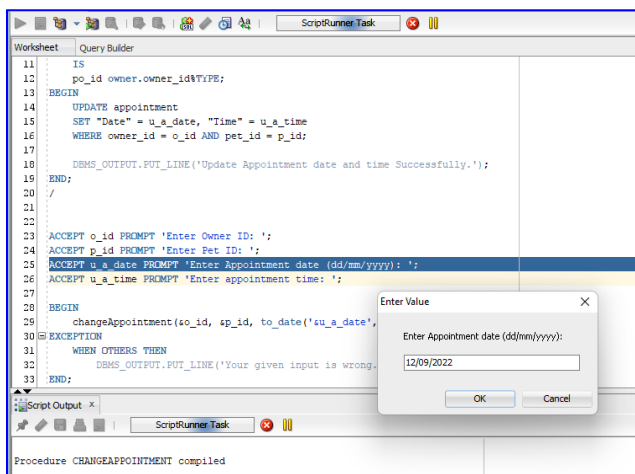
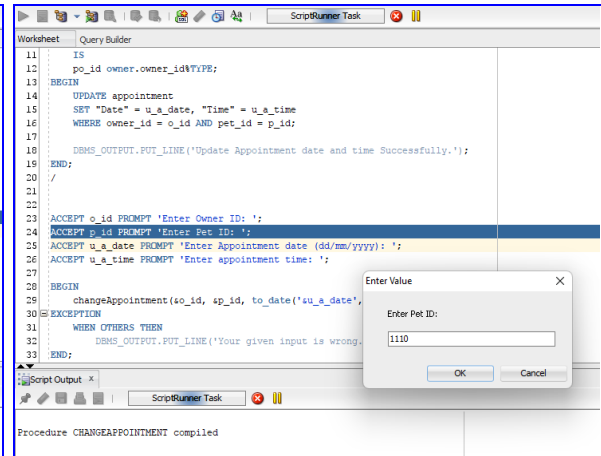
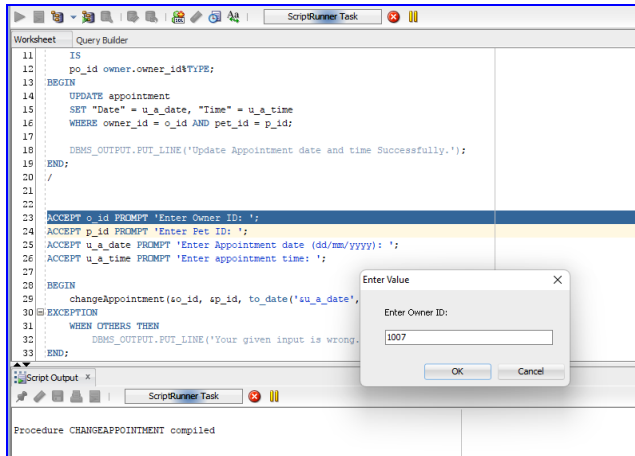
```
END;
```

The screenshot shows a SQL IDE interface with a 'Worksheet' tab. The main editor contains a PL/SQL script. The script starts with a comment: '-- Procedure 2: Create a procedure that can change the appointment date. (It takes input of owner id, pet id, new appointment date, new appointment time.)'. It then sets 'SERVEROUTPUT ON;' and 'VERIFY OFF;'. The main part is a 'CREATE OR REPLACE PROCEDURE changeAppointment' block. The procedure signature is: 'o\_id IN appointment.owner\_id%TYPE, p\_id IN appointment.pet\_id%TYPE, u\_a\_date IN appointment."Date"%TYPE, u\_a\_time IN appointment."Time"%TYPE'. The body of the procedure is: 'BEGIN UPDATE appointment SET "Date" = u\_a\_date, "Time" = u\_a\_time WHERE owner\_id = o\_id AND pet\_id = p\_id; DBMS\_OUTPUT.PUT\_LINE('Update Appointment date and time Successfully.');

```
END;  
/
```

Below the main editor, there is a 'Script Output' tab. It shows a status bar with a red flag icon, a green flag icon, and a green flag icon, followed by the text 'Task completed in 0.084 seconds'. Below the status bar, there is a table with two columns. The first column is 'Procedure' and the second column is 'Status'. The first row of the table has 'CHANGEAPPOINTMENT' in the first column and 'compiled' in the second column.

Procedure	Status
CHANGEAPPOINTMENT	compiled





- **Delete the appointment. (It takes input of owner id, pet id.)**

```
SET SERVEROUTPUT ON;
SET VERIFY OFF;

CREATE OR REPLACE PROCEDURE deleteAppointment
(
    o_id IN appointment.owner_id%TYPE,
    p_id IN appointment.pet_id%TYPE
)
IS
BEGIN
    DELETE FROM appointment
    WHERE owner_id = o_id AND pet_id = p_id;

    DBMS_OUTPUT.PUT_LINE('Delete Appointment Successfully.');
```

```
END;
/
```

```
ACCEPT o_id PROMPT 'Enter Owner ID: ';
ACCEPT p_id PROMPT 'Enter Pet ID: ';

BEGIN
    deleteAppointment(&o_id, &p_id);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');
```

```
END;
```

0.163 seconds

Worksheet Query Builder

```
1  -- Procedure 3: Delete the appointment. (It takes input of owner id, pet id.)
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE PROCEDURE deleteAppointment
5  (
6      o_id IN appointment.owner_id%TYPE,
7      p_id IN appointment.pet_id%TYPE
8  )
9  IS
10     po_id owner.owner_id%TYPE;
11 BEGIN
12     DELETE FROM appointment
13     WHERE owner_id = o_id AND pet_id = p_id;
14
15     DBMS_OUTPUT.PUT_LINE('Delete Appointment Successfully.');

Script Output x



Task completed in 0.163 seconds



Procedure DELETEAPPOINTMENT compiled


```

ScriptRunner Task

Worksheet Query Builder

```
7  p_id IN appointment.pet_id%TYPE
8  )
9  IS
10     po_id owner.owner_id%TYPE;
11 BEGIN
12     DELETE FROM appointment
13     WHERE owner_id = o_id AND pet_id = p_id;
14
15     DBMS_OUTPUT.PUT_LINE('Delete Appointment Successfully.');

Script Output x



ScriptRunner Task



Procedure DELETEAPPOINTMENT compiled



Enter Value



Enter Owner ID:



1007



OK Cancel


```

ScriptRunner Task

Worksheet Query Builder

```
7  p_id IN appointment.pet_id%TYPE
8  )
9  IS
10     po_id owner.owner_id%TYPE;
11 BEGIN
12     DELETE FROM appointment
13     WHERE owner_id = o_id AND pet_id = p_id;
14
15     DBMS_OUTPUT.PUT_LINE('Delete Appointment Successfully.');

Script Output x



ScriptRunner Task



Procedure DELETEAPPOINTMENT compiled



Enter Value



Enter Pet ID:



1110



OK Cancel


```

Worksheet Query Builder 76.41500092 seconds

```

7      p_id IN appointment.pet_id%TYPE
8    )
9    IS
10     po_id owner.owner_id%TYPE;
11 BEGIN
12     DELETE FROM appointment
13     WHERE owner_id = o_id AND pet_id = p_id;
14
15     DBMS_OUTPUT.PUT_LINE('Delete Appointment Successfully.');
```

```

16 END;
17 /
18
19
20 ACCEPT o_id PROMPT 'Enter Owner ID: ';
21 ACCEPT p_id PROMPT 'Enter Pet ID: ';
22
23 BEGIN
24     deleteAppointment(&o_id, &p_id);
25 EXCEPTION
26     WHEN OTHERS THEN
27         DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');
```

```

28 END;
29

```

Script Output x Task completed in 76.415 seconds

```

Procedure DELETEAPPOINTMENT compiled

Delete Appointment Successfully.

PL/SQL procedure successfully completed.

```

- Add a new owner. (It takes input of the owner's first name, last name, phone number, and address.)

```

SET SERVEROUTPUT ON;
SET VERIFY OFF;

CREATE OR REPLACE PROCEDURE newOwner
(
    ow_fname IN owner.o_fname%TYPE,
    ow_lname IN owner.o_lname%TYPE,
    ow_phone IN owner.o_phone%TYPE,
    ow_address IN owner.o_address%TYPE

```

```

)
IS
next_ow_id owner.owner_id%TYPE;
BEGIN
    SELECT MAX(owner_id)
    INTO next_ow_id
    FROM owner;

    next_ow_id := next_ow_id + 1;

    INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
    VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);

    DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

/

```

ACCEPT ow_fname PROMPT 'Enter your first name: ';
ACCEPT ow_lname PROMPT 'Enter your last name: ';
ACCEPT ow_phone PROMPT 'Enter your phone number: ';
ACCEPT ow_address PROMPT 'Enter your address: ';

BEGIN
    newOwner('&ow_fname', '&ow_lname', '&ow_phone', '&ow_address');
```

EXCEPTION

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

0.118 seconds

Worksheet Query Builder

```
1  -- Procedure 4: Add a new owner. (It takes input of the owner's first name, last name, phone number, and address.)
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE PROCEDURE newOwner
5  (
6      ow_fname IN owner.o_fname%TYPE,
7      ow_lname IN owner.o_lname%TYPE,
8      ow_phone IN owner.o_phone%TYPE,
9      ow_address IN owner.o_address%TYPE
10 )
11 IS
12     next_ow_id owner.owner_id%TYPE;
13 BEGIN
14     SELECT MAX(owner_id)
15     INTO next_ow_id
16     FROM owner;
17
18     next_ow_id := next_ow_id + 1;
19
20     INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
21     VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
22
23     DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

Script Output x

Task completed in 0.118 seconds

Procedure NEWOWNER compiled

ScriptRunner Task

Worksheet Query Builder

```
17  next_ow_id := next_ow_id + 1;
18
19  INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
20  VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
21
22  DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

ACCEPT ow\_fname PROMPT 'Enter your first name: ';

ACCEPT ow\_lname PROMPT 'Enter your last name: ';

ACCEPT ow\_phone PROMPT 'Enter your phone number: ';

ACCEPT ow\_address PROMPT 'Enter your address: ';

BEGIN

newOwner('ow\_fname', 'ow\_lname', 'ow\_phone', 'ow\_address');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

Script Output x

ScriptRunner Task

Procedure NEWOWNER compiled

Enter Value

Enter your first name:

Mc

OK Cancel

ScriptRunner Task

Worksheet Query Builder

```
17  next_ow_id := next_ow_id + 1;
18
19  INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
20  VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
21
22  DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

ACCEPT ow\_fname PROMPT 'Enter your first name: ';

ACCEPT ow\_lname PROMPT 'Enter your last name: ';

ACCEPT ow\_phone PROMPT 'Enter your phone number: ';

ACCEPT ow\_address PROMPT 'Enter your address: ';

BEGIN

newOwner('ow\_fname', 'ow\_lname', 'ow\_phone', 'ow\_address');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

Script Output x

ScriptRunner Task

Procedure NEWOWNER compiled

Enter Value

Enter your last name:

Emon

OK Cancel

Worksheet Query Builder

```

17 next_ow_id := next_ow_id + 1;
18
19 INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
20 VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
21
22 DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

ACCEPT ow\_fname PROMPT 'Enter your first name: ';

ACCEPT ow\_lname PROMPT 'Enter your last name: ';

ACCEPT ow\_phone PROMPT 'Enter your phone number: ';

ACCEPT ow\_address PROMPT 'Enter your address: ';

BEGIN

newOwner('ow\_fname', 'ow\_lname', 'ow\_phone', 'ow\_address');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

Script Output x

ScriptRunner Task

Procedure NEWOWNER compiled

Enter Value

Enter your phone number:

0116212641

OK Cancel

Worksheet Query Builder

```

17 next_ow_id := next_ow_id + 1;
18
19 INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
20 VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
21
22 DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

ACCEPT ow\_fname PROMPT 'Enter your first name: ';

ACCEPT ow\_lname PROMPT 'Enter your last name: ';

ACCEPT ow\_phone PROMPT 'Enter your phone number: ';

ACCEPT ow\_address PROMPT 'Enter your address: ';

BEGIN

newOwner('ow\_fname', 'ow\_lname', 'ow\_phone', 'ow\_address');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

Script Output x

ScriptRunner Task

Procedure NEWOWNER compiled

Enter Value

Enter your address:

LIA, Gombak

OK Cancel

181.14399719 seconds

Worksheet Query Builder

```

17
18 next_ow_id := next_ow_id + 1;
19
20 INSERT INTO owner(owner_id, o_fname, o_lname, o_phone, o_address)
21 VALUES(next_ow_id, ow_fname, ow_lname, ow_phone, ow_address);
22
23 DBMS_OUTPUT.PUT_LINE('New owner created Successfully.');
```

END;

ACCEPT ow\_fname PROMPT 'Enter your first name: ';

ACCEPT ow\_lname PROMPT 'Enter your last name: ';

ACCEPT ow\_phone PROMPT 'Enter your phone number: ';

ACCEPT ow\_address PROMPT 'Enter your address: ';

BEGIN

newOwner('ow\_fname', 'ow\_lname', 'ow\_phone', 'ow\_address');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Your given input is wrong.');

END;

Script Output x

Task completed in 181.144 seconds

Procedure NEWOWNER compiled

New owner created Successfully.

PL/SQL procedure successfully completed.

- **Delete an owner. (It takes input of the owner's id.)**

```
SET SERVEROUTPUT ON;
SET VERIFY OFF;

CREATE OR REPLACE PROCEDURE deleteOwner
(
    ow_id IN owner.owner_id%TYPE
)
IS
BEGIN
    DELETE FROM owner
    WHERE owner_id = ow_id;

    DELETE FROM pet
    WHERE owner_id = ow_id;

    DBMS_OUTPUT.PUT_LINE('Delete owner Successfully.');
```

END;

/

```
ACCEPT ow_id PROMPT 'Enter Owner ID: ';

BEGIN
    deleteOwner(&ow_id);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');
```

END;

0.106 seconds

Worksheet Query Builder

```
1  -- Procedure 5: Delete an owner. (It takes input of the owner's id.)
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE PROCEDURE deleteOwner
5  (
6      ow_id IN owner.owner_id%TYPE
7  )
8  IS
9  BEGIN
10     DELETE FROM owner
11     WHERE owner_id = ow_id;
12
13     DELETE FROM pet
14     WHERE owner_id = ow_id;
15
16     DBMS_OUTPUT.PUT_LINE('Delete owner created Successfully.');

Script Output x



Task completed in 0.106 seconds



Procedure DELETEOWNER compiled


```

Project~2 OnlineVetAppSys\_procedure.sql ScriptRunner Task

Worksheet Query Builder

```
6      ow_id IN owner.owner_id%TYPE
7  )
8  IS
9  BEGIN
10     DELETE FROM owner
11     WHERE owner_id = ow_id;
12
13     DELETE FROM pet
14     WHERE owner_id = ow_id;
15
16     DBMS_OUTPUT.PUT_LINE('Delete owner created Successfully.');

Script Output x



ScriptRunner Task



Procedure DELETEOWNER compiled



Enter Value



Enter Owner ID:



1012



OK Cancel


```



The screenshot displays the Oracle SQL Developer environment. The top pane, titled 'Query Builder', contains a PL/SQL procedure named 'deleteOwner'. The procedure takes 'ow\_id' as input and deletes records from the 'owner' and 'pet' tables where 'owner\_id' matches 'ow\_id'. It includes a 'BEGIN' block with 'DELETE FROM' statements and a 'DBMS\_OUTPUT.PUT\_LINE' call. An 'EXCEPTION' block handles errors with a message 'Your given input is wrong.'. The bottom pane, titled 'Script Output', shows the execution results: 'Procedure DELETEOWNER compiled', 'Delete owner created Successfully.', and 'PL/SQL procedure successfully completed.'. The task was completed in 34.67 seconds.

```
6      ow_id IN owner.owner_id%TYPE
7    )
8  IS
9  BEGIN
10     DELETE FROM owner
11     WHERE owner_id = ow_id;
12
13     DELETE FROM pet
14     WHERE owner_id = ow_id;
15
16     DBMS_OUTPUT.PUT_LINE('Delete owner created Successfully.');
```

```
17 END;
18 /
19
20
21 ACCEPT ow_id PROMPT 'Enter Owner ID: ';
22
23 BEGIN
24     deleteOwner(ow_id);
25 EXCEPTION
26     WHEN OTHERS THEN
27         DBMS_OUTPUT.PUT_LINE('Your given input is wrong.');
```

```
28 END;
```

Script Output x

Task completed in 34.67 seconds

Procedure DELETEOWNER compiled

Delete owner created Successfully.

PL/SQL procedure successfully completed.

## Functions

- Create a function to give a discount on the charge of an appointment. Give 20% discount to those who live in Selangor, 10% discount to those who live in Kuala Lumpur.

```
SET SERVEROUTPUT ON;
SET VERIFY OFF;
```

```
CREATE OR REPLACE FUNCTION getDiscount (ow_address IN
owner.o_address%TYPE)
RETURN NUMBER
IS
```

```

        discount NUMBER;
BEGIN
    IF ow_address LIKE '%Selangor%' THEN discount := 0.2;
        ELIF ow_address LIKE '%Kuala Lumpur%' THEN discount := 0.2;
        ELSE discount := 0;
    END IF;
    RETURN discount;
END;
/

```

```

ACCEPT ow_id PROMPT 'Enter Owner ID: ';
DECLARE
    o_id owner.owner_id%TYPE;
    ow_fname owner.o_fname%TYPE;
    ow_lname owner.o_lname%TYPE;
    ow_address owner.o_address%TYPE;
    charge_of_vet vet.charge_per_app%TYPE;
    discount NUMBER;
    discount_bill NUMBER;
BEGIN
    o_id := &ow_id;

    SELECT charge_per_app
    INTO charge_of_vet
    FROM vet
    WHERE vet_id = (SELECT vet_id FROM appointment WHERE owner_id = o_id);

    SELECT o_fname, o_lname, o_address
    INTO ow_fname, ow_lname, ow_address
    FROM owner
    WHERE owner_id = o_id;

    discount := getDiscount(ow_address);
    discount_bill := charge_of_vet - (charge_of_vet * discount);

    DBMS_OUTPUT.PUT_LINE('Pet Owner Name: ' || ow_fname || ' ' || ow_lname);
    DBMS_OUTPUT.PUT_LINE('Pet Owner Address: ' || ow_address);
    DBMS_OUTPUT.PUT_LINE('Vet fee: ' || charge_of_vet);
    DBMS_OUTPUT.PUT_LINE('Eligible Discount: ' || (discount * 100) || '%' || chr(10));

    DBMS_OUTPUT.PUT_LINE('Vet fee after discount: ' || discount_bill);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Your given Owner ID does not have any appointment.
Or Your given Owner ID is Invalid. ');
END;

```

The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for file operations, editing, and execution. The top status bar shows the time taken for the current task: 0.125 seconds.

The main workspace is divided into two panes. The left pane, titled "Worksheet", contains the SQL code for creating a function. The right pane, titled "Query Builder", is currently empty.

The SQL code in the Worksheet pane is as follows:

```
1  -- Function 1: Create a function to give a discount on the charge of an appointment. Give 20% discount to those who live in Selangor,
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE FUNCTION getDiscount (ow_address IN owner.o_address%TYPE)
5  RETURN NUMBER
6  IS
7      discount NUMBER;
8  BEGIN
9      IF ow_address LIKE '%Selangor%' THEN discount := 0.2;
10      ELSIF ow_address LIKE '%Kuala Lumpur%' THEN discount := 0.2;
11      ELSE discount := 0;
12      END IF;
13      RETURN discount;
14  END;
```

Below the main code block, there is a section for declaring variables:

```
17  ACCEPT ow_id PROMPT 'Enter Owner ID: ';
18  DECLARE
19      o_id owner.owner_id%TYPE;
20      ow_fname owner.o_fname%TYPE;
21      ow_lname owner.o_lname%TYPE;
22      ow_address owner.o_address%TYPE;
23      charge_of_vet vet.charge_per_app%TYPE;
```

The bottom pane, titled "Script Output", shows the result of the compilation:

Function GETDISCOUNT compiled

OnlineVetAppSys\_function.sql | Project~3 | Assignment\_5.sql

ScriptRunner Task

Worksheet | Query Builder

```
7 discount NUMBER;
8 BEGIN
9 IF ow_address LIKE '%Selangor%' THEN discount := 0.2;
10 ELSIF ow_address LIKE '%Kuala Lumpur%' THEN discount := 0.2;
11 ELSE discount := 0;
12 END IF;
13 RETURN discount;
14 END;
15
16
17 ACCEPT ow_id PROMPT 'Enter Owner ID: ';
18 DECLARE
19 o_id owner.owner_id%TYPE;
20 ow_fname owner.o_fname%TYPE;
21 ow_lname owner.o_lname%TYPE;
22 ow_address owner.o_address%TYPE;
23 charge_of_vet vet.charge_per_app%TYPE;
24 discount NUMBER;
25 discount_bill NUMBER;
26 BEGIN
27 o_id := &ow_id;
28
29 SELECT charge_per_app
```

Enter Value

Enter Owner ID:

1001

OK Cancel

Script Output x

ScriptRunner Task

Function GETDISCOUNT compiled

Worksheet Query Builder 26.43499947 seconds

```

26 BEGIN
27   o_id := &ow_id;
28
29   SELECT charge_per_app
30   INTO charge_of_vet
31   FROM vet
32   WHERE vet_id = (SELECT vet_id FROM appointment WHERE owner_id = o_id);
33
34   SELECT o_fname, o_lname, o_address
35   INTO ow_fname, ow_lname, ow_address
36   FROM owner
37   WHERE owner_id = o_id;
38
39   discount := getDiscount(ow_address);
40   discount_bill := charge_of_vet - (charge_of_vet * discount);
41
42   DBMS_OUTPUT.PUT_LINE('Pet Owner Name: ' || ow_fname || ' ' || ow_lname);
43   DBMS_OUTPUT.PUT_LINE('Pet Owner Address: ' || ow_address);
44   DBMS_OUTPUT.PUT_LINE('Vet fee: ' || charge_of_vet);
45   DBMS_OUTPUT.PUT_LINE('Eligible Discount: ' || (discount * 100) || '%' || chr(10));
46
47   DBMS_OUTPUT.PUT_LINE('Vet fee after discount: ' || discount_bill);
48 EXCEPTION

```

Script Output x Task completed in 26.435 seconds

```

Function GETDISCOUNT compiled

Pet Owner Name: Esmee Benitez
Pet Owner Address: Gombak, Selangor
Vet fee: 80
Eligible Discount: 20%

Vet fee after discount: 64

PL/SQL procedure successfully completed.

```

- Create a function to check how many appointments a vet in a month. Function takes vet\_id, month name, and year as input.

```

SET SERVEROUTPUT ON;
SET VERIFY OFF;

CREATE OR REPLACE FUNCTION countAppointmentOfVet
(
  v_id IN vet.vet_id%TYPE,
  month_name IN VARCHAR2,
  g_year IN VARCHAR2
)
RETURN NUMBER
IS
  t_count NUMBER;
  s_date VARCHAR2(30);
  f_date VARCHAR2(30);

```

```

BEGIN
  IF month_name = 'January' THEN s_date := '01/01/' || g_year; f_date := '31/01/' ||
g_year;
    ELSIF month_name = 'February' THEN s_date := '01/02/' || g_year; f_date := '28/02/'
|| g_year;
    ELSIF month_name = 'March' THEN s_date := '01/03/' || g_year; f_date := '31/03/' ||
g_year;
    ELSIF month_name = 'April' THEN s_date := '01/04/' || g_year; f_date := '30/04/' ||
g_year;
    ELSIF month_name = 'May' THEN s_date := '01/05/' || g_year; f_date := '31/05/' ||
g_year;
    ELSIF month_name = 'June' THEN s_date := '01/06/' || g_year; f_date := '30/06/' ||
g_year;
    ELSIF month_name = 'July' THEN s_date := '01/07/' || g_year; f_date := '31/07/' ||
g_year;
    ELSIF month_name = 'August' THEN s_date := '01/08/' || g_year; f_date := '31/08/' ||
g_year;
    ELSIF month_name = 'September' THEN s_date := '01/09/' || g_year; f_date :=
'30/09/' || g_year;
    ELSIF month_name = 'October' THEN s_date := '01/10/' || g_year; f_date := '31/10/' ||
g_year;
    ELSIF month_name = 'November' THEN s_date := '01/11/' || g_year; f_date := '30/11/'
|| g_year;
    ELSIF month_name = 'December' THEN s_date := '01/12/' || g_year; f_date :=
'31/12/' || g_year;
    ELSE DBMS_OUTPUT.PUT_LINE('You Enter a Invalid month name. ');
    END IF;

  SELECT COUNT(*)
  INTO t_count
  FROM appointment
  WHERE vet_id = v_id
  AND "Date" BETWEEN to_date(s_date, 'dd/mm/yyyy') AND to_date(f_date,
'dd/mm/yyyy');

  RETURN t_count;
END;
/

```

```

ACCEPT v_id PROMPT 'Enter a Vet ID: ';
ACCEPT month_name PROMPT 'Enter a month (eg. January, February ): ';
ACCEPT g_year PROMPT 'Enter the year: ';
DECLARE
  vid vet.vet_id%TYPE;
  vet_fname vet.v_fname%TYPE;
  vet_lname vet.v_lname%TYPE;

```

```

g_month VARCHAR2(20);
t_count NUMBER;
BEGIN
    vid := &v_id;
    g_month := '&month_name';

    SELECT v_fname, v_lname
    INTO vet_fname, vet_lname
    FROM vet
    WHERE vet_id = vid;

    t_count := countAppiontmentOfVet(vid, g_month, '&g_year');

    DBMS_OUTPUT.PUT_LINE(vid || ' ' || vet_fname || ' ' || vet_lname || ' have ' || t_count || '
appointment in ' || g_month);
END;

```

The screenshot displays the Oracle SQL Developer environment. The main window shows a PL/SQL script in the 'Query Builder' tab. The script defines a function named 'countAppiontmentOfVet' that takes three parameters: 'v\_id' (NUMBER), 'month\_name' (VARCHAR2), and 'g\_year' (VARCHAR2). It returns a 'NUMBER'. The function body uses a series of IF-ELSIF statements to set date ranges for the first seven months of the year. The script is then executed, and the 'Script Output' window shows the message: 'Function COUNTAPPIONTMENTOFVET compiled'.

```

1  -- Function - 2: Create a function to check how many appointments have each vet. Function only take vet_id as input.
2  SET SERVEROUTPUT ON;
3  SET VERIFY OFF;
4  CREATE OR REPLACE FUNCTION countAppiontmentOfVet
5  (
6      v_id IN vet.vet_id%TYPE,
7      month_name IN VARCHAR2,
8      g_year IN VARCHAR2
9  )
10     RETURN NUMBER
11 IS
12     t_count NUMBER;
13     s_date VARCHAR2(30);
14     f_date VARCHAR2(30);
15
16 BEGIN
17     IF month_name = 'January' THEN s_date := '01/01/' || g_year; f_date := '31/01/' || g_year;
18     ELSIF month_name = 'February' THEN s_date := '01/02/' || g_year; f_date := '28/02/' || g_year;
19     ELSIF month_name = 'March' THEN s_date := '01/03/' || g_year; f_date := '31/03/' || g_year;
20     ELSIF month_name = 'April' THEN s_date := '01/04/' || g_year; f_date := '30/04/' || g_year;
21     ELSIF month_name = 'May' THEN s_date := '01/05/' || g_year; f_date := '31/05/' || g_year;
22     ELSIF month_name = 'June' THEN s_date := '01/06/' || g_year; f_date := '30/06/' || g_year;
23     ELSIF month_name = 'July' THEN s_date := '01/07/' || g_year; f_date := '31/07/' || g_year;

```

Script Output x

Task completed in 0.11 seconds

Function COUNTAPPIONTMENTOFVET compiled

