Weather Data

Case Study - Semester 2

PL/SQL

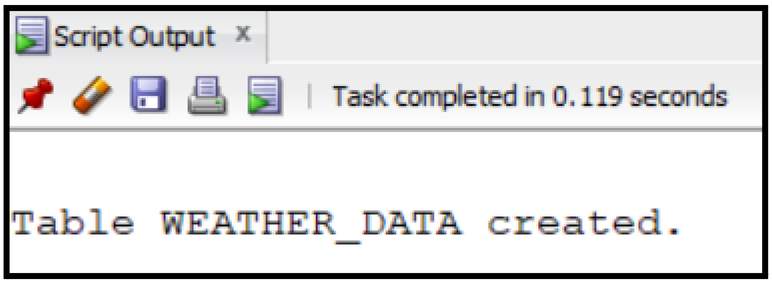
Group member - Zainab Hashmi, Roll no:. 1

Group member - Nandita Jaiswal, Roll no:. 14

```
SET SERVEROUTPUT ON;
```

-- create a table weather data with attributes (state_id , state_name, country , temperature, humidity, pressure, timestamp, AQI, message)

```
CREATE TABLE weather_data (
state_id     NUMBER(10),
state_name   VARCHAR2(50),
country      VARCHAR2(50),
temperature  NUMBER(5,2),
humidity     NUMBER(5,2),
pressure     NUMBER(7,2),
timestamp    TIMESTAMP,
AQI          NUMBER(5),
message      VARCHAR2(200)
);
```



```
Table WEATHER_DATA created.
```

-- Insert 20 Values in the table.

```
INSERT INTO weather_data VALUES(1, 'California', 'USA', 75.20, 62.50, 1013.25, TIMESTAMP
'2023-03-19 09:30:00', 75, 'Sunny');

INSERT INTO weather_data VALUES(2, 'New York', 'USA', 50.50, 45.20, 1012.50, TIMESTAMP
'2023-03-19 09:45:00', 55, 'Partly Cloudy');

INSERT INTO weather_data VALUES(3, 'Delhi', 'India', 80.80, 70.50, 1011.75, TIMESTAMP
'2023-03-19 10:00:00', 85, 'Clear Sky');

INSERT INTO weather_data VALUES(4, 'Ontario', 'Canada', 60.70, 55.80, 1014.00,
TIMESTAMP '2023-03-19 10:15:00', 65, 'Mostly Sunny');

INSERT INTO weather_data VALUES(5, 'Quebec', 'Canada', 40.90, 35.20, 1015.50,
TIMESTAMP '2023-03-19 10:30:00', 45, 'Cloudy with a chance of rain');
```

INSERT INTO weather_data VALUES(6, 'Punjab', 'India', 80.10, 60.40, 1012.25, TIMESTAMP '2023-03-19 10:45:00', 70, 'Partly Cloudy');

INSERT INTO weather_data VALUES(7, 'Arizona', 'USA', 85.30, 50.10, 1010.50, TIMESTAMP '2023-03-19 11:00:00', 80, 'Sunny');

INSERT INTO weather_data VALUES(8, 'Kerala', 'India', 65.60, 52.80, 1013.75, TIMESTAMP '2023-03-19 11:15:00', 60, 'Mostly Sunny');

INSERT INTO weather_data VALUES(9, 'Alberta', 'Canada', 55.40, 48.20, 1015.00, TIMESTAMP '2023-03-19 11:30:00', 50, 'Mostly Cloudy');

INSERT INTO weather_data VALUES(10, 'Nevada', 'USA', 70.20, 40.80, 1012.00, TIMESTAMP '2023-03-19 11:45:00', 65, 'Partly Sunny');

INSERT INTO weather_data VALUES(11, 'Utah', 'USA', 75.80, 42.70, 1011.00, TIMESTAMP '2023-03-19 12:00:00', 75, 'Sunny');

INSERT INTO weather_data VALUES(12, 'Gujarat', 'India', 50.10, 43.60, 1014.50, TIMESTAMP '2023-03-19 12:15:00', 45, 'Mostly Cloudy');

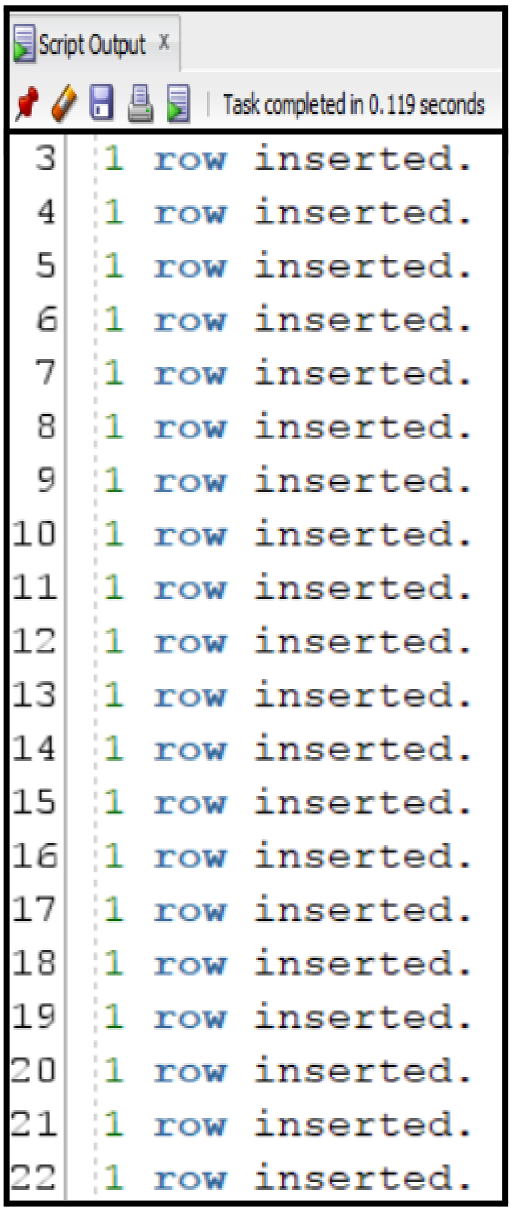
INSERT INTO weather_data VALUES(13, 'Manitoba', 'Canada', 55.80, 40.90, 1016.25, TIMESTAMP '2023-03-19 12:30:00', 50, 'Partly Cloudy');

INSERT INTO weather_data VALUES(14, 'Colorado', 'USA', 65.50, 55.20, 1011.75, TIMESTAMP '2023-03-19 12:45:00', 60, 'Mostly Sunny');

INSERT INTO weather_data VALUES(15, 'Montana', 'USA', 45.60, 35.90, 1014.00, TIMESTAMP '2023-03-19 12:15:00', 75, 'Sunny');

INSERT INTO weather_data VALUES(16, 'Mumbai', 'India', 27.5, 84.4, 1008.9, TIMESTAMP '2023-03-19 11:15:00', 72, 'Humid day');

INSERT INTO weather_data VALUES(17, 'Bangkok', 'Thailand', 32.7, 75.9, 1006.9, TIMESTAMP '2023-03-19 12:50:00', 70, 'Humid with scattered clouds');



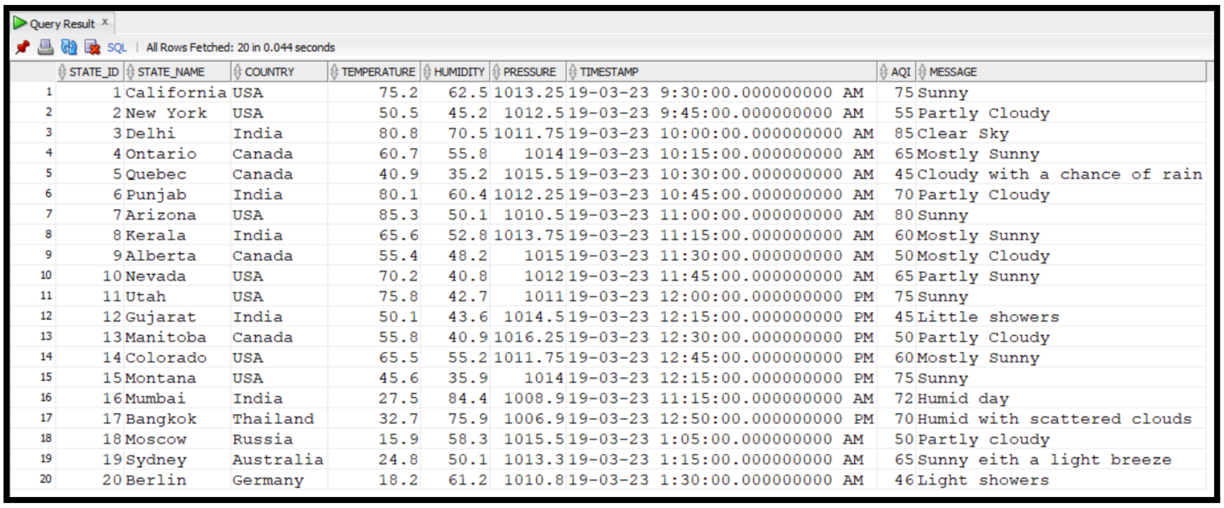
```
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
```

INSERT INTO weather_data VALUES(18, 'Moscow' , 'Russia', 15.9, 58.3, 1015.5, TIMESTAMP '2023-03-19 01:05:00', 50, 'Partly cloudy');

INSERT INTO weather_data VALUES(19, 'Sydney' , 'Australia', 24.8, 50.1, 1013.3, TIMESTAMP '2023-03-19 01:15:00', 65, 'Sunny eith a light breeze');

INSERT INTO weather_data VALUES(20, 'Berlin', 'Germany', 18.2, 61.2, 1010.8, TIMESTAMP '2023-03-19 01:30:00', 46, 'Light showers');

–Dataset

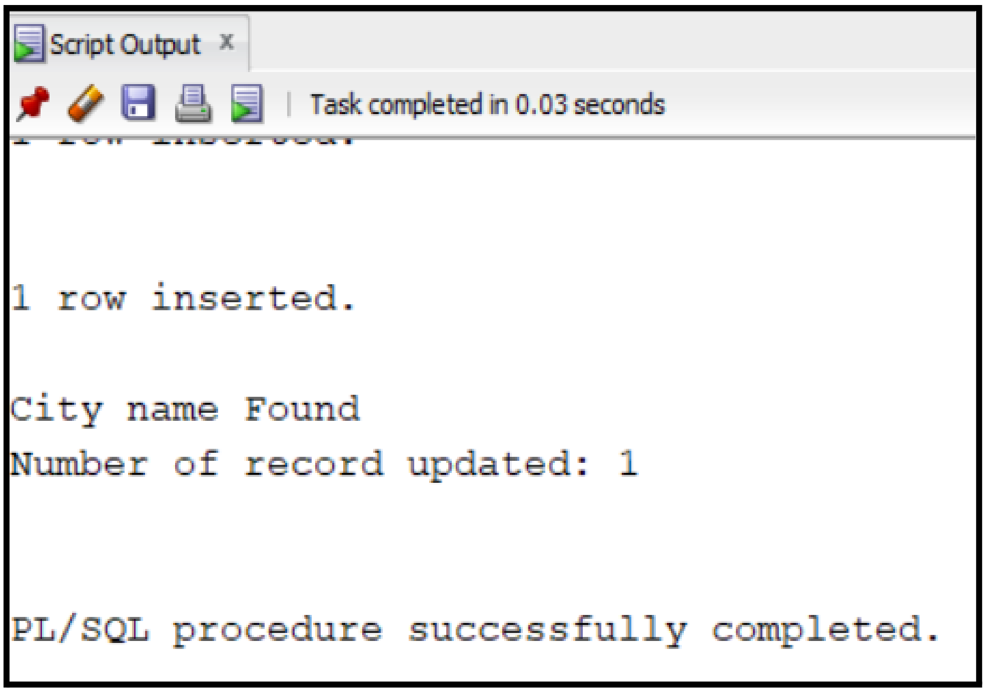

Query Result ×
SQL | All Rows Fetched: 20 in 0.044 seconds

	STATE_ID	STATE_NAME	COUNTRY	TEMPERATURE	HUMIDITY	PRESSURE	TIMESTAMP	AQI	MESSAGE
1	1	California	USA	75.2	62.5	1013.25	19-03-23 9:30:00.000000000 AM	75	Sunny
2	2	New York	USA	50.5	45.2	1012.5	19-03-23 9:45:00.000000000 AM	55	Partly Cloudy
3	3	Delhi	India	80.8	70.5	1011.75	19-03-23 10:00:00.000000000 AM	85	Clear Sky
4	4	Ontario	Canada	60.7	55.8	1014	19-03-23 10:15:00.000000000 AM	65	Mostly Sunny
5	5	Quebec	Canada	40.9	35.2	1015.5	19-03-23 10:30:00.000000000 AM	45	Cloudy with a chance of rain
6	6	Punjab	India	80.1	60.4	1012.25	19-03-23 10:45:00.000000000 AM	70	Partly Cloudy
7	7	Arizona	USA	85.3	50.1	1010.5	19-03-23 11:00:00.000000000 AM	80	Sunny
8	8	Kerala	India	65.6	52.8	1013.75	19-03-23 11:15:00.000000000 AM	60	Mostly Sunny
9	9	Alberta	Canada	55.4	48.2	1015	19-03-23 11:30:00.000000000 AM	50	Mostly Cloudy
10	10	Nevada	USA	70.2	40.8	1012	19-03-23 11:45:00.000000000 AM	65	Partly Sunny
11	11	Utah	USA	75.8	42.7	1011	19-03-23 12:00:00.000000000 PM	75	Sunny
12	12	Gujarat	India	50.1	43.6	1014.5	19-03-23 12:15:00.000000000 PM	45	Little showers
13	13	Manitoba	Canada	55.8	40.9	1016.25	19-03-23 12:30:00.000000000 PM	50	Partly Cloudy
14	14	Colorado	USA	65.5	55.2	1011.75	19-03-23 12:45:00.000000000 PM	60	Mostly Sunny
15	15	Montana	USA	45.6	35.9	1014	19-03-23 12:15:00.000000000 PM	75	Sunny
16	16	Mumbai	India	27.5	84.4	1008.9	19-03-23 11:15:00.000000000 AM	72	Humid day
17	17	Bangkok	Thailand	32.7	75.9	1006.9	19-03-23 12:50:00.000000000 PM	70	Humid with scattered clouds
18	18	Moscow	Russia	15.9	58.3	1015.5	19-03-23 1:05:00.000000000 AM	50	Partly cloudy
19	19	Sydney	Australia	24.8	50.1	1013.3	19-03-23 1:15:00.000000000 AM	65	Sunny eith a light breeze
20	20	Berlin	Germany	18.2	61.2	1010.8	19-03-23 1:30:00.000000000 AM	46	Light showers

--CURSOR
-- There are two types of cursors explicit and implicit

--IMPLICIT CURSOR
-- Create an implicit cursor

```
BEGIN
--updating city id
UPDATE weather_data
SET message = 'Little showers'
WHERE country = 'India' and message = 'Mostly Cloudy';
--using implicit cursor attributes to find particular city name
IF SQL%FOUND THEN DBMS_OUTPUT.PUT_LINE('City name Found ');  --print if city name found
END IF;
IF SQL%NOTFOUND THEN
DBMS_OUTPUT.PUT_LINE('No City name Found ');  --print if city name not found
END IF;
IF SQL%ROWCOUNT > 0 THEN
DBMS_OUTPUT.PUT_LINE ('Number of record updated: '||SQL%ROWCOUNT);  --print how many record updated
ELSE
DBMS_OUTPUT.PUT_LINE('No updation'); --print if no row updated
END IF;
END;
/
```



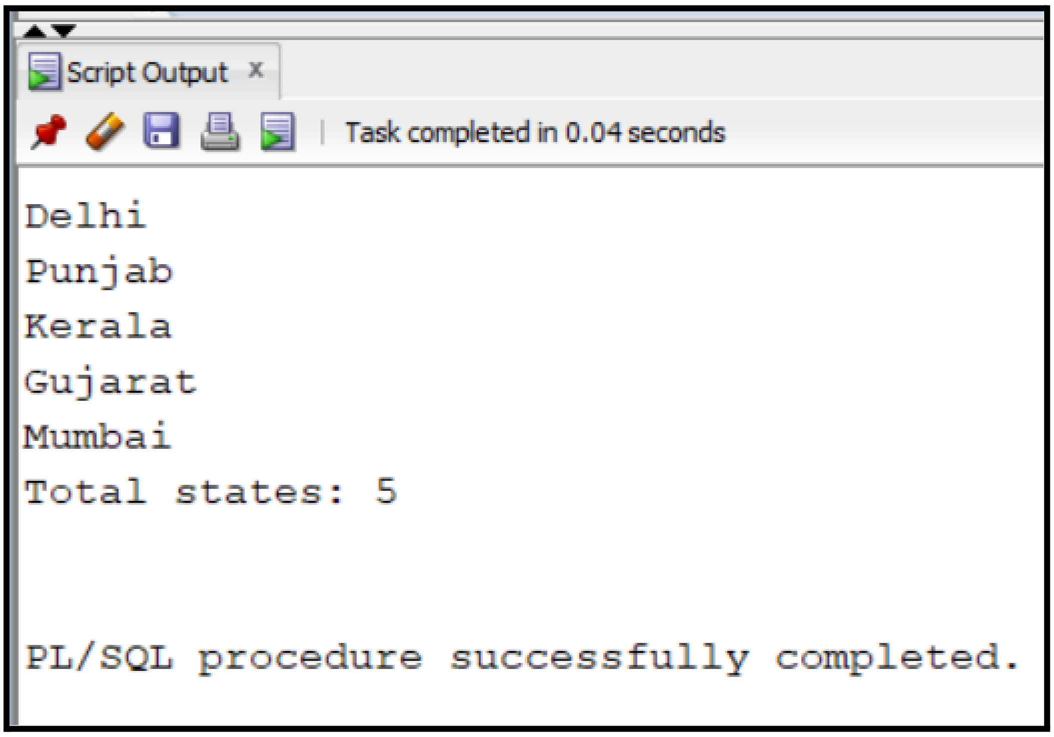
```
1 row inserted.

City name Found
Number of record updated: 1


PL/SQL procedure successfully completed.
```

-- Create an explicit cursor

--EXPLICIT CURSOR

```
SET SERVEROUTPUT ON;
DECLARE
-- declare a variable to hold the name of each state in the cursor
c_state_name weather_data.state_name%TYPE;
-- declare a cursor that will select all states in India
CURSOR indian_states IS
SELECT state_name FROM weather_data
WHERE country = 'India';
-- declare a variable to hold the name of each state in the cursor
BEGIN
-- open the cursor
OPEN indian_states;
-- loop through each row in the cursor
LOOP
-- fetch the next row and store the state name in the variable
FETCH indian_states INTO c_state_name;
-- exit the loop if there are no more rows to fetch
EXIT WHEN indian_states%NOTFOUND;
-- print the name of the state
DBMS_OUTPUT.PUT_LINE(c_state_name);
END LOOP;
-- print the total number of states in the table using an implicit cursor attribute
DBMS_OUTPUT.PUT_LINE('Total states: ' || indian_states%ROWCOUNT);
```

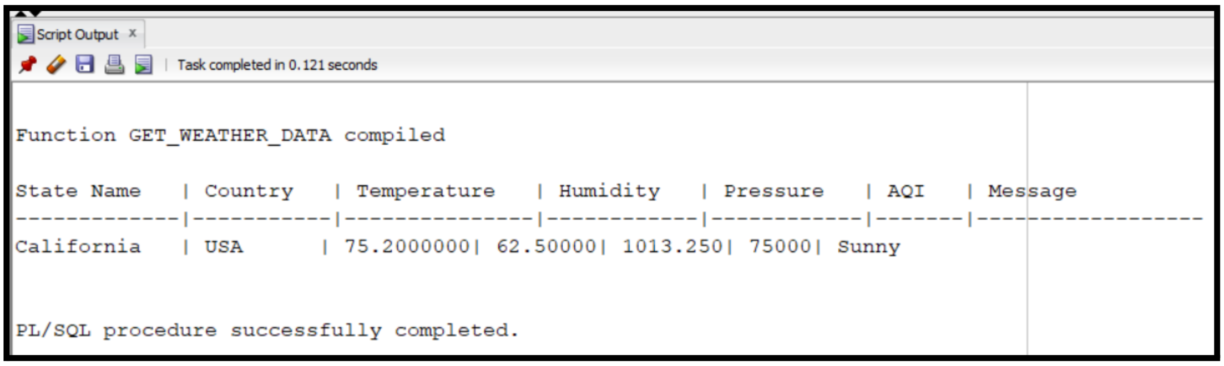
```
-- close the cursor
CLOSE indian_states;
END;
/
```



```
Delhi
Punjab
Kerala
Gujarat
Mumbai
Total states: 5


PL/SQL procedure successfully completed.
```

--FUNCTION
-- Create a PL/SQL function to retrieve weather data based on state name and country.

```
CREATE OR REPLACE FUNCTION get_weather_data
(p_state_name IN VARCHAR2,
p_country IN VARCHAR2)
RETURN SYS_REFCURSOR AS
-- Declare a cursor to store the query result
c_weather_data SYS_REFCURSOR;
BEGIN
-- Open the cursor with a SELECT statement that retrieves weather data
OPEN c_weather_data FOR
SELECT * FROM weather_data
WHERE state_name = p_state_name
AND country = p_country;
-- Return the cursor
RETURN c_weather_data;
END;
/
-- Usage of the function with an example
-- Declare a variable to store the cursor
DECLARE
c_weather_data SYS_REFCURSOR;
v_state_id NUMBER(10);
v_temperature NUMBER(5,2);
v_humidity NUMBER(5,2);
v_pressure NUMBER(7,2);
v_aqi NUMBER(5);
v_timestamp TIMESTAMP;
v_message VARCHAR2(200);
v_state_name VARCHAR2(50) := 'California';
v_country VARCHAR2(50) := 'USA';
BEGIN
-- Call the function to retrieve weather data for California in the USA
c_weather_data := get_weather_data(v_state_name, v_country);
-- Display the results
DBMS_OUTPUT.PUT_LINE('State Name   | Country   | Temperature   | Humidity   | Pressure
| AQI   | Message');
DBMS_OUTPUT.PUT_LINE('-------------|-----------|---------------|------------|------------|-------|-------
-----------');
LOOP
FETCH c_weather_data INTO
v_state_id,
v_state_name,
v_country,
v_temperature,
v_humidity,
v_pressure,
v_timestamp,
v_aqi,
v_message;
EXIT WHEN c_weather_data%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(
      RPAD(v_state_name, 13) || '| ' ||
      RPAD(v_country, 9) || '| ' ||
      RPAD(v_temperature, 10, '0') || '| ' ||
      RPAD(v_humidity, 8, '0') || '| ' ||
      RPAD(v_pressure, 8, '0') || '| ' ||
      RPAD(v_aqi, 5, '0') || '| ' ||
      v_message
   );
END LOOP;
CLOSE c_weather_data;
END;
/
```


Script Output ×

Task completed in 0.121 seconds

```
Function GET_WEATHER_DATA compiled

State Name    | Country    | Temperature    | Humidity    | Pressure    | AQI    | Message
-------------|-----------|---------------|------------|------------|-------|------------------
California    | USA       | 75.2000000| 62.50000| 1013.250| 75000| Sunny


PL/SQL procedure successfully completed.
```

-- The function can be used to retrieve weather data for any state in any country by passing the appropriate parameters.

--PROCEDURE

/* Create a PL/SQL procedure to retrieve weather data
from the "weather_data" table for a specific state
and display it in a formatted manner. */

-- PROCEDURE - IN MODE

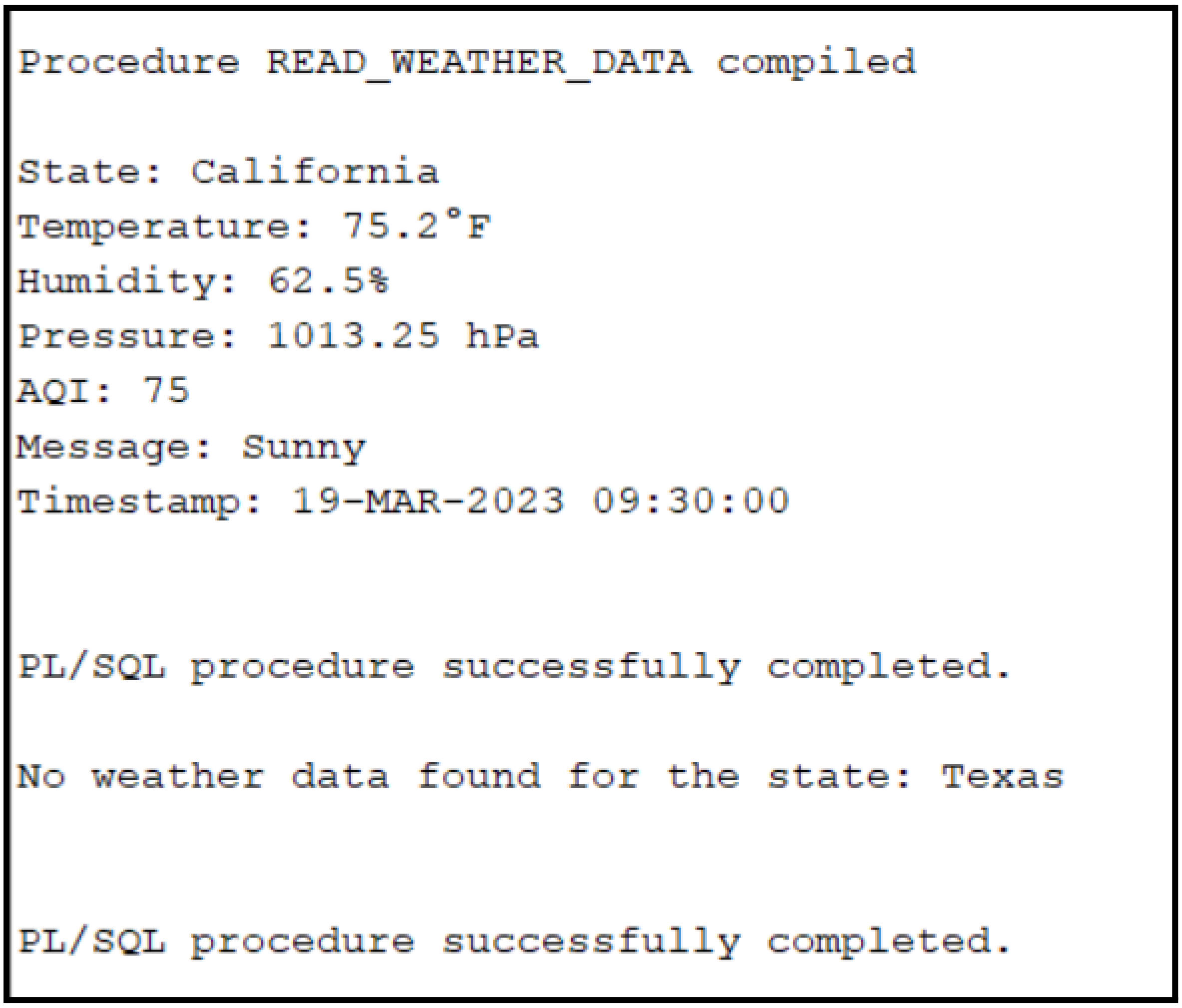
```
CREATE OR REPLACE PROCEDURE read_weather_data(p_state_name IN VARCHAR2) IS
v_state_name weather_data.state_name%TYPE := p_state_name;
v_temperature weather_data.temperature%TYPE;
v_humidity weather_data.humidity%TYPE;
v_pressure weather_data.pressure%TYPE;
v_aqi weather_data.AQI%TYPE;
v_message weather_data.message%TYPE;
v_timestamp weather_data.timestamp%TYPE;
BEGIN
-- Select weather data for the given state name
SELECT temperature, humidity, pressure, AQI, message, timestamp
INTO v_temperature, v_humidity, v_pressure, v_aqi, v_message, v_timestamp
FROM weather_data
WHERE state_name = v_state_name;

-- Display the weather data in a formatted manner
DBMS_OUTPUT.PUT_LINE('State: ' || v_state_name);
DBMS_OUTPUT.PUT_LINE('Temperature: ' || v_temperature || '°F');
DBMS_OUTPUT.PUT_LINE('Humidity: ' || v_humidity || '%');
DBMS_OUTPUT.PUT_LINE('Pressure: ' || v_pressure || ' hPa');
DBMS_OUTPUT.PUT_LINE('AQI: ' || v_aqi);
DBMS_OUTPUT.PUT_LINE('Message: ' || v_message);
DBMS_OUTPUT.PUT_LINE('Timestamp: ' || TO_CHAR(v_timestamp, 'DD-MON-YYYY
HH24:MI:SS'));

EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('No weather data found for the state: ' || v_state_name);
END;
/
```

-- Procedure created successfully
-- Call the procedure to retrieve weather data for a state

```
BEGIN
read_weather_data('California');
END;
/
```

-- Call the procedure for a state which does not exist in the table

```
BEGIN
read_weather_data('Texas');
END;
/
```



```
Procedure READ_WEATHER_DATA compiled

State: California
Temperature: 75.2°F
Humidity: 62.5%
Pressure: 1013.25 hPa
AQI: 75
Message: Sunny
Timestamp: 19-MAR-2023 09:30:00


PL/SQL procedure successfully completed.

No weather data found for the state: Texas


PL/SQL procedure successfully completed.
```

--TRIGGER
/*trigger that will be triggered before an insert into the comment_update table.
The trigger will insert a comment into the message column based on the weather conditions.
***/**

– Create a table comment_update

```
create TABLE comment_update
(
state_id NUMBER(10),
state_name VARCHAR2(50),
country VARCHAR2(50),
temperature NUMBER(5,2),
humidity NUMBER(5,2),
pressure NUMBER(7,2),
timestamp TIMESTAMP,
AQI NUMBER(5),
new_message VARCHAR2(200)
);
```

– Create trigger

```
CREATE OR REPLACE TRIGGER weather_comment
before
INSERT OR UPDATE
ON comment_update
FOR EACH ROW
BEGIN
IF :NEW.temperature > 80 AND :NEW.humidity > 60 THEN
:NEW.new_message := 'Hot and Humid';
ELSIF :NEW.temperature < 50 AND :NEW.humidity < 40 THEN
:NEW.new_message := 'Cold and Dry';
ELSIF :NEW.AQI >= 70 THEN
:NEW.new_message := 'Poor Air Quality';
ELSE
:NEW.new_message := 'Weather conditions are normal';
END IF;
END;
/
```

– check trigger by inserting values

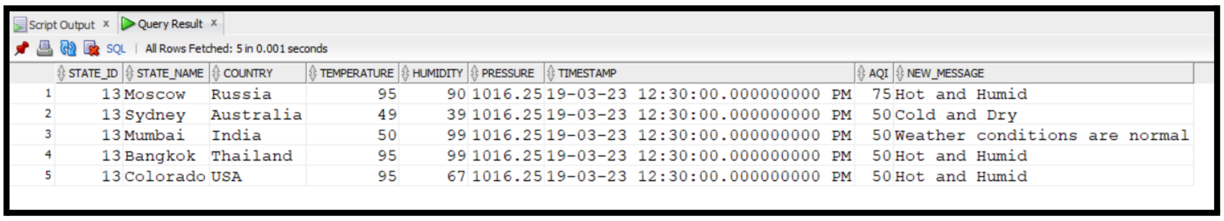
```
insert into comment_update values(13,'Moscow' , 'Russia', 95, 90 , 1016.25, TIMESTAMP
'2023-03-19 12:30:00', 75, ' ');
insert into comment_update values(13, 'Sydney' , 'Australia', 49, 39 , 1016.25, TIMESTAMP
'2023-03-19 12:30:00', 50, ' ');
insert into comment_update values(13,'Mumbai', 'India', 50, 99 , 1016.25, TIMESTAMP
'2023-03-19 12:30:00', 50, ' ');
```

insert into comment_update values(13, 'Bangkok', 'Thailand', 95, 99 , 1016.25, TIMESTAMP '2023-03-19 12:30:00', 50, ' ');
insert into comment_update values(13, 'Colorado', 'USA', 95, 99 , 1016.25, TIMESTAMP '2023-03-19 12:30:00', 50, ' ');

– check trigger by updating values

update comment_update
set humidity = 67
where country = 'USA';

select * from comment_update;



Script Output × Query Result ×

SQL | All Rows Fetched: 5 in 0.001 seconds

	STATE_ID	STATE_NAME	COUNTRY	TEMPERATURE	HUMIDITY	PRESSURE	TIMESTAMP	AQI	NEW_MESSAGE
1	13	Moscow	Russia	95	90	1016.25	19-03-23 12:30:00.000000000 PM	75	Hot and Humid
2	13	Sydney	Australia	49	39	1016.25	19-03-23 12:30:00.000000000 PM	50	Cold and Dry
3	13	Mumbai	India	50	99	1016.25	19-03-23 12:30:00.000000000 PM	50	Weather conditions are normal
4	13	Bangkok	Thailand	95	99	1016.25	19-03-23 12:30:00.000000000 PM	50	Hot and Humid
5	13	Colorado	USA	95	67	1016.25	19-03-23 12:30:00.000000000 PM	50	Hot and Humid

/*
This trigger will add a comment to the message column based on the temperature, humidity, and AQI values inserted into the weather_data table. The comments are as follows:
If the temperature is > 80 degrees and the humidity is greater than 60%, the message will be 'Hot and Humid'.
If the temperature is < 50 degrees and the humidity is less than 40%, the message will be 'Cold and Dry'.
If the AQI is >= 70, the message will be 'Poor Air Quality'.
If none of the above conditions are met, the message will be 'Weather conditions are normal'.
*/

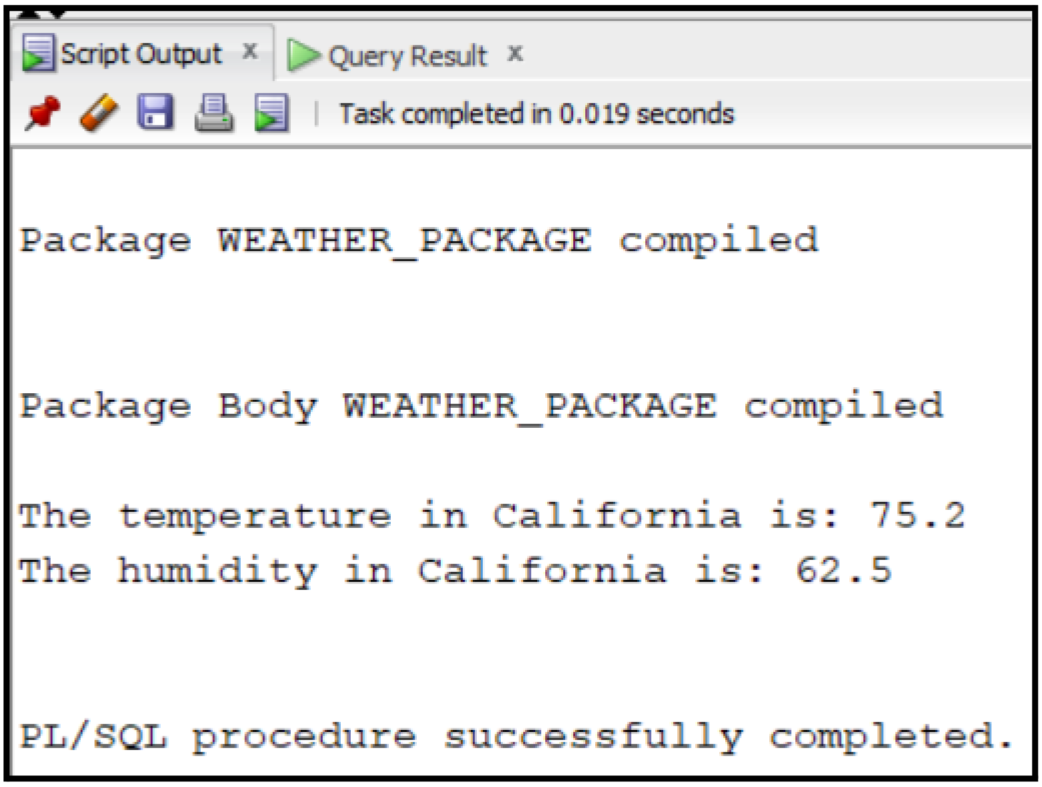
--PACKAGE

– Create a Package name weather_package

```
CREATE OR REPLACE PACKAGE weather_package AS
  FUNCTION get_temperature(city_name IN VARCHAR2) RETURN NUMBER;
  PROCEDURE get_humidity(city_name IN VARCHAR2, humidity OUT NUMBER);
END weather_package;
/
```

– Create package body weather_package
CREATE OR REPLACE PACKAGE BODY weather_package AS

```
  FUNCTION get_temperature(city_name IN VARCHAR2) RETURN NUMBER AS
   temperature NUMBER;
  BEGIN
   SELECT temperature INTO temperature
   FROM weather_data
   WHERE state_name = city_name;

   RETURN temperature;
  END;

  PROCEDURE get_humidity(city_name IN VARCHAR2, humidity OUT NUMBER) AS
  BEGIN
   SELECT humidity INTO humidity
   FROM weather_data
   WHERE state_name = city_name;
  END;
END weather_package;
/

– call package
DECLARE
  temp NUMBER;
  hum NUMBER;
BEGIN
  temp := weather_package.get_temperature('California');
  DBMS_OUTPUT.PUT_LINE('The temperature in California is: ' || temp);

  weather_package.get_humidity('California', hum);
  DBMS_OUTPUT.PUT_LINE('The humidity in California is: ' || hum);
END;
/
```

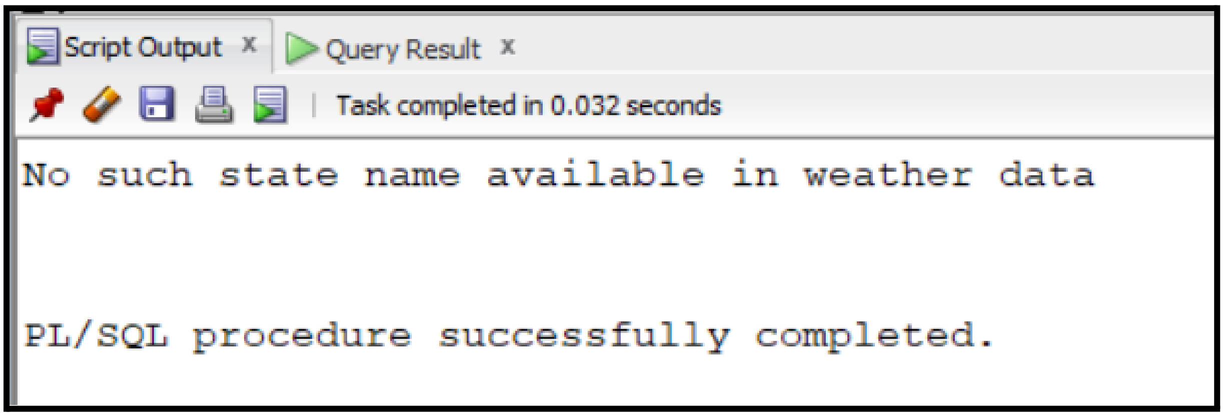


```
Package WEATHER_PACKAGE compiled


Package Body WEATHER_PACKAGE compiled

The temperature in California is: 75.2
The humidity in California is: 62.5


PL/SQL procedure successfully completed.
```

/* The above program is created to get the temperature and humidity of the state california */

– EXCEPTION HANDLING

```
DECLARE
exp_state_id weather_data.state_id%TYPE;
exp_state_name weather_data.state_name%TYPE := 'Texas';
exp_country weather_data.country%TYPE;
exp_message weather_data.message%TYPE;
BEGIN
SELECT state_id, state_name, country, message
INTO exp_state_id, exp_state_name, exp_country, exp_message
FROM weather_data
WHERE state_name = exp_state_name;
DBMS_OUTPUT.PUT_LINE('State ID: ' || exp_state_id);
DBMS_OUTPUT.PUT_LINE('State Name: ' || exp_state_name);
DBMS_OUTPUT.PUT_LINE('Country Name: '|| exp_country);
DBMS_OUTPUT.PUT_LINE('Weather Message: ' || exp_message);

EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('No such state name available in weather data');
WHEN others THEN
DBMS_OUTPUT.PUT_LINE('Error!');
END;
/
```



```
No such state name available in weather data


PL/SQL procedure successfully completed.
```

/*
The above program displays the state id, state name, country name and weather message from weather data.
Since there is no state name with Texas in our database,
The program raises the run-time exception NO_DATA_FOUND, which is captured in the EXCEPTION block.
*/