--QUESTION1--

--CREATE DATABASE--

CREATE DATABASE ADC;

--CREATE TABLES--

CREATE TABLE Customers (

cust\_id VARCHAR2(20) PRIMARY KEY,

cust\_fname VARCHAR2(50) NOT NULL,

cust\_sname VARCHAR2(50) NOT NULL,

cust\_address VARCHAR2(100) NOT NULL,

cust\_contact VARCHAR2(20) NOT NULL

);

CREATE TABLE Dive (

dive\_id NUMBER PRIMARY KEY,

dive\_name VARCHAR2(50) NOT NULL,

dive\_duration VARCHAR2(50) NOT NULL,

dive\_location VARCHAR2(50) NOT NULL,

dive\_exp\_level NUMBER NOT NULL,

dive\_cost NUMBER NOT NULL

);

CREATE TABLE Instructors (

ins\_id NUMBER PRIMARY KEY,

ins\_fname VARCHAR2(50) NOT NULL,

ins\_sname VARCHAR2(50) NOT NULL,

ins\_contact VARCHAR2(20) NOT NULL,

ins\_level NUMBER NOT NULL

);

CREATE TABLE DiveEvent (

dive\_event\_id VARCHAR2(10) PRIMARY KEY,

dive\_date DATE NOT NULL,

dive\_participants NUMBER NOT NULL CHECK (dive\_participants > 0 AND dive\_participants <= 20),

dive\_id NUMBER NOT NULL,

cust\_id VARCHAR2(20) NOT NULL,

ins\_id NUMBER NOT NULL,

FOREIGN KEY (dive\_id) REFERENCES Dive(dive\_id),

FOREIGN KEY (cust\_id) REFERENCES Customers(cust\_id),

FOREIGN KEY (ins\_id) REFERENCES Instructors(ins\_id)

);

--INSERT INTO TABLES--

--Insert into Instructors table

INSERT ALL

INTO Instructors (ins\_ID, ins\_fname, ins\_sname, ins\_contact, ins\_level) VALUES (101, 'James', 'Willis', '0843569851', 7)

INTO Instructors (ins\_ID, ins\_fname, ins\_sname, ins\_contact, ins\_level) VALUES (102, 'Sam', 'Wait', '0763698521', 2)

INTO Instructors (ins\_ID, ins\_fname, ins\_sname, ins\_contact, ins\_level) VALUES (103, 'Sally', 'Gumede', '0786598521', 8)

INTO Instructors (ins\_ID, ins\_fname, ins\_sname, ins\_contact, ins\_level) VALUES (104, 'Bob', 'Du Preez', '0796369857', 3)

INTO Instructors (ins\_ID, ins\_fname, ins\_sname, ins\_contact, ins\_level) VALUES (105, 'Simon', 'Jones', '0826598741', 9)

SELECT \* FROM dual;

-- Insert into Customers table

INSERT ALL

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C115', 'Heinrich', 'Willis', '3 Main Road', '0821253659')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C116', 'David', 'Watson', '13 Cape Road', '0769658547')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C117', 'Waldo', 'Smith', '3 Mountain Road', '0863256574')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C118', 'Alex', 'Hanson', '8 Circle Road', '0762356587')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C119', 'Kuhle', 'Bitterhout', '15 Main Road', '0821235258')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C120', 'Thando', 'Zolani', '88 Summer Road', '0847541254')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C121', 'Philip', 'Jackson', '3 Long Road', '0745556658')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C122', 'Sarah', 'Jones', '7 Sea Road', '0814745745')

INTO Customers (cust\_id, cust\_fname, cust\_sname, cust\_address, cust\_contact) VALUES ('C123', 'Catherine', 'Howard', '31 Lake Side Road', '0822232521')

SELECT \* FROM dual;

-- Insert into Dive table

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (550, 'Shark Dive', '3 hours', 'Shark Point', 8, 500);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (551, 'Coral Dive', '1 hour', 'Break Point', 7, 300);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (552, 'Wave', '2 hours', 'Ship wreck ally', 3, 800);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (553, 'Underwater Exploration', '1 hour', 'Coral ally', 2, 250);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (554, 'Underwater Adventure', '3 hours', 'Sandy Beach', 3, 750);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (555, 'Deep Blue Ocean', '30 minutes', 'Lazy Waves', 2, 120);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (556, 'Rough Seas', '1 hour', 'Pipe', 9, 700);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (557, 'White Water', '2 hours', 'Drifts', 5, 200);

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (558, 'Current Adventure', '2 hours', 'Rock Lands', 3, 150);

SELECT \* FROM dual;

-- Insert into DiveEvents table

INSERT ALL

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_101', TO\_DATE('15-Jul-17', 'DD-Mon-YY'), 5, 103, 'C115', '558')

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_102', TO\_DATE('16-Jul-17', 'DD-Mon-YY'), 7, 102, 'C117', '555')

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_103', TO\_DATE('18-Jul-17', 'DD-Mon-YY'), 8, 104, 'C118', '552')

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_104', TO\_DATE('19-Jul-17', 'DD-Mon-YY'), 3, 101, 'C119', '551')

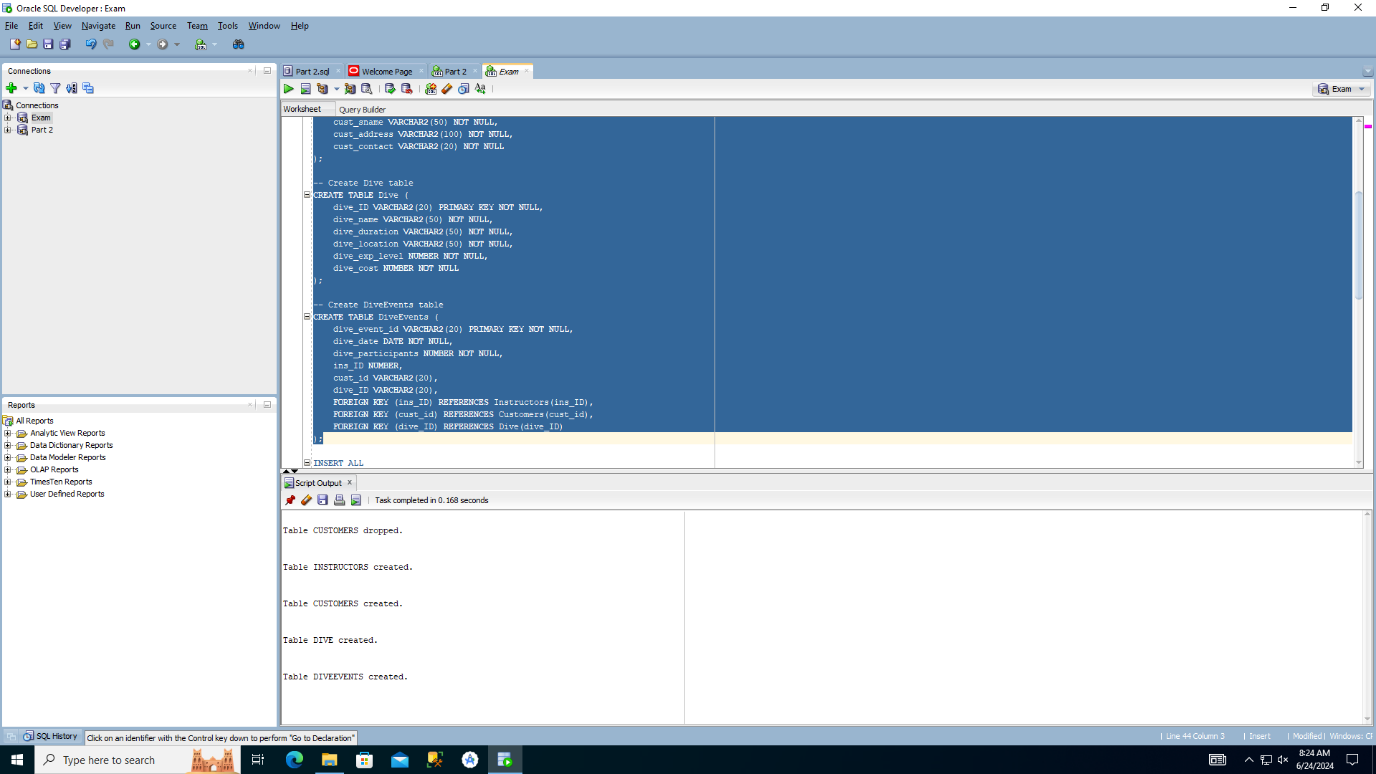
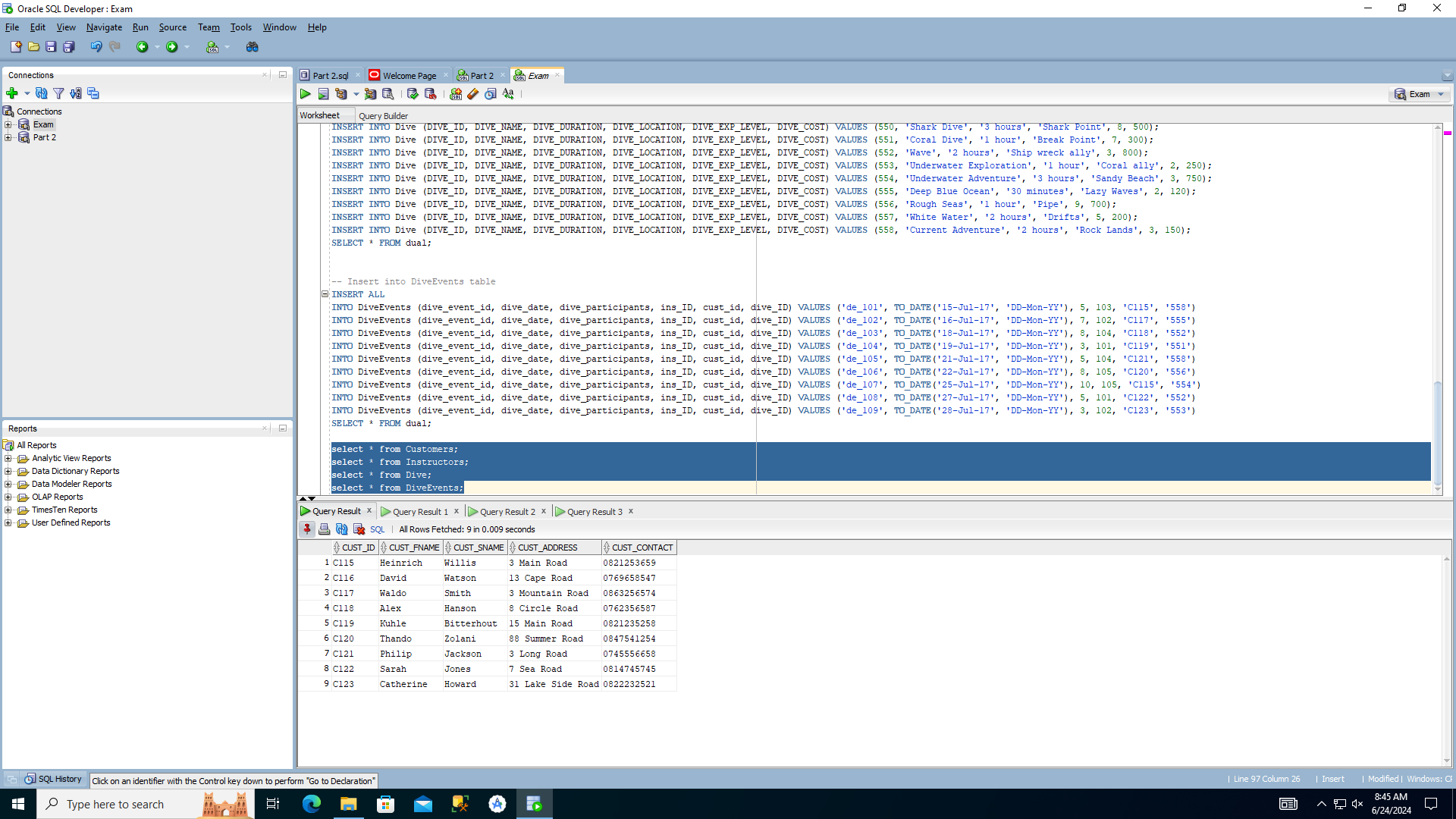
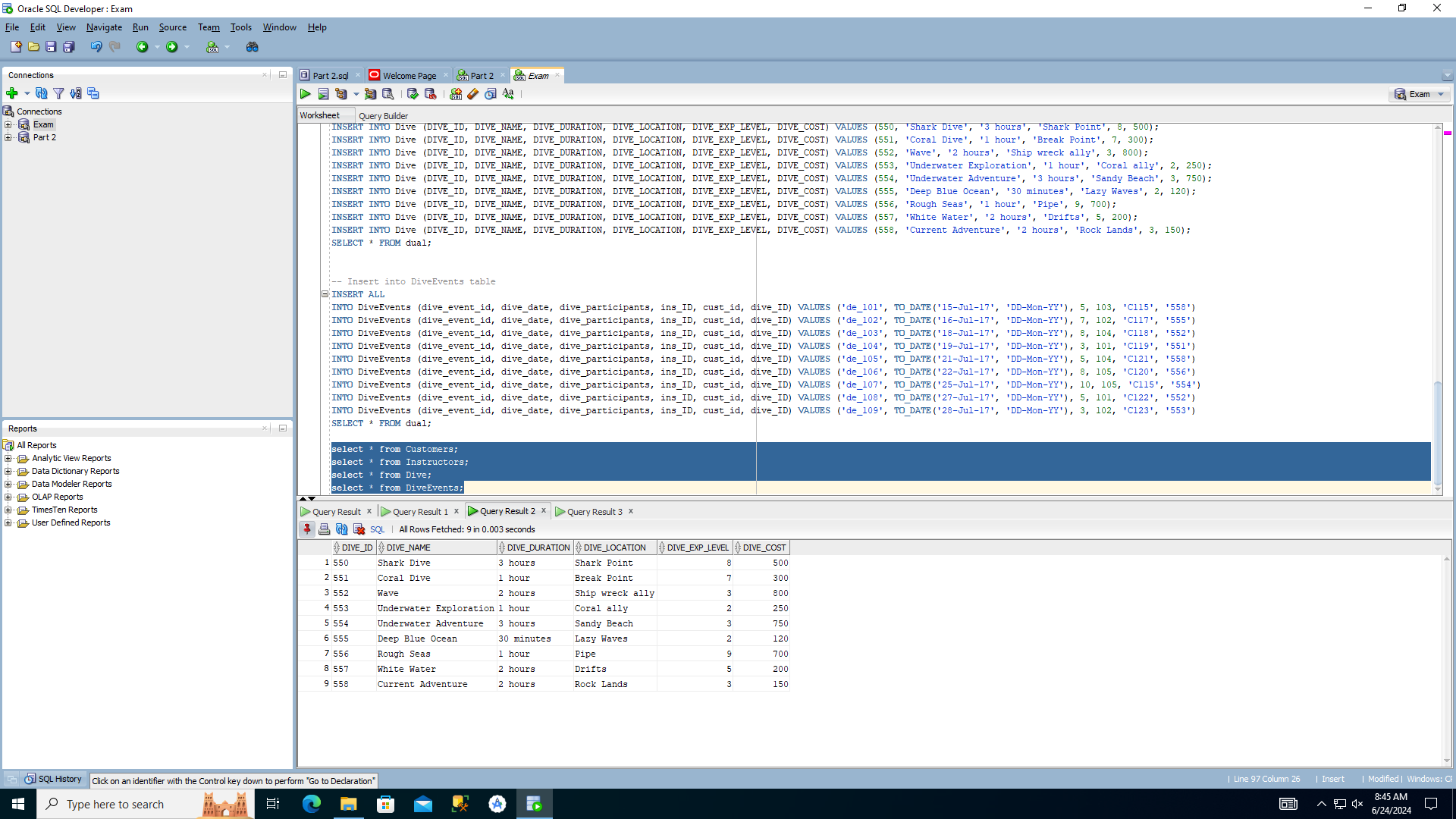
INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_105', TO\_DATE('21-Jul-17', 'DD-Mon-YY'), 5, 104, 'C121', '558')

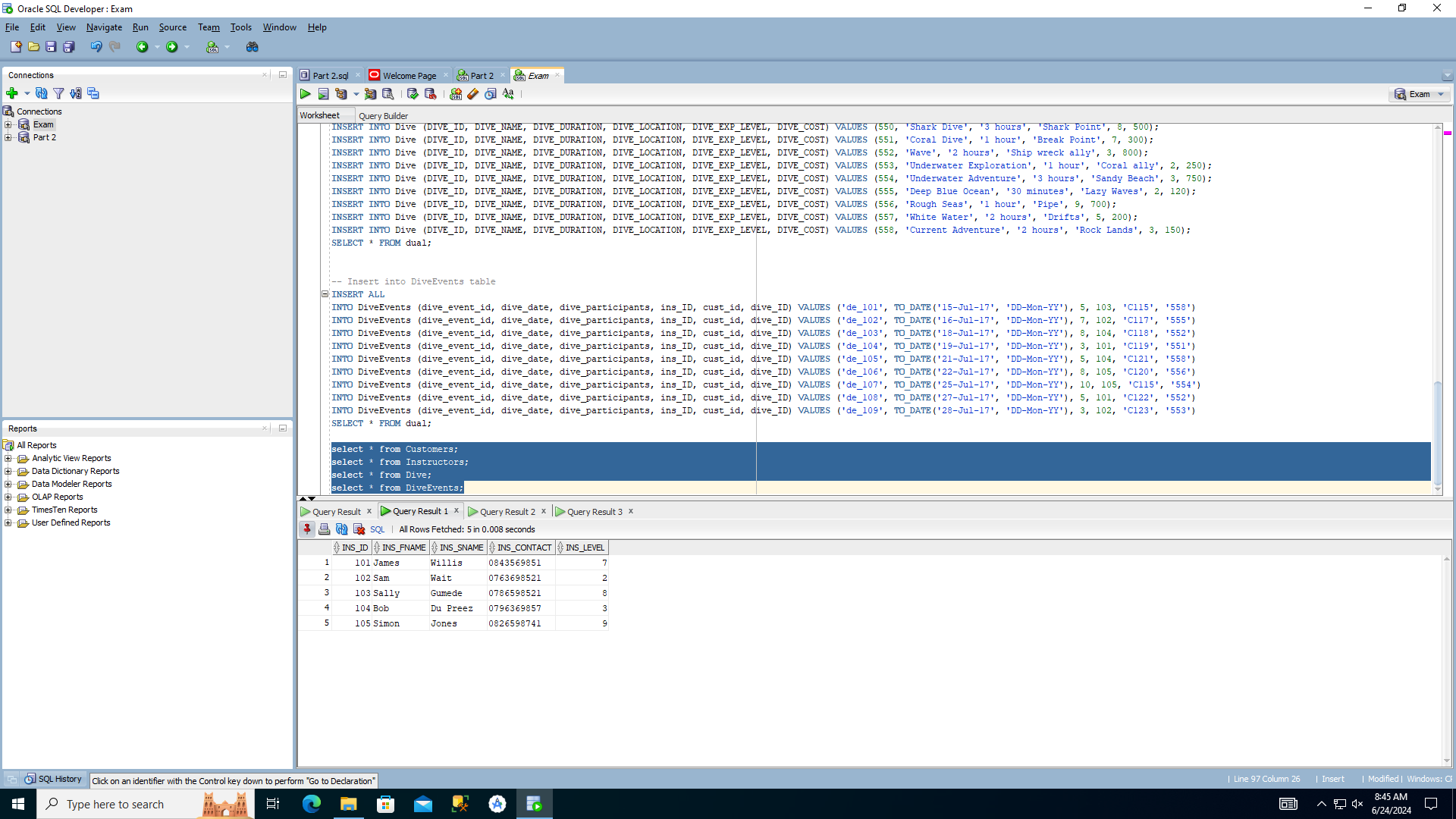
INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_106', TO\_DATE('22-Jul-17', 'DD-Mon-YY'), 8, 105, 'C120', '556')

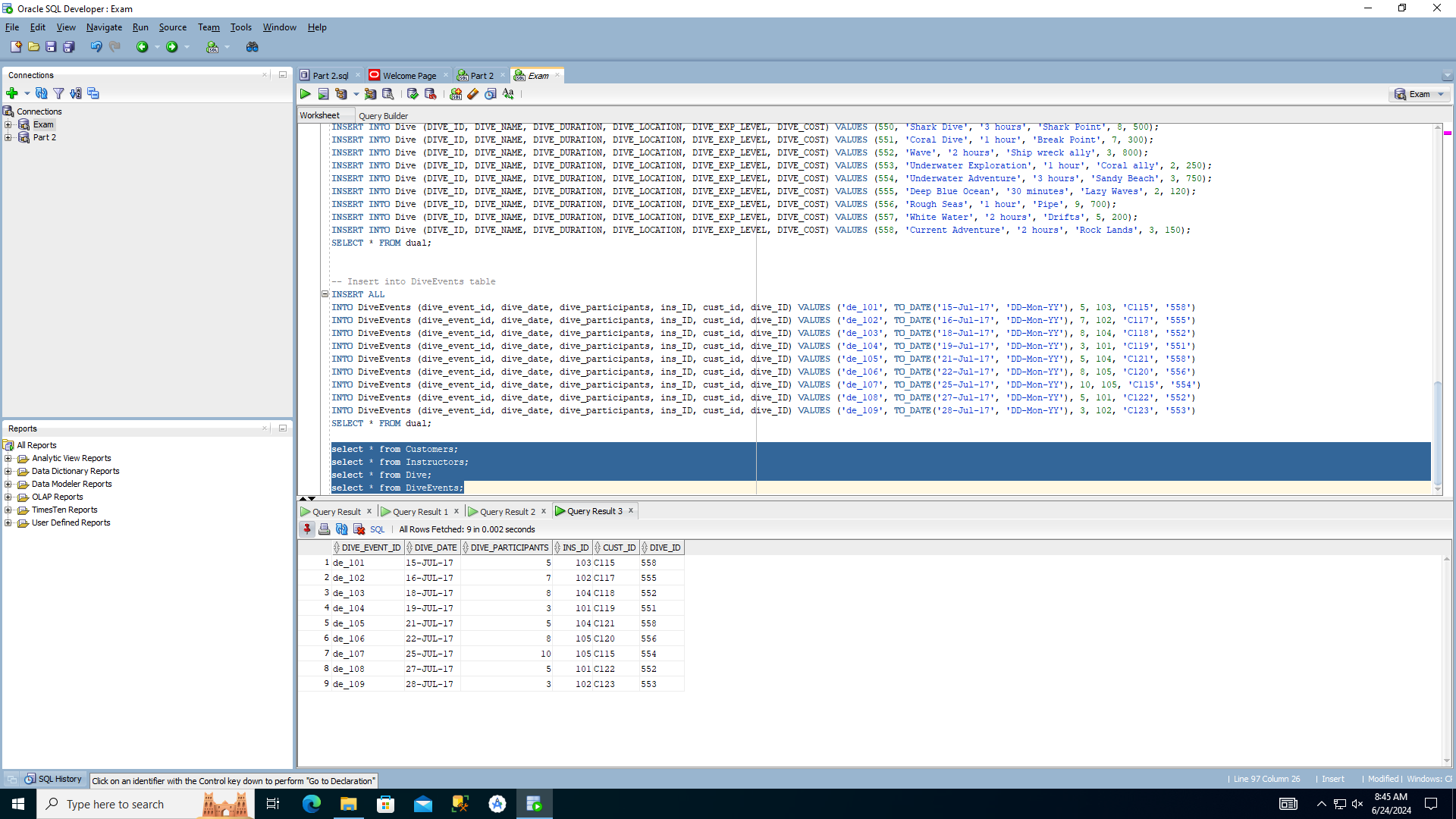
INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_107', TO\_DATE('25-Jul-17', 'DD-Mon-YY'), 10, 105, 'C115', '554')

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_108', TO\_DATE('27-Jul-17', 'DD-Mon-YY'), 5, 101, 'C122', '552')

INTO DiveEvents (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID) VALUES ('de\_109', TO\_DATE('28-Jul-17', 'DD-Mon-YY'), 3, 102, 'C123', '553')

SELECT \* FROM dual;





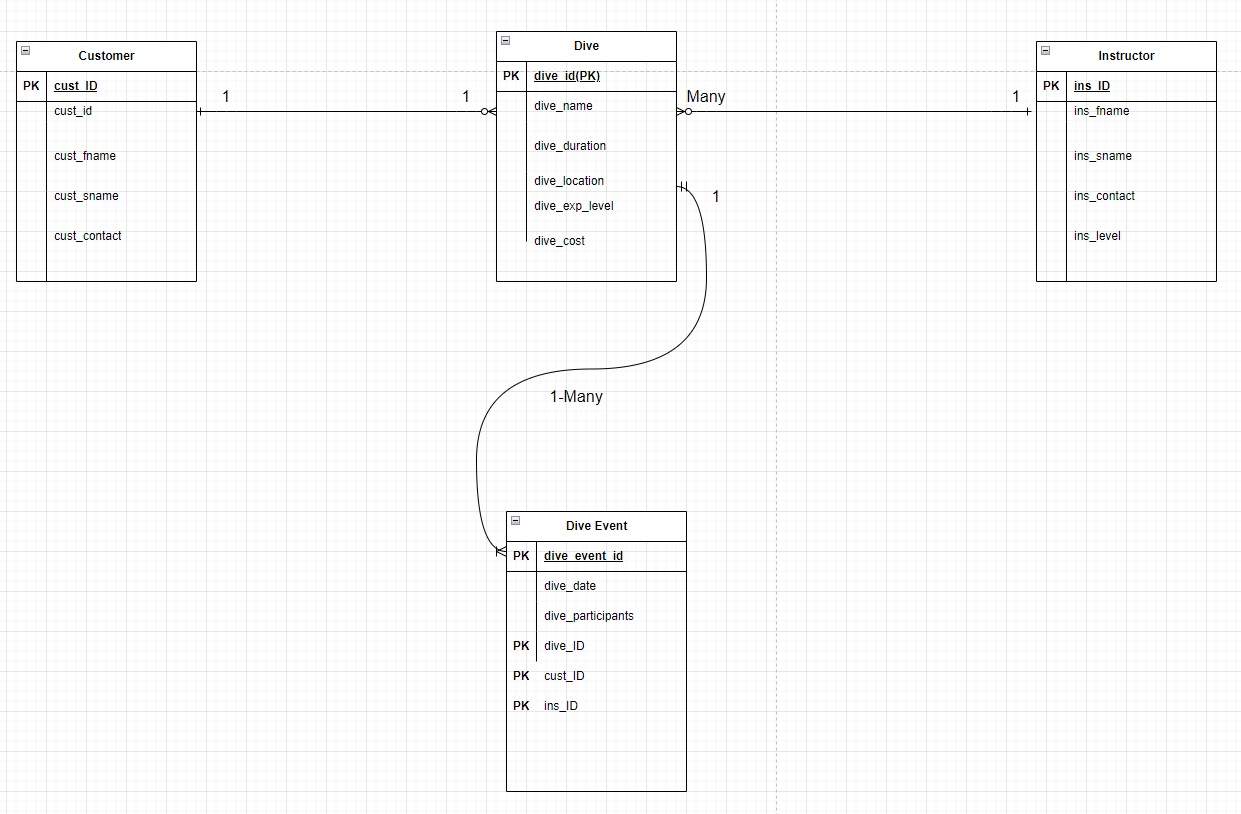
select \* from Customers;

select \* from Instructors;

select \* from Dive;

select \* from DiveEvents;

--ENTITY RELATIONSHIP DIAGRAM—



--QUESTION 2—

-- Creating the admin role with elevated privileges

CREATE ROLE admin\_role;

-- Creating the general user role with limited privileges

CREATE ROLE general\_user\_role;

-- Granting privileges to admin\_role

GRANT CREATE SESSION TO admin\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Dive TO admin\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Customers TO admin\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Instructors TO admin\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON DiveEvent TO admin\_role;

GRANT CREATE, ALTER, DROP TABLE TO admin\_role;

GRANT CREATE USER, DROP USER, GRANT ANY PRIVILEGE, REVOKE ANY PRIVILEGE TO admin\_role;

-- Granting privileges to general\_user\_role

GRANT CREATE SESSION TO general\_user\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Dive TO general\_user\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Customers TO general\_user\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON Instructors TO general\_user\_role;

GRANT SELECT, INSERT, UPDATE, DELETE ON DiveEvent TO general\_user\_role;

-- Creating the admin user and assigning the admin role

CREATE USER admin\_user IDENTIFIED BY password;

GRANT admin\_role TO admin\_user;

-- Creating the general user and assigning the general user role

CREATE USER general\_user IDENTIFIED BY password;

GRANT general\_user\_role TO general\_user;

-- Enable server output to display the result

SET SERVEROUTPUT ON;

-- Connect as admin\_user

CONNECT admin\_user/password;

-- Verify privileges

SELECT \* FROM Dive;

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (559, 'New Dive', '2 hours', 'New Location', 4, 400);

CREATE TABLE TestTable (id NUMBER);

-- Connect as general\_user

CONNECT general\_user/password;

-- Verify privileges

SELECT \* FROM Dive;

INSERT INTO Dive (DIVE\_ID, DIVE\_NAME, DIVE\_DURATION, DIVE\_LOCATION, DIVE\_EXP\_LEVEL, DIVE\_COST) VALUES (560, 'Another Dive', '1 hour', 'Another Location', 3, 300);

-- The following should fail

CREATE TABLE AnotherTestTable (id NUMBER);

--QUESTION 3—

SELECT

i.ins\_fname || ' ' || i.ins\_sname AS instructor\_name,

c.cust\_fname || ' ' || c.cust\_sname AS customer\_name,

d.dive\_location,

de.dive\_participants

FROM

DiveEvent de

JOIN Instructors i ON de.ins\_ID = i.ins\_ID

JOIN Customers c ON de.cust\_id = c.cust\_id

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

WHERE

de.dive\_participants BETWEEN 8 AND 10;

A white rectangular object with a white border

Description automatically generated

--QUESTION 4—

SET SERVEROUTPUT ON;

DECLARE

-- Declare a cursor to select dive name and dive date for dive events with 10 or more participants

CURSOR dive\_event\_cursor IS

SELECT d.dive\_name, de.dive\_date

FROM DiveEvent de

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

WHERE de.dive\_participants >= 10;

-- Variables to hold the fetched data

v\_dive\_name Dive.DIVE\_NAME%TYPE;

v\_dive\_date DiveEvent.dive\_date%TYPE;

BEGIN

-- Open the cursor

OPEN dive\_event\_cursor;

-- Loop through the cursor and fetch each row

LOOP

FETCH dive\_event\_cursor INTO v\_dive\_name, v\_dive\_date;

-- Exit the loop when no more rows are found

EXIT WHEN dive\_event\_cursor%NOTFOUND;

-- Display the dive name and dive date

DBMS\_OUTPUT.PUT\_LINE('Dive Name: ' || v\_dive\_name || ', Dive Date: ' || TO\_CHAR(v\_dive\_date, 'DD-Mon-YYYY'));

END LOOP;

-- Close the cursor

CLOSE dive\_event\_cursor;

END;

/

A screenshot of a computer

Description automatically generated

--QUESTION 5—

SET SERVEROUTPUT ON;

DECLARE

CURSOR dive\_cursor IS

SELECT c.cust\_fname || ' ' || c.cust\_sname AS customer\_name,

d.dive\_name,

de.dive\_participants,

CEIL(de.dive\_participants / 5) AS instructors\_required

FROM DiveEvent de

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

JOIN Customers c ON de.cust\_id = c.cust\_id

WHERE d.dive\_cost > 500;

v\_customer\_name VARCHAR2(100);

v\_dive\_name VARCHAR2(100);

v\_dive\_participants NUMBER;

v\_instructors\_required NUMBER;

BEGIN

OPEN dive\_cursor;

DBMS\_OUTPUT.PUT\_LINE('Customer Name | Dive Name | Participants | Status');

DBMS\_OUTPUT.PUT\_LINE('-------------------------------------------------------------');

LOOP

FETCH dive\_cursor INTO v\_customer\_name, v\_dive\_name, v\_dive\_participants, v\_instructors\_required;

EXIT WHEN dive\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_customer\_name || ' | ' || v\_dive\_name || ' | ' || v\_dive\_participants || ' | ' || v\_instructors\_required);

END LOOP;

IF dive\_cursor%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('No records found.');

END IF;

IF dive\_cursor%ISOPEN THEN

CLOSE dive\_cursor;

END IF;

END;

/

A white background with black text

Description automatically generated

--QUESTION 6—

CREATE VIEW Vw\_Dive\_Event AS

SELECT

de.ins\_ID,

c.cust\_address,

d.dive\_duration

FROM

DiveEvent de

JOIN Customers c ON de.cust\_id = c.cust\_id

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

WHERE

de.dive\_date < TO\_DATE('19-Jul-2017', 'DD-Mon-YYYY');

SELECT \* FROM Vw\_Dive\_Event;

A white rectangular object with a white border

Description automatically generated

--QUESTION 7—

CREATE OR REPLACE TRIGGER New\_Dive\_Event

BEFORE INSERT OR UPDATE ON DiveEvent

FOR EACH ROW

BEGIN

IF :NEW.dive\_participants <= 0 OR :NEW.dive\_participants > 20 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Number of participants must be between 1 and 20.');

END IF;

END;

/

-- Test case: Insert with 0 participants (should fail)

BEGIN

INSERT INTO DiveEvent (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID)

VALUES ('de\_110', TO\_DATE('30-Jul-17', 'DD-Mon-YY'), 0, '101', 'C115', 558);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

-- Test case: Insert with 21 participants (should fail)

BEGIN

INSERT INTO DiveEvent (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID)

VALUES ('de\_111', TO\_DATE('31-Jul-17', 'DD-Mon-YY'), 21, '102', 'C116', 555);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

-- Test case: Insert with 5 participants (should succeed)

BEGIN

INSERT INTO DiveEvent (dive\_event\_id, dive\_date, dive\_participants, ins\_ID, cust\_id, dive\_ID)

VALUES ('de\_112', TO\_DATE('01-Aug-17', 'DD-Mon-YY'), 5, '103', 'C117', 554);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

-- Test case: Update to 0 participants (should fail)

BEGIN

UPDATE DiveEvent

SET dive\_participants = 0

WHERE dive\_event\_id = 'de\_112';

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

-- Test case: Update to 15 participants (should succeed)

BEGIN

UPDATE DiveEvent

SET dive\_participants = 15

WHERE dive\_event\_id = 'de\_112';

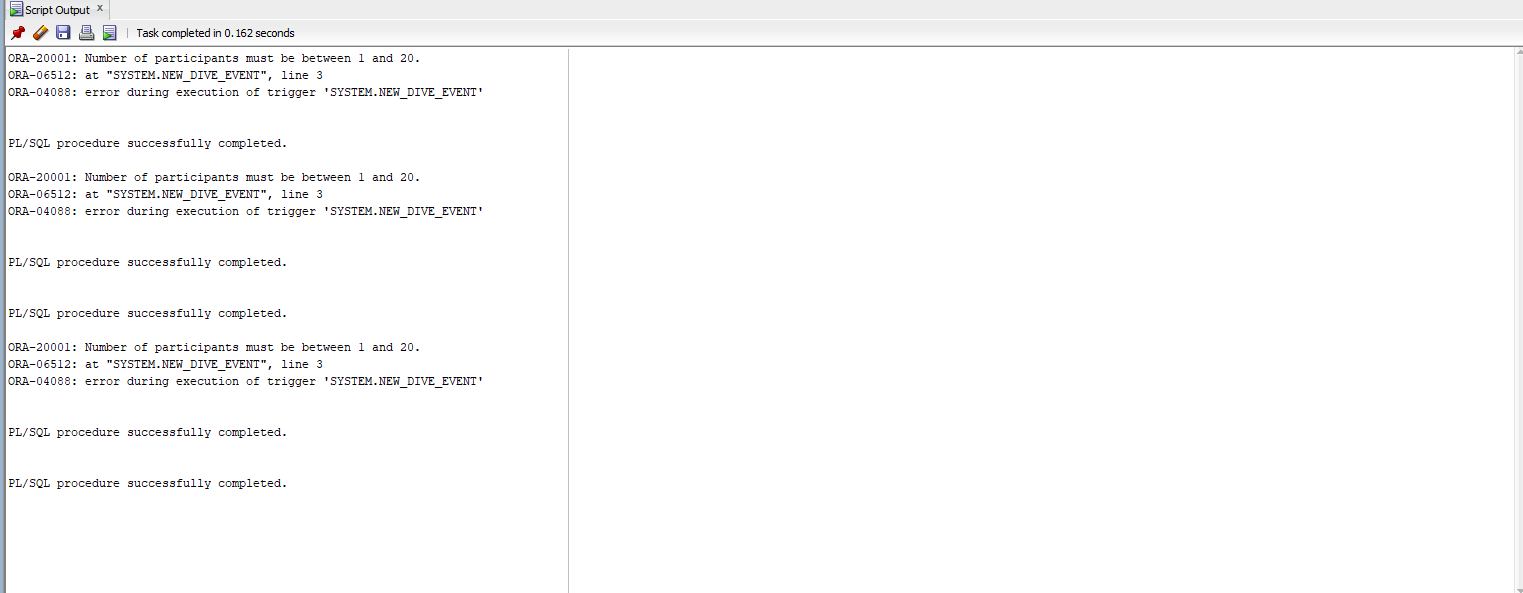
EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/



--QUESTION 8—

CREATE OR REPLACE PROCEDURE sp\_Customer\_Details (

p\_cust\_id IN Customers.cust\_id%TYPE,

p\_dive\_date IN DiveEvent.dive\_date%TYPE

)

IS

v\_customer\_name VARCHAR2(100);

v\_dive\_name VARCHAR2(100);

BEGIN

SELECT c.cust\_fname || ' ' || c.cust\_sname,

d.dive\_name

INTO v\_customer\_name, v\_dive\_name

FROM DiveEvent de

JOIN Customers c ON de.cust\_id = c.cust\_id

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

WHERE de.cust\_id = p\_cust\_id

AND de.dive\_date = p\_dive\_date;

DBMS\_OUTPUT.PUT\_LINE('Customer Name: ' || v\_customer\_name);

DBMS\_OUTPUT.PUT\_LINE('Dive Name: ' || v\_dive\_name);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No booking found for customer ID ' || p\_cust\_id || ' on date ' || TO\_CHAR(p\_dive\_date, 'DD-Mon-YYYY'));

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An unexpected error occurred: ' || SQLERRM);

END sp\_Customer\_Details;

/

-- Setting server output to display the result

SET SERVEROUTPUT ON;

-- Test case 1: Valid customer ID and dive date

BEGIN

sp\_Customer\_Details('C115', TO\_DATE('15-Jul-2017', 'DD-Mon-YYYY'));

END;

/

-- Test case 2: Valid customer ID but no booking on the given date

BEGIN

sp\_Customer\_Details('C115', TO\_DATE('01-Jan-2022', 'DD-Mon-YYYY'));

END;

/

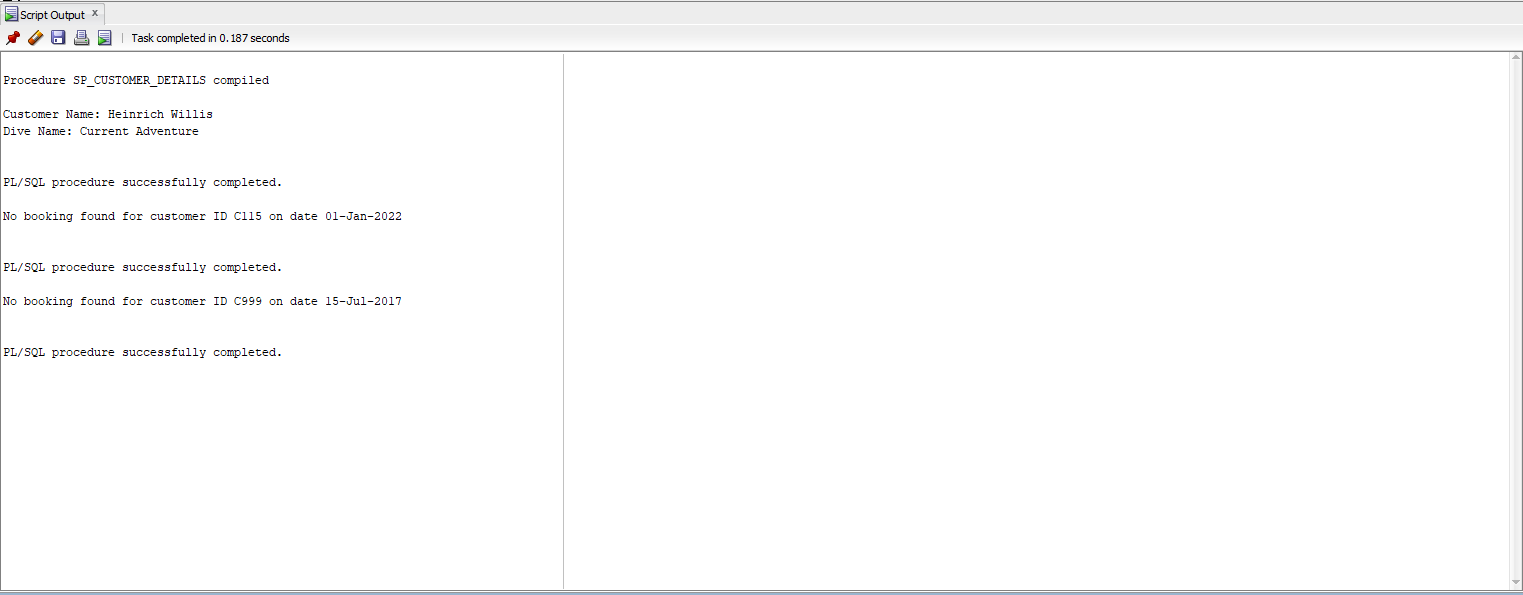
-- Test case 3: Invalid customer ID

BEGIN

sp\_Customer\_Details('C999', TO\_DATE('15-Jul-2017', 'DD-Mon-YYYY'));

END;

/



--QUESTION 9—

CREATE OR REPLACE FUNCTION fn\_Calculate\_Dive\_Cost (

p\_dive\_event\_id IN DiveEvent.dive\_event\_id%TYPE

) RETURN NUMBER

IS

v\_dive\_cost\_per\_participant NUMBER;

v\_dive\_participants NUMBER;

v\_total\_cost NUMBER;

BEGIN

-- Retrieve the cost per participant and the number of participants

SELECT d.dive\_cost, de.dive\_participants

INTO v\_dive\_cost\_per\_participant, v\_dive\_participants

FROM DiveEvent de

JOIN Dive d ON de.dive\_ID = d.DIVE\_ID

WHERE de.dive\_event\_id = p\_dive\_event\_id;

-- Calculate the total cost

v\_total\_cost := v\_dive\_cost\_per\_participant \* v\_dive\_participants;

-- Return the total cost

RETURN v\_total\_cost;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

-- Handle the case where no dive event is found for the given ID

RAISE\_APPLICATION\_ERROR(-20002, 'Dive event not found for ID: ' || p\_dive\_event\_id);

WHEN OTHERS THEN

-- Handle any other unexpected errors

RAISE\_APPLICATION\_ERROR(-20003, 'An unexpected error occurred: ' || SQLERRM);

END fn\_Calculate\_Dive\_Cost;

/

-- Test case 1: Valid dive event ID

DECLARE

v\_total\_cost NUMBER;

BEGIN

v\_total\_cost := fn\_Calculate\_Dive\_Cost('de\_101');

DBMS\_OUTPUT.PUT\_LINE('Total Cost for dive event de\_101: ' || v\_total\_cost);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

-- Test case 2: Invalid dive event ID

DECLARE

v\_total\_cost NUMBER;

BEGIN

v\_total\_cost := fn\_Calculate\_Dive\_Cost('de\_999');

DBMS\_OUTPUT.PUT\_LINE('Total Cost for dive event de\_999: ' || v\_total\_cost);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;

/

SET SERVEROUTPUT ON;

