

# DISTRIBUTED AND PARALLEL DATABASE SYSTEMS

## PROJECT PART – 5

### FUNCTIONS:

1. Function to return the most popular genre of a user.  
user\_most\_popular\_genre(u\_id) - It takes user id(u\_id) as input parameter and gives genre as output.

```
2  -- Function which the user look most.
3  CREATE OR REPLACE FUNCTION user_most_popular_genre(u_id IN NUMBER)
4  RETURN VARCHAR2
5  IS
6  popular_genre VARCHAR2(30);
7  BEGIN
8  SELECT b.Genre
9  INTO popular_genre
10 FROM books_transactions_India bt
11 JOIN books_India b ON bt.b_id = b.b_id
12 WHERE bt.u_id = u_id
13 GROUP BY b.Genre
14 ORDER BY COUNT(*) DESC
15 FETCH FIRST 1 ROW ONLY;
16
17 RETURN popular_genre;
18 END;
19 /
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Function USER\_MOST\_POPULAR\_GENRE compiled  
Elapsed: 00:00:00.185

Calling the function user\_most\_popular\_genre.

```
20
21 SELECT user_most_popular_genre(110) FROM dual;
22
```

Query Result   Script Output   DBMS Output   Explain Plan   Autotrace

Download   Execution time: 0.019 seconds

	USER_MOST_POPULAR_GENRE
1	Fiction

2. Function to get the total salary of all the employees in a library.

get\_total\_salary(l\_id) - It takes library ID(l\_id) as input parameter and returns total salary of employees working in that library as output.

```
82 |
83 | CREATE or replace FUNCTION get_total_salary (l_id IN NUMBER) RETURN NUMBER
84 | IS
85 |     total_salary NUMBER;
86 | BEGIN
87 |     SELECT SUM(salary)
88 |     INTO total_salary
89 |     FROM EMPLOYEES_USA
90 |     WHERE l_id = get_total_salary.l_id;
91 |
92 |     RETURN total_salary;
93 | END;
94 | /
95 |
96 | DECLARE
97 |     total_salary NUMBER;
98 | BEGIN
99 |     total_salary := get_total_salary(11);
100 |     DBMS_OUTPUT.PUT_LINE('Total salary: ' || total_salary);
101 | END;
```

Query Result   Script Output   DBMS Output   Explain Plan   Autotrace   SQL History



Elapsed: 00:00:00.007

Total salary: 72000

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.008

3. Function to get total number of employees working in the library at France.

get\_employee\_count\_by\_library(p\_library\_id) - It takes Library ID(p\_library\_id) as input parameter and returns total number of employees working for that library. We created this function for Libraries at France. In this function, we used NO\_DATA\_FOUND exception to handle if the given library ID is not found.

```

1  -- total number of employees working in specified library.
2
3  CREATE OR REPLACE FUNCTION get_employee_count_by_library (p_library_id IN NUMBER)
4  RETURN NUMBER
5  IS
6      v_count NUMBER;
7  BEGIN
8      SELECT COUNT(*) INTO v_count
9      FROM employees_France
10     WHERE l_id = p_library_id;
11
12     RETURN v_count;
13 EXCEPTION
14     WHEN NO_DATA_FOUND THEN
15         RAISE_APPLICATION_ERROR(-20001, 'Library ID ' || p_library_id || ' not found.');
```

Function GET\_EMPLOYEE\_COUNT\_BY\_LIBRARY compiled  
Elapsed: 00:00:00.019

```

19 SELECT get_employee_count_by_library(21) FROM dual;
20
21
22
23
24
```

Query Result Script Output DBMS Output Explain Plan Autotr

Download Execution time: 0.008 seconds

GET_EMPLOYEE_COUNT_BY_LIBRARY	
1	2

#### 4. Function to get number of users who have made donations to the library.

get\_donors(l\_id) - It takes Library ID(l\_id) as input and returns the number of users that have made donations to that library.



```

2  CREATE OR REPLACE FUNCTION get_donors (l_id IN NUMBER) RETURN NUMBER
3  IS
4      num_donors NUMBER;
5  BEGIN
6      SELECT COUNT(DISTINCT u_id)
7      INTO num_donors
8      FROM donations_England
9      WHERE l_id = get_donors.l_id;
10
11     RETURN num_donors;
12 END;
13
```

Function GET\_DONORS compiled  
Elapsed: 00:00:00.036

```
--
15 DECLARE
16     num_donors NUMBER;
17 BEGIN
18     num_donors := get_donors(41);
19     DBMS_OUTPUT.PUT_LINE('Number of donors: ' || num_donors);
20 END;
21 /
22
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Number of donors: 1

PL/SQL procedure successfully completed.



Elapsed: 00:00:00.014

5. Function to find the number of available cabins in a library.

get\_available\_cabins(l\_id) - It takes Library ID(l\_id) as input and returns the number of available cabins in that library.

```
1  --function to get the number of available cabins in a library
2  CREATE OR REPLACE FUNCTION get_available_cabins (l_id IN NUMBER) RETURN NUMBER
3  IS
4      num_available NUMBER;
5  BEGIN
6      SELECT COUNT(*)
7      INTO num_available
8      FROM cabin_Canada
9      WHERE l_id = get_available_cabins.l_id
10     AND availability = 'available';
11
12     RETURN num_available;
13 END;
14 /
15
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Function GET\_AVAILABLE\_CABINS compiled

Elapsed: 00:00:00.006

```

45
46 DECLARE
47     num_available NUMBER;
48 BEGIN
49     num_available := get_available_cabins(31);
50     DBMS_OUTPUT.PUT_LINE('Available Cabins: ' || num_available);
51 END;
52
53

```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Available Cabins: 3

## PROCEDURES:

1. Procedure to insert a new employee into employees table.

add\_employee() - It takes all the attributes of the employees table as input and inserts it into employees table. Here, we have inserted into employees\_usa fragment.

```

3 CREATE OR REPLACE PROCEDURE add_employee (
4     e_name IN VARCHAR2,
5     e_salary IN NUMBER,
6     e_contact IN VARCHAR2,
7     e_role IN VARCHAR2,
8     e_doj IN DATE,
9     l_id IN NUMBER
10 ) AS
11     max_id NUMBER;
12 BEGIN
13     SELECT MAX(e_id)
14     INTO max_id
15     FROM EMPLOYEES_USA;
16
17     IF max_id IS NULL THEN
18         max_id := 1;
19     ELSE
20         max_id := max_id + 1;
21     END IF;
22
23     INSERT INTO employees_usa (e_id, name, salary, contact, role, doj, l_id)
24     VALUES (max_id, e_name, e_salary, e_contact, e_role, e_doj, l_id);
25
26     EXCEPTION
27     WHEN others THEN
28         DBMS_OUTPUT.PUT_LINE('Exception occurred');
29
30     COMMIT;
31 END;
32 /
33
34 BEGIN
35     add_employee('John Doe', 50000, '9995551234', 'Librarian', TO_DATE('2022-05-02', 'YYYY-MM-DD'), 1);
36 END;

```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Procedure ADD\_EMPLOYEE compiled

Elapsed: 00:00:00.020

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.019

## 2. Procedure to update the availability of a cabin in a library.

update\_cabin\_availability() - It takes all the attributes of the cabin table as input and updates the availability column of cabin. Here, we implemented this procedure on cabin\_india fragment.

```
3 CREATE OR REPLACE PROCEDURE update_cabin_availability (  
4     cabin_id IN VARCHAR2,  
5     availability IN VARCHAR2,  
6     l_id IN NUMBER  
7 ) AS  
8 BEGIN  
9     UPDATE cabin_india  
10    SET availability = availability  
11    WHERE cabin_id = cabin_id AND l_id = l_id;  
12    COMMIT;  
13    EXCEPTION  
14    When OTHERS THEN  
15        DBMS_OUTPUT.PUT_LINE('Error occurred');  
16 END;  
17  
18  
19 begin  
20     update_cabin_availability('C1002', 'not available', 1);  
21 END;
```

Query Result    Script Output    DBMS Output    Explain Plan    Autotrace



Procedure UPDATE\_CABIN\_AVAILABILITY compiled

Elapsed: 00:00:00.105

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.013

## 3. Procedure to add a new book to the books table.

add\_book() - It takes an entire new row of books table as input and inserts it into the books table. Here, we have implemented this procedure on books\_france fragment.

```

28 CREATE OR REPLACE PROCEDURE add_book (
29     isbn IN VARCHAR2,
30     status IN VARCHAR2,
31     name IN VARCHAR2,
32     author IN VARCHAR2,
33     l_id IN NUMBER,
34     dept_id IN NUMBER,
35     genre IN VARCHAR2
36 ) AS
37     max_id NUMBER;
38 BEGIN
39     SELECT MAX(b_id)
40     INTO max_id
41     FROM books_france;
42
43     IF max_id IS NULL THEN
44         max_id := 1;
45     ELSE
46         max_id := max_id + 1;
47     END IF;
48
49     INSERT INTO books_france (b_id, isbn, status, name, author, l_id, dept_id, genre)
50     VALUES (max_id, isbn, status, name, author, l_id, dept_id, genre);
51
52     EXCEPTION
53     when others THEN
54         DBMS_OUTPUT.PUT_LINE('error occured')
55
56     COMMIT;
57 END;
58 BEGIN
59     add_book('9783161484100', 'available', 'The Trial', 'Franz Kafka', 1, 2, 'Fiction');
60     COMMIT;
61 END;
62 select * from books_france;

```

Query Result    Script Output    DBMS Output    Explain Plan    Autotrace    SQL History



Procedure ADD\_BOOK compiled

Elapsed: 00:00:00.024

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.011

#### 4. Procedure to note a book issued to a user information to book\_transactions table.

Issue\_book() - It takes an entire row of book\_transactions table and inserts it into the table. Here, we have implemented this procedure on book\_transactions\_canada fragment.

```

28 CREATE OR REPLACE PROCEDURE issue_book (
29     b_id IN NUMBER,
30     u_id IN NUMBER,
31     issued_date IN DATE
32 ) AS
33     max_t_id NUMBER;
34 BEGIN
35     SELECT MAX(t_id)
36     INTO max_t_id
37     FROM book_transactions_canada;
38
39     IF max_t_id IS NULL THEN
40         max_t_id := 1;
41     ELSE
42         max_t_id := max_t_id + 1;
43     END IF;
44
45     INSERT INTO book_transactions_canada (t_id, b_id, u_id, issued_date)
46     VALUES (max_t_id, b_id, u_id, issued_date);
47     COMMIT;
48
49     UPDATE books_canada
50     SET status = 'unavailable'
51     WHERE b_id = b_id;
52
53     Exception
54     When others then
55         DBMS_OUTPUT.PUT_LINE('error occured')
56     COMMIT;
57 END;
58 BEGIN
59     issue_book(1, 101, SYSDATE);
60 END;

```

Query Result   Script Output   DBMS Output   Explain Plan   Autotrace   SQL History



Procedure ISSUE\_BOOK compiled

Elapsed: 00:00:00.017

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.011



5. Procedure to check in details of the book a user has returned.

checkin\_book() - It takes all the attributes of library checkin for the user returning book and inserts it into the library\_checkin table. Here, this procedure is implemented on library\_checkin\_england fragment.

```
28 CREATE OR REPLACE PROCEDURE checkin_book (  
29     b_id IN NUMBER,  
30     u_id IN NUMBER,  
31     l_id IN NUMBER,  
32     checkin_date IN DATE  
33 ) AS  
34     max_checkin_id NUMBER;  
35 BEGIN  
36     SELECT MAX(checkin_id)  
37     INTO max_checkin_id  
38     FROM library_checkin_england;  
39  
40     IF max_checkin_id IS NULL THEN  
41         max_checkin_id := 1;  
42     ELSE  
43         max_checkin_id := max_checkin_id + 1;  
44     END IF;  
45  
46     INSERT INTO library_checkin_england (checkin_id, l_id, u_id, checkin_date)  
47     VALUES (max_checkin_id, l_id, u_id, checkin_date);  
48     COMMIT;  
49  
50     UPDATE books_england  
51     SET status = 'available'  
52     WHERE b_id = b_id;  
53  
54     EXCEPTION  
55     when others THEN  
56         DBMS_OUTPUT.PUT_LINE('error occured');  
57     COMMIT;  
58 END;  
59  
60 BEGIN  
61     checkin_book(b_id => 1, l_id => 1, u_id => 2, checkin_date => SYSDATE);  
62 END;  
63 /
```

Query Result   Script Output   DBMS Output   Explain Plan   Autotrace   SQL History



Procedure CHECKIN\_BOOK compiled

Elapsed: 00:00:00.049

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.012

## TRIGGERS:

1. Trigger to stop the insertion of user with same contact number as an existing user – Applied for India fragment.

```
1  --Trigger to prevent the insertion of a user with the same contact number as an existing user:
2  CREATE OR REPLACE TRIGGER prevent_duplicate_user_trigger
3  BEFORE INSERT ON users_India
4  FOR EACH ROW
5  DECLARE
6  user_count NUMBER;
7  BEGIN
8  SELECT COUNT(*) INTO user_count FROM users_India WHERE contact = :NEW.contact;
9  IF user_count > 0 THEN
10 RAISE_APPLICATION_ERROR(-20001, 'User with this contact number already exists');
11 END IF;
12 END;
13 /
14
15
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

Trigger PREVENT\_DUPLICATE\_USER\_TRIGGER compiled  
Elapsed: 00:00:00.119

Here we have triggered the exception by using the existing contact number.

```
18 INSERT INTO users_India (u_id, u_name, address, contact, dob, gender) VALUES
19 (71, 'Johnny Smith', '123 Main Street', '919314508623', DATE '1990-01-20', 'F');
20
21
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

ORA-20001: User with this contact number already exists  
ORA-06512: at "SREEKAR.PREVENT\_DUPLICATE\_USER\_TRIGGER", line 6  
ORA-04088: error during execution of trigger 'SREEKAR.PREVENT\_DUPLICATE\_USER\_TRIGGER'  
Error at Line: 7 Column: 0

```
19
20 INSERT INTO users_India (u_id, u_name, address, contact, dob, gender) VALUES
21 (76, 'Johnny', '126 Main Street', '919314508999', DATE '1990-01-21', 'M');
22
23
24
```



Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

1 row inserted.  
Elapsed: 00:00:00.004

2. Trigger to update the employee salary not more than 20% of the existing salary – Applied for Canada fragment.

```
1  --Create a trigger to update the employee salary not more than 20% of the existing salary
2  CREATE OR REPLACE TRIGGER salary_increase_trigger
3  BEFORE UPDATE ON employees_Canada
4  FOR EACH ROW
5  DECLARE
6  max_increase_pct NUMBER := 1.2; -- 20% increase
7  BEGIN
8  dbms_output.put_line('I got triggered');
9  IF :new.salary > :old.salary * max_increase_pct THEN
10     RAISE_APPLICATION_ERROR(-20001, 'Salary increase cannot exceed 20%.');
11  END IF;
12  END;
13  /
```



Query Result   Script Output   DBMS Output   Explain Plan   Autotrace   SQL History

Trigger SALARY\_INCREASE\_TRIGGER compiled  
Elapsed: 00:00:00.022

```
14
15  UPDATE employees_Canada
16  SET salary = salary * 1.2
17  WHERE e_id = 419;
18
```

Query Result   Script Output   DBMS Output   Explain

I got triggered

1 row updated.

Elapsed: 00:00:00.034

This exception will get triggered when user tries to update the salary of employee more than 20%

```
15 UPDATE employees_Canada
16 SET salary = salary * 1.5
17 WHERE e_id = 419;
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

ORA-20001: Salary increase cannot exceed 20%.

ORA-06512: at "LAVANYA.SALARY\_INCREASE\_TRIGGER", line 6

ORA-04088: error during execution of trigger 'LAVANYA.SALARY\_INCREASE\_TRIGGER'

Error at Line: 7 Column: 0

### 3. Trigger to prevent the insertion of a book with the same ISBN as an existing book in the same library

```
1 --Trigger to prevent the insertion of a book with the same ISBN as an existing book in the same library
2 CREATE OR REPLACE TRIGGER prevent_duplicate_book_trigger
3 BEFORE INSERT ON books_France
4 FOR EACH ROW
5 DECLARE
6     book_count NUMBER;
7 BEGIN
8     SELECT COUNT(*) INTO book_count FROM books_France WHERE isbn = :NEW.isbn AND l_id = :NEW.l_id;
9     IF book_count > 0 THEN
10         RAISE_APPLICATION_ERROR(-20001, 'Book with this ISBN already exists in this library');
11     END IF;
12 END;
13 /
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Trigger PREVENT\_DUPLICATE\_BOOK\_TRIGGER compiled

Elapsed: 00:00:00.064

The exception is triggered as the we are trying to insert a book with existing isbn.

```
19
20 INSERT INTO books_France (b_id, isbn, status, name, author, l_id, dept_id, Genre) VALUES
21 (12, '9780142424179', 'available', 'One night at the call centre', 'Chetan Bhagat', 1, 1, 'Romance');
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

ORA-20001: Book with this ISBN already exists in this library

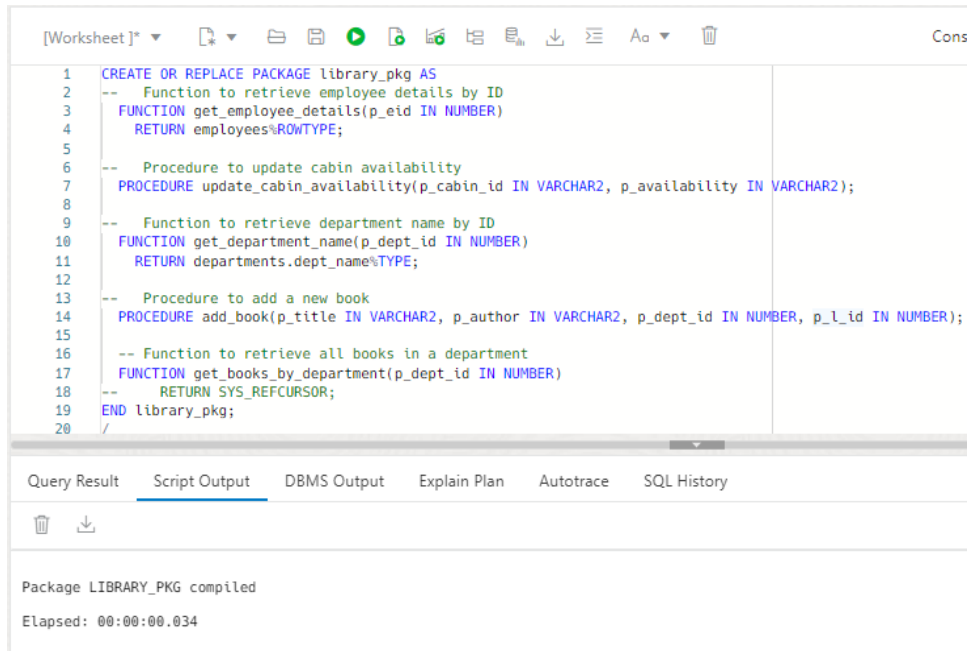
ORA-06512: at "SHARANYA.PREVENT\_DUPLICATE\_BOOK\_TRIGGER", line 6

ORA-04088: error during execution of trigger 'SHARANYA.PREVENT\_DUPLICATE\_BOOK\_TRIGGER'

Error at Line: 7 Column: 0

## PACKAGE:

### Creation of Package named library\_pkg

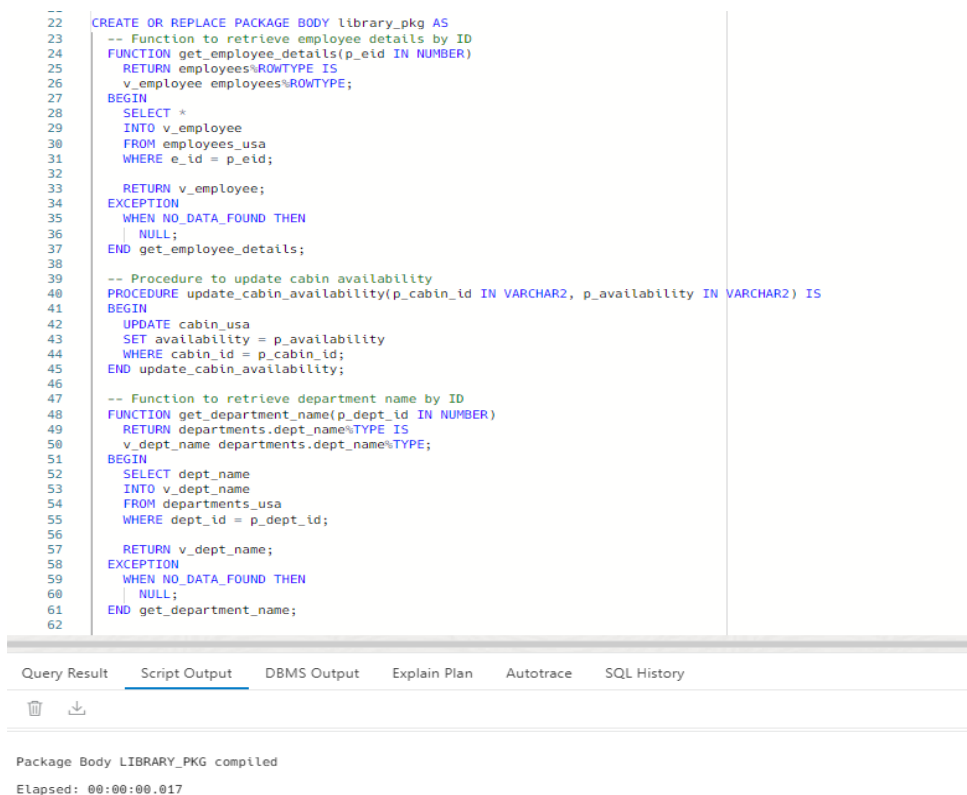


The screenshot shows the SQL Developer interface with a worksheet containing the following SQL code:

```
1 CREATE OR REPLACE PACKAGE library_pkg AS
2 -- Function to retrieve employee details by ID
3 FUNCTION get_employee_details(p_eid IN NUMBER)
4 RETURN employees%ROWTYPE;
5
6 -- Procedure to update cabin availability
7 PROCEDURE update_cabin_availability(p_cabin_id IN VARCHAR2, p_availability IN VARCHAR2);
8
9 -- Function to retrieve department name by ID
10 FUNCTION get_department_name(p_dept_id IN NUMBER)
11 RETURN departments.dept_name%TYPE;
12
13 -- Procedure to add a new book
14 PROCEDURE add_book(p_title IN VARCHAR2, p_author IN VARCHAR2, p_dept_id IN NUMBER, p_l_id IN NUMBER);
15
16 -- Function to retrieve all books in a department
17 FUNCTION get_books_by_department(p_dept_id IN NUMBER)
18 RETURN SYS_REFCURSOR;
19 END library_pkg;
20 /
```

Below the code editor, the 'Script Output' tab is selected, showing the message: 'Package LIBRARY\_PKG compiled' and 'Elapsed: 00:00:00.034'.

### Compilation of Package Body:



The screenshot shows the SQL Developer interface with a worksheet containing the following SQL code for the package body:

```
--
22 CREATE OR REPLACE PACKAGE BODY library_pkg AS
23 -- Function to retrieve employee details by ID
24 FUNCTION get_employee_details(p_eid IN NUMBER)
25 RETURN employees%ROWTYPE IS
26 v_employee employees%ROWTYPE;
27 BEGIN
28 SELECT *
29 INTO v_employee
30 FROM employees_usa
31 WHERE e_id = p_eid;
32
33 RETURN v_employee;
34 EXCEPTION
35 WHEN NO_DATA_FOUND THEN
36 NULL;
37 END get_employee_details;
38
39 -- Procedure to update cabin availability
40 PROCEDURE update_cabin_availability(p_cabin_id IN VARCHAR2, p_availability IN VARCHAR2) IS
41 BEGIN
42 UPDATE cabin_usa
43 SET availability = p_availability
44 WHERE cabin_id = p_cabin_id;
45 END update_cabin_availability;
46
47 -- Function to retrieve department name by ID
48 FUNCTION get_department_name(p_dept_id IN NUMBER)
49 RETURN departments.dept_name%TYPE IS
50 v_dept_name departments.dept_name%TYPE;
51 BEGIN
52 SELECT dept_name
53 INTO v_dept_name
54 FROM departments_usa
55 WHERE dept_id = p_dept_id;
56
57 RETURN v_dept_name;
58 EXCEPTION
59 WHEN NO_DATA_FOUND THEN
60 NULL;
61 END get_department_name;
62
```

Below the code editor, the 'Script Output' tab is selected, showing the message: 'Package Body LIBRARY\_PKG compiled' and 'Elapsed: 00:00:00.017'.

If you call the `get_employee_details` function with an existing employee ID, it will return a record with the employee's details. If the employee ID does not exist in the `employees_usa` table, the function will return null.

```
111 DECLARE
112     v_employee employees%ROWTYPE;
113 BEGIN
114     v_employee := library_pkg.get_employee_details(201);
115     DBMS_OUTPUT.PUT_LINE(v_employee.e_id || ', ' || v_employee.name || ', ' || v_employee.conatct);
116 END;
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

201, Emily Johnson, 8356248719

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.009

If you call the `get_department_name` function with an existing department ID, it will return the department name as a string. If the department ID does not exist in the `departments_usa` table, the function will return null.

```
105 BEGIN
106     dept_name := library_pkg.get_department_name(102);
107     DBMS_output.put_line('department name: ' || dept_name)
108 END;
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace

department name: Fiction

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.008

If you call the `get_books_by_department` function with an existing department ID, it will return a cursor with all the books that belong to that department. If the department ID does not exist in the `departments_usa` table or if there are no books belonging to that department in the `books_usa` table, the function will return an empty cursor.

```
93  
94 BEGIN  
95 library_pkg.add_book('The Great Gatsby', 'F. Scott Fitzgerald', 1, 1101);  
96 END;  
97  
98
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace   SQL History

PL/SQL procedure successfully completed.  
Elapsed: 00:00:00.007

If you call the update\_cabin\_availability procedure with an existing cabin ID and a valid availability value, it will update the availability column for that cabin in the cabin\_usa table. If the cabin ID does not exist in the cabin\_usa table, no rows will be updated.

```
94 BEGIN  
95 library_pkg.update_cabin_availability(1001, 'available');  
96 END;  
97
```

Query Result   **Script Output**   DBMS Output   Explain Plan   Autotrace

PL/SQL procedure successfully completed.  
Elapsed: 00:00:00.007

### Individual contribution: -

- I have created the Function to return the most popular genre of a user in India Site
- I have created the stored Procedure to insert a new employee into employees' table in India Site.
- I have created a trigger to stop the insertion of user with same contact number as an existing user in India Site
- We have distributed the work to ourselves and worked along with my all of them and helped them in solving their part of project.
- As a team, we have collaborated with each other to complete this project part.