# Que.Create a Book and Member Table for Library Management System

#### 1. Book Table:

```
Create table Book1
(Book_id number(10),
book_name varchar(20),
author varchar(20),
publication_year number(10));
```

### **Output:**

```
Results Explain Descri

Table created.

0.03 seconds
```

## 2.Member Table

```
Create table Member1
(Member_id number(10),
name varchar(20),
email varchar(20));
```

```
Results Explain Describe :

Table created.

0.00 seconds
```

#### Que. Insert the Records into Book and Member Table.

#### 1:Insert into Book table:

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (1, 'The Great Gatsby', 'F. Scott Fitzgerald', 1925);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (2, '1984', 'George Orwell', 1949);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (3, 'To Kill a Mockingbird', 'Harper Lee', 1960);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (4, 'Pride and Prejudice', 'Jane Austen', 1813);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (5, 'The Catcher in the Rye', 'J.D. Salinger', 1951);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (6, 'Moby Dick', 'Herman Melville', 1851);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (7, 'War and Peace', 'Leo Tolstoy', 1869);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (8, 'Brave New World', 'Aldous Huxley', 1932);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (9, 'The Odyssey', 'Homer', 800);

INSERT INTO Book1 (Book\_id, book\_name, author, publication\_year) VALUES (10, 'The Picture of Dorian Gray', 'Oscar Wilde', 1890);

#### Select Book table:

select \* from Book1

Results Ex	plain Describe S	aved SQL Histor	у
BOOK_ID	BOOK_NAME	AUTHOR	PUBLICATION_YEAR
3	The Great Gatsby	F. Scott Fitzgerald	1925
4	1984	George Orwell	1949
4	1984	George Orwell	1949
5	Mockingbird	Harper Lee	1960
6	Moby-Dick	Herman Melville	1851
7	War and Peace	Leo Tolstoy	1869
8	Pride and Prejudice	Jane Austen	1813
9	The Great Gatsby	F. Scott Fitzgerald	1925
10	The Alchemist	Paulo Coelho	1988

**CSV Export** 

9 rows returned in 0.02 seconds

#### 1:Insert into Member table:

INSERT INTO Member1 (Member\_id, name, email) VALUES (1, 'John Doe', 'john.doe@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (2, 'Jane Smith', 'jane.smith@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (3, 'Alice Johnson', 'alice.johnson@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (4, 'Bob Brown', 'bob.brown@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (5, 'Carol White', 'carol.white@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (6, 'David Wilson', 'david.wilson@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (7, 'Eva Green', 'eva.green@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (8, 'Frank Blue', 'frank.blue@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (9, 'Grace Black', 'grace.black@example.com');

INSERT INTO Member1 (Member\_id, name, email) VALUES (10, 'Henry Red', 'henry.red@example.com');

### Select member table:

select \* from member

Results Exp	lain Describe	Saved SQL History
MEMBER ID	) NAME	EMAIL
1	John Doe	john.doe@example.com
2	Jane Smith	jane.smith@ex.com
3	Alice Johnson	alice.johnson@ex.com
4	Bob Brown	bob.brown@ex.com
5	Carol White	carol.white@ex.com
6	David Wilson	david.wilson@ex.com
7	Eva Green	eva.green@ex.com
8	Frank Blue	frank.blue@ex.com
9	Grace Black	grace.black@ex.com
10	Henry Red	henry.red@ex.com

10 rows returned in 0.02 seconds

CSV Export

# Que.Perform Update Query in Book and Member Table.

### 1. Update Query on Book table

**UPDATE Book1** 

SET book\_name = 'Macbeth', publication\_year = 1983

WHERE Book\_id = 4;

Results Explain Describe Saved SQL

2 row(s) updated.

0.02 seconds

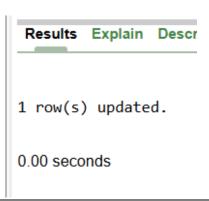
# **Select Query:**

Select \* from Book1

Results Ex	plain Describe Sa	aved SQL History	′
BOOK_ID	BOOK_NAME	AUTHOR	PUBLICATION_YEAR
3	The Great Gatsby	F. Scott Fitzgerald	1925
4	Macbeth	George Orwell	1983
4	Macbeth	George Orwell	1983
5	Mockingbird	Harper Lee	1960
6	Moby-Dick	Herman Melville	1851
7	War and Peace	Leo Tolstoy	1869
8	Pride and Prejudice	Jane Austen	1813
9	The Great Gatsby	F. Scott Fitzgerald	1925
10	The Alchemist	Paulo Coelho	1988
9 rows returned in 0.00 seconds		CSV Export	

# 2. Update Query on Member table

UPDATE Member1
SET name = 'Sakshi', email = 'sakshi@gmail.com'
WHERE Member\_id = 1;



# Que.Write SQL statement to make use of all phases of select commands

**SELECT** 

b.book\_name,

b.author,

COUNT(m.member\_id) AS member\_count

**FROM** 

Book1 b

**LEFT JOIN** 

Member1 m ON b.book\_id = m.member\_id

WHERE

b.publication\_year >= 1925

**GROUP BY** 

b.book\_name, b.author

**HAVING** 

COUNT(m.member\_id) > 0

**ORDER BY** 

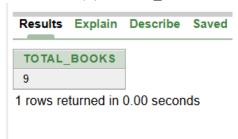
b.book\_name ASC;

Results Explain	Describe Save	ed SQL History
BOOK_NAME	AUTHOR	MEMBER_COUNT
Macbeth	George Orwell	2
Mockingbird	Harper Lee	1
The Alchemist	Paulo Coelho	1
The Great Gatsby	F. Scott Fitzgerald	2
4 rows returned in	0.00 seconds	CSV Export

Que.Write SQL Statement by using Oracle functions.

1. Count the Number of Books:

SELECT COUNT(\*) AS Total\_Books FROM Book1;



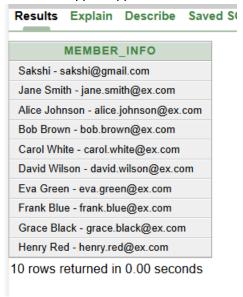
SELECT COUNT(\*) AS total members FROM Member1;



1 rows returned in 0.00 seconds

2. Concatenate Name and Email of Members

SELECT name | | ' - ' | | email AS member info FROM Member1;



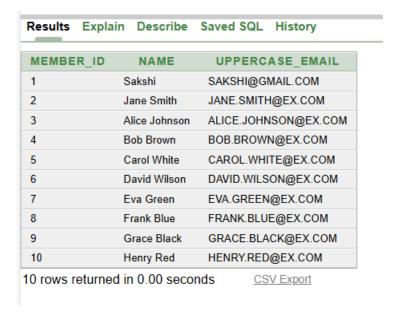
3.Get members whose names start with a specific letter (e.g., 'A'):

SELECT \* FROM Member1 WHERE UPPER(name) LIKE 'A%';



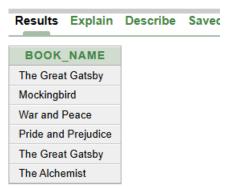
### 4. Retrieve Members with Emails in Uppercase

SELECT Member id, Name, UPPER(Email) AS Uppercase Email FROM Member1;



### 5. List Book Titles with Length Greater Than 10 Characters

SELECT Book\_name FROM Book1 WHERE LENGTH(Book\_name) > 10;



6 rows returned in 0.00 seconds

# Que.Create view by using table book an member perform select ,update,delete operation

CREATE VIEW BookMemberView AS

SELECT

B.Book\_id,

B.book\_name,

B.author,

B.publication\_year,

M.Member\_id,

M.name,

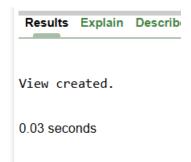
M.email

FROM Book1 B

JOIN

Member1 M ON B.Book id = M.Member id;

#### Output:



### 1. Update Operation

UPDATE Member1 SET Email = 'sakshi@gmail.com'
WHERE Member\_id = 1;

Results Explain Describe

1 row(s) updated.

0.01 seconds

### 2. Select operation

#### SELECT \* FROM BookMemberView;



### 3. Delete Operation:

DELETE FROM Book1 WHERE Book\_id = 3;



Que. write a sub query for the book and member table for library

### 1. Sub Query

```
SELECT name, email

FROM member

WHERE member_id IN (

SELECT member_id

FROM Book

WHERE Book_id = (SELECT Book_id FROM book WHERE publication_year=2021)
);
```

### **Output:**

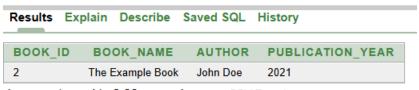


2 rows returned in 0.02 seconds

#### 2. Sub Query

SELECT book\_id, title, author, publication\_year FROM Book WHERE author IN (SELECT name FROM Member);

#### **Output:**



1 rows returned in 0.00 seconds

CSV Export

Que. write a PL/SQL by using if statement to display squares of all odd numbers between 1 to 25

```
FOR i IN 1..25 LOOP

-- Check if the number is odd

IF MOD(i, 2) = 1 THEN

-- Display the square of the odd number

DBMS_OUTPUT_LINE('Square of ' || i || ' is : ' || i * i);

END IF;

END LOOP;

END;
```

```
Square of 1 is : 1
Square of 3 is : 9
Square of 5 is : 25
Square of 7 is : 49
Square of 9 is : 81
Square of 11 is : 121
Square of 13 is : 169
Square of 15 is : 225
Square of 15 is : 225
Square of 15 is : 225
Square of 17 is : 289
Square of 19 is : 361
Square of 21 is : 441
Square of 23 is : 529
Square of 25 is : 625
Statement processed.
```

Que. write a PL/SQI to implement while loop for library table.

#### 1. Library Table

```
CREATE TABLE library (
book_id NUMBER PRIMARY KEY,
title VARCHAR2(255),
author VARCHAR2(255),
available_copies NUMBER
);
```

### 2. Insert Into Library

```
INSERT INTO library (book_id, title, author, available_copies) VALUES (1, '1984', 'George Orwell', 5);
INSERT INTO library (book_id, title, author, available_copies) VALUES (2, 'To Kill a Mockingbird', 'Harper Lee', 3);
```

### 3. While loop

```
DECLARE
  v_book_id
                NUMBER;
  v_title
             VARCHAR2(255);
  v_author
                VARCHAR2(255);
  v_available_copies NUMBER;
  CURSOR c library IS
    SELECT book_id, title, author, available_copies
    FROM library
    WHERE available copies > 0;
  v counter NUMBER := 0;
BEGIN
  OPEN c library;
  LOOP
    FETCH c_library INTO v_book_id, v_title, v_author, v_available_copies;
    EXIT WHEN c_library%NOTFOUND; -- Exit condition for the loop
    -- Process the fetched data (in this case, print it)
```

```
Results Explain Describe Saved SQL History

Book ID: 1, Title: 1984, Author: George Orwell, Available Copies: 5
Book ID: 2, Title: To Kill a Mockingbird, Author: Harper Lee, Available Copies: 3
Book ID: 4, Title: Moby Dick, Author: Herman Melville, Available Copies: 2
Total books with available copies: 3

Statement processed.

0.05 seconds
```

#### Que. write a PL/SQI to implement For loop for library table

```
DECLARE

-- Declare a cursor to fetch data from the library table

CURSOR library_cursor IS

SELECT book_id, title, author, available_copies

FROM library;

BEGIN

-- Loop through each record fetched by the cursor

FOR book_record IN library_cursor LOOP

-- Output the details of each book

DBMS_OUTPUT.PUT_LINE('Book ID: ' || book_record.book_id ||

', Title: ' || book_record.title ||

', Author: ' || book_record.author ||

', Available Copies: ' || book_record.available_copies);

END LOOP;

END;
```

```
Results Explain Describe Saved SQL History

Book ID: 1, Title: 1984, Author: George Orwell, Available Copies: 5
Book ID: 2, Title: To Kill a Mockingbird, Author: Harper Lee, Available Copies: 3
Book ID: 3, Title: The Great Gatsby, Author: F. Scott Fitzgerald, Available Copies: 0
Book ID: 4, Title: Moby Dick, Author: Herman Melville, Available Copies: 2

Statement processed.
```

Que.Implement Error handling in PI/SQL for library Book Table.

```
DECLARE
        v Book id NUMBER := 101; -- Example book ID
        v Book name VARCHAR2(100) := 'The Great Gatsby'; -- Example title
        v Author VARCHAR2(100) := 'F. Scott Fitzgerald'; -- Example author
        v_Publication_Year NUMBER := 1925; -- Example year
      BEGIN
        -- Attempt to insert a new book record
        INSERT INTO Book (Book id, Book name, Author, Publication Year)
        VALUES (v Book id, v Book name, v Author, v Publication Year);
        DBMS OUTPUT.PUT LINE('Book record inserted successfully.');
      EXCEPTION
        WHEN DUP_VAL_ON_INDEX THEN
          DBMS_OUTPUT.PUT_LINE('Error: Book ID already exists.');
        WHEN VALUE ERROR THEN
          DBMS_OUTPUT_LINE('Error: Invalid value for one of the fields.');
        WHEN OTHERS THEN
          DBMS OUTPUT.PUT LINE('An unexpected error occurred: ' | | SQLERRM);
      END;
Output:
Results Explain Describe Saved SQL History
Book record inserted successfully.
1 row(s) inserted.
0.05 seconds
```

### Que. write a PL/SQI to implement Procedure For the Given Table

```
CREATE OR REPLACE PROCEDURE insert_book_and_member (
  p_book_id IN NUMBER,
  p book name IN VARCHAR2,
  p_author IN VARCHAR2,
  p_publication_year IN NUMBER,
  p_member_id IN NUMBER,
  p name IN VARCHAR2,
  p email IN VARCHAR2
) AS
BEGIN
  -- Insert into the Book Table
  INSERT INTO Book (Book_ID, book_name, Author, publication_year)
  VALUES (p_book_id, p_book_name, p_author, p_publication_year);
  -- Insert into the Member Table
  INSERT INTO Member (Member_ID, Name, email)
  VALUES (p_member_id, p_name, p_email);
  -- Commit the transaction
  COMMIT;
END;
                                      Results Explain Describe
  Output:
                                     Procedure created.
                                     0.16 seconds
```

#### **BEGIN**

```
insert_book_and_member(
    p_book_id => 1,
    p_book_name => 'The Great Gatsby',
    p_author => 'F. Scott Fitzgerald',
    p_publication_year => 1925,
    p_member_id => 101,
    p_name => 'John Doe',
    p_email => 'johndoe@example.com'
    );
END;
```

```
Results Explain Describe Saved
Statement processed.

0.02 seconds
```

### Que. write a PL/SQI to create and manage cursor for Book Table

```
DECLARE
 -- Cursor to fetch Book table details
 CURSOR book cursor IS
  SELECT Book_ID, book_name, Author, publication_year
  FROM Book;
 -- Variables to hold book details fetched from the cursor
 v_book_id Book.Book_ID%TYPE;
 v_book_name Book.book_name%TYPE;
 v_author Book.Author%TYPE;
 v_publication_year Book.publication_year%TYPE;
BEGIN
 -- Open the cursor
 OPEN book_cursor;
 -- Loop through each row fetched by the cursor
 LOOP
  FETCH book_cursor INTO v_book_id, v_book_name, v_author, v_publication_year;
 -- Exit the loop when no more rows are fetched
  EXIT WHEN book_cursor%NOTFOUND;
  -- Display the book details
```

```
DBMS_OUTPUT_LINE('Book ID: ' || v_book_id || ', Name: ' || v_book_name || ',
Author: ' || v_author || ', Year: ' || v_publication_year);

END LOOP;

-- Close the cursor

CLOSE book_cursor;

END;
```

#### **Output:**

#### Results Explain Describe Saved SQL History

```
Book ID: 1, Name: Smith's Guide, Author: Jane Smith, Year: 2023
Book ID: 2, Name: The Example Book, Author: John Doe, Year: 2021
Book ID: 101, Name: The Great Gatsby, Author: F. Scott Fitzgerald, Year: 1925
Book ID: 1, Name: The Great Gatsby, Author: F. Scott Fitzgerald, Year: 1925
```

Statement processed.

0.00 seconds