Mini SQL project on fictional characters to join tables

By: Rakshita
June 2025

Introduction :

This mini project is based on SQL JOIN operations. It mainly focuses on combining data from two related tables using INNER JOIN, and comparing data within the same table using SELF JOIN. The project shows how JOINs help in retrieving connected and meaningful information from a database.

Objective :

The objective of this mini project is to implement and understand various SQL JOIN operations using fictional book and character data. The project aims to showcase the use of INNER JOIN and SELF JOIN to fetch meaningful relationships and connections between characters and the books they belong to. It focuses on practical usage of relational database concepts by creating and linking multiple tables, representing realistic connections between characters through custom relationships.

Tools used in this project :

- SQL (Structured Query Language) – for creating and managing relational tables.
- SQLite – lightweight database engine used for executing SQL queries.
- LibreOffice Writer – for project documentation and formatting.
- Fictional Dataset – manually created data based on book characters and their relationships.

Code :

```
/*This code helps to understand the use of JOIN operation in SQL.*/
/*Create a tables named book.*/
CREATE TABLE book (
id INTEGER PRIMARY KEY AUTOINCREMENT,
book_name TEXT,
series TEXT,
author TEXT);

/*Inserting values in book table.*/
INSERT INTO book (book_name, series, author) VALUES ("Shatter me",
"Shatter me series", "Tahereh mafi");
INSERT INTO book (book_name, series, author) VALUES ("Haunted adeline",
"Cat-mouse Duet", "H.D cartlon");
INSERT INTO book (book_name, series, author) VALUES ("Twisted love",
"Twisted series", "Ana huang");
INSERT INTO book (book_name, series, author) VALUES ("The predator",
"The dark verse", "Runyx");
INSERT INTO book (book_name, series, author) VALUES ("Twisted Games",
"Twisted series", "Ana huang");
INSERT INTO book (book_name, series, author) VALUES ("It start with us", "It
ends with us", "Collen Hoover");
INSERT INTO book (book_name, series, author) VALUES ("God of Ruin",
"Legacy of gods", "Rina kent");
INSERT INTO book (book_name, series, author) VALUES ("Harry potter and
the philospher stone", "Harry potter series", "J.K rowling");

/*Create a tables named character.*/
CREATE TABLE character (
id INTEGER PRIMARY KEY AUTOINCREMENT,
name TEXT,
book_id INTEGER,
```

```
gender TEXT);

/*Inserting values in character table.*/
INSERT INTO character (name, book_id, gender) VALUES ("Zade Meadows",
2, "Male");
INSERT INTO character (name, book_id, gender) VALUES ("Juilette Ferrars", 1
, "Female");
INSERT INTO character (name, book_id, gender) VALUES ("Adeline Reilly", 2 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Josh chen", 3 ,
"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Morana Vitalio",
4, "Female");
INSERT INTO character (name, book_id, gender) VALUES ("Aaron Warner", 1,
"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Tristian Caine",
4 ,"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Ava chen", 3 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Dante Maroni",
4 ,"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Ryle larsen", 5 ,
"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Lily Bloom", 6 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Atlas corrigon",
6 ,"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Bright von
Ascheberg", 5 ,"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Stella alonso", 3 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Cecily Knight", 7 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Christian Harper",
```

```
5 ,"Male");
INSERT INTO character (name, book_id, gender) VALUES ("Glydon Grace", 7 ,
"Female");
INSERT INTO character (name, book_id, gender) VALUES ("Jeremy Volkov", 7,
"Male");

/*Create a tables named relationship.*/
CREATE TABLE relationship (
id INTEGER PRIMARY KEY AUTOINCREMENT,
character1_id INTEGER,
character2_id INTEGER,
relation TEXT);

/*Inserting values in relationship table.*/
INSERT INTO relationship (character1_id, character2_id, relation) VALUES (1,
3, "Stalker");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES (2,
6, "Lovers");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES (5,
9, "Friends");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES (4,
8, "Siblings");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES (5,
7, "Lovers");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES
(10, 13, "Bodyguard");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES
(11, 12 ,"Childhood Lovers");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES
(14, 8 ,"Best-friends");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES
(15, 17 ,"Best-friends");
INSERT INTO relationship (character1_id, character2_id, relation) VALUES
(15, 18, "Lovers");
```

```
/*SELECT query to display the table rows*/
SELECT * FROM book;
SELECT * FROM character;
SELECT * FROM relationship;

/*This query display the relationship between characters by using JOIN operation.*/
SELECT a.name, b.name, relationship.relation FROM relationship
JOIN Character a
ON relationship.character1_id = a.id
JOIN Character b
ON relationship.character2_id = b.id;

/*This query will list all the characters, showing NULL for characters if a book has none by using LEFT OUTER JOIN operation.*/
SELECT book.book_name, character.name AS character_name
FROM book
LEFT OUTER JOIN character
ON book.id = character.book_id;

/*This query display each character name along with the name of the book they belong to.*/
SELECT name, book_name FROM character
JOIN book
ON character.book_id = book.id;

/*This is a self join on the character table. It shows pairs of characters who belong to the same book by matching their book_id. The condition character.id < a.id avoids duplicate pairings*/
SELECT character.name, a.name, character.book_id FROM character
JOIN character a
ON character.book_id = a.book_id
AND character.id < a.id;
```

Here I am attaching some screenshots of my SQL code.

```sql
--This code helps to understand the use of JOIN operation in SQL.
CREATE TABLE book (
id INTEGER PRIMARY KEY AUTOINCREMENT,
book_name TEXT,
series TEXT,
author TEXT);

INSERT INTO book (book_name, series, author) VALUES ("Shatter me", "Shatter me series", "Tahereh mafi");
INSERT INTO book (book_name, series, author) VALUES ("Haunted adeline", "Cat-mouse Duet", "H.D cartlon");
INSERT INTO book (book_name, series, author) VALUES ("Twisted love", "Twisted series", "Ana huang");
INSERT INTO book (book_name, series, author) VALUES ("The predator", "The dark verse", "Runyx");
INSERT INTO book (book_name, series, author) VALUES ("Twisted Games", "Twisted series", "Ana huang");
INSERT INTO book (book_name, series, author) VALUES ("It start with us", "It ends with us", "Collen Hoover");
INSERT INTO book (book_name, series, author) VALUES ("God of Ruin", "Legacy of gods", "Rina kent");
INSERT INTO book (book_name, series, author) VALUES ("Harry potter and the philospher stone", "Harry potter series", "J.K rowling");

CREATE TABLE character (
```

DATABASE SCHEMA

book	8 rows
id (PK)	INTEGER
book_name	TEXT
series	TEXT
author	TEXT

character	18 rows
id (PK)	INTEGER
name	TEXT
book_id	INTEGER
gender	TEXT

relationship	10 rows
id (PK)	INTEGER
character1_id	INTEGER
character2_id	INTEGER
relation	TEXT

QUERY RESULTS

```sql
VALUES (15, 18, "Lovers");

SELECT * FROM book;
SELECT * FROM character;
SELECT * FROM relationship;

SELECT a.name, b.name, relationship.relation FROM relationship
JOIN Character a
ON relationship.character1_id = a.id
JOIN Character b
ON relationship.character2_id = b.id;

SELECT book.book_name, character.name AS character_name
FROM book
LEFT OUTER JOIN character
ON book.id = character.book_id;

SELECT name, book_name FROM character
JOIN book
ON character.book_id = book.id;

SELECT character.name, a.name, character.book_id FROM character
JOIN character a
ON character.book_id = a.book_id
AND character.id < a.id;
```

QUERY RESULTS

id	book_name	series	author
1	Shatter me	Shatter me series	Tahereh mafi
2	Haunted adeline	Cat-mouse Duet	H.D cartlon
3	Twisted love	Twisted series	Ana huang
4	The predator	The dark verse	Runyx
5	Twisted Games	Twisted series	Ana huang
6	It start with us	It ends with us	Collen Hoover
7	God of Ruin	Legacy of gods	Rina kent

name	name	relation
Zade Meadows	Adeline Reilly	Stalker
Juilette Ferrars	Aaron Warner	Lovers
Morana Vitalio	Dante Maroni	Friends
Josh chen	Ava chen	Siblings
Morana Vitalio	Tristian Caine	Lovers
Ryle larsen	Bright von Ascheberg	Bodyguard
Lily Bloom	Atlas corrigon	Childhood Lovers
Stella alonso	Ava chen	Best-friends
Cecily Knight	Glydon Grace	Best-friends
Cecily Knight	Jeremy Volkov	Lovers

book_name	character_name
Shatter me	Aaron Warner
Shatter me	Juilette Ferrars
Haunted adeline	Adeline Reilly
Haunted adeline	Zade Meadows
Twisted love	Ava chen
Twisted love	Josh chen
Twisted love	Stella alonso
The predator	Dante Maroni
The predator	Morana Vitalio
The predator	Tristian Caine
Twisted Games	Bright von Ascheberg
Twisted Games	Christian Harper

God of Ruin	Jeremy Volkov
Harry potter and the philospher stone	NULL

name	book_name
Zade Meadows	Haunted adeline
Juilette Ferrars	Shatter me
Adeline Reilly	Haunted adeline
Josh chen	Twisted love
Morana Vitalio	The predator
Aaron Warner	Shatter me
Tristian Caine	The predator
Ava chen	Twisted love
Dante Maroni	The predator

name	name	book_id
Zade Meadows	Adeline Reilly	2
Juilette Ferrars	Aaron Warner	1
Josh chen	Ava chen	3
Josh chen	Stella alonso	3
Morana Vitalio	Dante Maroni	4
Morana Vitalio	Tristian Caine	4
Tristian Caine	Dante Maroni	4
Ava chen	Stella alonso	3
Ryle larsen	Bright von Ascheberg	5
Ryle larsen	Christian Harper	5
Lily Bloom	Atlas corrigon	6
Bright von Ascheberg	Christian Harper	5
Cecily Knight	Glydon Grace	7

Conclusion :

This database organizes books, their characters, and relationships between characters. It shows which characters belong to which books and how they are connected, like friends or lovers. This helps to understand the story and character links in different book series easily.