**Introduction**

The schema encompasses seven tables:

* **Users Table:** This table serves as the cornerstone of the database, storing essential user details including login credentials, first and last names, and the date of user registration. It is crucial for maintaining accurate and comprehensive user profiles.
* **Friends Table:** This table captures the social connections between users by recording friendship relationships. Each entry represents a unique friendship, identified by the users' login details, allowing for efficient tracking of user networks.
* **Entries Table:** This table logs user interactions and activities within the system, with each record detailing the time of the activity and the user involved. It provides a detailed chronological account of user engagement, supporting robust data analysis and user behavior tracking.
* **Comments Table:** This table stores comments made by users on various entries, including the content of the comment and the timestamp. Each comment is linked to a specific entry and user, facilitating detailed interaction records.
* **Links Table:** This table stores external links shared by users, providing a way to reference external resources.
* **Images Table:** This table stores images uploaded by users, along with optional captions, linked to specific entries.
* **Blurbs Table:** This table stores short text updates or blurbs posted by users, linked to specific entries.

**Users Table:**

**Login**  unique identifier for each user, primary login credential.  
**firstName** The first name of the user.  
**lastName** The last name of the user.  
**password** The user's password for authentication.  
**dateJoined**  The date when the user registered on the system.

**Friends Table:**

|  |  |
| --- | --- |
| **user** | The login identifier of the user. |
| **friend** | The login identifier of the user's friend. |
| **sinceDate** | The date when the friendship was established. |

**Entries Table:**

|  |  |
| --- | --- |
| **entryID** | The unique identifier for each entry. |

|  |  |
| --- | --- |
| **user** | The login identifier of the user who made the entry. |

|  |  |
| --- | --- |
| **time** | The timestamp of when the entry was made. |

**Comments Table:**

|  |  |
| --- | --- |
| **entry** | The unique identifier of the comment, referencing the entry it belongs to. |

|  |  |
| --- | --- |
| **parentEntry** | The unique identifier of the parent comment, if any. |

|  |  |
| --- | --- |
| **rootEntry** | The unique identifier of the root entry. |

|  |  |
| --- | --- |
| **text** | The content of the comment. |

**Links Table:**

|  |  |
| --- | --- |
| **user** | The login identifier of the user sharing the link. |

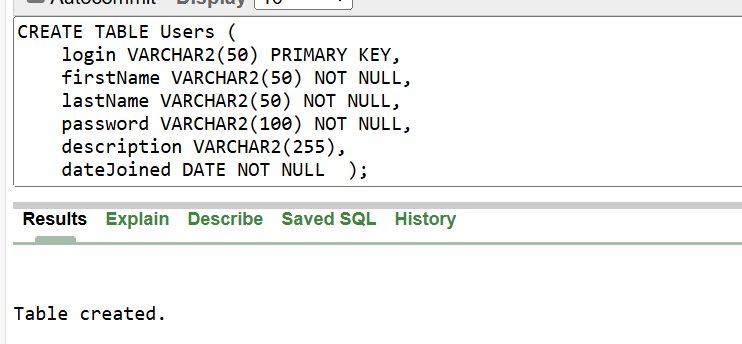
|  |  |
| --- | --- |
| **linkNo** | The unique identifier for each link. |

|  |  |
| --- | --- |
| **link** | The URL of the shared link. |

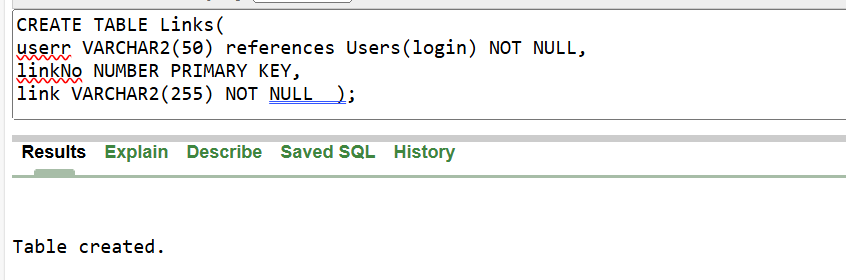
**Blurbs Table:**

|  |  |
| --- | --- |
| **entry** | The unique identifier of the blurb, referencing the entry it belongs to. |
| **text** | The content of the blurb. |

**Creating Tables** :  
  
**1. Table Users**   
  
CREATE TABLE Users (  
 login VARCHAR2(50) PRIMARY KEY,   
 firstName VARCHAR2(50) NOT NULL,   
 lastName VARCHAR2(50) NOT NULL,   
 password VARCHAR2(100) NOT NULL,   
 description VARCHAR2(255),   
 dateJoined DATE NOT NULL );

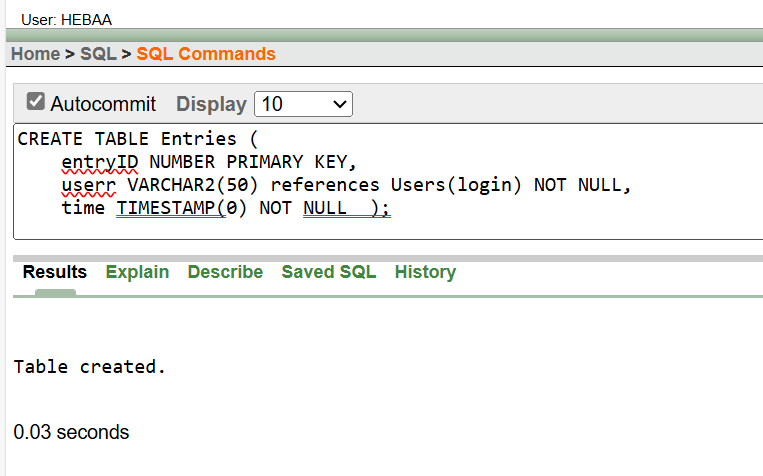
  
  
**2. Table Links**

CREATE TABLE Links(  
userr VARCHAR2(50) references Users(login) NOT NULL,   
linkNo NUMBER PRIMARY KEY,   
link VARCHAR2(255) NOT NULL );



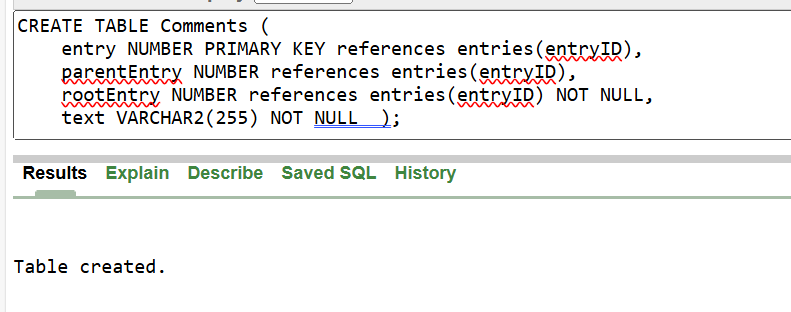
**3. Table Entries**

CREATE TABLE Entries (  
entryID NUMBER PRIMARY KEY,  
userr VARCHAR2(50) references Users(login) NOT NULL,   
time TIMESTAMP(0) NOT NULL );



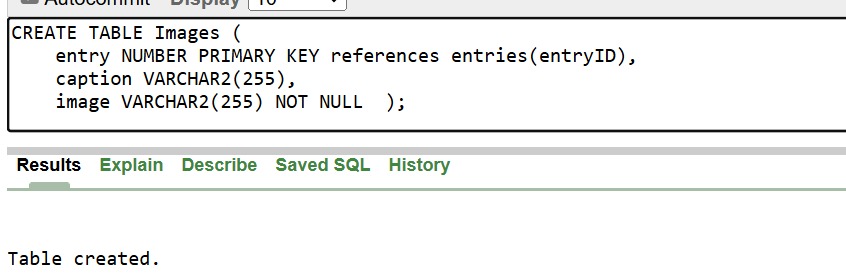
**4. Table Comments**

CREATE TABLE Comments (  
entry NUMBER PRIMARY KEY references entries(entryID),  
parentEntry NUMBER references entries(entryID),   
rootEntry NUMBER references entries(entryID) NOT NULL,   
text VARCHAR2(255) NOT NULL );



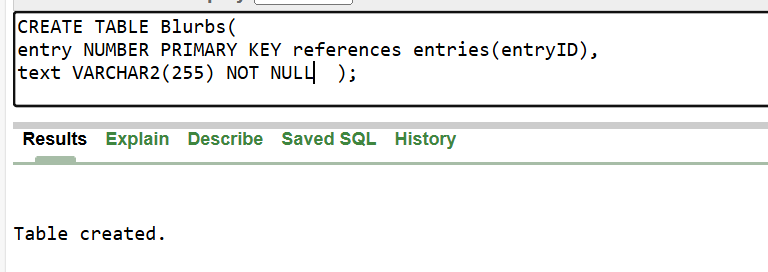
**5. Table Images**

**CREATE TABLE Images (  
entry NUMBER PRIMARY KEY references entries(entryID),  
caption VARCHAR2(255),  
image VARCHAR2(255) NOT NULL );**

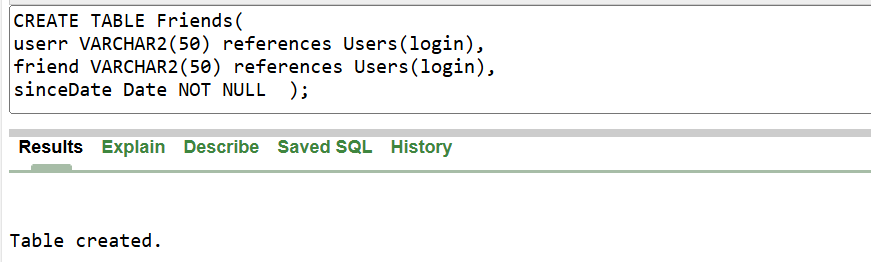
****

**6. Table Blurbs**

**CREATE TABLE Blurbs(  
entry NUMBER PRIMARY KEY references entries(entryID),  
text VARCHAR2(255) NOT NULL );**

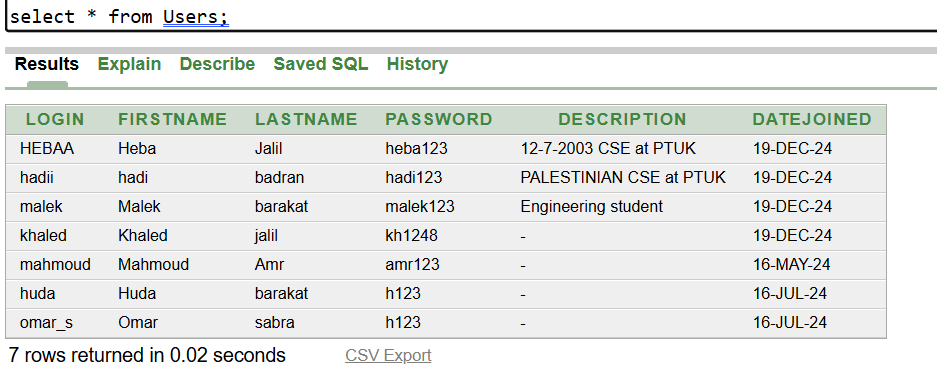
****

**7. Table Friends**

**CREATE TABLE Friends(  
userr VARCHAR2(50) references Users(login),   
friend VARCHAR2(50) references Users(login),  
sinceDate Date NOT NULL );  
**

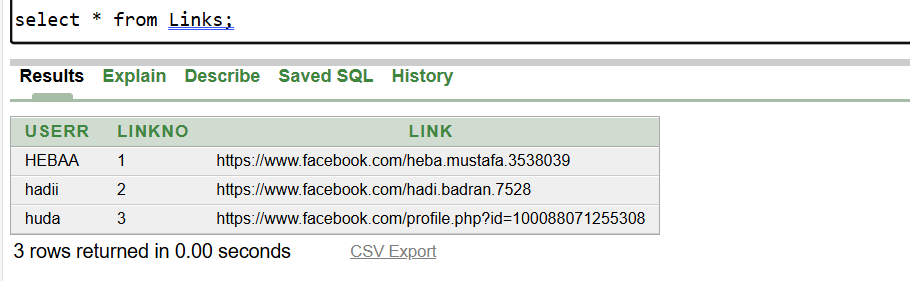
**INSERT INTO TABLES**   
**1. TABLE Users**

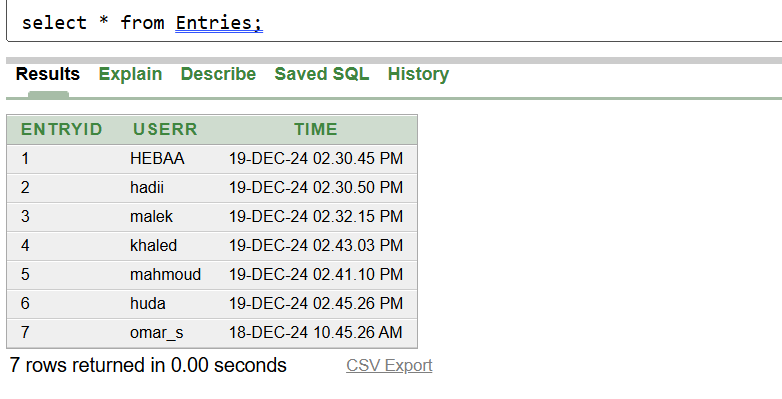
INSERT INTO Users  
VALUES (  
 'HEBAA', 'Heba', 'Jalil', 'heba123', '12-7-2003 CSE at PTUK',DATE '2024-12-19' );  
INSERT INTO Users  
VALUES (  
 'hadii', 'hadi', 'badran', 'hadi123', 'PALESTINIAN CSE at PTUK', DATE '2024-12-19');  
INSERT INTO Users  
VALUES (  
 'malek', 'Malek', 'barakat', 'malek123', 'Engineering student', DATE '2024-12-19' );  
INSERT INTO Users (login, firstName, lastName, password, dateJoined)  
VALUES (  
 'khaled', 'Khaled', 'jalil', 'kh1248', DATE '2024-12-19' );  
INSERT INTO Users (login, firstName, lastName, password, dateJoined)  
VALUES (  
 'mahmoud', 'Mahmoud', 'Amr', 'amr123',DATE '2024-05-16' );  
INSERT INTO Users (login, firstName, lastName, password, dateJoined)  
VALUES (  
 'huda','Huda','barakat','h123',DATE '2024-07-16' );  
INSERT INTO Users (login, firstName, lastName, password, dateJoined)  
VALUES (  
 'omar\_s','Omar', 'sabra', 'h123',DATE '2024-07-16' );



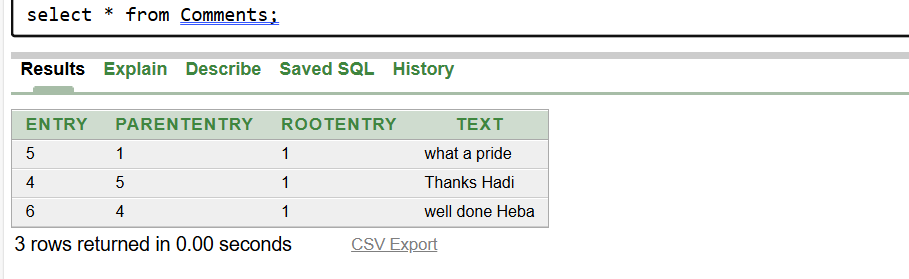
**2. Table Links**

INSERT INTO Links  
VALUES (  
 'HEBAA',1,'https://www.facebook.com/heba.mustafa.3538039');  
INSERT INTO Links   
VALUES (  
 'hadii', 2, 'https://www.facebook.com/hadi.badran.7528');  
INSERT INTO Links   
VALUES (  
 'huda', 3, 'https://www.facebook.com/profile.php?id=100088071255308');



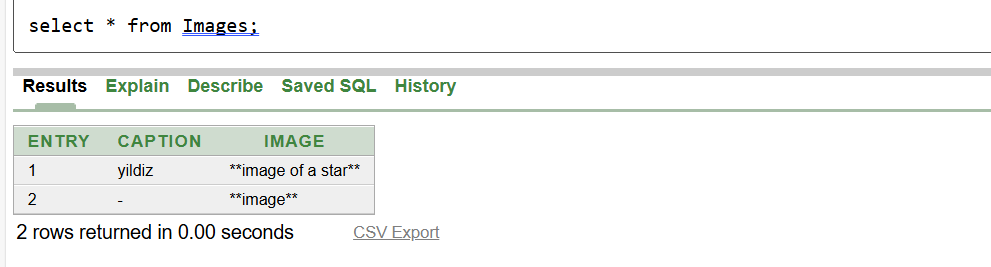
**3. Table Entries  
INSERT INTO Entries (entryID, userr, time) VALUES (1, 'HEBAA', TO\_DATE('2024-12-19 14:30:45', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (2, 'hadii', TO\_DATE('2024-12-19 14:30:50', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (3, 'malek', TO\_DATE('2024-12-19 14:32:15', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (4, 'khaled', TO\_DATE('2024-12-19 14:43:03', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (5, 'mahmoud', TO\_DATE('2024-12-19 14:41:10', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (6, 'huda', TO\_DATE('2024-12-19 14:45:26', 'YYYY-MM-DD HH24:MI:SS'));  
INSERT INTO Entries (entryID, userr, time) VALUES (7, 'omar\_s', TO\_DATE('2024-12-18 10:45:26', 'YYYY-MM-DD HH24:MI:SS'));  
**

**4. Table Comments**

INSERT INTO Comments VALUES (5, 1, 1, 'what a pride!');  
INSERT INTO Comments VALUES (4, 5, 1, 'Thanks Hadi');  
INSERT INTO Comments VALUES (6, 4, 1, 'well done Heba');  


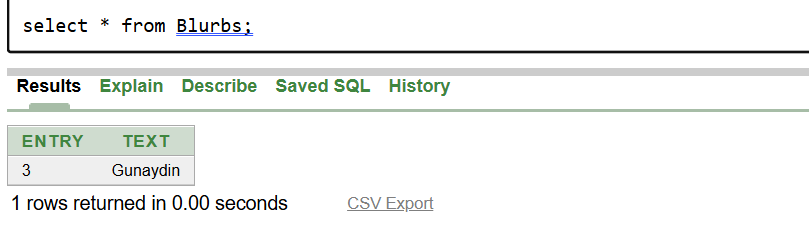
**5. Table Images**

**INSERT INTO Images VALUES (1, 'yildiz', '\*\*image of a star\*\*');  
INSERT INTO Images (entry, image) VALUES (2, '\*\*image\*\*');**



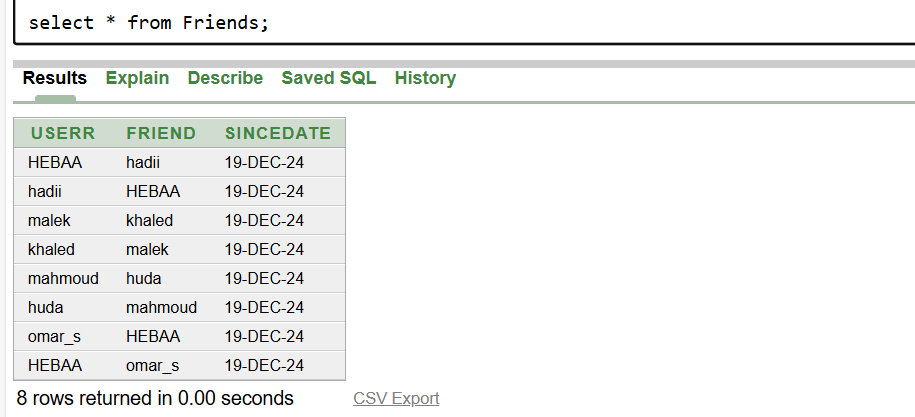
**6. Table Blurbs**

**INSERT INTO Blurbs (entry, text) VALUES (3, 'Gunaydin');**



**7. Table Friends**

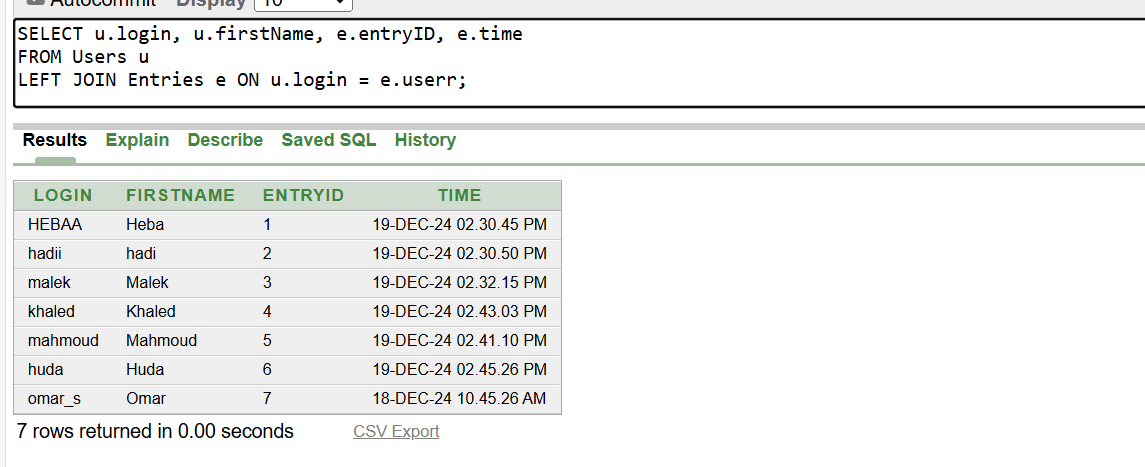
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('HEBAA', 'hadii', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('hadii', 'HEBAA', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('malek', 'khaled', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('khaled', 'malek', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('mahmoud', 'huda', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('huda', 'mahmoud', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('omar\_s', 'HEBAA', DATE '2024-12-19');  
INSERT INTO Friends (userr, friend, sinceDate) VALUES ('HEBAA', 'omar\_s', DATE '2024-12-19');



**3) Based on the inserted data, you need to provide examples that clarifies the following SQL joins concepts: left outer join, right outer join, and full outer join.**

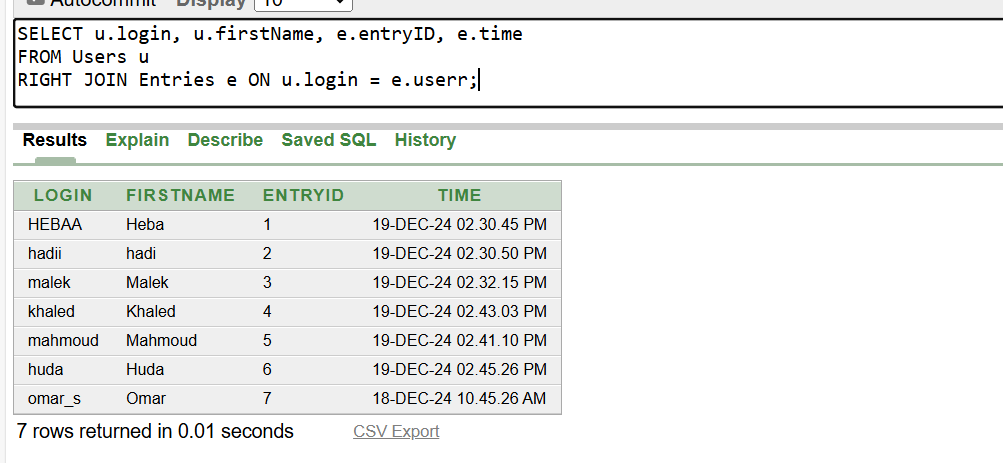
**Question: Retrieve all users and their corresponding entry records, if any. Include users even if they don't have any entries.**

**SELECT u.login, u.firstName, e.entryID, e.time  
FROM Users u  
LEFT JOIN Entries e ON u.login = e.userr;**

****

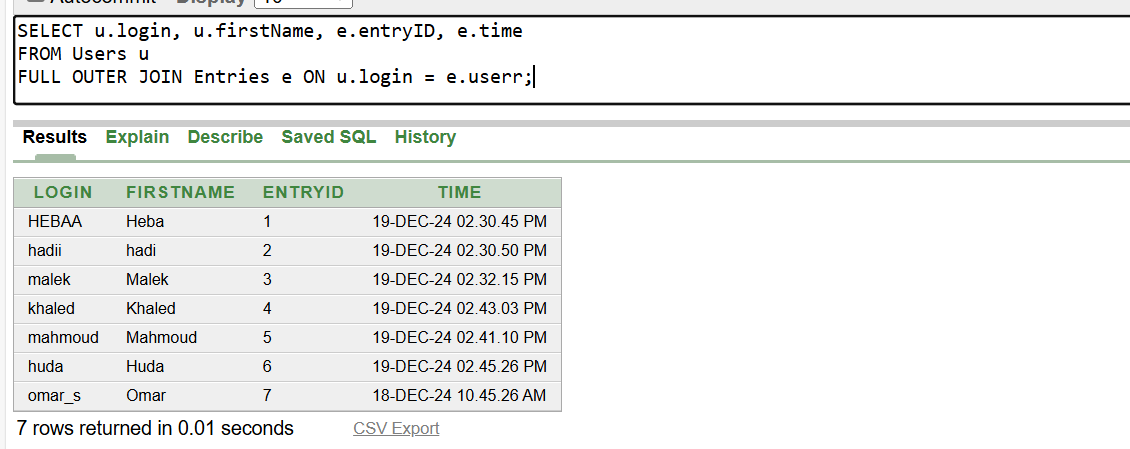
**Question: Retrieve all entry records and their corresponding users, if any. Include entries even if they don't have a corresponding user.**

**SELECT u.login, u.firstName, e.entryID, e.time  
FROM Users u  
RIGHT JOIN Entries e ON u.login = e.userr;**

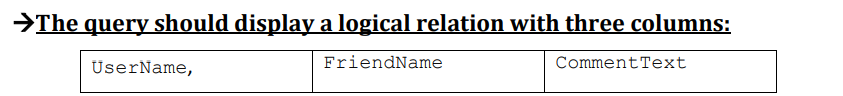
****

**Question: Retrieve all users and entry records. Include users without entries and entries without users.**

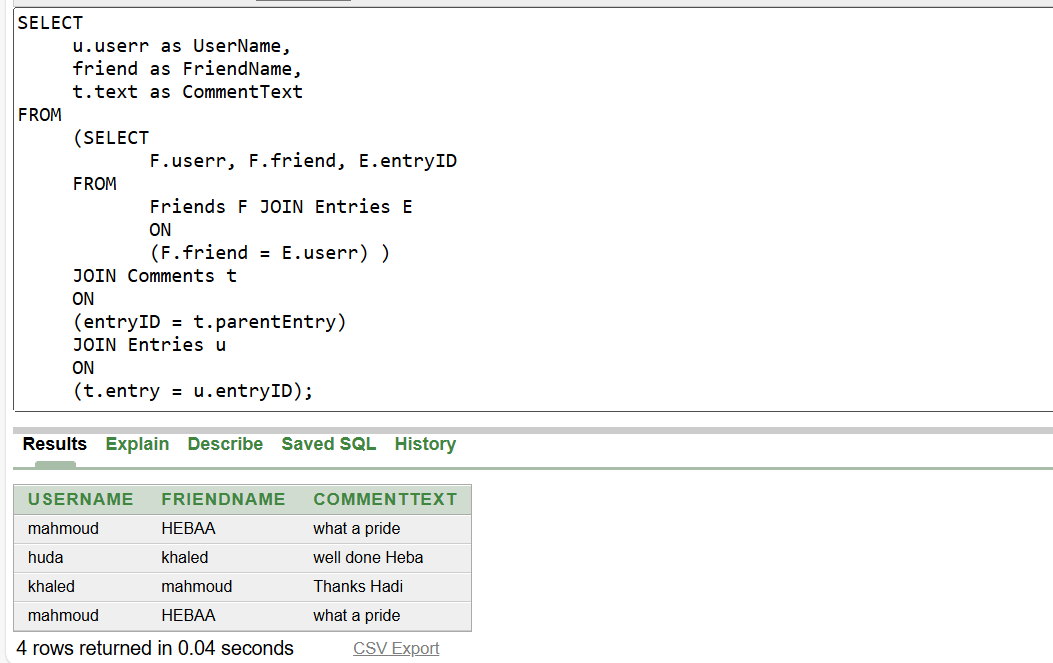
**SELECT u.login, u.firstName, e.entryID, e.time  
FROM Users u  
FULL OUTER JOIN Entries e ON u.login = e.userr;**

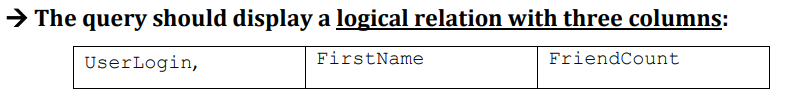
****

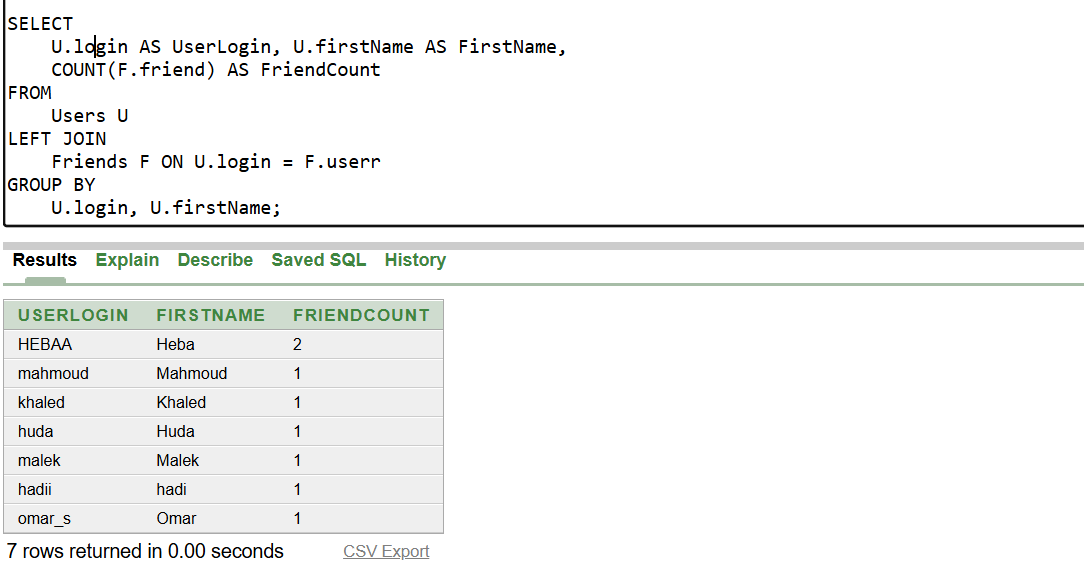
**4) Write an SQL to retrieve the names of users who have posted comments on entries made by their friends. Use a subquery to identify the entries created by each user's friends.**

****

**SELECT   
 u.userr as UserName,  
 friend as FriendName,  
 t.text as CommentText  
FROM  
 (SELECT   
 F.userr, F.friend, E.entryID  
 FROM   
 Friends F JOIN Entries E  
 ON   
 (F.friend = E.userr) )  
 JOIN Comments t  
 ON  
 (entryID = t.parentEntry)  
 JOIN Entries u  
 ON   
 (t.entry = u.entryID);**

****

**5) Write an SQL query to display the users and the number of friends for each user. Use the COUNT aggregate function with GROUP BY.  
SELECT   
 U.login AS UserLogin, U.firstName AS FirstName,   
 COUNT(F.friend) AS FriendCount  
FROM   
 Users U  
LEFT JOIN   
 Friends F ON U.login = F.userr  
GROUP BY   
 U.login, U.firstName;**

****