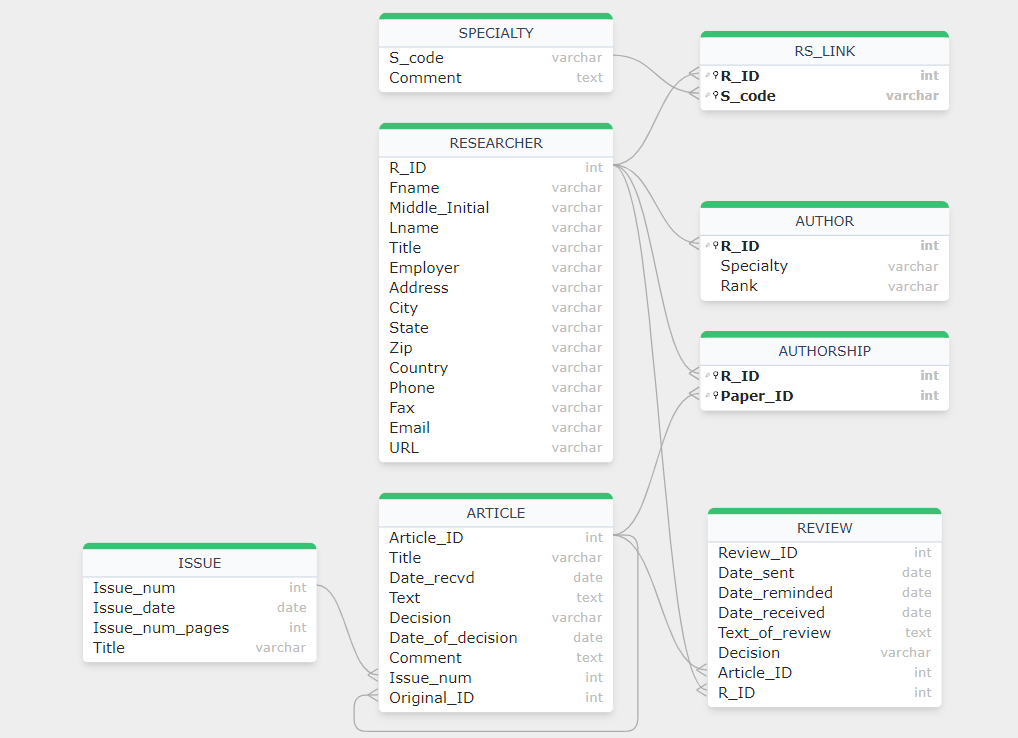
**Part 1 : ER Diagram**

****

**Part 2: Database Implementation and Data Population**

**(35 points)**

CREATE TABLE ARTICLE (

    Article\_ID INT PRIMARY KEY,

    Title VARCHAR(255),

    Date\_recvd DATE,

    Text TEXT,

    Decision VARCHAR(100),

    Date\_of\_decision DATE,

    comment TEXT,

    Issue\_num INT,

    Original\_ID INT,

    FOREIGN KEY (Issue\_num) REFERENCES ISSUE(Issue\_num),

    FOREIGN KEY (Original\_ID) REFERENCES ARTICLE(Article\_ID)

);

CREATE TABLE REVIEW (

    Review\_ID INT PRIMARY KEY,

    Date\_sent DATE,

    Date\_reminded DATE,

    Date\_received DATE,

    Text\_of\_review TEXT,

    Decision VARCHAR(100),

    Article\_ID INT,

    R\_ID INT,

    FOREIGN KEY (Article\_ID) REFERENCES ARTICLE(Article\_ID),

    FOREIGN KEY (R\_ID) REFERENCES RESEARCHER(R\_ID)

);

CREATE TABLE RESEARCHER (

    R\_ID INT PRIMARY KEY,

    Fname VARCHAR(100),

    Middle\_Initial CHAR(1),

    Lname VARCHAR(100),

    Title VARCHAR(100),

    Employer VARCHAR(255),

    Address VARCHAR(255),

    City VARCHAR(100),

    State VARCHAR(100),

    Zip VARCHAR(10),

    Country VARCHAR(100),

    Phone VARCHAR(15),

    Fax VARCHAR(15),

    Email VARCHAR(100),

    URL VARCHAR(255)

);

CREATE TABLE AUTHOR (

    R\_ID INT PRIMARY KEY,

    Specialty VARCHAR(255),

    Rank VARCHAR(100),

    FOREIGN KEY (R\_ID) REFERENCES RESEARCHER(R\_ID)

);

CREATE TABLE ISSUE (

    Issue\_num INT PRIMARY KEY,

    Issue\_date DATE,

    Issue\_num\_pages INT,

    Title VARCHAR(255)

);

CREATE TABLE SPECIALTY (

    S\_code INT PRIMARY KEY,

    comment TEXT

);

CREATE TABLE RS\_LINK (

    R\_ID INT,

    S\_code INT,

    PRIMARY KEY (R\_ID, S\_code),

    FOREIGN KEY (R\_ID) REFERENCES RESEARCHER(R\_ID),

    FOREIGN KEY (S\_code) REFERENCES SPECIALTY(S\_code)

);

CREATE TABLE AUTHORSHIP (

    R\_ID INT,

    Paper\_ID INT,

    PRIMARY KEY (R\_ID, Paper\_ID),

    FOREIGN KEY (R\_ID) REFERENCES RESEARCHER(R\_ID),

    FOREIGN KEY (Paper\_ID) REFERENCES ARTICLE(Article\_ID)

);

**Output :**





**Part 3: SQL Queries (45 points)**

**1. List the titles of all articles received in 2023.**

SELECT Title FROM ARTICLE WHERE YEAR(Date\_recvd) = 2023;

**Output :**



**2.Find the names of all researchers who have written an article about ‘Database Systems’.**

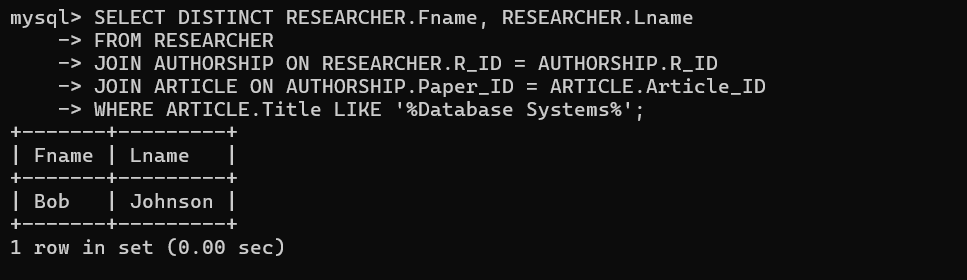
SELECT DISTINCT RESEARCHER.Fname, RESEARCHER.Lname

FROM RESEARCHER

JOIN AUTHORSHIP ON RESEARCHER.R\_ID = AUTHORSHIP.R\_ID

JOIN ARTICLE ON AUTHORSHIP.Paper\_ID = ARTICLE.Article\_ID

WHERE ARTICLE.Title LIKE '%Database Systems%';

****

**3.List the articles reviewed by ‘John Doe’.**

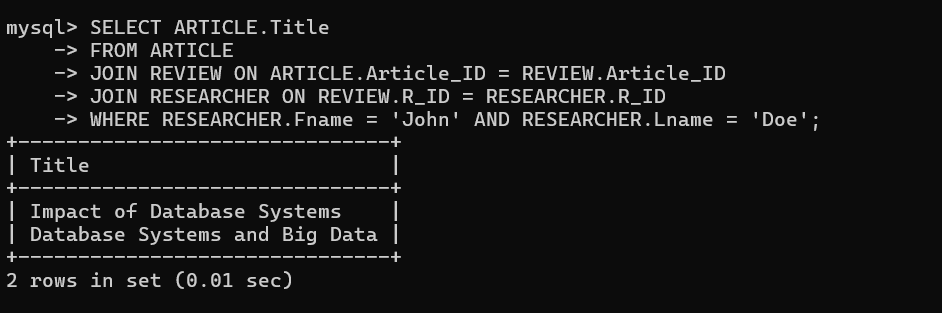
SELECT ARTICLE.Title

FROM ARTICLE

JOIN REVIEW ON ARTICLE.Article\_ID = REVIEW.Article\_ID

JOIN RESEARCHER ON REVIEW.R\_ID = RESEARCHER.R\_ID

WHERE RESEARCHER.Fname = 'John' AND RESEARCHER.Lname = 'Doe';

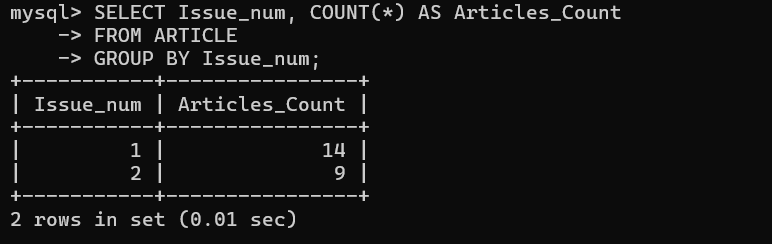
****

**4.Find the number of articles received by each issue.**

SELECT Issue\_num, COUNT(\*) AS Articles\_Count

FROM ARTICLE

GROUP BY Issue\_num;

****

**5.List the authors who have written more than 5 articles.**

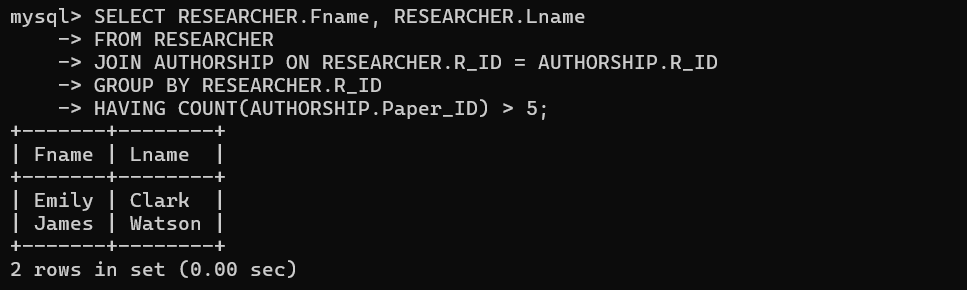
SELECT RESEARCHER.Fname, RESEARCHER.Lname

FROM RESEARCHER

JOIN AUTHORSHIP ON RESEARCHER.R\_ID = AUTHORSHIP.R\_ID

GROUP BY RESEARCHER.R\_ID

HAVING COUNT(AUTHORSHIP.Paper\_ID) > 5;

****

**6.Find the most common specialty among authors.**

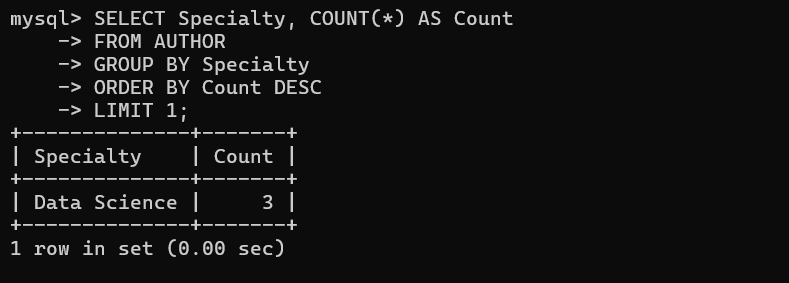
SELECT Specialty, COUNT(\*) AS Count

FROM AUTHOR

GROUP BY Specialty

ORDER BY Count DESC

LIMIT 1;

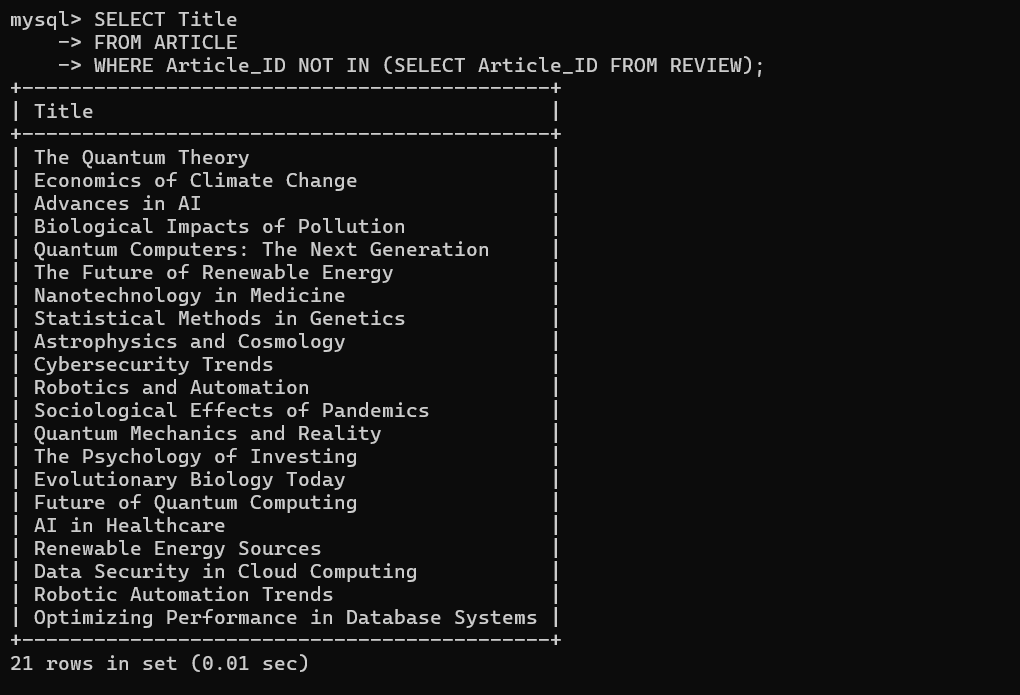


**7.List the articles that have not yet been reviewed.**

SELECT Title

FROM ARTICLE

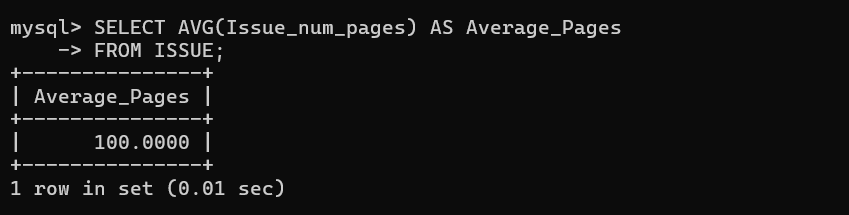
WHERE Article\_ID NOT IN (SELECT Article\_ID FROM REVIEW);

****

**8. Find the average number of pages per issue.**

SELECT AVG(Issue\_num\_pages) AS Average\_Pages

FROM ISSUE;



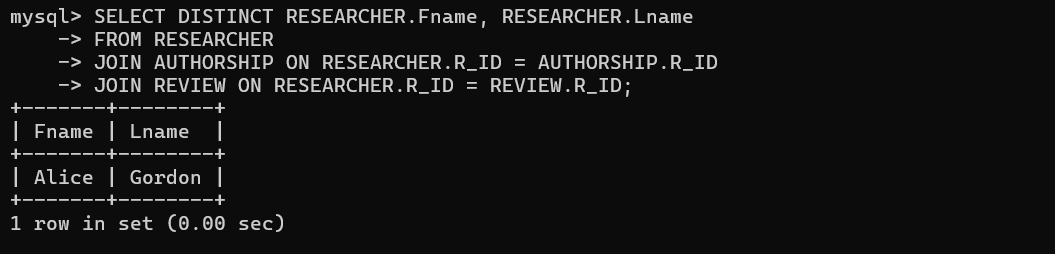
**9.List the researchers who have both written and reviewed articles.**

SELECT DISTINCT RESEARCHER.Fname, RESEARCHER.Lname

FROM RESEARCHER

JOIN AUTHORSHIP ON RESEARCHER.R\_ID = AUTHORSHIP.R\_ID

JOIN REVIEW ON RESEARCHER.R\_ID = REVIEW.R\_ID;



**10. Find the titles of articles that have been both accepted and rejected.**

SELECT Title

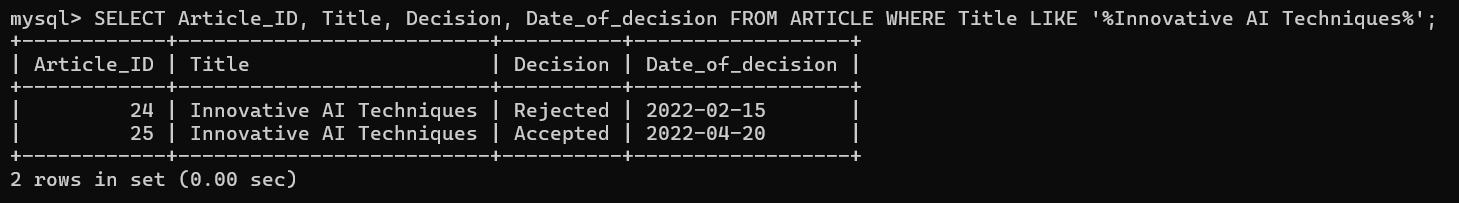
FROM ARTICLE

WHERE Article\_ID IN

    (SELECT Article\_ID FROM ARTICLE WHERE Decision = 'Accepted')

AND Article\_ID IN

    (SELECT Article\_ID FROM ARTICLE WHERE Decision = 'Rejected');



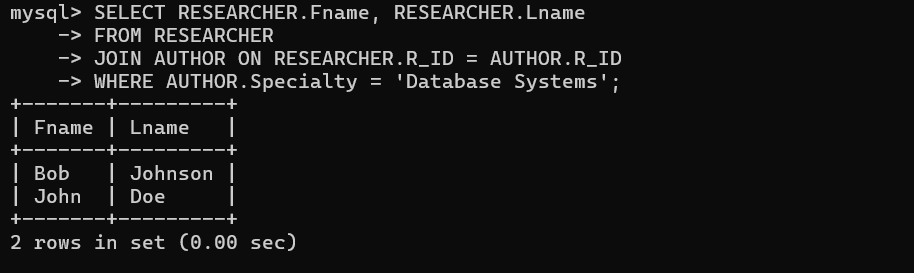
**11. List the researchers who have a specialty in ‘Database Systems’.**

SELECT RESEARCHER.Fname, RESEARCHER.Lname

FROM RESEARCHER

JOIN AUTHOR ON RESEARCHER.R\_ID = AUTHOR.R\_ID

WHERE AUTHOR.Specialty = 'Database Systems';

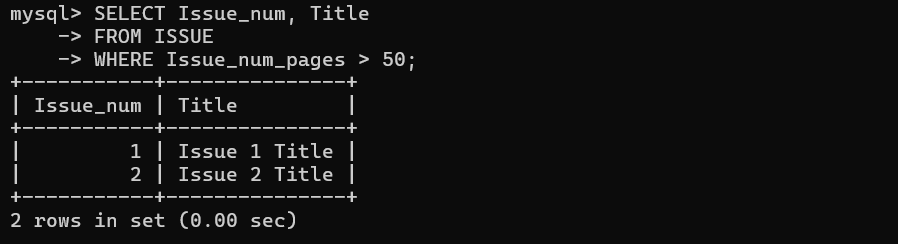


**12. Find the issues that have more than 50 pages.**

SELECT Issue\_num, Title

FROM ISSUE

WHERE Issue\_num\_pages > 50;



**13. List the articles that have been reviewed more than twice**

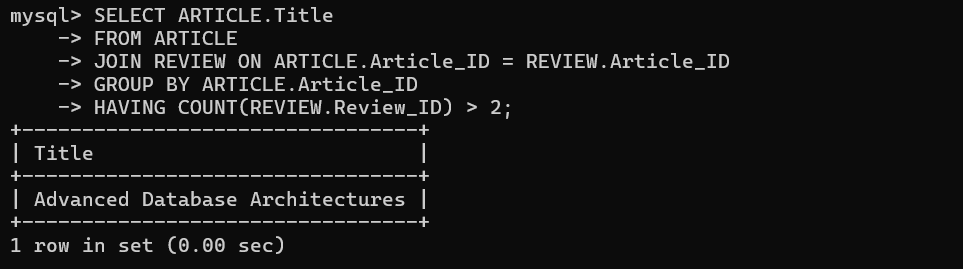
SELECT ARTICLE.Title

FROM ARTICLE

JOIN REVIEW ON ARTICLE.Article\_ID = REVIEW.Article\_ID

GROUP BY ARTICLE.Article\_ID

HAVING COUNT(REVIEW.Review\_ID) > 2;



**14. Find the researcher with the most articles.**

SELECT RESEARCHER.Fname, RESEARCHER.Lname

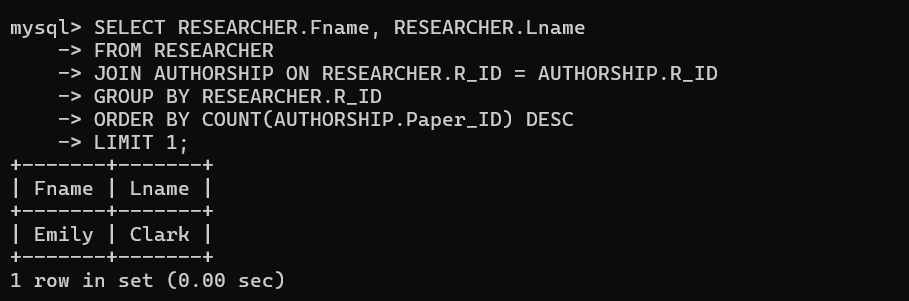
FROM RESEARCHER

JOIN AUTHORSHIP ON RESEARCHER.R\_ID = AUTHORSHIP.R\_ID

GROUP BY RESEARCHER.R\_ID

ORDER BY COUNT(AUTHORSHIP.Paper\_ID) DESC

LIMIT 1;

****

**15. List the specialties that have more than 10 researchers.**

SELECT Specialty, COUNT(\*) AS Researcher\_Count

FROM AUTHOR

GROUP BY Specialty

HAVING COUNT(\*) > 10;

