University Institute of Engineering

Department of Computer Science & Engineering

EXPERIMENT : 1

NAME : SAGEN HANSDA UID: 23BCS12396

BRANCH : BE-CSE SECTION/GROUP : KRG_1A

SEMESTER : 5TH SUBJECT CODE : 23CSP-333

SUBJECT NAME : ADBMS

1. Aim Of The Practical :

Author-Book Relationship Using Joins and Basic SQL Operations [EASY]

1. Design two tables — one for storing author details and the other for book details.
2. Ensure a foreign key relationship from the book to its respective author.
3. Insert at least three records in each table.
4. Perform an INNER JOIN to link each book with its author using the common author ID.
5. Select the book title, author name, and author’s country.

Department-Course Subquery and Access Control. [MEDIUM]

1. Design normalized tables for departments and the courses they offer, maintaining a foreign key relationship.
2. Insert five departments and at least ten courses across those departments.
3. Use a subquery to count the number of courses under each department.
4. Filter and retrieve only those departments that offer more than two courses.
5. Grant SELECT-only access on the courses table to a specific user.

2. Tools Used: SQL Server Management Studio

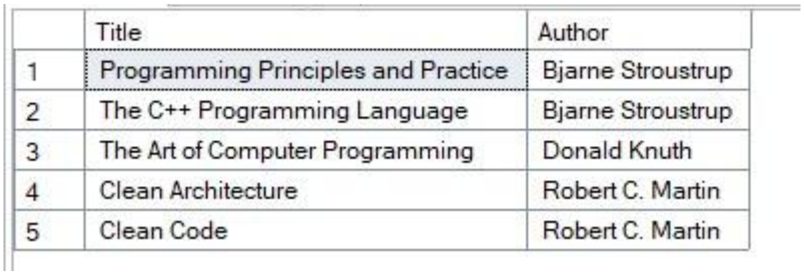
3. Code:

```
--easy question
 CREATE TABLE TBL_AUTHOR (
    AUTHOR_ID INT PRIMARY KEY,
    AUTHOR_NAME VARCHAR(30)
);
```

```sql
CREATE TABLE TBL_BOOK (
    BOOK_ID INT PRIMARY KEY,
    BOOK_TITLE VARCHAR(50),
    AUTHOR_ID INT,
    FOREIGN KEY (AUTHOR_ID) REFERENCES TBL_AUTHOR(AUTHOR_ID)
);

INSERT INTO TBL_AUTHOR (AUTHOR_ID, AUTHOR_NAME) VALUES
(1, 'Robert C. Martin'),
(2, 'Bjarne Stroustrup'),
(3, 'Donald Knuth');

INSERT INTO TBL_BOOK (BOOK_ID, BOOK_TITLE, AUTHOR_ID) VALUES
(101, 'Clean Code', 1),
(102, 'The C++ Programming Language', 2),
(103, 'The Art of Computer Programming', 3),
(104, 'Clean Architecture', 1),
(105, 'Programming Principles and Practice', 2);
 SELECT
    book.BOOK_TITLE AS Title,
author.AUTHOR_NAME AS Author
FROM
    TBL_BOOK AS book
INNER JOIN
    TBL_AUTHOR AS author ON book.AUTHOR_ID = author.AUTHOR_ID
ORDER BY
    Author, Title;



--medium question
CREATE TABLE University_Branches (
branch_code INT PRIMARY KEY,      branch_title
VARCHAR(100) NOT NULL
);

CREATE TABLE Class_Listings (
class_id INT PRIMARY KEY,
class_subject VARCHAR(100) NOT NULL,
    branch_code INT,
    FOREIGN KEY (branch_code) REFERENCES University_Branches(branch_code)
);

INSERT INTO University_Branches (branch_code, branch_title) VALUES
(1, 'Information Technology'),
(2, 'Aerospace Engineering'),
(3, 'Chemical Engineering'),
(4, 'Architecture'),
(5, 'Statistics');

INSERT INTO Class_Listings (class_id, class_subject, branch_code) VALUES
(101, 'Web Development', 1),
(102, 'Database Management', 1),
(103, 'Cybersecurity Fundamentals', 1),
(104, 'Aerodynamics', 2),
(105, 'Spacecraft Design', 2),
(106, 'Organic Chemistry', 3),
(107, 'Process Control', 3),
(108, 'Urban Planning', 4),
(109, 'Statistical Inference', 5),
(110, 'Regression Analysis', 5),
(111, 'Time Series Analysis', 5);


 SELECT
    branch.branch_title,
    COUNT(listing.class_id) AS number_of_classes
FROM
    University_Branches AS branch
LEFT JOIN
    Class_Listings AS listing ON branch.branch_code = listing.branch_code
GROUP BY
    branch.branch_title ORDER
BY
    branch.branch_title;
 SELECT
    branch.branch_title,
    COUNT(listing.class_id) AS class_count
FROM
    University_Branches AS branch
JOIN
    Class_Listings AS listing ON branch.branch_code = listing.branch_code
GROUP BY
    branch.branch_title
HAVING
    COUNT(listing.class_id) > 2
ORDER BY
branch.branch_title;
```

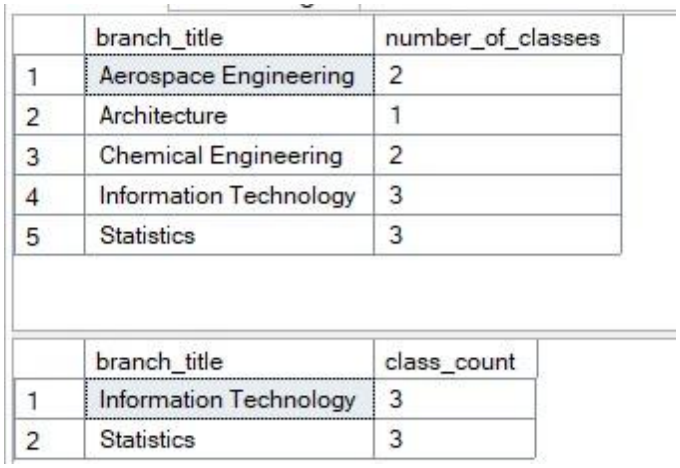
4. Output :

[EASY]



	Title	Author
1	Programming Principles and Practice	Bjarne Stroustrup
2	The C++ Programming Language	Bjarne Stroustrup
3	The Art of Computer Programming	Donald Knuth
4	Clean Architecture	Robert C. Martin
5	Clean Code	Robert C. Martin

[MEDIUM]



	branch_title	number_of_classes
1	Aerospace Engineering	2
2	Architecture	1
3	Chemical Engineering	2
4	Information Technology	3
5	Statistics	3

	branch_title	class_count
1	Information Technology	3
2	Statistics	3

5. Learning Outcomes:

- Learn to create and define relational database tables using the CREATE TABLE command, along with understanding common data types such as INT and VARCHAR.
- Build practical skills in setting up primary keys to ensure each record can be uniquely identified.
- Understand how to define and enforce foreign key constraints to preserve data consistency between linked tables (e.g., Books linked to Authors).
- Gain the ability to perform INNER JOIN operations to merge records from multiple tables using a shared key (such as author_id).
- Learn how to structure normalized relational schemas with foreign key relationships for real-world examples like departments and courses. • Become comfortable inserting several rows into related tables using the INSERT INTO statement.
- Master the use of subqueries alongside GROUP BY and HAVING to summarize and filter aggregated results.
- Apply query logic to select data from a parent table based on conditions derived from aggregated results in a related child table.