SQL Practice Workbook - Sriram Murali

---

1. WHERE Clause Practice

Practice 1: Simple WHERE
```
SELECT *
FROM employees
WHERE department = 'HR'
AND salary BETWEEN 40000 AND 80000;
```

Practice 2: WHERE with NOT IN
```
SELECT *
FROM employees
WHERE salary < 60000
AND department <> 'Finance';
```

Practice 3: WHERE with IN
```
SELECT *
FROM employees
WHERE salary IN (30000, 50000, 70000);
```

Practice 4: WHERE with NULL
```
SELECT *
FROM employees
WHERE department IS NULL;
```

Practice 5: WHERE combining AND and OR
```
SELECT *
FROM employees
WHERE department = 'HR'
OR salary > 90000;
```

Practice 6: WHERE with Parentheses
```
SELECT *
FROM employees
WHERE (department = 'Finance' OR department = 'HR')
AND salary > 60000;
```

Practice 7: LIKE Pattern Matching
```
SELECT *
FROM employees
WHERE emp_name LIKE 'S%';
```

---

2. INNER JOIN Practice

Practice 1: Students and Courses
```
SELECT s.student_name, c.course_name
FROM students AS s
INNER JOIN courses AS c
```

```
ON s.course_id = c.course_id;

Practice 2: Orders and Customers
SELECT o.order_id, c.customer_name
FROM orders AS o
INNER JOIN customers AS c
ON o.customer_id = c.customer_id;

Practice 3: Employees and Departments
SELECT e.emp_name, d.department_name
FROM employees AS e
INNER JOIN departments AS d
ON e.department_id = d.department_id;
```

---

3. LEFT JOIN Practice

```
Practice 1: Employees and Departments (keeping all employees)
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id;

Practice 2: Students and Courses (keeping all students)
SELECT s.student_name, c.course_name
FROM students AS s
LEFT JOIN courses AS c
ON s.course_id = c.course_id;

Practice 3: Employees without Department
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id
WHERE d.department_name IS NULL;
```

---

4. JOIN Cheat Sheet

```
INNER JOIN: Only matched rows from both tables
LEFT JOIN: All left rows + matched right rows (NULL if missing)
RIGHT JOIN: All right rows + matched left rows (rare)
FULL OUTER JOIN: Everything from both sides (matched and unmatched)
```

---

5. Subquery Practice

```
Subquery in WHERE Clause
SELECT emp_name, salary
FROM employees
WHERE salary > (SELECT AVG(salary) FROM employees);
```

```
SELECT emp_name, salary
FROM employees
WHERE department_id = (
    SELECT department_id
    FROM employees
    WHERE emp_name = 'Sriram'
);

SELECT emp_name, salary
FROM employees
WHERE salary = (
    SELECT MAX(salary)
    FROM employees
);

SELECT emp_name, salary
FROM employees
WHERE department_id IN (
    SELECT department_id
    FROM departments
    WHERE department_name IN ('Finance', 'HR')
);

Subquery in SELECT Clause
SELECT emp_name, salary,
       CASE
           WHEN salary > (SELECT AVG(salary) FROM employees) THEN 'Yes'
           ELSE 'No'
       END AS above_company_avg
FROM employees;

Subquery in FROM Clause (Derived Table)
SELECT department, num_of_employees, avg_salary
FROM (
    SELECT department, COUNT(emp_id) AS num_of_employees,
           AVG(salary) AS avg_salary
    FROM employees
    GROUP BY department
) AS dept_summary
WHERE num_of_employees < 5;

SELECT d.department_name, dept_summary.num_of_employees, dept_summary.avg_salary
FROM (
    SELECT department_id, COUNT(emp_id) AS num_of_employees, AVG(salary) AS avg_salary
    FROM employees
    GROUP BY department_id
) AS dept_summary
INNER JOIN departments d
ON dept_summary.department_id = d.department_id
WHERE dept_summary.avg_salary < 60000;

SELECT d.department_name, dept_summary.num_of_employees, dept_summary.max_salary
FROM (
```

```
    SELECT department_id, COUNT(emp_id) AS num_of_employees, MAX(salary) AS max_salary
    FROM employees
    GROUP BY department_id
) AS dept_summary
INNER JOIN departments d
ON dept_summary.department_id = d.department_id
WHERE dept_summary.max_salary > 90000;
```

Conclusion

You have now practiced:
- WHERE filters
- INNER & LEFT JOINs
- GROUP BY with aggregates
- CASE WHEN conditions
- Subqueries in WHERE, SELECT, FROM clauses

You're ready for real-world Data Analyst interviews and portfolio projects!

POWER BI

Situation	Relationship to Use	Notes
Star schema (Sales, Products, Customers)	One-to-Many (1:*)	Best practice
Shared dimension table (e.g., Date used for Orders and Returns)	One-to-Many + Role-playing
Many-to-many categories or tags	Use M:M **only if needed**	Prefer a **bridge table** whe
Lookup table with extra info	Consider merging instead	1:1 often doesn't justify a relati

SQL Practice Workbook - Sriram Murali

---

1. WHERE Clause Practice

Practice 1: Simple WHERE
```
SELECT *
FROM employees
WHERE department = 'HR'
AND salary BETWEEN 40000 AND 80000;
```

Practice 2: WHERE with NOT IN
```
SELECT *
FROM employees
WHERE salary < 60000
AND department <> 'Finance';
```

Practice 3: WHERE with IN
```
SELECT *
FROM employees
WHERE salary IN (30000, 50000, 70000);
```

Practice 4: WHERE with NULL
```
SELECT *
FROM employees
WHERE department IS NULL;
```

Practice 5: WHERE combining AND and OR
```
SELECT *
FROM employees
WHERE department = 'HR'
OR salary > 90000;
```

Practice 6: WHERE with Parentheses
```
SELECT *
FROM employees
WHERE (department = 'Finance' OR department = 'HR')
AND salary > 60000;
```

Practice 7: LIKE Pattern Matching
```
SELECT *
FROM employees
WHERE emp_name LIKE 'S%';
```

---

2. INNER JOIN Practice

Practice 1: Students and Courses
```
SELECT s.student_name, c.course_name
FROM students AS s
INNER JOIN courses AS c
```

```
ON s.course_id = c.course_id;

Practice 2: Orders and Customers
SELECT o.order_id, c.customer_name
FROM orders AS o
INNER JOIN customers AS c
ON o.customer_id = c.customer_id;

Practice 3: Employees and Departments
SELECT e.emp_name, d.department_name
FROM employees AS e
INNER JOIN departments AS d
ON e.department_id = d.department_id;
```

---

3. LEFT JOIN Practice

```
Practice 1: Employees and Departments (keeping all employees)
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id;

Practice 2: Students and Courses (keeping all students)
SELECT s.student_name, c.course_name
FROM students AS s
LEFT JOIN courses AS c
ON s.course_id = c.course_id;

Practice 3: Employees without Department
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id
WHERE d.department_name IS NULL;
```

---

4. JOIN Cheat Sheet

```
INNER JOIN: Only matched rows from both tables
LEFT JOIN: All left rows + matched right rows (NULL if missing)
RIGHT JOIN: All right rows + matched left rows (rare)
FULL OUTER JOIN: Everything from both sides (matched and unmatched)

Syntax Templates:

INNER JOIN
SELECT a.column1, b.column2
FROM tableA AS a
INNER JOIN tableB AS b
ON a.common_column = b.common_column;
```

```
LEFT JOIN
SELECT a.column1, b.column2
FROM tableA AS a
LEFT JOIN tableB AS b
ON a.common_column = b.common_column;

LEFT JOIN filtering NULLs
SELECT a.column1, b.column2
FROM tableA AS a
LEFT JOIN tableB AS b
ON a.common_column = b.common_column
WHERE b.column2 IS NULL;
```

---

5. JOIN Mini Test (With Correct Answers)

Q1: Show employees who have departments.
Answer: INNER JOIN

Q2: Show all employees even if they don't have departments.
Answer: LEFT JOIN

Q3: Show all customers and all orders, even if unmatched.
Answer: FULL OUTER JOIN

Conclusion:

- WHERE Clause: Mastered
- INNER JOIN: Mastered
- LEFT JOIN: Mastered
- JOIN Decision Making: Mastered

Sriram Murali is fully ready for SQL Interview Questions for Data Analyst roles!

Subquery Type	Where It's Used	Returns	Use Case Example
Scalar Subquery	`WHERE`, `SELECT`	A **single value**	Filter or compare with a
List Subquery	`WHERE`	**Multiple values**	Match against a list (e.
Correlated Subquery	`WHERE` or `SELECT`	Depends on outer query	Compare each row to
Table Subquery	`FROM`	A **virtual table**	Build a sub-result set t
Subquery in SELECT	`SELECT`	One value per row	Add a related calculatio

SQL Practice Workbook - Sriram Murali

---

1. WHERE Clause Practice

Practice 1: Simple WHERE

```
SELECT *
FROM employees
WHERE department = 'HR'
AND salary BETWEEN 40000 AND 80000;
```

Practice 2: WHERE with NOT IN

```
SELECT *
FROM employees
WHERE salary < 60000
AND department <> 'Finance';
```

Practice 3: WHERE with IN

```
SELECT *
FROM employees
WHERE salary IN (30000, 50000, 70000);
```

Practice 4: WHERE with NULL

```
SELECT *
FROM employees
WHERE department IS NULL;
```

Practice 5: WHERE combining AND and OR

```
SELECT *
FROM employees
WHERE department = 'HR'
OR salary > 90000;
```

Practice 6: WHERE with Parentheses

```
SELECT *
FROM employees
WHERE (department = 'Finance' OR department = 'HR')
AND salary > 60000;
```

Practice 7: LIKE Pattern Matching

```
SELECT *
FROM employees
WHERE emp_name LIKE 'S%';
```

---

2. INNER JOIN Practice

Practice 1: Students and Courses

```
SELECT s.student_name, c.course_name
FROM students AS s
INNER JOIN courses AS c
ON s.course_id = c.course_id;
```

Practice 2: Orders and Customers

```
SELECT o.order_id, c.customer_name
FROM orders AS o
INNER JOIN customers AS c
ON o.customer_id = c.customer_id;
```

Practice 3: Employees and Departments

```
SELECT e.emp_name, d.department_name
FROM employees AS e
INNER JOIN departments AS d
ON e.department_id = d.department_id;
```

---

3. LEFT JOIN Practice

Practice 1: Employees and Departments (keeping all employees)

```
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id;
```

Practice 2: Students and Courses (keeping all students)

```
SELECT s.student_name, c.course_name
FROM students AS s
LEFT JOIN courses AS c
ON s.course_id = c.course_id;
```

Practice 3: Employees without Department

```
SELECT e.emp_name, d.department_name
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id
WHERE d.department_name IS NULL;
```

---

4. JOIN Cheat Sheet

INNER JOIN: Only matched rows from both tables
LEFT JOIN: All left rows + matched right rows (NULL if missing)
RIGHT JOIN: All right rows + matched left rows (rare)
FULL OUTER JOIN: Everything from both sides (matched and unmatched)

Syntax Templates:

INNER JOIN

```
SELECT a.column1, b.column2
FROM tableA AS a
INNER JOIN tableB AS b
ON a.common_column = b.common_column;
```

LEFT JOIN

```
SELECT a.column1, b.column2
FROM tableA AS a
LEFT JOIN tableB AS b
ON a.common_column = b.common_column;

LEFT JOIN filtering NULLs
SELECT a.column1, b.column2
FROM tableA AS a
LEFT JOIN tableB AS b
ON a.common_column = b.common_column
WHERE b.column2 IS NULL;
```

---

5. JOIN Mini Test (With Correct Answers)

Q1: Show employees who have departments.
Answer: INNER JOIN

Q2: Show all employees even if they don't have departments.
Answer: LEFT JOIN

Q3: Show all customers and all orders, even if unmatched.
Answer: FULL OUTER JOIN

---

6. Ultimate Challenge: LEFT JOIN + Filter + Sort

Scenario:
List employee names, department names, and salaries
- Only for employees in 'Finance' or 'Operations', OR those with no department assigned
- Only if salary is greater than 60000
- Sort by salary in descending order

```
SELECT e.emp_name, d.department_name, e.salary
FROM employees AS e
LEFT JOIN departments AS d
ON e.department_id = d.department_id
WHERE (d.department_name IN ('Finance', 'Operations') OR d.department_name IS NULL)
AND e.salary > 60000
ORDER BY e.salary DESC;
```

Conclusion:

- WHERE Clause: Mastered
- INNER JOIN: Mastered
- LEFT JOIN: Mastered
- JOIN Decision Making: Mastered
- Real-World Challenges: Completed

Sriram Murali is fully ready for SQL Interview Questions for Data Analyst roles!