Session 10: Data querying: SQL and R

Jamuna S Murthy
Assistant Professor
Department of CSE

Academic Year - 2025-26

What is SQL?

SQL (Structured Query Language) is a standard programming language specifically designed for managing and manipulating relational databases. It enables users to create, read, update, and delete data stored in relational database systems.

Key Features of SQL

- Designed for managing structured data
- Works with relational database management systems (RDBMS)
- Supports data manipulation, definition, control, and access
- Uses English-like syntax for writing queries

Common SQL Operations

- **SELECT** – retrieve data from tables
- **INSERT** – add new data to tables
- **UPDATE** – modify existing data
- **DELETE** – remove data from tables
- **CREATE** and **DROP** – manage database objects like tables

Introduction to SQL in R

R provides powerful integration with SQL queries using packages such as `sqldf` and `DBI`. This allows analysts to manipulate data frames in R using SQL-like syntax. This session explores key SQL clauses and their usage inside R.

Prerequisites

- Install and load `sqldf` package
- Basic understanding of SQL and data frames in R

Example Dataset

We will use the following data frame in our examples:

```
data <- data.frame(
  id = 1:5,
  name = c("Alice", "Bob", "Charlie", "David", "Eve"),
  age = c(25, 30, 35, 40, 22),
  salary = c(50000, 60000, 70000, 80000, 45000)
)
```

1. SELECT and FROM

Usage: Extract columns from a data frame.

Examples:

```
library(sqldf)
sqldf("SELECT * FROM data")
sqldf("SELECT name, age FROM data")
sqldf("SELECT id, salary FROM data")
```

2. WHERE Clause

Usage: Filter rows based on conditions.

Examples:

```
sqldf("SELECT * FROM data WHERE age > 30")
sqldf("SELECT * FROM data WHERE salary <= 60000")
sqldf("SELECT * FROM data WHERE name = 'Alice'")
```

3. IS and LIKE Operators

Usage: Filter using NULL checks and pattern matching.

Examples:

```
data$name[3] <- NA
sqldf("SELECT * FROM data WHERE name IS NULL")
sqldf("SELECT * FROM data WHERE name IS NOT NULL")
sqldf("SELECT * FROM data WHERE name LIKE 'A%'")
```

4. ORDER BY

Usage: Sort results based on columns.

Examples:

```
sqldf("SELECT * FROM data ORDER BY age")
sqldf("SELECT * FROM data ORDER BY salary DESC")
sqldf("SELECT * FROM data ORDER BY name ASC")
```

5. LIMIT Clause

Usage: Limit number of returned rows.

Examples:

```
sqldf("SELECT * FROM data LIMIT 2")
sqldf("SELECT * FROM data ORDER BY age DESC LIMIT 1")
sqldf("SELECT name FROM data LIMIT 3")
```

6. MAX and MIN Functions

Usage: Find maximum and minimum values.

Examples:

```
sqldf("SELECT MAX(salary) FROM data")
sqldf("SELECT MIN(age) FROM data")
sqldf("SELECT name, salary FROM data WHERE salary = (
   SELECT MAX(salary) FROM data)")
```

Practise Dataset

Dataset: Student Scores

```
students <- data.frame(
  roll_no = 101:105,
  student_name = c("Raj", "Simran", "Amit", "Priya", "
     Neha"),
  subject = c("Math", "Science", "Math", "English", "
     Science"),
  marks = c(78, 85, 67, 90, 88)
)
```

Conclusion

With the `sqldf` package, R users can seamlessly run SQL queries on data frames. This integration combines the strengths of SQL's querying abilities with R's statistical power.