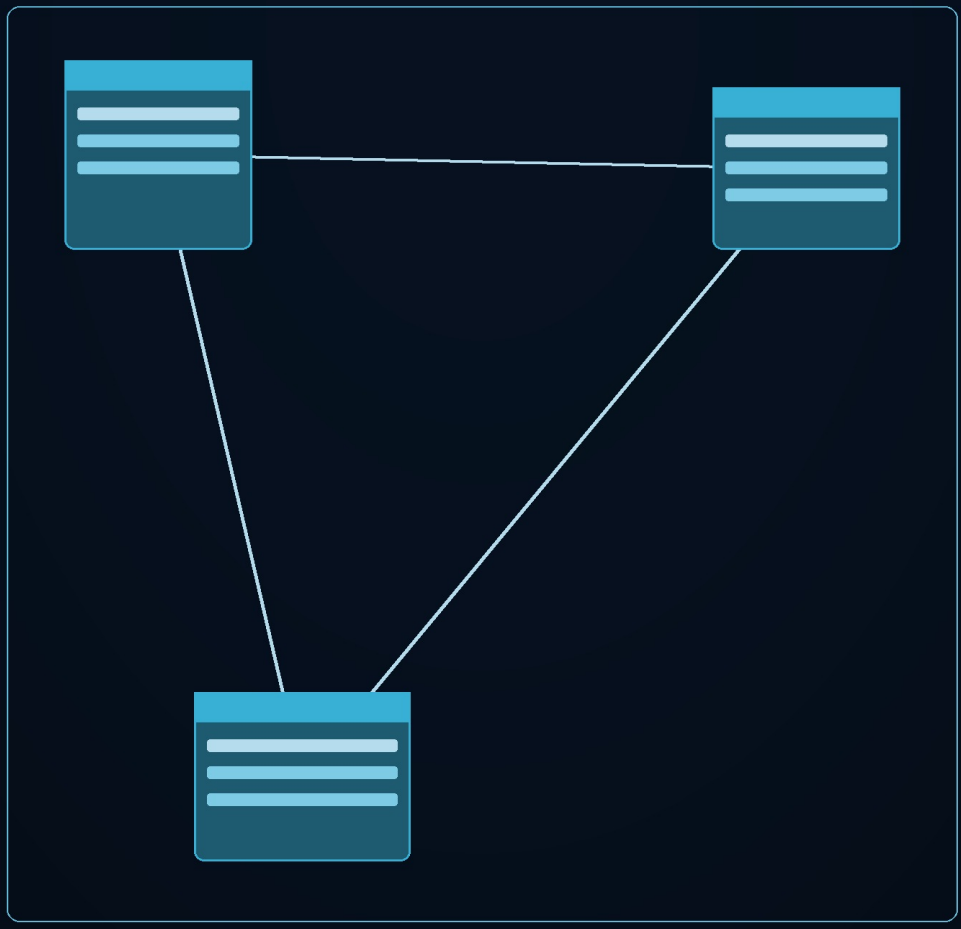
DATA ANALYTICS SERIES

BOOK 04

SQL FOR DATA ANALYSIS

A Practical Guide to Queries, Joins, CTEs, Window Functions and Business Reporting



QUERIES · JOINS · WINDOW FUNCTIONS

ANOOP GUPTA

Contents

Copyright	4
Disclaimer	5
How to Use This Book	6
Chapter 1: Introduction to SQL and Databases	7
Chapter 2: SQL Basics – SELECT, FROM and DISTINCT	38
Chapter 3: Filtering Data with WHERE	69
Chapter 4: Sorting, Limiting and Handling Data	103
Chapter 5: Aggregate Functions and GROUP BY	140
Chapter 6: HAVING and Advanced Aggregation	175
Chapter 7: SQL Joins	203
Chapter 8: Subqueries and Set Operations	241
Chapter 9: Common Table Expressions (CTEs)	278
Chapter 10: SQL Date and Time Functions	313
Chapter 11: Window Functions for Data Analysis	349
Chapter 12: Advanced Analytical SQL	386
Chapter 13: SQL Query Performance and Optimization Basics	422
Chapter 14: Building a Complete Sales Analysis Project	450
Chapter 15: SQL Interview Preparation and Practical Assessments	486
Chapter 16: SQL Job Readiness and Real-World Problem Solving	522
Quick Reference Cheat Sheet	553
Practice Datasets	556
What to Learn Next	559

SQL for Data Analysis

A Practical Guide to Queries, Joins, CTEs, Window Functions and Business Reporting

Data Analytics Series, Book 4

Anoop Gupta

Copyright

Copyright © 2026 Anoop Gupta. All rights reserved.

No part of this publication may be reproduced, distributed, stored in a retrieval system or transmitted in any form or by any means without prior written permission from the copyright holder, except for brief quotations used in reviews, educational references or other uses permitted by applicable copyright law.

This book is an independent educational publication. It is not affiliated with, endorsed by or sponsored by Microsoft, Tableau, the Python Software Foundation, or any other software company, educational institution, professional body or organisation.

Product names and trademarks mentioned in this book belong to their respective owners and are used only for educational identification.

Disclaimer

This book is intended for educational and informational purposes only. It does not constitute professional career, employment, investment, financial, statistical, legal or business advice. It does not guarantee a job, a salary or a particular career outcome.

The examples, datasets, company names and figures used in this book are illustrative sample data unless explicitly stated otherwise. They are not real company results or labour-market statistics.

Software features, function names, menus, licences and interfaces change. Always check the official documentation for the tool you are using, and verify important calculations against original records and appropriate professional guidance.

How to Use This Book

Work through the chapters in order. Each chapter explains one idea, shows a worked example with sample data, then gives practice exercises, a short project and likely interview questions.

Type the formulas, queries or code yourself rather than only reading them. When a screen or menu name does not match your version of the software, look up the current official documentation instead of guessing.

Keep a simple practice folder for your workbooks, query files and notes. The Quick Reference Cheat Sheet and Practice Datasets at the end of the book, where provided, are there for revision after you finish the chapters.

Chapter 1: Introduction to SQL and Databases

In this chapter

- Understand what SQL is and why data analysts use it.
- Learn what databases, tables, rows and columns mean.
- Understand relational databases and database management systems.
- Learn the difference between SQL and Excel.
- Explore how SQL queries retrieve business information.
- Set up a SQL practice environment.
- Understand the sample sales database used throughout this book.

1.1 What Is SQL?

SQL stands for **Structured Query Language**.

It is a language used to communicate with relational databases. We use SQL to retrieve, insert, update, delete and analyse data stored in database tables.

Imagine a company that sells laptops, keyboards, mice and other products.

Every day, the company generates information about:

- Customers
- Products
- Sales orders
- Payments
- Employees
- Inventory
- Transactions

When this information becomes large, storing everything in separate Excel worksheets can become difficult.

A database provides an organized way to store and manage this information.

SQL allows us to ask questions about that data.

For example:

- What were the total sales?
- Which product generated the highest revenue?
- Which customers purchased products?
- How many orders were placed last month?
- Which region generated the most sales?
- Which customers have never placed an order?

SQL helps retrieve the information required to answer these questions.

A simple SQL example

Suppose a company has a table called `products`.

product_id	product_name	price
101	Laptop	50000
102	Keyboard	2000
103	Mouse	800
104	Monitor	12000

To retrieve all records from this table, we write:

```
SELECT *
FROM products;
```

Expected output:

product_id	product_name	price
101	Laptop	50000
102	Keyboard	2000
103	Mouse	800
104	Monitor	12000

Let's understand this query.

- `SELECT` tells the database what information we want.

- `*` means all columns.
- `FROM` tells the database which table contains the information.
- `products` is the table name.
- `;` marks the end of the SQL statement in common SQL environments.

This is one of the simplest SQL queries, but it introduces the basic structure used throughout this book.

1.2 Why SQL Matters in a Real Data Analyst Job

A data analyst often works with information stored in databases rather than receiving every dataset as an Excel file.

Consider a retail business.

It may have separate tables for customers, products, orders and payments.

The management team wants to understand sales performance.

A data analyst might need to:

1. Retrieve all sales transactions.
2. Identify the products sold.
3. Calculate total revenue.
4. Compare monthly sales.
5. Identify high-value customers.
6. Compare regional performance.
7. Find customers who have not purchased recently.

SQL makes these tasks possible by allowing analysts to retrieve and combine information from different tables.

Examples of SQL in business departments

Department	SQL use
Sales	Revenue and order analysis
Finance	Payment and transaction analysis
Marketing	Customer segmentation

Department	SQL use
Inventory	Stock and product analysis
Operations	Order processing analysis
Customer service	Customer history analysis
Management	Business performance reporting

SQL and Excel: Understanding the difference

Excel and SQL are both useful analytical tools, but they solve different problems.

Feature	Excel	SQL
Main purpose	Spreadsheet analysis	Database querying
Data organization	Worksheets and tables	Database tables
Data retrieval	Filters and formulas	SQL queries
Data combination	Lookups and Power Query	JOIN operations
Calculations	Functions and formulas	SQL expressions and aggregate functions
Large datasets	Depends on workbook limits and performance	Designed to query database-managed data
Automation	Formulas, macros and Power Query	Saved queries, scripts and database workflows
Visualization	Charts and dashboards	Usually combined with BI tools

A data analyst may use SQL to retrieve information and Excel or Power BI to present the results.

For example:

Database → SQL query → Excel report → Management decision

Understanding SQL adds another important skill to your data analytics toolkit.

1.3 What Is a Database?

A database is an organized collection of information stored in a structured way so that it can be accessed, managed and updated.

Think about a company's customer register.

It may contain:

- Customer ID
- Customer name
- Phone number
- Email address
- City

Instead of maintaining thousands of separate documents, the business can store this information in a database table.

Sample customer table

The following is fictional sample data.

customer_id	customer_name	city
1	Rahul	Delhi
2	Priya	Noida
3	Amit	Gurugram
4	Neha	Delhi

The table stores customer information in a structured format.

A database can contain many such tables.

For example:

- `customers`
- `products`
- `orders`
- `order_items`
- `payments`

Each table stores a particular type of information.

Why separate tables?

Imagine storing every customer's details, every product description and every transaction in one enormous table.

The same customer information might be repeated hundreds of times.

This can create problems:

- Duplicate information.
- Inconsistent customer details.
- Difficult updates.
- Increased storage requirements.
- More complicated reporting.

Relational databases organize related information into separate tables and connect them through common identifiers.

This makes data management more structured.

1.4 Understanding Tables, Rows and Columns

Three terms are essential when learning SQL:

1. Table
2. Row
3. Column

A. Table

A table stores related information.

For example, `customers` stores customer records.

B. Row

A row represents one record in a table.

Example:

customer_id	customer_name	city
1	Rahul	Delhi

This is one customer record.

A row is also commonly called a record or tuple.

C. Column

A column represents a particular attribute of the records.

In the customers table:

- `customer_id` stores customer identifiers.
- `customer_name` stores customer names.
- `city` stores city names.

A column is also called a field or attribute.

Visual understanding

Sample customer table

customer_id	customer_name	city
1	Rahul	Delhi
2	Priya	Noida
3	Amit	Gurugram

> ```
> **Column**
> Vertical field of information
>
> **Row**
> One complete record
> ```

Understanding these terms will make SQL queries easier to read.

1.5 What Is a Relational Database?

A relational database stores information in multiple tables that can be connected through relationships.

For example, a sales business might have these tables:

Customers

customer_id	customer_name
1	Rahul
2	Priya
3	Amit

Orders

order_id	customer_id	order_date
1001	1	2026-01-05
1002	2	2026-01-07
1003	1	2026-01-10

Notice that `customer_id` appears in both tables.

In the Customers table, it identifies a customer.

In the Orders table, it tells us which customer placed an order.

This is a relationship.

Primary key

A primary key is a column, or combination of columns, used to uniquely identify each record in a table.

For example:

`customer_id` is the primary key of the Customers table.

Each customer must have a unique customer ID.

Important primary key properties:

- Values must be unique.
- A primary key cannot contain NULL values.
- A table has one primary key constraint, although it may contain multiple columns.

Foreign key

A foreign key is a column, or group of columns, that references a key in another table.

For example:

`customer_id` in the Orders table references `customer_id` in the Customers table.

This helps maintain relationships between records.

Understanding the relationship

CustomersPrimary key: customer_id customer_id · customer_name

One customer can place multiple orders

```
**OrdersForeign key: customer_id**
order_id · customer_id · order_date
```

This is a one-to-many relationship:

- One customer can have multiple orders.
- Each order belongs to a particular customer in this sample design.

You will study how to retrieve information from related tables using JOIN operations in Chapter 7.

1.6 What Is a DBMS?

DBMS stands for **Database Management System**.

It is software used to create, store, manage and access databases.

A DBMS provides facilities to:

- Create databases.
- Create tables.
- Insert records.
- Update records.
- Delete records.
- Retrieve information.
- Manage access permissions.
- Maintain data integrity.
- Support backup and recovery.

SQL is the language used to communicate with many relational DBMS platforms.

Anoop Gupta Books

End of free sample – buy the full book at onlineebook.in/books/sql-for-data-analysis.html

This file is a free sample. The complete ebook is sold at onlineebook.in.