KNOWLEDGE & DEVELOPMENT SOLUTIONS

Basic SQL

SQL Day 1 Class

Learning Objectives

By the end of this lesson, you will be able to:

1. Define SQL, statement, and query
2. Use SELECT to query fields from a table

What is SQL

- SQL (Structured Query Language) is the cornerstone of modern database management.
  - It serves as the universal programming language for data management in databases.
  - SQL seamlessly integrates with widely-used database systems such as Oracle, MySQL, SQL Server, and others.
  - SQL's declarative nature allows you to specify what data you want without getting bogged down in how to retrieve it.
  - Its user-friendly, English-like syntax makes SQL accessible and easy to learn.

Why Learn SQL?

- SQL is indispensable for numerous technology roles, encompassing:
  - Database Administrators
  - Data Analysts
  - Data Engineers
  - Application Developers
- Proficiency in SQL empowers professionals to effectively interact with data-driven systems.
- SQL proficiency unlocks career opportunities across industries reliant on databases.
- Adding SQL to your technical skill set enhances your overall capabilities and value.
-

What is a Query?

- A query is a specific request for data from a database.
  - It is a type of SQL statement that retrieves information from one or more tables.
  - Queries are used to interact with databases and retrieve meaningful insights from stored data.
- SQL queries can be simple, retrieving individual records, or complex, involving multiple tables and advanced filtering criteria.
  - The results of a query are presented in a structured format that makes it easy to analyze and work with the retrieved data.

What is a Query?

- SQL queries can be categorized into various types based on their purpose and functionality.

- Common types of SQL queries include:
  - SELECT Queries: Retrieve data from one or more tables.
  - INSERT Queries: Add new data records into a table.
  - UPDATE Queries: Modify existing data records in a table.
  - JOIN Queries: Combine data from multiple tables.
  - AGGREGATE Queries: Perform calculations on data (e.g., SUM, AVG, COUNT).

SQL Statements

SQL is written in **statements** - strings of characters that conform to formatting and syntax rules specified in the international standard

- For example, a basic SELECT statement fetching data from a Customers table would be:

```
SELECT id, name, address
    FROM Customers
  WHERE state = 'CA'
    ORDER BY name;
```

SQL Statements

- Writing an SQL statement involves several key steps, which help you communicate effectively with the database engine.
  - The first step is to specify the operation type. Common operation types include SELECT, INSERT, UPDATE, and DELETE.
  - Let's take a closer look at an example using the SELECT statement to query data from a Customers table.

```
SELECT id, name, address
    FROM Customers
  WHERE state = 'CA'
    ORDER BY name;
```

SQL Statements

Writing an SQL statement involves:

- Specify the table and columns:
  - The statement needs to indicate from which database table and columns you want to access or operate on. Often this is done after the operation type using a FROM clause.

```
SELECT id, name, address
    FROM Customers
  WHERE state = 'CA'
    ORDER BY name;
```

SQL Statements

Writing an SQL statement involves:

- Specify any filtering conditions (optional):
  - You can filter which data rows you want using a WHERE clause and conditional logic.

```
SELECT id, name, address
    FROM Customers
WHERE state = 'CA'
ORDER BY name;
```

SQL Statements

Writing an SQL statement involves:

- Specify sorting/grouping (optional):
  - Statements like ORDER BY and GROUP BY can control row sorting and grouping in the output.

```
SELECT id, name, address
    FROM Customers
  WHERE state = 'CA'
  ORDER BY name;
```

SQL Statements

Writing an SQL statement involves:

- End with a semicolon:
  - All SQL statements are terminated with a semicolon ";"

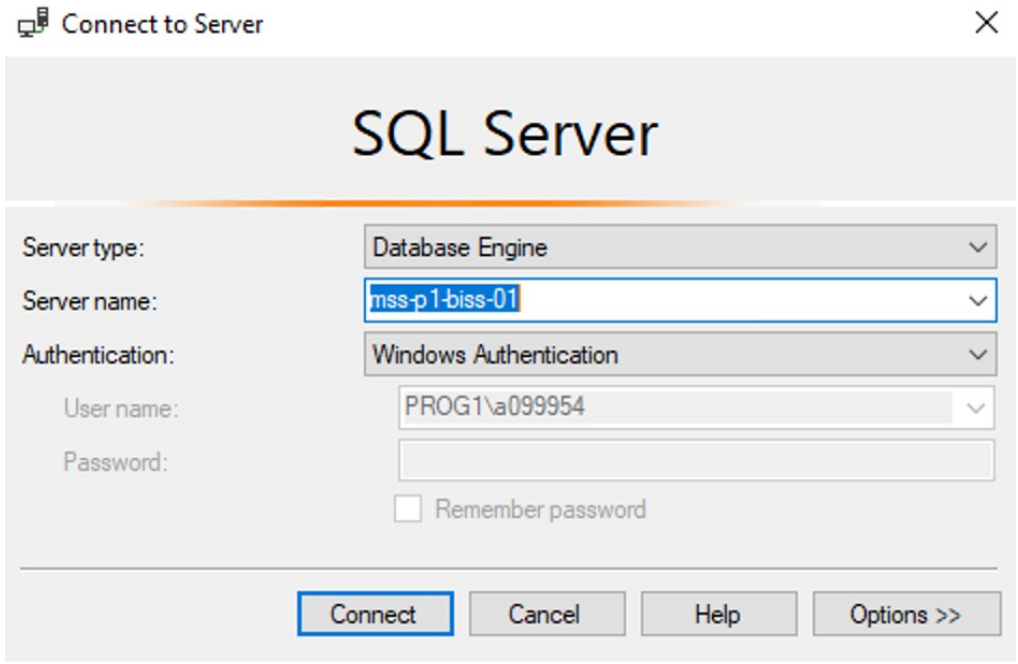
```
SELECT id, name, address
    FROM Customers
  WHERE state = 'CA'
    ORDER BY name;
```

Summary

- SQL Data Querying - Best Practices
  - SQL enables powerful data retrieval & manipulation
  - Flexible data access across tables
  - Optimization for fast performance
- Tips for Efficiency:
  - Focus queries only on required columns and conditions
  - Look for inefficient scans or expensive operators
- Improve Over Time:
  - Start simple and targeted, add complexity gradually
  - Validate results at each iteration
  - Practice makes efficient SQL use

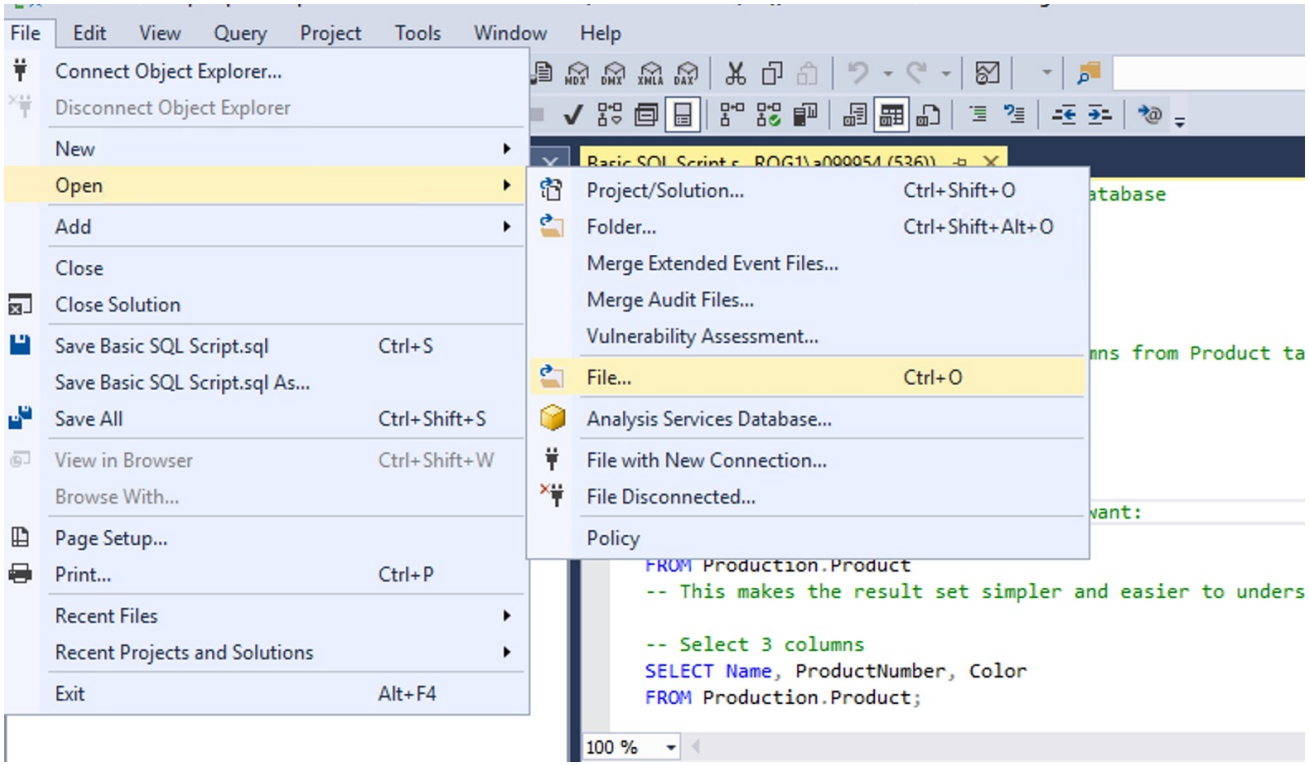
Hands On Examples

- Using the AdventureWorks2012 Database
  - Please open up SSMS
  - Connect to BISS
    - Server = mss-p1-biss-01



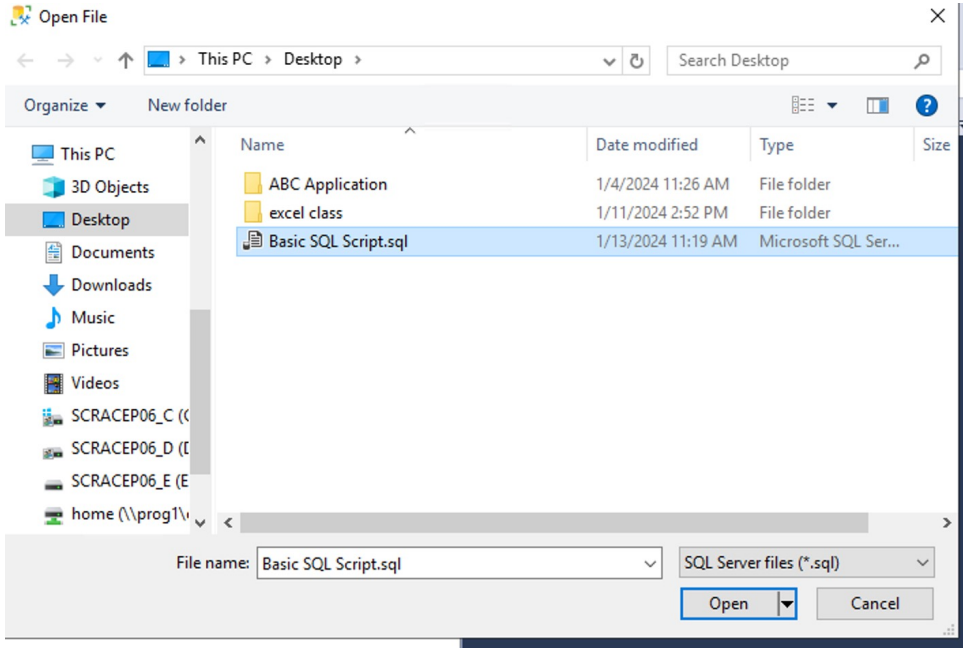
Hands On Examples

- Click File -> Open -> File



Hands On Examples

- Open the SQL Day 1 Basic SQL Script.sql file



Exercise

- Open the Day 1 Exercise.sql file
  - Answer the questions from each section