**GitHub Copilot for MSSQL extension**

The GitHub Copilot integration with the MSSQL extension for Visual Studio Code introduces AI-assisted development directly within the SQL development workflow.

This new feature empowers developers to write and optimize database code, generate and evolve schemas, understand existing logic, and work more confidently with code-first and data-first database development patterns using GitHub Copilot's intelligent, context-aware suggestions.

This preview is designed for developers, with a focus on the following key personas:

* **Modern Application Developers**: Building feature-rich, scalable applications with frameworks like React, Angular, .NET, ango, and Node.js.
* **AI / Cloud-Native Developers**: Specializing in containerized, serverless, and microservices-based applications, often integrating AI-powered capabilities.
* **Solutions Architects**: Designing end-to-end systems that align database solutions with broader architectural goals.
* **Database Developers**: Focusing on T-SQL, database-specific concepts, and optimizing database workflows.

**What is the MSSQL extension for Visual Studio Code?**

The [MSSQL extension in Visual Studio Code](https://aka.ms/vscode-mssql-marketplace) is designed to help developers seamlessly work with their databases, simplifying the use of SQL database in Fabric, all Azure SQL offerings, and SQL Server as the backend for their applications.

To learn more about the extension, visit the [GitHub repository](https://github.com/microsoft/vscode-mssql).

**What is GitHub Copilot for the MSSQL extension for Visual Studio Code?**

GitHub Copilot for the MSSQL extension integrates the power of AI-driven assistance directly into your SQL development workflow within Visual Studio Code. It enables developers to:

* Explore, design, and evolve database schemas using intelligent, code-first / data-first guidance.
* Apply contextual suggestions for SQL syntax, relationships, and constraints.
* Write, optimize, and troubleshoot SQL queries with AI-recommended improvements.
* Generate mock data and seed scripts to support testing and development environments.
* Accelerate schema evolution by autogenerating ORM (Object-Relational Mapping) migrations or T-SQL change scripts.
* Understand and document business logic embedded in stored procedures, views, and functions.
* Get security-related recommendations, such as avoiding SQL injection or excessive permissions.
* Receive natural language explanations to help developers unfamiliar with T-SQL write and understand code confidently.
* Help reverse-engineering existing databases by explaining SQL schemas and relationships.
* Help scaffold backend components (for example, data access layers) based on your current database context.

**Supported SQL Server platforms**

GitHub Copilot for the MSSQL extension works with all of these SQL Server platforms:

**SQL Server**

* SQL Server 2019
* SQL Server 2022
* SQL Server 2025 (Public Preview)
* SQL Server running on any platform:
  + Windows
  + Linux
  + Containers (local and Kubernetes deployments)

**Azure SQL**

* Azure SQL Database
* Azure SQL Managed Instance
* SQL Server on Azure Virtual Machines

**Microsoft Fabric**

* SQL database in Fabric
* Fabric Data Warehouse
* Fabric Lakehouse (SQL analytics endpoint)

**Target Audience**

GitHub Copilot for the MSSQL extension is designed for developers who work with applications and SQL databases in Visual Studio Code. It provides intelligent assistance to help you write, optimize, and understand database code more effectively.

| **Persona** | **Description** |
| --- | --- |
| **Modern Application Developers** | Building robust, scalable applications using frameworks like React, Angular, .NET, Django, and Node.js, often following a code-first approach where application logic drives database design. GitHub Copilot streamlines database interactions by assisting with schema generation, query authoring, and integration patterns directly from your codebase, boosting productivity and minimizing context switching. |
| **AI / Cloud-Native Developers** | Building serverless, containerized, and microservices-based solutions that integrate AI capabilities, both within the application and the database layer. GitHub Copilot accelerates development by generating intelligent T-SQL queries, managing schema changes, and assisting with data access patterns common in cloud-native architectures. It also helps developers use AI features built into the database, such as vector search and hybrid retrieval scenarios, to build intelligent, AI-ready applications. |
| **Solutions Architects** | Designing data-centric systems and ensuring consistency across services and environments. GitHub Copilot helps visualize, validate, and prototype database interactions, making it easier to align database design with system architecture goals. |
| **Database Engineers** | Managing schema evolution, writing complex T-SQL queries, and optimizing performance. GitHub Copilot speeds up development by offering suggestions, explaining code, and identifying potential optimizations, boosting both speed and confidence. |

**Features (Preview)**

| **Feature** | **Description** |
| --- | --- |
| **Chat / Inline Copilot Suggestions** | Engage in natural language conversations with the @mssql chat participant or use inline completions for T-SQL or ORM code. Suggestions adapt based on your database schema and active files. |
| **Schema Explorer & Designer** | Understand, design, and evolve your database schema using AI assistance. Supports object creation, relationships, and reverse engineering. |
| **Smart Query Builder** | Generate SQL and ORM (Object-Relational Mapping) queries using filters, joins, groupings, and conditions, based on schema awareness and natural language prompts. |
| **Code Generation** | Quickly scaffold database code, stored procedures, or ORM-based data access layers based on your current database schema and active files. GitHub Copilot understands your context and can generate repeatable patterns such as CRUD operations or access methods aligned with your development stack. |
| **Query Optimizer Assistant** | Get AI-generated tips for improving the performance of your SQL queries. GitHub Copilot might suggest better indexing strategies, refactor joins, or spot inefficiencies in WHERE clauses, helpful for developers who aren't experts in performance tuning. Additionally, it supports analysis of execution plans for in-depth analysis, enabling more precise recommendations tailored to your query's actual execution context. |
| **Business Logic Explainer** | Ask GitHub Copilot to explain in simple terms what a stored procedure, view, or user-defined function does. This is especially useful for onboarding new developers who need to understand how business rules are implemented without reading hundreds of lines of T-SQL code. |
| **Security Analyzer** | GitHub Copilot can analyze patterns that might expose your code to SQL injection, overly permissive roles, or unencrypted sensitive data. It can also recommend safer ways to handle credentials, user input, and authentication flows, directly in your context. |
| **Localization & Formatting Helper** | Whether you're building multilingual apps or just ensuring proper sorting and encoding, GitHub Copilot can suggest appropriate collation settings, Unicode usage, and query patterns that support language- and region-specific requirements. |
| **Test Data Generator** | Generate close to realistic, schema-aware sample data (via SQL INSERTs or ORM seeders) to populate your development environment. GitHub Copilot can even extrapolate schema from an existing sample (JSON, CSV, TXT) or autogenerate themed data to support prototyping or testing. |

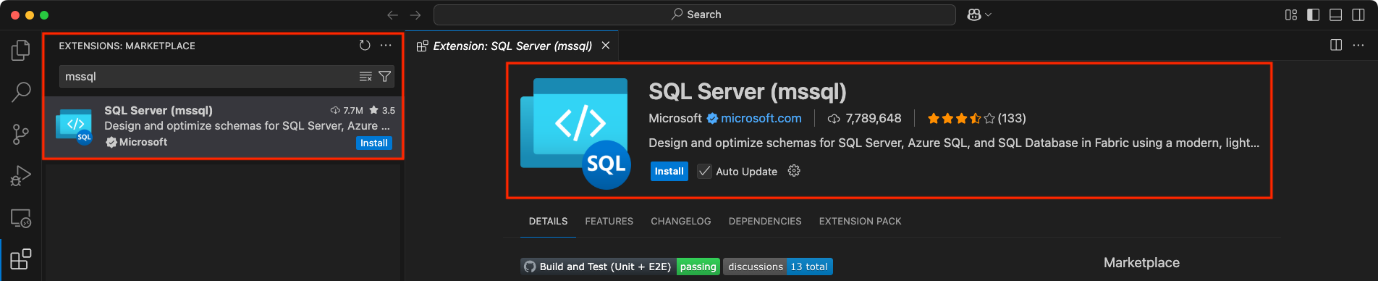
**Prerequisites**

**Install Visual Studio Code**

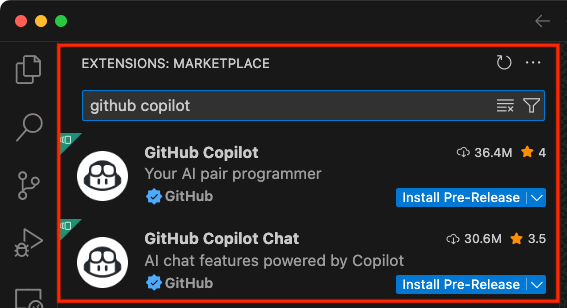
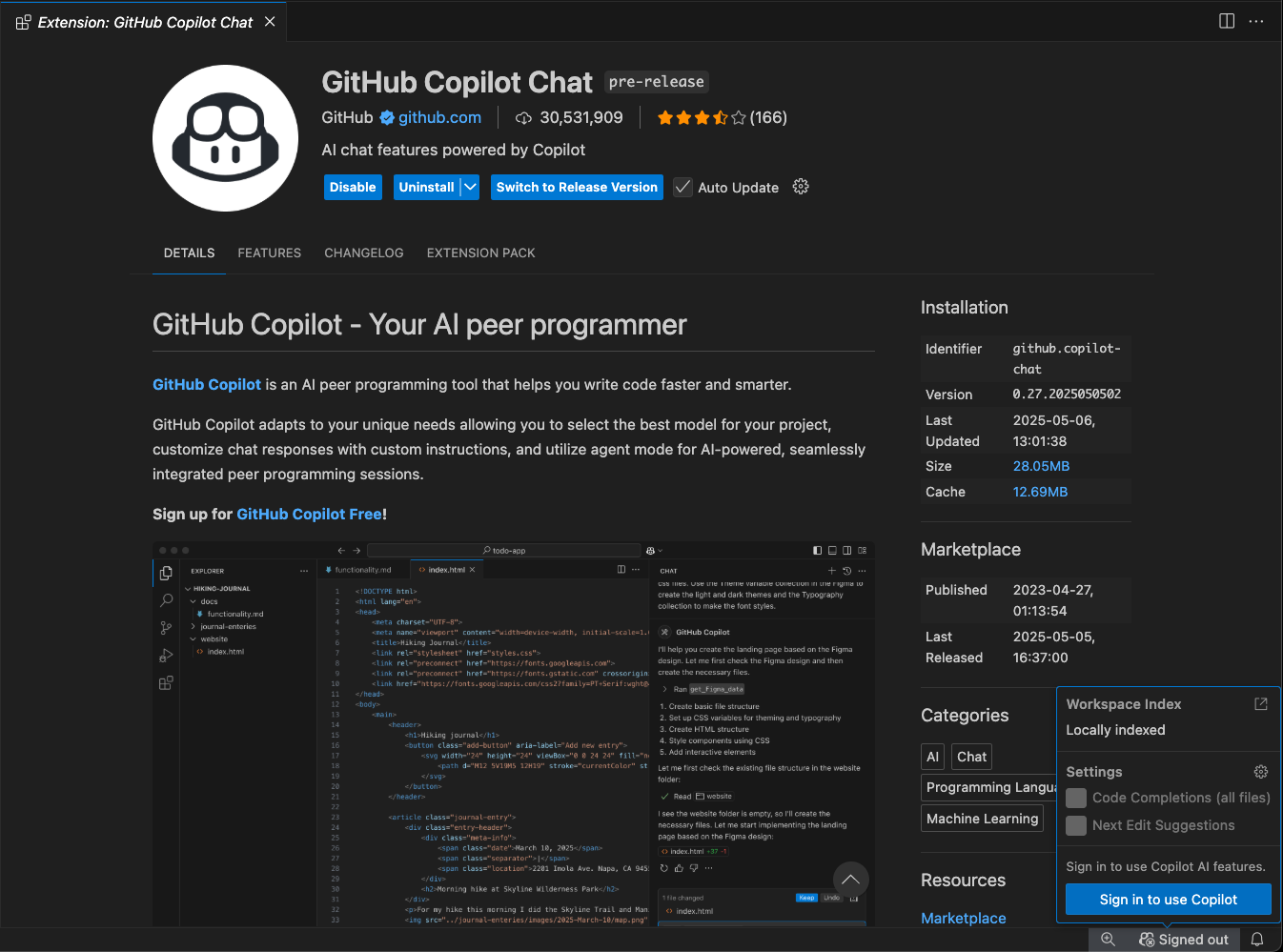
1. Download Visual Studio Code .
2. Complete the installation following the wizard.

**Install the MSSQL extension in Visual Studio Code**

To get started with SQL development in Visual Studio Code, install the **MSSQL extension**:

1. Open **Visual Studio Code**.
2. Select the **Extensions** icon in the Activity Bar (press Cmd+Shift+X on macOS or Ctrl+Shift+X on Windows/Linux).
3. In the **search bar**, type mssql.
4. Find **SQL Server (mssql)** in the results and select it.
5. Select the **Install** button. [](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/media/overview/mssql-extension-vscode.png?view=sql-server-ver17#lightbox)

**Set up GitHub Copilot in Visual Studio Code**

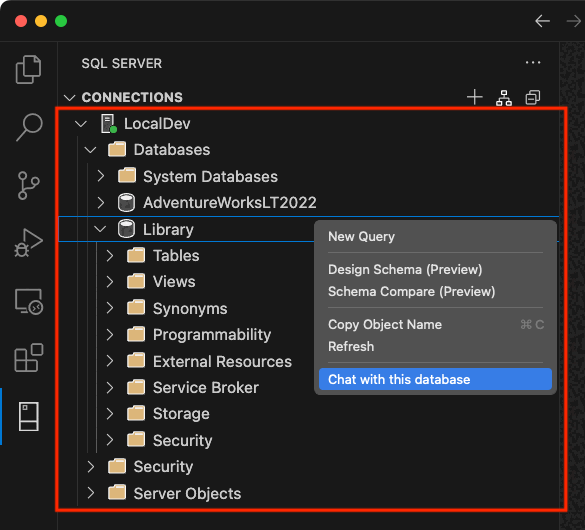
1. Make sure you have a GitHub account. If you don't have one, sign up for free at GitHub.
2. Ensure you have an active GitHub Copilot subscription. You can start a free trial or purchase a subscription at GitHub Copilot.
3. In Visual Studio Code, open the **Extensions** view (press Cmd+Shift+X on macOS or Ctrl+Shift+X on Windows/Linux).
4. Search for and install both **GitHub Copilot** and **GitHub Copilot Chat** extensions. 
5. After installation, you'll need to sign in to your GitHub account. Use Visual Studio Code command palette GitHub Copilot: Sign in or sign in using the GitHub Copilot icon from the status bar at the bottom of the window. [](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/media/overview/github-copilot-signin-vscode.png?view=sql-server-ver17#lightbox)
6. After signing in, you might need to authorize the GitHub Copilot extension to access your GitHub account. Follow the prompts to complete the authorization process.
7. Once signed in, GitHub Copilot will be active and ready to assist as you write code in your editor.

**Connect to a database**

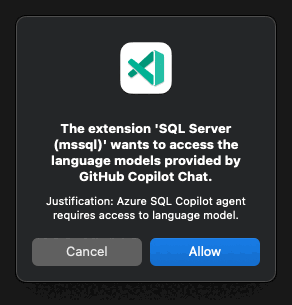
To get started with GitHub Copilot for the MSSQL extension, simply connect to a supported SQL Server or Azure SQL database from the **Connections** view in Visual Studio Code.

**Start Chatting with Your Database**

You can begin using GitHub Copilot by initiating a chat session with your database:

1. Go to the **Connections** view in the MSSQL extension.
2. Right-click on a connected database.
3. Select **Chat with this database** from the context menu. 

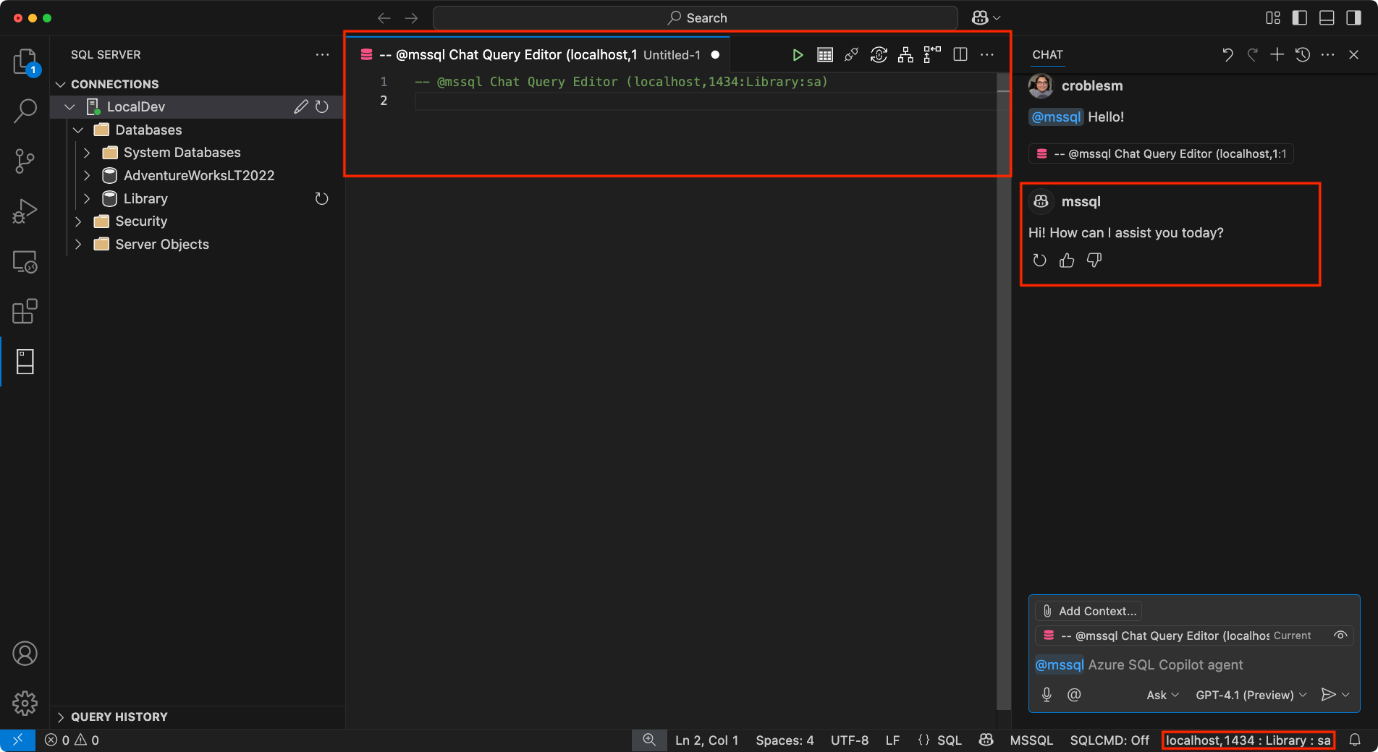
After selecting this option, you'll be prompted to allow the MSSQL extension to access the language models provided by GitHub Copilot Chat.  
This is required to enable contextual conversations about your database.



Once approved, a GitHub Copilot chat window opens in the context of the selected database. You're now ready to start asking questions, generating SQL, or exploring schema insights using natural language.

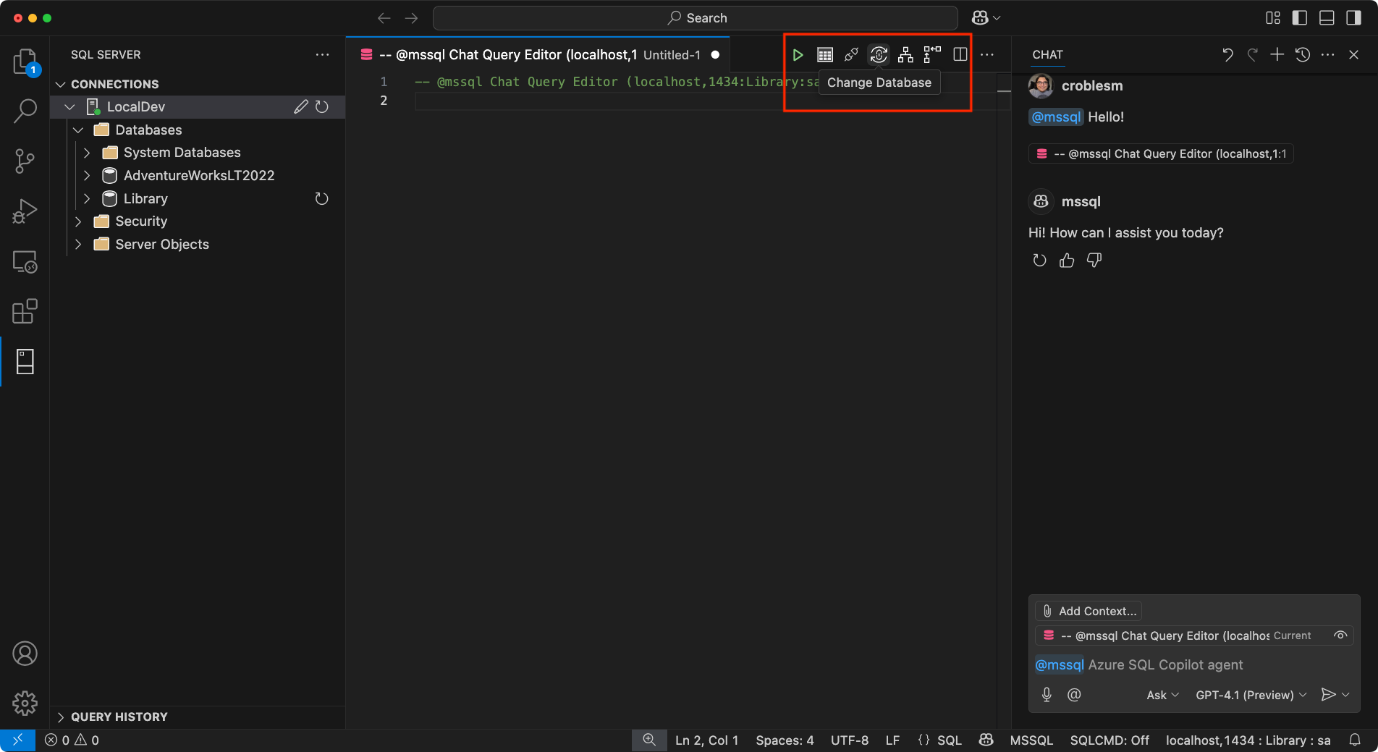
**Manage database context**

GitHub Copilot uses your current database connection to provide schema-aware suggestions. You can view your current database context by checking the status bar in Visual Studio Code.

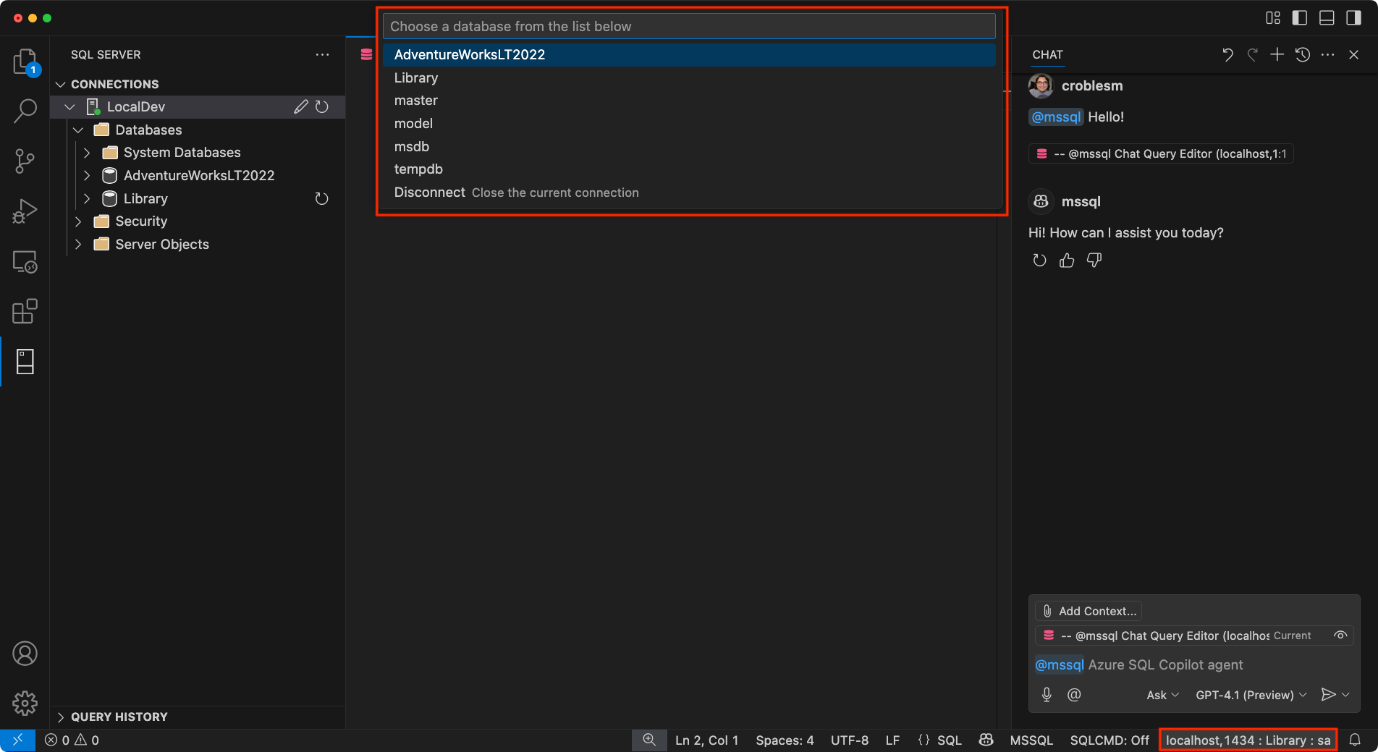
[](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/media/overview/vscode-copilot-chat-window.png?view=sql-server-ver17#lightbox)

If you want to switch to a different database:

* Use the **Change Database** button in the MSSQL extension sidebar, or

[](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/media/overview/vscode-copilot-change-database-1.png?view=sql-server-ver17#lightbox)

* Select the status bar panel displaying the current connection (server, database, user). This opens a dropdown list to select a different database from your configured profiles.

[](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/media/overview/vscode-copilot-change-database-2.png?view=sql-server-ver17#lightbox)

**Important**

GitHub Copilot requires an active database connection to provide meaningful suggestions. Without it, the chat experience won't be able to access schema or data context.

This gives you the flexibility to switch between environments while retaining context-aware assistance from GitHub Copilot.

**Share Your Experience**

When submitting feedback, consider including:

* **Scenarios tested** – Let us know which areas you focused on, for example, schema creation, query generation, security, localization.
* **What worked well** – Describe any experiences that felt smooth, helpful, or exceeded your expectations.
* **Issues or bugs** – Include any problems, inconsistencies, or confusing behaviors. Screenshots or screen recordings are especially helpful.
* **Suggestions for improvement** – Share ideas for improving usability, expanding coverage, or enhancing the GitHub Copilot's responses.

Here are common use cases and examples of what you can ask via the chat participant:

**List or explore objects in your database schema**

Ask questions about tables, columns, schemas, and object metadata in your database.

* Show all objects in the SalesLT schema of my current database, grouped by type.
* List the columns, data types, and nullability of the SalesLT.Customer table.
* How many tables, views, and procedures are defined in my current database?

**Write simple queries**

Get help writing common SQL queries for filtering, aggregation, and joins.

* Write a T-SQL query to list all customers from SalesLT.Customer who placed an order in the last 30 days based on the latest order date.
* Generate a query that calculates the average order total per customer from the SalesLT.SalesOrderHeader table, sorted descending.
* Update the previous query to include the full name of each customer from the SalesLT.Customer table.

**Explain relationships or concepts**

Ask for simplified explanations of schema relationships, query logic, or features relevant to development.

* Describe the foreign key relationship between SalesLT.SalesOrderHeader and SalesLT.Customer tables in my current database.
* I'm a developer new to T-SQL. Explain how SalesLT.SalesOrderHeader is related to SalesLT.Customer, and what keys are involved.
* Explain how vector data types work in SQL Server and when to use them for AI scenarios.

**Generate migration or integration code**

Request help generating SQL or ORM-based migration scripts.

* Create a T-SQL script to add a foreign key constraint on SalesLT.SalesOrderDetail.ProductID referencing SalesLT.Product.ProductID.
* Generate a Sequelize migration to add a foreign key from SalesLT.SalesOrderDetail.ProductID to SalesLT.Product.ProductID, assuming both exist.

**Related content**

* [Quickstart: Use Chat and inline GitHub Copilot suggestions (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/inline-copilot-suggestions?view=sql-server-ver17)
* [Quickstart: Generate code (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/code-generation?view=sql-server-ver17)
* [Quickstart: Use the Schema Explorer and designer (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/schema-explorer-designer?view=sql-server-ver17)
* [Quickstart: Use the Smart Query Builder (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/smart-query-builder?view=sql-server-ver17)
* [Quickstart: Query Optimizer Assistant (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/query-optimizer-assistant?view=sql-server-ver17)
* [Quickstart: Use the Business Logic Explainer (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/business-logic-explainer?view=sql-server-ver17)
* [Quickstart: Security Analyzer (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/security-analyzer?view=sql-server-ver17)
* [Quickstart: Localization & Formatting Helper (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/localization-formatting-helper?view=sql-server-ver17)
* [Quickstart: Generate data for testing and mocking (Preview)](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/test-and-mocking-data-generator?view=sql-server-ver17)
* [Limitations and Known Issues](https://learn.microsoft.com/en-us/sql/tools/visual-studio-code-extensions/github-copilot/limitations-and-known-issues?view=sql-server-ver17)