



# Azure Data Studio Extension Development

**Drew Skwiers-Koballa**

Program Manager

✉ Drew.SkwiersKoballa@microsoft.com

📱 @sysadmindrew

# Who?

## **Drew Skwiers-Koballa (squire's co- ball –a)**

he/him

✉ Drew.SkwiersKoballa@microsoft.com

📱 @sysadmin drew

Former small company DBA and DB dev  
Loves pugs, chickens, gardening, and SQL  
Program Manager @ Microsoft

Ask me about:

- Starting a new job remotely
- Moving across the country
- SQL Tools

---

# Agenda

- 
1. Why Develop an Extension?
  2. Extension Development Framework
  3. Building Extension Features
  4. Notebooks
  5. Package and Publish

# Azure Data Studio

Cross-platform desktop client for data-centric and data-adjacent users

Open source fork of Visual Studio Code

Core query editing experience + extensions



# Why Develop an Extension?



Improve Your  
Workflow

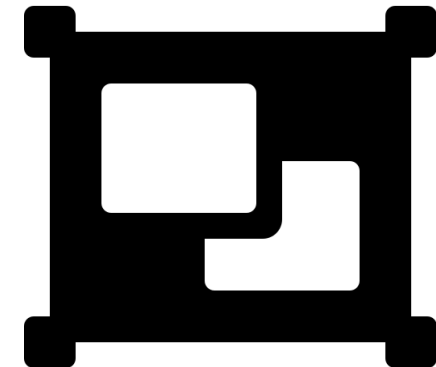
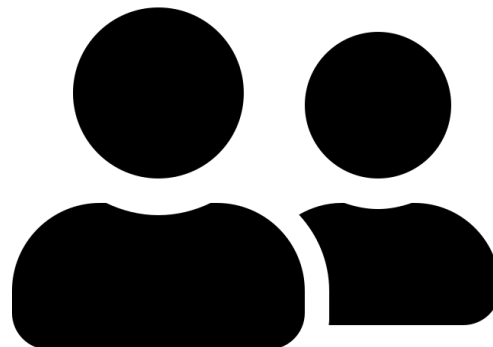
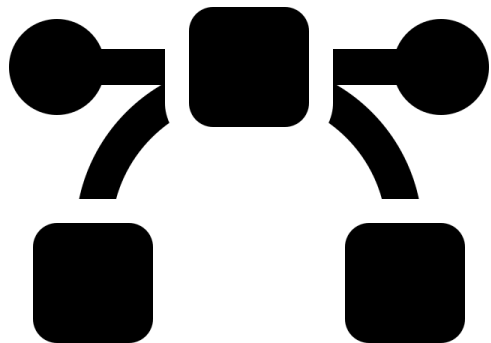
Share with the  
SQL Community

Close To Your  
Comfort Zone

Enterprise process  
Specific shortcut

Better tools  
Best practices

Development  
related to data  
platform





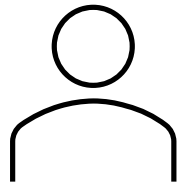
# Azure Data Studio Extension Development Framework

# Extension Development Framework

## 1. Design Elements

How does the user interact with the extension?

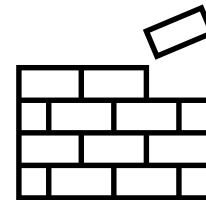
Azure Data Studio adds wizard-UIs to VS Code's available APIs



## 2. Contribution Points

How/what does the extension add to Azure Data Studio?

Places to connect your functionality to Azure Data Studio under the hood



# Design Elements

How does the user interact with the extension?

Azure Data Studio adds wizard-UIs to VS Code's available APIs

## **Azure Data Studio & VS Code**

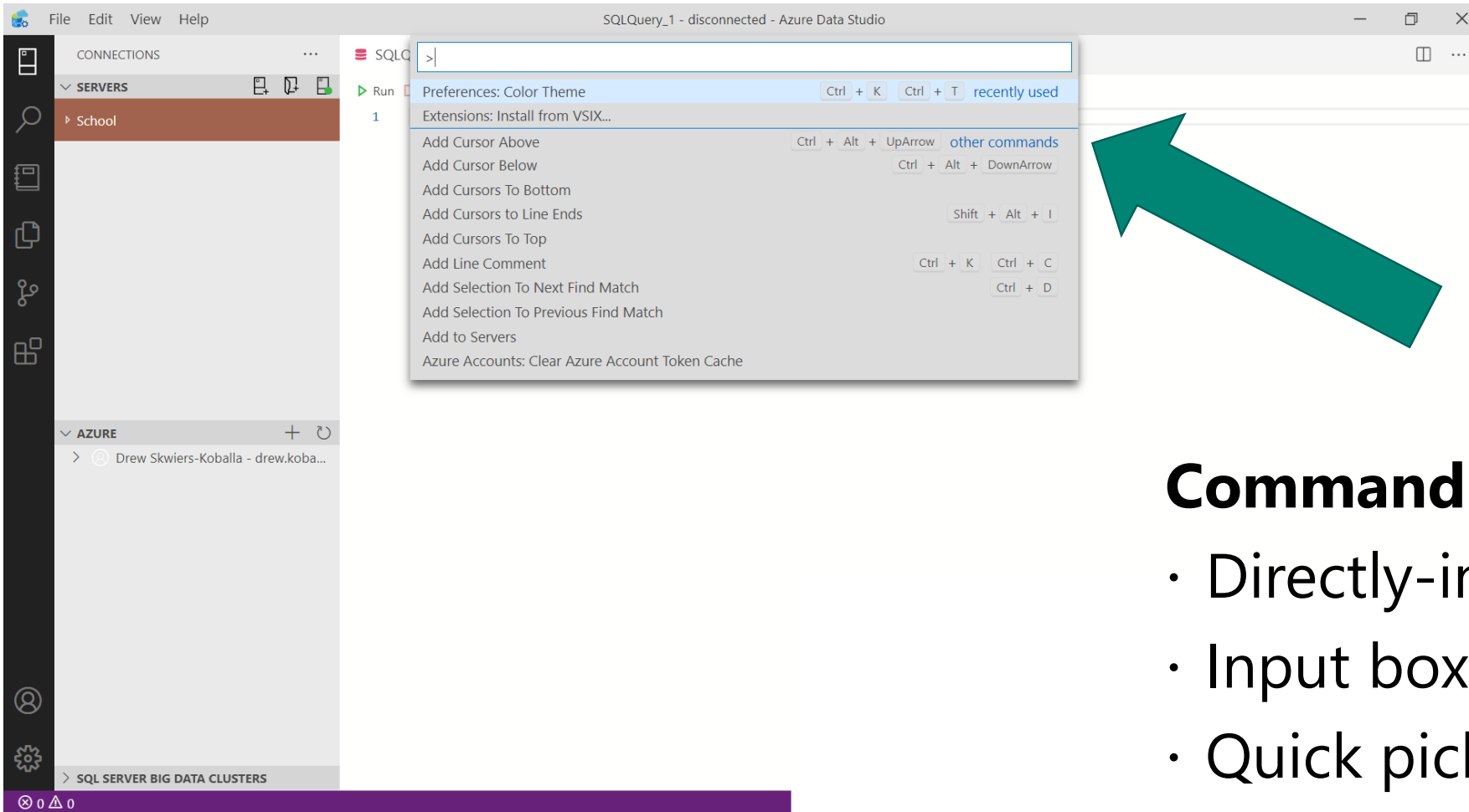
- Command palette
- Sidebar view
- Status bar items
- Webview

## **Azure Data Studio only**

- Dialog view
- Wizard view



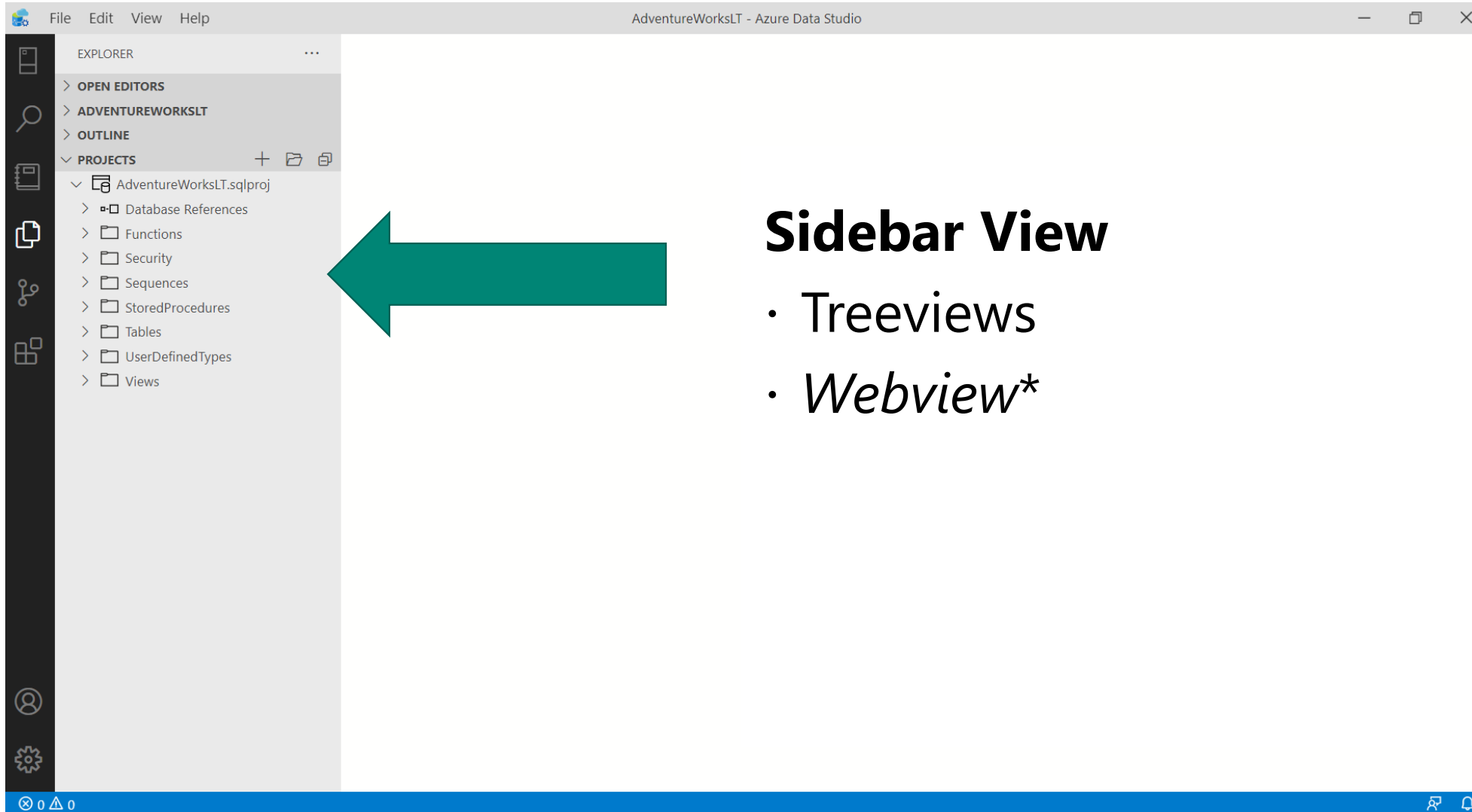
# Design Elements



## Command Palette

- Directly-invoked commands
- Input box
- Quick pick

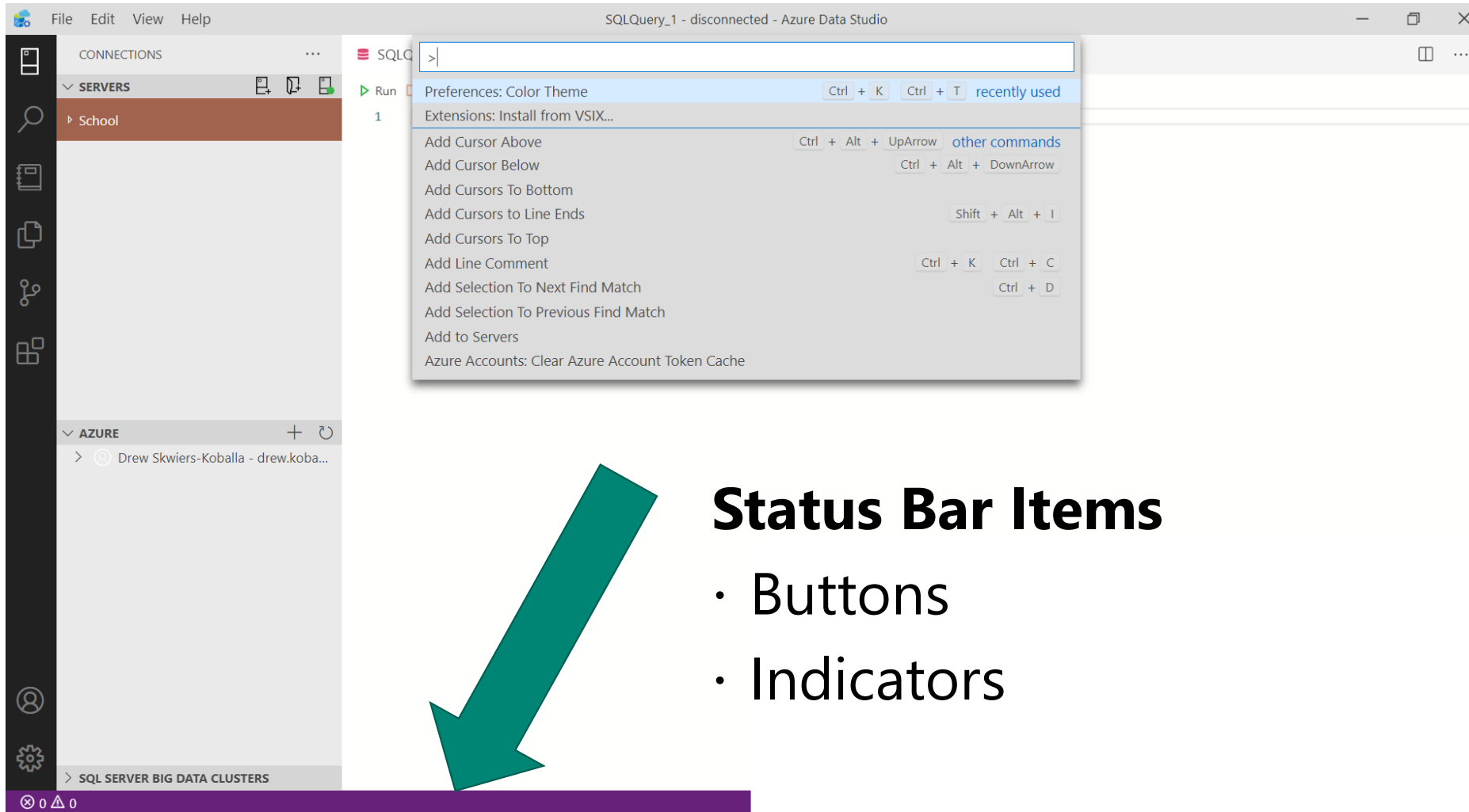
# Design Elements



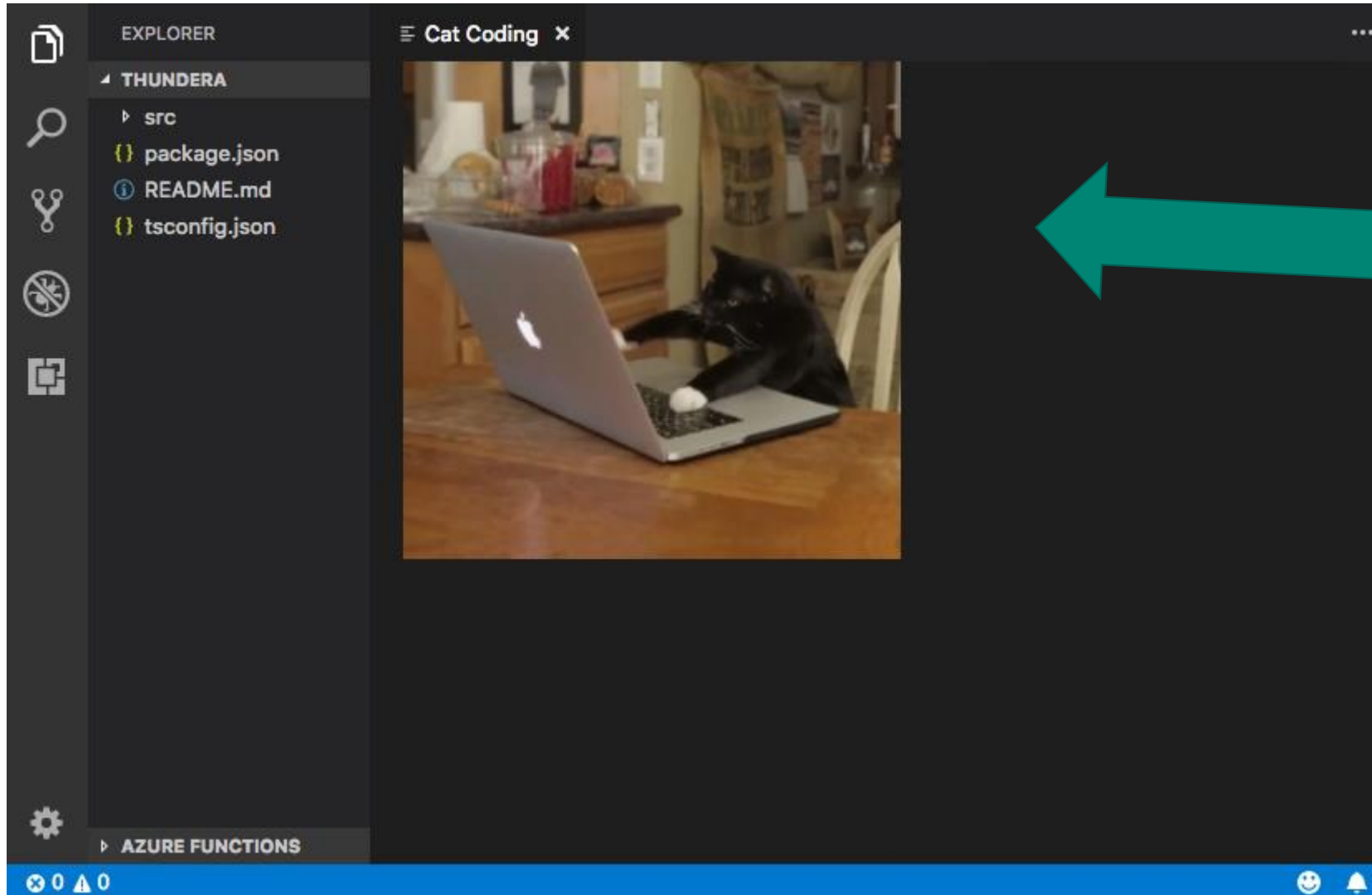
## Sidebar View

- Treeviews
- *Webview\**

# Design Elements



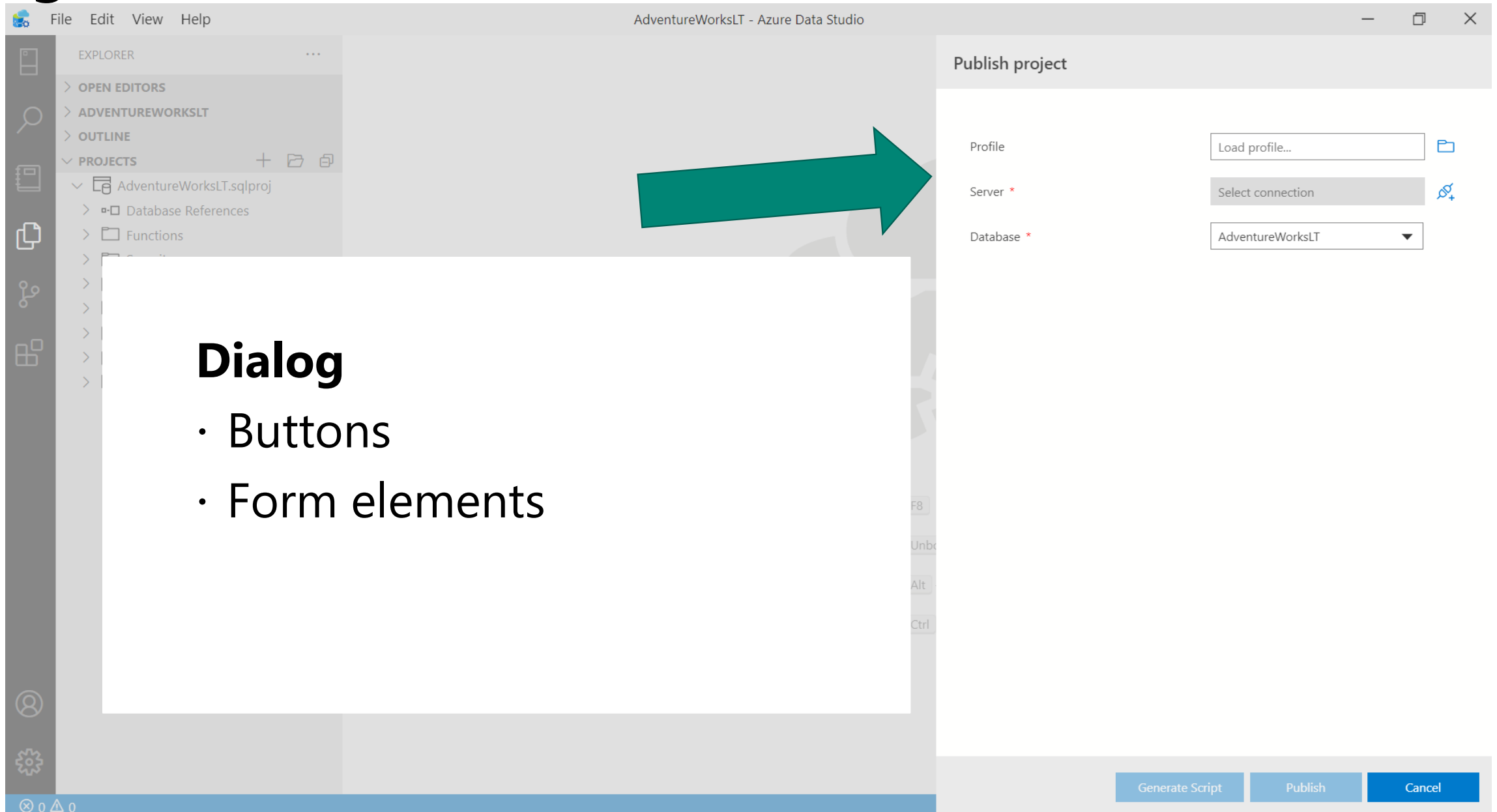
# Design Elements



## Webview

- HTML
- JavaScript

# Design Elements



# Design Elements

AdventureWorksLT - Azure Data Studio

File Edit View Help

CONNECTIONS

SERVERS

School

TestLab

192.168.6.62,61433 (sa)

Database the table is created in \*

AdventureWorksLT

Location of the file to be imported \*

Browse

New table name \*

Table schema \*

dbo

Step 1

**Specify Input File**

Server the database is in \*

192.168.6.62,61433 (sa)

Database the table is created in \*

AdventureWorksLT

Location of the file to be imported \*

Browse

New table name \*

Table schema \*

dbo

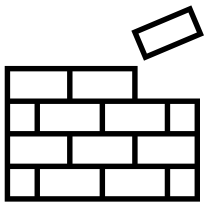
Next Cancel

**Wizard**  
(Multiple dialogs chained together)

# Contribution Points

How/what does the extension add to Azure Data Studio?

Places to connect your functionality to Azure Data Studio under the hood



- Commands
- Configuration
- Menu
- Keybindings
- Themes
- Snippets
- *Dashboard*
- *Container*


**Defined in package.json**

# Extension Generator *Types*

Contribution point(s) bundled into an extension template:

- TypeScript/JavaScript
- Dashboard Insight
- Color Theme
- Code Snippets
- Keymap
- Extension Pack
- Language Pack

```
PS C:\Users\drewk> yo azuredatstudio
```



Welcome to the Azure  
Data Studio Extension  
generator!

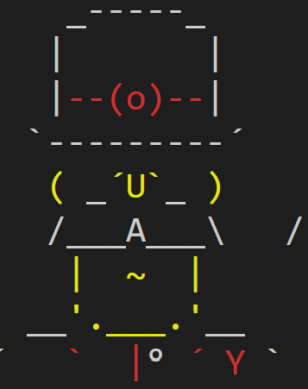
```
? What type of extension do you want to create?  
> New Extension (TypeScript)  
New Extension (JavaScript)  
New Dashboard Insight  
New Color Theme  
New Code Snippets  
New Keymap  
New Extension Pack  
(Move up and down to reveal more choices)
```



# Extension Generator *Types*

More added recently...

```
PS C:\Users\drewk> yo azuredatstudio
```



Welcome to the Azure  
Data Studio Extension  
generator!

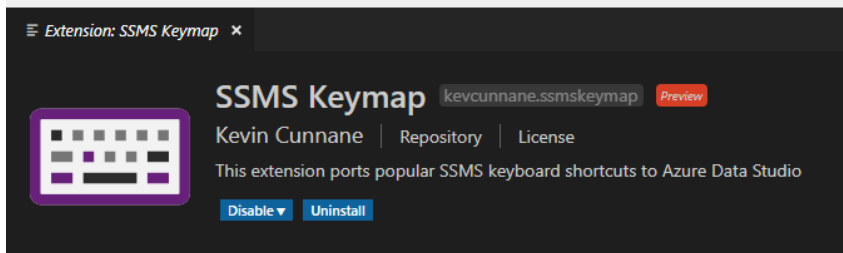
```
? What type of extension do you want to create?  
> New Extension (TypeScript)  
New Extension (JavaScript)  
New Dashboard Insight  
New Color Theme  
New Code Snippets  
New Keymap  
New Extension Pack  
(Move up and down to reveal more choices)
```

- Notebook
- Jupyter Book
- Wizard
- Dialog

# Type Examples

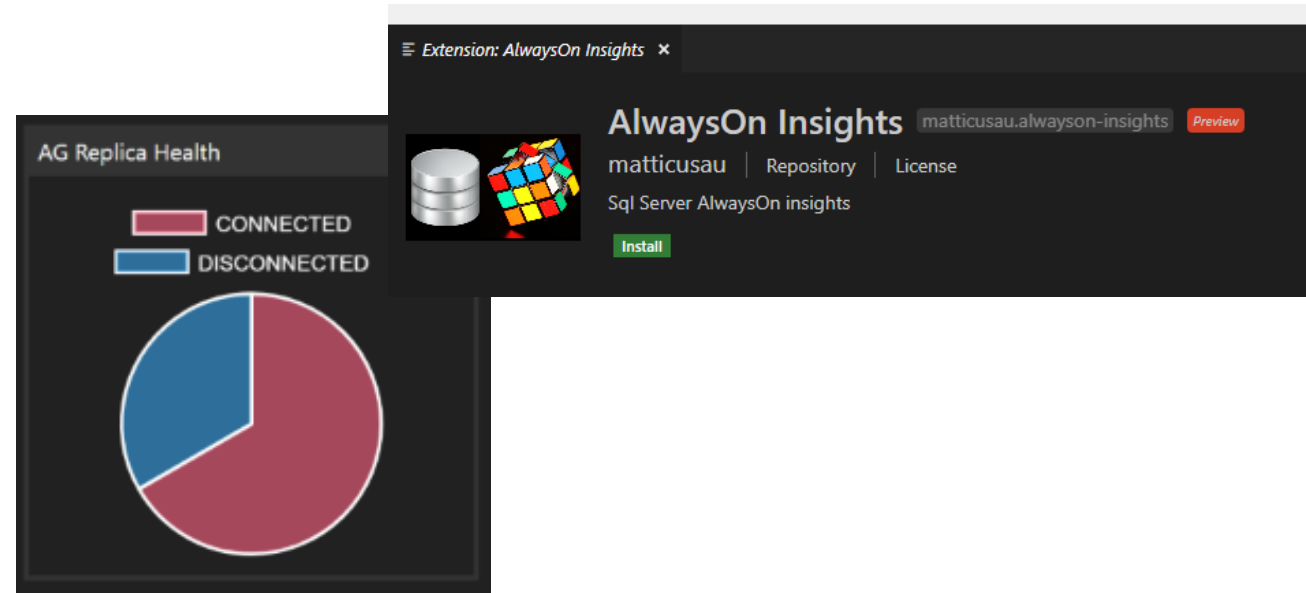
## Keymap

Ties keystrokes to commands



## Dashboard/Insight

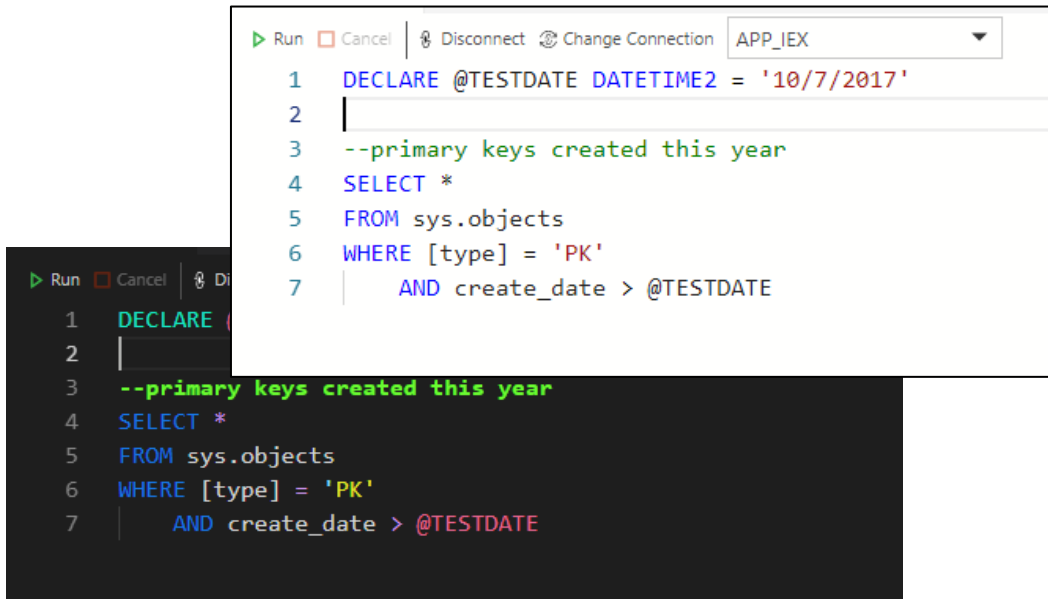
Packaged visualizations for database or server dashboards



# Type Examples

## Color Theme

Editor color customization



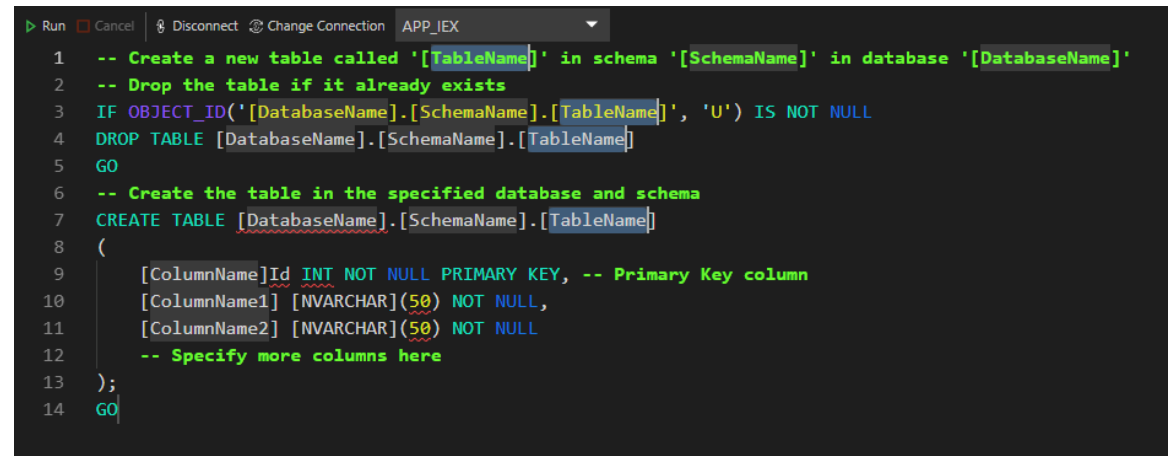
The screenshot shows two overlapping windows from SQL Server Enterprise Manager. The top window is titled 'APP\_IEX' and contains the following TSQL code:

```
1 DECLARE @TESTDATE DATETIME2 = '10/7/2017'
2
3 --primary keys created this year
4 SELECT *
5 FROM sys.objects
6 WHERE [type] = 'PK'
7      AND create_date > @TESTDATE
```

The bottom window also shows the same code, but with a dark-themed color scheme. The code is color-coded: keywords are blue, string literals are red, and comments are green.

## Snippets

Packaged TSQL with tabstops



The screenshot shows a window titled 'APP\_IEX' containing a TSQL snippet for creating a table. The code is color-coded and uses tabstops for indentation:

```
1 -- Create a new table called '[TableName]' in schema '[SchemaName]' in database '[DatabaseName]'
2 -- Drop the table if it already exists
3 IF OBJECT_ID('[DatabaseName].[SchemaName].[TableName]', 'U') IS NOT NULL
4 DROP TABLE [DatabaseName].[SchemaName].[TableName]
5 GO
6 -- Create the table in the specified database and schema
7 CREATE TABLE [DatabaseName].[SchemaName].[TableName]
8 (
9     [ColumnName]Id INT NOT NULL PRIMARY KEY, -- Primary Key column
10    [ColumnName1] [NVARCHAR](50) NOT NULL,
11    [ColumnName2] [NVARCHAR](50) NOT NULL
12    -- Specify more columns here
13 );
14 GO
```

# Type Examples

## Extension Pack

Admin Pack for SQL Server is a collection of extensions that you will download the following

- [SQL Server Agent](#)
  - List SQL Server Agents
  - View Job History with details
  - Basic Job Control tools
- [SQL Server Profiler](#)
  - Browse through extended events
  - View and manage extended events
  - Filter search of events
- [SQL Server Import](#)
  - Use the Import Flat File Wizard
- [SQL Server dacpac](#)
  - Use the Data-Tier Application Wizard

## TypeScript/JavaScript

`sp_executesql`

2

SQL

### sp\_executesql to SQL

Pejman Nikram | Repository

Convert sp\_executesql to sql

Install



### Combine Scripts

Bateleur IO | Repository

Create a single combined script

Install



# Extension Development Prerequisites

# Your Prerequisites

## Workstation OS

Windows, macOS, or Linux

*\*ChromeOS*

## Knowledge Requirements

Beginner Git

Beginner TypeScript  
*or Similar Language*

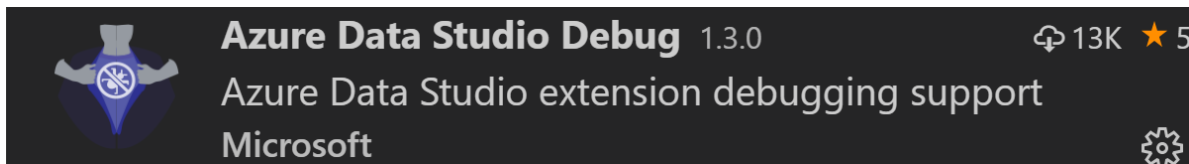
# System Prerequisites

## Applications

[VS Code](#)

[Azure Data Studio Debug](#)  
(extension for VS Code)

[Azure Data Studio](#)



## Development Tools

[Git](#)

[NodeJS](#)



# Checking Installs

## PowerShell

```
PS C:\Users\drewk> git --version  
git version 2.22.0.windows.1
```

```
PS C:\Users\drewk> node -v  
v10.16.0
```

## Bash

```
drewsk@drewsk-2018:~$ git --version  
git version 2.17.1  
drewsk@drewsk-2018:~$
```

```
drewsk@drewsk-2018:~$ nodejs -v  
v8.10.0  
drewsk@drewsk-2018:~$
```

You don't need command line experience to get started with extension development.



# System Prerequisites

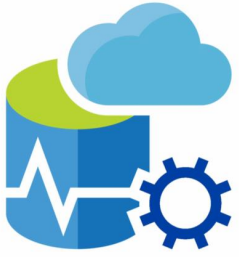
## Install Through *npm*

- TypeScript
  - JavaScript with type checking + more
- Yeoman Extension Generator
  - *yo*
  - Open source extension template builder
- VS Code Extension Manager
  - *vsce*
  - Packages extension into .vsix for installing into Azure Data Studio

```
npm install -g typescript
```

```
npm install -g yo azuredatstudio
```

```
npm install -g vsce
```



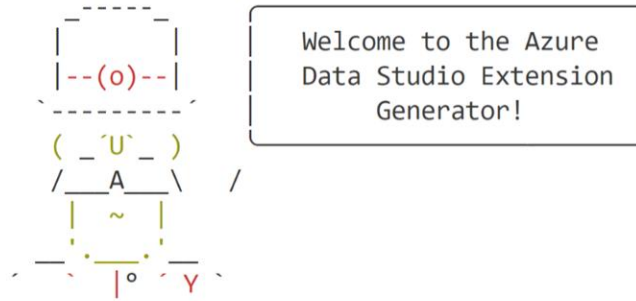
# Let's Build an Extension!

# Yo

## *yo azuredatastudio*

- For this sample, leave the *TypeScript Extension* selection
- Enter a name, identifier, description, and your publishing name
- Select the initialization of a Git repository

```
PS C:\Users\drewk\Documents\AzureDataStudio_Dev> yo azuredatastudio
```



```
? What type of extension do you want to create? New Extension (TypeScript)
? What's the display name of your extension? Sample Extension
? What's the publisher name for your extension? drewsk
? What's the unique identifier for your extension? (Appears as drewsk.your-identifier) sample-extension
? What's the description of your extension? Sample extension for learning
? Initialize a git repository? Yes
? Which package manager to use? npm
create sample-extension\.vscode\extensions.json
create sample-extension\.vscode\launch.json
create sample-extension\.vscode\settings.json
create sample-extension\.vscode\tasks.json
create sample-extension\src\test\runTest.ts
create sample-extension\src\test\suite\extension.test.ts
create sample-extension\src\test\suite\index.ts
create sample-extension\src\test\suite\update-extended-data.ts
```

# Demo Outline

## We're Going to Explore:

- Activation events: when the extension is "started", code entry point
- Contribution points: additions to the application interface
- VS Code APIs + Azure Data Studio APIs
- Alternative NodeJS packages
- *Extended extension architecture*
- *Notebooks*

# Initial Extension Files

```
.
├── .vscode
│   ├── launch.json
│   └── tasks.json
├── .gitignore
├── README.md
├── src
│   └── extension.ts
├── package.json
└── tsconfig.json
```

`.vscode` folder controls how VS Code interacts with the project

`README.md` stores documentation

**`extension.ts*` contains code entry point(s)**

**`package.json` = Extension Manifest**

# Package.json

## Activation Events

```
13     ],  
14     "activationEvents": [  
15         "onCommand:extension.sayHello",  
16         "onCommand:extension.showCurrentConnection"  
17     ],  
18     "main": "out/extension"
```

When the extension is loaded – consumes CPU/memory

- onCommand
- workspaceContains
- onFileSystem
- onView
- \* *(only when necessary)*

# Package.json

## Contribution Points

Commands

Configuration

Menu

Keybindings

Themes

Snippets

Dashboard

Container

```
"contributes": {
  "commands": [ ...
  ],
  "keybindings": [ ...
  ],
  "configuration": [
    {
      "title": "NewQueryTemplate",
      "properties": {
        "newquerytemplate.DefaultQueryTemplate": {
          "type": "array",
          "default": [
            "--set a default new query template with",
            "",
            ""
          ],
          "description": "Query text to insert into n",
        },
        "newquerytemplate.DefaultQueryLine": {
          "type": "number",
          "default": -1
        },
        "newquerytemplate.DefaultQueryCharacter": {
          "type": "number",
          "default": -1
        }
      }
    }
  ]
}
```

# Activation vs Contribution Point

## Activation

Creates UI element  
statusBarToggle

## Contribution

Command for use at a later  
time

```
5  export function activate(context: vscode.ExtensionContext) {
6
7      const statusBarToggle = new StatusBarToggle();
8
9      context.subscriptions.push(vscode.commands.registerCommand('dsk.enableDemoMode', () => {
10         var theSettings = vscode.workspace.getConfiguration();
11
12         let oldFontSize = theSettings.get('editor.fontSize');
13         let newFontSize = theSettings.get('demomode.demoFontSize');
14         theSettings.update('editor.fontSize', newFontSize, vscode.ConfigurationTarget.Global);
15         theSettings.update('demomode.originalFontSize', oldFontSize, vscode.ConfigurationTarget.Global);
16
17         statusBarToggle.toggle(true);
18     }));
19
```

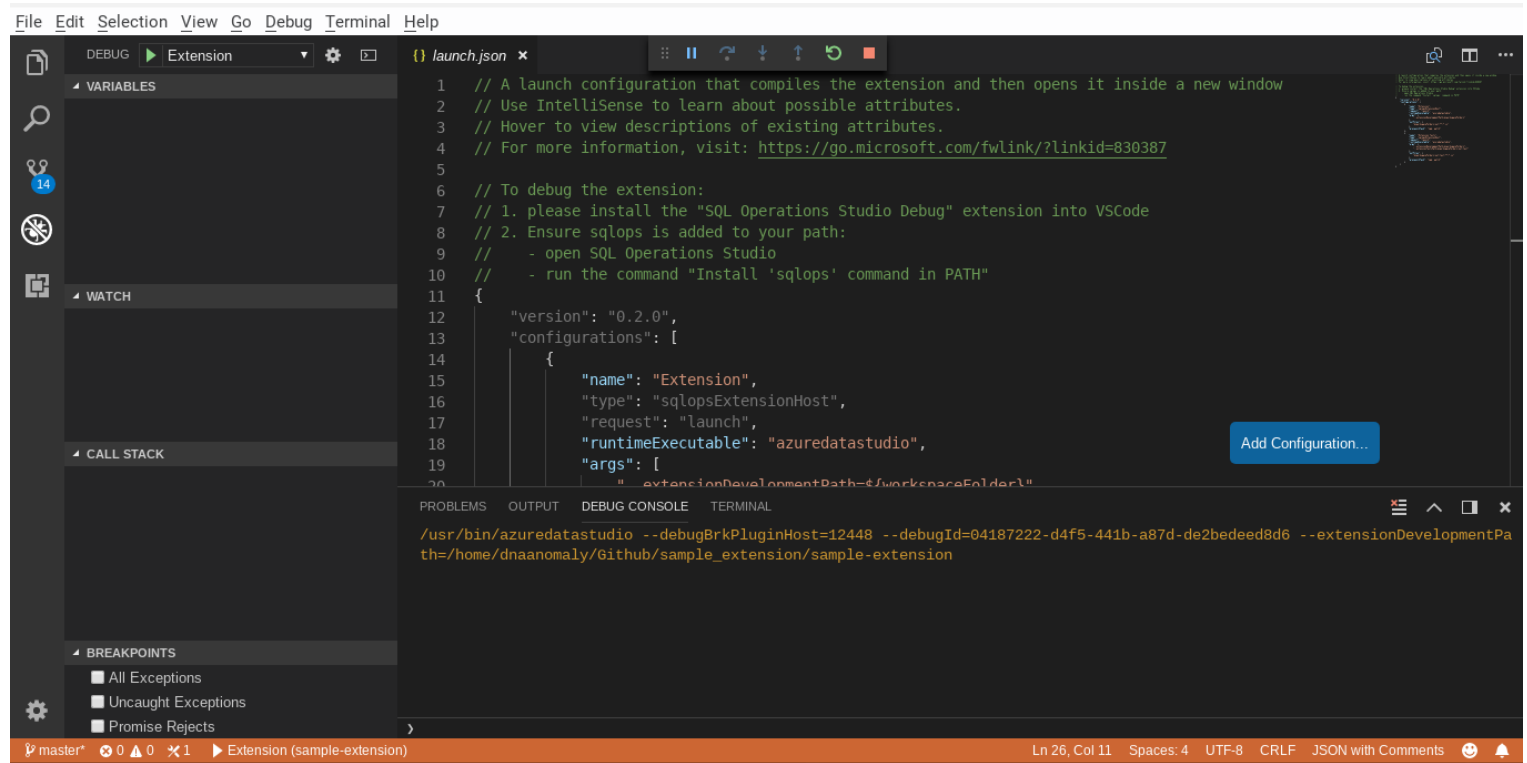


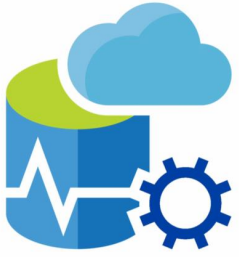
# Test Extension

## Select Start Debugging from the Debug menu

A special *Extension Development* instance of ADS opens

Test the extension  
from the command  
palette (ctrl+shift+P)





# Building Extension Features

# Azure Data Studio APIs

## A few examples...



### connection

ConnectionProfile

getCurrentConnection()

ServerInfo

### objectexplorer

NodeInfo

Traverse object tree

findNodes()

<https://github.com/microsoft/azuredatstudio/blob/master/src/sql/azdata.d.ts>

# Azure Data Studio APIs

**DataProvider** extends to:

ConnectionProvider – server connections

MetadataProvider – object info

ScriptingProvider – “script as”

QueryProvider – execute queries

ProfilerProvider – extended events

AgentServicesProvider – SQL agent

BackupProvider – backup and restore



<https://github.com/microsoft/azuredatstudio/blob/master/src/sql/azdata.d.ts>

# Let's Try It.

How many downvotes does the StackOverflow user with the most downvotes have?

We need to run a command that pulls some data from the database.

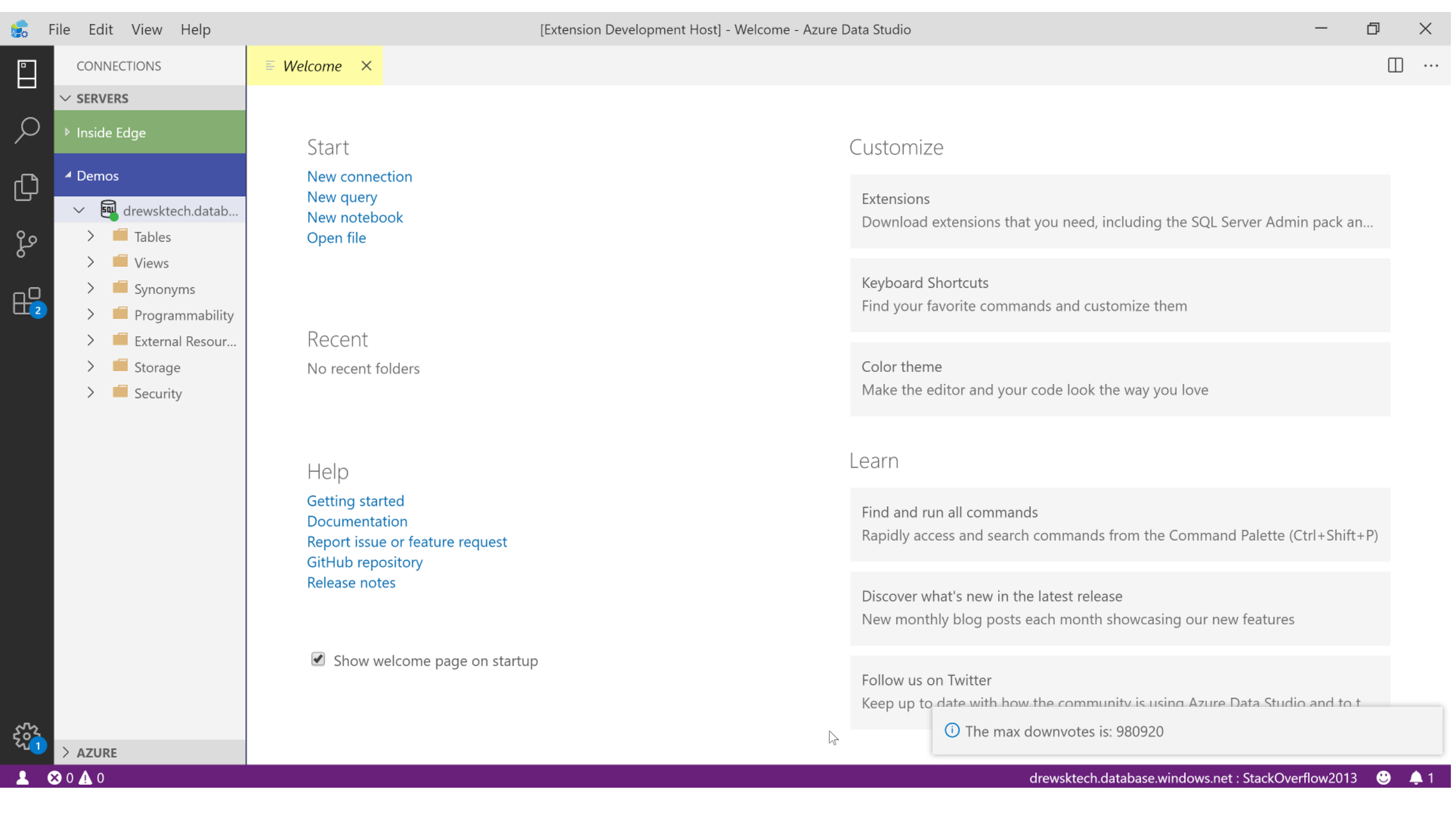
1. Get the current connection
2. Setup the query
3. Display the results to the user

# extension.ts

```
29
30 context.subscriptions.push(vscode.commands.registerCommand('extension.getDown', async () => {
31     let connection = await azdata.connection.getCurrentConnection();
32     let connectId = connection.connectionId;
33     if (connection && connectId) {
34         let query = `SELECT MAX(DOWNVOTES) AS DOWNVOTES FROM USERS`;
35
36         let connectionUri = await azdata.connection.getUriForConnection(connectId);
37         let queryProvider = azdata.dataprotocol.getProvider<azdata.QueryProvider>(connection.providerId,
            azdata.DataProviderType.QueryProvider);
38
39         let results = await queryProvider.runQueryAndReturn(connectionUri, query);
40         let cell = results.rows[0][0];
41
42         let downvotes = cell.displayValue;
43         vscode.window.showInformationMessage('The max downvotes is: '+downvotes);
44     }
45 });
46
47 }));
```

# package.json

```
13     ],
14     "activationEvents": [
15         "onCommand:extension.sayHello",
16         "onCommand:extension.showCurrentConnection",
17         "onCommand:extension.getDown"
18     ],
19     "main": "./out/extension",
20     "contributes": {
21         "commands": [
22             {
23                 "command": "extension.sayHello",
24                 "title": "Hello World"
25             },
26             {
27                 "command": "extension.showCurrentConnection",
28                 "title": "Show Current Connection"
29             },
30             {
31                 "command": "extension.getDown",
32                 "title": "Show Max Downvotes"
33             }
34         ]
35     },
```



CONNECTIONS

Welcome ×

SERVERS

Inside Edge

Demos

drewsktech.datab...

- Tables
- Views
- Synonyms
- Programmability
- External Resour...
- Storage
- Security

Start

- New connection
- New query
- New notebook
- Open file

Recent

No recent folders

Help

- Getting started
- Documentation
- Report issue or feature request
- GitHub repository
- Release notes

☒ Show welcome page on startup

Customize

Extensions

Download extensions that you need, including the SQL Server Admin pack an...

Keyboard Shortcuts

Find your favorite commands and customize them

Color theme

Make the editor and your code look the way you love

Learn

Find and run all commands

Rapidly access and search commands from the Command Palette (Ctrl+Shift+P)

Discover what's new in the latest release

New monthly blog posts each month showcasing our new features

Follow us on Twitter

Keep up to date with how the community is using Azure Data Studio and to t

*The max downvotes is: 980920*

AZURE



# VS Code APIs



## window

Text editors

Terminals

User inputs

## workspace

Currently open  
folder

FileSystem

## tasks

Background  
scripts/processes

```
const editor = vscode.window.activeTextEditor;  
let cursorPosition = editor.selection.start;  
let lineCount = editor.document.lineCount;
```

More Info: <https://code.visualstudio.com/api/references/vscode-api>

# NodeJS Packages

## Additional Functionality

Parsing SQL, JSON, etc

HTTP requests

Filesystem access

Operating system info

## npm/yarn

Package managers for NodeJS:

- Yarn - performance
- npm - interoperability

Start here: <https://www.npmjs.com/>

# Install an Additional Node Package

```
npm install <package-name>
```

Run from the terminal

```
import * as <alias> from '<package-name>';
```

Add to the top of a Typescript file

# Let's Try It.

What operating system (OS) is the user running?

We need to run a command that is different based on the OS of the user.

1. Use an additional JS package for operating system information.
2. Use the OS type to return different information to the user.

```
import * as vscode from 'vscode';
```

```
// The module 'azdata' contains the Azure Data Studio extensibility API  
// This is a complementary set of APIs that add SQL / Data-specific functionality to the app  
// Import the module and reference it with the alias azdata in your code below
```

```
import * as azdata from 'azdata';
```

```
import * as os from 'os';
```

• • • • •

```
// The commandId parameter must match the command field in package.json
```

```
context.subscriptions.push(vscode.commands.registerCommand('extension.sayHello', () => {
```

```
    // The code you place here will be executed every time your command is executed
```

```
    // Display a message box to the user
```

```
    vscode.window.showInformationMessage('Hello World!');
```

```
    let hostname = os.hostname();
```

```
    if (os.type() !== 'Windows_NT') {
```

```
        vscode.window.showInformationMessage('Aren\'t you glad Azure Data Studio runs on other platforms
```

```
    } else {
```

```
        vscode.window.showInformationMessage('You can still use SSMS when you have the RAM to spare!');
```

```
    }
```

```
    }));
```

Make the editor and your code look the way you love

## Learn

Find and run all commands


Rapidly access and search commands from the Command Palette (Ctrl+Shift+P)

Discover what's new in the latest release

New monthly blog posts each month showcasing our new features

Follow us on Twitter

Keep up to date with how the community is using Azure Data Studio and to talk directly wit

 You can still use SSMS when you have the RAM to spare!

 Hello World!



# Extended Architecture

## Native Code

Node binaries

.NET Core

Package with extension or  
install on activation

*Cross Platform*

## Web Framework

Support for rendering  
content in a Webview

# Webview Example:

File Edit View Help

SQL Search Results: vote - Azure Data Studio

drewnsktech.database.windows.net:StackOverflow2013

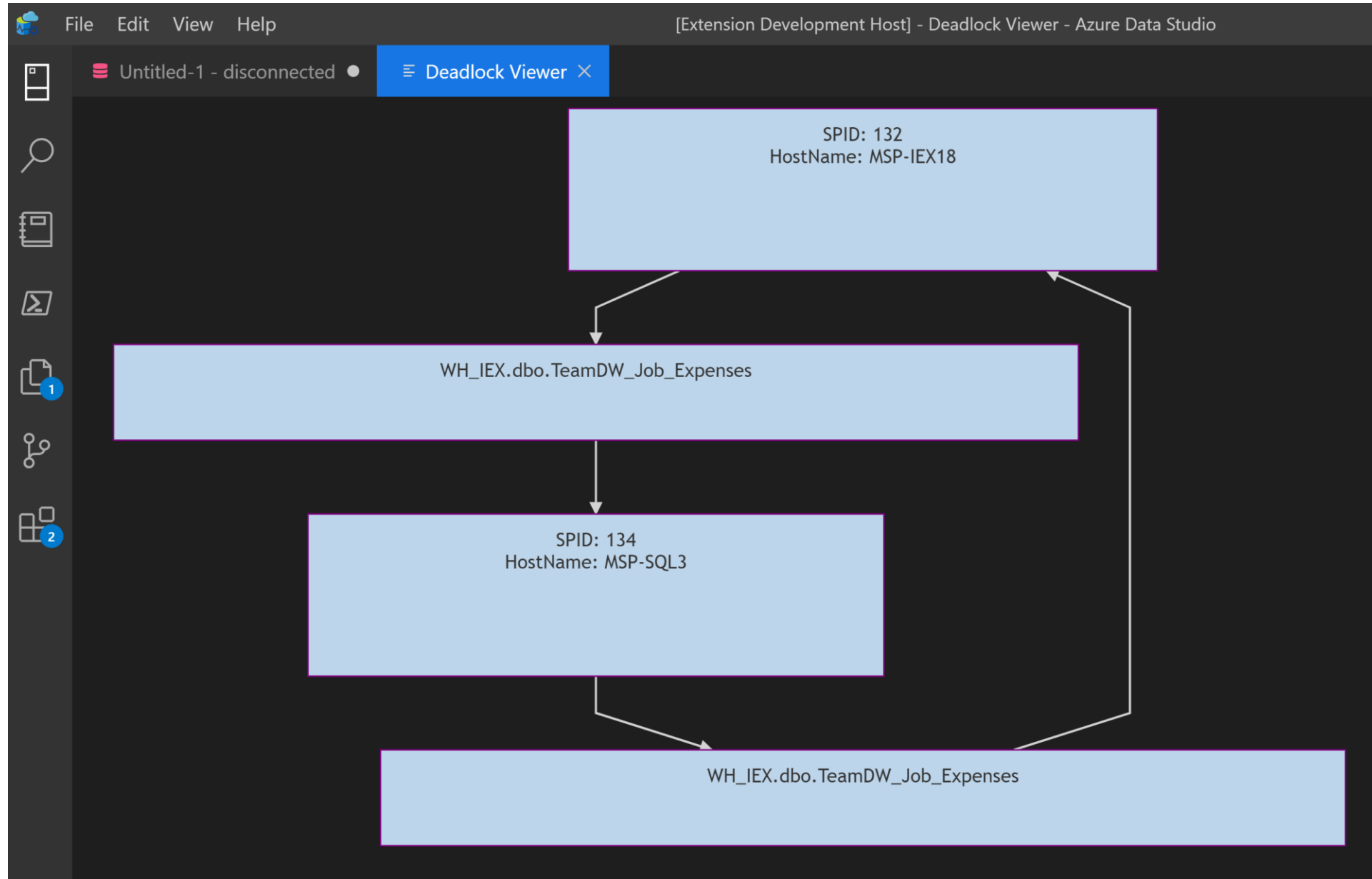
SQL Search Results: vote X

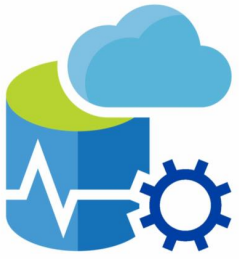
Scope: drewnsktech.database.windows.net, StackOverflow2013

Object name	Option	Schema	Database	Type	Matches on
VoteTypes	⋮	dbo	StackOverflow2013	UserTable	Column
Votes	⋮	dbo	StackOverflow2013	UserTable	Column
Users	⋮	dbo	StackOverflow2013	UserTable	Column
PK_VoteType_Id	⋮	dbo	StackOverflow2013	PrimaryKeyConstraint	Name
VoteTypes	⋮	dbo	StackOverflow2013	UserTable	Name
PK_Votes_Id	⊙	dbo	StackOverflow2013	PrimaryKeyConstraint	Name
Votes	⋮	dbo	StackOverflow2013	UserTable	Name



# Webview Example:





# Notebooks... as an Extension

# Notebook-Based Extension Templates

**Notebook** = a single file that contains of code and text cells

**Jupyter book** = an organized collection of notebooks packaged with a table of contents

The extension generator (*yo azuredatastudio*) contains templates for extensions that wrap around a notebook or Jupyter book.

These extension templates are best edited from Azure Data Studio.

**Must package extension to test,**  
**run *vsce package* from terminal**

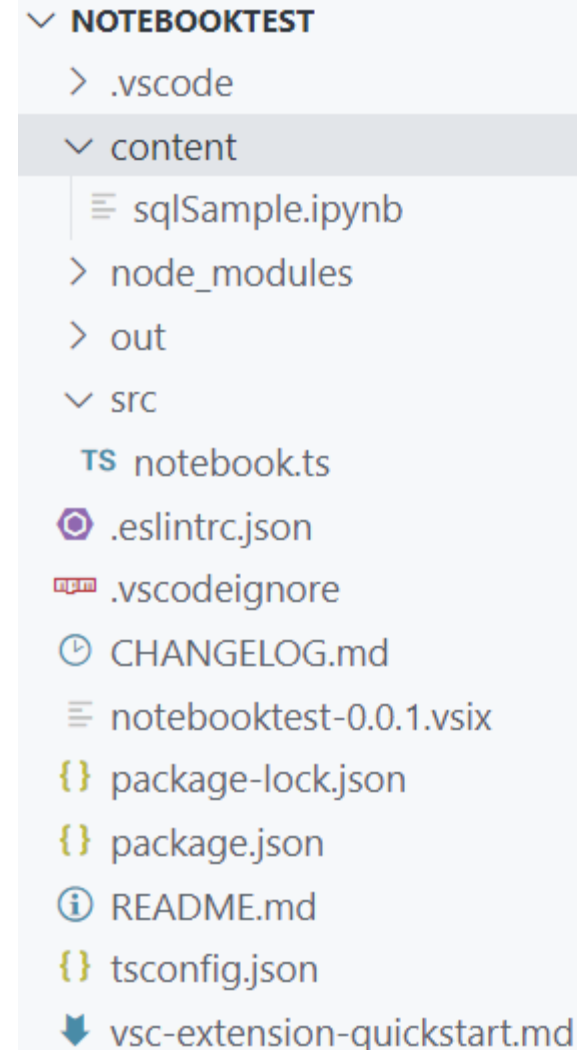
# Notebook(s) Extension Template

- Notebook(s) in the *content* folder are bundled into extension
- Uses showNotebookDocument API to access the files

**Must package extension to test, run**

*vsce package*

**from terminal**

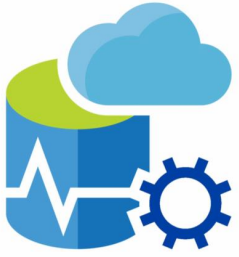


```
▼ NOTEBOOKTEST
  > .vscode
  ▼ content
    sqlSample.ipynb
  > node_modules
  > out
  ▼ src
    TS notebook.ts
  .eslintrc.json
  .vscodeignore
  CHANGELOG.md
  notebooktest-0.0.1.vsix
  package-lock.json
  package.json
  README.md
  tsconfig.json
  vsc-extension-quickstart.md
```

# Jupyter Book Extension Template

- Notebook(s) and Markdown in the *content* folder are bundled into extension
- Structure of Jupyter Book dictated by *toc.yml* in *\_data* folder
- Jupyter Book title/description set in *\_config.yml*
- Uses `bookTreeView.openBook` API to access the files

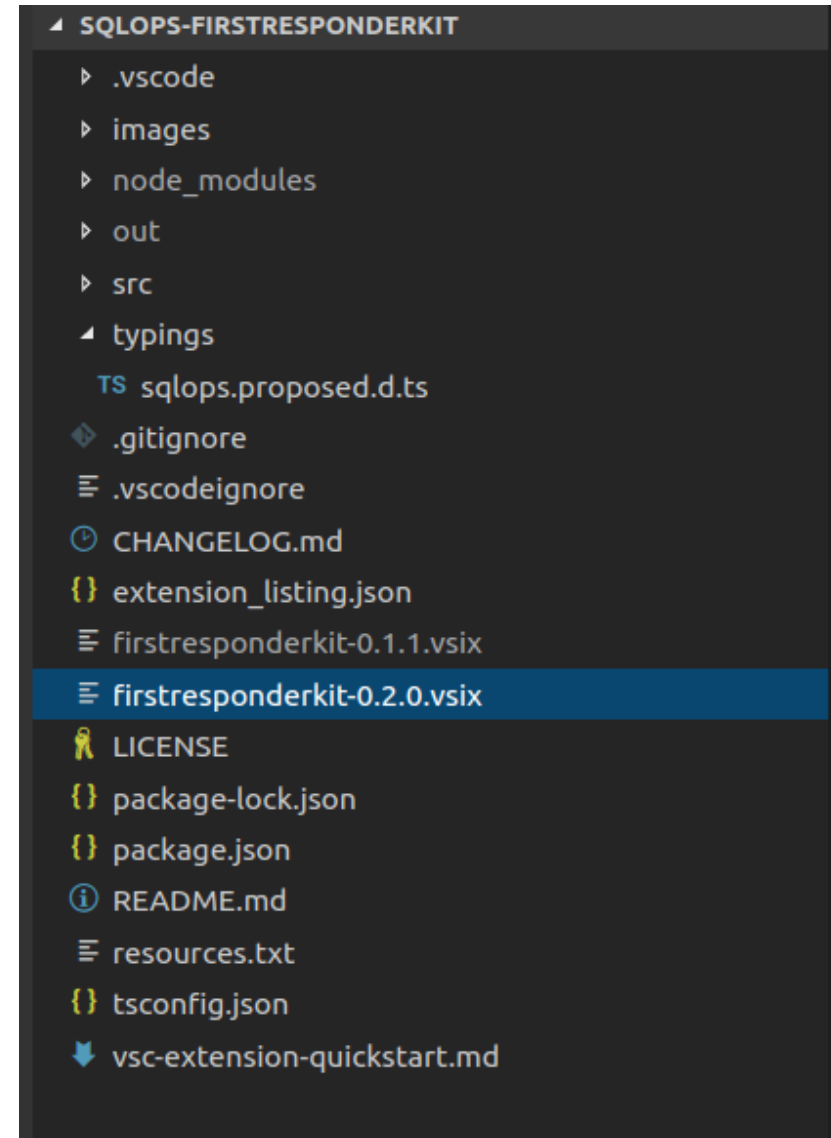




# Package and Publish

# Package to .vsix

- (update README.md)
- Run **vsce package** from terminal
- Load .vsix into Azure Data Studio via command palette



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: powershell ▼



```
PS C:\Users\drewk\Documents\AzureDataStudio_Dev\best-pass-summit> vsce package
Executing prepublish script 'npm run vscode:prepublish'...
```

```
> best-pass-summit@0.0.1 vscode:prepublish C:\Users\drewk\Documents\AzureDataStudio_Dev\best-pass-summit
> npm run compile
```

```
> best-pass-summit@0.0.1 compile C:\Users\drewk\Documents\AzureDataStudio_Dev\best-pass-summit
> tsc -p ./
```

**WARNING** A 'repository' field is missing from the 'package.json' manifest file.

Do you want to continue? [y/N] y

**DONE** Packaged: C:\Users\drewk\Documents\AzureDataStudio\_Dev\best-pass-summit\best-pass-summit-0.0.1.vsix (12 files, 6.67KB)

**INFO**

The latest version of vsce is 1.69.0 and you have 1.65.0.

Update it now: npm install -g vsce

```
PS C:\Users\drewk\Documents\AzureDataStudio_Dev\best-pass-summit> █
```



# Options for Publishing

## A philosophical question:

### Open Source

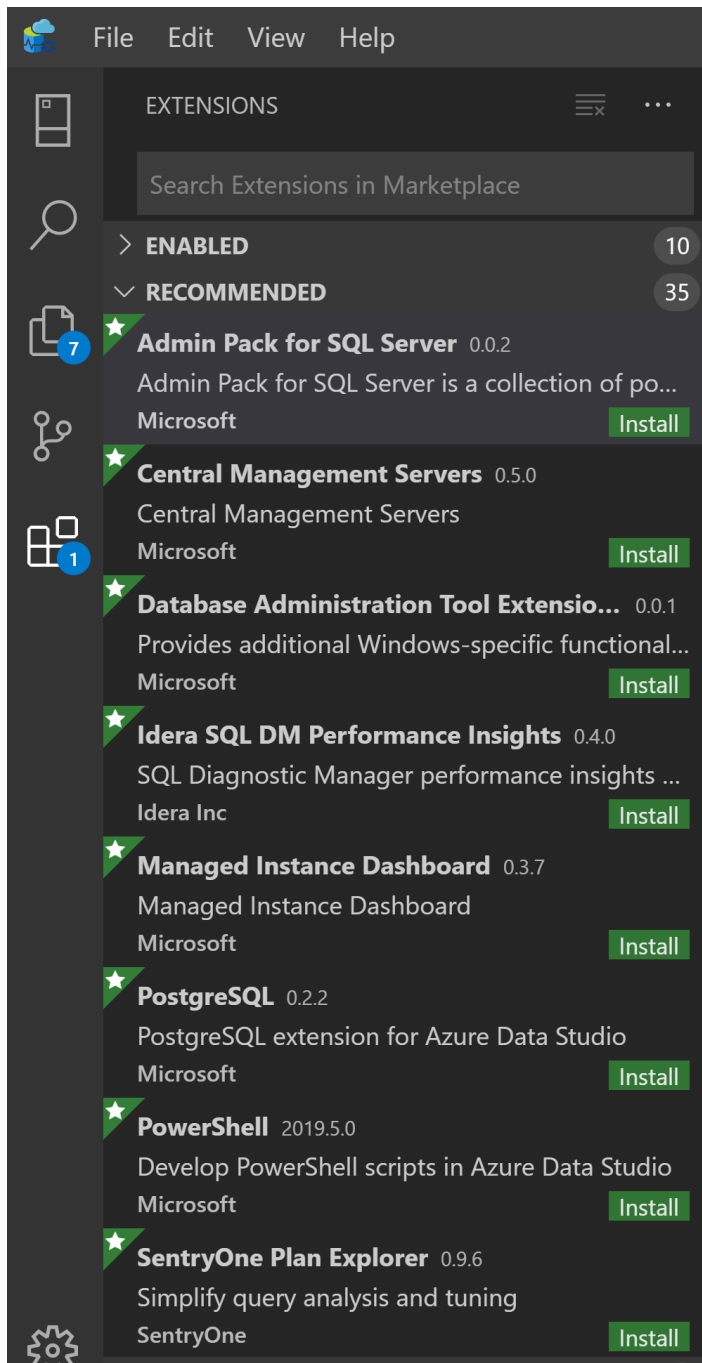
Others can view your code,  
contribute improvements or  
fixes

*Derivative future work*

### Closed Source

Protect your intellectual  
property

Others can download and  
install the .vsix



# Azure Data Studio Marketplace

Pull Request to release/extensions  
branch

Update [extensionsGallery.json](#) and  
[extensionsGallery-insiders.json](#)

# extensionsGallery.json

files

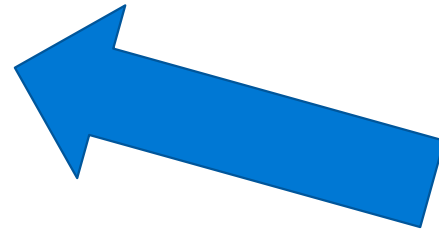
**Values don't have  
to be links to  
GitHub**

- SQLOps.DownloadPage
- VisualStudio.Services.Icons.Default, *optional*
- VisualStudio.Services.Content.Details
- VisualStudio.Code.Manifest
- VisualStudio.Services.Content.License

# extensionsGallery.json

## properties

- VisualStudio.Code.ExtensionDependencies
- VisualStudio.Services.Links.Source, *optional*
- AzDataEngine
- VisualStudio.Code.Engine



**Set minimum  
versions**



# Let's Wrap Up

# Key Takeaways

## Building an Extension in Azure Data Studio

- You can do it.
- VS Code extension + Azure Data Studio APIs
- Extension functionality can derive from both NodeJS and extended capabilities
- Extensions can be monetized and/or proprietary
- Open source contributions elevate the data platform community's capabilities

# Additional Resources

- [https://github.com/dzsquared/AzureDataStudio\\_ExtensionDevelopment](https://github.com/dzsquared/AzureDataStudio_ExtensionDevelopment)
- <https://code.visualstudio.com/docs/extensions/overview>
- <https://docs.microsoft.com/sql/azure-data-studio/extensions/extension-authoring>
- <https://github.com/microsoft/vscode-extension-samples>
- <https://github.com/microsoft/azuredatstudio/tree/main/extensions>
- <https://medium.com/@kevcunnane/extending-sql-operations-studio-hello-connected-world-part-1-of-n-e868542c6157>
- <https://medium.com/ingeniouslysimple/how-we-built-an-extension-for-sql-operations-studio-f93532ce4456>
- <https://cultivatehq.com/posts/how-we-built-a-visual-studio-code-extension-for-iot-prototyping/>

Ask questions anytime @sysadmindrew