Thank you, Sponsors!

SEMINOLE STATE COLLEGE OF FLORIDA

Microsoft

AMD

COZYROC

ROYAL BLUE ANALYTICS

SQLGrease

O'REILLY®

PER SCHOLAS

Join Our Local User Groups:

SQLOrlando

ONETUG Orlando .NET User Group

Visual Studio SQL Projects

- https://tinyurl.com/SqlProjCodeCamp
  - (sample code and slides)

LinkedIn https://www.linkedin.com/in/juantarquino

Juan Pablo Tarquino
Software Engineer at SouthState Bank

Intro

- Juan Pablo Tarquino
- Software Developer with about 20 years of experience
  - LinkedIn
    - https://www.linkedin.com/in/juantarquino
  - Blog (not updated often ☹)
    - https://jptarqu.blogspot.com
- Sample repo for this session:
  - https://tinyurl.com/SqlProjCodeCamp

Today’s talk

- Level: mostly beginner, with some deeper dives
- Outline
    - Why SQL Projects
    - How to Create SQL Projects
    - How to Deploy (including CI/CD integration)
    - How to do Automated Testing
- Conference sessions are great! They allow interactions with speakers and other attendees.

Once upon a time in Pizzas’R’Us™…

- Pizzas’R’Us™ is a small but profitable company, trying to be as nimble as possible, providing pizza services to restaurants and super-markets.

Once upon a time in Pizzas’R’Us™…

- Alexandria
  - Junior Dev always seeking the most efficient and safe way to accomplish the business goals
- Michael J
  - Alexandria’s co-worker, also a junior dev, working hard with Alexandria, wearing multiple hats at the company.
- David
  - Alexandria’s and Michael’s boss. A rocker by night, a manager by day.

The Business Goal

- Develop an app to help the Business Units manage and publish the weekly podcasts. The podcasts are mostly interviews with Pizza Chefs all over the world.
- The app must maintain state in a database, MS SQL Server is what the company uses because it already paid a hefty license fee for it due to other 3rd party software that uses it.
- The data needs to be able to be queried from other tools/apps to help the BU audit and ask questions about how well the podcast is progressing.

Initial attempts

SSMS

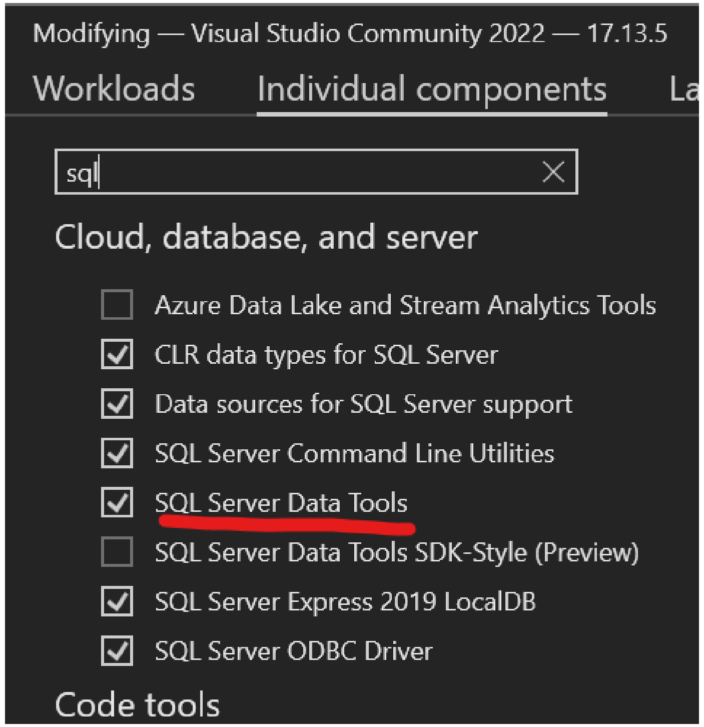
- Need source control.

.NET ORM, (e.g.: Entity Framework)

- Cannot define views and Stored Procedures in C#, still relies on SQL code.
- Migration mechanism to update the actual db does not look right (not declarative, delta operations vs desired state).

Visual Studio SQL Projects to the rescue

- Two types
  - “Legacy” MSBuild based proj file format
  - SDK-style proj file (Preview)
- Similar to other VS projects
  - IntelliSense and CoPilot
  - Find references
- Folder organization
- Source control
- “New Item” menu, examples:
  - Roles
  - SQL types



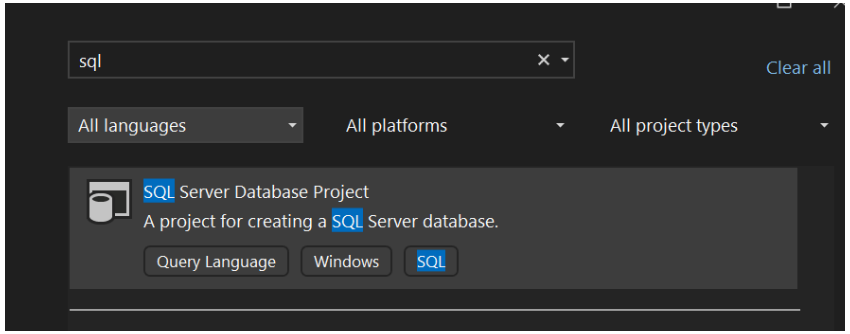
Modifying — Visual Studio Community 2022 — 17.13.5

sql

Cloud, database, and server

□ Azure Data Lake and Stream Analytics Tools
☑ CLR data types for SQL Server
☑ Data sources for SQL Server support
☑ SQL Server Command Line Utilities
☑ SQL Server Data Tools
□ SQL Server Data Tools SDK-Style (Preview)
☑ SQL Server Express 2019 LocalDB
☑ SQL Server ODBC Driver

Code tools

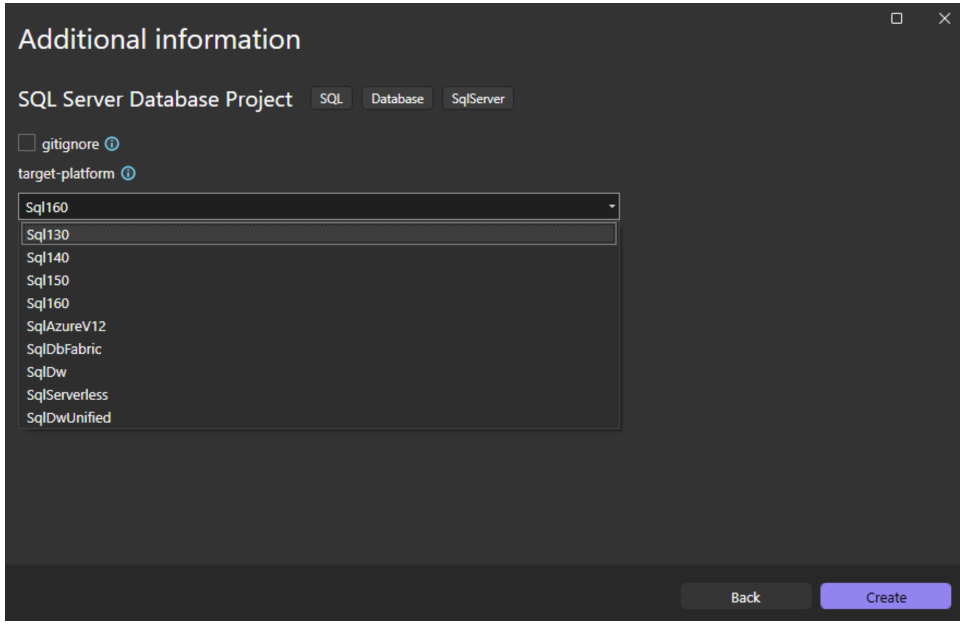


sql

SQL Server Database Project
A project for creating a SQL Server database.

SDK style projects

- dotnet tool install -g microsoft.sqlpackage
- dotnet new install microsoft.build.sql.templates
  - Installs template for cli and also for Visual Studio



Additional information

SQL Server Database Project

□ gitignore

target-platform

Sql160

What about deployments?

- I have the database defined in Visual Studio, how do I “push” it to an actual SQL server?

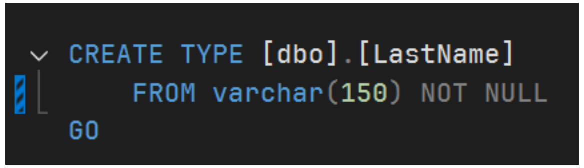
Deployment

- Right-click publish
- What happens in a deployment
  - “Compilation” to DACPAC, catches syntax and reference errors.
  - “Publishing” of sql objects to destination
- CI/CD
- Credentials
- JIT Access suggestions

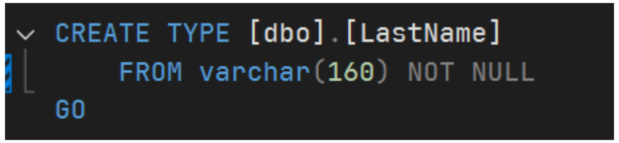
What happens in a Deployment?

1. VS compares the objects defined in the SQL project vs what is on the server.
2. Anything that is in the SQL project but not in the server is scripted as an "add" (e.g. CREATE VIEW).
3. Anything that is in the SQL project and the SQL server but differs in definition (e.g. a view has a different order of columns or an extra column), gets changed. Sometimes via an alter statement and sometimes via a drop/create. The deployment engine decides that for you.
    1. It is important to note that, by default, the deployment engine will NOT do any changes that have the possibility of losing data in the server. For example, a column being removed from a table or a column changing to a smaller type.
    2. Removing a column from a View or Stored Proc does not constitute data loss. You might be thinking, if the view is missing the column would not that affect anything else that depends on the column? The answer is yes and there are two ways to prevent issues:
        1. Any SQL object that refers to that column within the db, will fail to compile in the project so you would know that you are referencing something that is now missing so you would not even be able to get to the deployment step because compilation would fail.
        2. If your external code references that missing column, then you would do the same thing you would do if you .net public API has a breaking change. We will talk more about it later when we talk of the db as a service.
4. Anything that is not in this SQL project, but it is in the db, will NOT be touched by default.
5. Database settings (such as Read Committed/Snapshot)are applied by default.

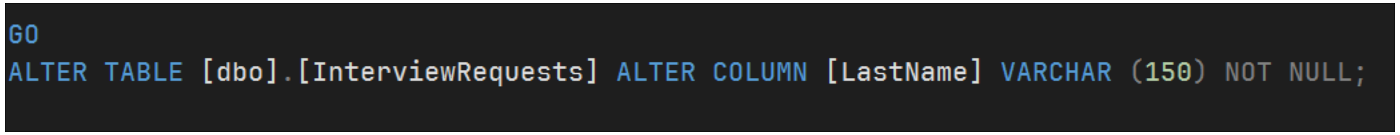
Change Sample – UDT change



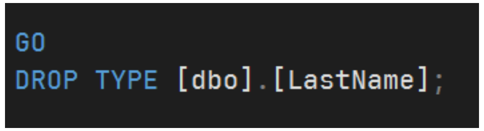
```
CREATE TYPE [dbo].[LastName]
    FROM varchar(150) NOT NULL
GO
```



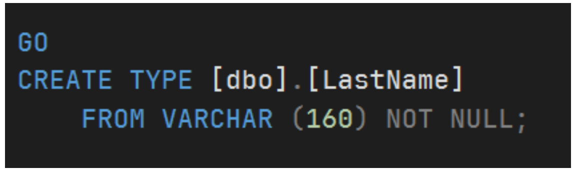
```
CREATE TYPE [dbo].[LastName]
    FROM varchar(160) NOT NULL
GO
```



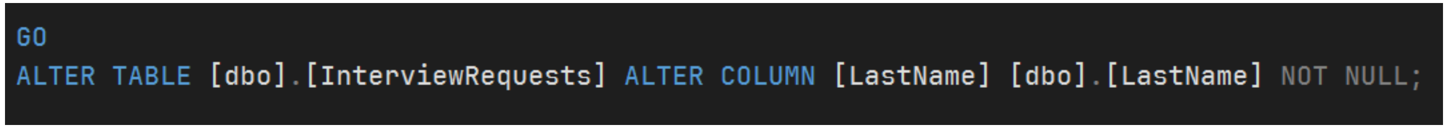
```
GO
ALTER TABLE [dbo].[InterviewRequests] ALTER COLUMN [LastName] VARCHAR (150) NOT NULL;
```



```
GO
DROP TYPE [dbo].[LastName];
```



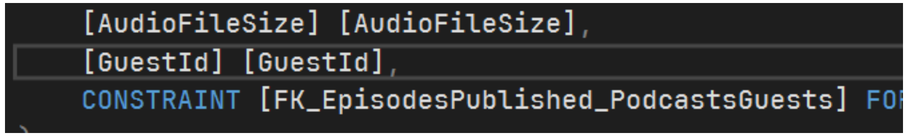
```
GO
CREATE TYPE [dbo].[LastName]
    FROM VARCHAR (160) NOT NULL;
```



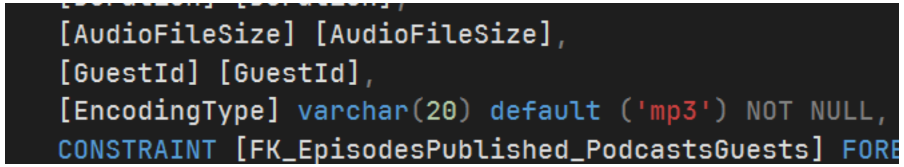
```
GO
ALTER TABLE [dbo].[InterviewRequests] ALTER COLUMN [LastName] [dbo].[LastName] NOT NULL;
```

1. Changes the column to the generic equivalent of the type before the change
2. Then drops and re-creates the type with the new size.
3. Then it assigns the type to the column again.

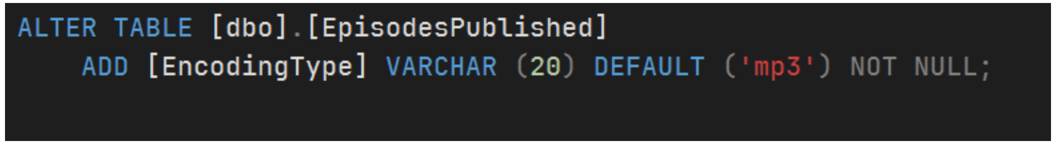
Change Sample – New column (at end)



```
    [AudioFileSize] [AudioFileSize],
    [GuestId] [GuestId],
    CONSTRAINT [FK_EpisodesPublished_PodcastsGuests] FO
```



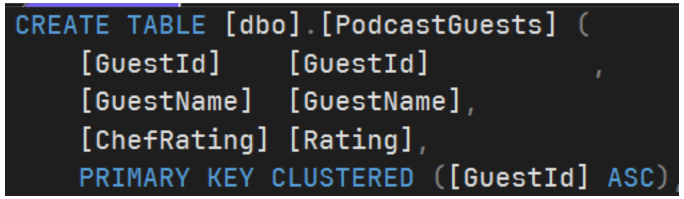
```
    [AudioFileSize] [AudioFileSize],
    [GuestId] [GuestId],
    [EncodingType] varchar(20) default ('mp3') NOT NULL,
    CONSTRAINT [FK_EpisodesPublished_PodcastsGuests] FORE
```



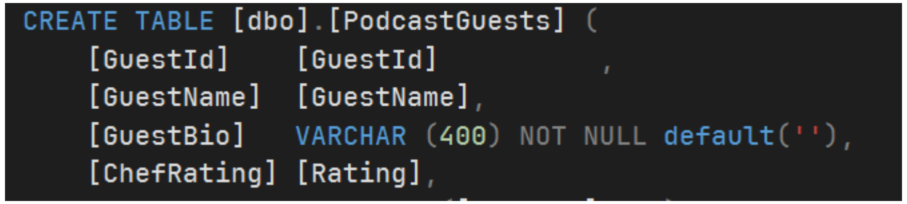
```
ALTER TABLE [dbo].[EpisodesPublished]
    ADD [EncodingType] VARCHAR (20) DEFAULT ('mp3') NOT NULL;
```

1. If the column is at the end a simple add column alter statement is generated

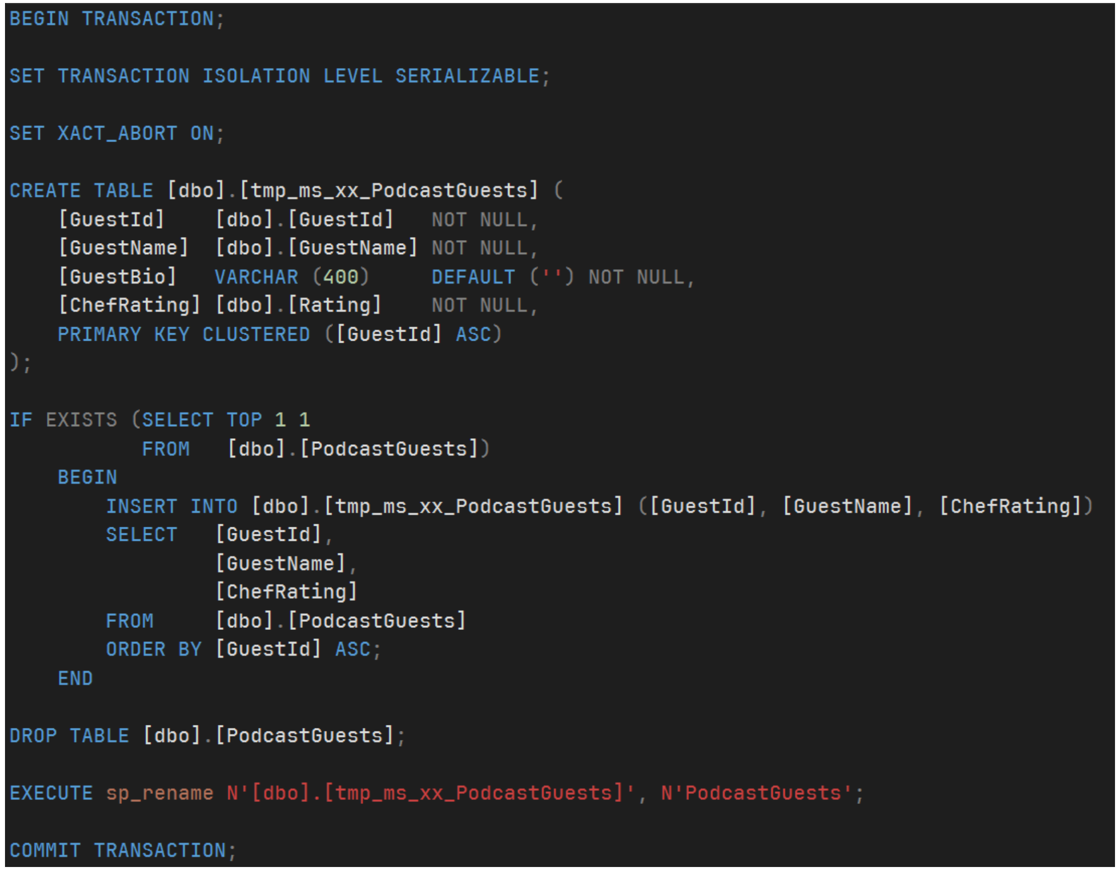
Change Sample – New column



```
CREATE TABLE [dbo].[PodcastGuests] (
    [GuestId]     [GuestId]             ,
    [GuestName]  [GuestName],
    [ChefRating] [Rating],
    PRIMARY KEY CLUSTERED ([GuestId] ASC),
```



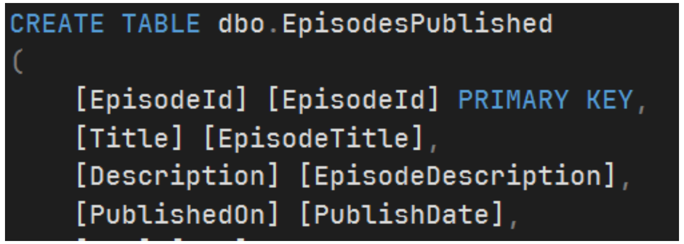
```
CREATE TABLE [dbo].[PodcastGuests] (
    [GuestId]     [GuestId]             ,
    [GuestName]  [GuestName],
    [GuestBio]   VARCHAR (400) NOT NULL default(''),
    [ChefRating] [Rating],
```



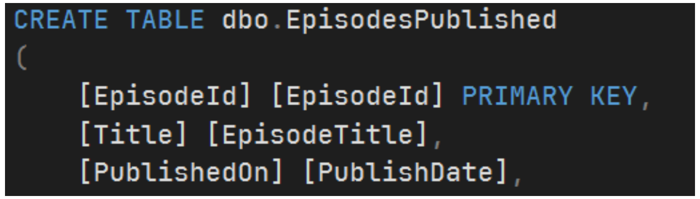
```
BEGIN TRANSACTION;

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

SET XACT_ABORT ON;

CREATE TABLE [dbo].[tmp_ms_xx_PodcastGuests] (
    [GuestId]    [dbo].[GuestId]   NOT NULL,
    [GuestName]  [dbo].[GuestName] NOT NULL,
    [GuestBio]   VARCHAR (400)     DEFAULT ('') NOT NULL,
    [ChefRating] [dbo].[Rating]    NOT NULL,
    PRIMARY KEY CLUSTERED ([GuestId] ASC)
);

IF EXISTS (SELECT TOP 1 1
           FROM   [dbo].[PodcastGuests])
    BEGIN
        INSERT INTO [dbo].[tmp_ms_xx_PodcastGuests] ([GuestId], [GuestName], [ChefRating])
        SELECT   [GuestId],
                 [GuestName],
                 [ChefRating]
        FROM     [dbo].[PodcastGuests]
        ORDER BY [GuestId] ASC;
    END

DROP TABLE [dbo].[PodcastGuests];

EXECUTE sp_rename N'[dbo].[tmp_ms_xx_PodcastGuests]', N'PodcastGuests';

COMMIT TRANSACTION;
```

1. New transaction
2. New temporary table with new column
3. Data from table gets copied to the temp table
4. Old table gets dropped
5. Temp table renamed to the original table name

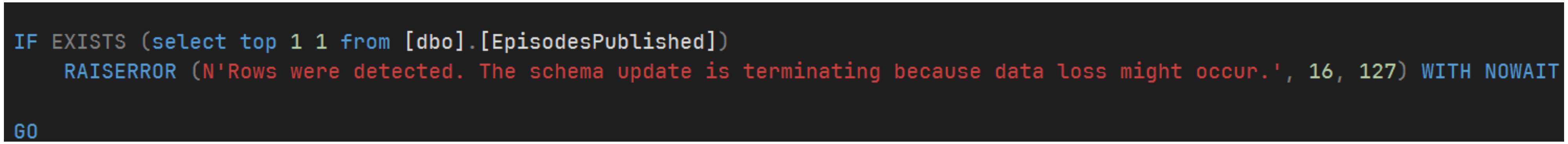
Change Sample – Drop column



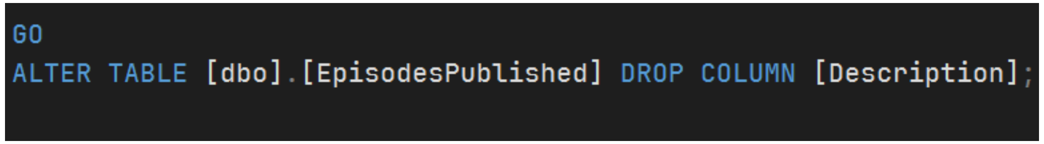
```
CREATE TABLE dbo.EpisodesPublished
(
    [EpisodeId] [EpisodeId] PRIMARY KEY,
    [Title] [EpisodeTitle],
    [Description] [EpisodeDescription],
    [PublishedOn] [PublishDate],
```



```
CREATE TABLE dbo.EpisodesPublished
(
    [EpisodeId] [EpisodeId] PRIMARY KEY,
    [Title] [EpisodeTitle],
    [PublishedOn] [PublishDate],
```



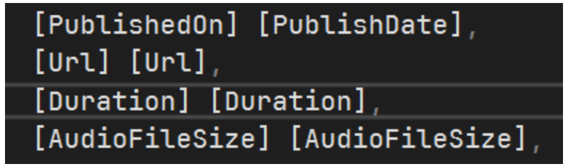
```
IF EXISTS (select top 1 1 from [dbo].[EpisodesPublished])
    RAISERROR (N'Rows were detected. The schema update is terminating because data loss might occur.', 16, 127) WITH NOWAIT

GO
```



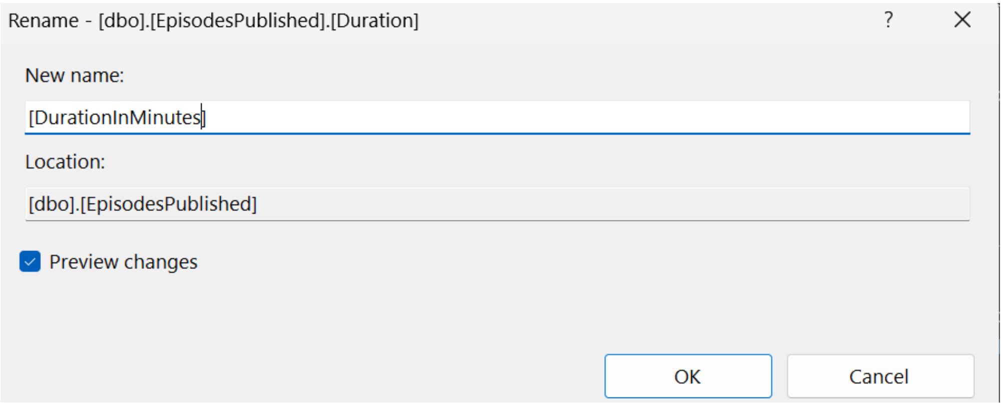
```
GO
ALTER TABLE [dbo].[EpisodesPublished] DROP COLUMN [Description];
```

1. A query is ran to determine if there is data in the table.
2. If there is no data, the script proceeds with dropping the column.

Change Sample – Rename column



```
[PublishedOn] [PublishDate],
[Url] [Url],
[Duration] [Duration],
[AudioFileSize] [AudioFileSize],
```

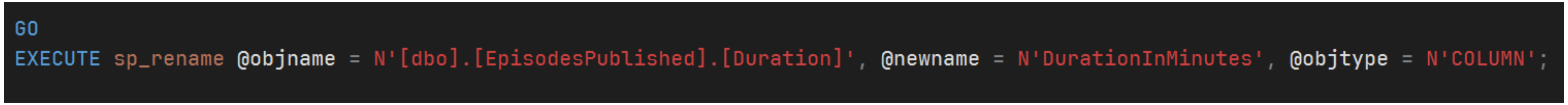


Rename - [dbo].[EpisodesPublished].[Duration]

New name: [DurationInMinutes]

Location: [dbo].[EpisodesPublished]

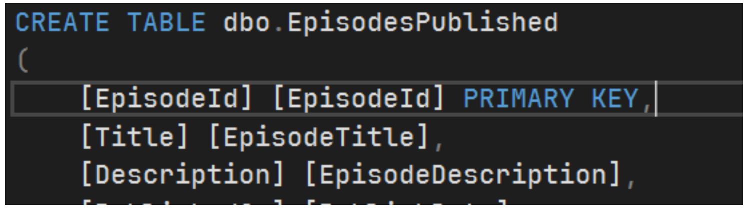
☑ Preview changes



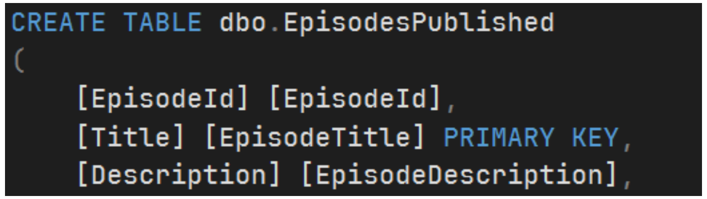
```
GO
EXECUTE sp_rename @objname = N'[dbo].[EpisodesPublished].[Duration]', @newname = N'DurationInMinutes', @objtype = N'COLUMN';
```

1. A sp_rename command is generated
2. Any other alters caused by the rename refactoring (e.g.: views referencing this column) are also issued.

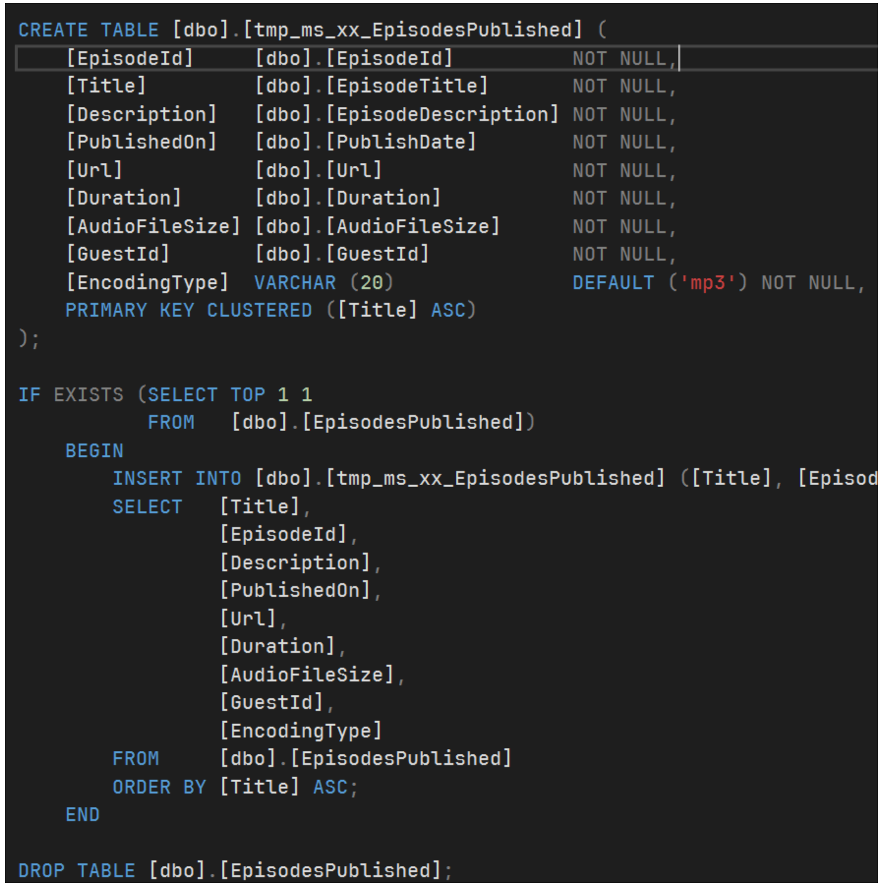
Change Sample – New primary key



```
CREATE TABLE dbo.EpisodesPublished
(
    [EpisodeId] [EpisodeId] PRIMARY KEY,
    [Title] [EpisodeTitle],
    [Description] [EpisodeDescription],
```



```
CREATE TABLE dbo.EpisodesPublished
(
    [EpisodeId] [EpisodeId],
    [Title] [EpisodeTitle] PRIMARY KEY,
    [Description] [EpisodeDescription],
```



```
CREATE TABLE [dbo].[tmp_ms_xx_EpisodesPublished] (
    [EpisodeId]       [dbo].[EpisodeId]          NOT NULL,
    [Title]           [dbo].[EpisodeTitle]       NOT NULL,
    [Description]     [dbo].[EpisodeDescription] NOT NULL,
    [PublishedOn]     [dbo].[PublishDate]        NOT NULL,
    [Url]             [dbo].[Url]                NOT NULL,
    [Duration]        [dbo].[Duration]           NOT NULL,
    [AudioFileSize]   [dbo].[AudioFileSize]      NOT NULL,
    [GuestId]         [dbo].[GuestId]            NOT NULL,
    [EncodingType]    VARCHAR (20)               DEFAULT ('mp3') NOT NULL,
    PRIMARY KEY CLUSTERED ([Title] ASC)
);

IF EXISTS (SELECT TOP 1 1
           FROM   [dbo].[EpisodesPublished])
    BEGIN
        INSERT INTO [dbo].[tmp_ms_xx_EpisodesPublished] ([Title], [Episod
        SELECT   [Title],
                 [EpisodeId],
                 [Description],
                 [PublishedOn],
                 [Url],
                 [Duration],
                 [AudioFileSize],
                 [GuestId],
                 [EncodingType]
        FROM     [dbo].[EpisodesPublished]
        ORDER BY [Title] ASC;
    END

DROP TABLE [dbo].[EpisodesPublished];
```

1. Full table re-creation with a data copy within a transaction

Initial seed data

- Many strategies for data that needs to be added to the db on deployment:
  - Post-deployment script
  - Script data as inserts/merges (if the data is not sensitive)
    - Or as a bulk data load (if the deployment identity has rights to do it)
  - Use a .NET tool to read a file from your environment and call that tool on your deployment pipeline.

What about CI/CD?

How can I deploy the changes without using a “right-click” publish?

Command line publishing (CI/CD tool friendly)

```
msbuild "nameofsqlproject.sqlproj" /p:Configuration=Release
```

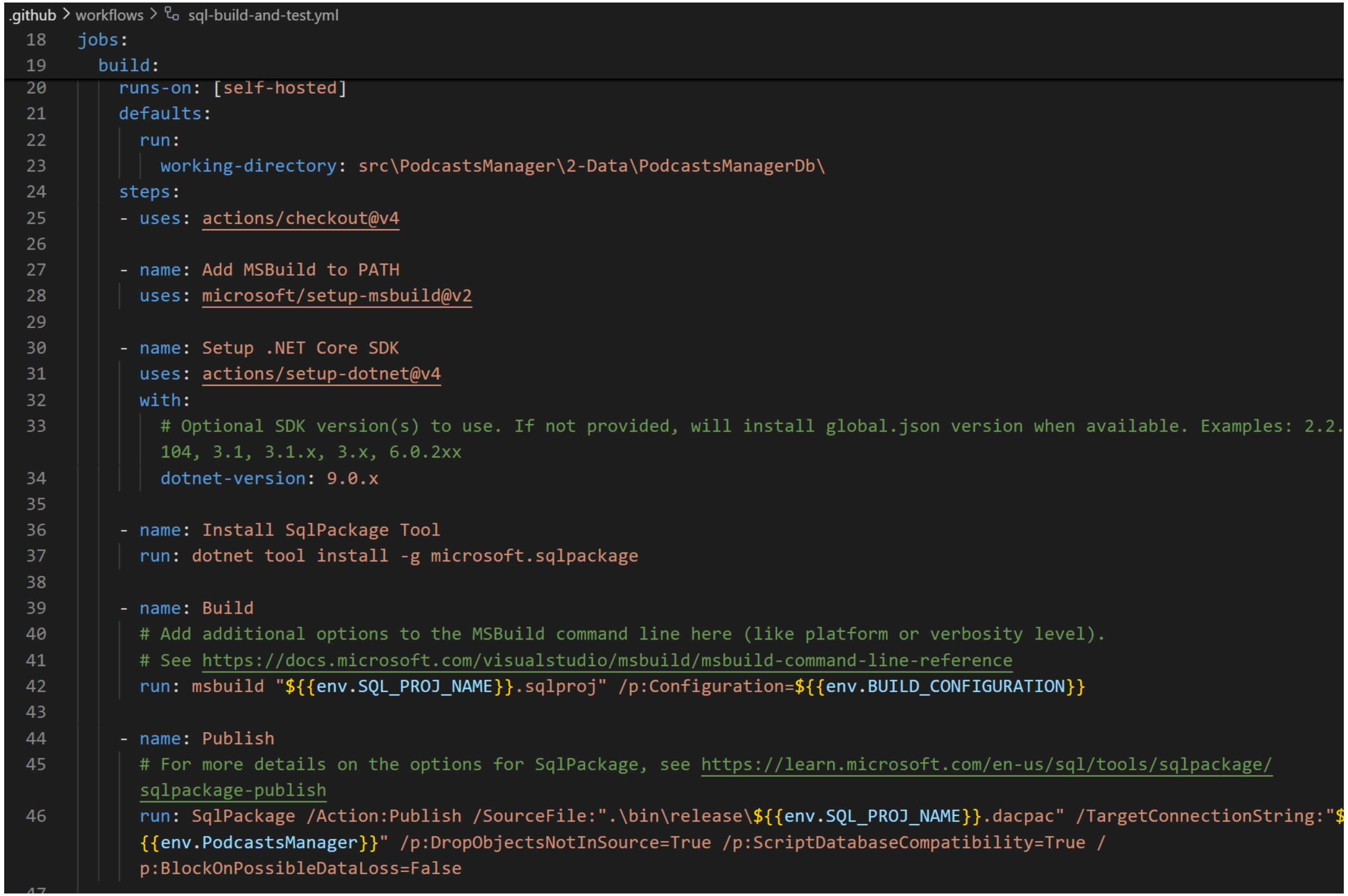
```
dotnet tool install -g microsoft.sqlpackage
```

```
SqlPackage /Action:Publish /SourceFile:".\bin\release\nameofsqlproject.dacpac" /TargetConnectionString:"ConnectionString
to the db you want to publish to” /p:ScriptDatabaseCompatibility=true
```

For example (PodcastsManager/src/PodcastsManager/2-Data/PodcastsManagerDb/publish-to-local-sqlexpress.bat at main · TheJuanitoLearnsShow/PodcastsManager):

```
msbuild "PodcastsManagerDb.sqlproj" /p:Configuration=Release
dotnet tool install -g microsoft.sqlpackage
SqlPackage /Action:Publish /SourceFile:".\bin\release\PodcastsManagerDb.dacpac" /TargetConnectionString:"Data
Source=.\sqlExpress;Database=PodcastsManager;Integrated Security=True;" /p:DropObjectsNotInSource=True
/p:ScriptDatabaseCompatibility=True /p:BlockOnPossibleDataLoss=False
```

CI/CD tools sample - deployment



.github > workflows > sql-build-and-test.yml

```yaml
jobs:
  build:
    runs-on: [self-hosted]
    defaults:
      run:
        working-directory: src\PodcastsManager\2-Data\PodcastsManagerDb\
    steps:
    - uses: actions/checkout@v4

    - name: Add MSBuild to PATH
      uses: microsoft/setup-msbuild@v2

    - name: Setup .NET Core SDK
      uses: actions/setup-dotnet@v4
      with:
        # Optional SDK version(s) to use. If not provided, will install global.json version when available. Examples: 2.2.
        104, 3.1, 3.1.x, 3.x, 6.0.2xx
        dotnet-version: 9.0.x

    - name: Install SqlPackage Tool
      run: dotnet tool install -g microsoft.sqlpackage

    - name: Build
      # Add additional options to the MSBuild command line here (like platform or verbosity level).
      # See https://docs.microsoft.com/visualstudio/msbuild/msbuild-command-line-reference
      run: msbuild "${{env.SQL_PROJ_NAME}}.sqlproj" /p:Configuration=${{env.BUILD_CONFIGURATION}}

    - name: Publish
      # For more details on the options for SqlPackage, see https://learn.microsoft.com/en-us/sql/tools/sqlpackage/
      sqlpackage-publish
      run: SqlPackage /Action:Publish /SourceFile:".\bin\release\${{env.SQL_PROJ_NAME}}.dacpac" /TargetConnectionString:"$
      {{env.PodcastsManager}}" /p:DropObjectsNotInSource=True /p:ScriptDatabaseCompatibility=True /
      p:BlockOnPossibleDataLoss=False
```

What about automated tests?

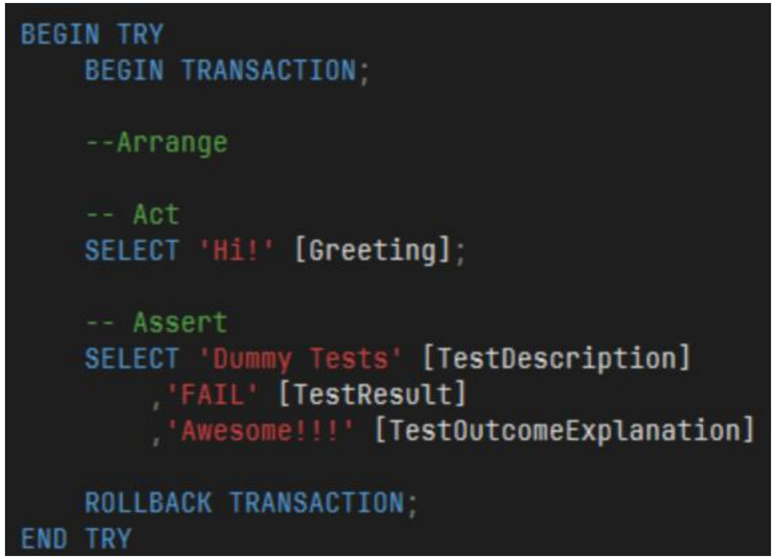
We have them in .NET, can we have them for our SQL projects?

Automated testing

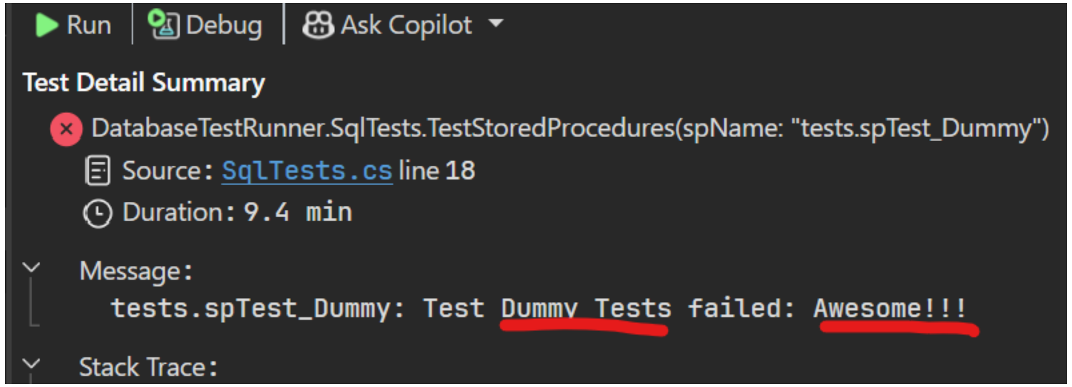
- Many ways to implement automated testing
  - Custom tests from .NET.
  - SQL-Specific Unit Test frameworks (e.g.: tUnit).
  - Custom, generic, convention-based automated testing using xUnit template.
- Demo of Reusable xUnit project
- CI/CD integration

Custom generic xUnit Project

- Sample in repo
  - https://github.com/TheJuanitoLearnsShow/PodcastsManager/tree/main/src/PodcastsManager/7-Tests/DatabaseTestRunner
- Uses a convention approach to discovering test stored procedures from the SQL db
- Each test stored procedure follows the arrange, act, assert pattern.
  - Uses transaction that are rolled back after the test
  - The results produced by the SP will be examined by the xUnit code to determine if the follow the PASS/FAIL convention
- The C# code is minimal and easily ported to another solution



```
BEGIN TRY
    BEGIN TRANSACTION;

    --Arrange

    -- Act
    SELECT 'Hi!' [Greeting];

    -- Assert
    SELECT 'Dummy Tests' [TestDescription]
        ,'FAIL' [TestResult]
        ,'Awesome!!!' [TestOutcomeExplanation]

    ROLLBACK TRANSACTION;
END TRY
```



Test Detail Summary

DatabaseTestRunner.SqlTests.TestStoredProcedures(spName: "tests.spTest_Dummy")

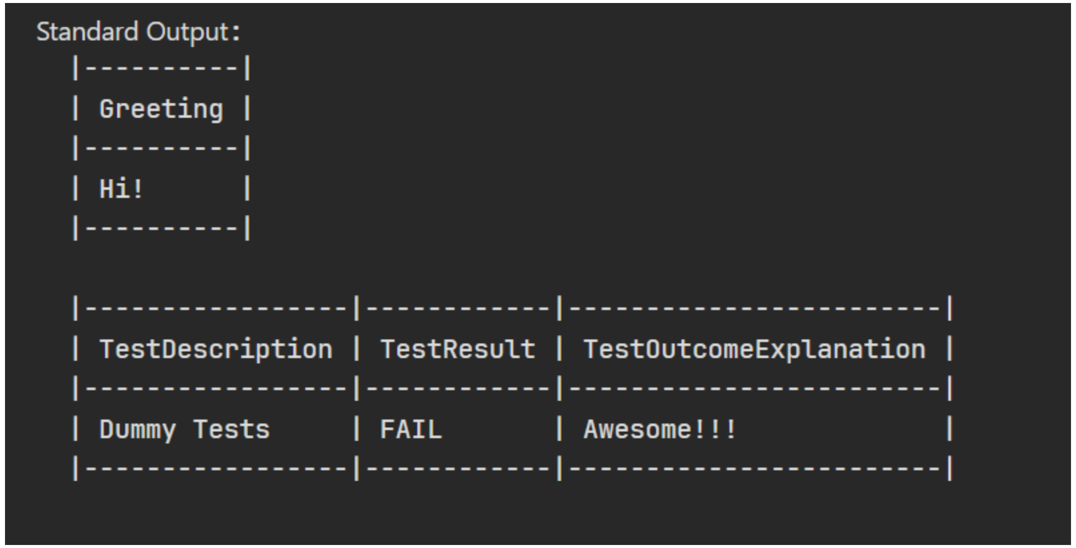
Source: SqlTests.cs line 18

Duration: 9.4 min

Message:

```
tests.spTest_Dummy: Test Dummy Tests failed: Awesome!!!
```

Stack Trace:



Standard Output:

```
|----------|
| Greeting |
|----------|
| Hi!      |
|----------|


|-----------------|------------|------------------------|
| TestDescription | TestResult | TestOutcomeExplanation |
|-----------------|------------|------------------------|
| Dummy Tests     | FAIL       | Awesome!!!             |
|-----------------|------------|------------------------|
```

Additional Resources

Formatter

Poor Man's T-Sql Formatter is a VS extension that allows you to format sql code in Visual Studio. It is my preferred formatter for T-SQL

https://marketplace.visualstudio.com/items?itemName=mick9956.PoorMansTSqlFormatter

Resources

- Get started with SQL database projects - SQL Server | Microsoft Learn (https://learn.microsoft.com/en-us/sql/tools/sql-database-projects/get-started)
- 9 Tips for Faster SQL Server Applications - Brent Ozar Unlimited® (https://www.brentozar.com/archive/2019/05/9-tips-for-faster-sql-server-applications)
- https://jptarqu.blogspot.com/2023/04/add-sql-server-database-projects-to-ci.html
- SqlPackage Publish - SQL Server | Microsoft Learn (https://learn.microsoft.com/en-us/sql/tools/sqlpackage/sqlpackage-publish)

Additional Resources

Resources

- sql-projects-devops-samples/Demos/2-ContinuousIntegration.md at main · Azure-Samples/sql-projects-devops-samples · GitHub (CI/CD with containers)
- .net conf 2024 talk: Next-gen SQL projects with Microsoft.Build.Sql

SQL Projects for the Win!!!

- Declarative way to define your databases.
- Source control friendly.
- Automated deployments
- Automated tests
- Model your databases as close to the business problem as possible
  - Your database is a service not just a repository of CRUD tables.

Questions, Questions, Questions???

- Best part of sessions like this one ☺

SQLSATURDAY

OCT 4TH 2025

Orlando, FL

SQLOrlando

Scan the QR code to fill out session evaluations

Visual Studio SQL Projects

- https://tinyurl.com/SqlProjCodeCamp
  - (sample code and slides)

LinkedIn https://www.linkedin.com/in/juantarquino

Juan Pablo Tarquino

Software Engineer at SouthState Bank