Demo & Step by Step Guide

Creating SSAS Tabular Data Models

By: Wylie Blanchard

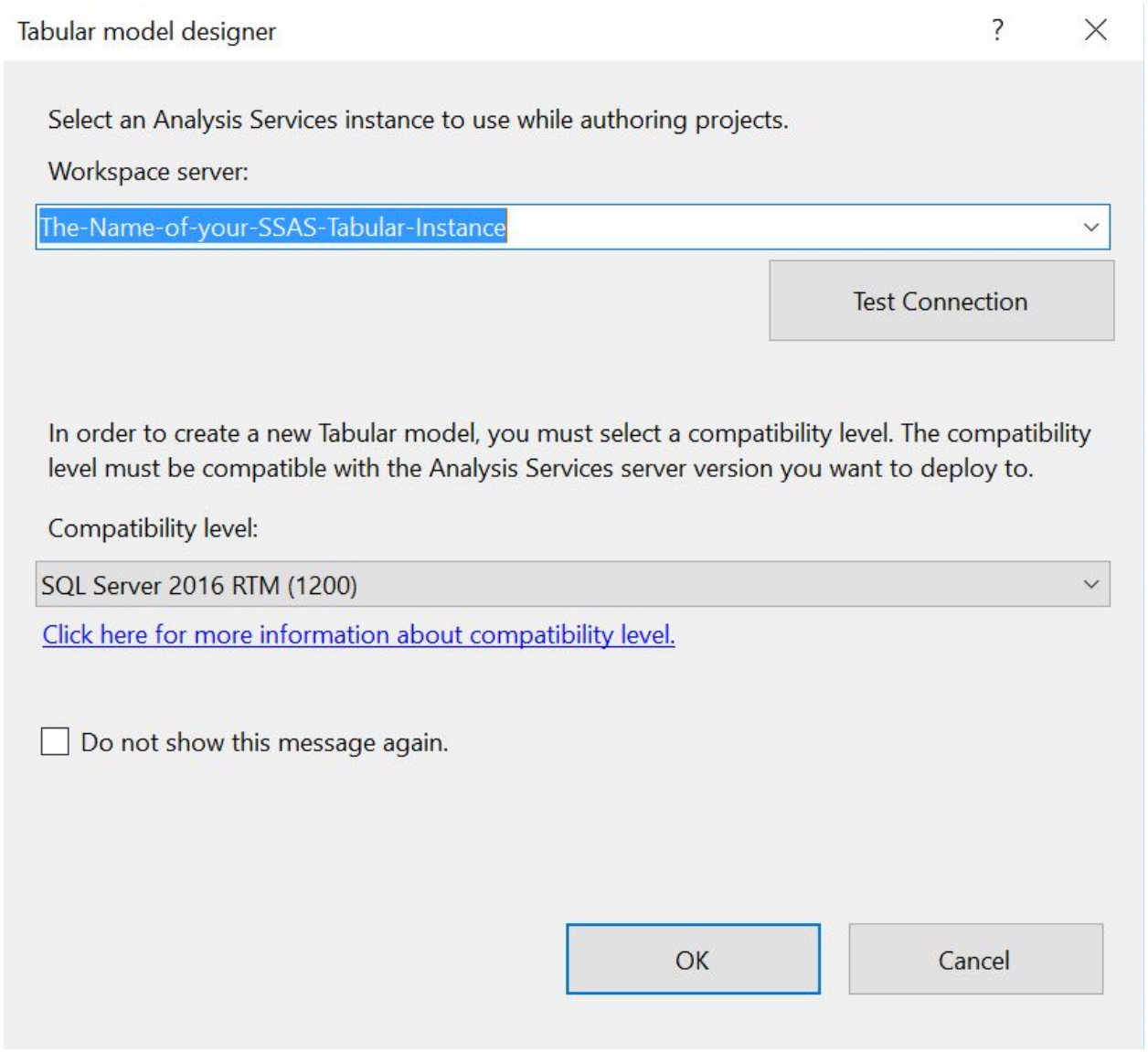
Table of Contents

Creating a Data Model (SSAS Tabular)

Create a New Tabular Model Project

To create a new tabular model project

1. In SQL Server Data Tools, on the **File** menu, click **New**, and then click **Project**.
2. In the **New Project** dialog box, under **Installed**, click **Business Intelligence**, then click **Analysis Services**, and then click **Analysis Services Tabular Project**.
3. In **Name**, type a unique name for your new data model, then specify a location for the project files.. For our demo we'll use the name **Healthcare Claims Tabular Model.**
4. Click **OK**.
5. In the **Tabular model designer** dialog box, in **Workspace server**, type **<The-Name-of-your-SSAS-Tabular-Instance>** which identifies the SQL Server 2016 Analysis Services instance where you have server development permissions, and then click **Test Connection**.
    a. Note: While you are creating your project the the workspace server instance you specify will host a Tabular model database with the same name as your project.
6. In **Compatibility level**, verify **SQL Server 2016 (1200)** is selected, and then click **OK**.



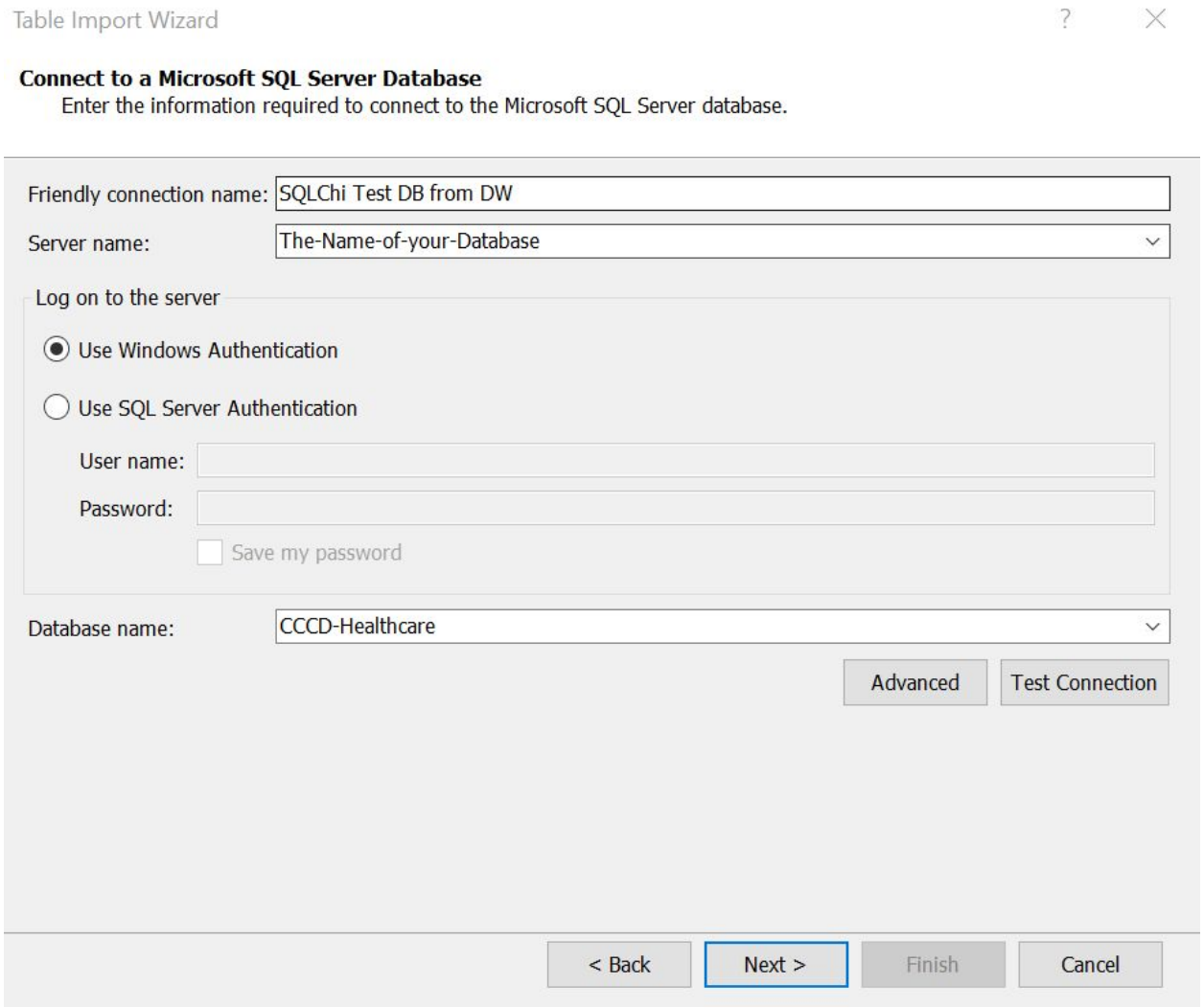
Create a Connection and Import Data

Connect to our datasource

Note: For our example we'll use the CCCD-Healthcare database as our datasource.

1. In SQL Server Data Tools, click on the **Model** menu, and then click **Import from Data Source**.
2. In the **Table Import Wizard**, under **Relational Databases**, click **Microsoft SQL Server**, and then click **Next**.
3. In the **Connect to a Microsoft SQL Server Database** page, in **Friendly Connection Name**, type **SQLChi Test DB from DW**.

4. In **Server name**, type <**The-Name-of-your-Database>** (or the name of the server containing the database that you are connecting to).
    a. Note: Try to always type the name of the server that you are connecting to. Refrain from using the drop down list option as the application will attempt to search for every instance of sql server available on your network which could take a significant amount of time.
5. In the **Database name** field, click the down arrow and select **CCCD-Healthcare** (or the name of the database that you are importing data from), and then click **Next**.

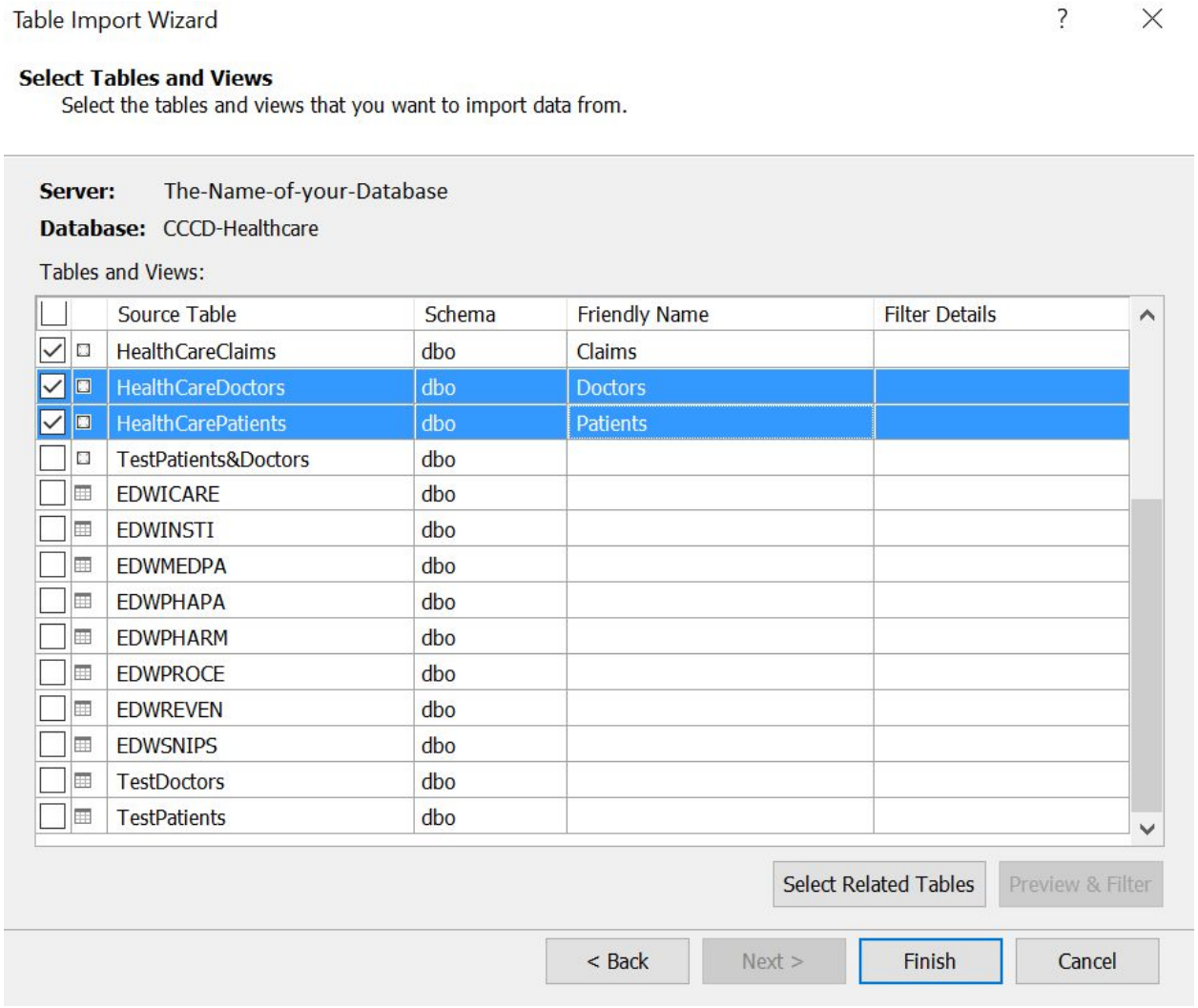


6. In the **Impersonation Information** page, you need to specify the credentials Analysis Services will use to connect to the data source when importing and processing data (contact your Database Administrator if instruction are needed for this step).
7. In the **Choose How to Import the Data** page, verify **Select from a list of tables and views to choose the data to import** is selected. You want to select from a list of tables and views, so click **Next** to display a list of all the source tables in the source database.
8. In the **Select Tables and Views** page, select the checkbox for the following views:

a. **HealthCareClaims**
b. **HealthCareDoctors**
c. **HealthCarePatients**

Note: These views utilize the “dm” schema.

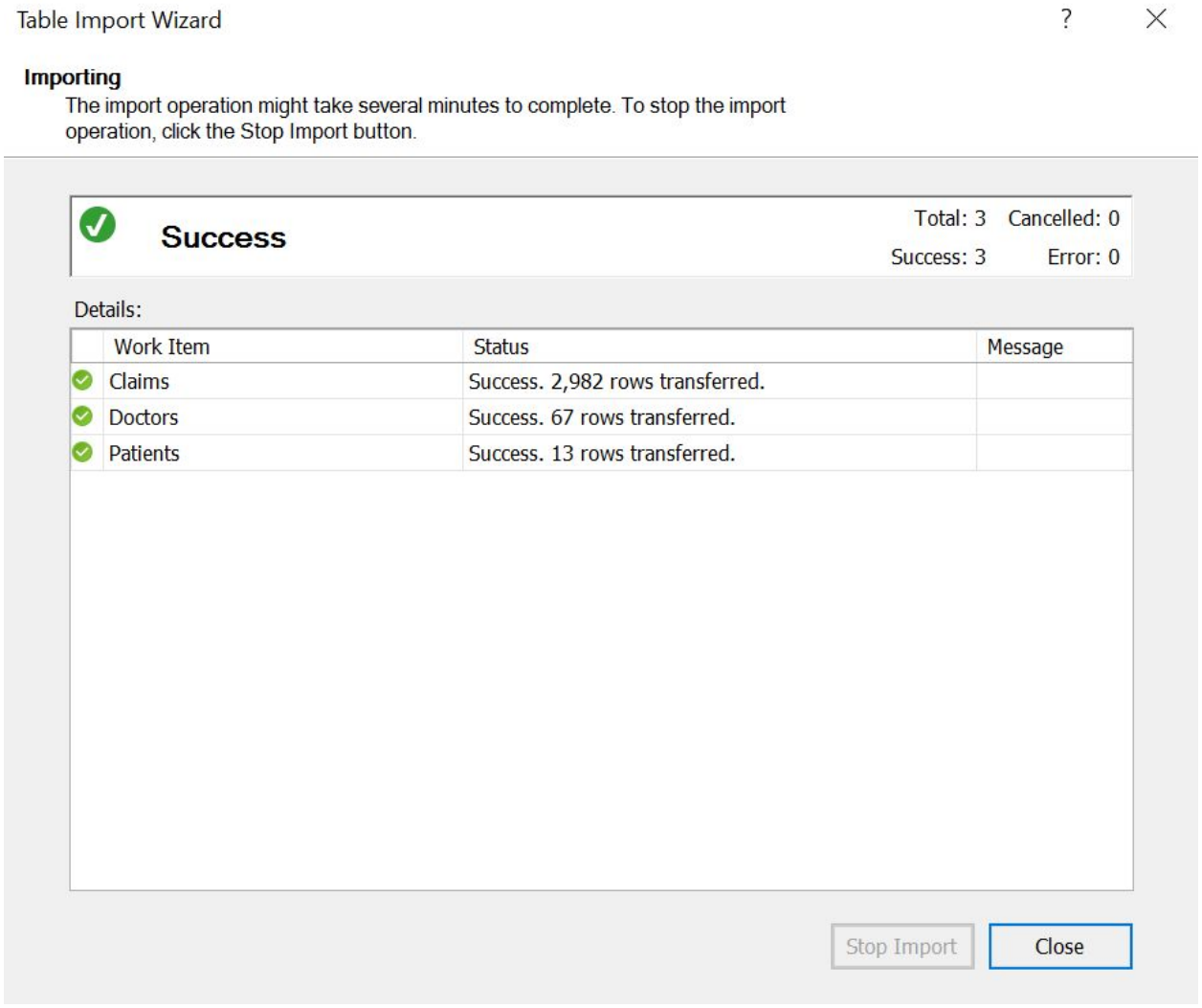
9. We want to give the tables in the model more easily understood names.
    a. Click on the cell in the **Friendly Name** column for **HealthCareClaims**. Rename the table to **Claims**.
    b. Click on the cell in the **Friendly Name** column for **HealthCareDoctors**. Rename the table to **Doctors**.
    c. Click on the cell in the **Friendly Name** column for **HealthCarePatients**. Rename the table to **Patients**.



Note: To remove specific columns from the model Select the table and then click the “Preview and Filter” button. In the Preview and Filter window you can specify whether there are columns that you do not want to include in your model.

Note: Only use Import From Data Source once per connection (database). After you've imported tables/views from your data source and need to use it again, click on the Model menu, and then click Existing Connections. Here you'll find the connections that you previously created available for reuse.

10. Review your selections. If everything looks OK, click **Finish**.
11. While importing the data, the wizard displays how many rows have been fetched. When all the data has been imported, a message indicating success is displayed.



12. Click **Close**.
13. The wizard closes and the model designer is visible. Each table has been added as a new tab in the model designer.

Save the Model Project

It is important to frequently save your model project.

To save the model project

- In SQL Server Data Tools, click on the **File** menu, and then click **Save All**.

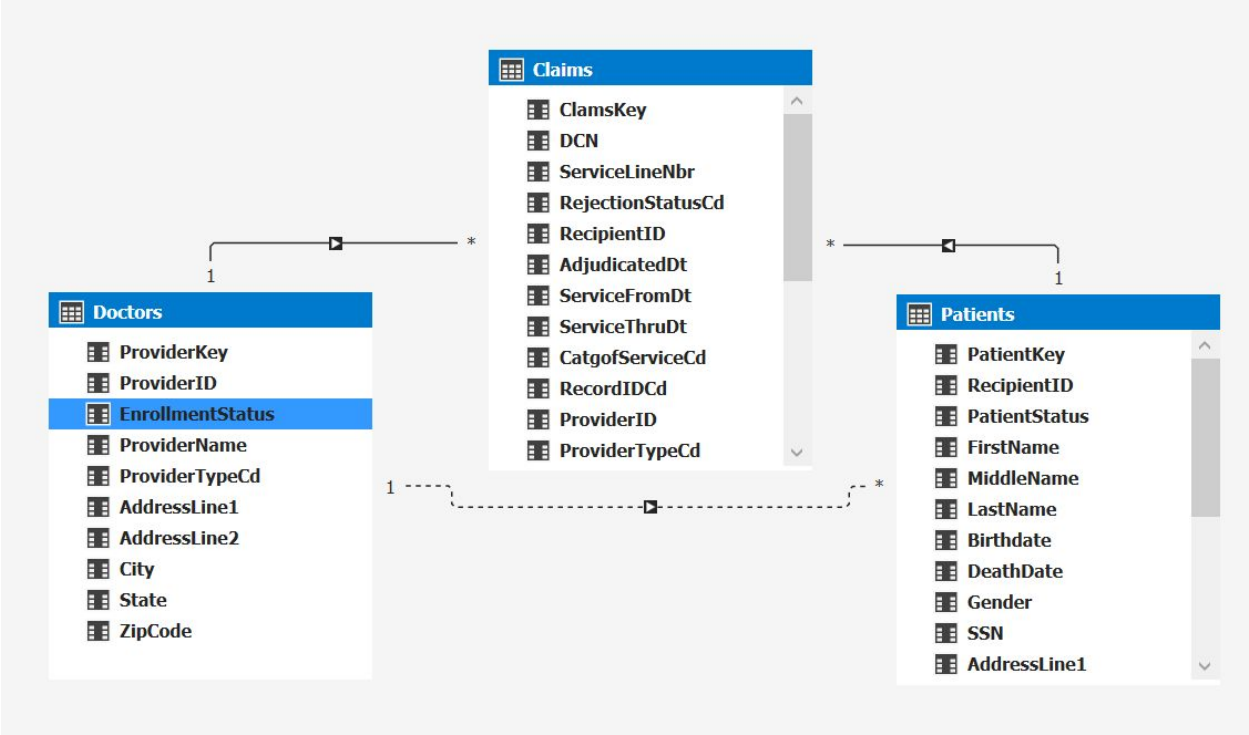
Review Existing Relationships and Add New Relationships

A relationship is a connection between two tables that establishes how the data in those tables should be correlated.

When you imported data by using the Table Import Wizard, you imported three tables from the CCCD-Healthcare database. Generally, if you import data from a relational source, existing relationships are automatically imported together with the data. However, before you proceed with authoring your model you should verify those relationships between tables were created properly. For this tutorial, you will also add two new relationships.

To add new relationships between tables

1. In SQL Server Data Tools, click on the **Model** menu, then point to **Model View**, and then click **Diagram View**.
2. Using the **Claims** table, click and hold on the **ProviderID** column, then drag the cursor to the **ProviderID** column in the **Doctors** table, and then release.
    a. A solid line appears showing you have created an active relationship between the **ProviderID** column in the **Claims** table and the **ProviderID** column in the **Doctors** table.
3. Using the **Patients** table, click and hold on the **RecipientID** column, then drag the cursor to the **RecipientID** column in the **Claims** table, and then release.
    a. A solid line appears showing you have created an active relationship between the **RecipientID** column in the **Patients** table and the **RecipientID** column in the **Claims** table.
4. In the **Patients** table, click and hold on the **PrimaryProviderID** column, then drag the cursor to the **ProviderID** column in the **Doctors** table, and then release.
    a. A dotted line appears showing you have created an inactive relationship between the **ProviderID** column in the **Patients** table and the **ProviderID** column in the **Doctors** table. You can have multiple relationships between tables, but only one relationship can be active at a time.



To review existing relationships

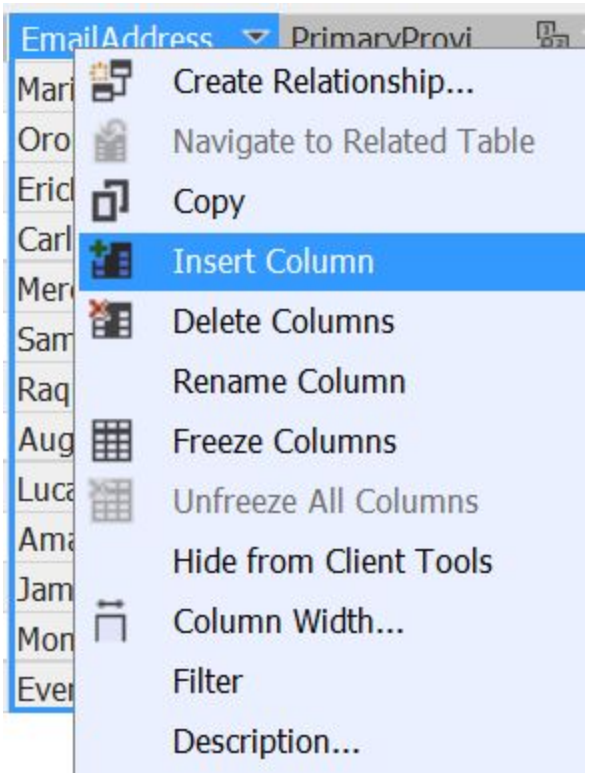
- Option 1: In SQL Server Data Tools, click on the **Table** menu, then click **Manage Relationships**.
    - The Manage Relationships dialog box shows the relationships that were either automatically created when you imported data or created manually.
    - Using this dialog box you can add/view/edit a relationship's active/inactive status, its relationship cardinality (many-to-one, one-to-one) and its filter direction.
- Options 2: In SQL Server Data Tools, click on the **Model** menu, then point to **Model View**, and then click **Diagram View**.
    a. The model designer now appears in Diagram View, a graphical format displaying all of the tables you imported with lines between them. The lines between tables indicate the relationships that were automatically created when you imported the data.
    b. Use the minimap controls in the lower-right corner of the model designer to adjust the view to include as many of the tables as possible. You can also click and drag tables to different locations, bringing tables closer together, or putting them in a particular order. Moving tables does not affect the relationships already between the tables. To view all of the columns in a particular table, click and drag on a table edge to expand or make it smaller.

Create a Calculated Column

A calculated column is a column created by a dax formula that created a value by calculating data that already exists in the model.

Create a Month Calendar calculated column in the Patients table

1. In SQL Server Data Tools, click the **Model** menu, then point to **Model View**, and then click **Data View**.
    a. Calculated columns can only be created by using the model designer in Data View.
2. In the model designer, click the **Patients** table (tab).
3. Right-click the **EmailAddress** column header, and then click **Insert Column**.

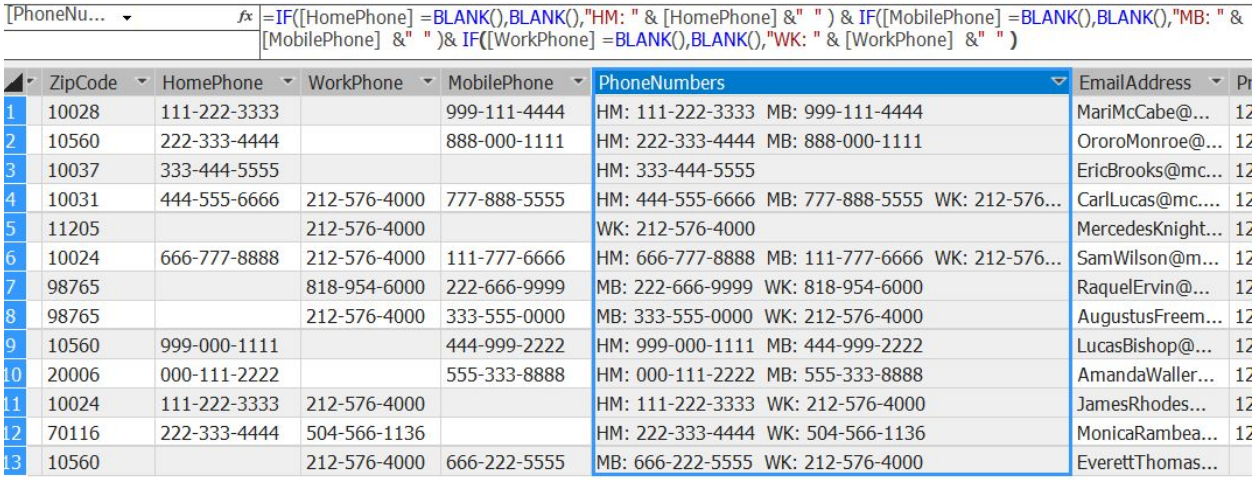


4. A new column named **Calculated Column 1** is inserted to the left of the **EmailAddress** column.
5. In the formula bar above the table, type the following formula. AutoComplete helps you type the fully qualified names of columns and tables, and lists the functions that are available.
    a. **=IF([HomePhone] =BLANK(),BLANK(),"HM: " & [HomePhone] &" ") & IF([MobilePhone] =BLANK(),BLANK(),"MB: " & [MobilePhone] &" ")& IF([WorkPhone] =BLANK(),BLANK(),"WK: " & [WorkPhone] &" ")**
    b. When you have finished building the formula, press ENTER.

c. Values are then populated for all the rows in the calculated column. If you scroll down through the table, you will see that rows can have different values for this column, based on the data that is in each row.

6. Rename this column to **PhoneNumbers**.

The PhoneNumbers calculated column provides a sortable name for all Patient Phone Numbers.



	ZipCode	HomePhone	WorkPhone	MobilePhone	PhoneNumbers	EmailAddress	Pr
1	10028	111-222-3333		999-111-4444	HM: 111-222-3333 MB: 999-111-4444	MariMcCabe@...	12
2	10560	222-333-4444		888-000-1111	HM: 222-333-4444 MB: 888-000-1111	OrororMonroe@...	12
3	10037	333-444-5555			HM: 333-444-5555	EricBrooks@mc...	12
4	10031	444-555-6666	212-576-4000	777-888-5555	HM: 444-555-6666 MB: 777-888-5555 WK: 212-576...	CarlLucas@mc....	12
5	11205		212-576-4000		WK: 212-576-4000	MercedesKnight...	12
6	10024	666-777-8888	212-576-4000	111-777-6666	HM: 666-777-8888 MB: 111-777-6666 WK: 212-576...	SamWilson@m...	12
7	98765		818-954-6000	222-666-9999	MB: 222-666-9999 WK: 818-954-6000	RaquelErvin@...	12
8	98765		212-576-4000	333-555-0000	MB: 333-555-0000 WK: 212-576-4000	AugustusFreem...	12
9	10560	999-000-1111		444-999-2222	HM: 999-000-1111 MB: 444-999-2222	LucasBishop@...	12
10	20006	000-111-2222		555-333-8888	HM: 000-111-2222 MB: 555-333-8888	AmandaWaller...	12
11	10024	111-222-3333	212-576-4000		HM: 111-222-3333 WK: 212-576-4000	JamesRhodes...	12
12	70116	222-333-4444	504-566-1136		HM: 222-333-4444 WK: 504-566-1136	MonicaRambea...	12
13	10560		212-576-4000	666-222-5555	MB: 666-222-5555 WK: 212-576-4000	EverettThomas...	

Create additional calculated columns in the indicated table(s):

Table	Calculated Column name	Formula
Claims	Year	=Year([AdjudicatedDt])
Claims	Month	=Month([AdjudicatedDt])
Claims	Day	=DAY([AdjudicatedDt])
Patients	FullAddress	=[AddressLine1]&", "&[City]&", "&[State]&" "&[ZipCode]

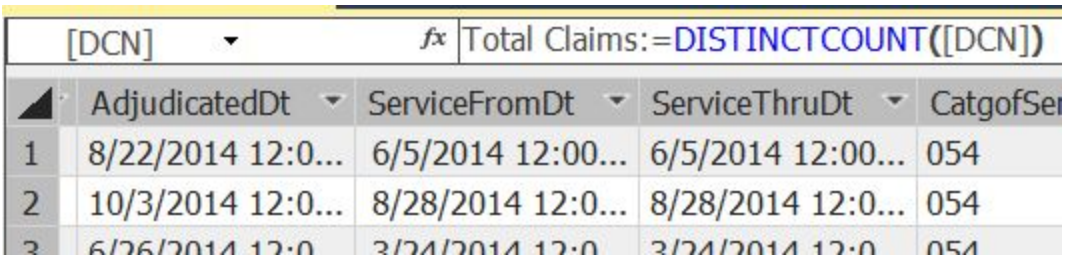
Create a Measure

Similar to the calculated columns you created in the previous lesson, a measure is essentially a calculation created using a DAX formula. However, unlike calculated columns, measures are evaluated based on a user selected *filter*; for example, a particular column or slicer added to the Row Labels field in a PivotTable. A value for each cell in the filter is then calculated by the

applied measure. Measures are powerful, flexible calculations that you will want to include in almost all tabular models, to perform dynamic calculations on numerical data.

To create a Daily Average Number of Claims measure in the Claims table

1. In the model designer, click the **Claims** table.
2. If an empty measure grid does not already appear beneath the table, click on the **Table** menu, and then click **Show Measure Grid**.
3. In the measure grid, click the top-left empty cell.
4. In the formula bar, above the table, type the following formula:
    a. **Total Claims:=DISTINCTCOUNT([DCN])**
    b. When you have finished building the formula, press ENTER.



Create additional measure(s):

You can copy and paste the content in the "Formula" column below

Measure name	**Table**	**Formula**
Daily Avg Claims	Claims	Daily Avg Claims:=[Total Claims]/DISTINCTCOUNT([Day])
Last Service Date	Claims	Last Service Date:= Max([ServiceThruDt])
Total Providers	Doctors	Total Providers:=DISTINCTCOUNT([ProviderID])
Total Dr Claims	Doctors	Total Dr Claims:=CALCULATE([Total Claims],Doctors[ProviderID])
Total Patients	Patients	Total Patients:=DISTINCTCOUNT([PatientProfileID])

.

Create Key Performance Indicators

Key Performance Indicators (KPIs) are used to gauge performance of a value, defined by a *Base* measure, against a *Target* value, also defined by a measure or by an absolute value. In reporting client applications, KPIs can provide business professionals a quick and easy way to understand a summary of business success or to identify trends.

To create a Daily Average Claims Performance KPI

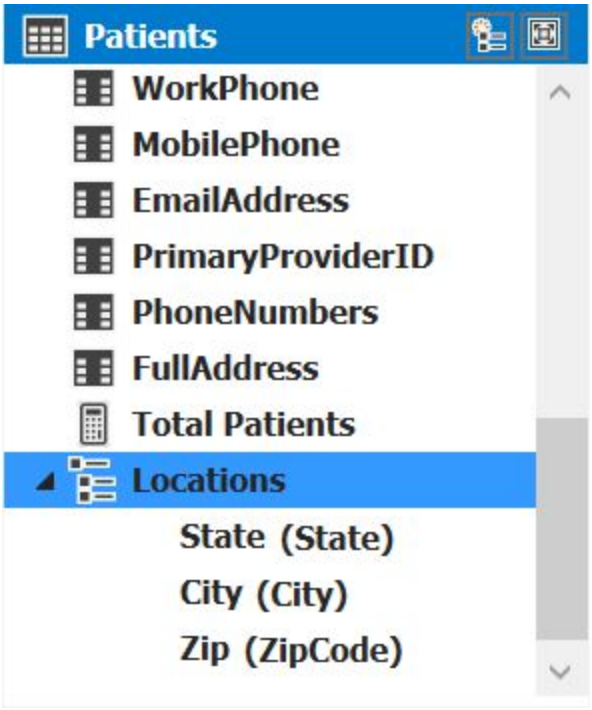
1. In the model designer, click the **Claims** table (tab).
2. In the measure grid, right-click the **Daily Avg Claims** measure, and then click **Create KPI**.
    a. The **Key Performance Indicator** dialog box opens.
3. In the **Key Performance Indicator (KPI)** dialog box, in **Target**, select the **Absolute Value** option.
4. In the **Absolute Value** field, type **125**, and then press ENTER.
5. In the left (low) slider field, type **50**, and then in the right (high) slider field, type **100**.
6. In **Select Icon Style**, select the first (red), (yellow), (green) icon type.
7. Click **OK** to complete the KPI.

Create a Hierarchy

Hierarchies are metadata that define relationships between two or more columns in a table, thus defining the relative position of column to another column.

To create a hierarchy in the Patients table

1. In the model designer, click on the **Model** menu, then point to **Model View**, and then click **Diagram View**.
2. Right-click the **Patients** table, and then click **Create Hierarchy**. A new hierarchy appears at the bottom of the table window.
3. In the hierarchy name, rename the hierarchy by typing **Locations**, and then press ENTER.
4. In the **Patients** table, right click the **State** column, in the pop up box go to **Add to Hierarchy** and then click **Locations**.
5. In the **Patients** table, click the **City** column, then drag it to the **Locations** hierarchy, releasing it below **State**.
6. In the **Patients** table, click the **ZipCode** column, then drag it to the **Locations** hierarchy, releasing it below **City**.
7. In the **Patients** table, click the **Address** column, then drag it to the **Locations** hierarchy, releasing it below **ZipCode** .
8. In the **Locations** hierarchy, right-click the **ZipCode** column, then click **Rename**, and then type **Zip**.
9. Note: Ensure that the columns in the hierarchy are in logical order. Typically the column with the least uniqueness (based on business) should be near the top of the list and the column with the highest uniqueness (based on the business) should be near the bottom of the list.



Create additional Hierarchies

In the **Patients** table create a hierarchy named **ContactInfo**. Add the following columns to the hierarchy in the order listed

1. FullAddress
2. PhoneNumbers
3. EmailAddress

In the **Doctors** table create a hierarchy named **OfficeLocations**. Add the following columns to the hierarchy in the order listed.

1. State
2. City
3. ZipCode (rename this object to “Zip”)

Deploy the Model

To configure the deployment properties

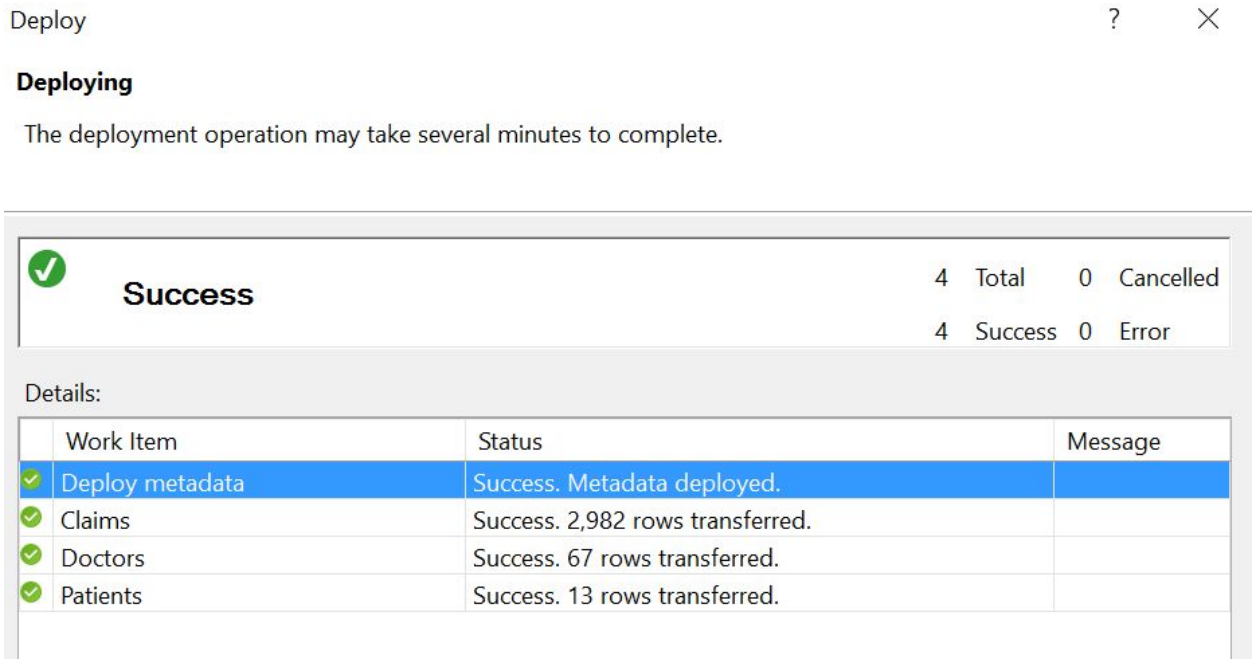
1. In SQL Server Data Tools, in **Solution Explorer**, right-click on the **Healthcare Claims Tabular Model** project, and then in the context menu, click **Properties**.
2. In the **Healthcare Claims Tabular Model Property Pages** dialog box, under **Deployment Server**, in the **Server** property, type the name of the target Analysis

Services instance running in Tabular mode. This will be the instance your model will be deployed to.

3. In the **Database** property, type **Healthcare Claims Tabular Model**.
4. In the **Model Name** property, type **SQL CHI Healthcare Claims Tabular Model**.
5. Verify your selections and then click **OK**.

To deploy the model

1. In SQL Server Data Tools, click the **Build** menu, and then click **Build Healthcare Claims Tabular Model**.
2. In SQL Server Data Tools, click the **Build** menu, and then click **Deploy Solution**.
3. The Deploy dialog box appears and displays the deployment status of the metadata as well as each table included in the model.



4. When deployment successfully completes, go ahead and click **Close**.

Conclusion

Congratulations!

You are finished authoring and deploying your first Analysis Services Tabular model. This tutorial has helped guide you through completing the most common tasks in creating a tabular model. Now that your Healthcare Claims Tabular Model is deployed, you can use SQL Server Management Studio to manage the model; create process scripts and a backup plan. Users can connect to the model using a reporting client application such as Microsoft Excel or Power View.

Next Steps

Quickly visualize your solution from an end user perspective

There are many tools that can share and enhance your Tabular data models:

- SSRS,
- MS Excel,
- PowerPivot,
- PowerBI,
- Tableau,
- etc...

With SQL Server Data Tools you can quickly view your tabular model in MS Excel

1. In SQL Server Data Tools, click the **Model** menu, and then click **Analyze in Excel**.
2. When the Analyze in Excel dialog box opens, click **OK**.
3. MS Excel Opens, creates a connection to your data model and creates a PivotTable in MS Excel. In the PivotTable Fields pane you'll see all of the objects (tables, columns, calculated columns, measures, KPI's and hierarchies,) that you created in your data model. You can then select and/or drag and drop objects to begin analyzing your data model.

Other SSAS Tabular Data Model Topics

Below you'll find a list of items/topics not mentioned in this presentation that are important to creating and managing SSAS Tabular Data Models.

- **Roles:** Creating and managing SSAS Tabular Roles - which define member permissions for a model.
- **Perspectives:** Creating and managing SSAS Tabular Perspectives - which defe viewable subsets of a model that provide focused, business-specific, or application-specific viewpoints of the model.
- **Partitions:** Creating and Managing SSAS Tabular Partitions - which divide a table into logical parts.
- **DAX:** Creating custom formulas and expressions with Data Analysis Expressions (DAX).

Sources

- Tabular Modeling
- Care Coordination Claims Data - Illinois.gov

- The database used in this demo is based on the test healthcare data set made public by I-HFS.

END