FLORIDA CODE
Collaborative on Operational Data for Educators

Are My Students Improving in Math?

Essential Question 24 in the Middle Grades

Developed by the Florida Collaborative on Operational Data for Educators
in Collaboration with the North East Florida Educational Consortium

Contents

Purpose

The purpose of this document is to help individuals and organizations deploy the Essential Question 24 project from the Ed-Fi Exchange. Please note that these instructions are for a first-time deployment of the views, model, security groups, and solution.

Additionally, the second section of this document is dedicated to the dynamic security for the model and visualizations.

The associated visualization combines data from multiple assessment to provide a comprehensive view of achievement in mathematics over time without the need to access multiple data platforms.

Tabular Model, Groups, and Visual Deployment

Prerequisites

- An Ed-Fi ODS (v 2.5) running via a cloud-deployed or on-premise ODS.
- Microsoft Azure Subscription
- Access as an Azure Global Administrator
- The latest version of SQL Server Management Studio[1]
    - We used v.17.9.1
- The latest version of SQL Server Data Tools[2]
    - We used v.15.9.3
- Power BI Desktop
- Power BI Pro License.
- Power BI Workspace[3]
- Service Account for your ODS.
- A database “PowerBI” to hold static files and a service account.
- *Optional: Microsoft Visual Studio*

Notes

Azure Analysis Services Tier – Tabular Models are hosted entirely in-memory, so it is important to select a tier for Azure Analysis Services that is powerful enough to hold your Tabular Model as well as having enough overhead for viewing reports and database processing.
On-premise Data Gateway – If using an on-premise ODS, an Azure Data Gateway service application must be installed on the network. The gateway will provide a secure access point for Azure Analysis Services to query data from the ODS. This must be done before deploying the Tabular Model.

Links – As of April 2020

[1] https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-_management-studio-ssms

[2] https://docs.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt

[3] https://www.powerbi.com

Create Power BI database

The database will hold two look up CSV files for the assessments. You may adjust or remove this option later.

a. Create a new Database on the same server as the ODS named “PowerBI.”
b. Create a service account to access this database.
c. Import two provided CSV files into Power BI database:

   FSA_LearningGains. - holds min and max scale scores for each Florida State Assessment Level.

   ObjectiveAssessmentIdentificationCode - holds Identification Codes for Florida State Assessment Objectives.

Install views and a function

a. Open SQL Server Management Studio and connect to your ODS server.
b. Create a bi schema in your ODS.
c. Open the provided SQL file. In the first line change [EdFi_Ods] name to your ODS name and execute the file.
d. Open each view and check if WHERE clause in some views will need to be adjusted based on your data.
e. Adjust filters in the following two views to have a matching [IdentificationCode] column.
   - Make sure one view has Points Possible only and the other Points Earned only.
   - Provided views assume that all records for Objective Assessments Points Earned and Points Possible are loaded in the same table.

     edfi.StudentAssessmentStudentObjectiveAssessmentScoreResult

   - We separated these records into two views for easy visualization:

     [bi].[eq24.StudentAssessmentStudentObjectiveAssessmentPointsPossible]

     [bi].[eq24.StudentAssessmentStudentObjectiveAssessmentScoreResult]

f. Adjust StudentAssessmentIdentifier and AssessmentIdentifier in four views:

   - [bi].[eq24.StudentAssessmentScoreResult]
   - [bi].[eq24.StudentAssessmentPerformanceLevel]
   - [bi].[eq24.StudentAssessmentStudentObjectiveAssessmentScoreResult]
   - [bi].[eq24.StudentAssessmentStudentObjectiveAssessmentPointsPossible]

   Select the case that matches your assessment identifier from the following:

Case 1 Unmatched Assessment Identifiers

In our use case, State assessments score results and performance levels are imported into one table StudentAssessmentScoreResult with different AssessmentReportingMethodTypeID. AssessmentIdentifier and StudentAssessmentIdentifier for the same student assessment do not match. All above views have these strings removed from AssessmentIdentifier and StudentAssessmentIdentifier to fix the issue: “_SS_”, “_AL_”. You will need to adjust these views based on your data to make sure that you have matching AssessmentIdentifier and StudentAssessmentIdentifier coming out of every view so the tabular model can connect the information for the same student assessment.

Case 2 Matching Assessment Identifiers

If in your use case AssessmentIdentifier and StudentAssessmentIdentifier match for the same Student Assessment, remove REPLACE function from every SELECT statement in all 4 views.

Note: You will need to go through every view and adjust WHERE clause as needed based on your Data.

Option A - Tabular Model Deployment

To deploy from XMLA file, use this option.

1. Open SQL Server Management Studio.
2. Navigate to File – Open the EQ 24.xmla file in the EQ 24 directory.
3. You will be prompted to connect to an Analysis Services server. Enter the connection string for your Analysis Services instance. You can find this on the resource panel in the Azure Portal, and it should be formatted like this:

   asazure://**eastus**.asazure.windows.net/**myservername**

4. Enter you email for UserID (the one you use in Azure Active Directory). You can also connect to Azure in the SQL SMS first and then open the file.
5. Click “Connect.” You will be prompted to log in with your Azure AD account.
6. We will need to modify a few things in the script before executing:

   - ODS Connection String
   - Power BI database connection string

   Search the document for “connectionString.” You will be brought to a placeholder connectionstring toward the top of the document. There are two connection strings to connect to two data bases.

7. In the connectionsStrings replace the placeholders (including the <> brackets) with the relevant information. This will include the *ODSSERVERADDRESS*, *ODSDATABASENAME*, *USERNAME*, and *PASSWORD*. Leave Power BI database name the same.

   [Make sure you use service account credentials set for your ODS and Power BI data bases]

8. Press F5 to execute the script. The tabular model will be deployed to the Azure Analysis Services server.

[If everything is entered correctly, the model will process in the next step. If the process fails, please repeat these steps.]

Process the Model

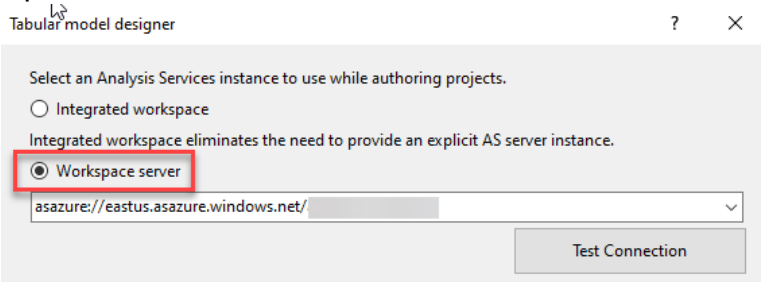
After deployment, Tabular Model will need to be processed. It will pull information into the Azure Analysis Services Tabular Model from the ODS and allow the individuals to view the Power BI reports.

1. In SQL Server Management Studio, using Azure connection, right click on the EQ 24 model and choose “Process Database.” You must be an administrator on the AAS server in order to make changes and process the data. You can modify this in the Azure Portal.
2. In the popup window, click on the drop-down menu to extend the “Mode.” Chose “Process Full.” Click “OK.” The Model should get processed. This means that the model will get populated with Data.

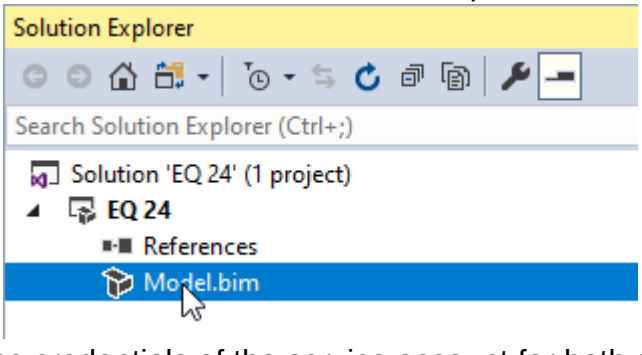
Option B - Deployment from Visual Studio

You do not need to complete this step if you successfully deployed and processed the model from the XMLA file.

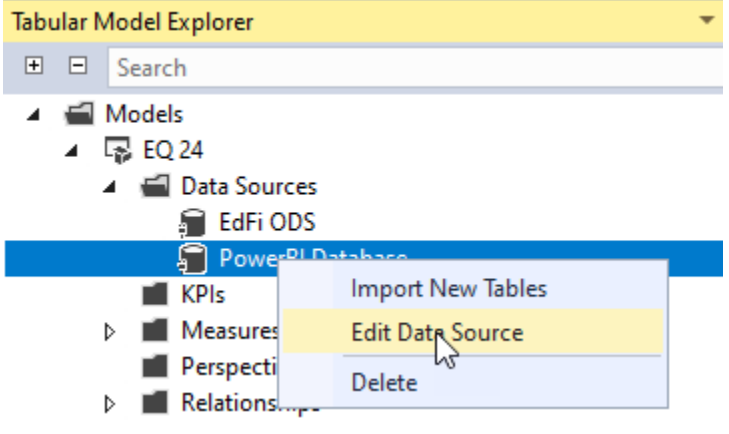
1. Open provided EQ 24 solution in Visual Studio. Solution is in the EQ 24 folder.
2. Tabular model designer popup will open and will prompt you to select a workspace. For this model we used Workspace server.

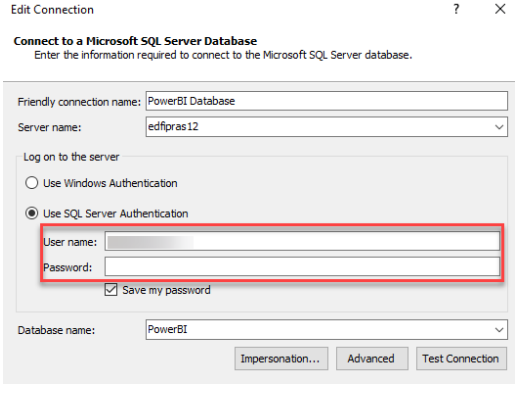


3. After solution is loaded, double click on Model.bim file to open the model.

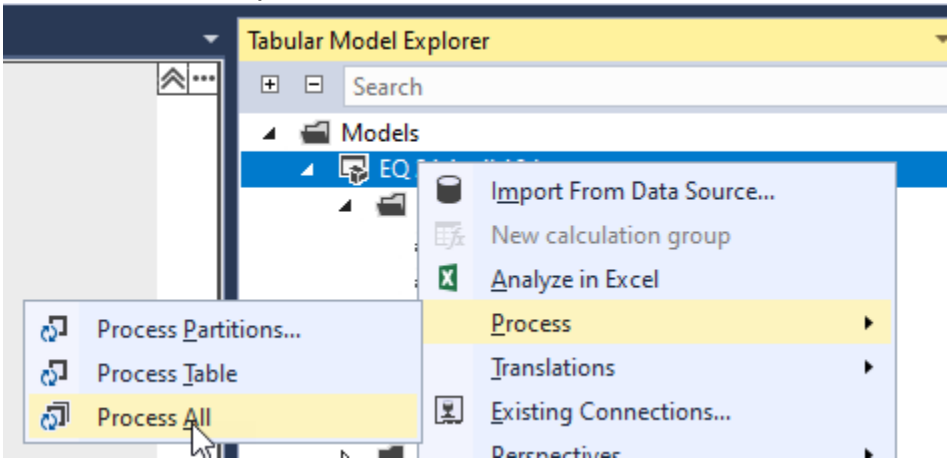


4. Edit Data Sources with the credentials of the service account for both databases: ODS and PowerBI.



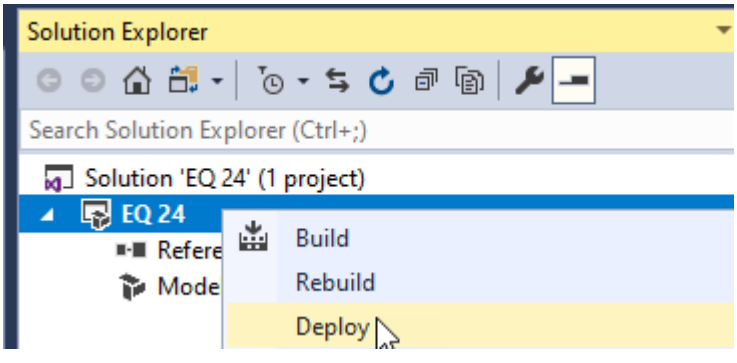


5. To load data into the model, process each table.



If you are not able to process all tables at the same time, process each table separately. Right click on each table, chose Table Properties, and process the table in a pop up by clicking OK.

- Once all tables processed without any errors, deploy the model.
  a. Right-click on the project in Solution Explorer.
  b. Select Deploy. This may prompt you for the Analysis Services connection string.



  c. You will then be prompted for your data source (ODS database) connection information. This validates the data model against the data source schema.

After Successfully Deploying from XMLA or Visual Studio

Continue these steps once the model has been successfully deployed and processed

Create Security Groups on Azure

a. Create three security groups with any name you would like:

    EQ 24 District Level
    EQ 24 School Level
    EQ 24 Teacher Level

b. Your groups eventually will be populated with Azure Active Directory users who will have access to appropriate level of the Model.

Add users to Azure Active Directory

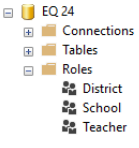
a. Refer to the link below to add users to your Azure Active Directory:

    https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/add-users-azure-active-directory

b. After users are added to Azure Active Directory, add them to the appropriate Azure security group (e.g., EQ 24 District Level).

Add Security groups to the model

a. Open SQL Server Management Studio and connect to Azure.
b. Extend Databases, find EQ 24 Database, and extend to this point:



c. Right click on the role, select “Properties.” In the pop up select a page “Membership.” Click “Add.” In a new pop up, search for emails or an appropriate group to add to this security role. For district level, add “EQ 24 District Level” group. Do this for each role.

Publish in Power BI

1. Open the EQ 24 Template.pbix file in Power BI Desktop.
2. You will be prompted to log in to your Azure account.
3. You may receive an error message stating that your account does not have access to server. This is correct – it is trying to authenticate your account to an invalid server (the one used for initial development of this model).
4. Select ‘Edit’ on the warning message and enter your Analysis Services connection string.
    a. It should be formatted like this: asazure://**eastus**.asazure.windows.net/**myservername**
5. Leave the database name as is – this is hardcoded in the Tabular Model.
6. When prompted, select “Model.” This will select the Tabular Model within the Analysis Services server.
7. Once connected, select “Publish” in the top ribbon menu. You may be prompted to log in to your Power BI account.
8. Select the workspace where you would like to deploy the project. This should match the context for each report.

Select the publishing and sharing options in Power BI appropriate for your use cases.

Dynamic Row Level Security

Security Goals

Security is based on the roles of all logged in users, as well as the section associations in the Ed-Fi ODS for the Teacher Level. The security model uses DAX and Azure Analysis Services to work properly.

District Level

Superintendents and district-level personnel must have access to all their district data.

Security Steps Taken

No additional steps. Users have access to all tables.

School Level

Principals and school administrators must have access to all data for only their school.

Security Steps Taken

The measure [RLS_SchoolId] holds SchoolId of a logged in user:

```
RLS_SchoolId:=  CALCULATE(
                FIRSTNONBLANK(StaffEducationOrganizationAssignmentAssociation[SchoolId], TRUE()),
                FILTER(StaffEducationOrganizationAssignmentAssociation,
                StaffEducationOrganizationAssignmentAssociation[LoginID] = [Username])
                            ) [LoginID] = [Username])
                    )
```

Tables of this level are filtered on SchoolID of a logged in user. See following table. SchoolID is referenced from the StaffEducationOrganizationAssignmentAssociation table.

Our main goal is to filter students who associate with a logged in Staff member. Because we must move through Staff>StaffSectionAssociation >StudentSectionAssociation, it is not possible to use StudentSchoolAssociation table to filter Student table and attached StudentAssessmentScoreResult Tables. Doing so would create an ambiguous path among the tables.

The column CurrentSchoolID helps to filter out students who left the school but are still attached to staff Members in the previous school.

CurrentSchoolID column is added to the following tables:

- Student
- StudentAssessmentScoreResult
- MC_ StudentAssessmentScoreResult
- iReady_ StudentAssessmentScoreResult

Table 1 School Level Security

Table	DAX Filter
Student	=Student[Current School ID] = [RLS_SchoolID]
StudentSectionAssociation	=StudentSectionAssociation[SchoolID] = [RLS_SchoolID]
School	=School[SchoolID] = [RLS_SchoolID]
StudentAssessmentScoreResult	=StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]
StaffSectionAssociation	=StaffSectionAssociation[SchoolID] = [RLS_SchoolID]
Section	=Section[SchoolID] = [RLS_SchoolID]
MC_StudentAssessmentScore Result	=MC_StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]
iReadyStudentAssessmentScoreResult	=iReady_StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]

Teacher Level

Teachers must have access to only the students in their courses.

The teacher level has all filters from the School Level and the following additional security filters:

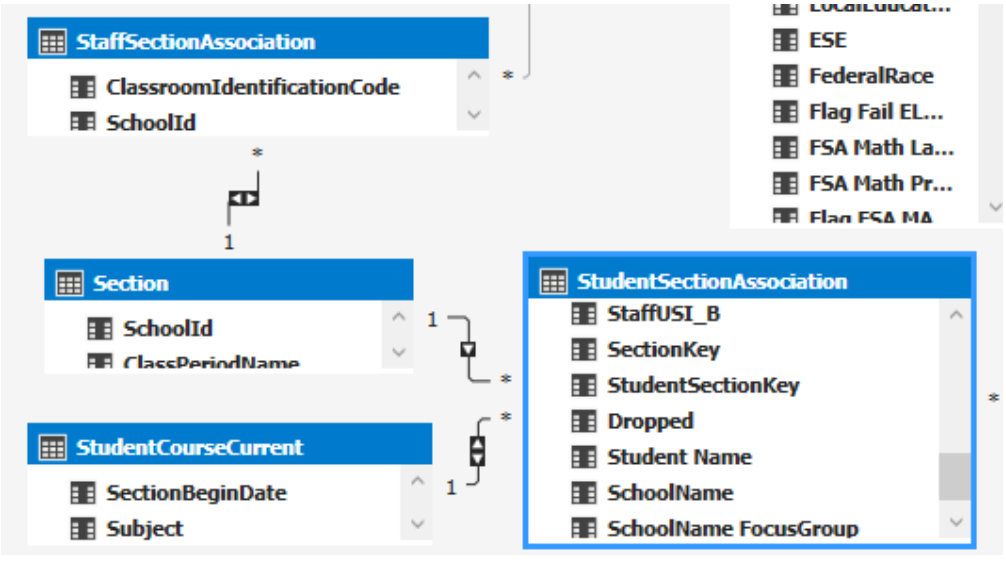
- Staff
- StaffEducationOrganizationAssignmentAssociation
- Section

Table 2 Teacher Level Security

Table	**DAX Filter**
Staff	=Staff[StaffUSI] = [RLS_StaffUSI]
StaffEducationOrganizationAssignmentAssociation	=StaffEducationOrganizationAssignmentAssociation[StaffUSI] = [RLS_StaffUSI]
Student	=Student[Current School ID] = [RLS_SchoolID]
StudentSectionAssociation	=StudentSectionAssociation[SchoolID] = [RLS_SchoolID]
School	=School[SchoolID] = [RLS_SchoolID]
StudentAssessmentScoreResult	=StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]
StaffSectionAssociation	=StaffSectionAssociation[SchoolID] = [RLS_SchoolID]
Section	=Section[SchoolID] = [RLS_SchoolID]
Section	=[SectionKey] IN SELECTCOLUMNS(FILTER(StaffSectionAssociation,RELATED(Staff[StaffUSI]),"SectionKey",[SectionKey])
MC_StudentAssessmentScore Result	=MC_StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]
iReadyStudentAssessmentScoreResult	=iReady_StudentAssessmentScoreResult[Current School ID] = [RLS_SchoolID]

In the following diagram, the Section table works as a fact table between StaffSectionAssociation and StudentSectionAssociation. The reason it is set as a fact table is because we may have more than one teacher teaching the same students in the same class/section. Defining this relationship helps attach the same records in the StudentSectionAssociation table to different records in the StaffSectionAssociation table:

Figure 1 Relationship between StaffSectionAssociation and StudentSectionAssociation



Security Steps Taken

1. Create a measure that finds StaffUSI of the logged in User:

```
RLS_StaffUSI:= CALCULATE(
FIRSTNONBLANK(StaffEducationOrganizationAssignmentAssociation[StaffUSI], TRUE()),
FILTER(StaffEducationOrganizationAssignmentAssociation,
StaffEducationOrganizationAssignmentAssociation[LoginID] = [Username])
        )
```

2. Filter StaffEducationOrganizationAssignmentAssociation table by logged in User using the measure above:

```
= StaffEducationOrganizationAssignmentAssociation[StaffUSI] = [RLS_StaffUSI]
```

3. Filter Staff Table by the logged in User using measure in step 1:

```
= Staff[StaffUSI] = [RLS_StaffUSI]
```

No need to filter StaffSectionAssociation since Staff to StaffSectionAssociation is a “one to many” relationship and StaffSectionAssociation table will be filtered at the same time and on the same StaffUSI as the Staff table.

4. Filter Section table. StaffSectionAssociation to StudentSectionAssociation is a “many to many” relationship which is not currently supported by Visual Studio Data Tools Compatibility Level 1200. The table Section was added as a fact or bridge table to relate two tables mentioned above. SectionKey column is created in each table to relate these three tables. SectionKey column concatenates these columns:

   - SchoolId
   - ClassroomIdentificationCode
   - LocalCourseCode
   - Term
   - UniqueSectionCode
   - ClassPeriodName

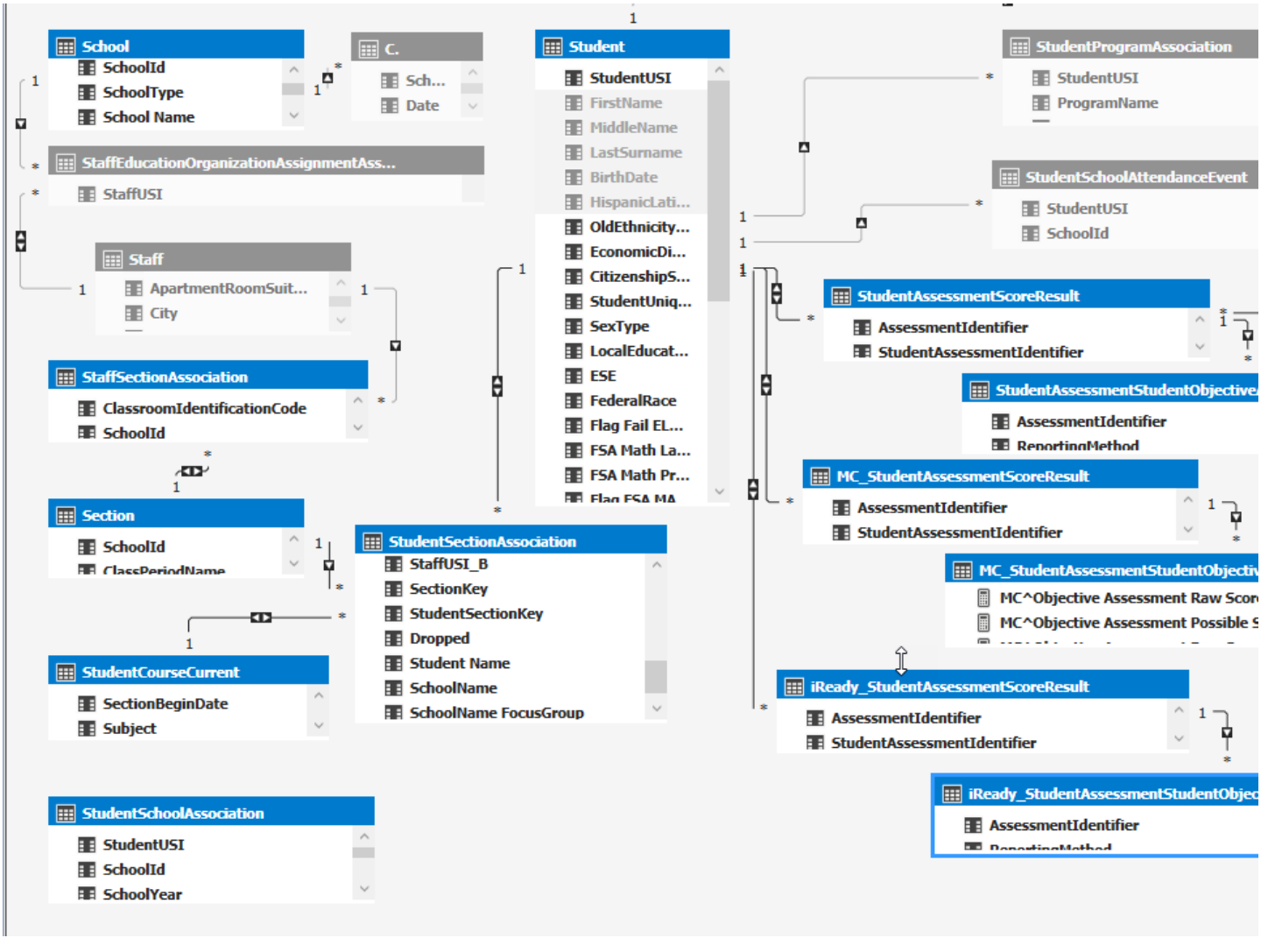
The measure below filters the Section table by the SectionKey which belongs to StaffUSI in the already filtered StaffSectionAssociation table:

```
= [SectionKey] IN SELECTCOLUMNS(FILTER(
                StaffSectionAssociation,RELATED(Staff[StaffUSI]) = [RLS_StaffUSI]
                  ),
                "SectionKey", [SectionKey])
```

The table StudentSectionAssociation will be filtered together with the Section table on the same SectionKey. Now logged in users can only see students who are belong to the courses they teach.

Security Flow Chart

Figure 2 Dynamic RLS Security Flow Chart



Technical Contacts for Project

Julia Brown	| Associate Full Stack Developer	| brownj@nefec.org
Sherod Keen	| Senior Applications Support Analyst	| keens@nefec.org
Shane Jay Fairbairn, PhD, MEd	| Supervisor of Instructional Technology	| fairbairns@nefec.org

A Special Thank You

Ronda Bourn & Jake Massey, NEFEC Math Subject Matter Experts.

Each of the teachers from the Math Focus group.

The Center for Educational Leadership and Technology for the first iteration of this important work.