

Developing Streaming Datasets and Real-time Dashboards



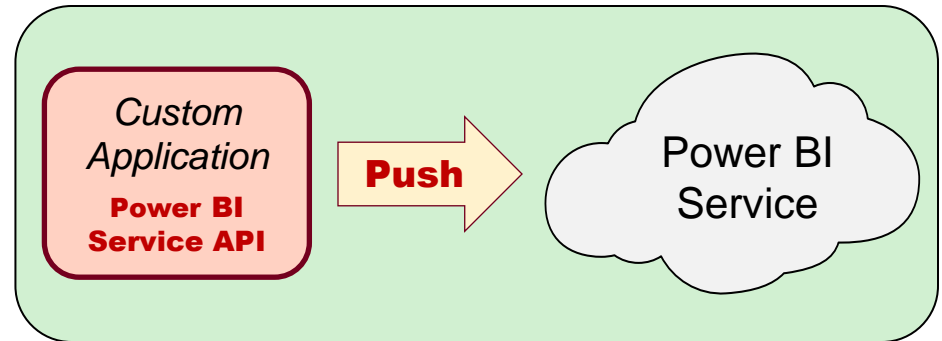
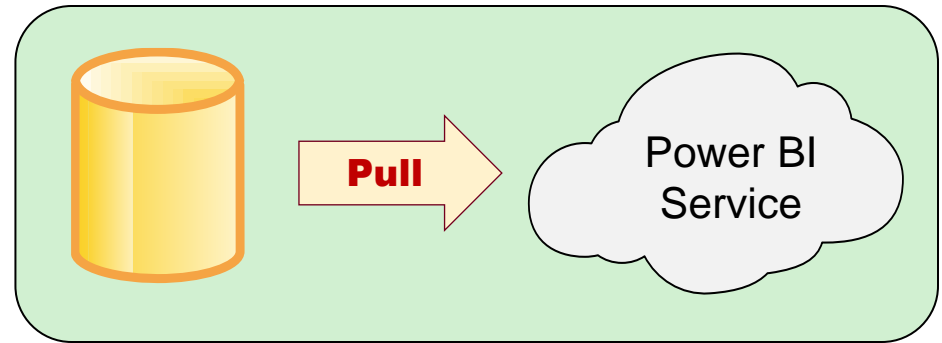
Agenda

- Introduction to Real-time Datasets
 - Creating a Streaming Dataset with the API
 - Designing Dashboards with Streaming Data Tiles
 - Creating a Push Dataset with Real-time Data
 - Integrating Azure Streaming Analytics Jobs



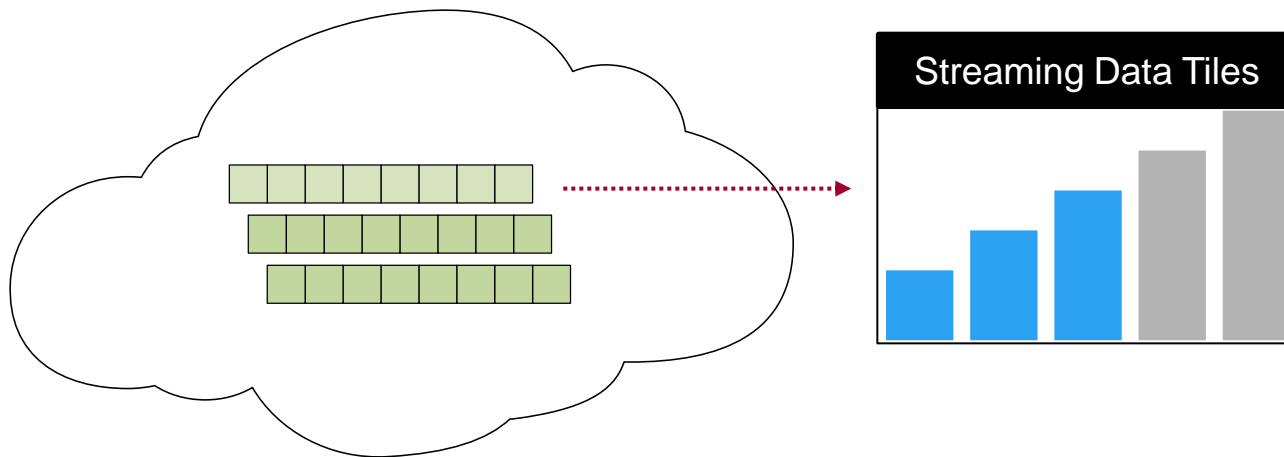
Pull Datasets versus Real-time Datasets

- Pull Datasets
 - Imported Datasets
 - DirectQuery Datasets
 - Live Connect Datasets
- Real-time Datasets
 - Streaming Datasets
 - Push Datasets
 - Hybrid Datasets



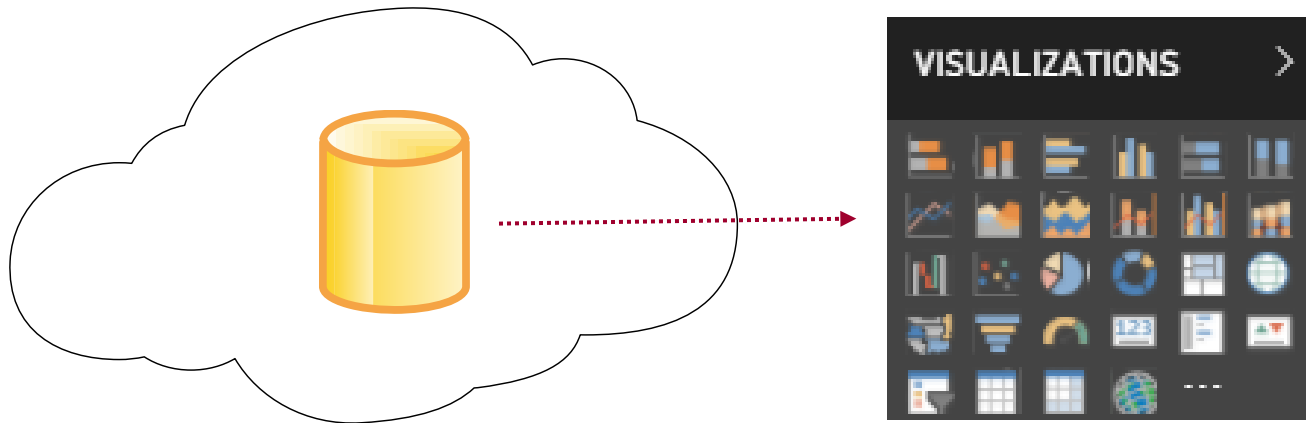
Streaming Datasets

- Data stored in cloud-based cache – not persisted in DB
- Restricted to single table - no rich data modeling
- Not supported by standard Power BI report designer
- Dashboard created using specialized streaming data tiles
- No support for DAX, aggregation or filtering



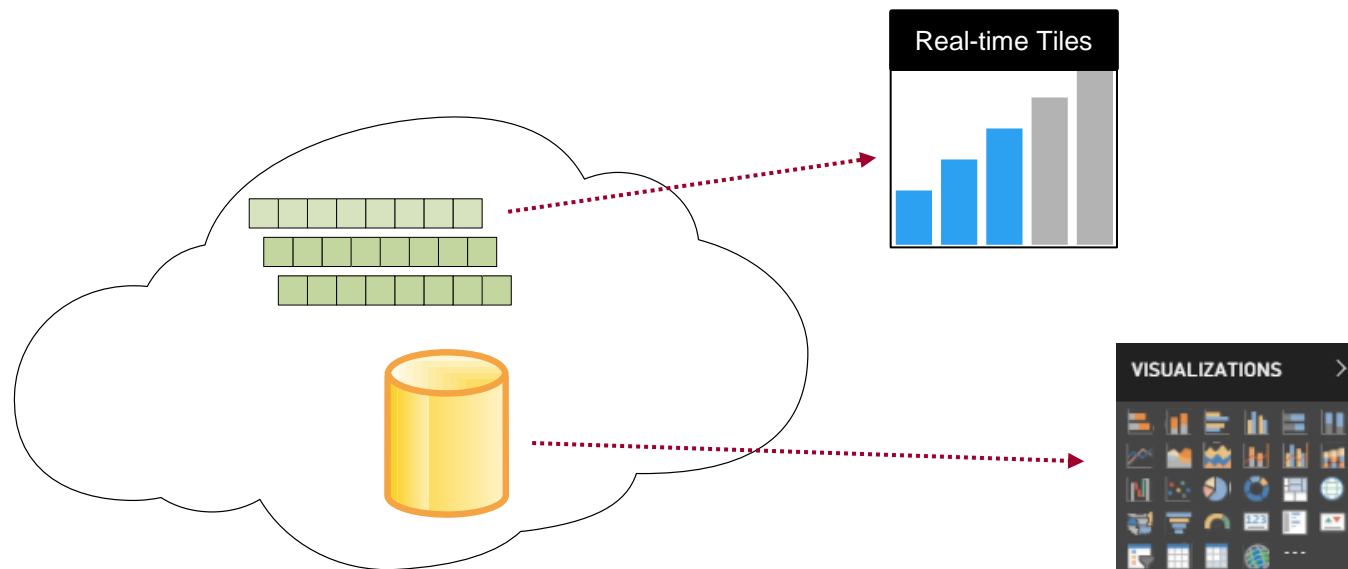
Push Datasets

- Data stored in Azure SQL DB – not in cache
- Supports multiple tables and table relationships
- Supported by standard Power BI report designer
- Supports DAX, measures, aggregation & filtering



Hybrid Datasets

- Data stored in cloud-based cache **AND** in Azure SQL DB
- Restricted to single table and no rich data modeling
- Supported by streaming data tiles
- Supported by Power BI report designer





DEMO

Creating a Hybrid Dataset by Hand

Agenda

- ✓ Introduction to Real-time Datasets
- Creating a Streaming Dataset with the API
 - Designing Dashboards with Streaming Data Tiles
 - Creating a Push Dataset with Real-time Data
 - Integrating Azure Streaming Analytics Jobs



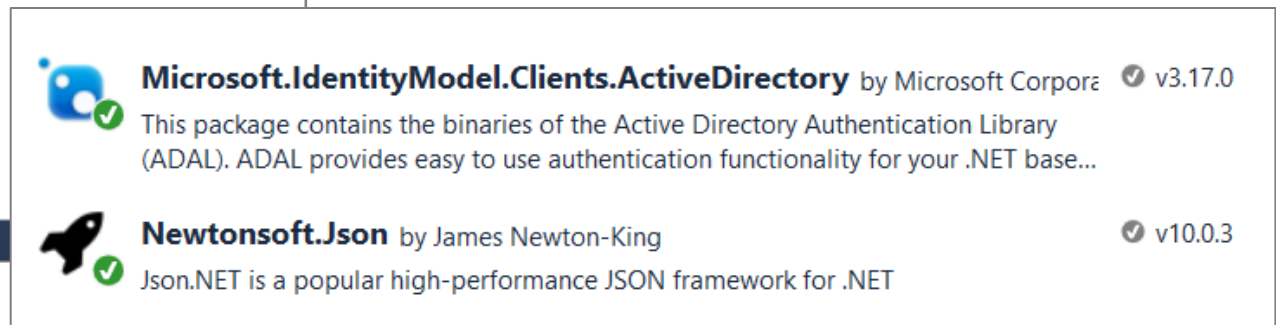
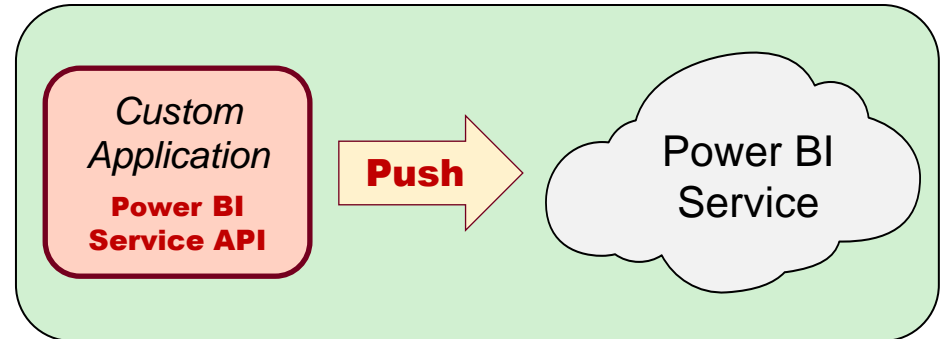
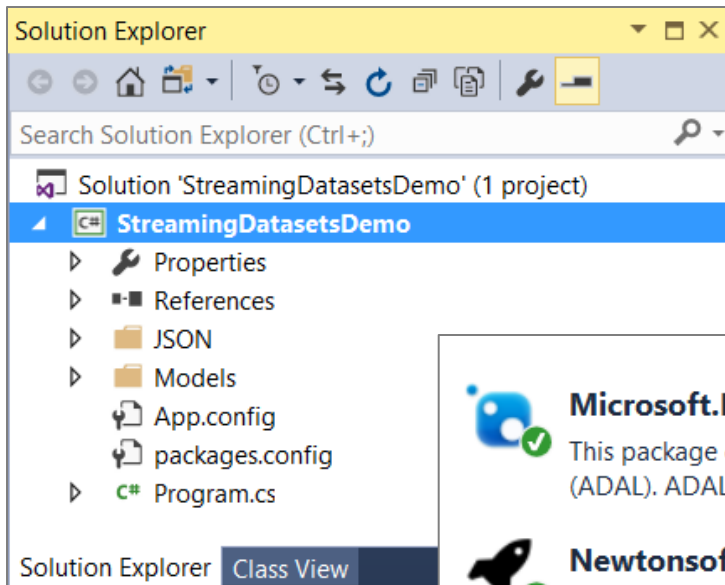
Real-time Scenario Being Simulated

- Heisenberg labs processes chemicals
 - Required to monitor temperatures during processing
 - Temperature must be monitored on per-second basis
 - Send alerts if temperature exceeds preset threshold



The StreamingDatasetsDemo Project

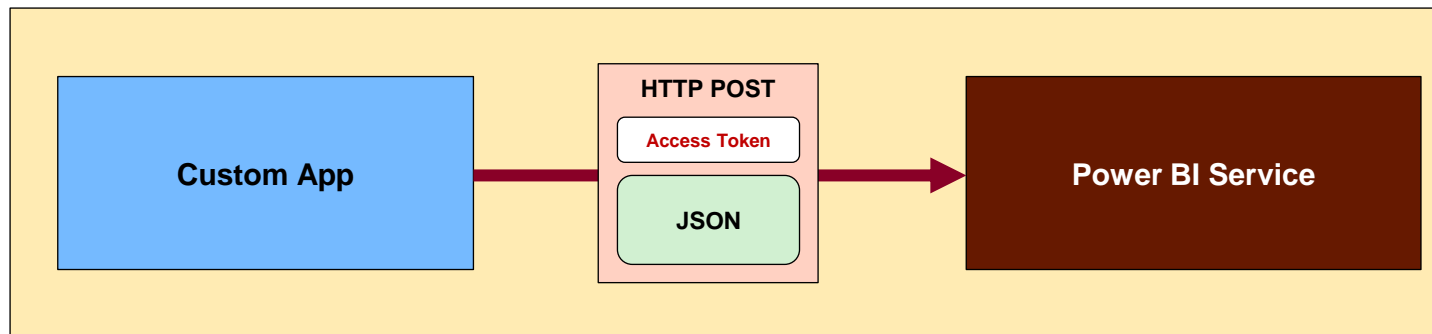
- Console application project in Visual Studio 2017
 - Installed package for Azure AD Authentication library
 - Installed package to serialize .NET objects to JSON



Executing Power BI Service API Calls

- Generic helper methods designed to execute HTTP operations

```
private void ExecutePostRequest(string restUri, string postBody) {  
    1 // prepare REST call  
    HttpContent body = new StringContent(postBody);  
    body.Headers.ContentType = new MediaTypeWithQualityHeaderValue("application/json");  
    HttpClient client = new HttpClient();  
    client.DefaultRequestHeaders.Add("Accept", "application/json");  
    client.DefaultRequestHeaders.Add("Authorization", "Bearer " + AccessToken);  
    2 // execute REST call  
    HttpResponseMessage response = client.PostAsync(restUri, body).Result;  
}
```



Creating a Streaming Dataset

- Streaming dataset created using JSON schema definition
 - Streaming dataset limited to a single table
 - Columns defined using name and datatype
 - No support for any other column properties (e.g. formatting)

```
DemoStreamingDataset.json  X
{
  "name": "TemperatureReadings",
  "defaultMode": "Streaming",
  "tables": [
    { "name": "TemperatureReadings",
      "columns": [
        { "name": "Run", "dataType": "string" },
        { "name": "Time", "dataType": "Datetime" },
        { "name": "TimeWindow", "dataType": "string" },
        { "name": "TargetTemperature", "dataType": "Double" },
        { "name": "MinTemperature", "dataType": "Double" },
        { "name": "MaxTemperature", "dataType": "Double" },
        { "name": "BatchA", "dataType": "Double" },
        { "name": "BatchB", "dataType": "Double" },
        { "name": "BatchC", "dataType": "Double" }
      ]
    }
  ]
}
```

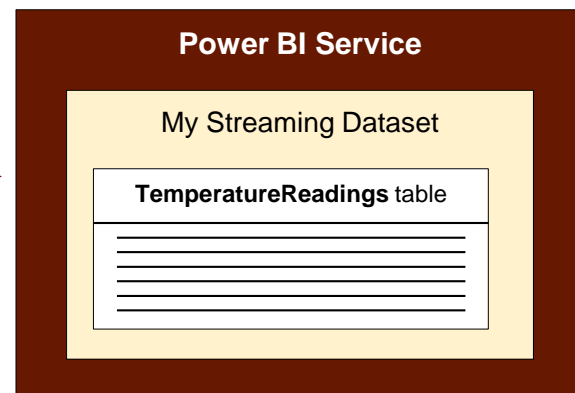
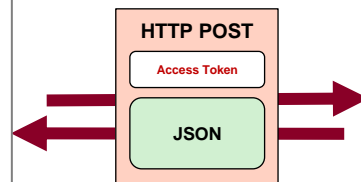


Creating a Custom Dataset

- Dataset created by executing HTTP POST operation
 - One-time operation done as application begins running

```
1 // prepare call to create new dataset
2 string restUrlDatasets = ProgramGlobalConstants.PowerBiServiceRootUrl + "datasets";
3 string jsonNewDataset = Properties.Resources.NewDataset_json;
// execute REST call to create new dataset
string json = ExecutePostRequest(restUrlDatasets, jsonNewDataset);
// retrieve Guid to track dataset ID
Dataset dataset = JsonConvert.DeserializeObject<Dataset>(json);
CustomDatasetId = dataset.id;
```

```
DemoStreamingDataset.json
{
  "name": "TemperatureReadings",
  "defaultMode": "Streaming",
  "tables": [
    {
      "name": "TemperatureReadings",
      "columns": [
        { "name": "Run", "dataType": "string" },
        { "name": "Time", "dataType": "Datetime" },
        { "name": "TimeWindow", "dataType": "string" },
        { "name": "TargetTemperature", "dataType": "Double" },
        { "name": "MinTemperature", "dataType": "Double" },
        { "name": "MaxTemperature", "dataType": "Double" },
        { "name": "BatchA", "dataType": "Double" },
        { "name": "BatchB", "dataType": "Double" },
        { "name": "BatchC", "dataType": "Double" }
      ]
    }
  ]
}
```



Adding Rows by Converting C# to JSON

```
TemperatureReadingsRow row = new TemperatureReadingsRow {  
    Run = RunName,  
    Time = DateTime.Now,  
    TimeWindow = currentTimeWindow,  
    TargetTemperature = 212,  
    MinTemperature = 100,  
    MaxTemperature = 250,  
    BatchA = temperatureBatchA,  
    BatchB = temperatureBatchB,  
    BatchC = temperatureBatchC,  
};  
  
TemperatureReadingsRow[] rows = { row };  
TemperatureReadingsRows temperatureReadingsRows = new TemperatureReadingsRows { rows = rows };  
string jsonNewRows = JsonConvert.SerializeObject(temperatureReadingsRows);  
string restUrlTargetTableRows = string.Format("{0}/{1}/tables/TemperatureReadings/rows", restUrlDatasets, DatasetId);  
string jsonResultAddExpenseRows = ExecutePostRequest(restUrlTargetTableRows, jsonNewRows);
```

```
public class TemperatureReadingsRow {  
    public string Run { get; set; }  
    public DateTime Time { get; set; }  
    public string TimeWindow { get; set; }  
    public double TargetTemperature { get; set; }  
    public double MinTemperature { get; set; }  
    public double MaxTemperature { get; set; }  
    public double BatchA { get; set; }  
    public double BatchB { get; set; }  
    public double BatchC { get; set; }  
}  
  
class TemperatureReadingsRows {  
    public TemperatureReadingsRow[] rows { get; set; }  
}
```



```
{  
  "rows": [  
    {  
      "Run": "Run 06",  
      "Time": "2017-10-05T22:43:40.364569-04:00",  
      "TimeWindow": "22:43:30",  
      "TargetTemperature": 212.0,  
      "MinTemperature": 100.0,  
      "MaxTemperature": 250.0,  
      "BatchA": 152.73999999999995,  
      "BatchB": 152.78,  
      "BatchC": 152.25  
    }  
  ]  
}
```





DEMO

Creating a Streaming Dataset using the Power BI Service API

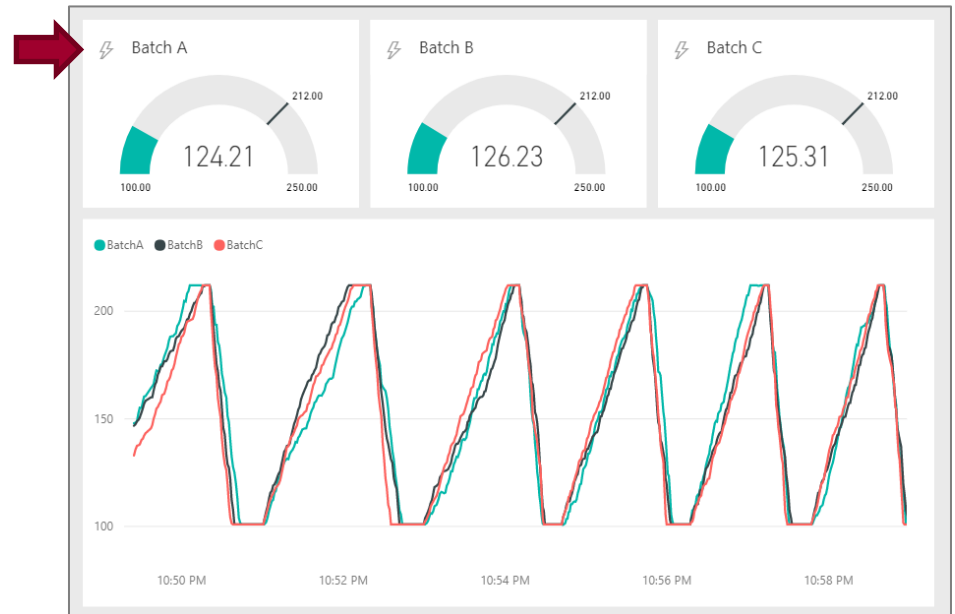
Agenda

- ✓ Introduction to Real-time Datasets
- ✓ Creating a Streaming Dataset with the API
- Designing Dashboards with Streaming Data Tiles
 - Creating a Push Dataset with Real-time Data
 - Integrating Azure Streaming Analytics Jobs



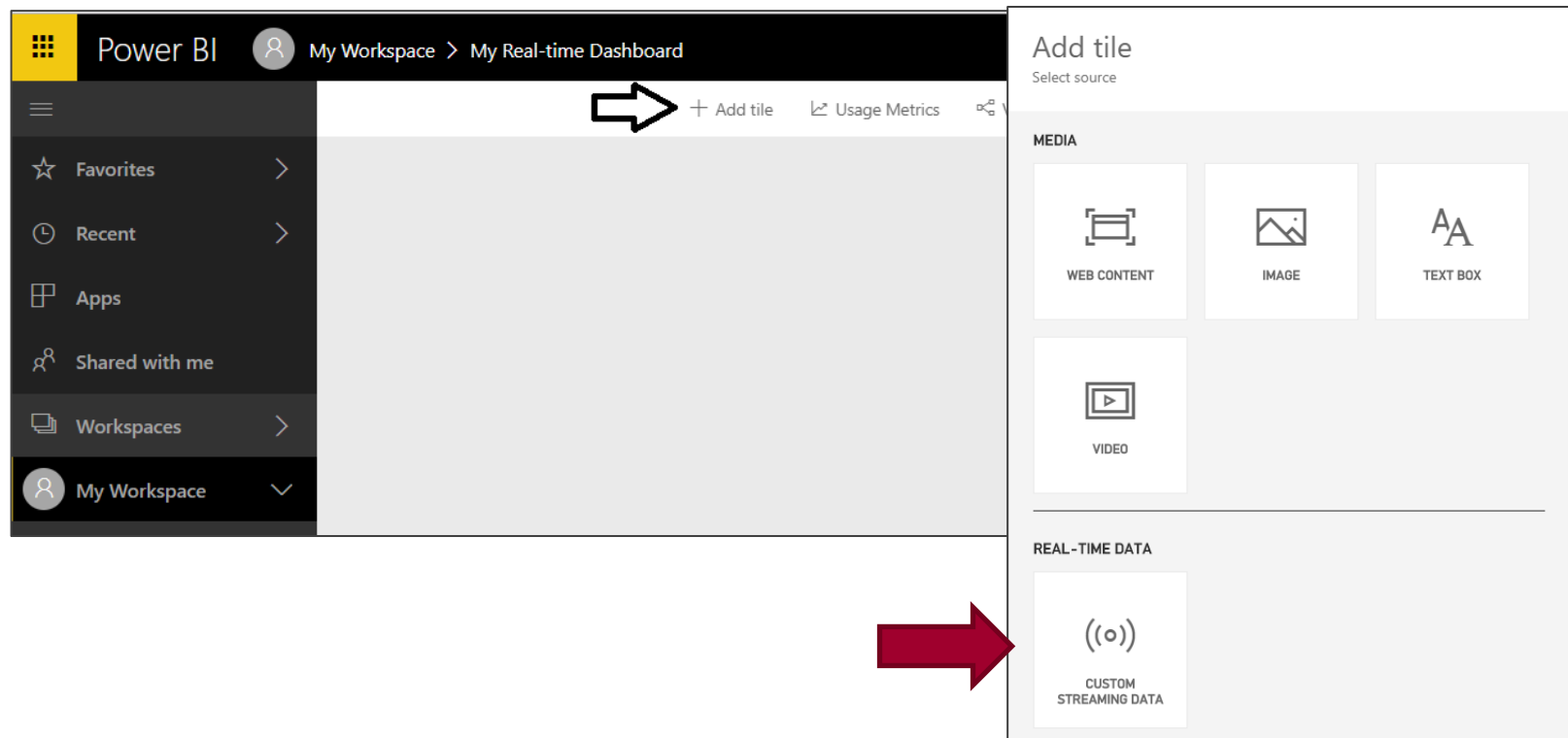
Streaming Data Tiles

- Streaming Data surfaced using Streaming Data Tiles
 - Used to surface data values from streaming dataset cache
 - Optimized to provide smooth animations for real-time data
- Available set of streaming titles as of today
 - Card
 - Gauge
 - Line chart
 - Cluster Bar Chart
 - Cluster Column Chart



Creating Dashboards with Streaming Datasets

- You cannot use the Power BI report designer
 - Instead, you add real-time data tiles directly to a dashboard
 - Real-time data tiles different from standard set of Power BI visuals



Creating a New Streaming Data Tile

- When creating a streaming data tile...
 - Select dataset that is a streaming dataset or a hybrid dataset
 - Choose the type of data streaming tile

Add a custom streaming data tile

Choose a streaming dataset

[+ Add streaming dataset](#)

YOUR DATASETS

Demo 1: Streaming Dataset

[Manage datasets](#)

[Back](#) [Next](#) [Cancel](#)

Add a custom streaming data tile

Choose a streaming dataset > Visualization design

Visualization Type

Card

Card

Line chart

Clustered bar chart

Clustered column chart

Gauge

[Manage datasets](#)

[Back](#) [Next](#) [Cancel](#)



Real-time Data Tile Field Pane

Add a custom streaming data tile
Choose a streaming dataset > Visualization design

Visualization Type

Card

Fields

BatchA

+ Add value

[Manage datasets](#)

Back Next Cancel



Real-time Data Tile Format Pane

Add a custom streaming data tile
Choose a streaming dataset > Visualization design

Visualization Type
Card

Data label

Display units
None

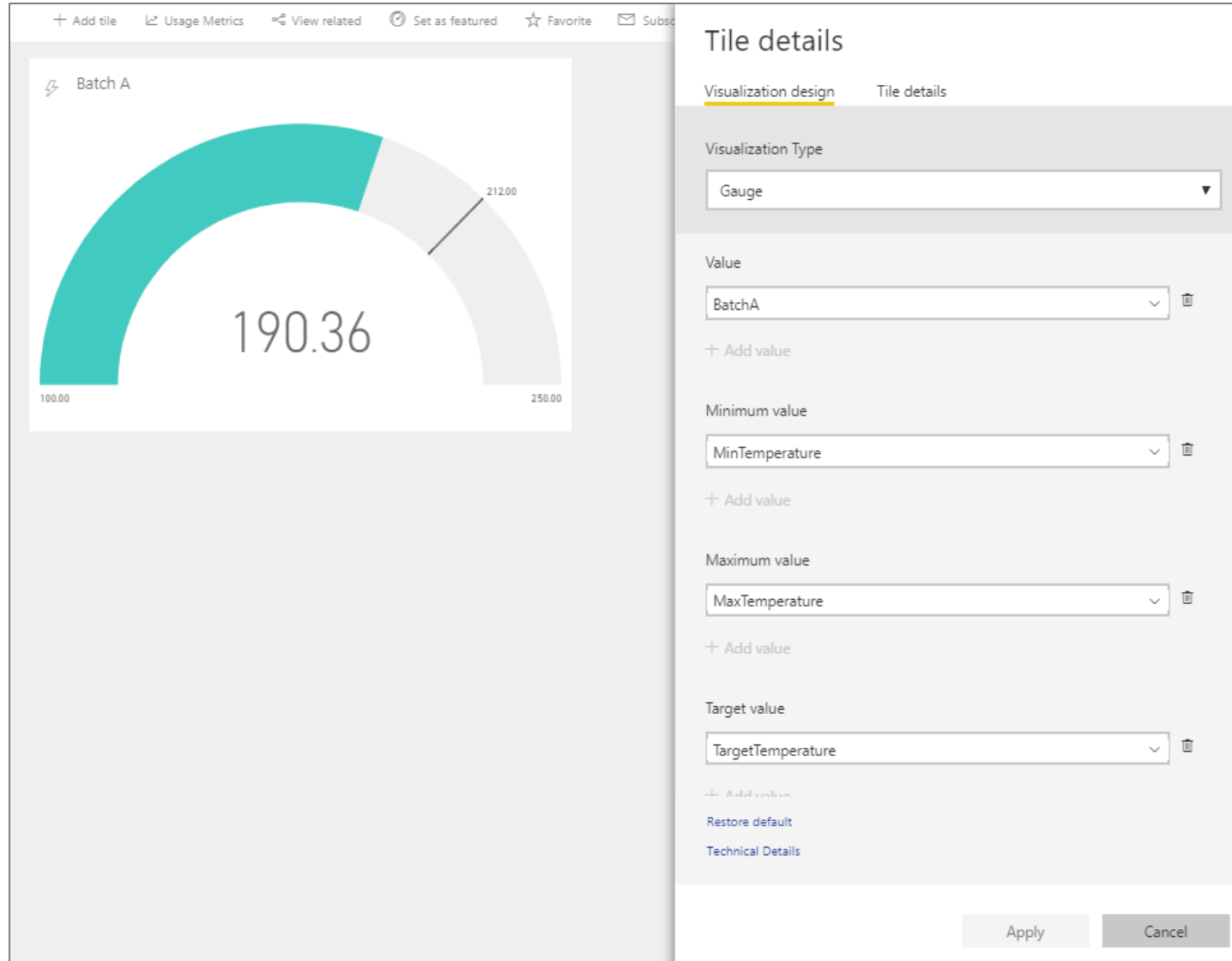
Value decimal places
2

[Manage datasets](#)

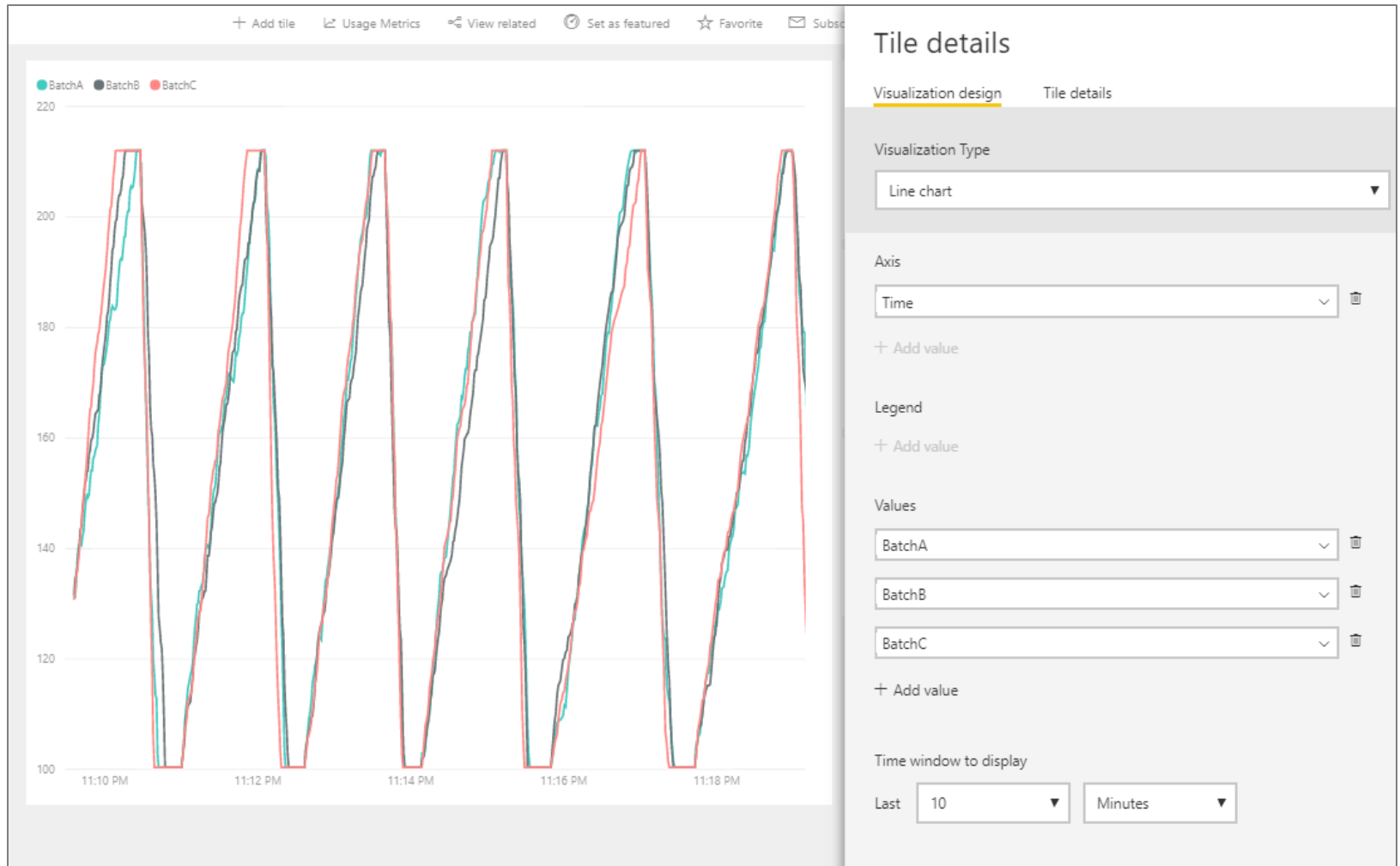
Back Next Cancel



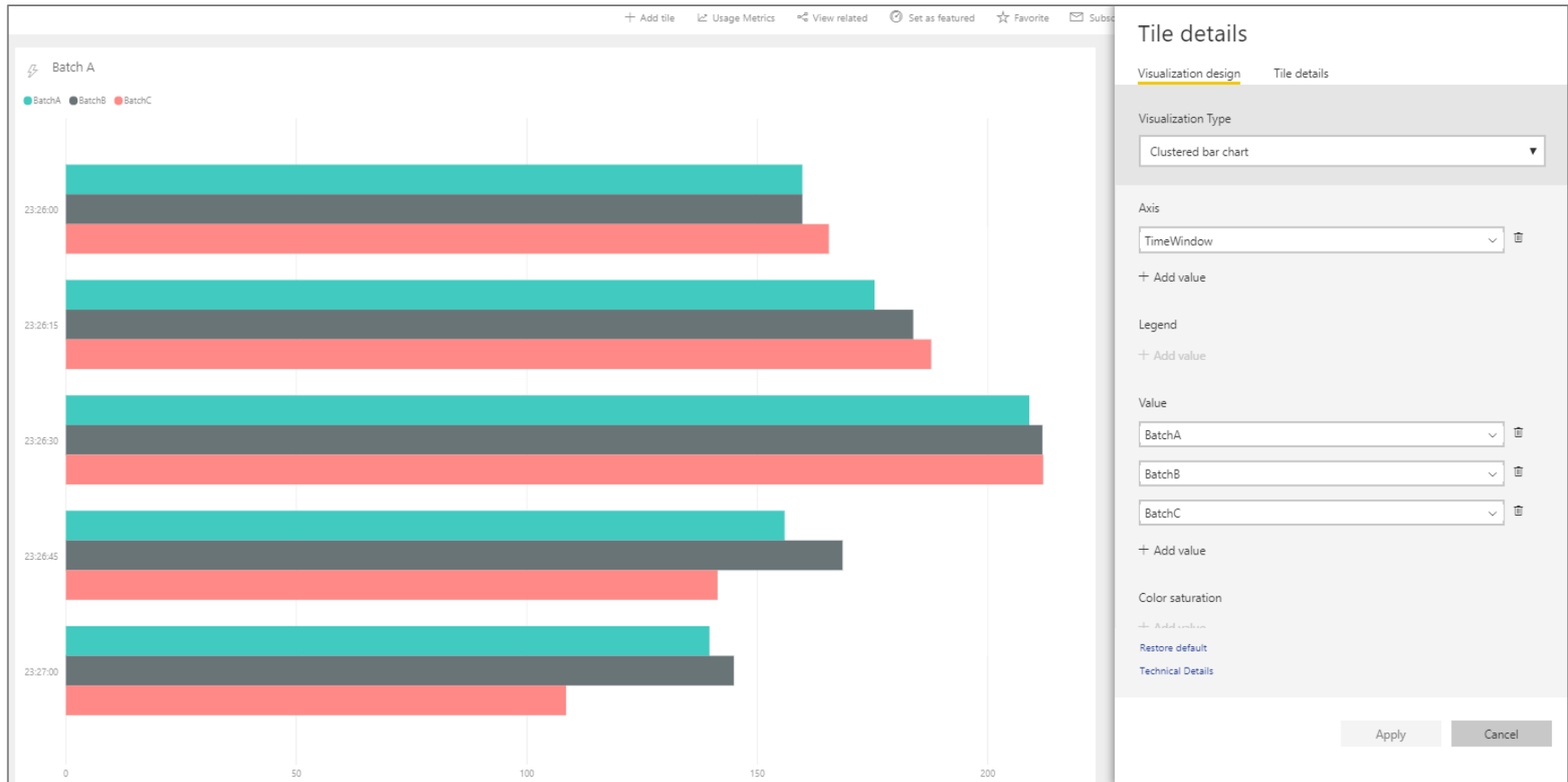
Streaming Gauge Tile



Streaming Line Chart Tile



Streaming Clustered Bar Chart Tile



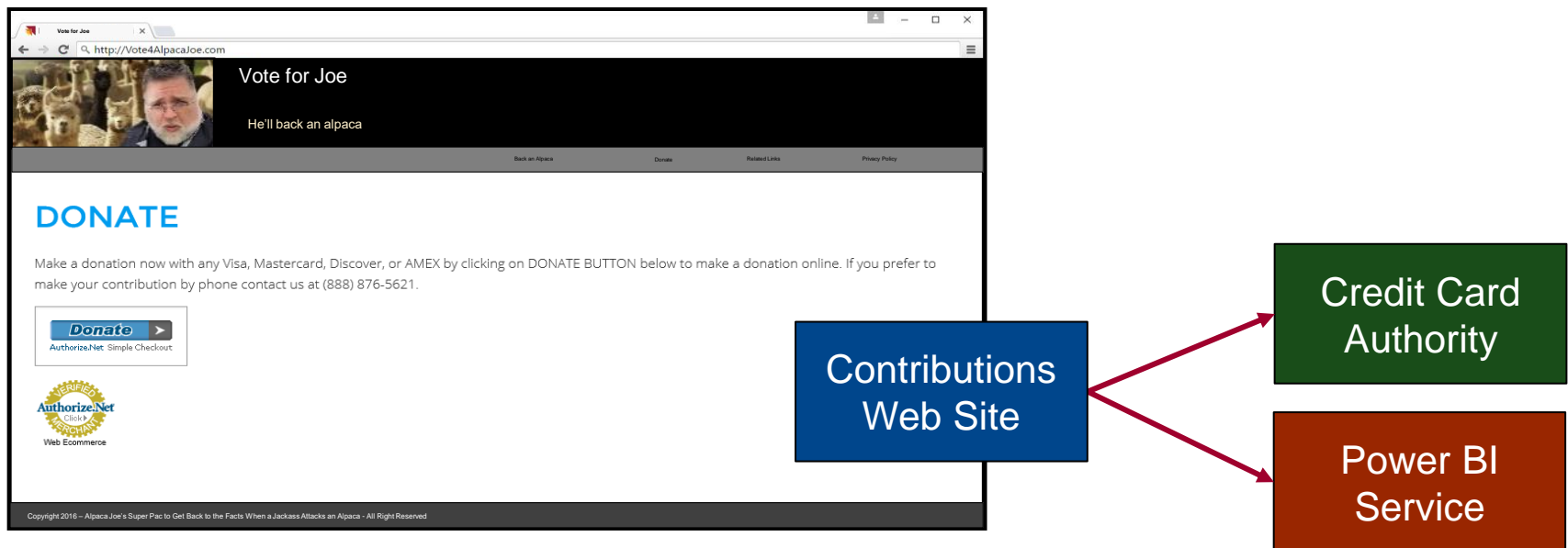
Agenda

- ✓ Introduction to Real-time Datasets
- ✓ Creating a Streaming Dataset with the API
- ✓ Designing Dashboards with Streaming Data Tiles
- Creating a Push Dataset with Real-time Data
 - Integrating Azure Streaming Analytics Jobs



Real-time Scenario Being Simulated

- Contribution website accepts contributions from donors
 - Website calls web service to process credit card transaction
 - Website calls to Power BI REST API to create & update dataset



Creating a Push Dataset

- Push dataset created using JSON schema definition
 - Push dataset can contain multiple tables and table relationships
 - Tables can contains measures as well as columns
 - Columns & measures can be defined with formatString & dataCategory

```
{
  "name": "@DatasetName",
  "defaultMode": "Push",
  "tables": [
    {
      "name": "Contributions",
      "columns": [
        { "name": "ContributionID", "dataType": "Int64" },
        { "name": "Contributor", "dataType": "string" },
        { "name": "State", "dataType": "string", "dataCategory": "StateOrProvince" },
        { "name": "City", "dataType": "string", "dataCategory": "Place" },
        { "name": "Zipcode", "dataType": "Int64", "dataCategory": "PostalCode" },
        { "name": "Gender", "dataType": "string" },
        { "name": "Time", "dataType": "Datetime", "formatString": "hh:mm:ss" },
        { "name": "TimeWindow", "dataType": "string" },
        { "name": "Amount", "dataType": "Double", "formatString": "$#,##0" }
      ],
      "measures": [
        { "name": "Total Contributions", "expression": "SUM(Contributions[Amount])", "formatString": "$#,##0" },
        { "name": "Contribution Count", "expression": "COUNTROWS(Contributions)", "formatString": "#,##0" },
        { "name": "Max Contribution", "expression": "MAX(Contributions[Amount])", "formatString": "$#,##0" },
        { "name": "Average Contribution", "expression": "AVERAGE(Contributions[Amount])", "formatString": "$#,##0.00" }
      ]
    }
  ]
}
```

Creating a Tumbling Time Window

| ContributionID | Contributor | City | Zipcode | Gender | Time | Amount |
|----------------|--------------------|--|---------|--------|----------|----------|
| 1 | Travis Solomon | Lutz, FL | 33559 | Female | 12:03:00 | 50.00 |
| 2 | Rudolph Harris | Tarpon Springs, FL | 34688 | Female | 12:03:00 | 100.00 |
| 3 | Josh Franco | Davis Island, FL | 33606 | Male | 12:03:00 | 250.00 |
| 4 | Tim Duffy | Largo, FL | 33774 | Female | 12:03:00 | 200.00 |
| 5 | Napoleon Frye | Lutz, FL | 33558 | Male | 12:03:30 | 50.00 |
| 6 | Rodrigo Hart | Largo, FL | 33774 | Female | 12:03:30 | 100.00 |
| 7 | Will Levine | Tarpon Springs, FL | 34688 | Female | 12:03:30 | 75.00 |
| 8 | Eddie McCullough | Lutz, FL | 33558 | Male | 12:03:30 | 100.00 |
| 9 | Kirk Herrera | Lutz, FL | 33558 | Male | 12:03:30 | 150.00 |
| 10 | Antonia Bridges | Lutz, FL | 33558 | Female | 12:03:30 | 100.00 |
| 11 | Preston Cote | Davis Island, FL | 33606 | Female | 12:03:30 | 1,000.00 |
| 12 | Lonnie McCarty | Largo, FL | 33774 | Female | 12:03:30 | 50.00 |
| 13 | Humberto Parsons | Lutz, FL | 33558 | Female | 12:03:30 | 50.00 |
| 14 | Abel Perkins | Odessa, FL | 33556 | Female | 12:03:30 | 1,000.00 |
| 15 | Ernesto McLaughlin | Odessa, FL | 33556 | Female | 12:03:30 | 150.00 |
| 16 | Elbert Kinney | Macdill Air Force Base, Port Tampa, FL | 33621 | Male | 12:03:30 | 50.00 |
| 17 | Coleman Petersen | Lutz, FL | 33558 | Male | 12:03:30 | 125.00 |
| 18 | Nicky Roberson | Odessa, FL | 33556 | Female | 12:03:30 | 100.00 |
| 19 | Fritz Parrish | Lutz, FL | 33558 | Female | 12:03:30 | 50.00 |
| 20 | Freddie Sparks | Largo, FL | 33774 | Male | 12:03:30 | 50.00 |





DEMO

Designing a Real-time Dashboard using a Push Dataset

Real-time Dataset Matrix

| Feature | Streaming | Hybrid | Push |
|--------------------------|-------------------------------|--------|----------------------------------|
| Updates in real-time | Yes | Yes | Yes |
| Smooth animations | Yes | Yes | No |
| Backed by Azure SQL DB | No | Yes | Yes |
| Report Designer Support | No | Yes | Yes |
| Allow Rich Data Modeling | No | No | Yes |
| Ingestion Rate | 5 request/sec 15KB/request | | 1 request/second 16MB/request |



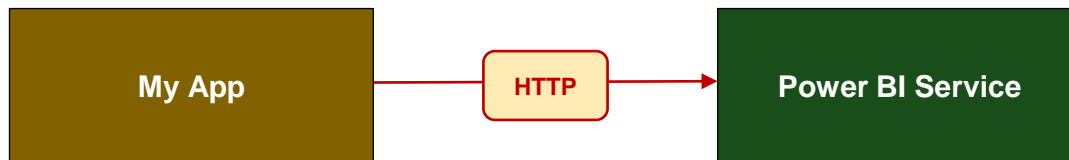
Agenda

- ✓ Introduction to Real-time Datasets
- ✓ Creating a Streaming Dataset with the API
- ✓ Designing Dashboards with Streaming Data Tiles
- ✓ Creating a Push Dataset with Real-time Data
- Integrating Azure Streaming Analytics Jobs

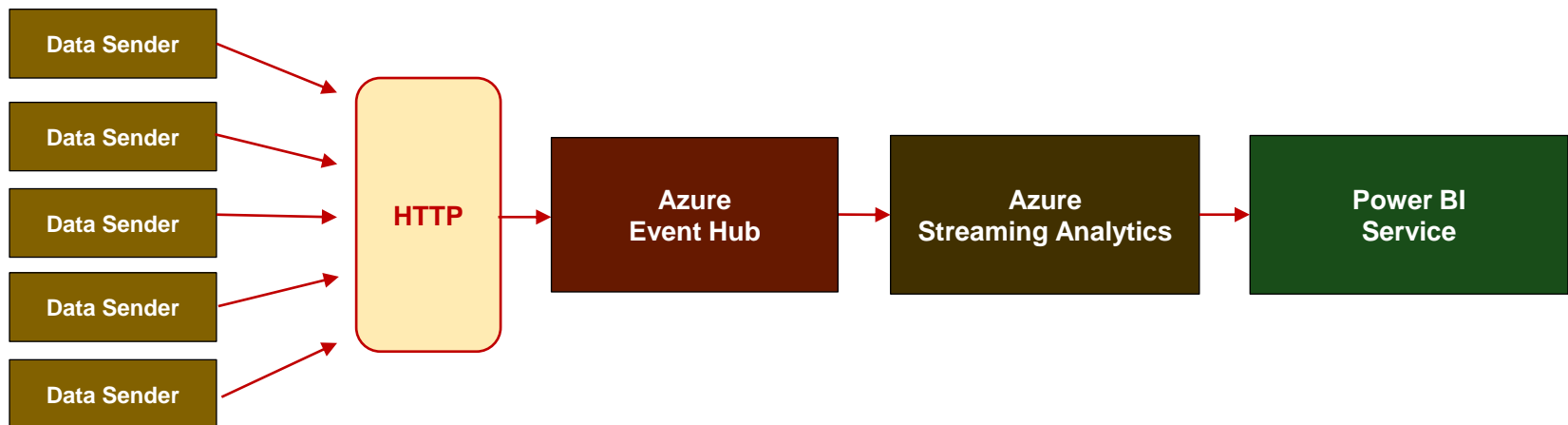


Scaling a Real-time Dashboard

- Low velocity data scenario

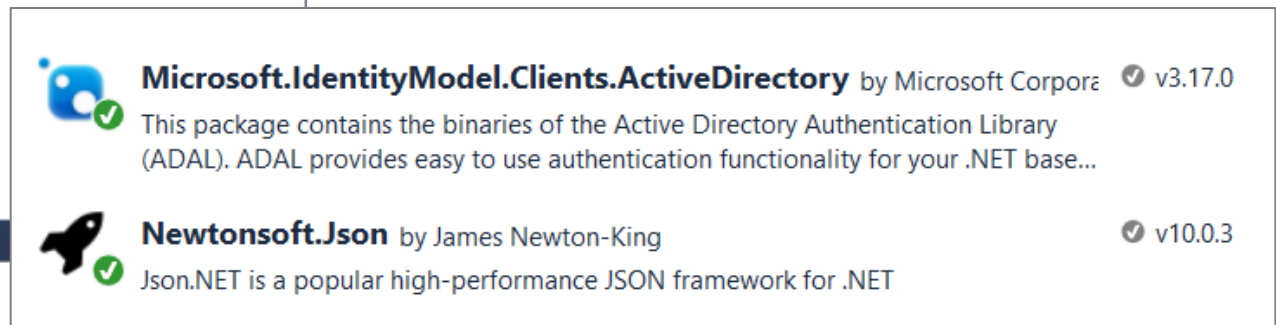
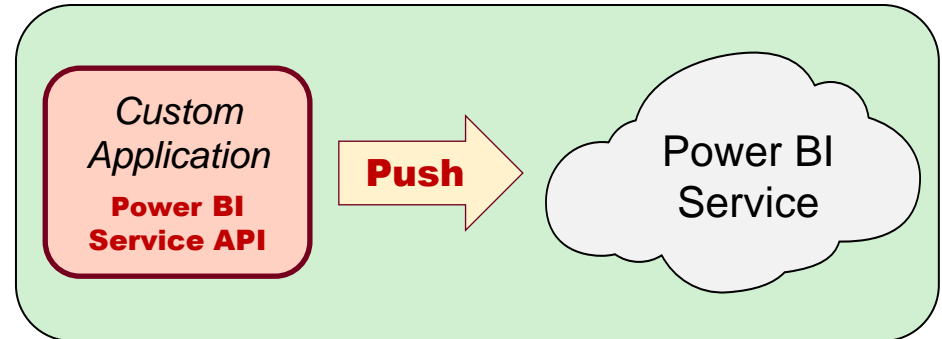
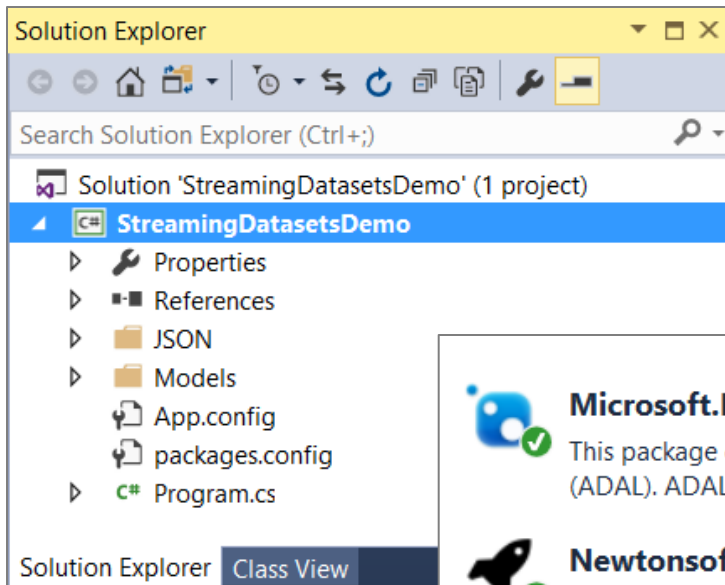


- High velocity data scenario



The StreamingDatasetsDemo Project

- Console application project in Visual Studio 2017
 - Installed package for Azure AD Authentication library
 - Installed package to serialize .NET objects to JSON





DEMO

Creating Streaming Datasets using Azure Streaming Analytics Jobs

Summary

- ✓ Introduction to Real-time Datasets
- ✓ Creating a Streaming Dataset with the API
- ✓ Designing Dashboards with Streaming Data Tiles
- ✓ Creating a Push Dataset with Real-time Data
- ✓ Integrating Azure Streaming Analytics Jobs

