

Build massive and lightning-fast analytics solutions with Azure Data Explorer

Looking to help your customers make business decisions with immediate impact based on real-time analysis of terabytes of data in seconds? In this session, you will build a real-time analytical solution with Azure Data Explorer (ADX), which supports interactive ad-hoc queries of petabytes of data.

Walk away with a solution for your frustrated customers, so they can make immediate and impactful business decisions from their data using ADX.

Contents

Infrastructure	2
Ingestion	5
Exploration	6
Questions	6
KQL.....	6
Results	7
Self-Study	8
Kusto Query Language (KQL).....	8
Power BI	9
PBI demo script.....	9
Connect to Help cluster	9
Create Power BI report	12
KQL – Results	16

Infrastructure

1. Open Lab: <http://bit.ly/2WCFDdz>
 - **Register Now:** Complete registration information on page

- Select **LAUNCH LAB**: Opens the **Environment Details** page
 - Open the Lab guide: **XXX**
2. Open Azure portal in private mode: <https://portal.azure.com>
- Connect with the **Azure Credentials**:

Azure Credentials

Here are your credentials to login to Microsoft Azure and access the On Demand Lab

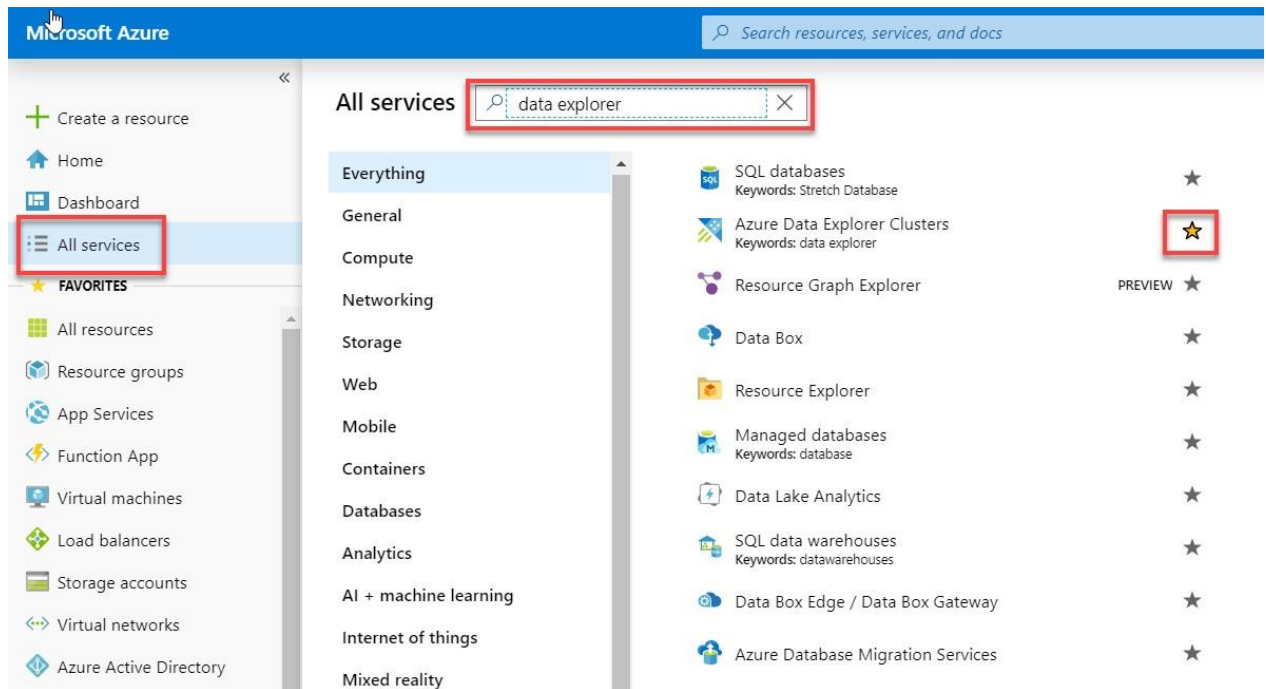
Username

Password

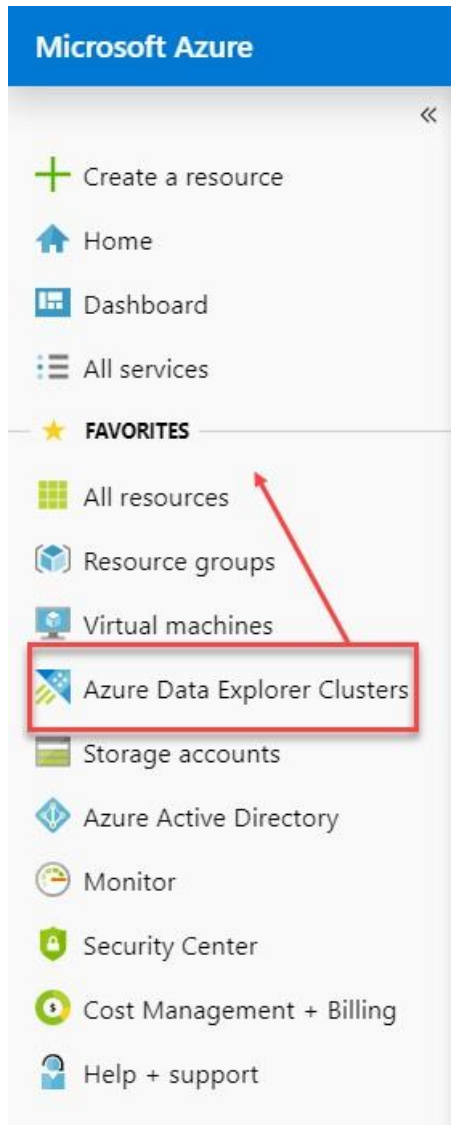
- The portal opens with your credentials:



3. Select **All Services** from left hand pane, Search for **data explorer** and click the **star** icon.



4. Drag **Azure Data Explorer Clusters** to the top of the **Favorite** menu.



5. Select **Azure Database Explorer** from **Favorite** menu and select the pre-deployed **Kusto cluster**.

Microsoft Azure

Home > Azure Data Explorer Clusters

Azure Data Explorer Clusters

CloudLabs AI Outlook


+ Add Edit columns Refresh

Subscriptions: Microsoft Managed Labs

Filter by name...

1 items

☐ NAME ↑↓

<input type="checkbox"/>	 kusto
--------------------------	---

Left sidebar navigation:

- Create a resource
- Home
- Dashboard
- All services
- FAVORITES**
- Azure Data Explorer Clusters**
- All resources
- Resource groups
- Virtual machines
- Storage accounts
- Azure Active Directory
- Monitor
- Security Center
- Cost Management + Billing
- Help + support

6. Select **Data** → **Databases + Add database**

kusto72719
Azure Data Explorer Cluster

Search (Ctrl+/)

+ Add database Stop Refresh Move Delete Fee

To use Azure Data Explorer, create at least one database. →

Resource group (change) : ODL-lab-72719
Location : East US
Subscription (change) : Microsoft Managed Labs Spektra - 05
Subscription ID : df085af0-1d07-4890-83c6-c1e6519aa212

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Query

Settings
Scale up
Scale out
Properties
Locks
Export template

Data
Databases

Welcome to Azure Data Explorer

STEP 1 - CLUSTER CREATION
STEP 2 - DATABASE CREATION

Your "kusto72719" cluster was successfully created.
Now you can create a database.

Create database

7. In the **Azure Data Explorer Database** window:

Azure Data Explorer Database
Create new database

Admin ⓘ : odl_user_72719@cloudlabsaioutloc


* Database name
Enter database name

Retention period (in days) ⓘ
3650
☐ Unlimited

Cache period (in days) ⓘ
31
☐ Unlimited

- Database name:** <alias>_adxdb
- Retention period** (cold compressed data (Azure Blob Storage)): **365**
- Cache period** (hot compressed data (SSD)): **31**

8. In **Databases**, select your new **alias_adxdb** database and Select **Query**

 **alias_adxdb (kusto72795/alias_adxdb)**
Azure Data Explorer Database

Search (Ctrl+/) <<

Query Edit Refresh Delete

Overview
Permissions
Query
Settings
Data ingestion
Properties

Resource group (change)
ODL-lab-72795

Retention period (in days)
3650 days

Location
East US

Cache period (in days)
31 days

Subscription (change)
Microsoft Managed Labs Spektra - 06

Subscription ID
f33373bf-4282-4926-a9a4-fcc8c3c6ff62

9. In the Web UI, create a table

```
// Create a table
.create table SampleTable
(timestamp:datetime, ApiVersion:string, User:string, RawHeader:dynamic)
```

Ingestion

Home > pmlabgithub > Databases > tzgitin1 (pmlabgithub/tzgitin1) - Query

tzgitin1 (pmlabgithub/tzgitin1) - Query

Search (Ctrl+/)

Overview
Permissions
Query
Settings
Data ingestion
Properties

New Tab x pmlabgithub.westus.tzgitin1

Open in web UI

Search Favorites

Run Recall Scope: pmlabgithub.WestUS.tzgitin1

```
1 // Create a json ingestion mapping
2 // Create a table
3
4 .create table SampleTable
5 (Timestamp:datetime, ApiVersion:string, User:string, RawHeader:dynamic)
6
7 .drop table SampleTable
8
9 // Create a json ingestion mapping
10 .create table SampleTable ingestion json mapping "Mapping01" ([{"column":"Timestamp","path":"$.header.time"}, {"column":"ApiVersion","path":"$.header.apiVersion"}])
11
12 // Create a json ingestion mapping
13 .create table SampleTable ingestion json mapping "Mapping01" ([{"column":"Timestamp","path":"$.header.time"}, {"column":"ApiVersion","path":"$.header.apiVersion"}])
14
15 // View ingestion mappings
16
17 .show table SampleTable ingestion json mappings
18
19 // Ingest from public blob:
20
21 .ingest into table SampleTable @'https://westuspublic.blob.core.windows.net/public/SampleData-500-4394582f-668f-4d03-8bba-58f87a7e88a0.json' with (j)
22
23
24 // Explore the data
25
26 SampleTable | count
27
28
```

Table 1 Done (0.171 s) 1 records

Name	Kind	Mapping
Mapping01	JSON	[{"column":"Timestamp","path":"\$.header.time","datatype":"null","transform":"0"}, {"column":"ApiVersion","path":"\$.header.apiVersion","datatype":"null","transform":"0"}, {"column":"RawHeader","path":"\$.header.rawHeader","datatype":"dynamic","transform":"0"}]

1. Create table mapping

```
// Create a json ingestion mapping
.create table SampleTable ingestion json mapping
```

```
"Mapping01" ' [{"column": "Timestamp", "path": "$.header.time"},
{"column": "ApiVersion", "path": "$.header.api_version"},
{"column": "RawHeader", "path": "$.header"}, {"column": "User", "path": "$.payload.user"} ]'
```

2. View ingestion mapping

```
// View ingestion mappings
.show table SampleTable ingestion json mappings
```

3. Ingestion from public blob // Ingest from public blob

```
.ingest into table SampleTable
@'https://westuskustopublic.blob.core.windows.net/public/SampleData-500-4394582f-668f-4d03-8bba-58f87a7e48a0.json' with
(jsonMappingReference = "Mapping01")
```

Exploration

Questions

If you skipped the previous step(s), please click on database: **text**.

1. How many lines were ingested?
2. Add calculated column of *Transaction Id* from column *RawHeader: attribute Id*
3. Take a 10 row sample of *RawHeader*
4. How many records were ingested from version 1 and 2?
5. Create a time chart with 10 minute bins of `RawHeader['time']`
6. Drop table `SampleTable`
7. Run `.show queries`

Kusto Query Language (KQL)

- ... | **count**
Counts records in input table (e.g. T)
- ... | **take 10**
Get few records to become familiar with the data. No order ensured.
- ... | **where** Timestamp > ago(1) and UserId = 'abdcdef'
Filters on specific fields
- ... | **project** Col1, Col2, ...
Select some columns (use if input table has many columns)

- ... | **extend** NewCol1=Col1+Col2
Introduces new calculated columns
 - ... | **render** timechart
Plots the data (in KE and KWE) while exploring
 - ... | **summarize** count(), dcount(Id) by Col1, Col2 *Analytics: aggregations*
 - ... | **top** 10 by count_desc
Finds the needle in the haystack
 - ... | **join** (...) on Key1, Key2 *Joins data sets*
 - ... | **mvexpand** Col1,Col2 ...
Turns dynamic arrays to rows (multi-value expansion)
 - ... | **parse** Col1 with <pattern>... *Deals with unstructured data*
-

Results

1. SampleTable | **count**
2. SampleTable | **extend** TransactionId = RawHeader.id
3. SampleTable
| **extend** TransactionId = RawHeader.id
| **take** 10
4. Option 1
 - a. SampleTable
| **extend** recordversion = tostring(RawHeader.api_version)
| **summarize** count() by recordversion b. SampleTable
| **where** tostring(RawHeader.api_version) **has** "1"
5. SampleTable
| **extend** x = todatetime(RawHeader['time'])
| **summarize** count() by bin(x, 10m)
| **render** timechart
6. Drop table // Drop the table **.drop** table SampleTable
7. Show queries **.show** queries

Self-Study

Kusto Query Language (KQL)

We'll use GitHub public data to query using Azure Data Explorer (Kusto) and visualize using Power BI.

1. Open the Browser and connect with the temporary Lab user credentials.

Azure Credentials

Here are your credentials to login to Microsoft Azure and access the On Demand Lab

Username



Password



2. Open <https://dataexplorer.azure.com/clusters/demo12.westus/databases/GitHub>
 - Cluster URL: <http://demo12.westus.kusto.windows.net>
 - Database: GitHub

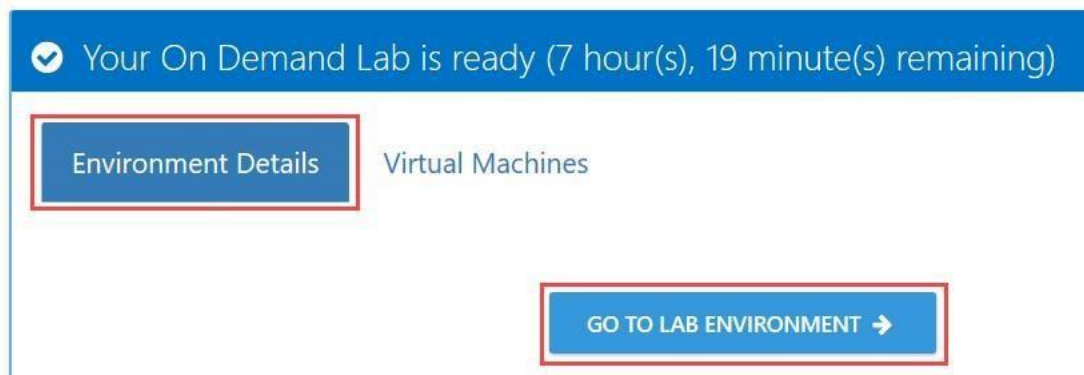
-
1. What was the date yesterday?
 2. How many events were in the last 60 days?
 3. Take a sample of 10 rows of your data?
 4. What is the number of *Repos* overall?
 5. What is the number of unique *Repos* values?
 6. What is the number of unique *Repos* names?
 7. Linus Torvalds, actor on GitHub is 'torvalds' (*Actor.display_name*). What are the top 3 event *Types* to which he contributed?
 8. How many Torvalds are there? How many events did they produce?
 9. What are the top 10 most watched *Repos*?
 10. (**) Plot the history of all of the events for the past 2 years for *Repos* from #9.
-

Power BI

Power BI is used to visualize the data. Note that Power BI is a visualization tool with data size limitations. Default: 500,000 records and 700MB data size.

PBI demo script

1. Open Lab **Environment Details** page: <http://bit.ly/2WCFDdz>
2. Select **GO TO LAB ENVIRONMENT->**



Connect to Help cluster

1. Connect with the Lab **Azure Credentials**:

Azure Credentials

Here are your credentials to login to Microsoft Azure and access the On Demand Lab

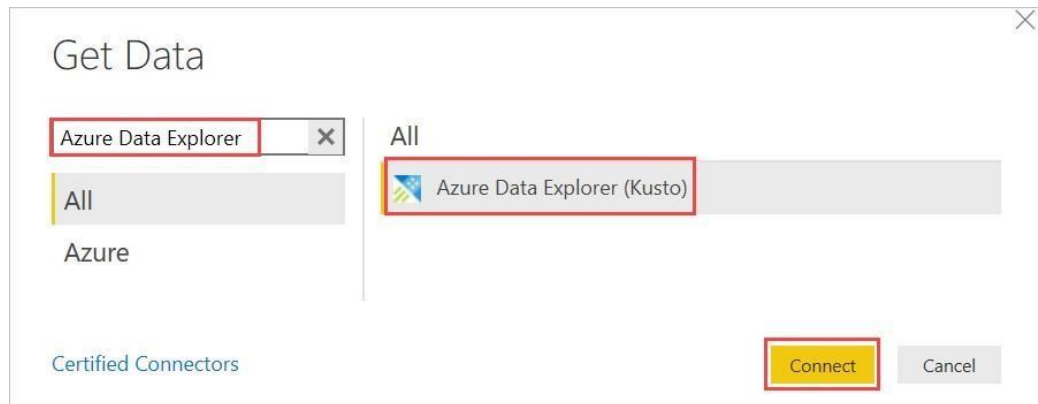
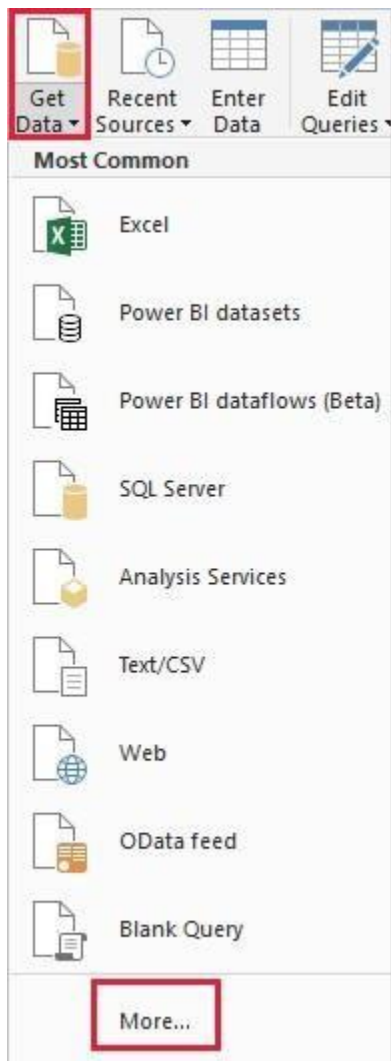
Username



Password



2. Open Power BI desktop, select **Get Data**, and **More...** Type **Data Explorer** in the search box.
4. Enter the following properties (leave all other fields empty) and then select **OK**:
Cluster: **Help**
Database: **Samples**
Table name or Azure Data Explorer query: **StormEvents**



3. Select **Azure Data Explorer (Kusto)** and **Connect**

Data Connectivity mode: **Import**

Kusto

Cluster

Help

Database (optional)

Samples

Table name or Azure Data Explorer query (optional)

StormEvents

Advanced options (optional)

Limit query result record number (optional)

Example: 500000

Limit query result data size in Bytes (optional)

Example: 67108864

Disable result-set truncation (optional)

Example: false

Data Connectivity mode ⓘ

☒ Import

☐ DirectQuery

OK

Cancel

- Expand the **Samples** database and select **StormEvents**. If the table looks ok, select **Load**. To make changes, select **Edit**.

Help: Samples: StormEvents

StartTime	EndTime	EpisodeId	EventId	State	EventType	InjuriesDirect	Injuries
9/29/2007 8:11:00 AM +00:00	9/29/2007 8:11:00 AM +00:00	11091	61032	ATLANTIC SOUTH	Waterspout	0	
9/18/2007 8:00:00 PM +00:00	9/19/2007 6:00:00 PM +00:00	11074	60904	FLORIDA	Heavy Rain	0	
9/20/2007 9:57:00 PM +00:00	9/20/2007 10:05:00 PM +00:00	11078	60913	FLORIDA	Tornado	0	
12/30/2007 4:00:00 PM +00:00	12/30/2007 4:05:00 PM +00:00	11749	64588	GEORGIA	Thunderstorm Wind	0	
12/20/2007 7:50:00 AM +00:00	12/20/2007 7:53:00 AM +00:00	12554	68796	MISSISSIPPI	Thunderstorm Wind	0	
12/20/2007 10:32:00 AM +00:00	12/20/2007 10:36:00 AM +00:00	12554	68814	MISSISSIPPI	Tornado	2	
12/20/2007 8:47:00 AM +00:00	12/20/2007 8:48:00 AM +00:00	12554	68834	MISSISSIPPI	Thunderstorm Wind	0	
12/28/2007 2:03:00 AM +00:00	12/28/2007 2:11:00 AM +00:00	12561	68846	MISSISSIPPI	Hail	0	
12/7/2007 2:00:00 PM +00:00	12/8/2007 4:00:00 AM +00:00	13183	73241	AMERICAN SAMOA	Flash Flood	0	
12/13/2007 9:02:00 AM +00:00	12/13/2007 10:30:00 AM +00:00	11780	64725	KENTUCKY	Flood	0	
12/23/2007 6:02:00 AM +00:00	12/23/2007 6:07:00 AM +00:00	11781	64726	OHIO	Thunderstorm Wind	0	
12/23/2007 6:38:00 AM +00:00	12/23/2007 6:43:00 AM +00:00	11781	64727	OHIO	Thunderstorm Wind	0	
12/23/2007 6:36:00 AM +00:00	12/23/2007 6:41:00 AM +00:00	11781	64728	OHIO	Thunderstorm Wind	0	
12/23/2007 7:14:00 AM +00:00	12/23/2007 7:19:00 AM +00:00	11781	64729	OHIO	Thunderstorm Wind	0	
12/11/2007 9:45:00 PM +00:00	12/12/2007 4:45:00 PM +00:00	12826	70787	KANSAS	Flood	0	
12/28/2007 2:47:00 AM +00:00	12/28/2007 3:05:00 AM +00:00	12561	68867	MISSISSIPPI	Hail	0	
12/28/2007 3:05:00 AM +00:00	12/28/2007 3:16:00 AM +00:00	12561	68868	MISSISSIPPI	Hail	0	
12/10/2007 1:10:00 PM +00:00	12/13/2007 2:30:00 AM +00:00	12068	65995	KENTUCKY	Flood	0	
12/15/2007 1:00:00 PM +00:00	12/15/2007 3:00:00 PM +00:00	11895	65282	KENTUCKY	Flood	0	
12/15/2007 12:00:00 PM +00:00	12/15/2007 6:00:00 PM +00:00	11895	65283	KENTUCKY	Flood	0	

ⓘ The data in the preview has been truncated due to size limits.

Load

Edit

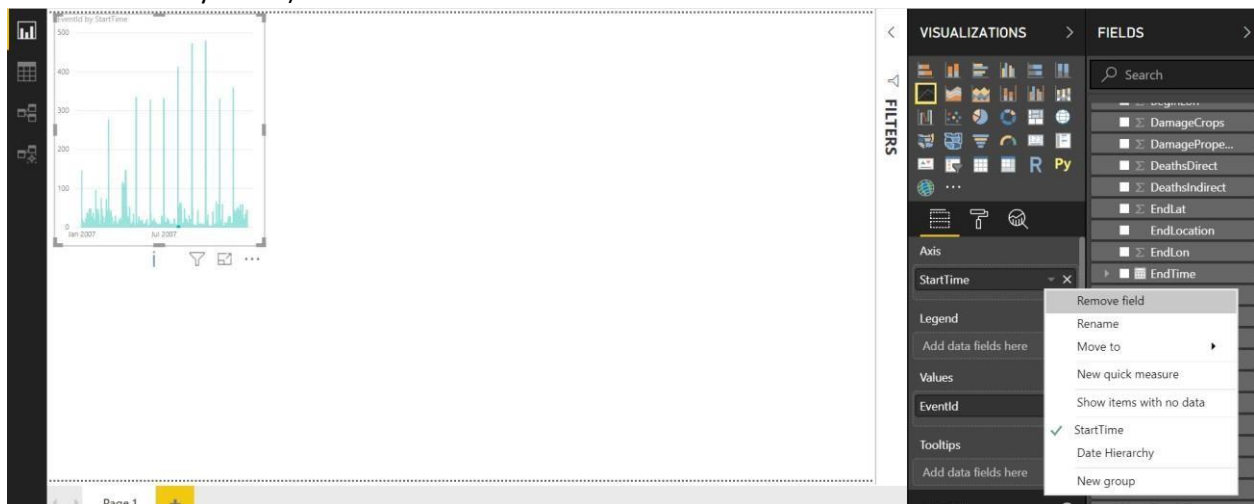
Cancel

- The new **StormEvents** table was added to the Power BI report.

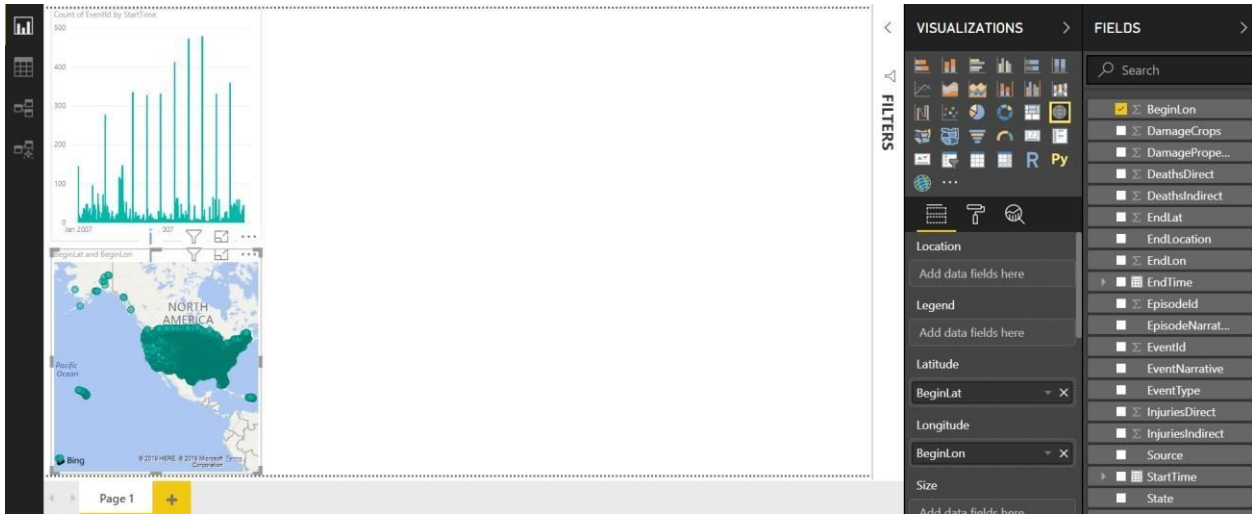


Create a Power BI report

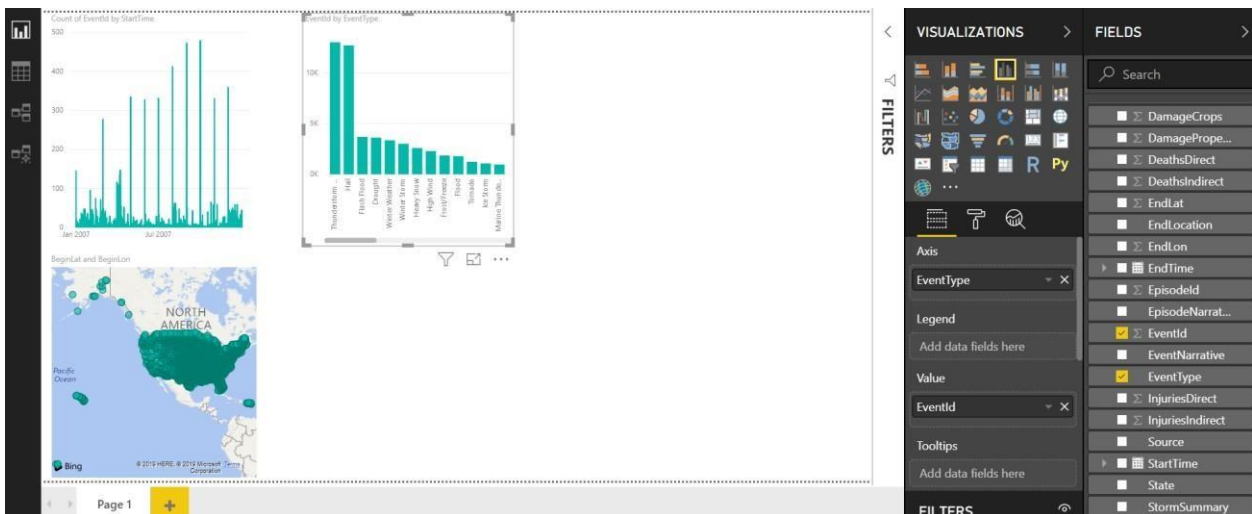
1. Create a line chart with the total number of events, by putting “Start Time” in the Axis box (not in Date Hierarchy mode) and EventId in the Values box.



2. Add a Map tile by putting “BeginLat” in the Latitude box and putting “BeginLon” in the Longitude box.



3. Create a Clustered column chart by putting “Event Type” in the Axis box and (count) “Event Id” in the value box.



4. Create 4 separate card tiles with “DeathDirect”, “DeathIndirect”, “InjuriesDirect” and “InjuriesIndirect in the Fields box.



5. Create a pie chart of reporting sources by putting the “Source” in the legend box and putting the (count) “EventId” in the values box.

6. Now arrange the tiles on the canvas and you’re ready to slice and dice.



3 Power BI Connectors

1. Native Connector for Power BI

- Native Connector 7 data explorer 7 Connect 7 Preview Feature (accept) continue.
- Cluster: demo12.westus
- Database: GitHub
- Table: GithubEvent
- Import 7 load data in advanced
- Seamless browsing experience
- Data size limitation

- **(Click) Direct Query** ➤ load data per request
 - Load per request
 - Longer response time
 - Sign-in ➤ connect
 - Data sample ➤ load
 - Drag ID from the Fields on the right side of the screen
 - Drag CreatedAt
 - Drag CreatedAt into ID square
2. Blank Query for Power BI
 - Get Data ➤ Blank Query
 - Kusto Explorer ➤ Tools ➤ Query to Power BI (Query & PBI adaptor)
 - Connect - Organization account ➤ use your account
 - Click on ➤ Advanced editor ➤ Delete everything ➤ Paste everything
 3. MS-TDS (SQL) client for Power BI
 - (ODBS Connector) End Point: Azure ➤ Azure SQL database
 - Kusto Cluster as destination <https://docs.microsoft.com/enus/azure/dataexplorer/power-bi-sql-query>

KQL – Results

4. How many events are there in *Repos* that have 'Azure' in their name?

```
// 1. What is count of events in Repos that have 'Azure' word in their name?
GithubEvent
| where Repo has 'Azure'
| count
```

5. What is the total number of *Repos*?

```
// 2. What is the amount of the Repos overall?
GithubEvent
| summarize dcount(tostring(Repo))
```

6. Linus Torvalds Actor on GitHub is 'torvalds' (*Actor.display_login*). What are the top 3 event *Types* he contributes to?

```
// 3. Linus Torvalds Actor on GitHub is 'torvalds' (Actor.display_name). What are
top 3 events Types he contributes to?
GithubEvent
| where Actor.display_login == 'torvalds'
| summarize count() by Type
| top 3 by count_
```


7. How many Torvalds are there and how many events they produce?

```
// 4. How many Torvalds are there and how much events they produce?
//
GithubEvent
| where Actor has 'torvalds'
| summarize count() by name=tostring(Actor.display_login)
| where isnoteempty(name)
```

8. What are the top 10 most watched Repos?

```
//5. What are the top 10 most watched Repos?
GithubEvent
| where Type == 'WatchEvent'
| summarize count() by tostring(Repo.name)
| top 10 by count_
```

9. Plot the history of all events for Repos which are the answer for #5.

10. // 6. Plot the history of all events for the past 2 years for Repos coming out #5.

```
let repos = GithubEvent |
where Type == 'WatchEvent'
| summarize count() by name=tostring(Repo.name)
| top 10 by count_
| project name; GithubEvent
| where Repo.name in (repos)
| extend repo = tostring(Repo.name)
| summarize count() by bin(CreatedAt, 1d), repo
| render timechart
```

11. Show the top 10 repos by WatchEvent, along with their WatchEvent count and their total events count (hint: use join)

```
// 7. Show top 10 repos with most Watch Event and their total count of events
(hint: use join)
GithubEvent
| where Type == 'WatchEvent'
| summarize WatchCounts = count() by name=tostring(Repo.name)
| top 10 by WatchCounts
| join hint.strategy = broadcast
(
    GithubEvent
    | extend repo = tostring(Repo.name)
    | summarize TotalEvents=count() by repo
) on $left.name == $right.repo
| project repo, TotalEvents, WatchCounts
| order by TotalEvents
```