Thank you to our sponsors!

Gold Sponsors

INSPARK

betabit

codit

Motion10

macaw

Silver Sponsors

WINVISION

aims

Community Sponsors

Microsoft Integration User Group The Netherlands

AZURE IOT COMMUNITY

WAZUG.NL AZURE USER GROUP NL

Experts Live Netherlands

betabit
samen bijzonder maken;

An intro to

Azure Data Lake

Rick van den Bosch
M +31 (0)6 52 34 89 30
r.van.den.bosch@betabit.nl

Calendar

Data Lakes

About Azure Data Lake

Azure Data Lake Store

- **DEMO**

Azure Data Lake HDInsights

- **DEMO**

Azure Data Lake Analytics

- **DEMO**

Power BI

- **DEMO**

Resources

betabit {...}

samen bijzonder maken;

betabit {...}

Rick van den Bosch

Cloud Solutions Architect

@rickvdbosch

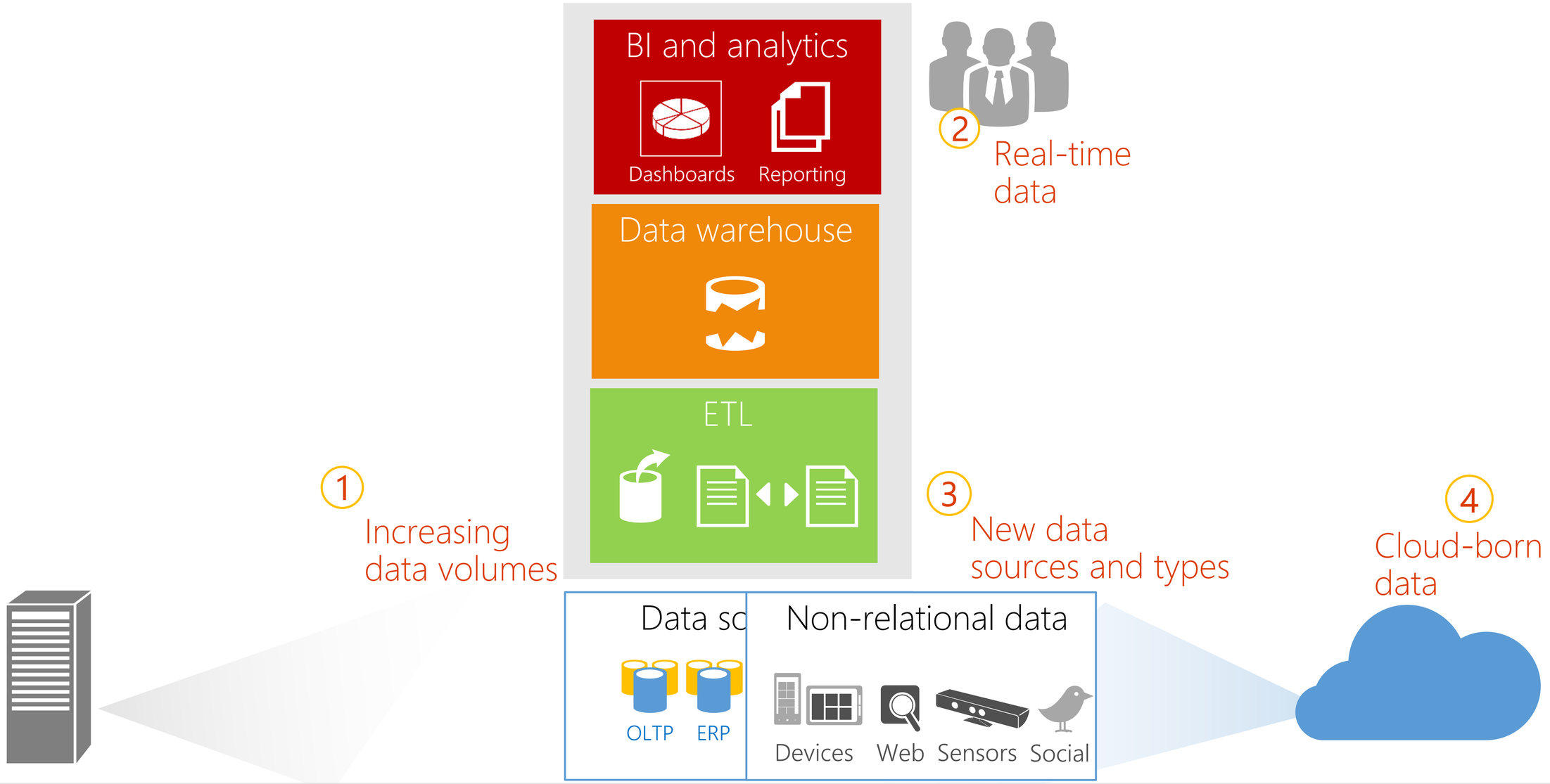
rickvandenbosch.net

r.van.den.bosch@betabit.nl

Data Lakes

betabit

The Traditional Data Warehouse



The Data Lake Approach

Ingest all data
regardless of requirements

Store all data
in native format without schema definition

Do analysis
Hadoop, Spark, R, Azure Data Lake Analytics (ADLA)

Designed for the questions you don't yet know!

About Azure Data Lake

Azure Data Lake

- Store and analyze petabyte-size files and trillions of objects
- Develop massively parallel programs with simplicity
- Debug and optimize your big data programs with ease
- Enterprise-grade security, auditing, and support
- Start in seconds, scale instantly, pay per job
- Built on YARN, designed for the cloud

Azure Data Lake

Microsoft

No limits Data Lake

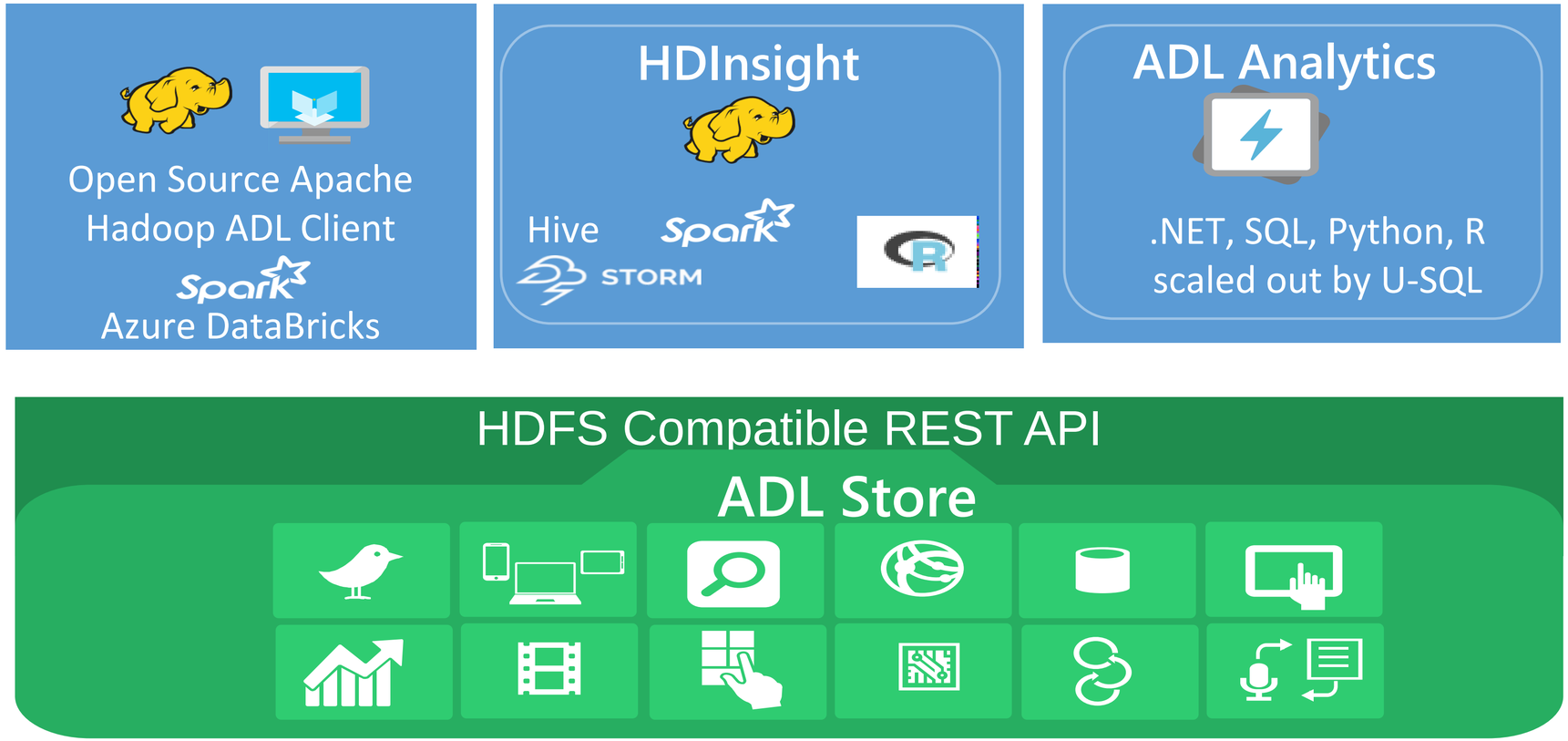
Analytics job service

Managed Clusters

Why Azure Data Lake?

an on-demand, real-time stream processing service with no-limits data lake built to support massively parallel analytics

- Performance at scale
- Optimized for analytics
- Multiple analytics engines
- Single repository sharing



Azure Data Lake *Store*

betabit

Store

- Enterprise-wide hyper-scale repository
- Data of any size, type and ingestion speed
- Operational and exploratory analytics

- WebHDFS-compatible API
- Specifically designed to enable analytics
- Tuned for (data analytics scenario) performance

- Out of the box:
  security, manageability, scalability, reliability, and availability

Store

Architected and built for very high throughput at scale for Big Data workloads

- No limits to file size, account size or number of files

Single-repository for sharing

- Cloud-scale distributed filesystem with file/folder ACLS and RBAC
- Encryption-at-rest by default with Azure Key Vault
- Authenticated access with Azure Active Directory integration

The Big Data platform for Microsoft

Key capabilities

Built for Hadoop

Unlimited storage, petabyte files

Performance-tuned for big data analytics

Enterprise-ready: Highly-available and secure

All data

betabit

Security

Authentication

- Azure Active Directory integration
- Oauth 2.0 support for REST interface

Access control

- Supports POSIX-style permissions (exposed by WebHDFS)
- ACLs on root, subfolders and individual files

Encryption

Compatibility

Open Source Software	Distribution
Apache Sqoop	HDInsight 3.2, 3.4, 3.5, and 3.6
MapReduce	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Storm	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Hive	HDInsight 3.2, 3.4, 3.5, and 3.6
HCatalog	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Mahout	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Pig/Pig Latin	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Oozie	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Zookeeper	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Tez	HDInsight 3.2, 3.4, 3.5, and 3.6
Apache Spark	HDInsight 3.4, 3.5, and 3.6

Store

DEMO - Store

Ingest data – Ad hoc

Local computer

- Azure Portal
- Azure PowerShell
- Azure CLI
- Using Data Lake Tools for Visual Studio

Azure Storage Blob

- Azure Data Factory
- AdlCopy tool
- DistCp running on HDInsight cluster

Ingest data

Streamed

- Azure Stream Analytics
- Azure HDInsight Storm
- EventProcessorHost

Relational

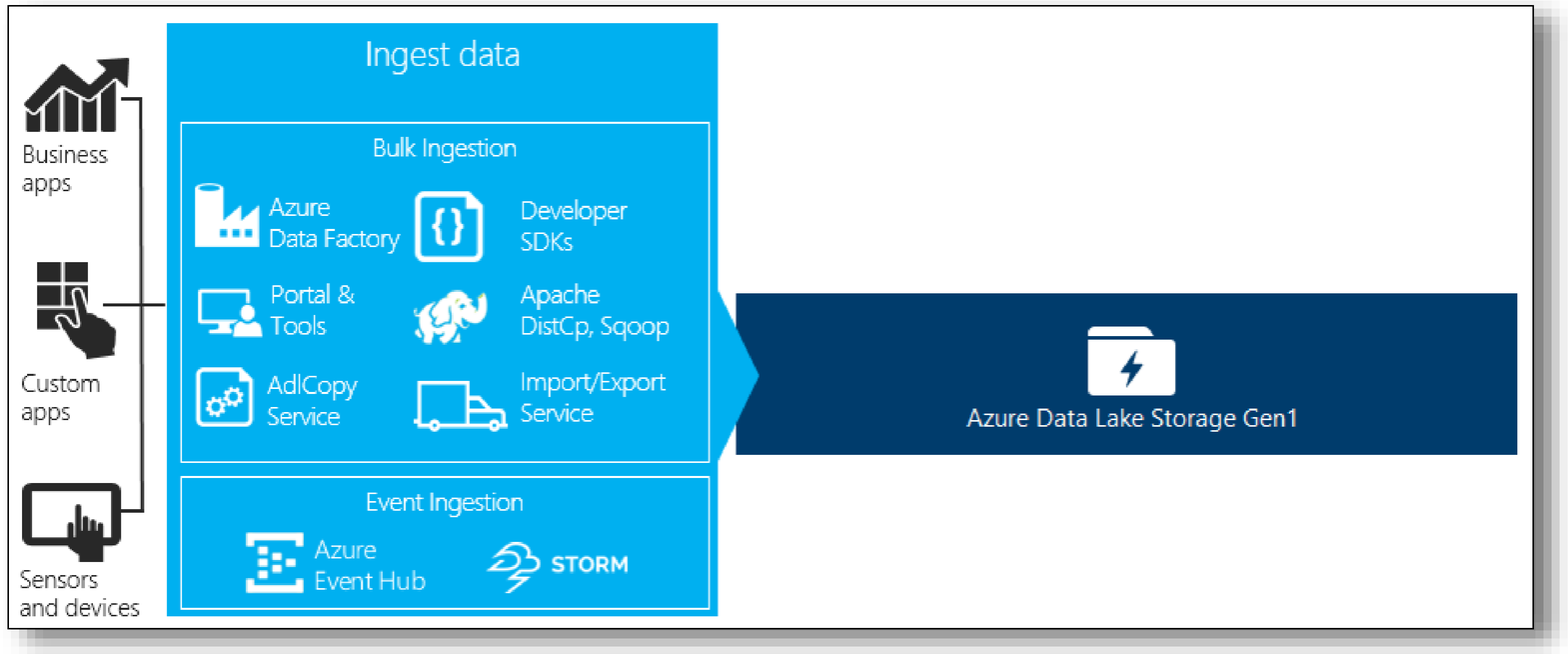
- Apache Sqoop
- Azure Data Factory

Web server

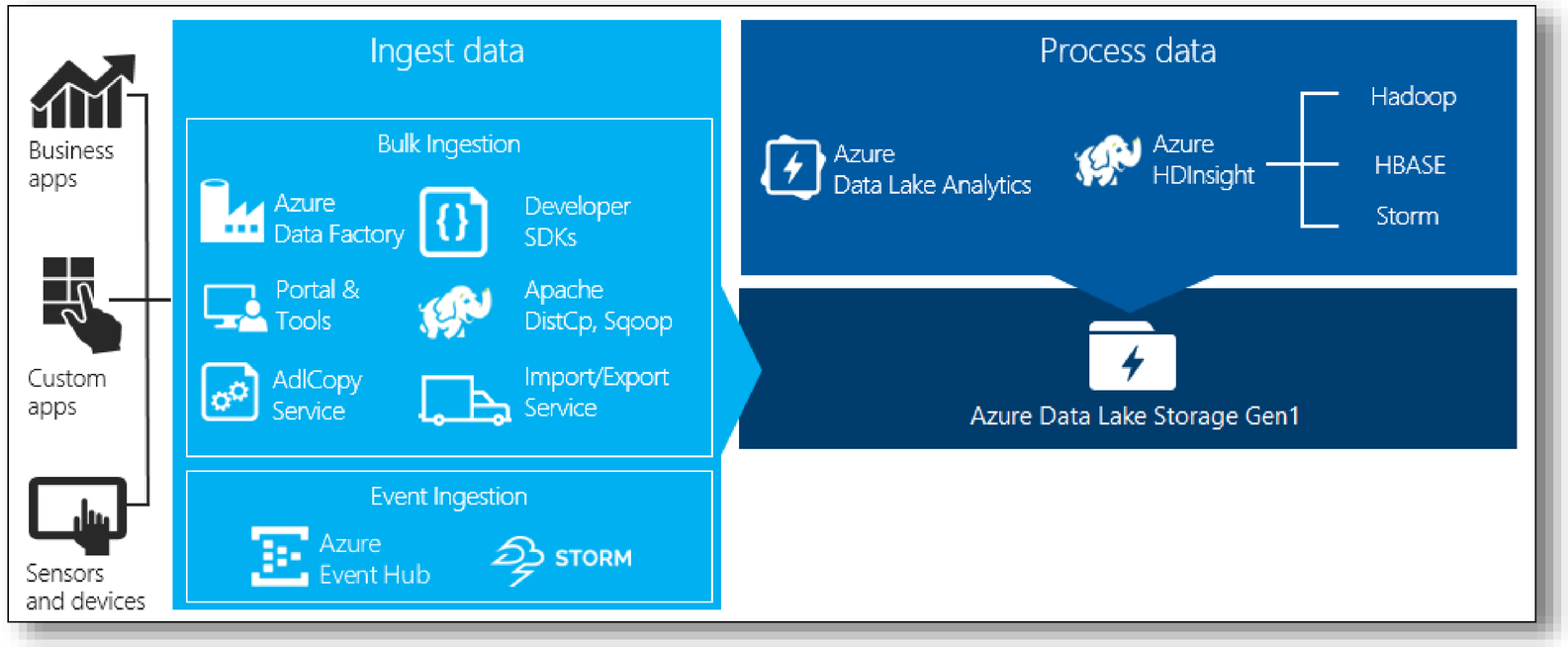
Upload using custom applications

- Azure CLI
- Azure PowerShell
- Azure Data Lake Storage Gen1 .NET SDK
- Azure Data Factory

Ingest data

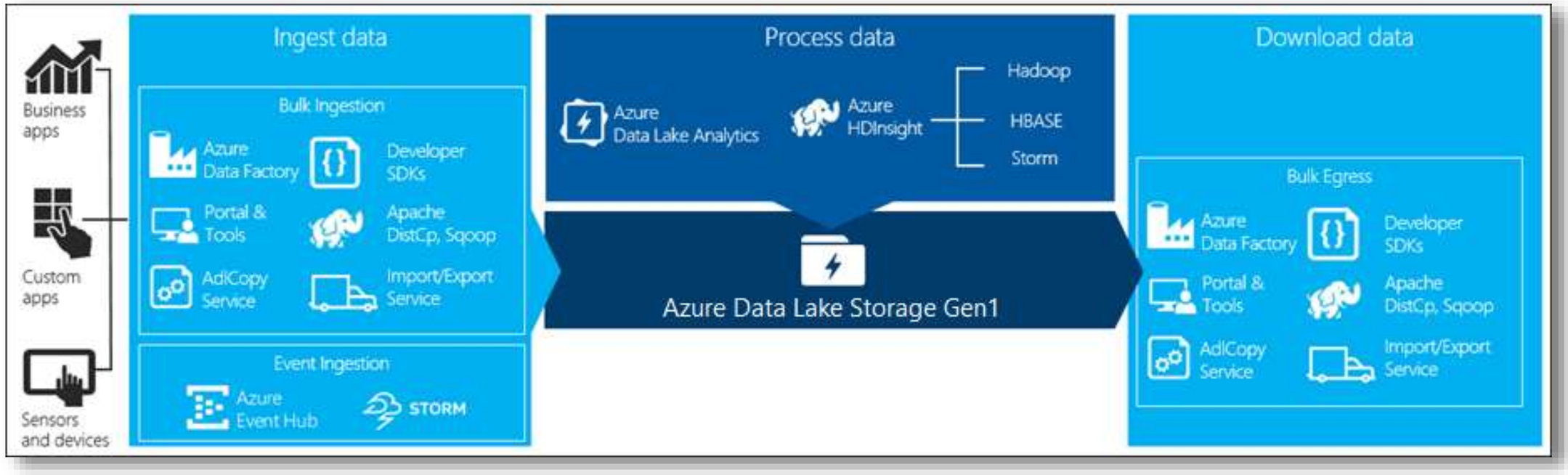


Process data

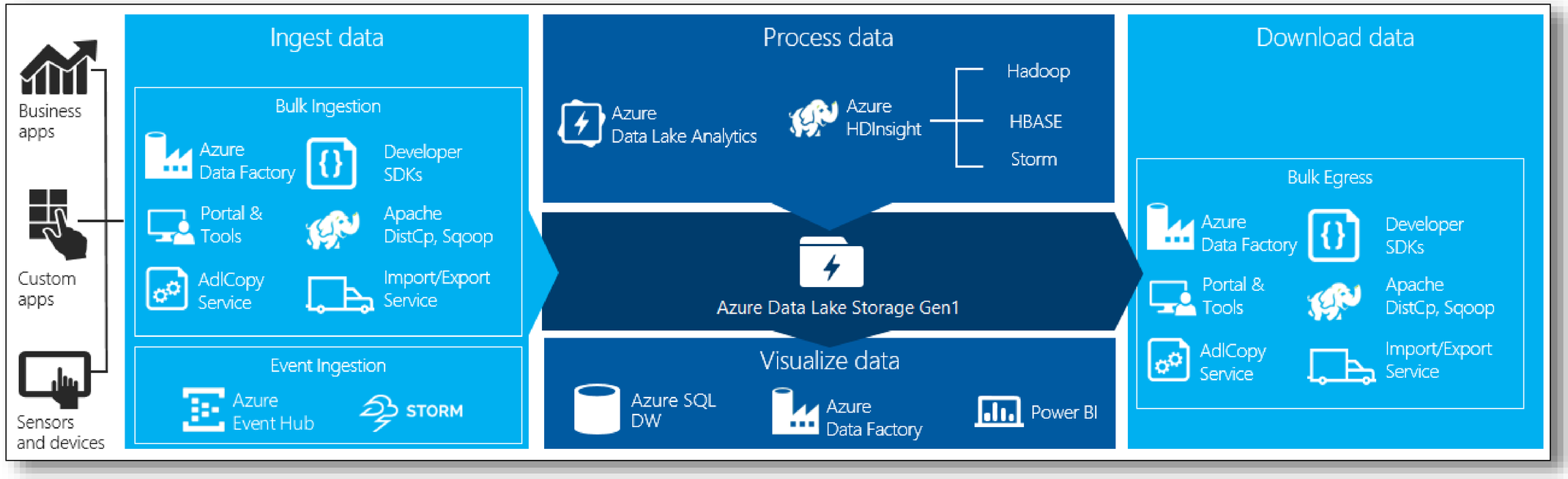


betabit

Download data



Visualize data



ADLS Gen 2

Takes core capabilities from Azure Data Lake Storage Gen1 such as

- a Hadoop compatible file system
- Azure Active Directory
- POSIX based ACLs

and integrates them into Azure Blob Storage

Additional benefits

Unlimited scale and performance

Performance improvements reading/writing individual objects (> throughput & concurrency)

Removes need to decide a priority: run analytics or not at data ingestion time

Data protection capabilities: encryption at rest

Integrated network Firewall capabilities

Durability options (Zone and Geo-Redundant Storage: high-availability and disaster recovery)

Linux integration – BlobFUSE

- mount Blob Storage from Linux VMs
- interact using standard Linux shell commands.

Data Lake Storage Gen2

"In Data Lake Storage Gen2, all the qualities of object storage remain while adding the advantages of a file system interface optimized for analytics workloads."

Known issues

Blob Storage APIs and Azure Data Lake Gen2 APIs aren't interoperable

Blob storage APIs not available

Azure Storage Explorer >= 1.6.0

AZCopy >= v10

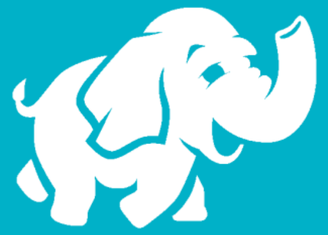
Event Grid doesn't receive events

Soft Delete and Snapshots not available

Object level storage tiers not available

Diagnostic logs not available

Azure Data Lake *HDInsight*



betabit

HDInsight

Cloud distribution of the (Hortonworks) Hadoop components

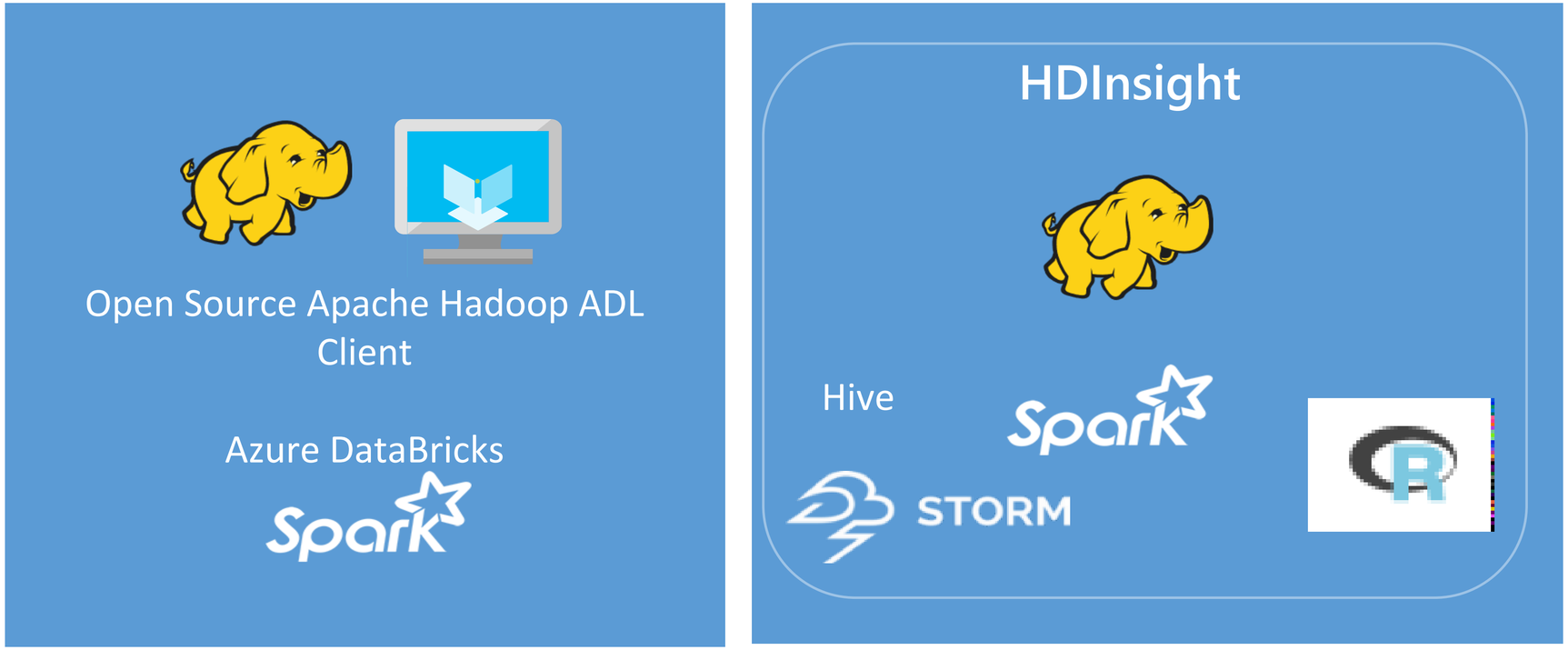
Supports multiple Hadoop cluster versions (can be deployed any time)

Hadoop

- YARN for job scheduling & resource management
- MapReduce for parallel processing
- HDFS

Component	HDInsight 4.0 (Preview)	HDInsight 3.6 (Default)	HDInsight 3.5	HDInsight 3.4	HDInsight 3.3	HDInsight 3.2	HDInsight 3.1	HDInsight 3.0
Hortonworks Data Platform	3.0	2.6	2.5	2.4	2.3	2.2	2.1.7	2.0
Apache Hadoop and YARN	3.1.1	2.7.3	2.7.3	2.7.1	2.7.1	2.6.0	2.4.0	2.2.0
Apache Tez	0.9.1	0.7.0	0.7.0	0.7.0	0.7.0	0.5.2	0.4.0	-
Apache Pig	0.16.0	0.16.0	0.16.0	0.15.0	0.15.0	0.14.0	0.12.1	0.12.0
Apache Hive and HCatalog	-	1.2.1	1.2.1	1.2.1	1.2.1	0.14.0	0.13.1	0.12.0
Apache Hive	3.1.0	2.1.0	-	-	-	-	-	-
Apache Tez Hive2	-	0.8.4	-	-	-	-	-	-

HDInsight



DEMO - HDInsight

Azure Data Lake *Analytics*

betabit

Analytics

Dynamic scaling

Develop faster, debug and optimize smarter using familiar tools

U-SQL: simple and familiar, powerful, and extensible

Integrates seamlessly with your IT investments

Affordable and cost effective

Works with all your Azure data

Analytics

On-demand analytics job service to simplify big data analytics

Can handle jobs of any scale instantly

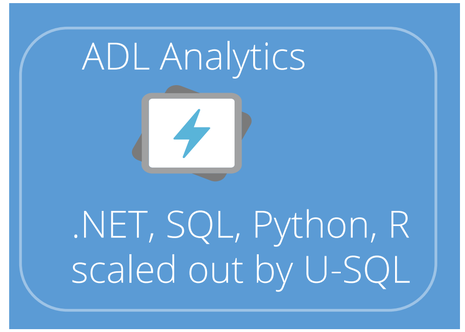
Azure Active Directory integration

U-SQL

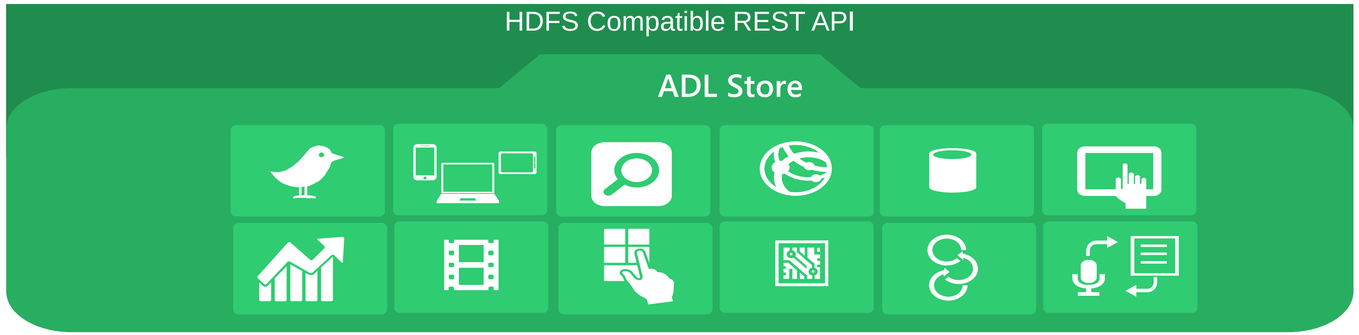
Azure Data Lake Analytics

Analytics

- Serverless. Pay per job. Starts in seconds. Scales instantly.
- Develop massively parallel programs with simplicity
- Federated query from multiple data sources



Storage



betabit

U-SQL

Language that combines declarative SQL with imperative C#

```
@searchlog =
    EXTRACT UserId          int,
            Start           DateTime,
            Region          string,
            Query           string,
            Duration        int?,
            Urls            string,
            ClickedUrls     string
    FROM "/Samples/Data/SearchLog.tsv"
    USING Extractors.Tsv();

OUTPUT @searchlog
    TO "/output/SearchLog-first-u-sql.csv"
    USING Outputters.Csv();
```

U-SQL – Key concepts

Rowset variables

- Each query expression that produces a rowset can be assigned to a variable.

EXTRACT

- Reads data from a file & defines the schema on read *

OUTPUT

- Writes data from a rowset to a file *

U-SQL – Scalar variables

```
DECLARE @in  string = "/Samples/Data/SearchLog.tsv";
DECLARE @out string = "/output/SearchLog-scalar-variables.csv";

@searchlog =
    EXTRACT     UserId          int,
                ClickedUrls     string
    FROM @in
    USING Extractors.Tsv();

OUTPUT @searchlog
    TO @out
    USING Outputters.Csv();
```

U-SQL – Transform rowsets

```
@searchlog =
    EXTRACT UserId    int,
            Region      string
    FROM "/Samples/Data/SearchLog.tsv"
    USING Extractors.Tsv();


@rs1 =
    SELECT UserId, Region
    FROM @searchlog
WHERE Region == "en-gb";


OUTPUT @rs1
    TO "/output/SearchLog-transform-rowsets.csv"
    USING Outputters.Csv();
```

U-SQL – Extractor parameters

delimiter

encoding

escapeCharacter

nullEscape

quoting

rowDelimiter

silent

skipFirstNRows

charFormat

betabit

U-SQL – Outputter parameters

delimiter

dateTimeFormat

encoding

escapeCharacter

nullEscape

quoting

rowDelimeter

charFormat

outputHeader

betabit

U-SQL

Built-in extractors and outputters:

Text

Csv

Tsv

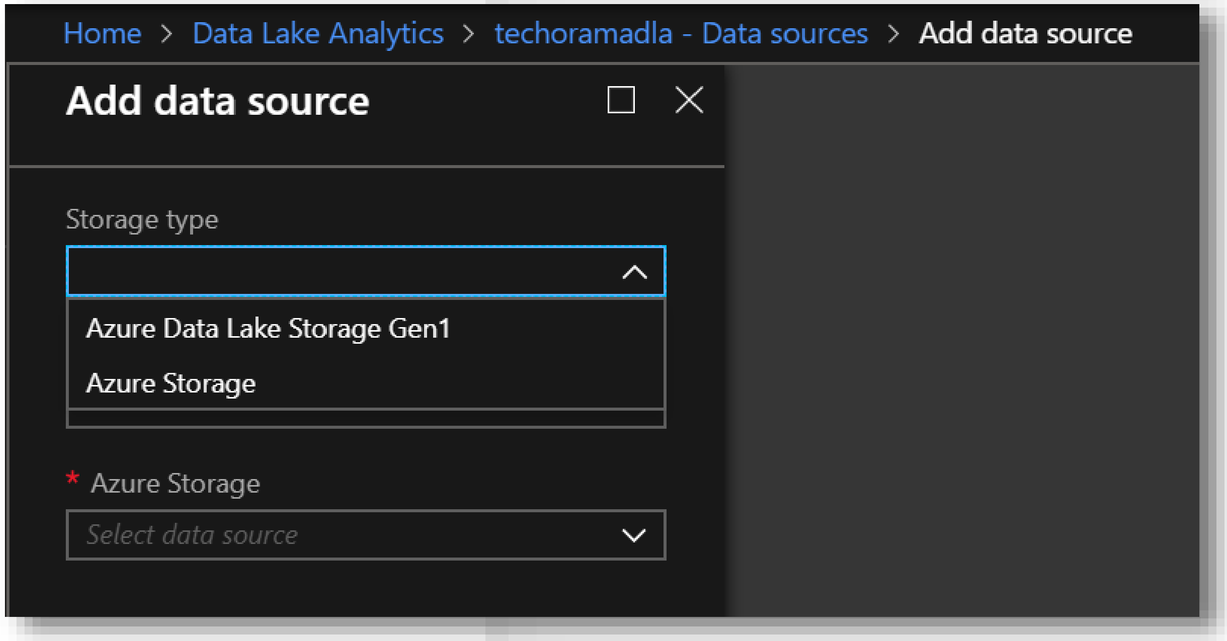
A (for instance) CSV Extractor or Outputter is
EXACTLY THAT

betabit

Data sources

Options in the Azure Portal:

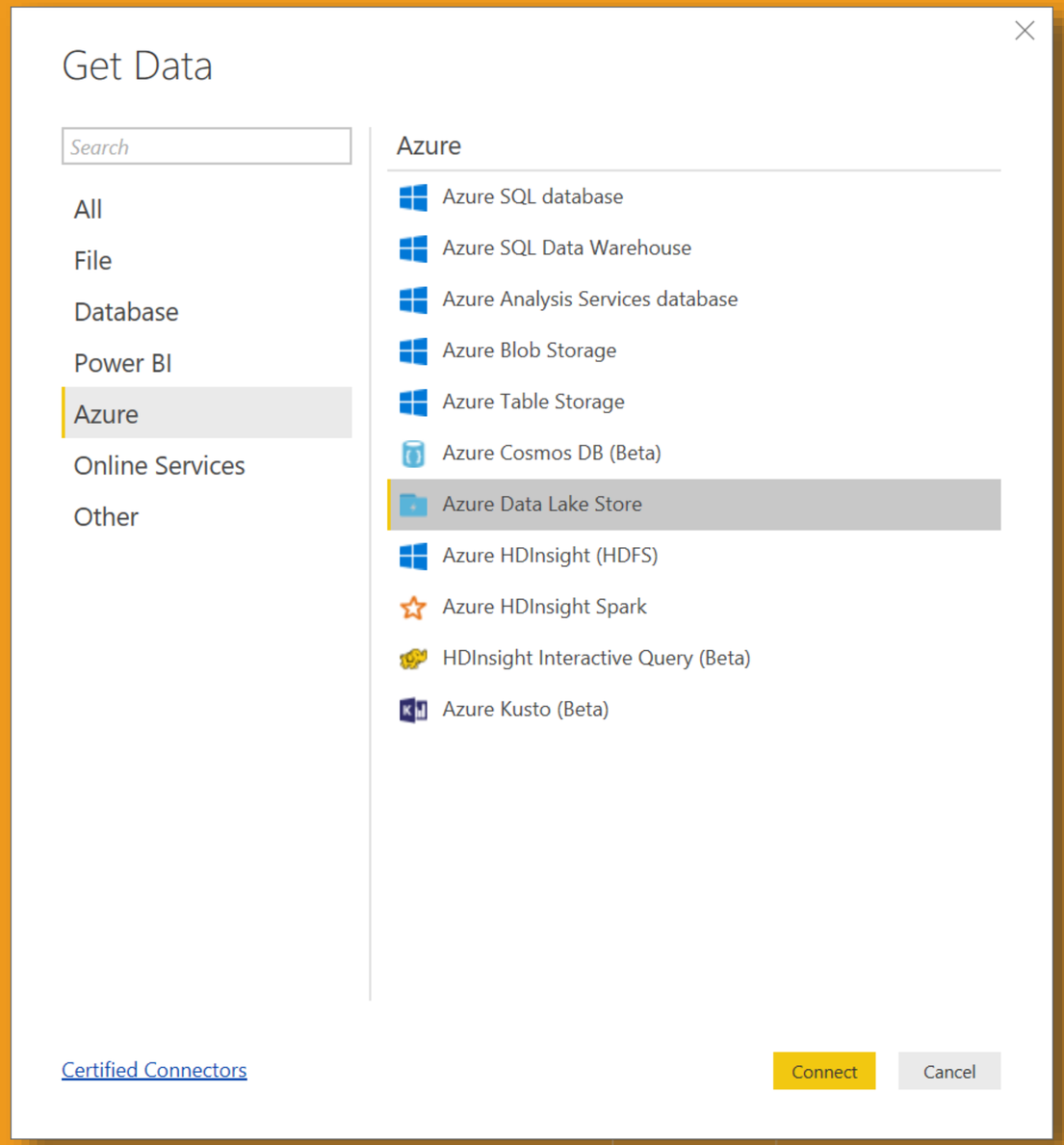
- Data Lake Storage Gen1
- Azure Storage



betabit

DEMO - Analytics

DEMO - Power BI



betabit

Resources

betabit

Resources

Basic example

Advanced example

Create Database (U-SQL) & Create Data Source (U-SQL)

This example

HDInsight quickstart

Azure blog

Azure roadmap

betabit

samen bijzonder maken;

Bedankt voor je aandacht

Track 1

15:35 – 16:20

Skynet Is Talking - Microsoft Bot Framework

Kris van der Mast

Track 2

15:35 – 16:20

Enter The Matrix: Securing Azure's Assets

Mike Martin