PolyBase in SQL Server 2016

A big data story

PASS

PASS SQL saturday

Nico Jacobs

- Researcher @KULeuven till 2004 (Machine Learning)
- Trainer @U2U since 2004
  - SQL Server
  - Business Intelligence
- Nico@u2u.be
- @SQLWaldorf

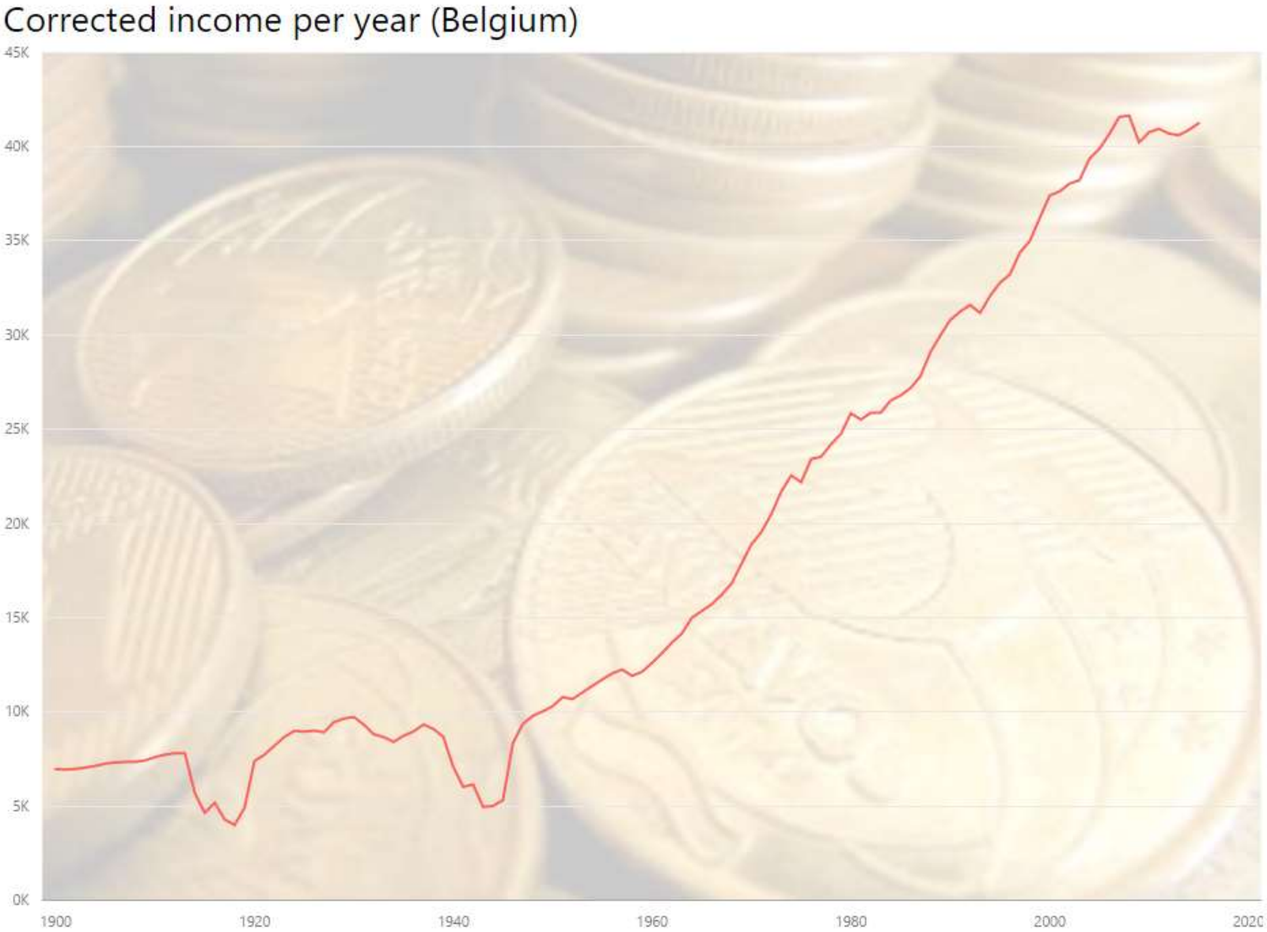
Agenda

- Why PolyBase?
- First things first: installing and configuring
- External Data Sources
- External File Formats
- External Tables
- Compute here or compute there
- Conclusions

WHY POLYBASE?

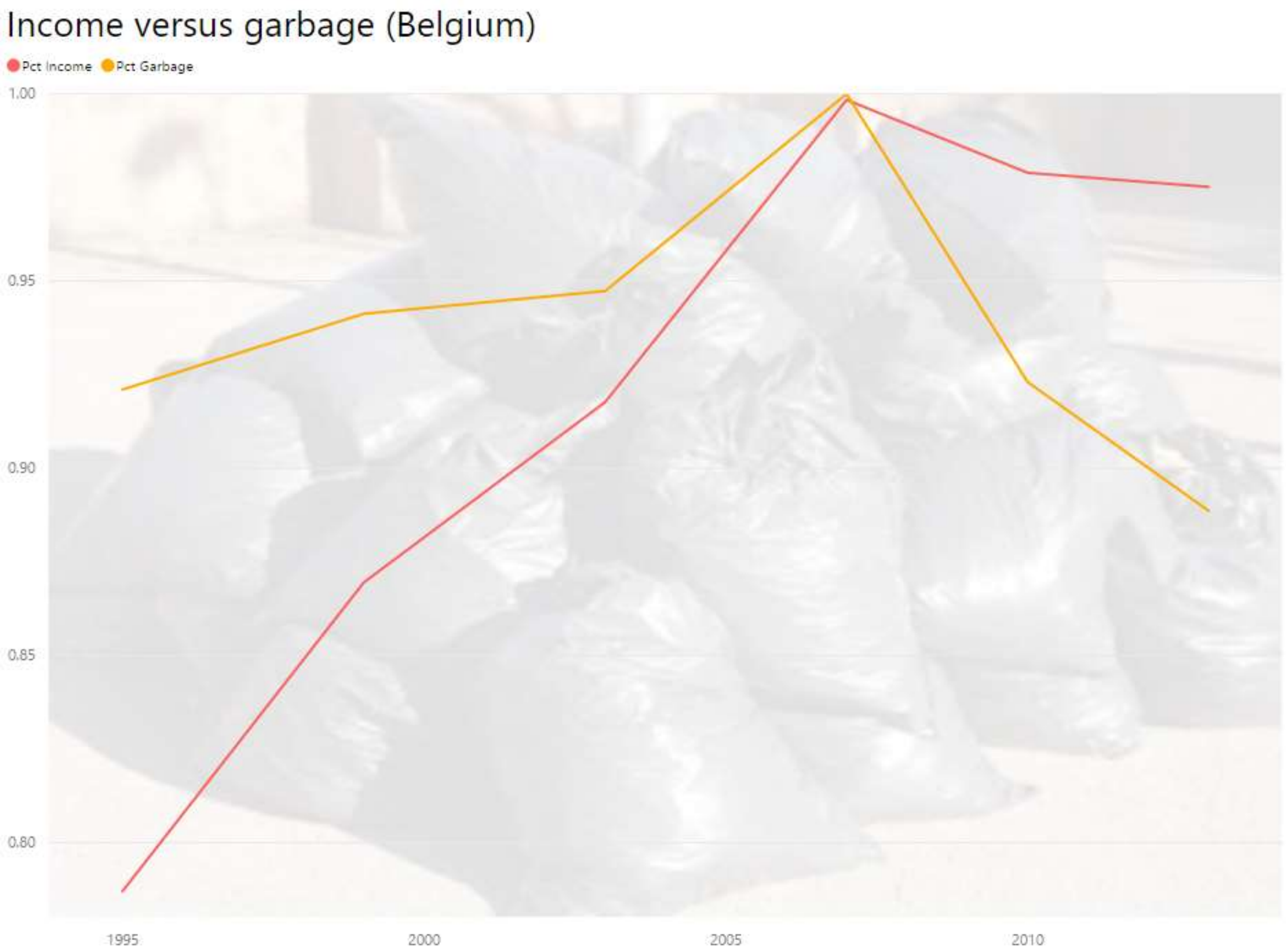
Let’s first talk about… controlling waste!

Income



Corrected income per year (Belgium)
45K
40K
35K
30K
25K
20K
15K
10K
5K
0K
1900
1920
1940
1960
1980
2000
2020

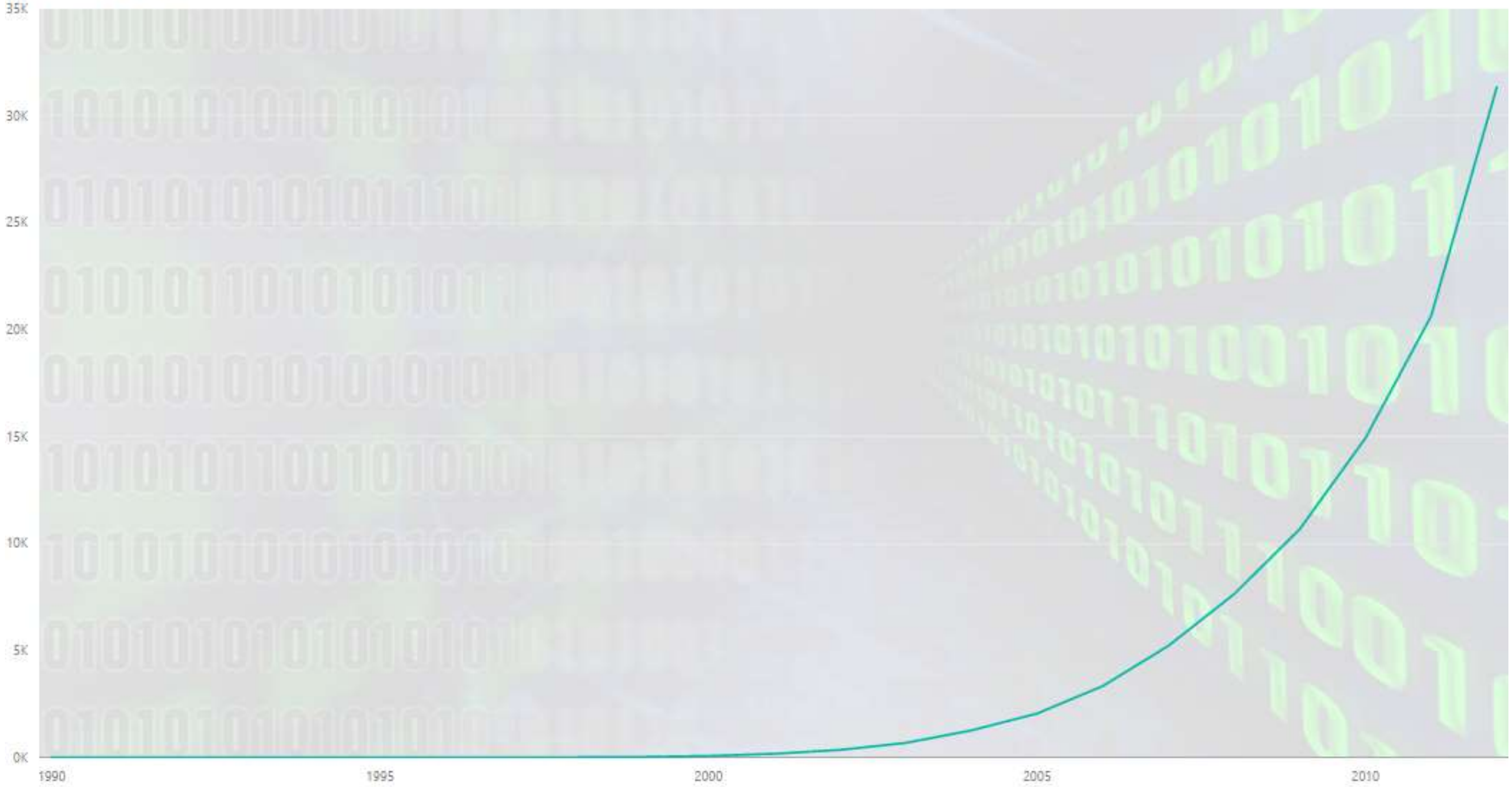
Garbage


Income versus garbage (Belgium)
● Pct Income
● Pct Garbage
1.00
0.95
0.90
0.85
0.80
1995
2000
2005
2010

Controlling waste

Data

Internet traffic



But what about wasting… data!

- Admin:
  - SQL Server error log keeps 7 files
  - Web logs, server logs, …
- Business:
  - Updates versus inserts
  - Social media, IoT, …

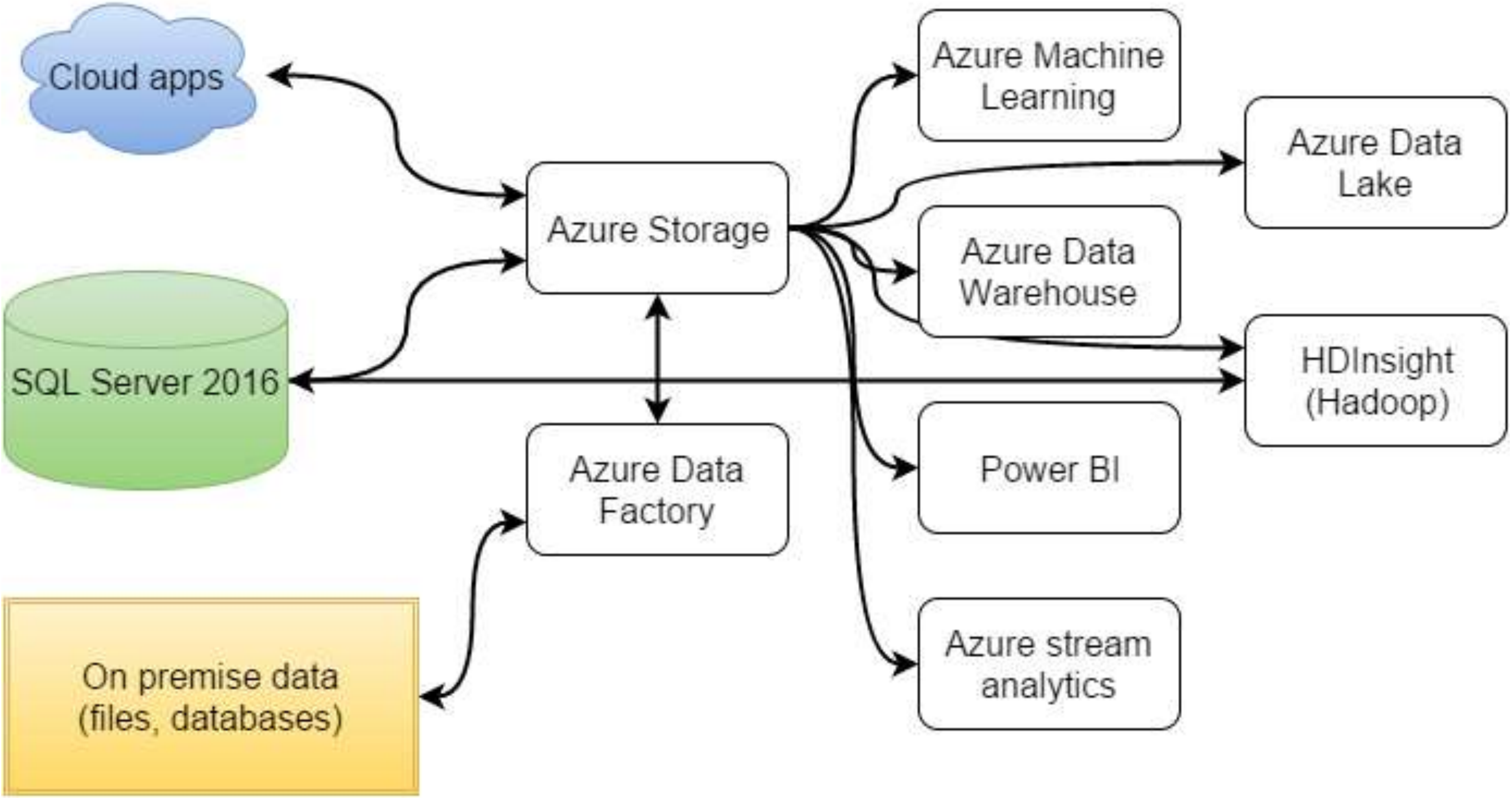
Why do we throw so much data away?

- No immediate analysis need
- A lot of work to setup
- Hard to predict needed storage capacity
- Data not well structured
- Structure changes

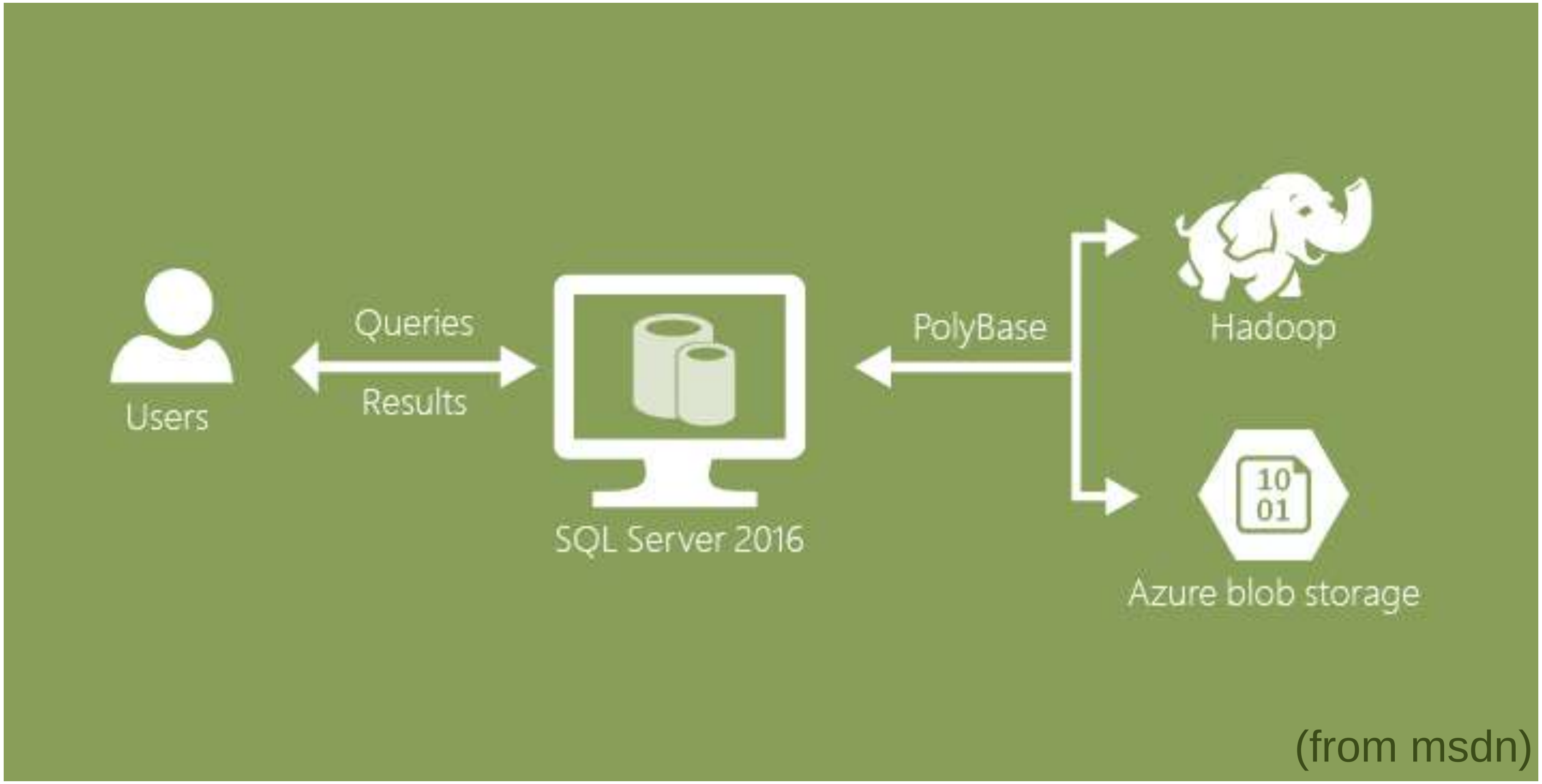
Big data: Start recycling your data

- Low cost: ± 21€/month/Tb
- Infinite storage at pay per use
- Many tools to help upload data
  - PowerShell, AZCopy, Azure Storage Explorer, …
- Many APIs to upload data
  - .Net, Python, REST, …
- Azure Data Factory to automate upload, conversion and cleanup

Cortana Analytics



PolyBase



(from msdn)

PolyBase

- PolyBase allows files in the public or private cloud to be treated as if they are tables in SQL Server
- New in SQL Server 2016
- Was already part of Parallel Data Warehouse / Analytics Platform Services
- Similar functionality is available in Azure SQL Data Warehouse (Cortana Analytics)

Why PolyBase?

- 3 common scenarios
  - Data exploration
  - Simplify ETL (Extract, Transform Load) and Data Warehouse
  - Offload rarely used data

Why PolyBase? Data Exploration!

- Data exploration of cloud based data
  - Using T-SQL instead of Hive, Pig, U-SQL, …
- Mixing structured and semi structured data in T-SQL queries
- Using tools that connect to SQL Server
  - Reporting Services
  - Excel
  - Power BI Desktop, …

Why PolyBase? Simplify ETL & DWH!

- ETL can be simplified
  - Instead of looping over a set of cloud-stored text files and copy them into a SQL table we immediately treat this collection of files as a staging table
- In some cases we can even skip staging and use the files in the cloud as a fact table directly
  - Real-time facts

Why PolyBase? Offload data

- Instead of deleting old records we can store them in Azure Storage or Hadoop
- This can even be directly written from SQL Server PolyBase!
  - No updates or deletes though

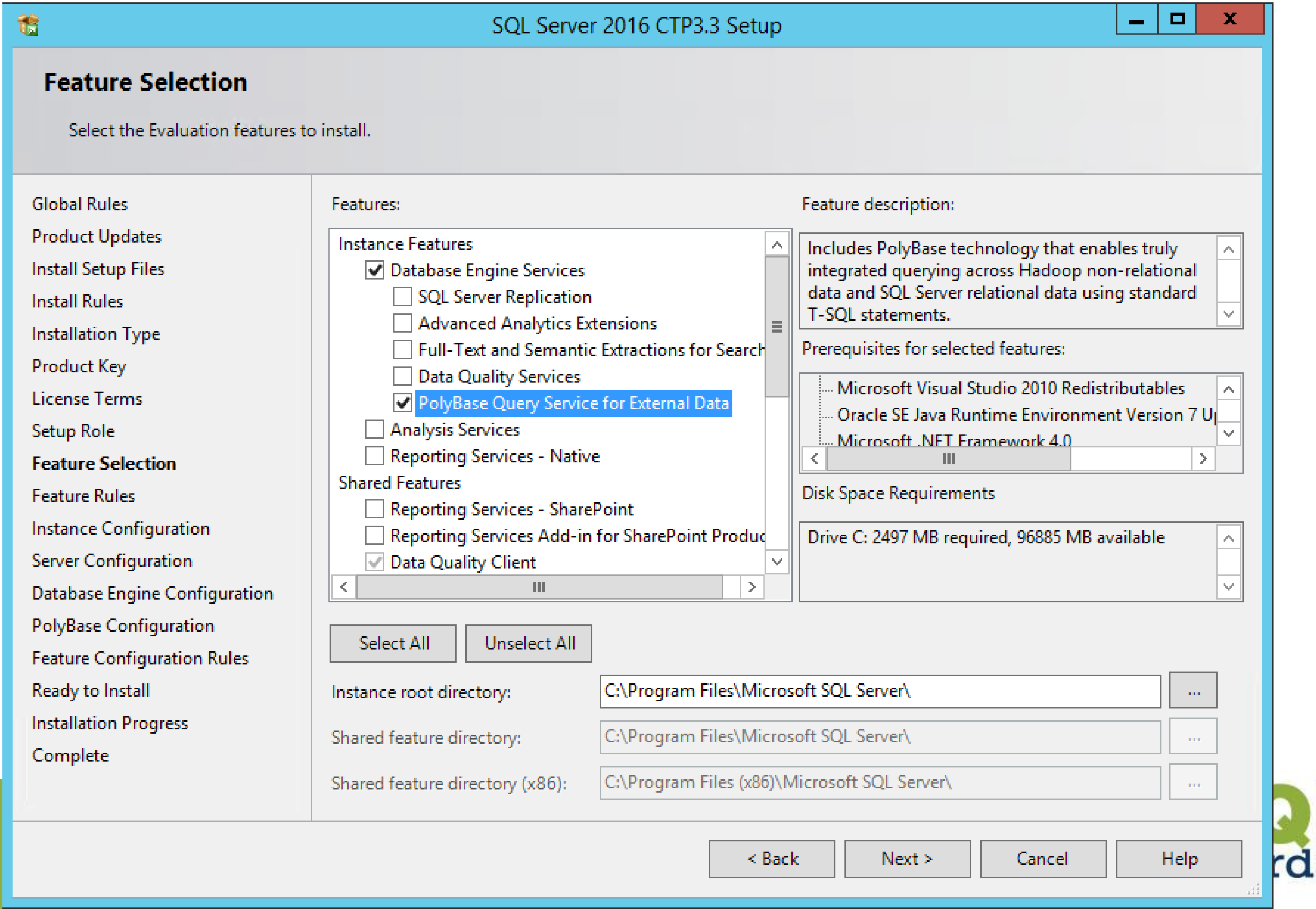
First things first

INSTALLING AND CONFIGURING POLYBASE

Prerequisites

- 64 bit version of SQL Server 2016
- Java SE RunTime Environment (JRE) version 7.51 or higher (64-bit)

Setup



SQL Server 2016 CTP3.3 Setup

Feature Selection

Select the Evaluation features to install.

Features:

- Instance Features
  - ☑ Database Engine Services
    - □ SQL Server Replication
    - □ Advanced Analytics Extensions
    - □ Full-Text and Semantic Extractions for Search
    - □ Data Quality Services
    - ☑ PolyBase Query Service for External Data
  - □ Analysis Services
  - □ Reporting Services - Native
- Shared Features
  - □ Reporting Services - SharePoint
  - □ Reporting Services Add-in for SharePoint Produc
  - ☑ Data Quality Client

Feature description:

Includes PolyBase technology that enables truly integrated querying across Hadoop non-relational data and SQL Server relational data using standard T-SQL statements.

Prerequisites for selected features:

- Microsoft Visual Studio 2010 Redistributables
- Oracle SE Java Runtime Environment Version 7 U
- Microsoft .NET Framework 4.0

Disk Space Requirements

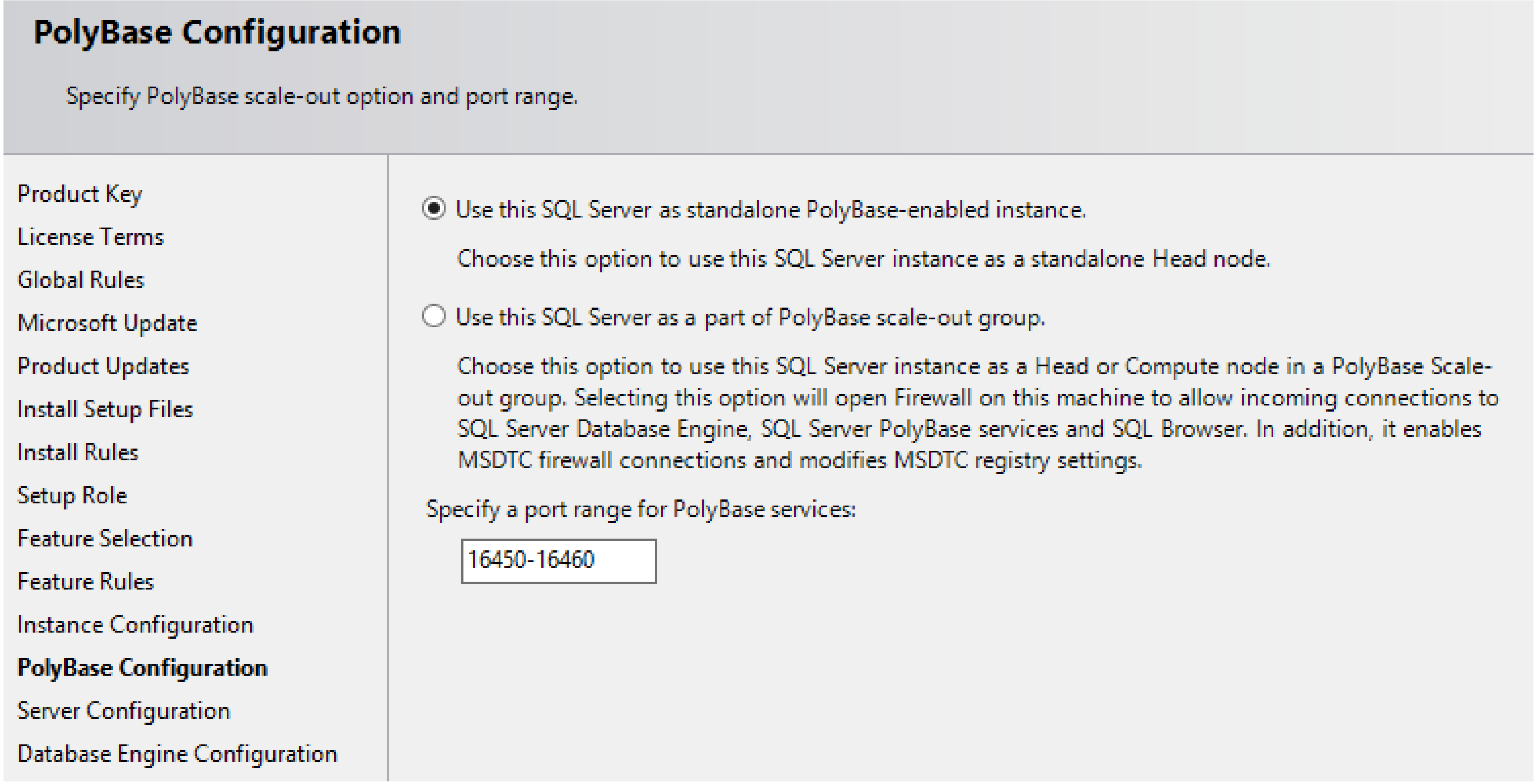
Drive C: 2497 MB required, 96885 MB available

Instance root directory: C:\Program Files\Microsoft SQL Server\

Shared feature directory: C:\Program Files\Microsoft SQL Server\

Shared feature directory (x86): C:\Program Files (x86)\Microsoft SQL Server\

Setup



PolyBase Configuration

Specify PolyBase scale-out option and port range.

● Use this SQL Server as standalone PolyBase-enabled instance.

Choose this option to use this SQL Server instance as a standalone Head node.

○ Use this SQL Server as a part of PolyBase scale-out group.

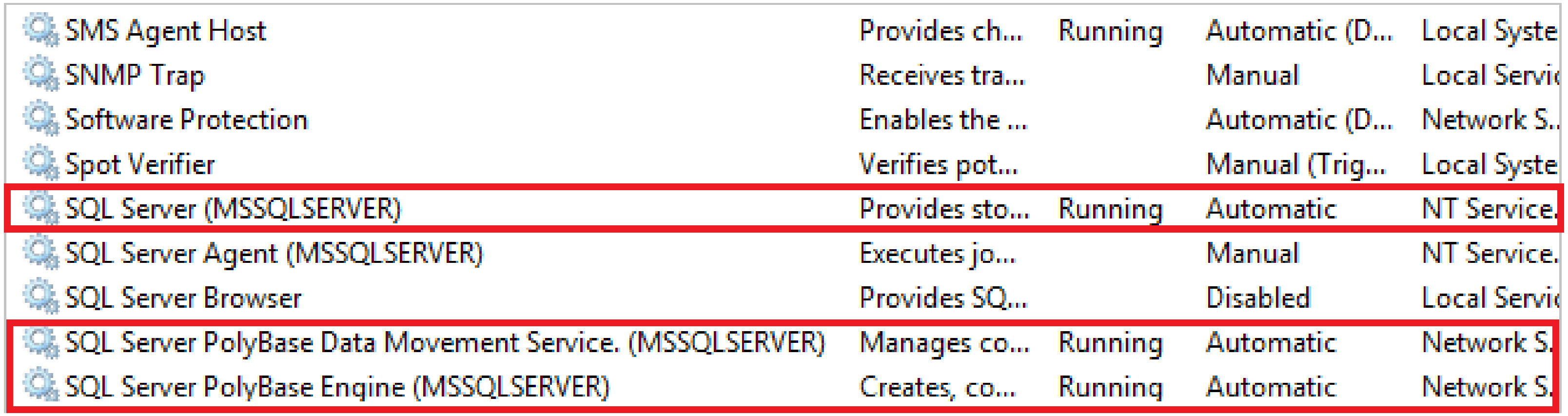
Choose this option to use this SQL Server instance as a Head or Compute node in a PolyBase Scale-out group. Selecting this option will open Firewall on this machine to allow incoming connections to SQL Server Database Engine, SQL Server PolyBase services and SQL Browser. In addition, it enables MSDTC firewall connections and modifies MSDTC registry settings.

Specify a port range for PolyBase services:

16450-16460

PolyBase Services

- Installing PolyBase creates two extra services
  - Don’t forget to restart them when changing advanced settings



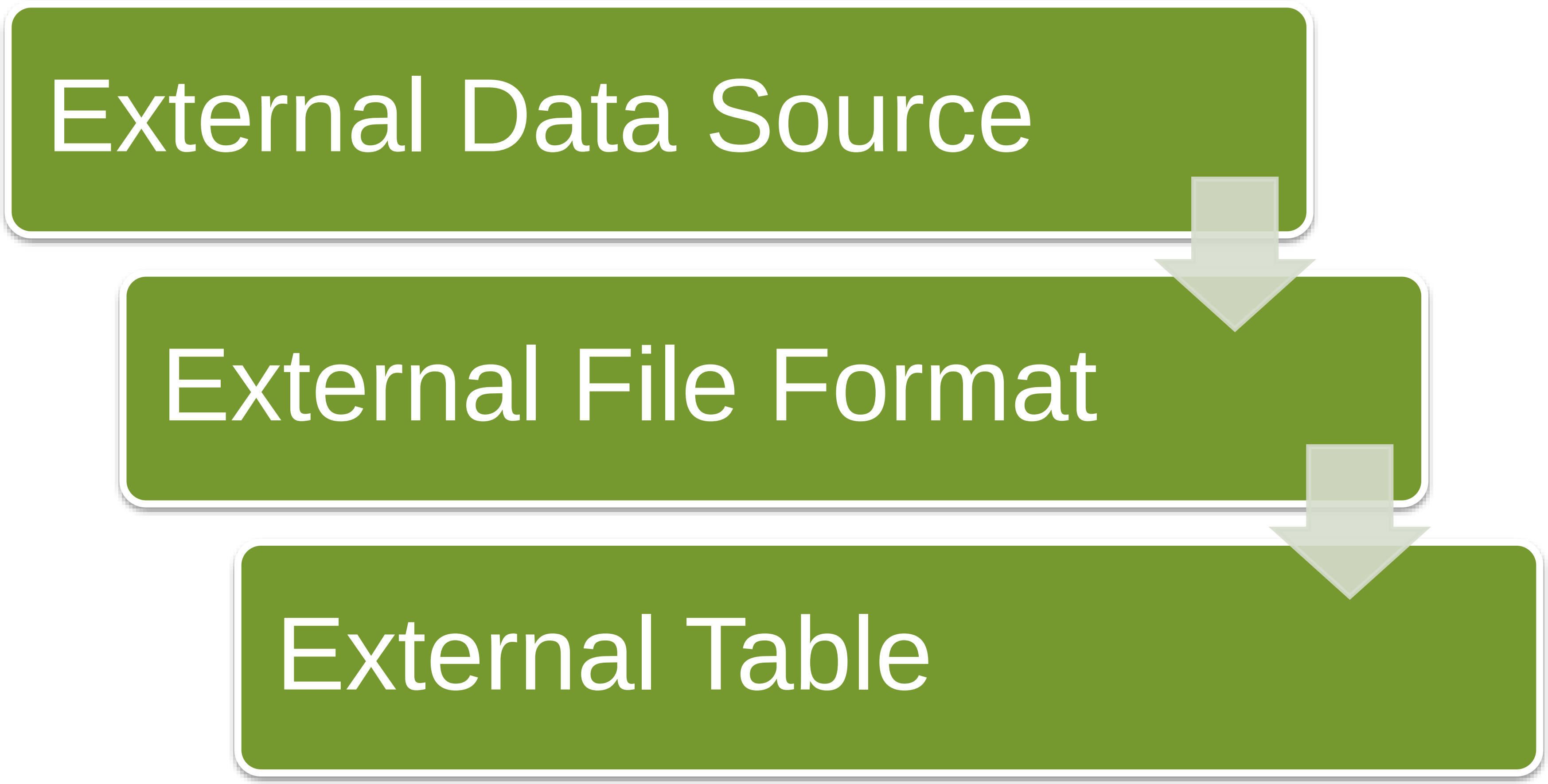
SMS Agent Host	Provides ch...	Running	Automatic (D...	Local Syste
SNMP Trap	Receives tra...		Manual	Local Servi
Software Protection	Enables the ...		Automatic (D...	Network S...
Spot Verifier	Verifies pot...		Manual (Trig...	Local Syste
SQL Server (MSSQLSERVER)	Provides sto...	Running	Automatic	NT Service.
SQL Server Agent (MSSQLSERVER)	Executes jo...		Manual	NT Service.
SQL Server Browser	Provides SQ...		Disabled	Local Servi
SQL Server PolyBase Data Movement Service. (MSSQLSERVER)	Manages co...	Running	Automatic	Network S.
SQL Server PolyBase Engine (MSSQLSERVER)	Creates, co...	Running	Automatic	Network S.

Configuration

- We need to set the type of Hadoop connectivity allowed at the server level
  - Hence only one type per server allowed
  - But some types allow for different sorts of connections

```
exec sp_configure 'hadoop connectivity', 7;
Reconfigure;
```

Connecting in 3 steps



External Data Source

External File Format

External Table

External Data Source

- Points to Hadoop (Cloudera or Hortonworks) or Azure blob storage
- Provide credentials via SQL Server credential

Create external data source

```
CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential
WITH IDENTITY = ‘mystorageaccount',
SECRET = ‘azurestorageaccesskey#&48blablabla';

CREATE EXTERNAL DATA SOURCE AzureStorage
WITH (
TYPE = Hadoop,
LOCATION =
  'wasbs://blobstorename@mystorageaccount.blob.core.windows.net',
CREDENTIAL = AzureStorageCredential
);
```

External File Format

- Three file formats are supported
  - Delimited text files (csv, tsv, …)
    - This can be used both on Hadoop and on Azure storage
  - ORC and RCFILE
    - These are Hadoop specific types and cannot be used on Azure storage

Delimited text format

- For delimited files we can specify
  - Field terminator (defaults to |)
  - String delimiter
  - Date format
    - For the whole file, not per column
  - Use type default
    - How to replace missing values
  - Data compression
    - Support for gzip

Create an external file format

```
CREATE EXTERNAL FILE FORMAT CsvzipFile
WITH (
FORMAT_TYPE = DelimitedText,
FORMAT_OPTIONS (FIELD_TERMINATOR = ','),
DATA_COMPRESSION = 'org.apache.hadoop.io.compress.GzipCodec'
);
```

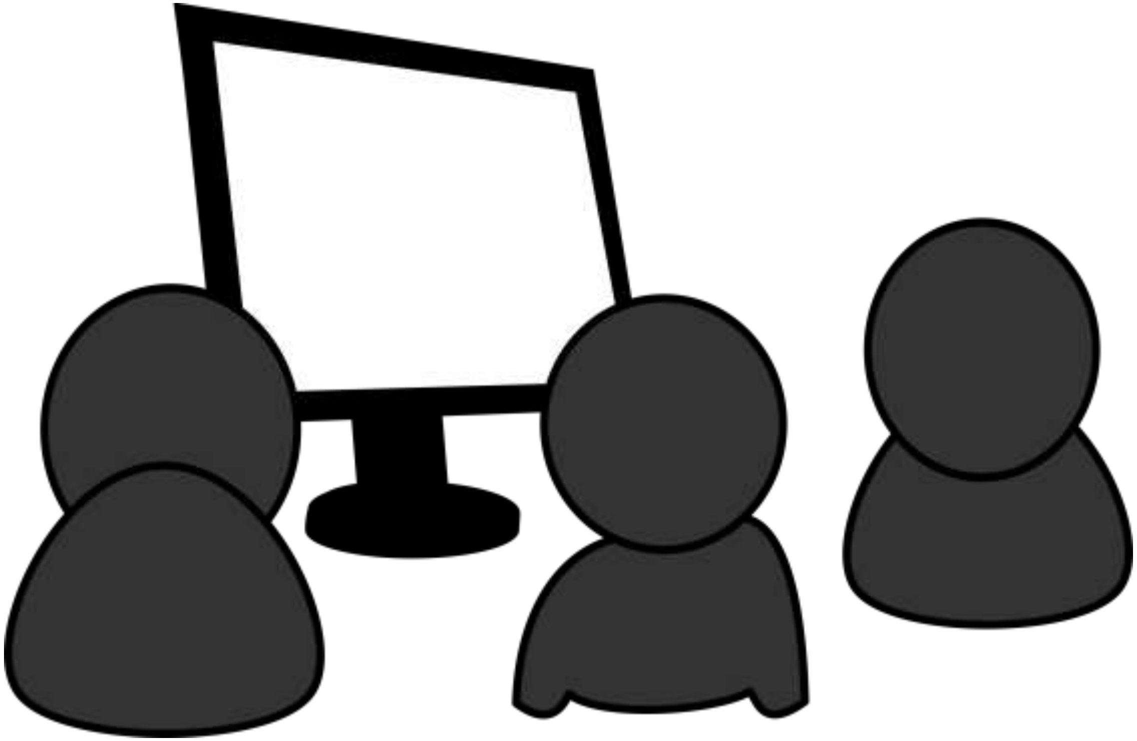
External table

- External tables are a sort of view on top of a file or folder in Azure or HDFS
  - They don’t contain data but query the underlying source
  - They only support selects and inserts, no updates or deletes
- We cannot create indexes on top of external tables
  - No PK or UNIQUE constraint either

Create an external table

```
CREATE EXTERNAL TABLE F_Sales_2016 (
  ProductID int not null,
  CustomerID int null,
  Quantity int null)
WITH (
  LOCATION = 'polydwh/Y2016',
  DATA_SOURCE=AzureStorage,
  FILE_FORMAT=CsvFile
);
```

Demo: External Tables



Demo wrap-up

- We can select from and insert into external tables
- We can create views on top of external tables
- Update, delete and merge are not supported on external tables

Error handling

- Missing values automatically translate to null values
- Rightmost missing column automatically translates into null value
- Wrong data types or more than 1 rightmost missing column requires error handling

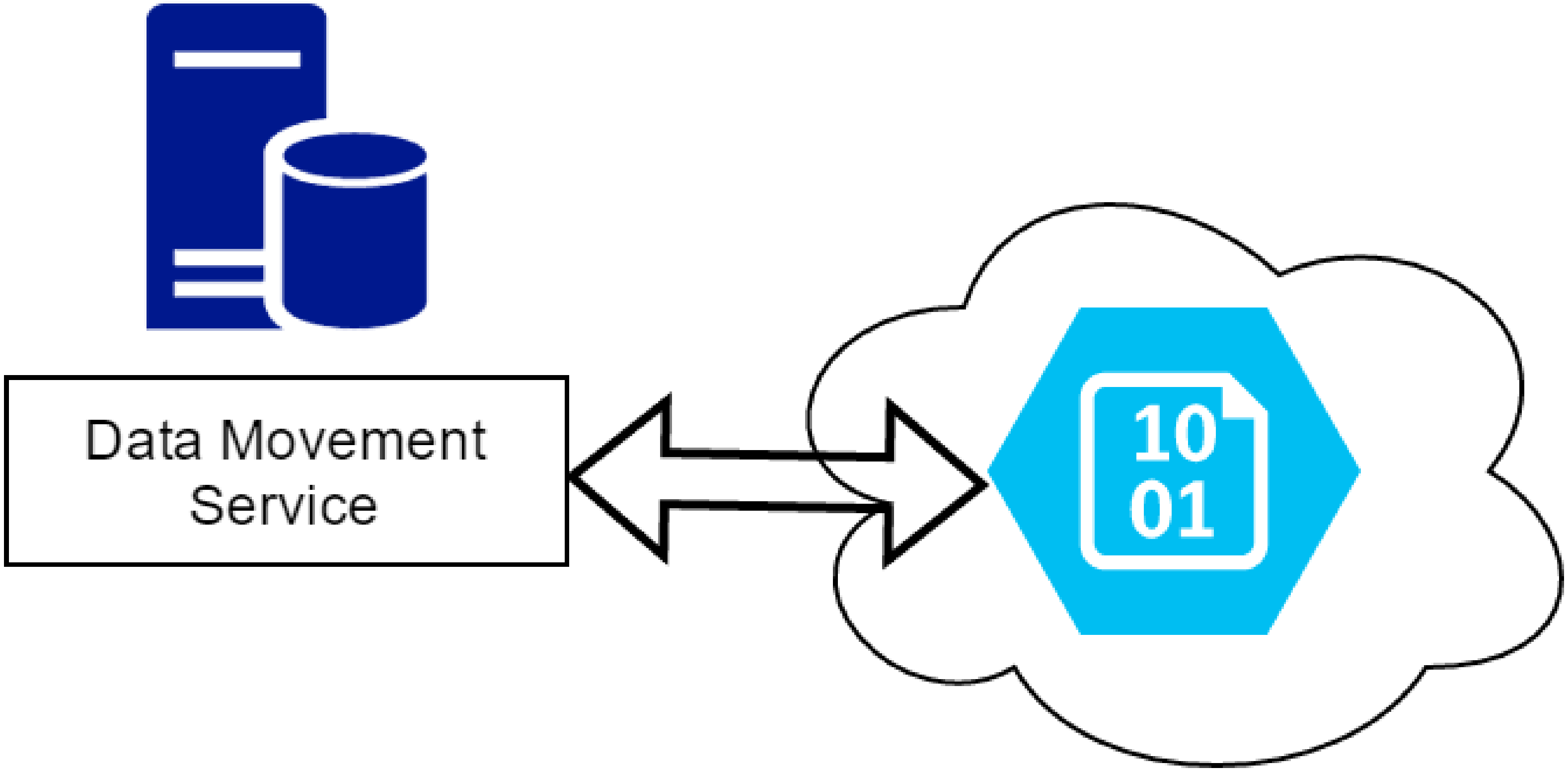
Error handling

```
CREATE EXTERNAL TABLE [Contracts]
(CustomerKey int , ContractID int ,
City nvarchar(50))
WITH (
   LOCATION='polydwh/errors',
   DATA_SOURCE = AzureStorage,
   FILE_FORMAT = CsvFile,
   REJECT_TYPE = VALUE,
   REJECT_VALUE = 5
   );
```

Demo: Under the hood

But what about BIG DATA?

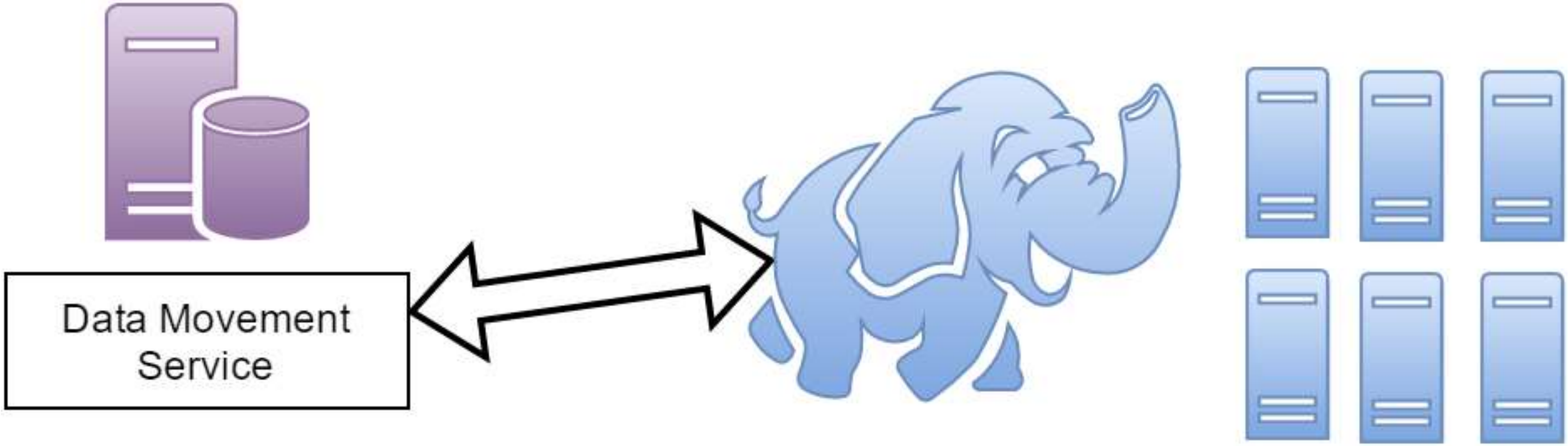
Simple setup



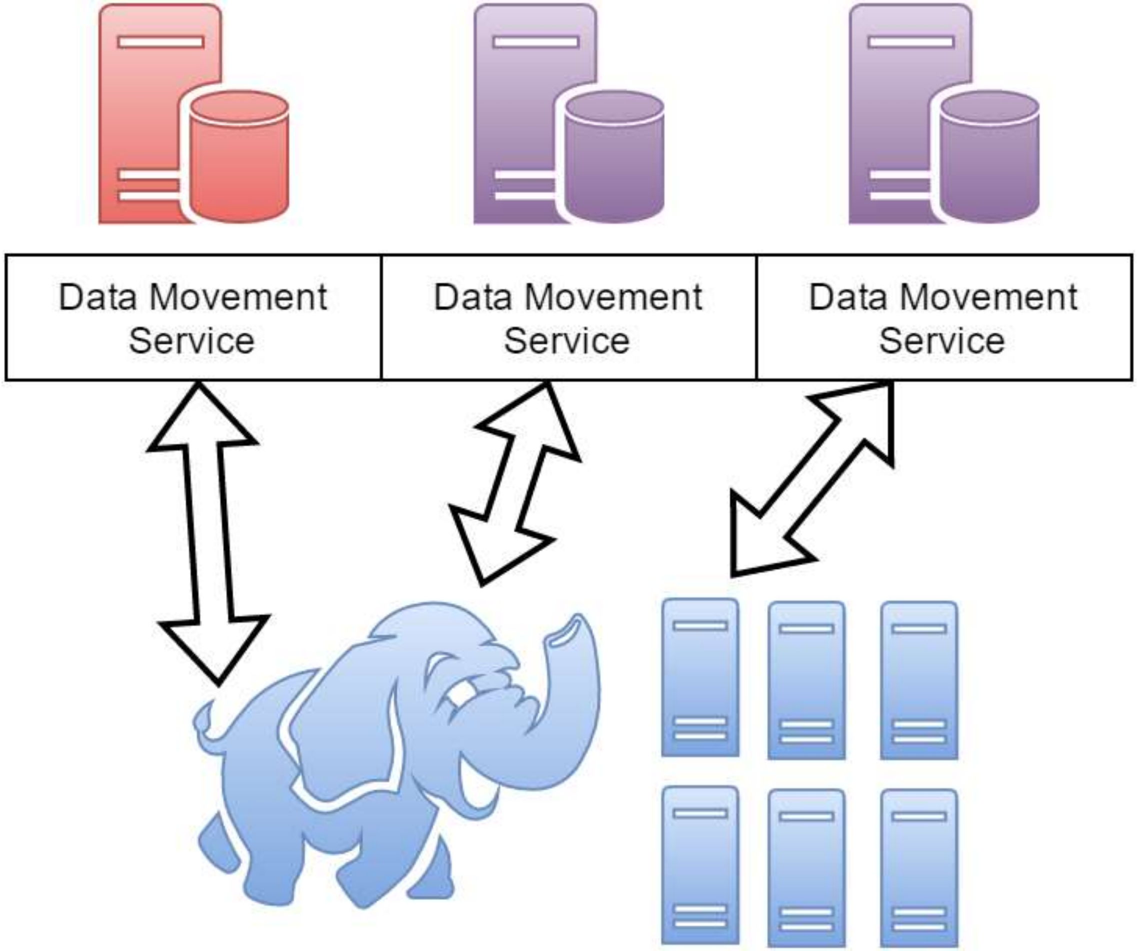
Two scale-out scenarios

- 1 SQL Server but a large Hadoop cluster
  - PolyBase can push work to Hadoop nodes via Map~~Reduce~~
- Multiple SQL Servers with PolyBase
  - 1 head node which receives query and distributes work (Enterprise)
  - Multiple worker nodes which pull data from Azure or HDFS and compute on this (Standard+)

Hadoop compute scale-out



PolyBase scale-out



Conclusions

- Don’t waste data!
- PolyBase can be useful for
  - Easy access to Big Data
  - Offloading cold data
  - Integrating IoT data in data warehouse, …
- Can be used against Azure Storage (cloud) or Hadoop (cloud or local)

Try it yourself

- 180 day free trial version SQL Server 2016
- 30 day free trial of Azure
  - 170€, or more than 8 Tb storage ☺

THANK
YOU

tillit we tell it like it is

PYRAMID ANALYTICS

Microsoft

DELL Software

THOMAS MORE

COZYROC™ Go to the next level

redgate ingeniously simple

monkey recruitment

PASS