Microsoft

Azure Data Lake & Factory

Technical Deep Dive –Day 3

Sashank Pappu
Azure Data & AI Specialist

Shiva Priya
Azure Data Engineer

Sashank Pappu

Azure Data & AI Specialist.

Certified from MIT on 'Advanced Analytics and Big Data Challenges'. Technology Enthusiast and Data Specialist. Certified Designer in Computer Vision

Certifications

- Microsoft Certified Power BI Developer.
- Microsoft Certified Azure Machine Learning Developer from Microsoft Virtual Academy
- Certified on Practical Data Analytics with Microsoft Cortana Intelligence Suite.

Role

Currently working as Independent Consultant on Azure Data & AI and Enterprise Trainer on Azure Data Analytics , Cortana Intelligence Suite and Cognitive Sciences

Experience

9+ years of total experience as Data Specialist and Data Engineer.

Training

Expert **facilitator and corporate behavioral trainer** with 5 years of experience in designing programs, content development , Data Analytics and Machine Learning.

- You can see me on – Sashankpappu.com

What we can learn together ??

- Azure Data Ecosystem
- Storage (Blob Vs Data Lake)
- Data Lake Storage
- Data Lake Analytics
- Data Factory

URL :https://github.com/aagasp/azuredataecosystem

Questionnaire

To be taken after each section .

Request you to ask the questions in Chat window .

Will respond to them after a logical break point in each topic.

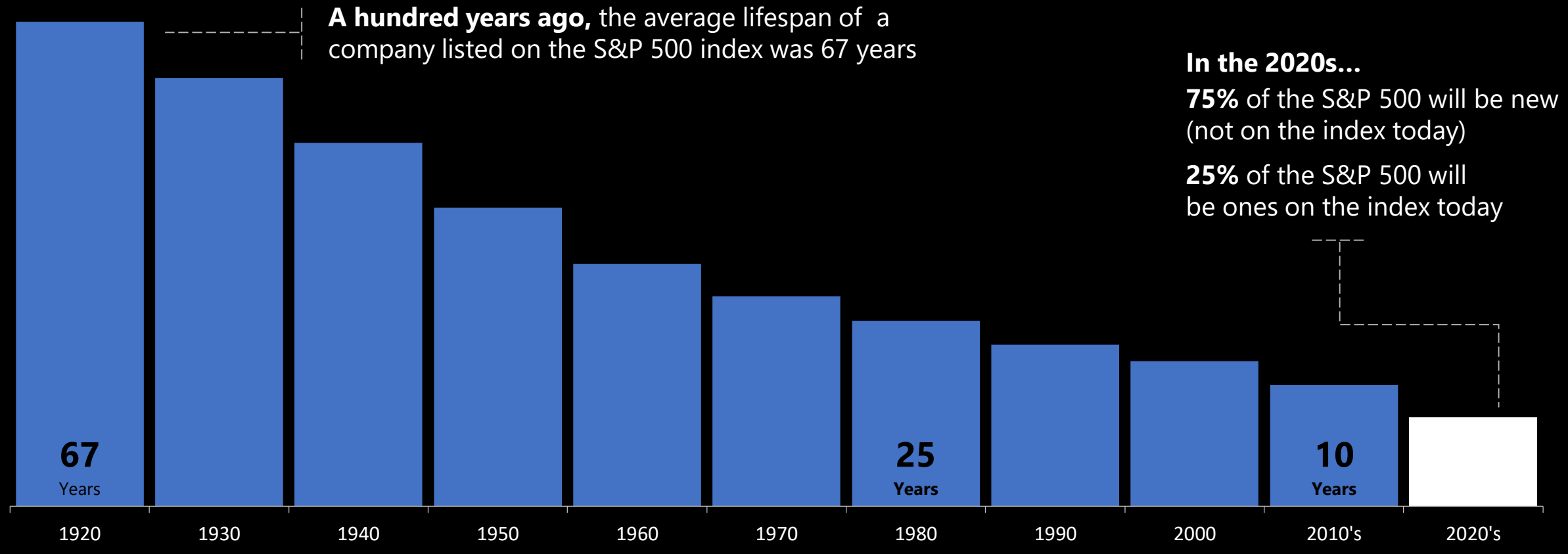
This would make the workshop interesting and approachable .

Request not to use Phones or get distracted for next 3 hours .

Expected Outcome!

- Azure Data Ecosystem – When to use what !
- Architectural Standards & Best Practices in Azure .
- Hands on Project ready confidence.
- Tackling Real world challenges in Azure Data Platform
- Real world use cases and architectures for huge clients.

Time to adapt is shrinking



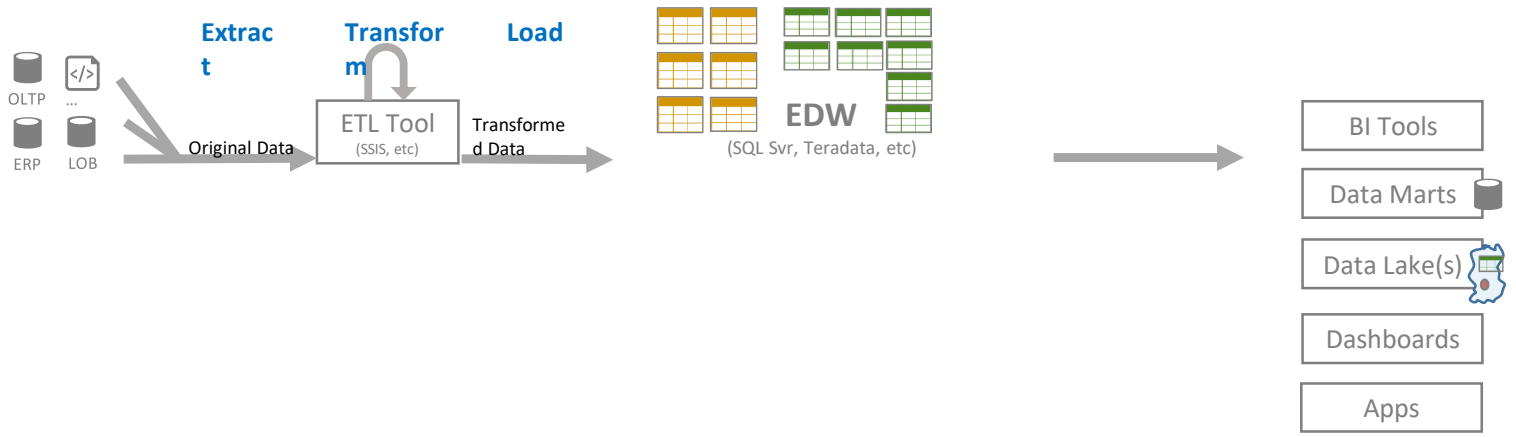
Source: BBC

Experts predict that in the next decade, **only 25%** of the companies currently listed on the S&P 500 will remain there, and the other **75% will be replaced** by new companies.

Approach of Data is Changing !!!

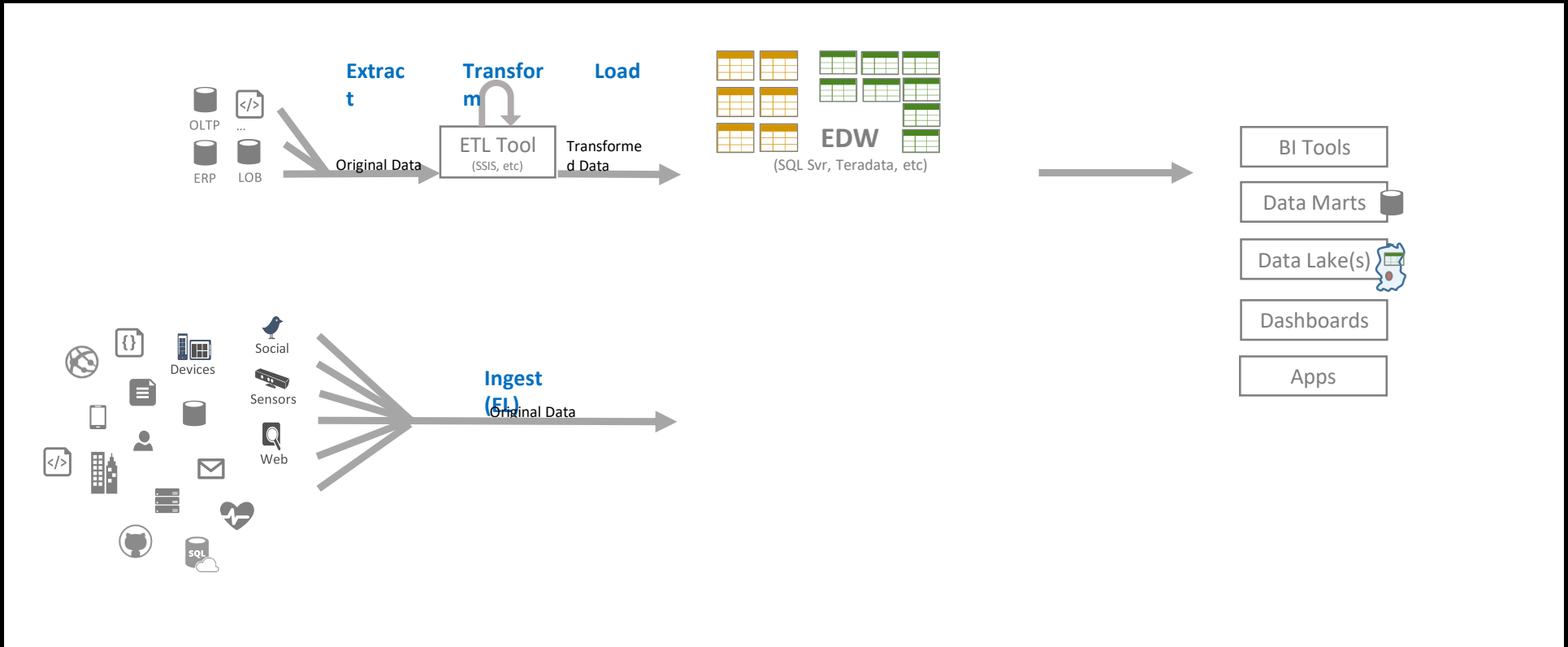
Evolving Approaches to Analytics

EDWs to Data Lakes



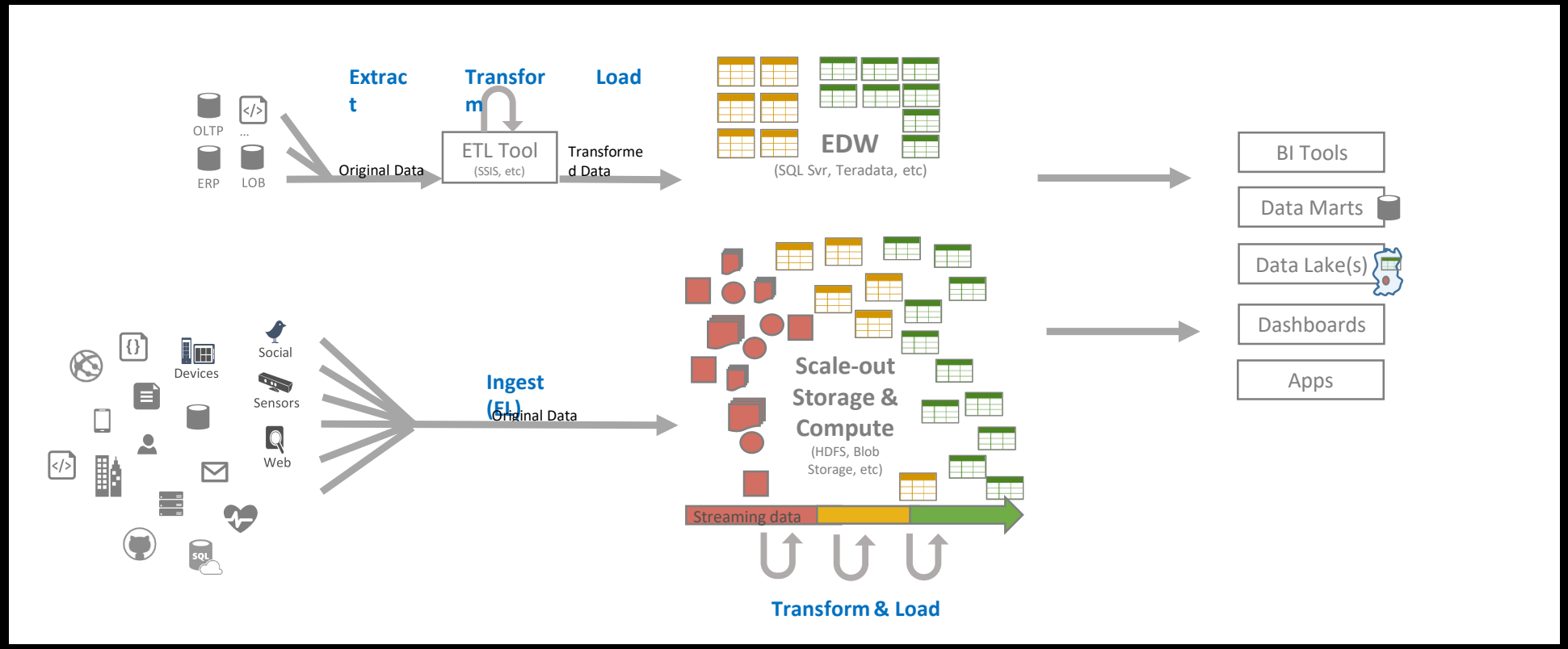
Evolving Approaches to Analytics

EDWs to Data Lakes



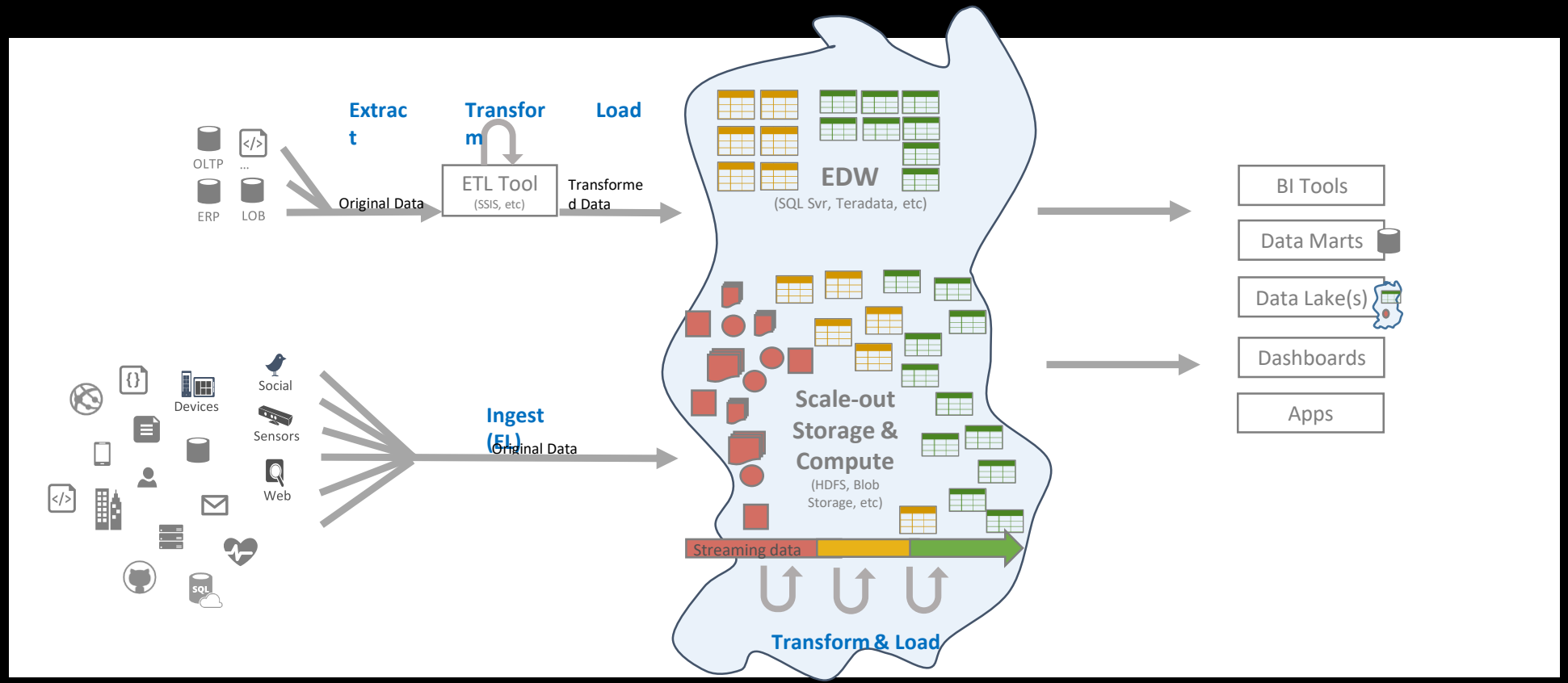
Evolving Approaches to Analytics

EDWs to Data Lakes



Evolving Approaches to Analytics

EDWs to Data Lakes



Data Lake - Customer Challenges

Data Silos

Data spans sources

Inefficiency in colocation

Analytics

Open interfaces to data Variety of analytics tools

Perf & Scale

Storage bottlenecks

IoT sources – Small writes

Price-performance

Data grows independently

Security

Compliance challenges

Effectively control access

Corporate policies

The Intelligent Lake

Petabytes of Data + Intelligence

Data Lake Store

No-limits data Lake

- Petabyte-sized files
- Trillions of objects
- Scalable I/O for parallel analytics
- Enterprise-grade Security

Data Lake Analytics

Analytics job service

- Scale instantly
- Scale per job
- Infused with Artificial Intelligence

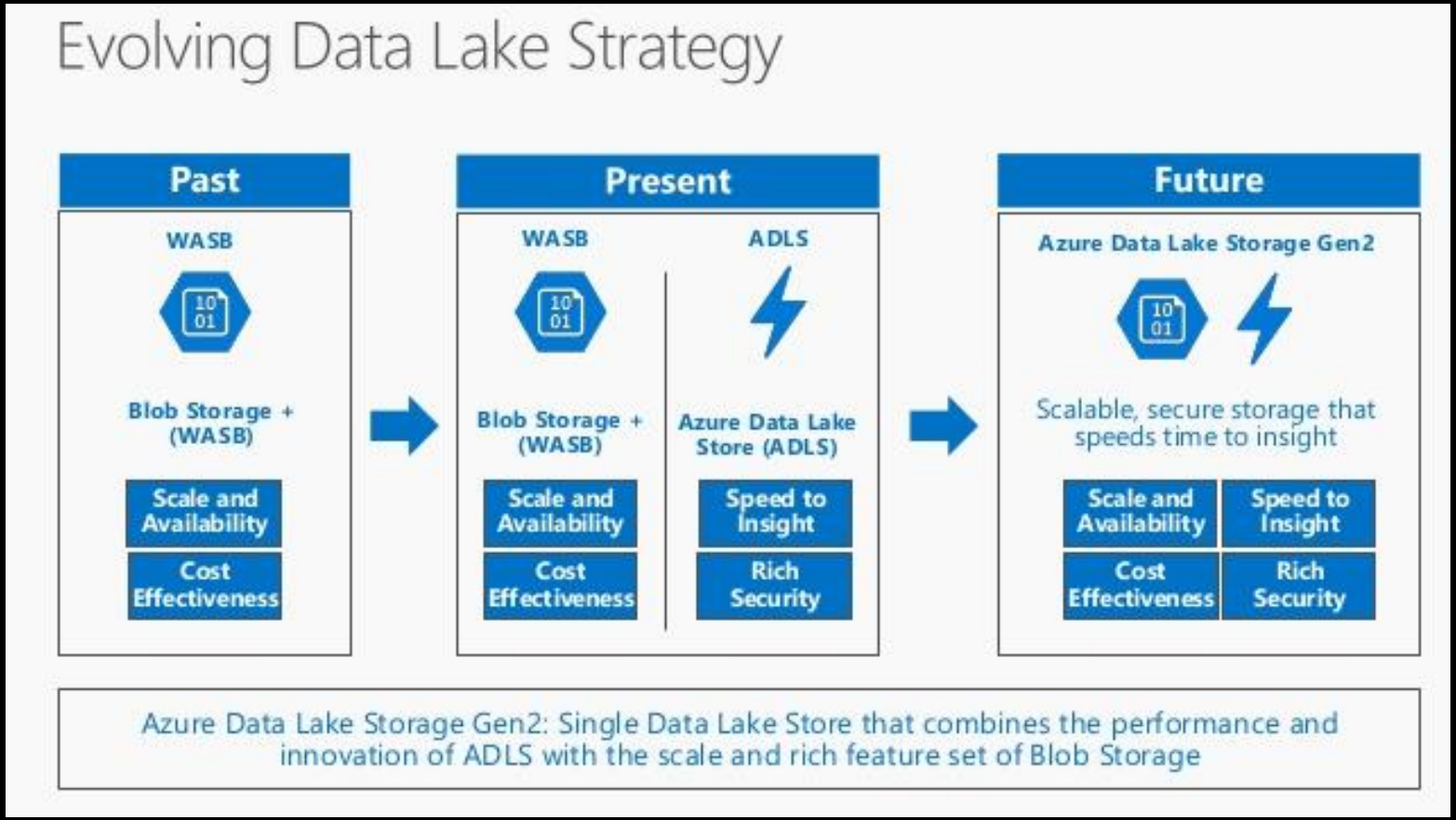
Azure Data Lake Storage Gen 1 Vs BLOB

Azure Blob Storage	Azure Data Lake
General purpose storage for storing applications and user data	Specialized storage for storing Big Data which is optimized for Analytics
Object based flat system	Hierarchical system with folders
Security based on shared secret keys and signed URLs	Security based on Azure AD authentication
LRS, ZRS, GRS and RA-GRS	LRS, ZRS, GRS and RA-GRS
SDKs available for .NET, Java, Python, Node.js, C++, Ruby	.NET, Java, Node.js

ADLS Gen1 + BLOB = ADLS Gen 2

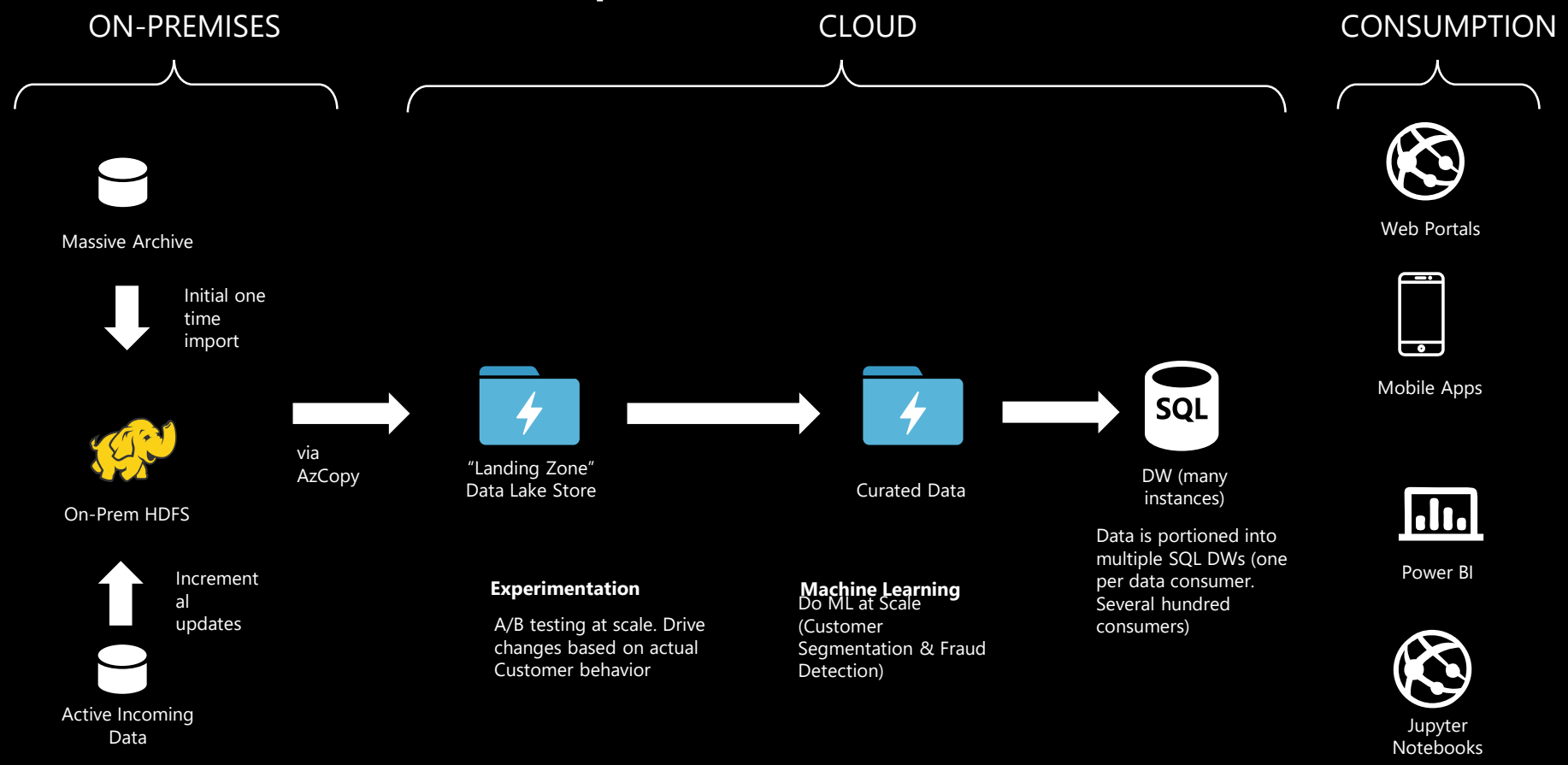
Use	Data Lake Store	Blob Storage	Data Lake Gen 2
Hot/Cold Storage Tiers	No	Yes	Yes
Redundant Storage	No	Yes	Yes
AD Security	Yes	No	Yes
HDFS Compatible	Yes	No	Yes

ADLS Gen1 + BLOB = ADLS Gen2



Azure Data Lake Use Cases

Retail Scenario Implementation



TYPES OF DATA

Structured Data

Un-structured Data

Images

Emails

DATA LAKE ZONE

Acquire/ Ingest

Apply Metadata, Protect sensitive

Data Quality and Validation

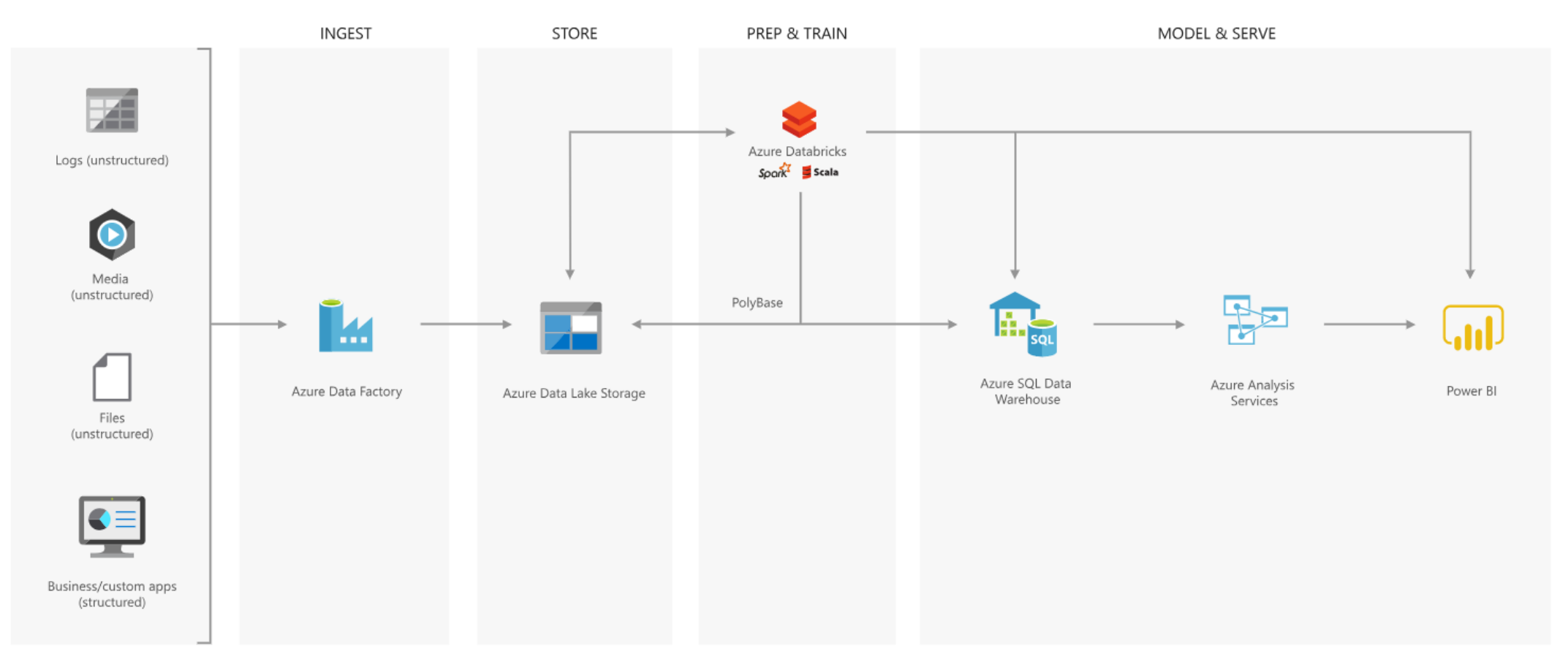
Data Quality and Validation

CONSUMER OUTPUT

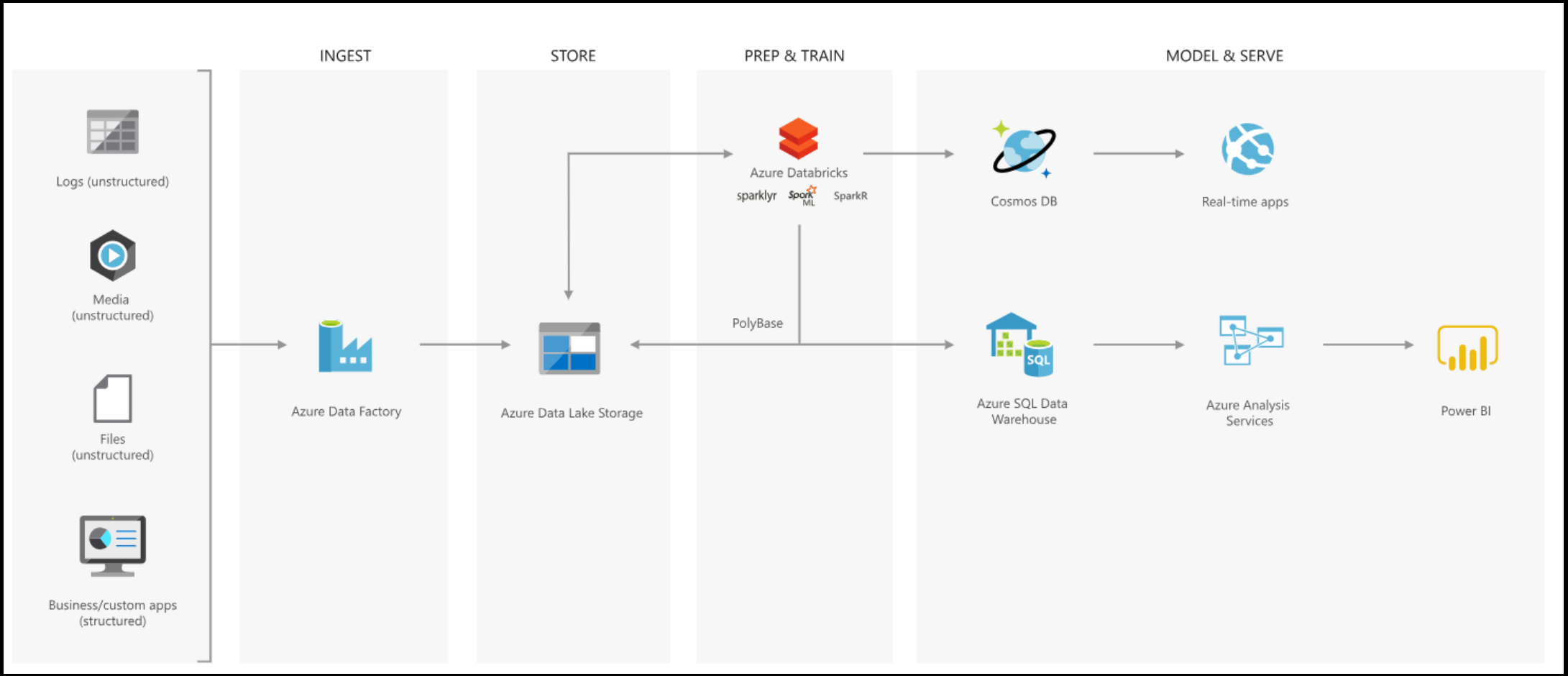
Data as and when required in formats suitable to consumers

Data Visualization
Data Prep Tools

Best Practices – Modern Data Warehouse



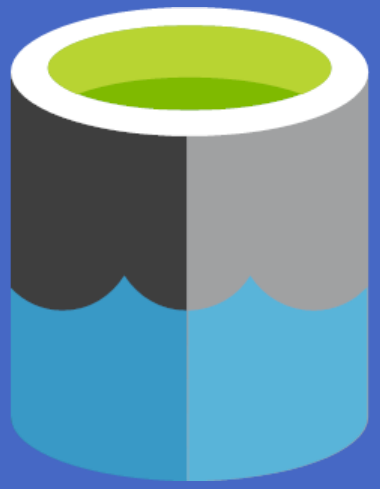
Advanced Analytics Approach



Azure Data Lake Store

Introducing Azure Data Lake Store

A hyper-scale repository for big data analytics workloads



Store **ANY DATA** in its native format

HADOOP FILE SYSTEM (HDFS) for the cloud

ENTERPRISE GRADE

No limits to **SCALE**

Optimized for analytic workload **PERFORMANCE**

The Data Lake Approach

Ingest all data
regardless of requirements

Store all data
in native format without schema definition

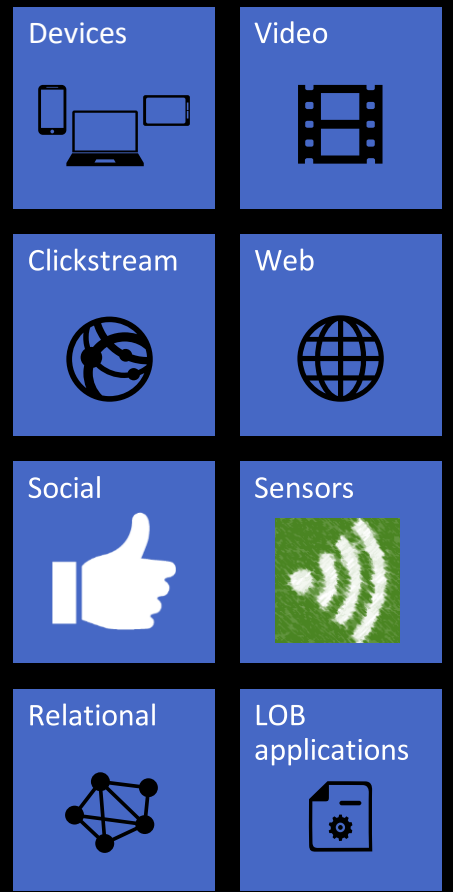
Do analysis
Using analytic engines like Hadoop and ADLA

Any data

Unstructured

Semi-structured

Structured



Durable and highly available

Automatically replicates your data

Three copies within a single region

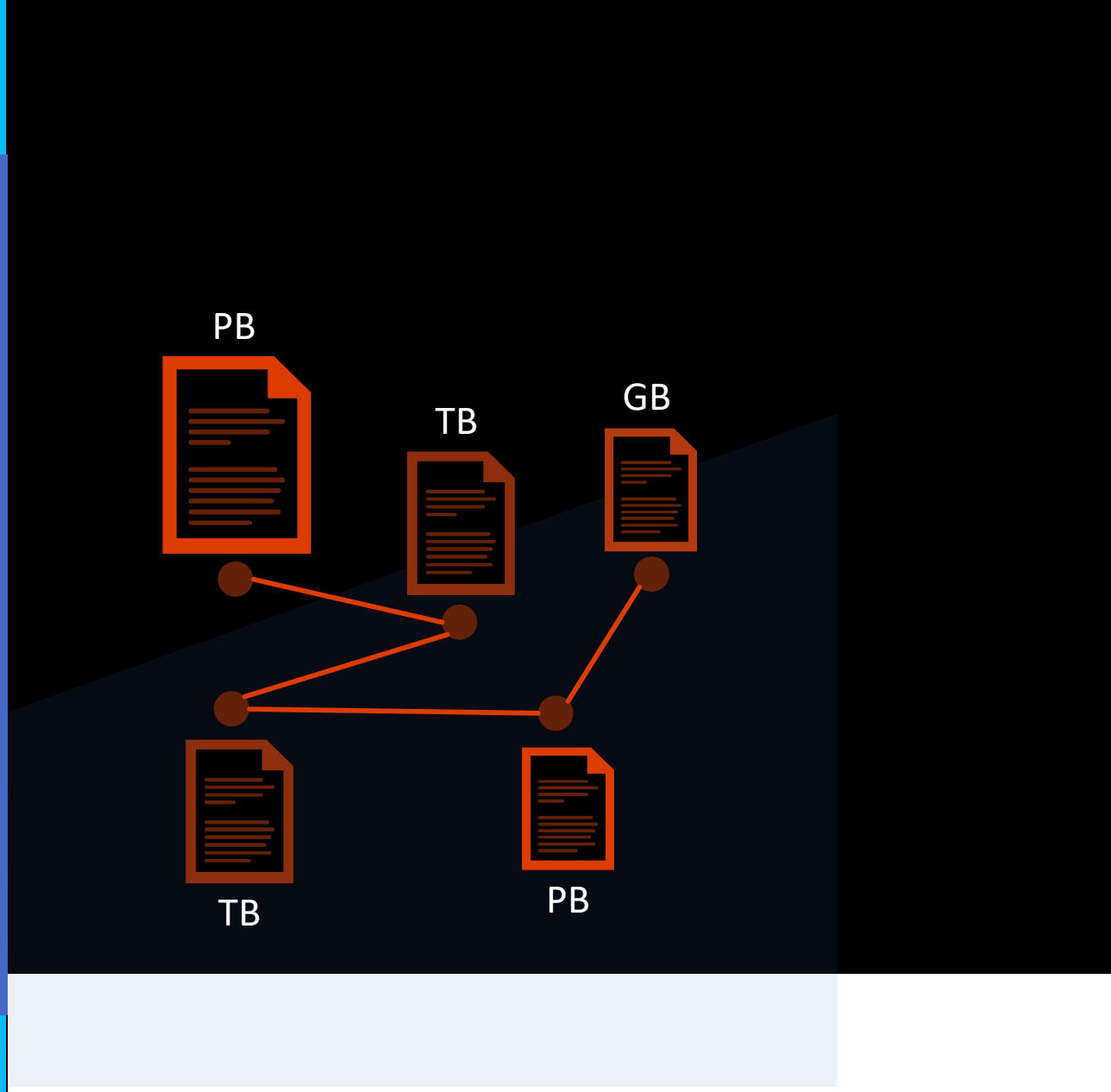
Highly available

Unlimited storage

Unlimited account sizes

Individual file sizes from gigabytes to petabytes

No limits to scale



Hands on – Data Lake Gen2

Creation of Data Lake Gen2 Account

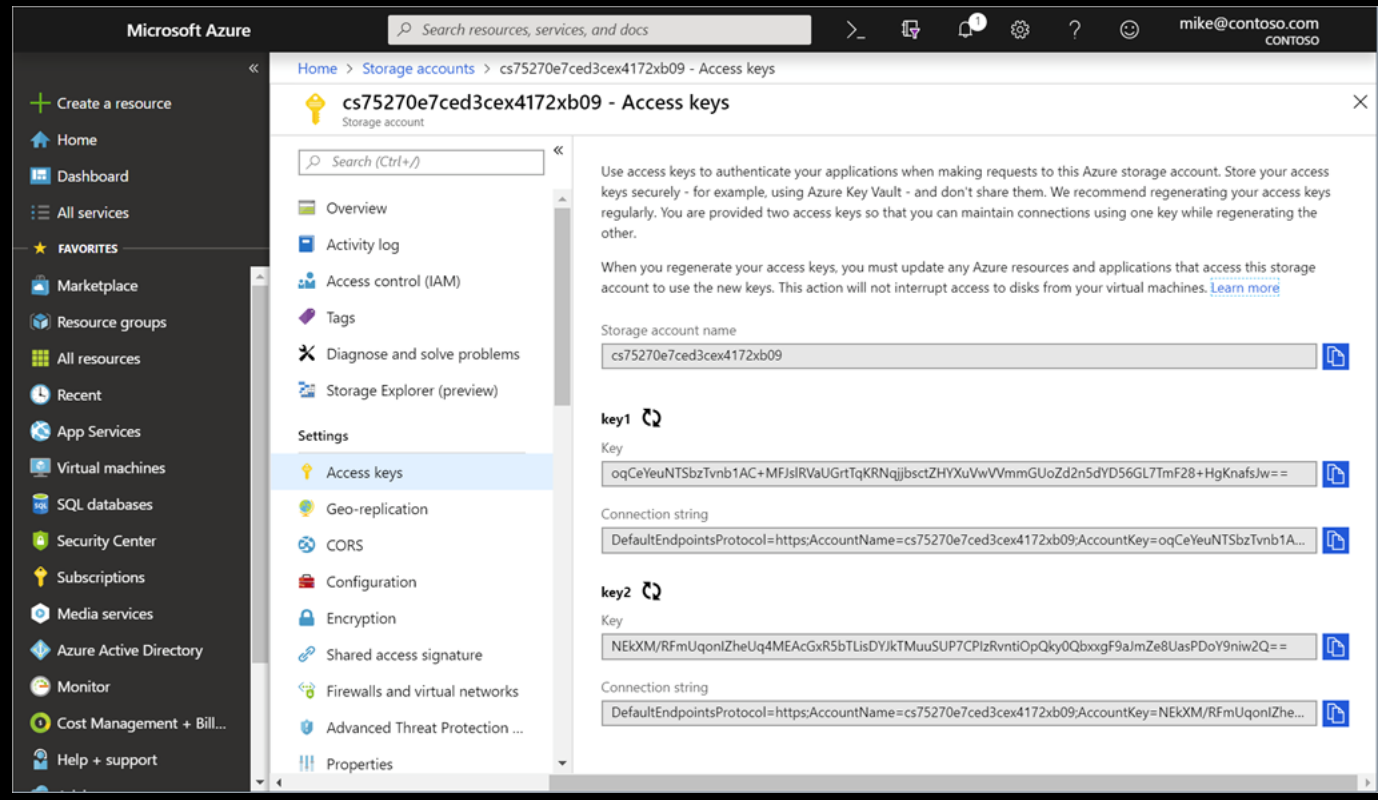
Security Considerations

Data Protection

- Protect the data at rest
- Protect the data in transit
- Support browser cross-domain access
- Control who can access data
- Audit storage access

Understand storage account keys

- Data is generated or consumed by custom applications
- The applications are written in various languages.
- Create authorized apps in Active Directory to control access to the data in blobs

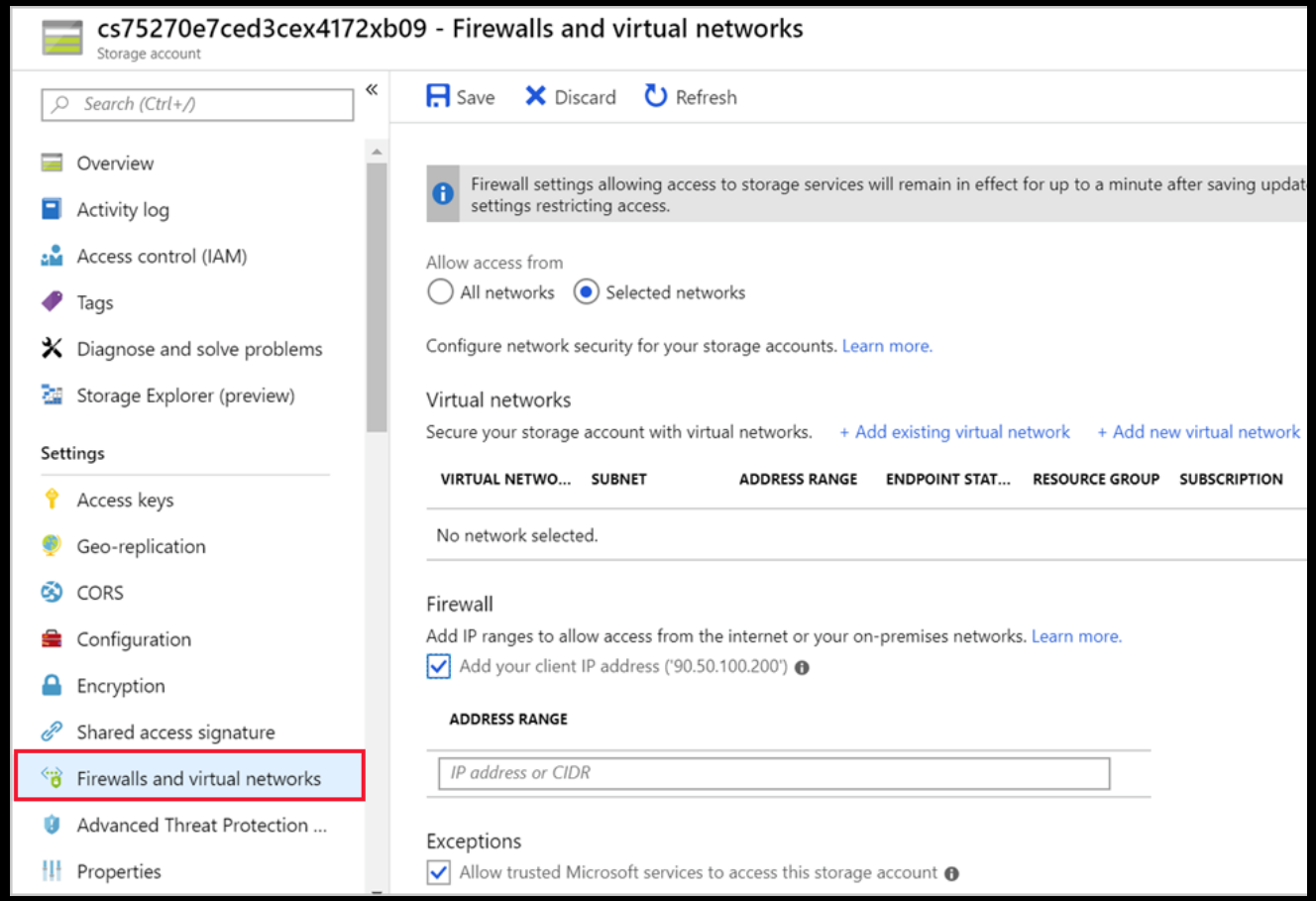


Understand shared access signatures

- You shouldn't share storage account keys with external third-party applications
- For untrusted clients, use a *shared access signature* (SAS)
- **Types of shared access signatures**
  - *service-level* shared access signature
  - *account-level* shared access signature

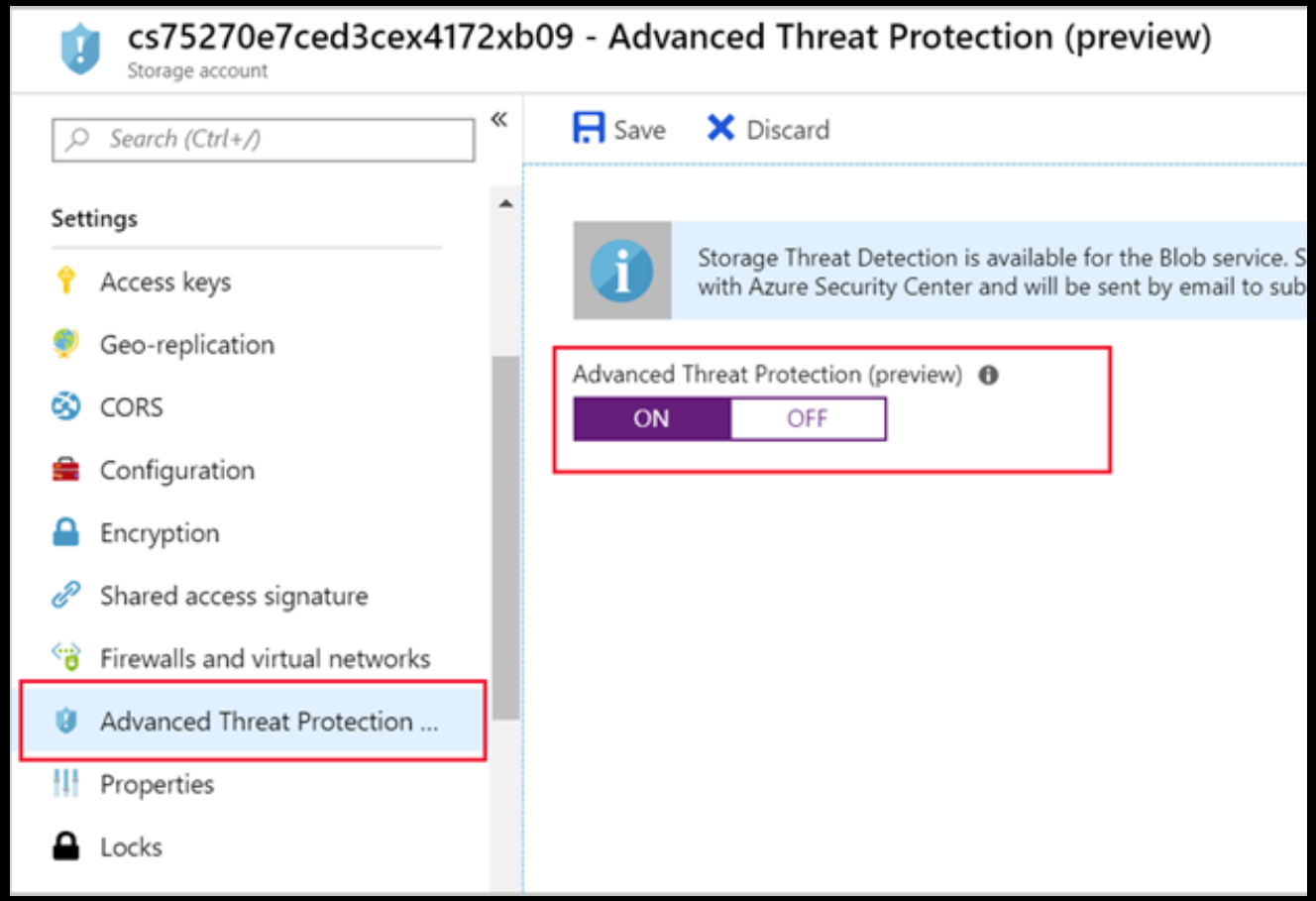
Control network access to your storage account

- Storage accounts accept connections from clients on any network
- To limit access to selected networks, you must first change the default action.
- You can restrict access to specific IP addresses, ranges, or virtual networks



Understand Advanced Threat Protection for Azure Storage

- Often any protection only shows you that an intrusion *has already occurred*.
- What you really want is a way to be notified when suspicious activity is happening
- Advanced Threat Protection, detects anomalies in account activity
- It then notifies you of potentially harmful attempts to access your account.



Azure Data Lake Analytics

Quick Recap

ADL VS SQL DB

Azure Data Lake

ADL Analytics + ADL Store

SQL DB

SQL Query + DB Storage

U-SQL

A new language for Big Data

Familiar syntax to millions of SQL & .NET developers

Unifies declarative nature of SQL with the imperative power of C#

Unifies structured, semi-structured and unstructured data

Distributed query support over all data

History

History

Bing needed to…

- Understand user behavior

And do it…

- At massive scale
- With agility and speed
- At low cost

So they built …

- Cosmos

Cosmos

- Batch Jobs
- Interactive
- Machine Learning
- Streaming

Thousands of Developers

Pricing

REGION: East US 2

CURRENCY: Indian Rupee (₹)

Pricing Details

Azure Data Lake Analytics query service is currently in preview and the prices below reflects a 50% preview discount.

USAGE	PRICE
Analytics Unit	₹1.02 / Minute
Completed Job	₹1.50 / Job

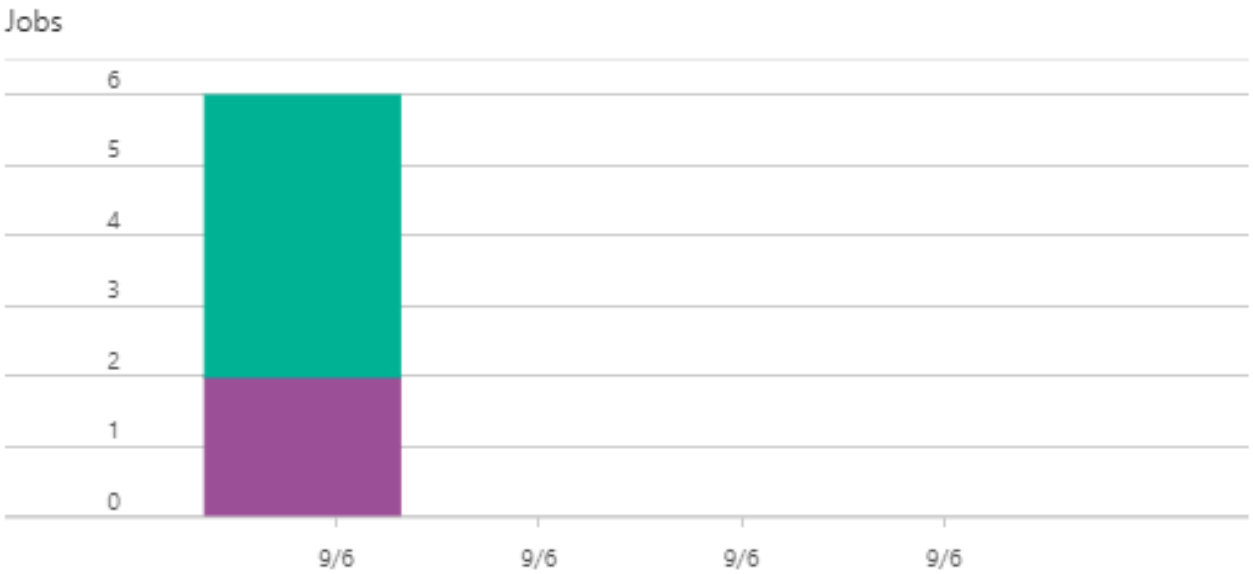
Finally, each Azure Data Lake Analytics account has configurable quotas limiting the number of AUs that can be assigned to jobs and the number of concurrent jobs.

UNIT	LIMIT
# of AU per job	Up to 20 *
# of concurrent jobs per account	Up to 3 jobs *

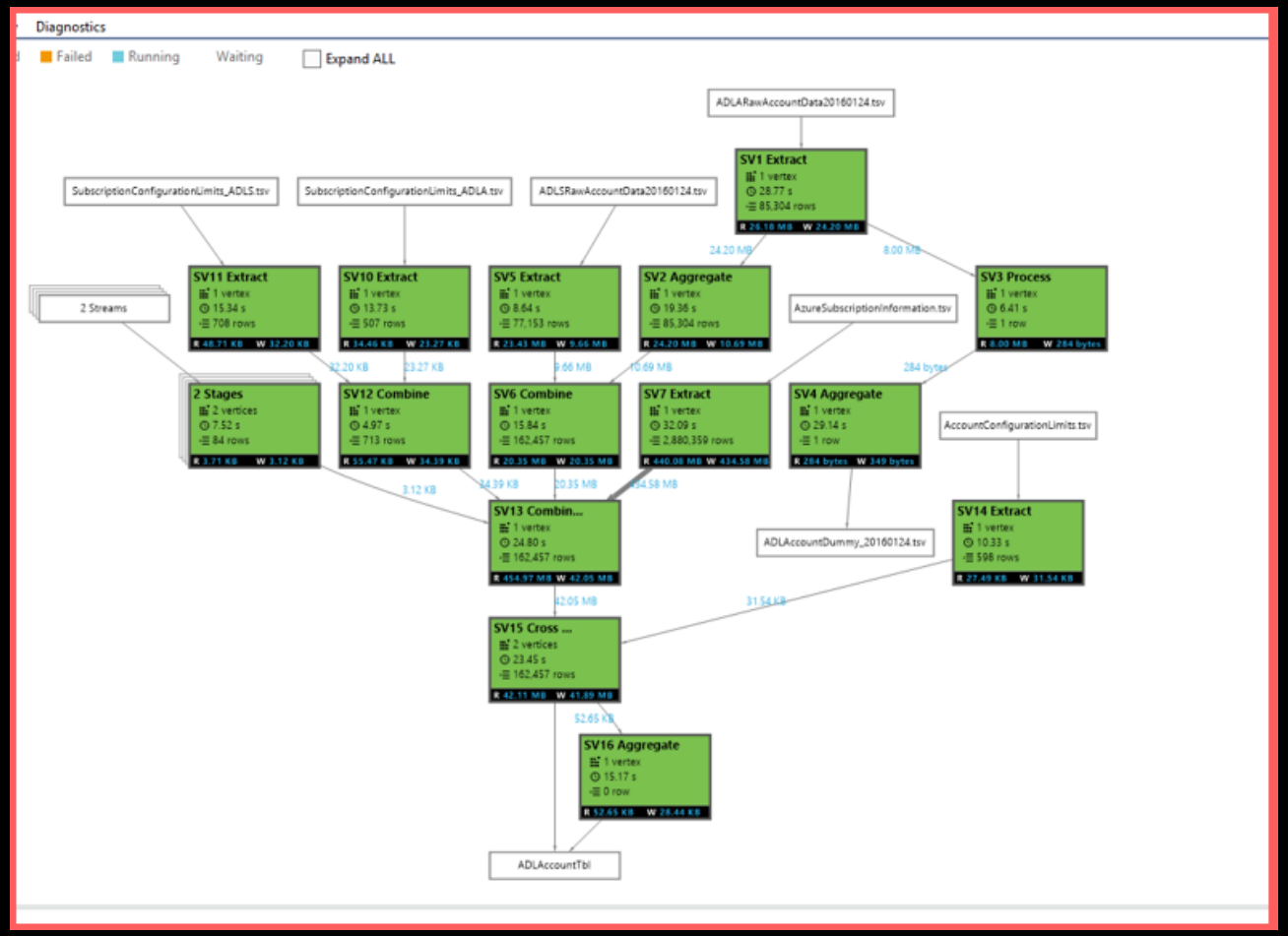
* To increase level please contact us

Support & SLA

Azure Data Lake Analytics (ADLA) Demo



TOTAL	SUCCEEDED	FAILED	CANCELLED	ACTIVE
6	4	2	0	0


Diagnostics
Failed
Running
Waiting
Expand ALL
ADLARawAccountData20160124.tsv
SV1 Extract
1 vertex
28.77 s
85,304 rows
R 26.10 MB W 24.20 MB
SubscriptionConfigurationLimits_ADLS.tsv
SubscriptionConfigurationLimits_ADLA.tsv
ADLSRawAccountData20160124.tsv
24.20 MB
8.00 MB
SV11 Extract
1 vertex
15.34 s
708 rows
R 48.71 KB W 32.20 KB
SV10 Extract
1 vertex
13.73 s
507 rows
R 34.46 KB W 23.27 KB
SV5 Extract
1 vertex
8.64 s
77,153 rows
R 23.43 MB W 9.66 MB
SV2 Aggregate
1 vertex
19.36 s
85,304 rows
R 24.20 MB W 10.69 MB
AzureSubscriptionInformation.tsv
SV3 Process
1 vertex
6.41 s
1 row
R 8.00 MB W 284 bytes
2 Streams
32.20 KB
23.27 KB
9.66 MB
10.69 MB
284 bytes
2 Stages
2 vertices
7.52 s
84 rows
R 3.71 KB W 3.12 KB
SV12 Combine
1 vertex
4.97 s
713 rows
R 55.47 KB W 34.39 KB
SV6 Combine
1 vertex
15.84 s
162,457 rows
R 20.35 MB W 20.35 MB
SV7 Extract
1 vertex
32.09 s
2,880,359 rows
R 440.08 MB W 434.58 MB
SV4 Aggregate
1 vertex
29.14 s
1 row
R 284 bytes W 349 bytes
AccountConfigurationLimits.tsv
3.12 KB
34.39 KB
20.35 MB
434.58 MB
SV13 Combin...
1 vertex
24.80 s
162,457 rows
R 454.97 MB W 42.05 MB
ADLAccountDummy_20160124.tsv
SV14 Extract
1 vertex
10.33 s
598 rows
R 27.49 KB W 31.54 KB
42.05 MB
31.54 KB
SV15 Cross ...
2 vertices
23.45 s
162,457 rows
R 42.11 MB W 41.89 MB
52.65 KB
SV16 Aggregate
1 vertex
15.17 s
0 row
R 52.65 KB W 28.44 KB
ADLAccountTbl

Job Scheduling

States, Queue, Priority

Job Status

Job Summary

Preparing	Queued	Running	Finalizing
37 seconds	0 seconds	9.9 minutes	

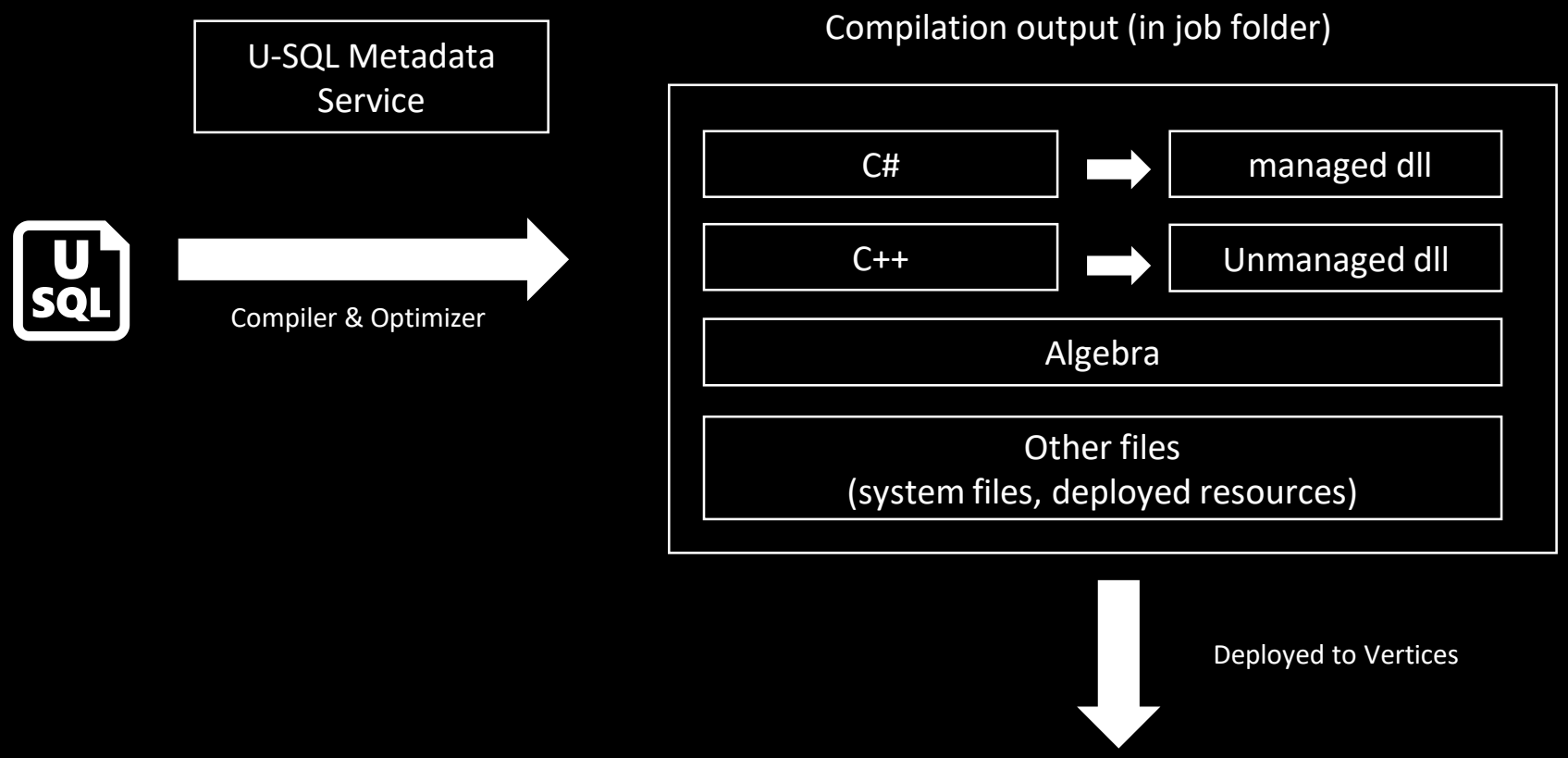
Job Result	Succeeded
Total Duration	10.8 minutes
Submit Time	1/24/2016 10:14:52 AM
Start Time	1/24/2016 10:15:48 AM
End Time	1/24/2016 10:25:42 AM
Compilation	37 seconds
Queued	0 seconds
Running	9.9 minutes
Account	datainsightsadhoc
Author	BD Telemetry Service@SPI

UX

Preparing	New	The script is being compiled by the Compiler Service
	Compiling	
Queued	Queued	All jobs enter the queue.
	Scheduling	Are there enough ADLAUs to start the job?
	Starting	If yes, then allocate those ADLAUs for the job
Running	Running	The U-SQL runtime is now executing the code on 1 or more ADLAUs or finalizing the outputs
Finalizing		
Ended (Succeeded, Failed, Cancelled)	Ended	The job has concluded.

U-SQL Job Compilation

U-SQL Compilation Process



The Job Folder

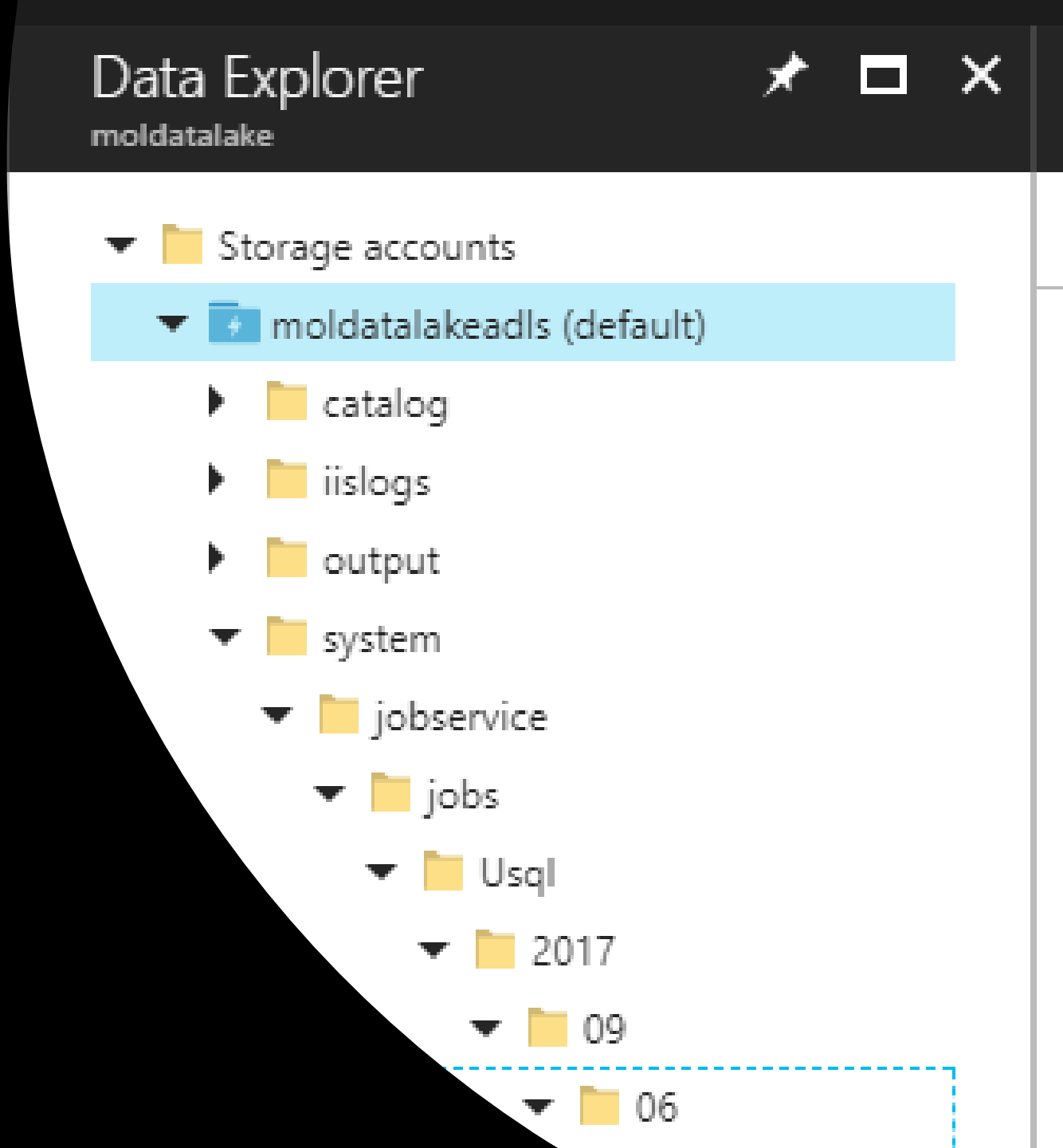
Inside the Default ADL Store:

/system/jobservice/jobs/Usql/YYYY/MM/DD/hh/mm/JOBID

/system/jobservice/jobs/Usql/2016/01/20/00/00/17972fc2-4737-48f7-81fb-49af9a784f64

Job folder Structure

- /system/jobservice/jobs/Usql/YYYY/MM/DD/hh/mm/JOBID



ADLAUs

Azure
Data
Lake
Analytics
Unit

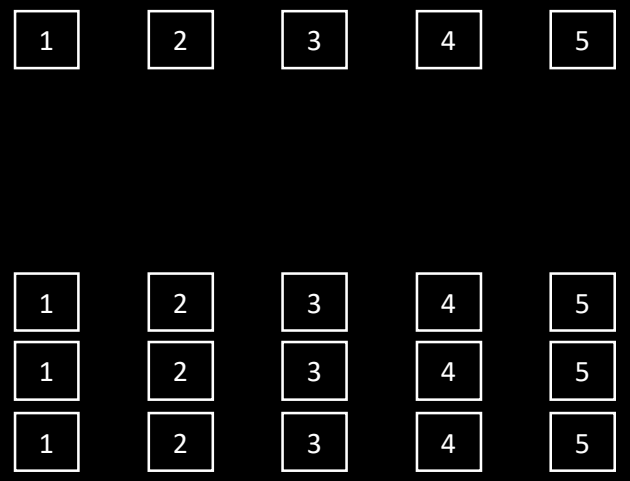
Parallelism N = N ADLAUs

1 ADLAU ~=
A VM with 2 cores and 6 GB of memory

Vertex Execution

Store Basics

A VERY BIG FILE



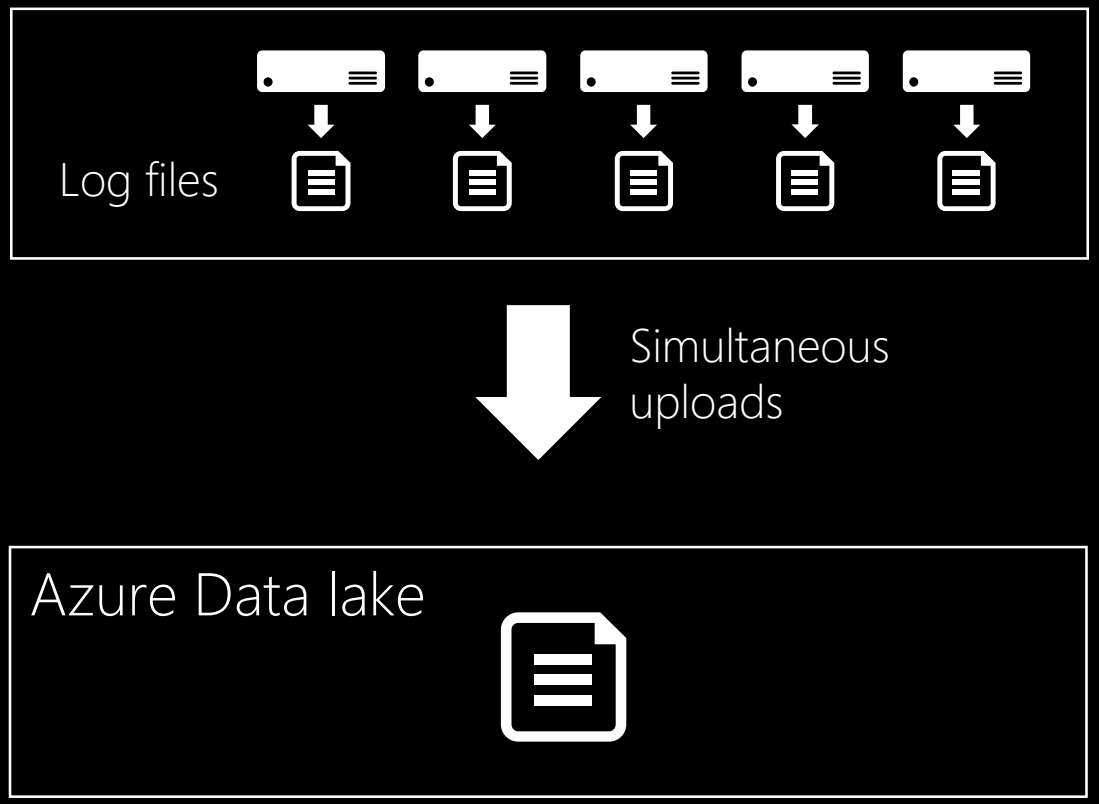
Files are split apart into **Extents**.

Extents can be up to 250MB in size.

For availability and reliability, extents are replicated (3 copies).

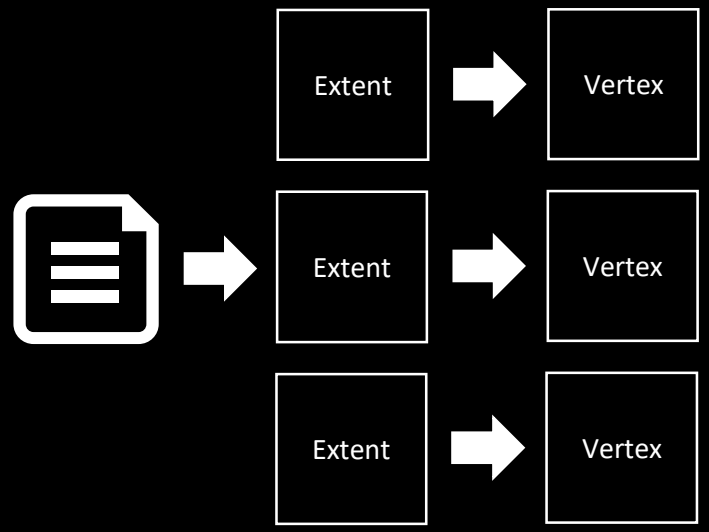
Enables parallelized read

Parallel writing



Front-end machines for a web service

As file size increases, more opportunities for parallelism



ADLA File Management

Hands on

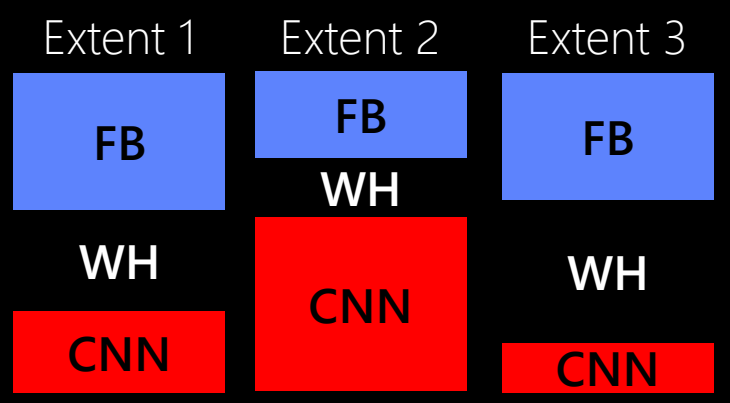
The importance of partitioning input data

Search engine clicks data set

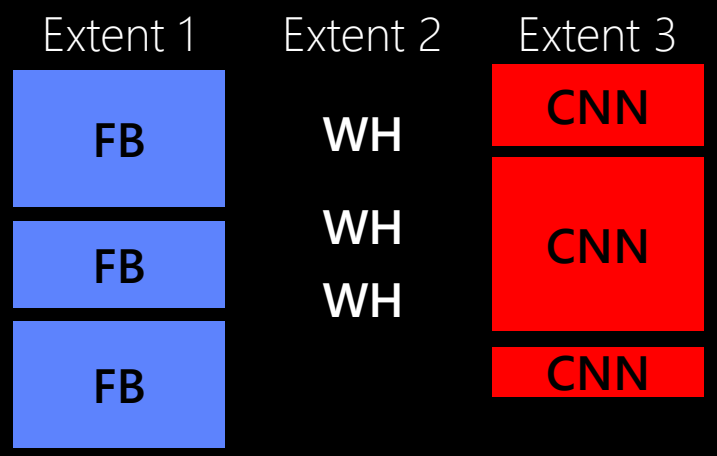
A log of how many clicks a certain domain got within a session

SessionID	Domain	Clicks
3	cnn.com	9
1	whitehouse.gov	14
2	facebook.com	8
3	reddit.com	78
2	microsoft.com	1
1	facebook.com	5
3	microsoft.com	11

Data Partitioning Compared



File:
Keys (Domain) are scattered across the extents



U-SQL Table partitioned on Domain
The keys are now “close together”
also the index tells U-SQL exactly which extents contain the key

Click Data :

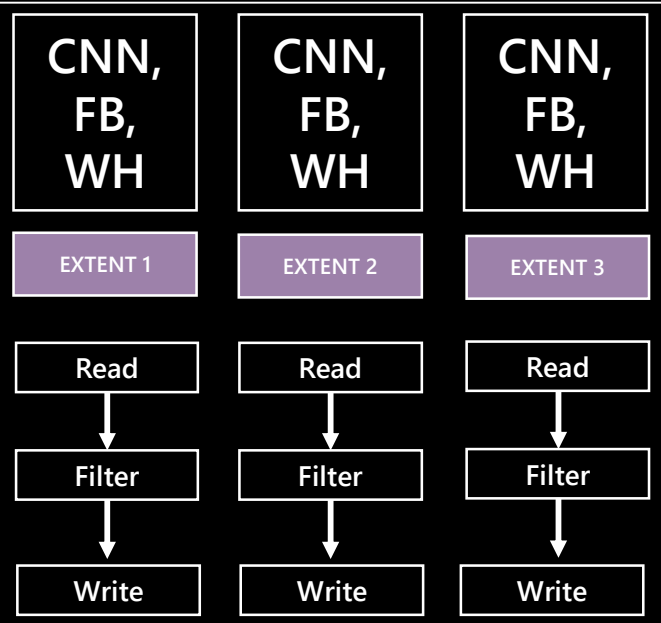
```
CREATE TABLE SampleDBTutorials.dbo.ClickData
(
    SessionId     int,
    Domain        string,
    Clinks        int,
  INDEX idx1 //Name of index
  CLUSTERED (Domain ASC) //Column to cluster by

);


INSERT INTO SampleDBTutorials.dbo.ClickData
SELECT *
FROM @clickdata;
```

Find all the rows for cnn.com

```
// Using a File

@ClickData =
    SELECT
      Session int,
      Domain string,
      Clicks int
FROM “/clickdata.tsv”
USING Extractors.Tsv();

@rows = SELECT *
    FROM @ClickData
    WHERE Domain == “cnn.com”;

OUTPUT @rows
    TO “/output.tsv”
    USING Outputters.tsv();
```

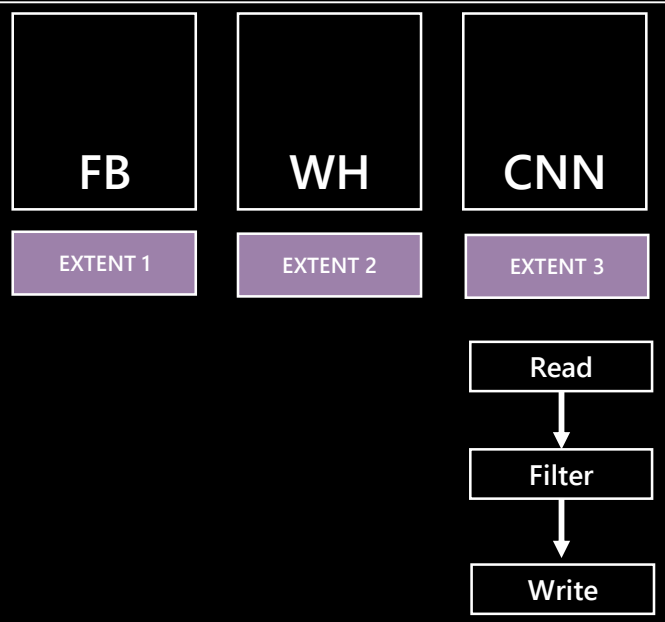
```
// Using a U-SQL Table partitioned by Domain

@ClickData =
    SELECT *
    FROM MyDB.dbo.ClickData;

@rows = SELECT *
    FROM @ClickData
    WHERE Domain == “cnn.com”;

OUTPUT @rows
    TO “/output.tsv”
    USING Outputters.tsv();
```

File



Because "CNN" could be anywhere, all extents must be read.

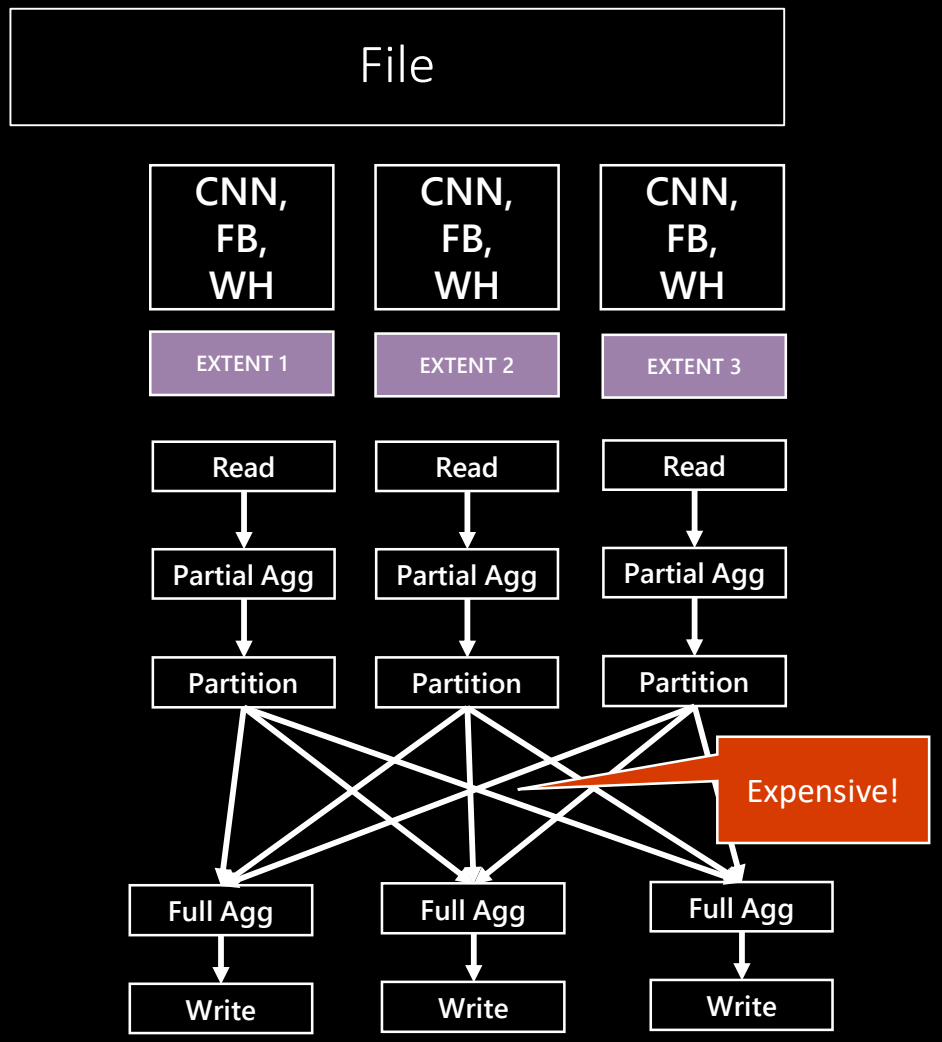
U-SQL Table Partitioned by Domain

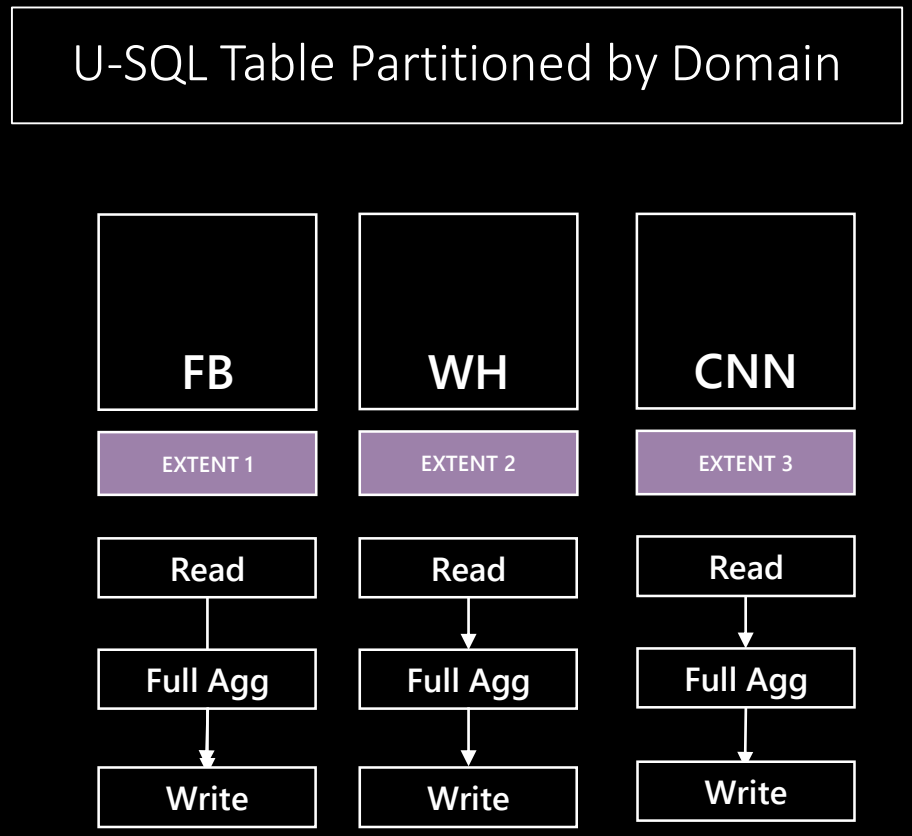


Thanks to "Partition Elimination" and the U-SQL Table, the job only reads from the extent that is known to have the relevant key

How many clicks per domain?

```
@rows = SELECT
               Domain,
               SUM(Clicks) AS TotalClicks
        FROM @ClickData
        GROUP BY Domain;
```


File
CNN, FB, WH
CNN, FB, WH
CNN, FB, WH
EXTENT 1
EXTENT 2
EXTENT 3
Read
Read
Read
Partial Agg
Partial Agg
Partial Agg
Partition
Partition
Partition
Expensive!
Full Agg
Full Agg
Full Agg
Write
Write
Write


U-SQL Table Partitioned by Domain
FB
WH
CNN
EXTENT 1
EXTENT 2
EXTENT 3
Read
Read
Read
Full Agg
Full Agg
Full Agg
Write
Write
Write

Hands on Lab -ADLA

Creating Data Catalog Objects

How about a Break !!!

Data Factory

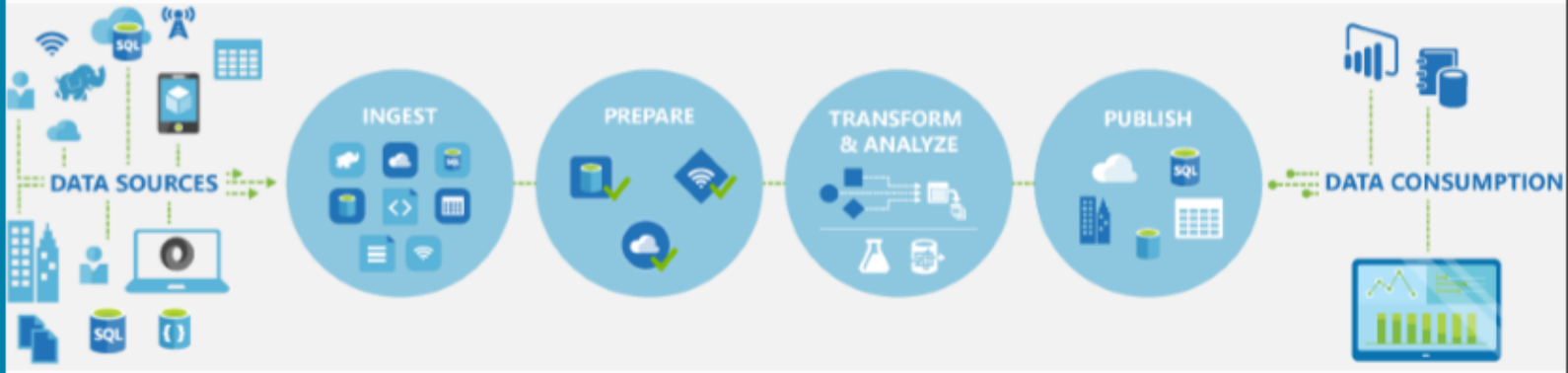
INTRODUCTION

AZURE DATA FACTORY

What is **Azure Data Factory**

Introduction to Azure Data Factory Service, a data integration service in the cloud

Data Factory is a cloud-based data integration service that orchestrates and automates the movement and trans formation of data.



You can create **data integration** solutions using the **Data Factory** service that can ingest data from various data stores, **transform/process** the data, and publish the result data to the data stores.

Azure Data Factory

Enables enterprises

To ingest data from multiple on-premises and cloud sources easily, and gets your data where it needs to go. Prepare and partition your data as you ingest it, or apply pre-processing steps.

Allows you to create data pipelines

That move and transform data, and then run the pipelines on a specified schedule (hourly, daily, weekly, etc.).

Provides rich visualizations

To display the lineage and dependencies between your data pipelines, and monitor all your data pipelines from a single unified view.

Produce trusted data

What can We use Data Factory for?

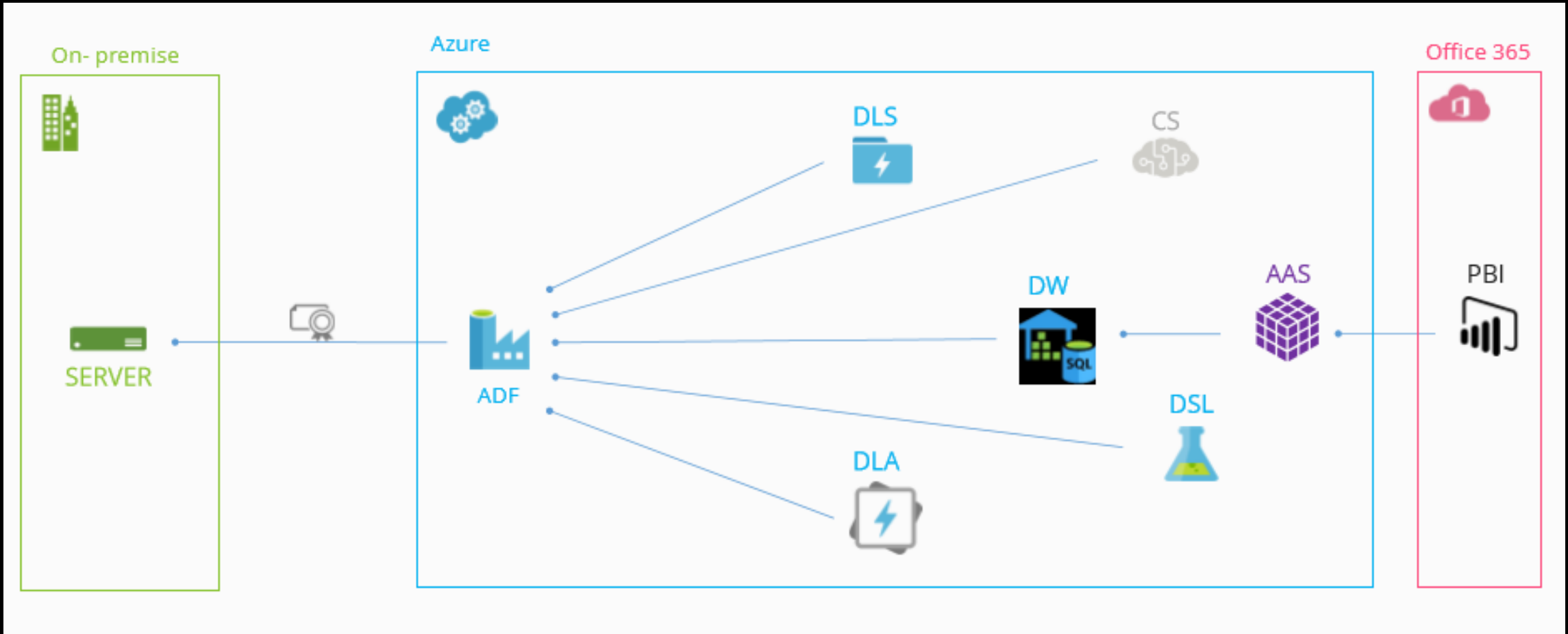
Use it to **ingest data** from multiple on-premises and cloud sources.

Schedule, orchestrate, and manage the **data transformation** and analysis process.

Transform raw data into finished or shaped data that's **ready for consumption** by BI tools or by your on-premises or cloud applications and services.

Manage your entire network of **data pipelines** at a glance to identify issues and take action

Data Sources Usage



What is **Pipeline**

A pipeline is a logical grouping of activities. They are used to group activities into a unit that together perform a task.

To understand pipelines better, you need to understand an activity first.

What is **Activity**

Activities define the actions to perform on your data. For example, you may use a Copy activity to copy data from one data store to another data store. Similarly, you may use a Hive activity, which runs a Hive query on an Azure HDInsight cluster to transform or analyze your data. You may also choose to create a custom .NET activity to run your own code.


PIPELINE
Activity 1
Activity 2
Activity 3

Activity Types

Data Factory supports two types of activities: and.

Data movement activities

Copy Activity in Data Factory copies data from a source data store to a sink data store. Data from any source can be written to any sink.

Data transformation activities

Data Transformation Activity transforms data to desired format and shape. Transformation activities that can be added to pipelines either individually or chained with another activity.

Linked Services

Linked services define the information needed for Data Factory to connect to external resources (Examples: Azure Storage, on-premises SQL Server, Azure HDInsight).

Linked **Services**

Linked services are used for two purposes in Data Factory:

To represent a data store

including, but not limited to, an on-premises SQL Server, Oracle database, file share, or an Azure Blob Storage account.

To represent a compute resource

that can host the execution of an activity. For example, the HDInsight Hive activity runs on an HDInsight Hadoop cluster.

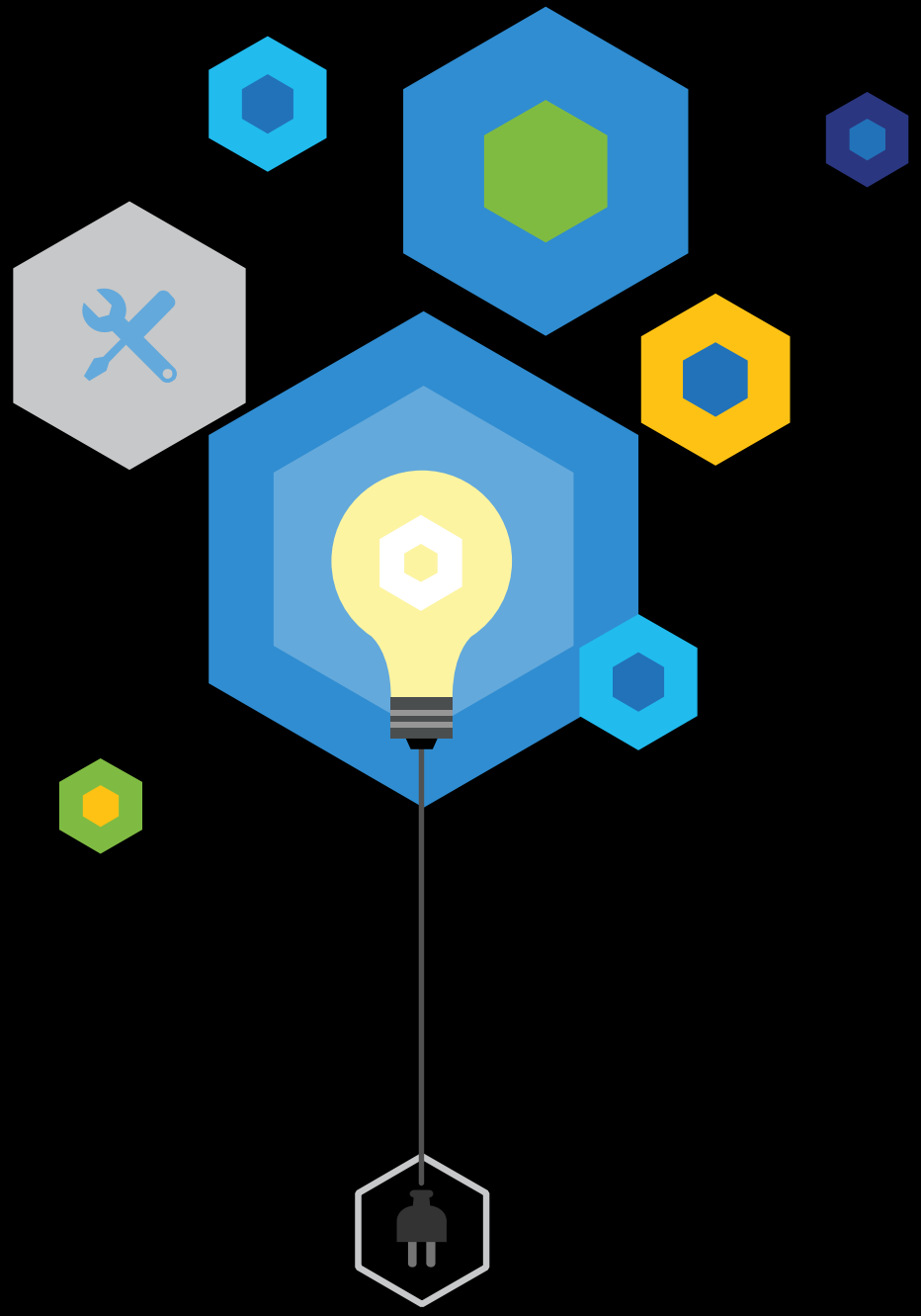
Connecting To The Cloud **Data Source**

Linked Services

Datasets

Linked services link data stores to an Azure data factory. Datasets represent data structures with in the data stores. For example, an Azure Storage linked service provides connection information for Data Factory to connect to an Azure Storage account. An Azure Blob dataset specifies the blob container and folder in the Azure Blob Storage from which the pipeline should read the data. Similarly, an Azure SQL linked service provides connection information for an Azure SQL database and an Azure SQL dataset specifies the table that contains the data.

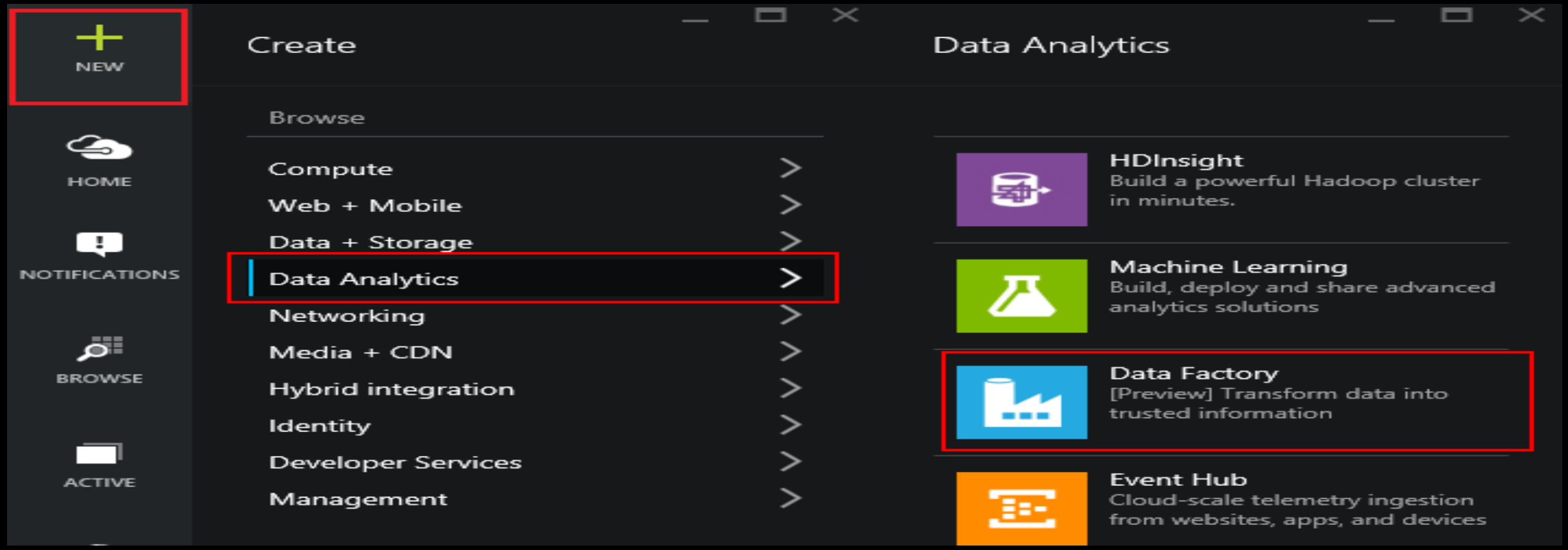
2. Create the Data Factory



Portal, PowerShell and Visual Studio

Using the Portal

- Use in Non-MS Clients
- Use for Exploration
- Use when teaching or in a Demo



ADF V2

Concepts

Data Source

Dataset

Pipeline

is a grouping of logically related activities. It is used to group activities into a unit that performs a task

Concepts

Activities define the actions to perform on your data. Each activity takes zero or more datasets as inputs and produces one or more datasets as output.

Activity

Linked services

computing environment

Expressions & Parameters

String functions – concat, substring, replace, indexof etc.

Collection functions – length, union, first, last etc.

Logic functions – equals, less than, greater than, and, or, not etc.

Conversation functions – coalesce, xpath, array, int, string, json etc.

Math functions – add, sub, div, mod, min, max etc.

Date functions – utcnow, addminutes, addhours, format etc.

System variables

Pipeline scope

Variable Name
@pipeline().DataFactory
@pipeline().Pipeline
@pipeline().RunId
@pipeline().TriggerType
@pipeline().TriggerId
@pipeline().TriggerName
@pipeline().TriggerTime

Trigger scope

Variable Name
trigger().scheduledTime
trigger().startTime

Trigger

Type of triggers

- Manual execution
- Schedule trigger: A trigger that invokes a pipeline on a wall-clock schedule.
- Tumbling window trigger: A trigger that operates on a periodic interval, while also retaining state.

Schedule trigger

A schedule trigger runs pipelines on a wall-clock schedule. This trigger supports periodic and advanced calendar options. For example, the trigger supports intervals like "weekly" or "Monday at 5:00 PM and Thursday at 9:00 PM."

```
{
    "properties": {
        "name": "MyTrigger",
        "type": "ScheduleTrigger",
        "typeProperties": {
            "recurrence": {
                "frequency": "Hour",
                "interval": 1,
                "startTime": "2017-11-01T09:00:00-08:00",
                "endTime": "2017-11-02T22:00:00-08:00"
            }
        },
        "pipelines": [{
                "pipelineReference": {
                    "type": "PipelineReference",
                    "referenceName": "SQLServerToBlobPipeline"
                },
                "parameters": {}
            },
            {
                "pipelineReference": {
                    "type": "PipelineReference",
                    "referenceName": "SQLServerToAzureSQLPipeline"
                },
                "parameters": {}
            }
        ]
    }
}
```

Tumbling window trigger

Tumbling window triggers are a type of trigger that fires at a periodic time interval from a specified start time, while retaining state. Tumbling windows are a series of fixed-sized, non-overlapping, and contiguous time intervals.

```
{
  "name": "PerfTWTrigger",
  "properties": {
    "type": "TumblingWindowTrigger",
    "typeProperties": {
      "frequency": "Minute",
      "interval": "15",
      "startTime": "2017-09-08T05:30:00Z",
      "delay": "00:00:01",
      "retryPolicy": {
        "count": 2,
        "intervalInSeconds": 30
      },
      "maxConcurrency": 50
    },
    "pipeline": {
      "pipelineReference": {
        "type": "PipelineReference",
        "referenceName": "DynamicsToBlobPerfPipeline"
      },
      "parameters": {
        "windowStart": "@trigger().outputs.windowStartTime",
        "windowEnd": "@trigger().outputs.windowEndTime"
      }
    },
    "runtimeState": "Started"
  }
```

Control flow

Control flow

ForEach activity

Filter activity

Get metadata

Execute Pipeline

If Condition activity

Web activity

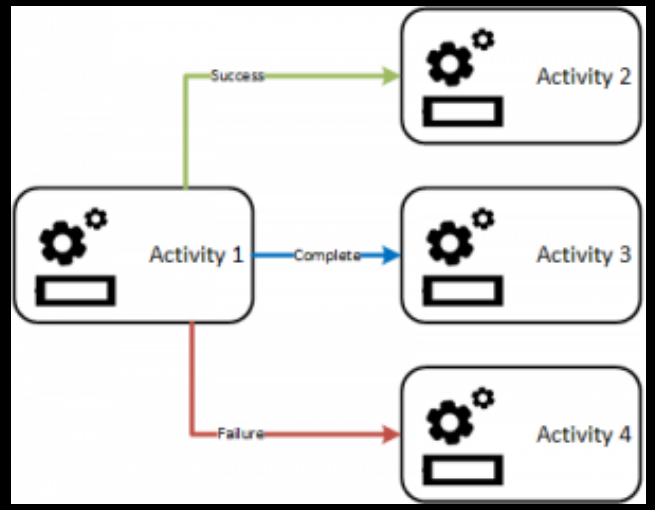
Lookup activity

Wait activity

Until activity

Branching

- On success
- On failure
- On completion
- On skip



Hands on Usecase

Usecase

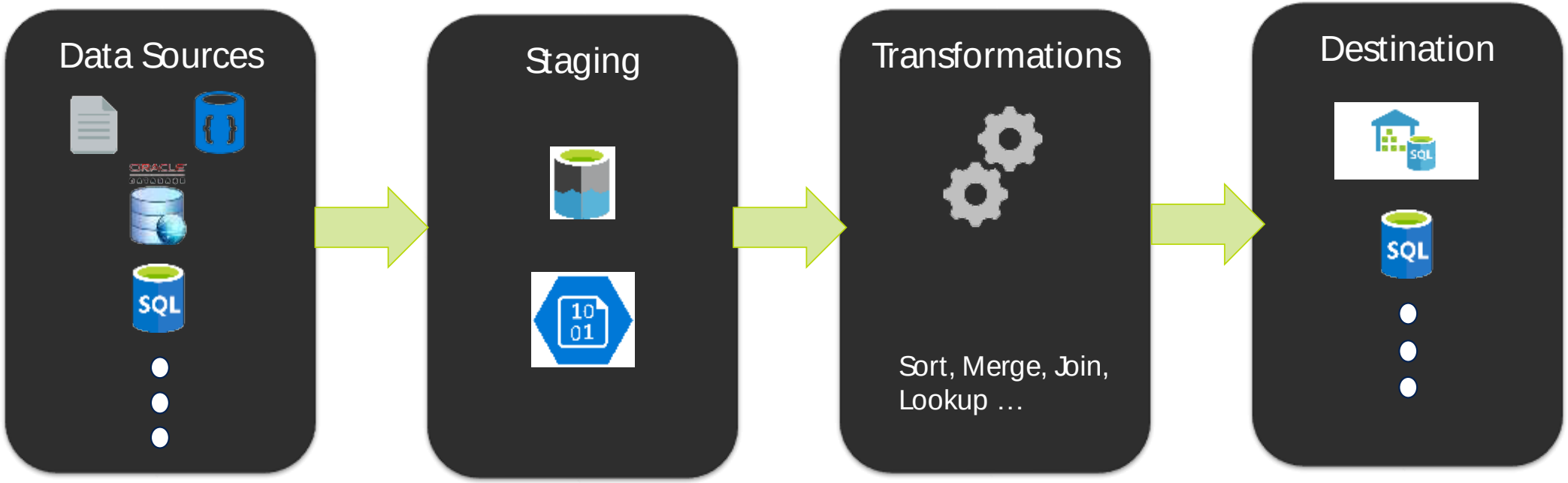
- One of the Manufacturing client needs data coming from multiple sources to be automated to generate reports and even send the email at the end.
- This should be a one time creation of pipeline but any new source integration should not impact the pipeline process.

Build 2019

Whats New !!!


TripData
Import data from
JoinMatchedData
Inner join on TripData and TripFare
AggregateDayStats
Aggregating data by DayOfTheWeek producing columns trip_distance, passenger_count
DayStatsSink
Export data to TaxiDemoDayStatsSink
TripFare
Import data from
JoinMatchedData
Inner join on TripData and TripFare
AggregateVendorStats
Aggregating data by vendor_id producing columns passenger_count, trip_time_in_secs, trip_distance
VendorStatsSink
Export data to TaxiDemoVendorStatsSink
TripFare
Import data from
AggregateByPaymentTy...
Aggregating data by payment_type producing columns fare_amount_total, surcharge_total, mta_tax_total
TotalPaymentByPaymen...
Export data to TaxiDemoTotalByPaymentType

ADF Data Flow Workstream

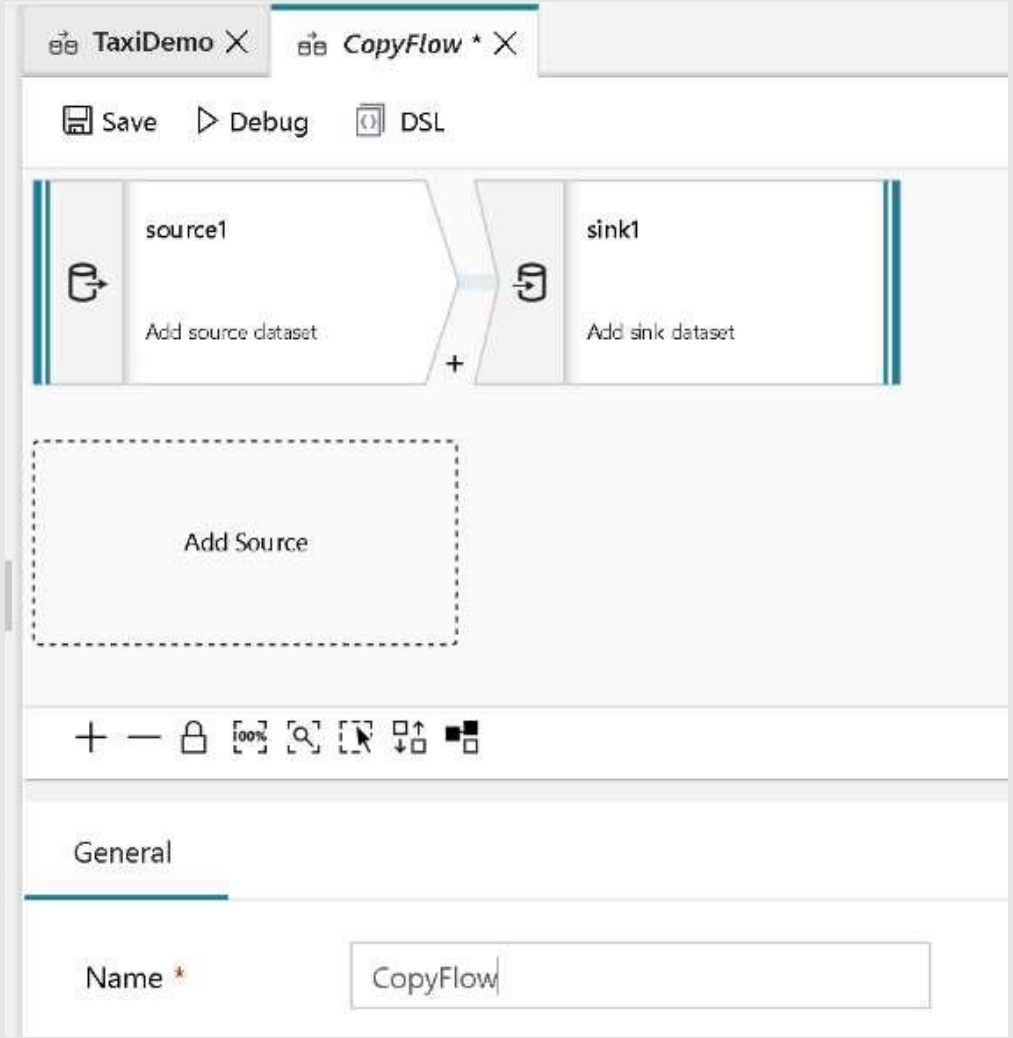


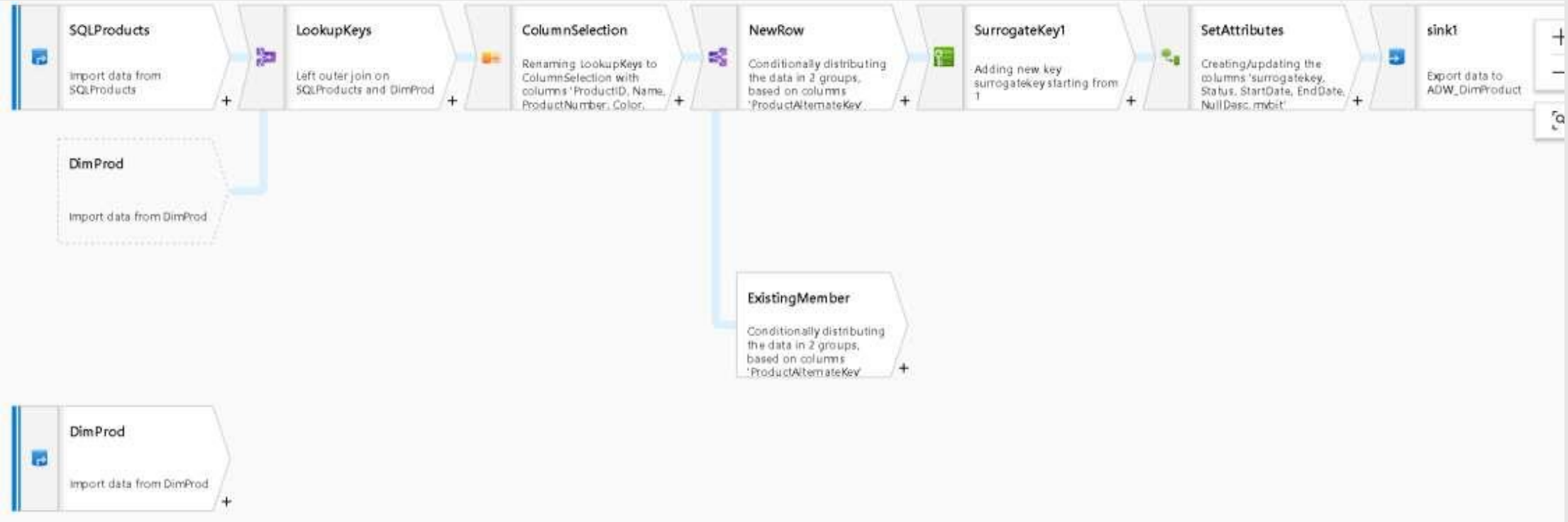
- Explicit user action
- User places data source(s) on design surface, from toolbox
- Select explicit sources

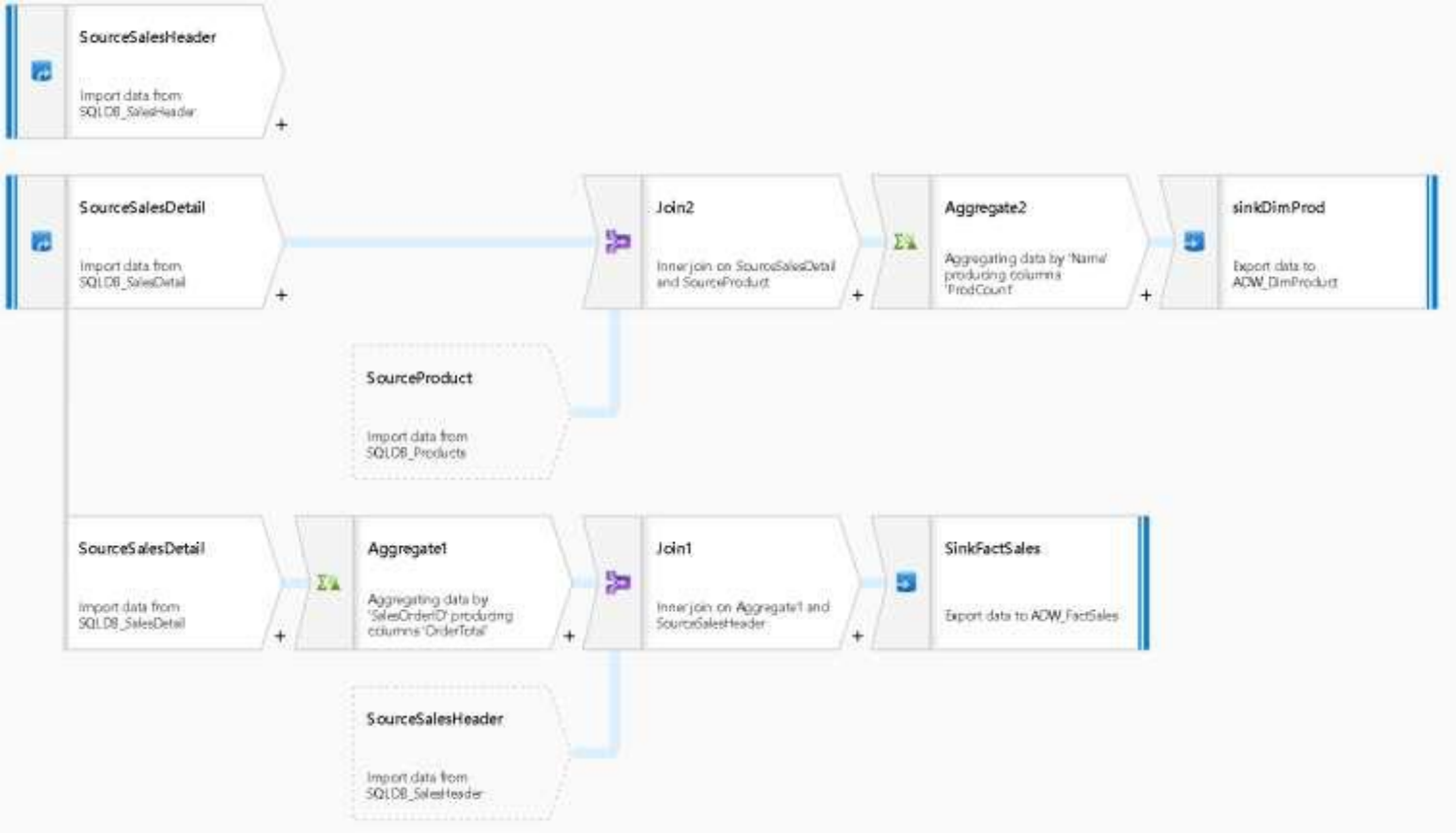
- Implicit/Explicit
- Data Lake staging area as default
- User does not need to configure this manually
- Advanced feature to set staging area options
- File Formats / Types (Parquet, JSON, txt, CSV …)

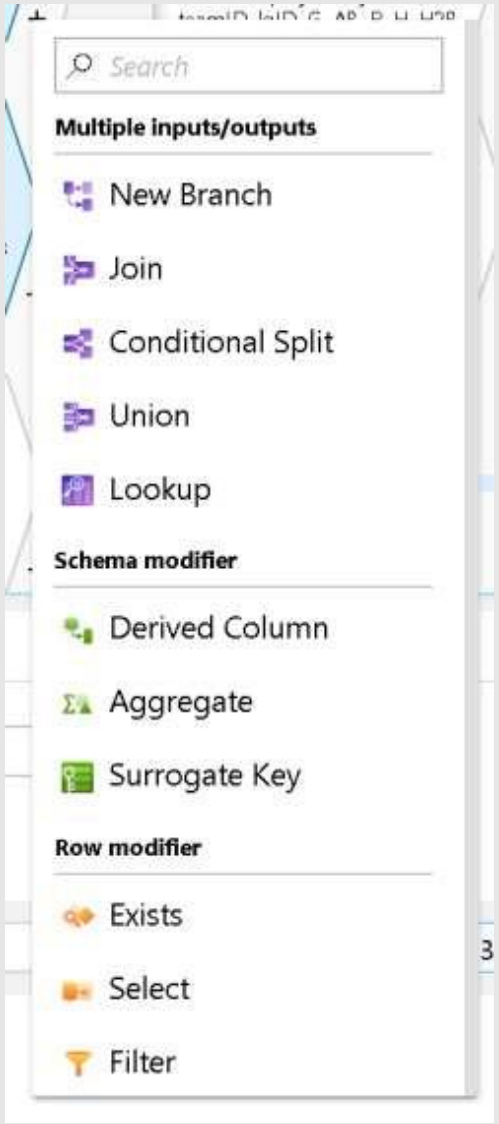
- Explicit user action
- User places transformations on design surface, from toolbox
- User must set properties for transformation steps and step connectors

- Explicit user action
- User chooses destination connector(s)
- User sets connector property options

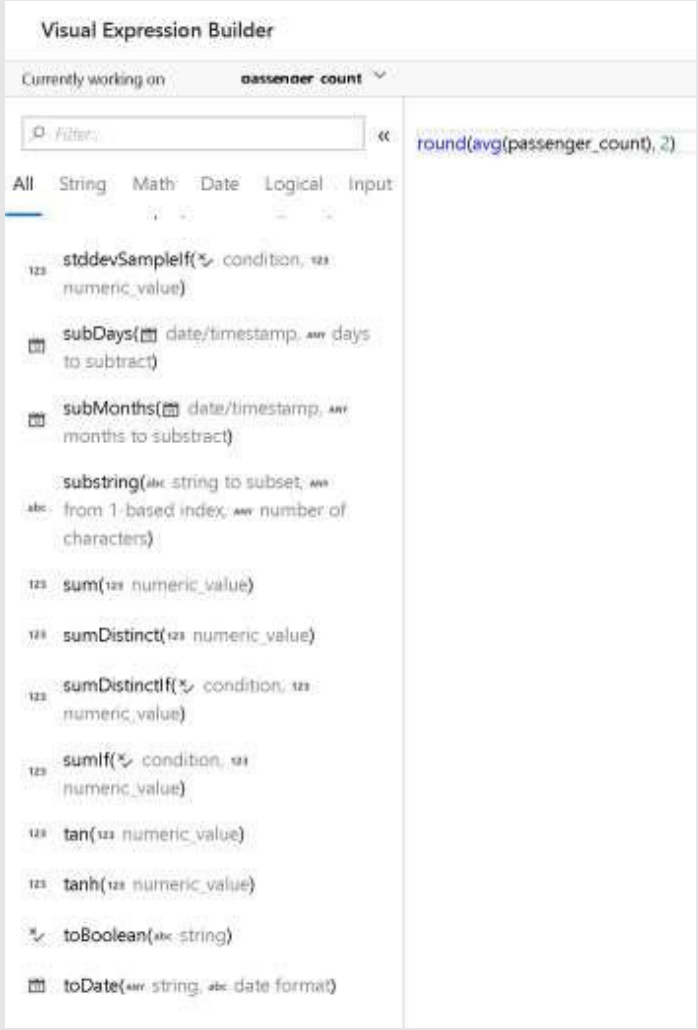

TaxiDemo
CopyFlow *
Save
Debug
DSL
source1
Add source dataset
sink1
Add sink dataset
Add Source
General
Name *
CopyFlow

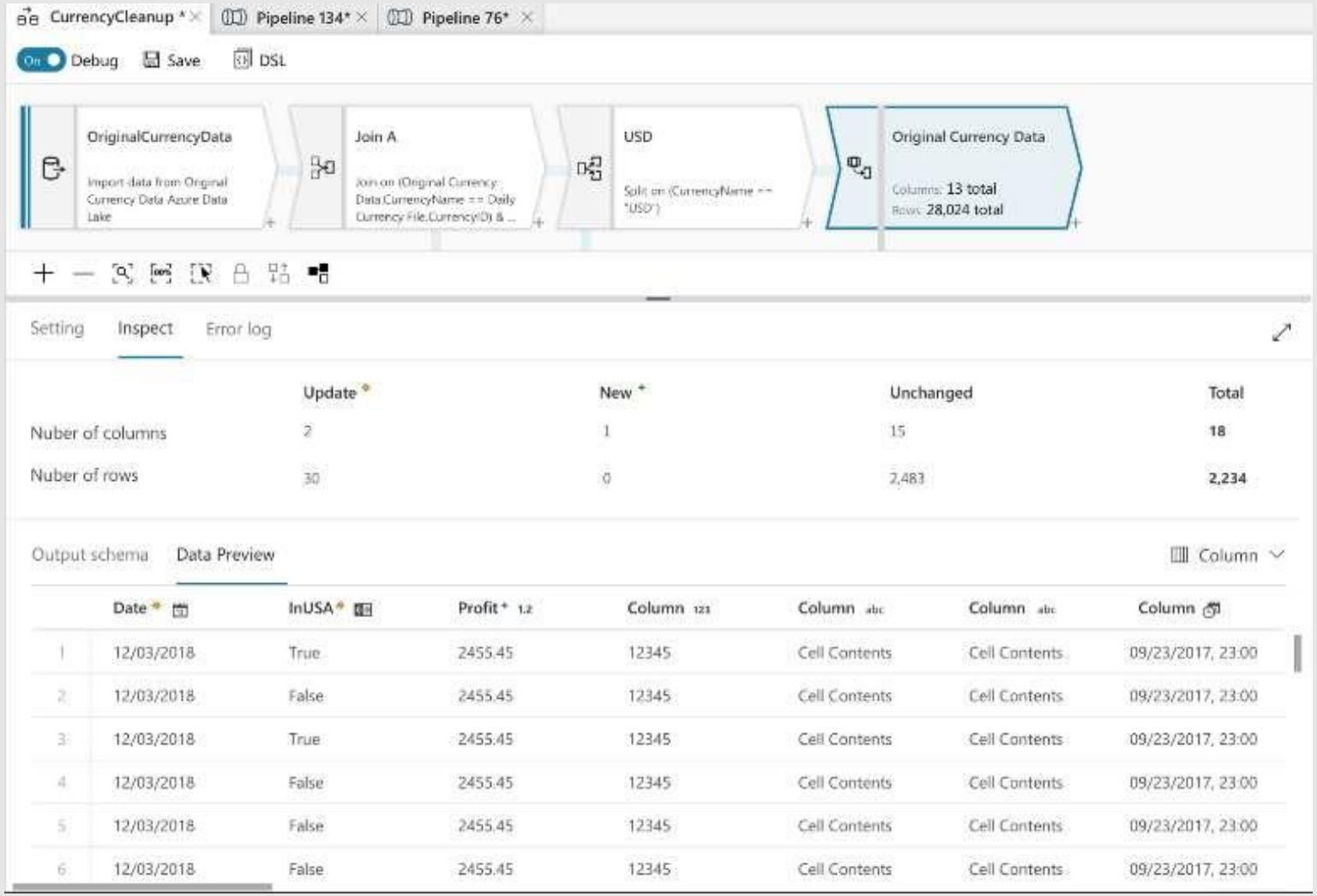

SQLProducts
Import data from SQLProducts
LookupKeys
Left outer join on SQLProducts and DimProd
ColumnSelection
Renaming LookupKeys to ColumnSelection with columns 'ProductID, Name, ProductNumber, Color,
NewRow
Conditionally distributing the data in 2 groups, based on columns 'ProductAlternateKey'
SurrogateKey1
Adding new key surrogatekey starting from 1
SetAttributes
Creating/updating the columns 'surrogatekey, Status, StartDate, EndDate, NullDesc, mybit'
sink1
Export data to ADW_DimProduct
DimProd
Import data from DimProd
ExistingMember
Conditionally distributing the data in 2 groups, based on columns 'ProductAlternateKey'
DimProd
Import data from DimProd

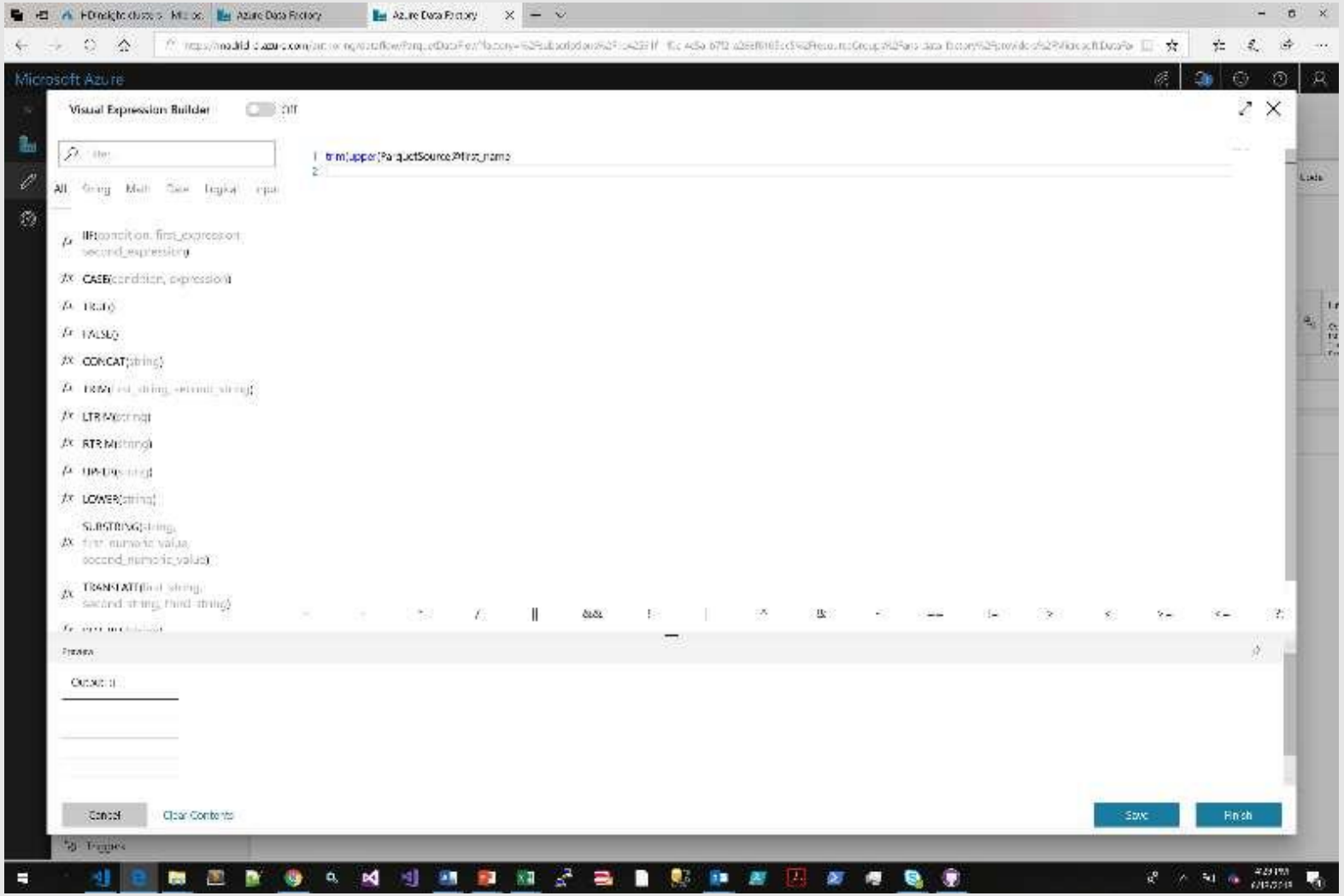

SourceSalesHeader
Import data from SQLDB_SalesHeader
SourceSalesDetail
Import data from SQLDB_SalesDetail
Join2
Inner join on SourceSalesDetail and SourceProduct
Aggregate2
Aggregating data by 'Name' producing columns 'ProdCount'
sinkDimProd
Export data to ADW_DimProduct
SourceProduct
Import data from SQLDB_Products
SourceSalesDetail
Import data from SQLDB_SalesDetail
Aggregate1
Aggregating data by 'SalesOrderID' producing columns 'OrderTotal'
Join1
Inner join on Aggregate1 and SourceSalesHeader
SinkFactSales
Export data to ADW_FactSales
SourceSalesHeader
Import data from SQLDB_SalesHeader

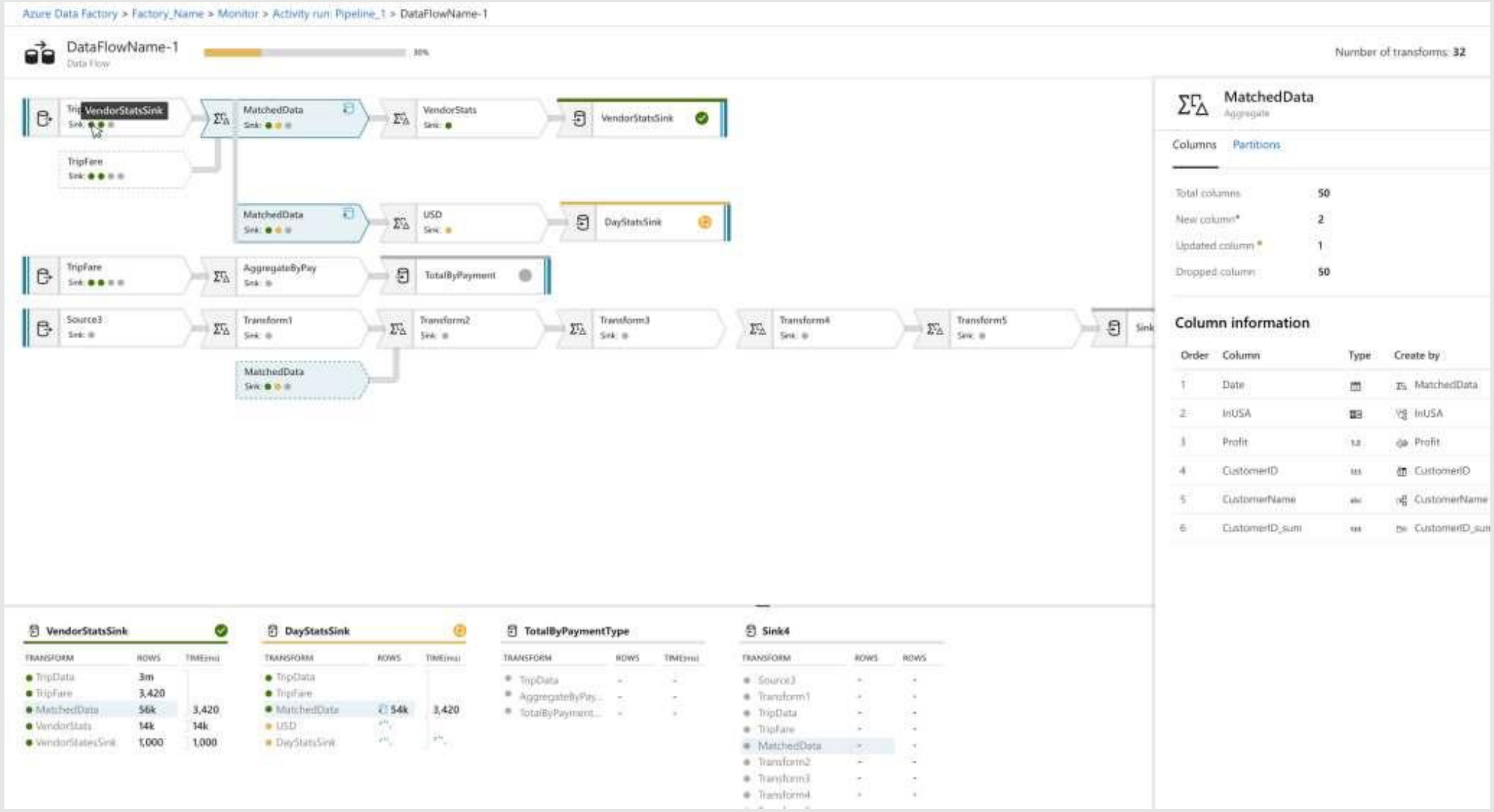

Search
Multiple inputs/outputs
New Branch
Join
Conditional Split
Union
Lookup
Schema modifier
Derived Column
Aggregate
Surrogate Key
Row modifier
Exists
Select
Filter


sourceBatting
CalcTotalBases
23 total
OrigSourceBatting
CalcTotalBases
createAggs
Join1
sink1
sourcePlayer
PlayerInfoPlusOrigStats
OrigSourceBatting


Visual Expression Builder
Currently working on
passenger_count
Filter...
All
String
Math
Date
Logical
Input
stddevSampleIf(condition, numeric_value)
subDays(date/timestamp, days to subtract)
subMonths(date/timestamp, months to substract)
substring(string to subset, from 1-based index, number of characters)
sum(numeric_value)
sumDistinct(numeric_value)
sumDistinctIf(condition, numeric_value)
sumIf(condition, numeric_value)
tan(numeric_value)
tanh(numeric_value)
toBoolean(string)
toDate(string, date format)
round(avg(passenger_count), 2)


CurrencyCleanup *
Pipeline 134*
Pipeline 76*
On
Debug
Save
DSL
OriginalCurrencyData
Import data from Original Currency Data Azure Data Lake
Join A
Join on (Original Currency Data.CurrencyName == Daily Currency File.CurrencyID) & ...
USD
Split on (CurrencyName == 'USD')
Original Currency Data
Columns: 13 total
Rows: 28,024 total
Setting
Inspect
Error log
Update
New
Unchanged
Total
Nuber of columns
2
1
15
18
Nuber of rows
30
0
2,483
2,234
Output schema
Data Preview
Column
Date
InUSA
Profit
Column
Column
Column
Column
1 12/03/2018 True 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00
2 12/03/2018 False 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00
3 12/03/2018 True 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00
4 12/03/2018 False 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00
5 12/03/2018 False 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00
6 12/03/2018 False 2455.45 12345 Cell Contents Cell Contents 09/23/2017, 23:00


Microsoft Azure
Visual Expression Builder
trim(upper(ParquetSource.Prior_name))
All
IIF(condition, first_expression, second_expression)
CASE(condition, expression)
TRUE()
FALSE()
CONCAT(string)
LTRIM(string)
RTRIM(string)
UPPER(string)
LOWER(string)
SUBSTRING(string, first_numeric_value, second_numeric_value)
TRANSLATE(first_string, second_string, third_string)
Preview
Cancel
Clear Contents
Save
Finish


Azure Data Factory > Factory_Name > Monitor > Activity run: Pipeline_1 > DataFlowName-1
DataFlowName-1
Data Flow
Number of transforms: 32
VendorStatsSink
MatchedData
VendorStats
VendorStatsSink
TripFare
MatchedData
USD
DayStatsSink
TripFare
AggregateByPay
TotalByPayment
Source3
Transform1
Transform2
Transform3
Transform4
Transform5
MatchedData
MatchedData
Aggregate
Columns
Partitions
Total columns 50
New column* 2
Updated column * 1
Dropped column 50
Column information
Order Column Type Create by
1 Date MatchedData
2 InUSA InUSA
3 Profit Profit
4 CustomerID CustomerID
5 CustomerName CustomerName
6 CustomerID_sum CustomerID_sum
VendorStatsSink
TRANSFORM ROWS TIME(ms)
TripData 3m
TripFare 3,420
MatchedData 56k 3,420
VendorStats 14k 14k
VendorStatesSink 1,000 1,000
DayStatsSink
TRANSFORM ROWS TIME(ms)
TripData
TripFare
MatchedData 54k 3,420
USD
DayStatsSink
TotalByPaymentType
TRANSFORM ROWS TIME(ms)
TripData
AggregateByPay...
TotalByPayment...
Sink4
TRANSFORM ROWS ROWS
Source3
Transform1
TripData
TripFare
MatchedData
Transform2
Transform3
Transform4

	Update	New	Unchanged	Total
Nuber of columns	2	1	15	**18**
Nuber of rows	30	0	2,483	**2,234**

Output schema | Data Preview

Order	Date	Column	Profit	Column	Continent_data	Column
1	12/03/2018	✓	2455.45	Cell Contents	North America	Cell Contents
2	12/03/2018	✓	2455.45	Cell Contents	Europe	Cell Contents
3	12/03/2018		2455.45	Cell Contents	North America	Cell Contents
4	12/03/2018		2455.45	Cell Contents	North America	Cell Contents
5	12/03/2018	✓	2455.45	Cell Contents	Africa	Cell Contents
6	12/03/2018	✓	2455.45	Cell Contents	Asia	Cell Contents
7	12/03/2018	✓	null	Cell Contents	Europe	Cell Contents
8	12/03/2018		null	Cell Contents	North America	Cell Contents
9	12/03/2018		49582.23	Cell Contents	Asia	Cell Contents
10	12/03/2018		49582.23	Cell Contents	Asia	Cell Contents
11	12/03/2018	✓	null	Cell Contents	Europe	Cell Contents
12	12/03/2018		49582.23	Cell Contents	Europe	Cell Contents
13	12/03/2018		49582.23	Cell Contents	North America	Cell Contents

Continent_data abc

528	51.8%	North America
203	20.3	Asia
136	13.6%	Africa
116	11.6%	Europe
17	1.7%	America
10	1.0%	Other

Content Type	String
Missing Value	0 (0%)
Unique Values	5 (5%)
Most Common	North America (52.8%)
Skewness	-0.043210436

Future is most promising for ADF

Data Flows are in Public Preview since Build 2019. (Since 1 Week)

Hands on Usecase

Usecase

- One of the Manufacturing client needs data coming from multiple sources to be automated to generate reports and even send the email at the end.
- This should be a one time creation of pipeline but any new source integration should not impact the pipeline process.