U-SQL Production Analytics

With Azure Data Lake

Paul Andrew | Data Analytics Consultant & Data Platform MVP

Microsoft MVP Most Valuable Professional

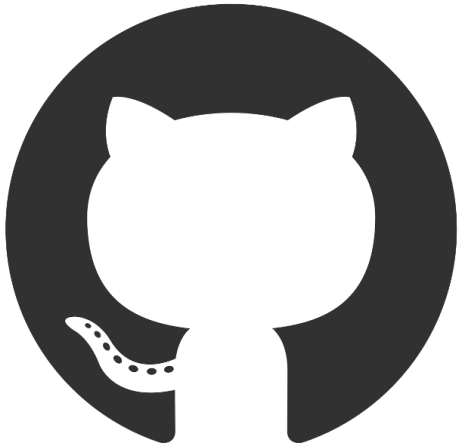
@MrPaulAndrew

Microsoft Partner
Microsoft
Gold Data Analytics
Gold Data Platform
Gold Cloud Platform

adatis

In appreciation of our sponsors

GitHub



https://github.com/mrpaulandrew

CommunityEvents

Demo code, content and slides from various community events.

C++

Agenda

Azure Data Lake What & Why	What is U-SQL	U-SQL as a Framework	Meta Data Driven U-SQL
Storage & Compute	A Quick Recap	Scale Out .Net, R & Python	Production Code Generation

Agenda

Azure Data Lake What & Why	What is U-SQL	U-SQL as a Framework	Meta Data Driven U-SQL
Storage & Compute	A Quick Recap	Scale Out .Net, R & Python	Production Code Generation

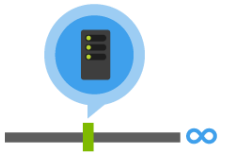
What is Azure Data Lake?

Analytics

U-SQL

Processing Engine

Node Node Node Node Node Node Node Node Node Node Node Node Node Node Node Node



Analytics Units (AU's) – Scale Out Compute

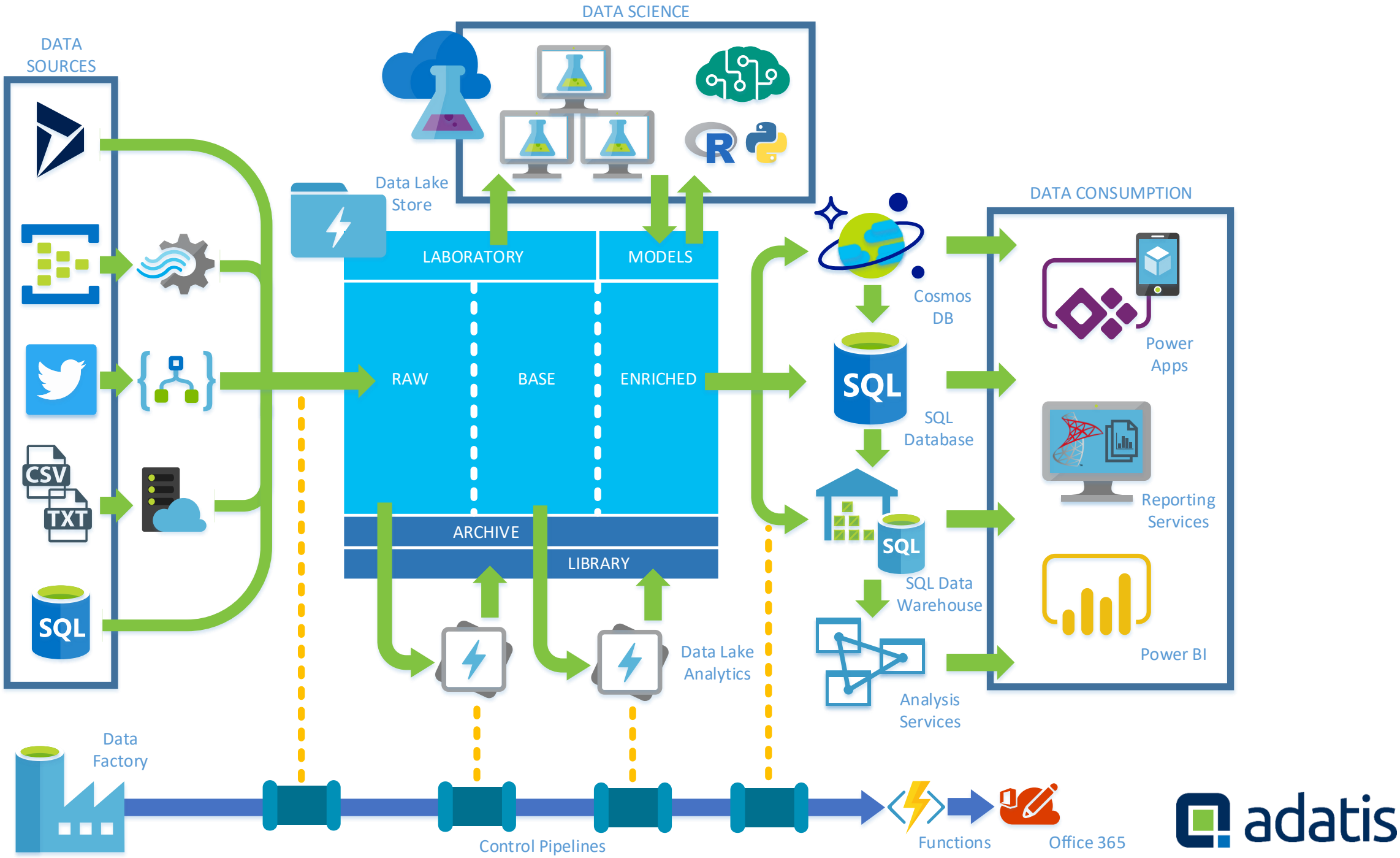
Storage

Data Storage (HDFS)

Structured Semi-Structured Unstructured

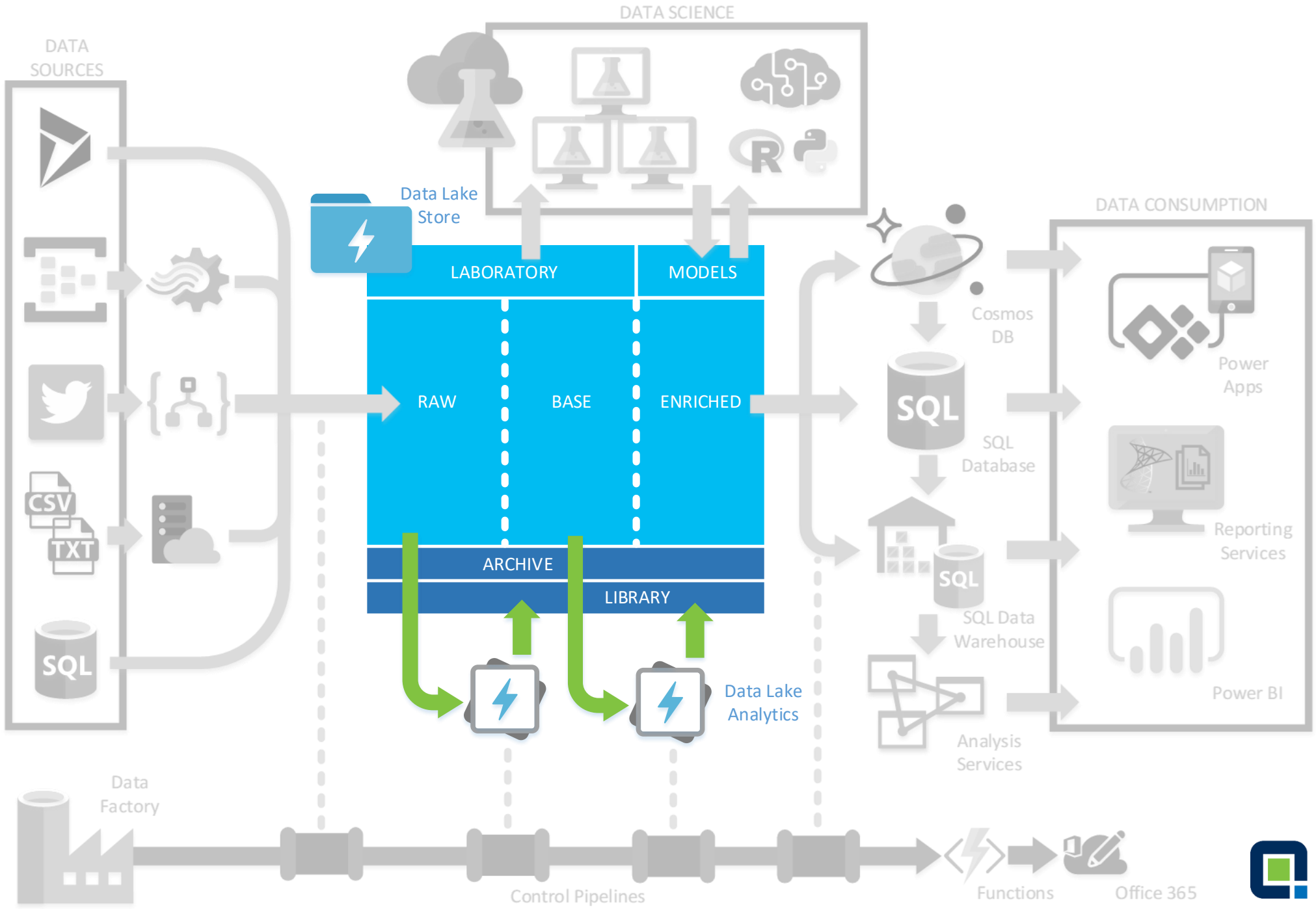
https://blogs.msdn.microsoft.com/azuredatalake/2016/10/12/understanding-adl-analytics-unit/

The Modern Data Analytics Platform



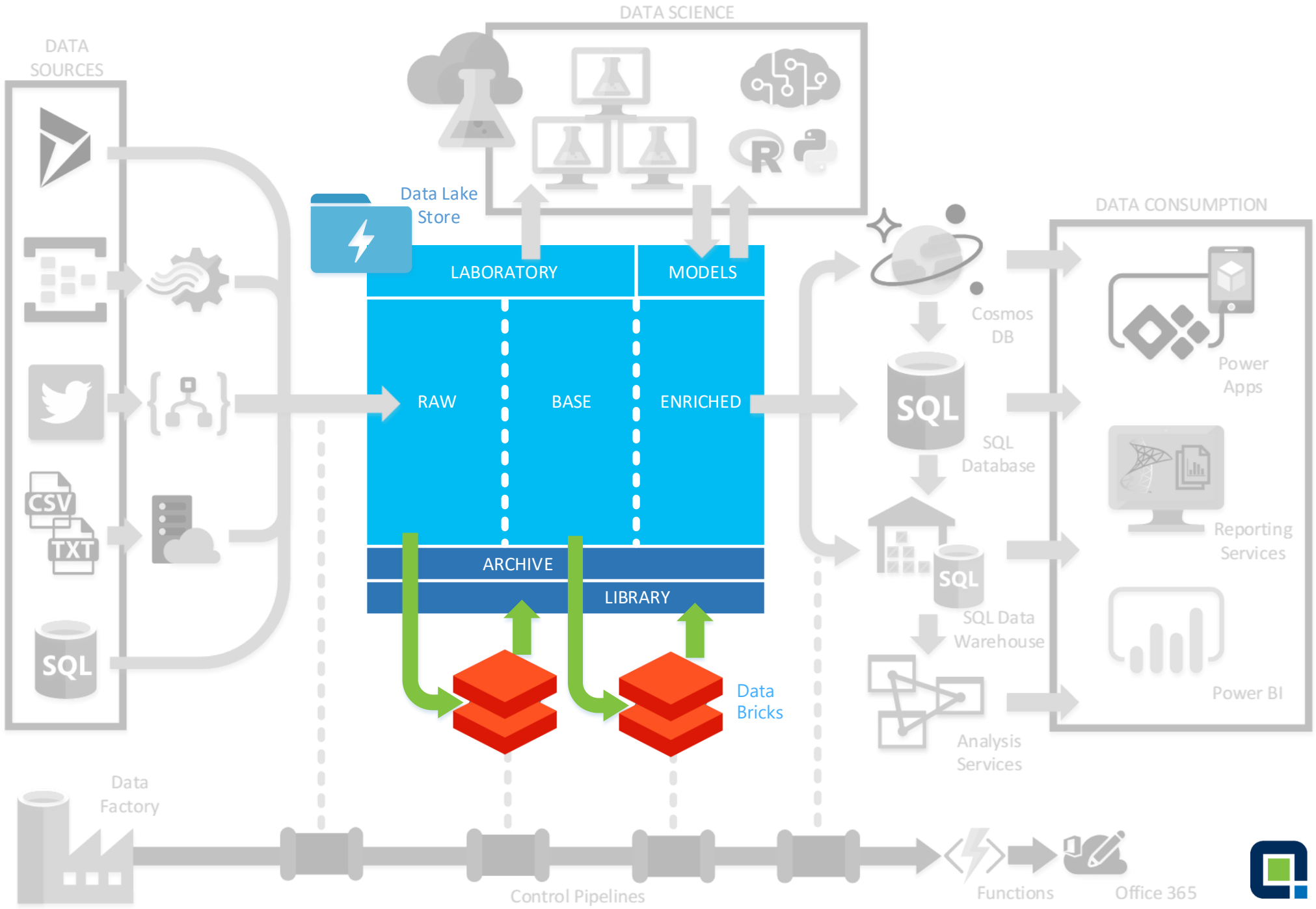
adatis

The Modern Data Analytics Platform



adatis

The Modern Data Analytics Platform

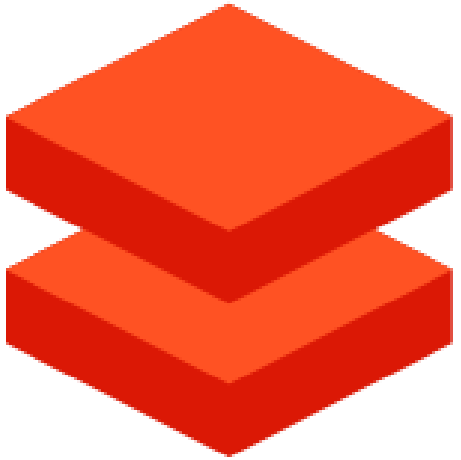


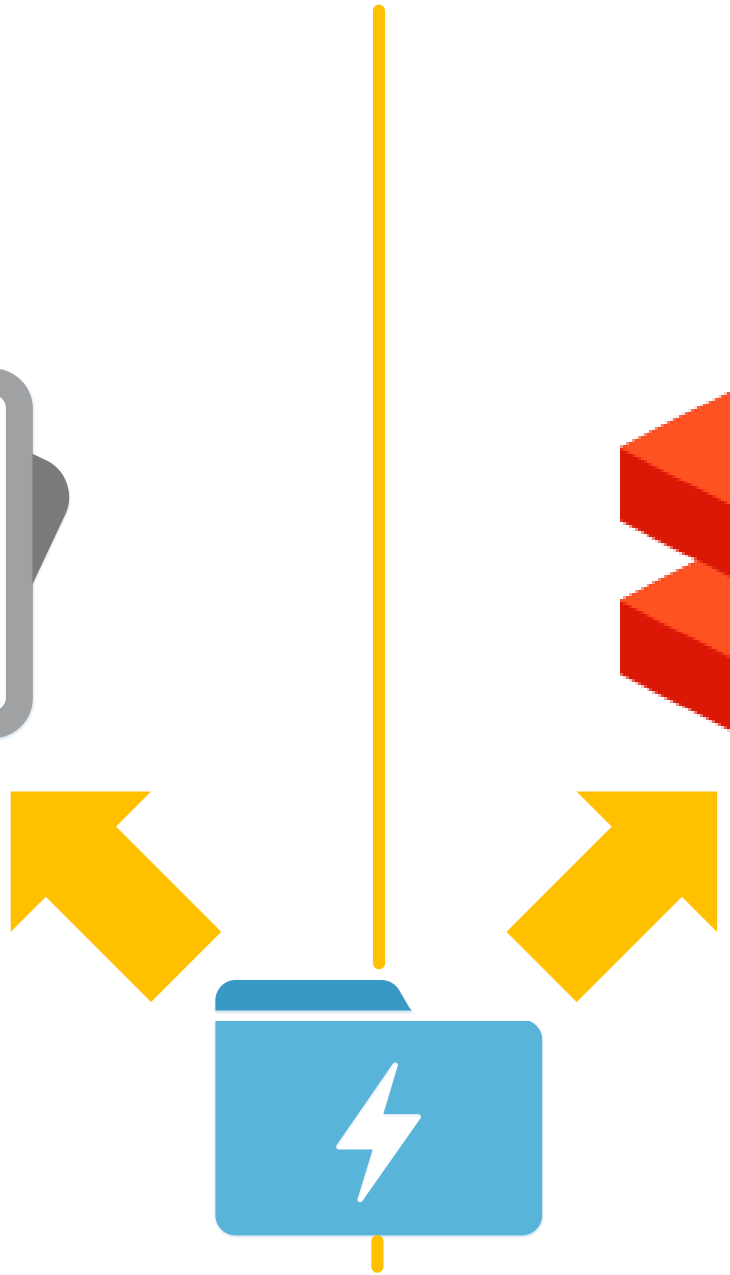
adatis

Azure Data Lake Analytics vs Azure Databricks

T-SQL
C#

Python
Scala

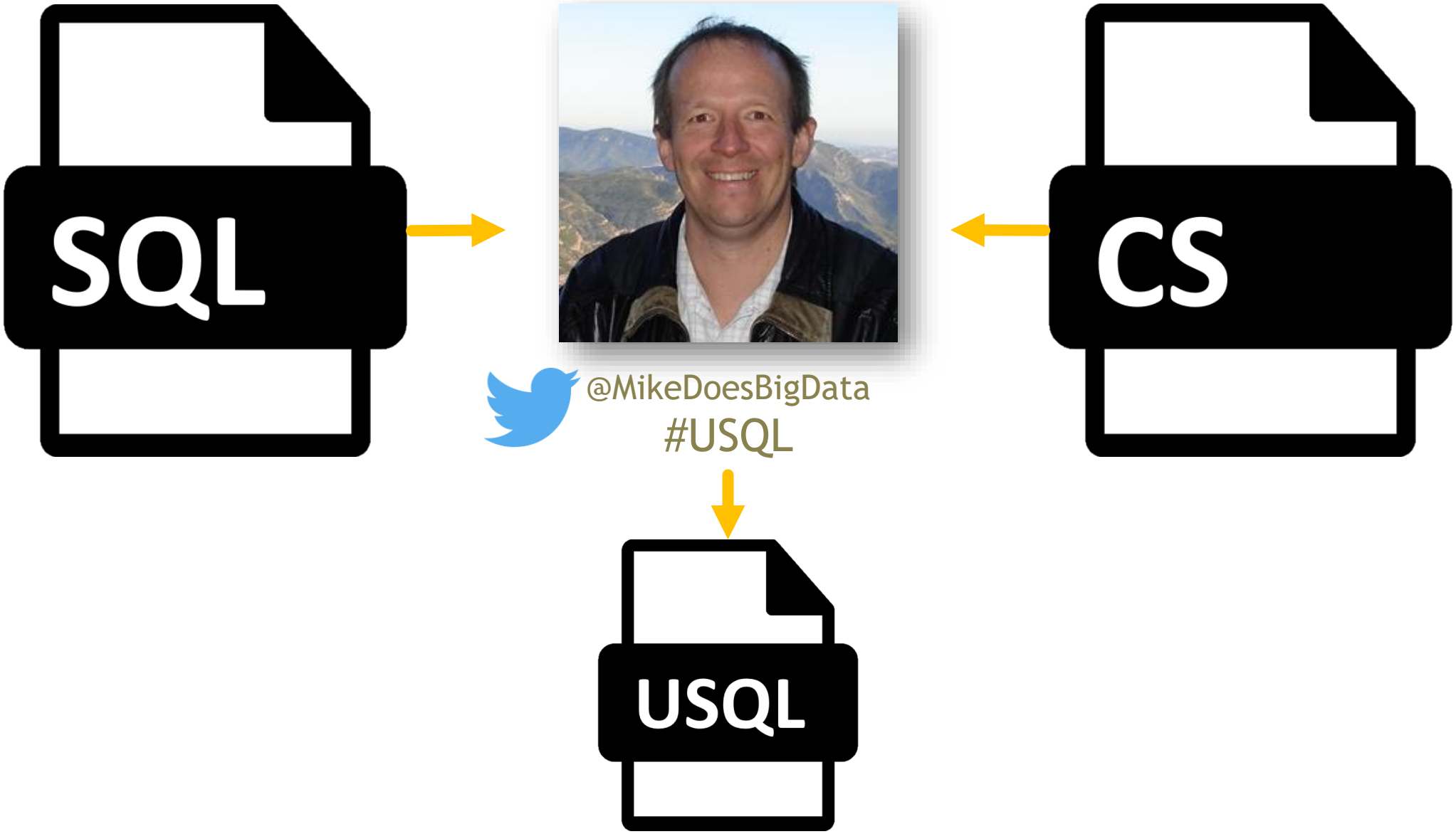




Agenda

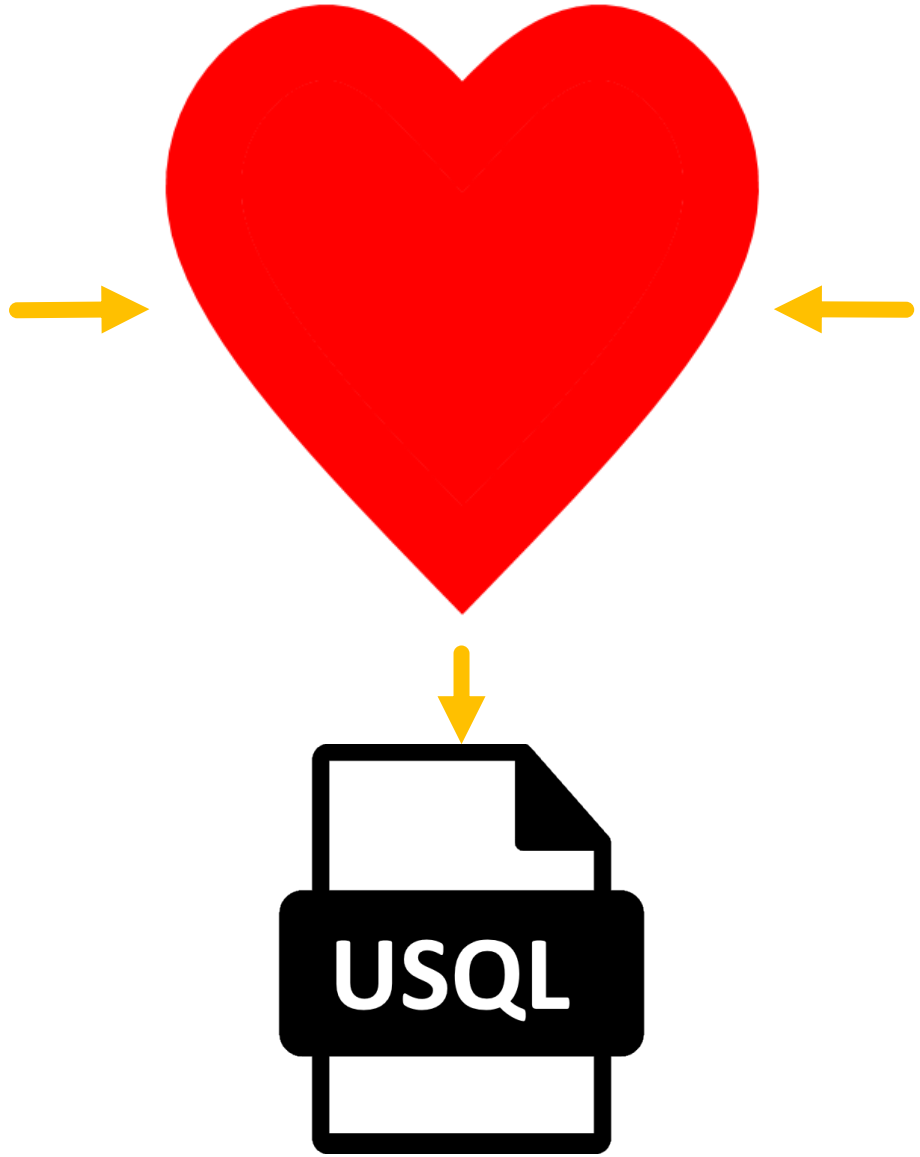
Azure Data Lake What & Why	What is U-SQL	U-SQL as a Framework	Meta Data Driven U-SQL
Storage & Compute	A Quick Recap	Scale Out .Net, R & Python	Production Code Generation

What is U-SQL?



What is U-SQL?

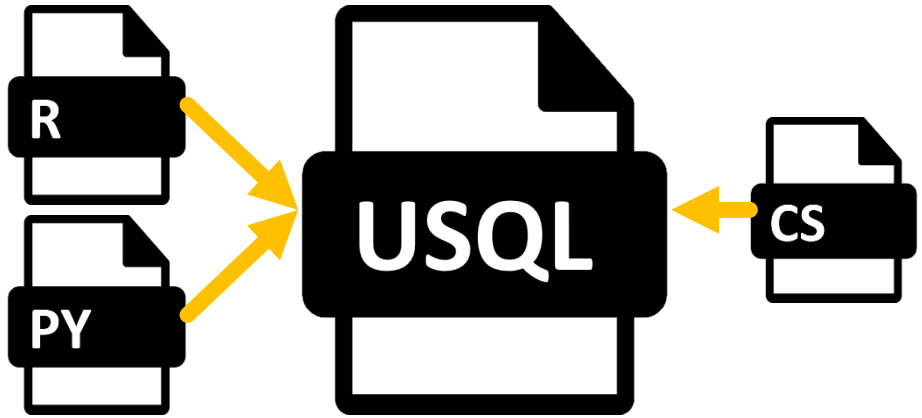
```
SELECT
    Domain,
    COUNT(*) AS Qty
FROM
    @Domains
GROUP BY
    Domain;
```




```
using System;

namespace USQLSampleApplication
{
    0 references | 0 changes | 0 authors, 0 changes
    public class CustomMethods
    {
        0 references | 0 changes | 0 authors, 0 changes
        static public int testMethod()
        {
            string testString = String.Empty;
            int testInt = Int32.MinValue;

            return testInt;
        }
    }
}
```

What is U-SQL?

```
@SizeAndCount =
    SELECT
        [ModifiedDate].ToString("yyyy") AS Year,
        [FileName].Substring([FileName].IndexOf(".") + 1, 3) AS FileExtension,
        COUNT(0) AS RecordCount,
        Math.Ceiling(Convert.ToDecimal(SUM([Size]))) AS FileSizeTotalsMB,
        Math.Ceiling(Convert.ToDecimal(SUM([Size])/1024)) AS FileSizeTotalsGB
    FROM
        @Raw
    WHERE
        [ActualFileName] == "FileDetailsTest.csv"
    GROUP BY
        [ModifiedDate].ToString("yyyy"),
        [FileName].Substring([FileName].IndexOf(".") + 1, 3);
```

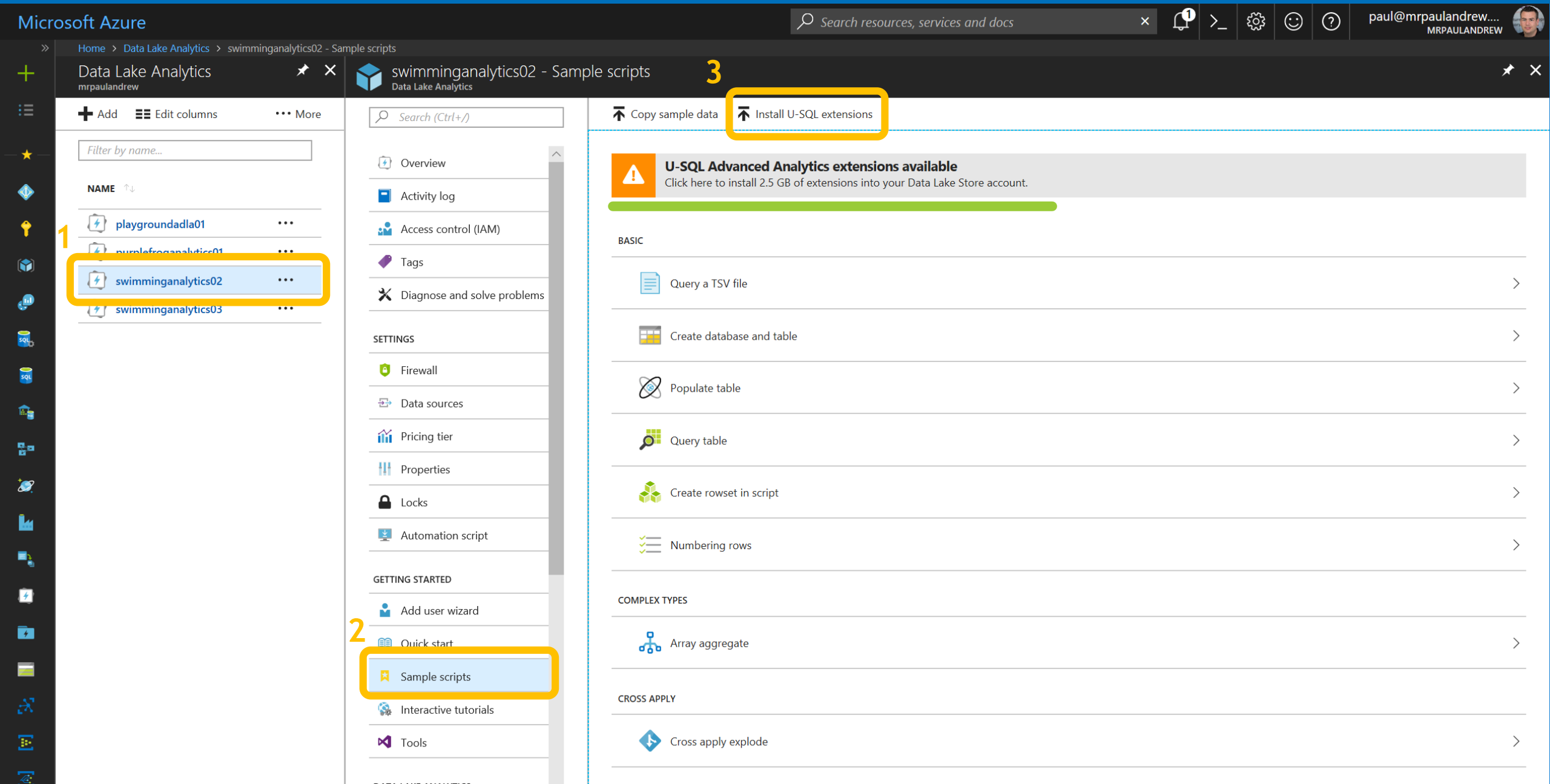


Agenda

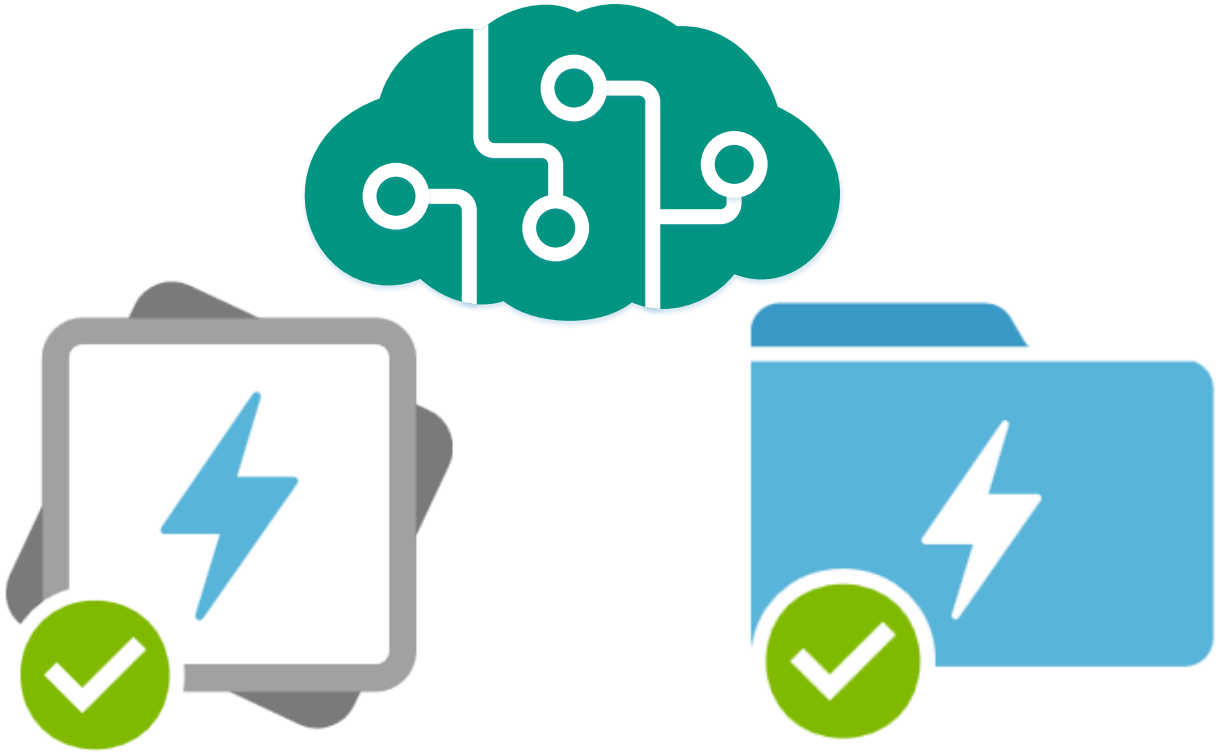
Azure Data Lake What & Why	What is U-SQL	U-SQL as a Framework	Meta Data Driven U-SQL
Storage & Compute	A Quick Recap	Scale Out .Net, R & Python	Production Code Generation

U-SQL As A Framework

Getting the U-SQL Extensions

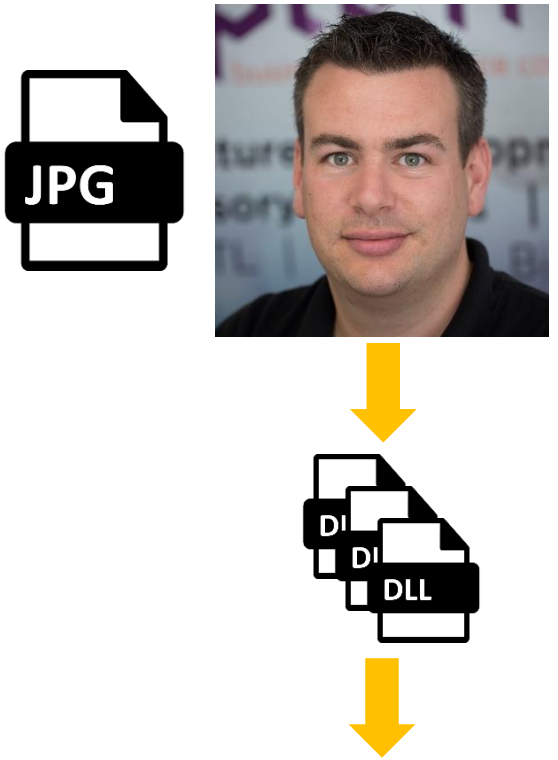


Azure Data Lake with Cognitive Services



U-SQL Image Tagging with Cognitive Services

```
// Load Assemblies
REFERENCE ASSEMBLY ImageCommon;
REFERENCE ASSEMBLY FaceSdk;
REFERENCE ASSEMBLY ImageEmotion;
REFERENCE ASSEMBLY ImageTagging;
REFERENCE ASSEMBLY ImageOcr;

// Load in images
@imgs =
    EXTRACT FileName string, ImgData byte[]
    FROM @"/Images/{FileName}.jpg"
    USING new Cognition.Vision.ImageExtractor();

//Tagging processor
@tags_from_processor =
    PROCESS @imgs
    PRODUCE FileName, NumObjects int, Tags SQL.MAP<string, float?>
    READONLY FileName USING new Cognition.Vision.ImageTagger();

@tags_from_processor_serialized =
    SELECT
        FileName,
        NumObjects,
        String.Join
        ("|", Tags.Select(x => String.Format("{0}", x.Key))) AS TagsString
    FROM
        @tags_from_processor;

//Output
OUTPUT @tags_from_processor_serialized
TO @"/Output/FileTags.csv"
USING Outputters.Csv(outputHeader : true);
```



	A	B	C
1	FileName	NumObjects	TagsString
2	Me	8	black|indoor|looking|male|man|person|posing|staring
3			

Scale Out Cognitive Service with Azure Data Lake

```
CREATE ASSEMBLY IF NOT EXISTS [ImageCommon]
FROM @"\usqlext\assembly\cognition\vision\common\ImageIO.dll"
WITH ADDITIONAL_FILES =
(
@"\ImageCommon.dll",
@"\FaceSdkManagedWrapper.dll",
@"\libiomp5md.dll",
@"\DetectionJDA.mdl"
);
```

Analytics

```
REFERENCE ASSEMBLY ImageCommon;
REFERENCE ASSEMBLY FaceSdk;
REFERENCE ASSEMBLY ImageEmotion;
REFERENCE ASSEMBLY ImageTagging;
REFERENCE ASSEMBLY ImageOcr;
```

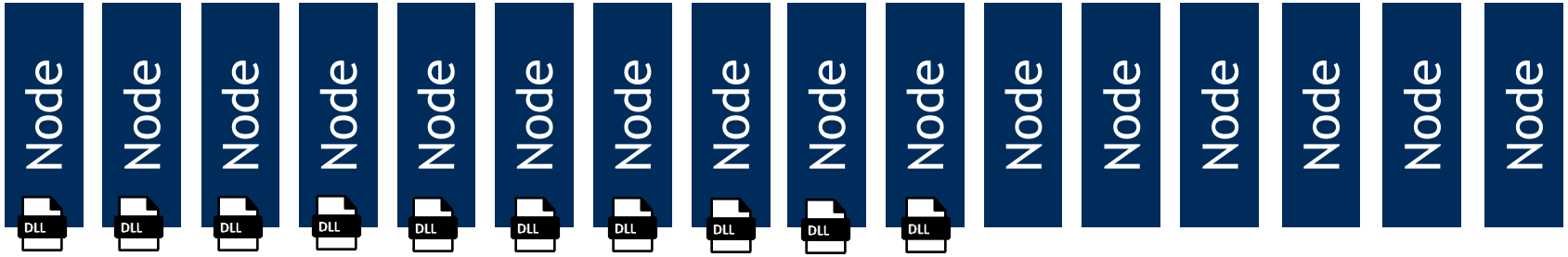
U-SQL

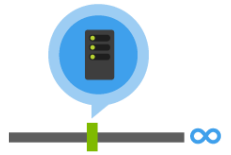
```
EXTRACT
    FileName string,
    ImgData byte[]
FROM @"/Images/{FileName}.jpg"
USING new Cognition.Vision.ImageExtractor();
```

```
PROCESS @imgs
PRODUCE FileName, NumObjects int,
    Tags SQL.MAP<string, float?>
READONLY FileName
USING new Cognition.Vision.ImageTagger();
```

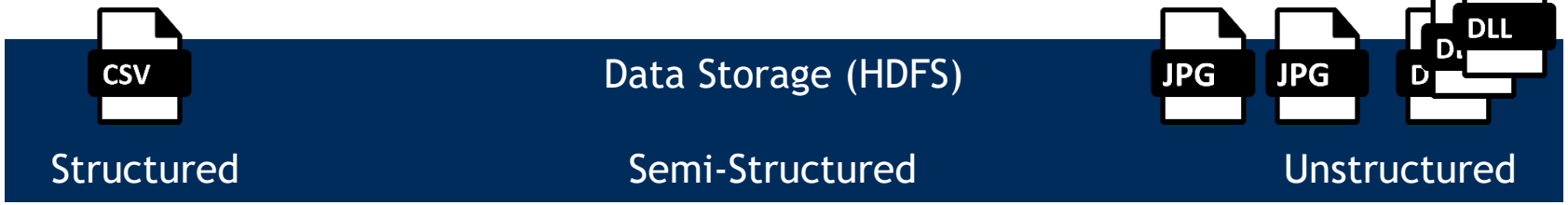
Processing Engine

```
OUTPUT @tags_from_processor
TO @"/Output/FileTags.csv"
USING
Outputters.Csv
(outputHeader : true);
```





Storage



https://docs.microsoft.com/en-us/azure/data-lake-analytics/data-lake-analytics-u-sql-cognitive

Scale Out Python with Azure Data Lake

Analytics

```
REFERENCE ASSEMBLY ExtPython;
```

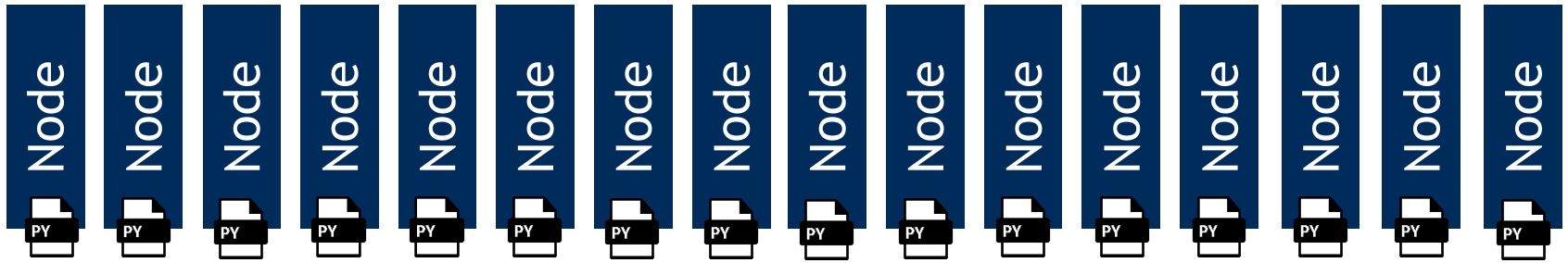
U-SQL

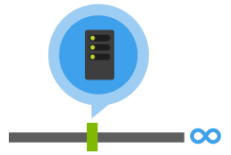
```
DECLARE @myScript = @"def get_mentions(tweet):
return ';'.join((w[1:] for w in tweet.split() if
w[0]=='@')) def usqlml_main(df): del df['time']
del df['author'] df['mentions'] = df.tweet.apply
(get_mentions) del df['tweet']    return df";
```

```
REDUCE @t ON date
PRODUCE date string, mentions string
USING new
Extension.Python.Reducer(pyScript:@myScr
ipt);
```

Processing Engine

```
OUTPUT @m
TO "/tweetmentions.csv"
USING Outputters.Csv();
```





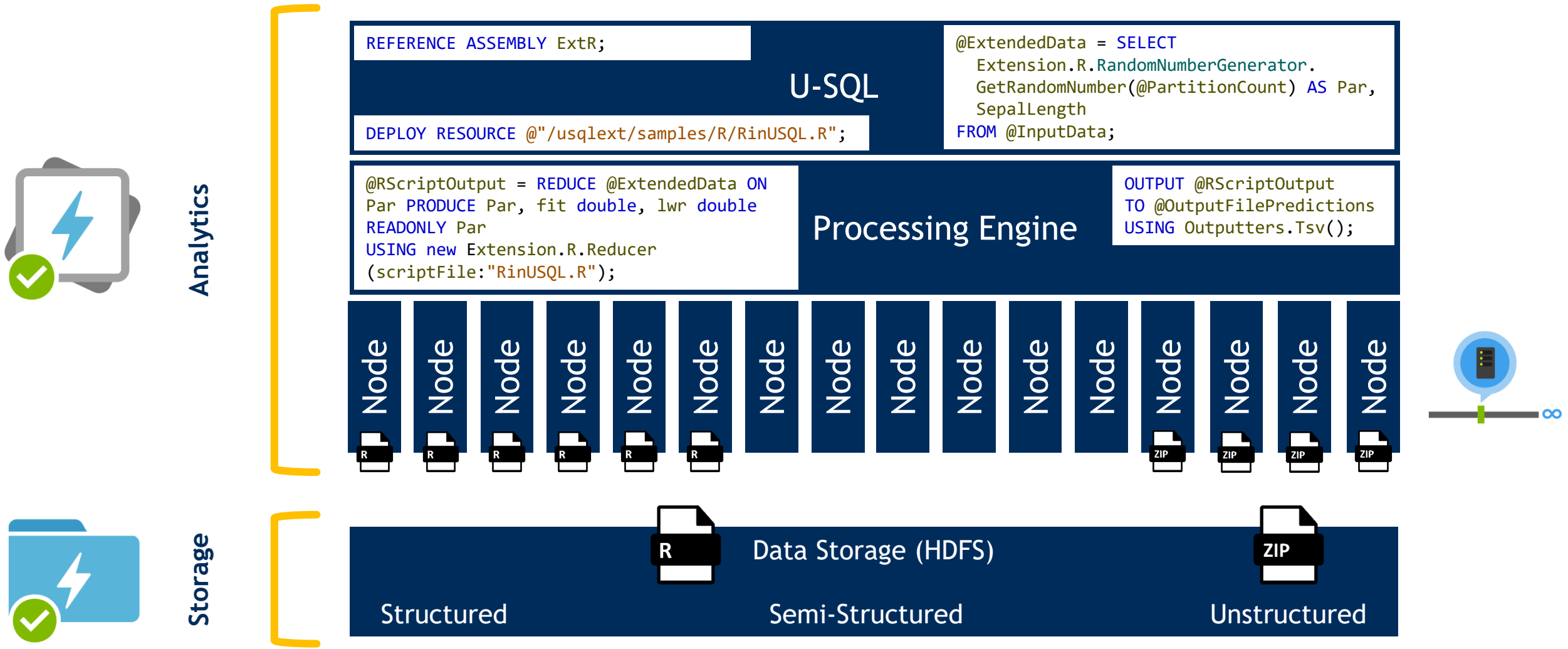
Storage

Data Storage (HDFS)

Structured | Semi-Structured | Unstructured

https://docs.microsoft.com/en-us/azure/data-lake-analytics/data-lake-analytics-u-sql-python-extensions

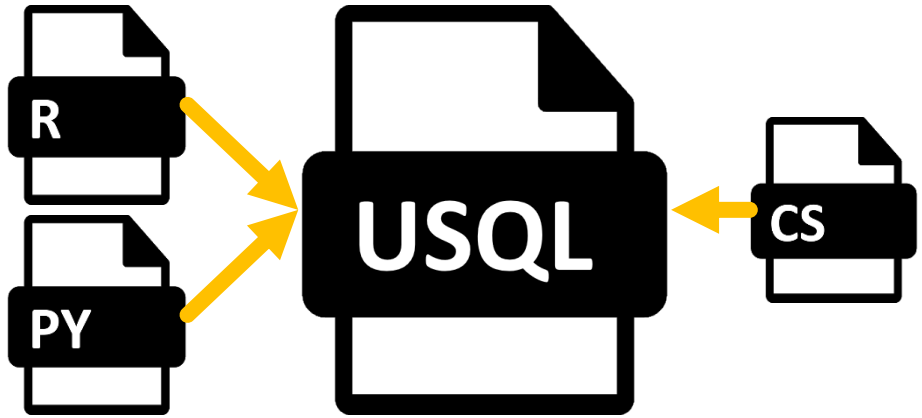
Scale Out R with Azure Data Lake



https://docs.microsoft.com/en-us/azure/data-lake-analytics/data-lake-analytics-u-sql-r-extensions

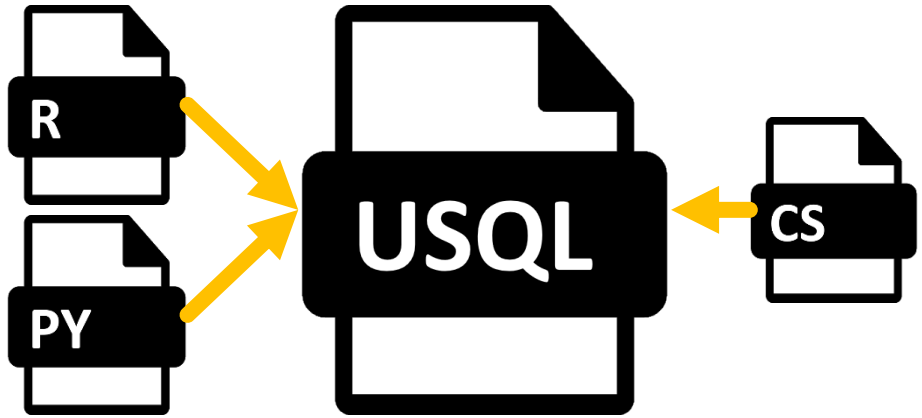
What is U-SQL? **Again**

```
@SizeAndCount =
    SELECT
        [ModifiedDate].ToString("yyyy") AS Year,
        [FileName].Substring([FileName].IndexOf(".") + 1, 3) AS FileExtension,
        COUNT(0) AS RecordCount,
        Math.Ceiling(Convert.ToDecimal(SUM([Size]))) AS FileSizeTotalsMB,
        Math.Ceiling(Convert.ToDecimal(SUM([Size])/1024)) AS FileSizeTotalsGB
    FROM
        @Raw
    WHERE
        [ActualFileName] == "FileDetailsTest.csv"
    GROUP BY
        [ModifiedDate].ToString("yyyy"),
        [FileName].Substring([FileName].IndexOf(".") + 1, 3);
```



What is U-SQL?

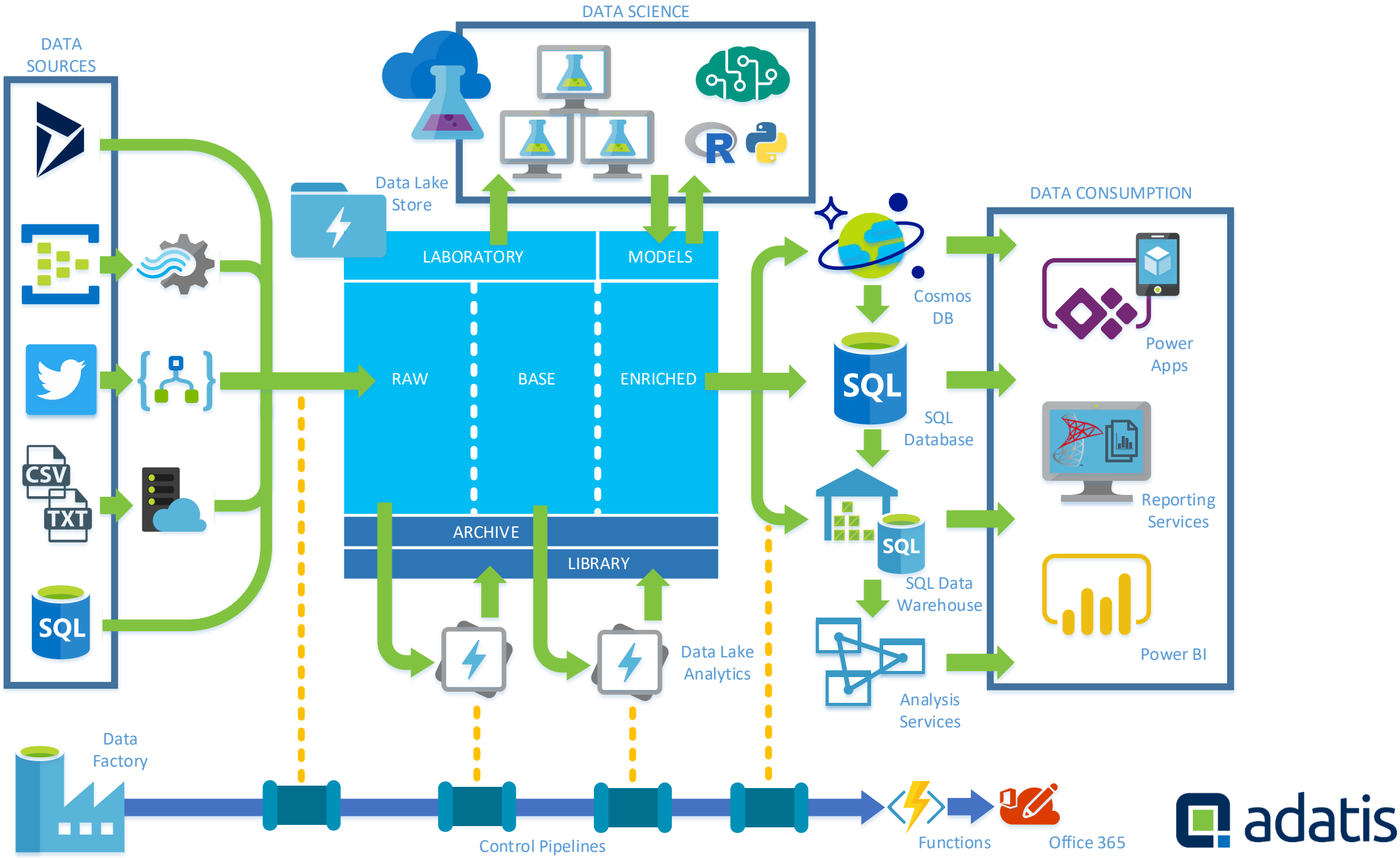
Answer: A scalable hybrid query framework.



Agenda

Azure Data Lake What & Why	What is U-SQL	U-SQL as a Framework	Meta Data Driven U-SQL
Storage & Compute	A Quick Recap	Scale Out .Net, R & Python	Production Code Generation

The Modern Data Analytics Platform

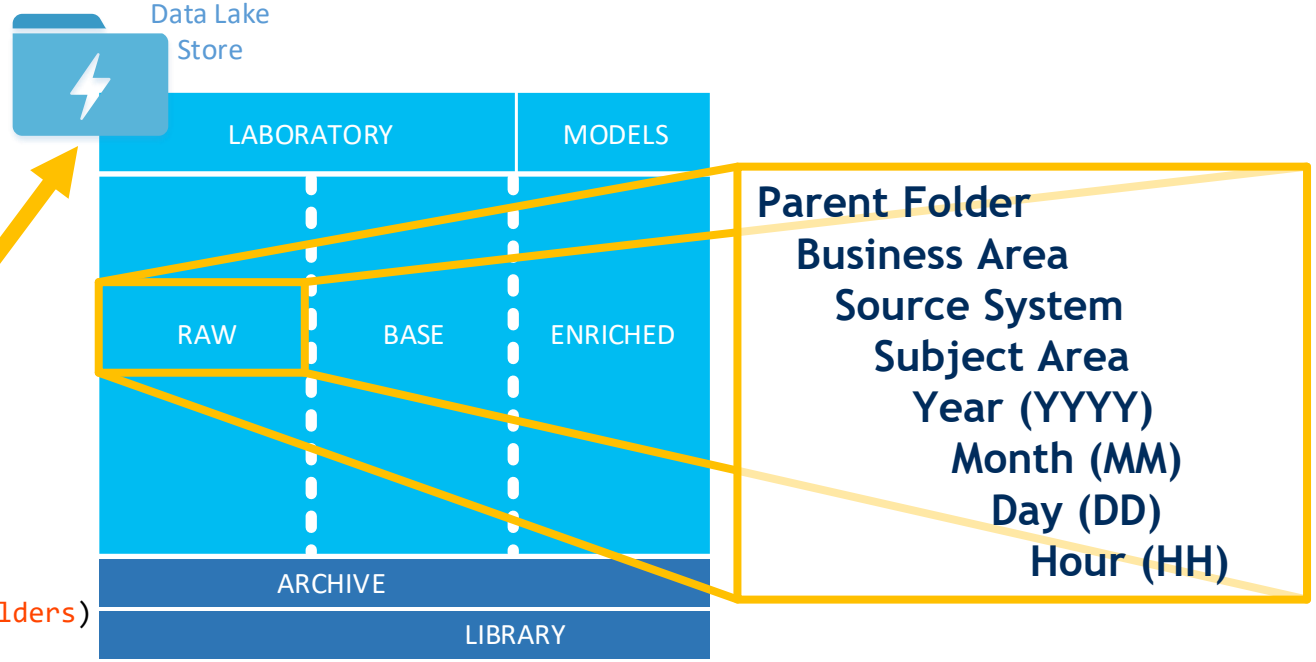


adatis

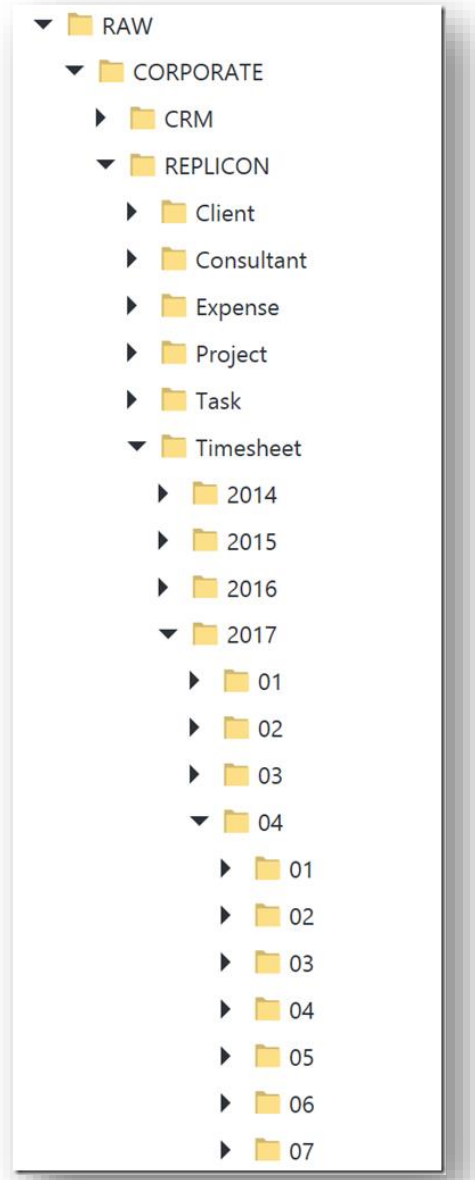
Setting Up Azure Data Lake Storage in Production

http://blogs.adatis.co.uk/ustoldfield/post/Shaping-The-Lake-Data-Lake-Framework

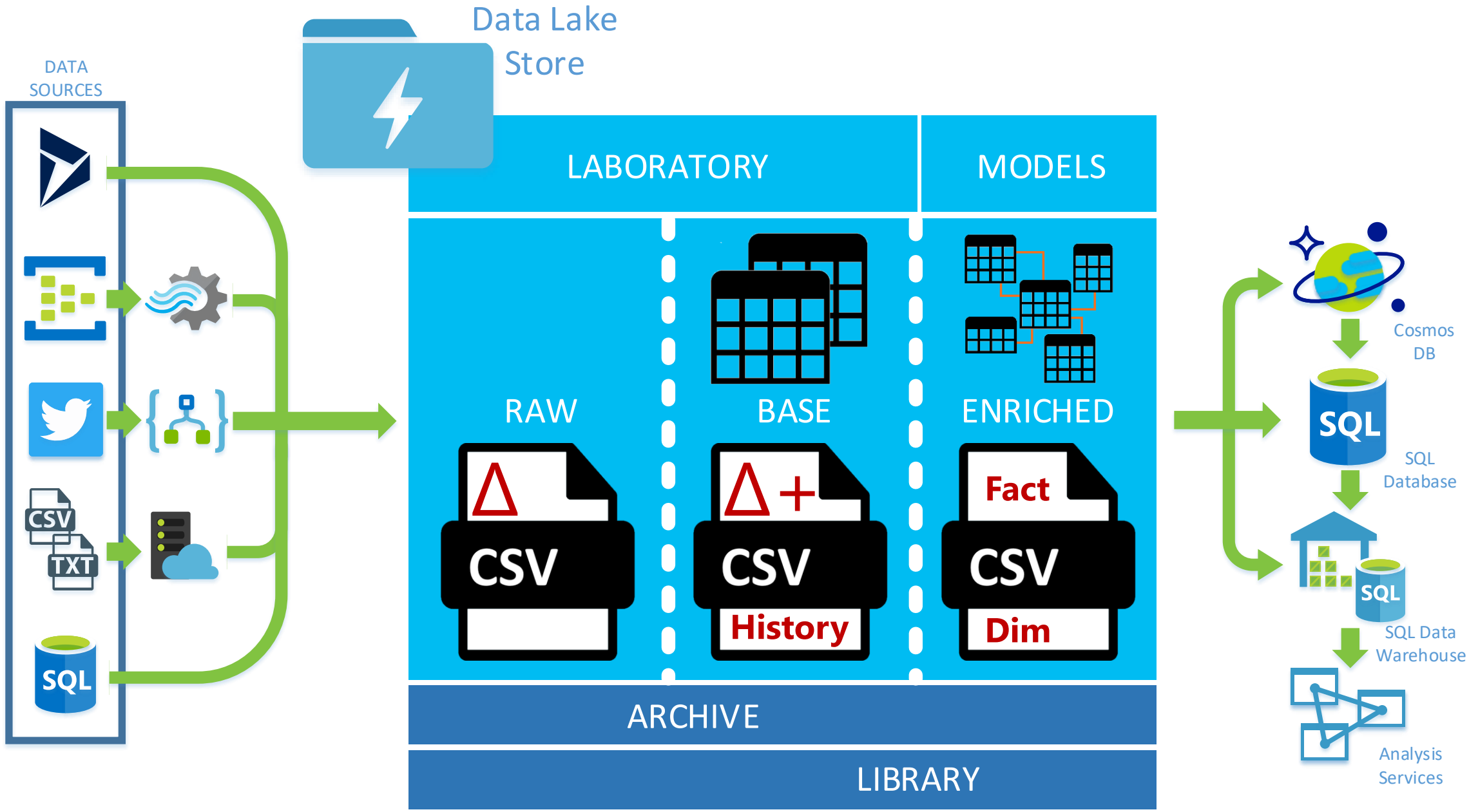

```
[
{
"path": "//RAW",
"security": []
},
{
"path": "//BASE",
"security": []
},
{
"path": "//ENRICHED",
"security": []
},
{
"path": "//MODEL",
"security": []
},
{
"path": "//LIBRARY",
"security": []
},
{
"path": "//LABORATORY",
"security": []
},
{
"path": "//UTILITIES",
"security": []
}
]
```




```
foreach ($folder in $folders)
{
    $path = $myrootdir + $folder.path

    $dl = Get-AzureRmDataLakeStoreItem -AccountName $dataLakeStoreName -Path $path

    if (!$dl)
    {
        New-AzureRmDataLakeStoreItem -Folder -AccountName $dataLakeStoreName -Path $path
        Write-Host "New Path Created" -ForegroundColor Cyan
    }
    else
    {
        Write-Host $path " already exists in account "  $dataLakeStoreName -ForegroundColor Cyan
    }  }
```



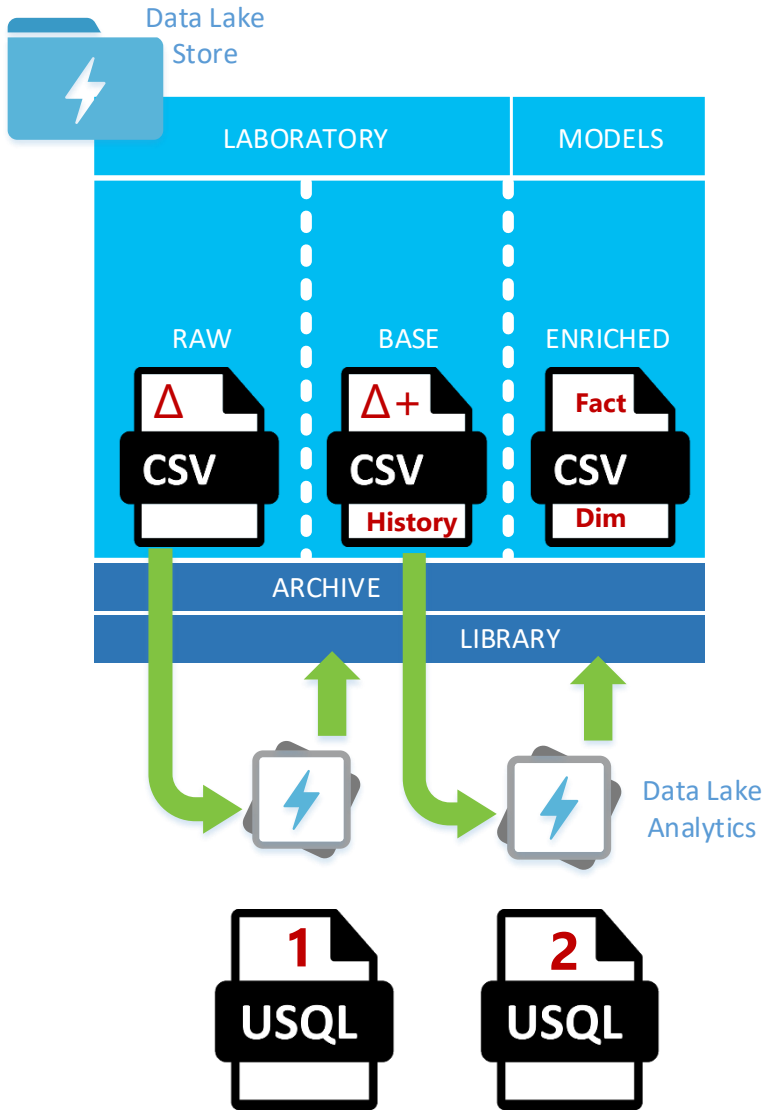
Using Azure Data Lake Storage in Production



Using Azure Data Lake Analytics in Production

1. RAW to BASE

```
/*
INSERT
CODE
HERE
*/
```



2. BASE to ENRICHED

```
/*
INSERT
CODE
HERE
*/
```

Using Azure Data Lake Analytics in Production

1. RAW to BASE

```
DECLARE @LocalRunDate string = "20180604";

DECLARE @InputFilePath string =
"/RAW/SourceSystem/{yyyy}/{mm}/{dd}/FileName.csv";

@Extracted =
    EXTRACT
        [Columns] string,
        //virtual columns
        [yyyy] string,
        [mm] string,
        [dd] string
    FROM
        @InputFilePath
    USING
        Extractors.Csv(skipFirstNRows:1);


@Delta =
    SELECT
        *
    FROM
        @Extracted
    WHERE
        [yyyy] + [mm] + [dd] == @LocalRunDate;
```

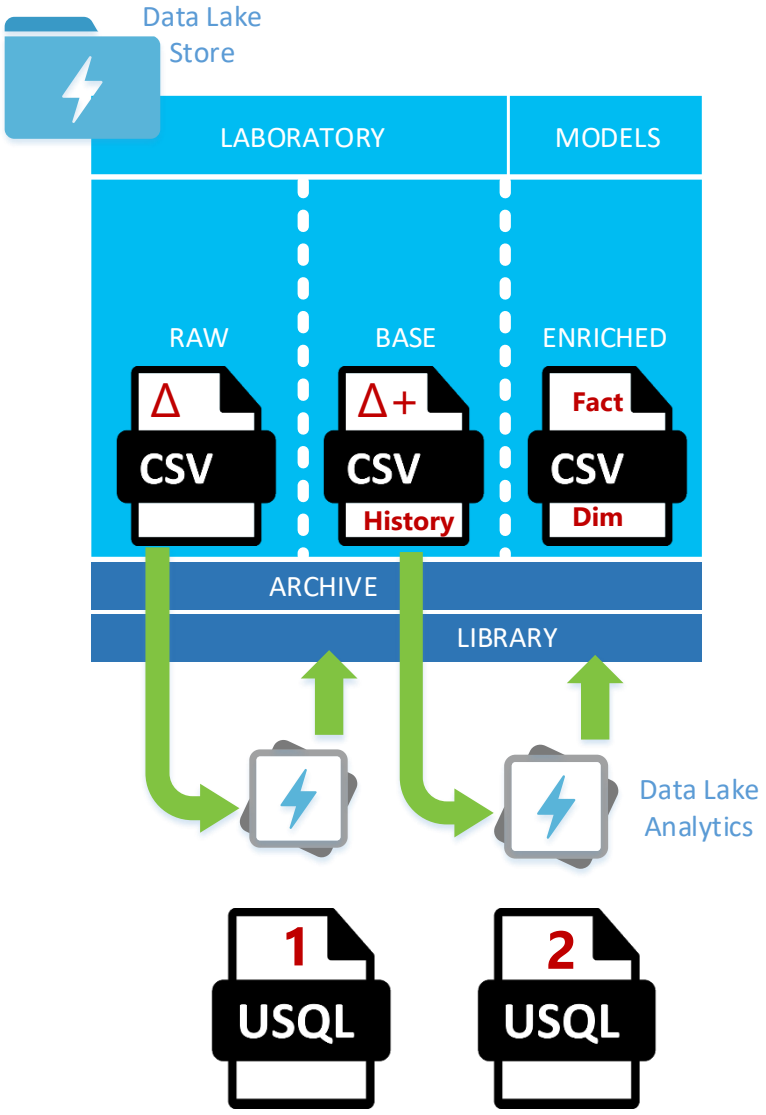
https://semanticinsight.wordpress.com/2018/04/19/finer-points-usql-merging-datasets-part-2/

```
@Merged =
    SELECT
        (string)([checkFlag] ? [src_Field1] : [tgt_Field1]) AS Field1,
        (string)([checkFlag] ? [src_Field2] : [tgt_Field2]) AS Field2
    FROM
        (
        SELECT
            (
                ([source].[PK] == [target].[PK] & [source].[PK] != null)
                || ([PK].[PK] == null) ? true : false
            ) AS checkFlag,
            //source data
            source.[Field1] AS src_Field1,
            source.[Field2] AS src_Field2,
            //target data
            target.[Field1] AS tgt_Field1,
            target.[Field2] AS tgt_Field2
        FROM
            @Delta AS source
            FULL OUTER JOIN [base].[Table1] AS target
                ON [source].[PK] == [target].[PK]
        ) AS dataMerge;

TRUNCATE TABLE [base].[Table1];
INSERT INTO [base].[Table1] ([Field1],[Field2])
SELECT [Field1],[Field2] FROM @Merged;
```

Using Azure Data Lake Analytics in Production

1. RAW to BASE

```
CREATE PROCEDURE IF NOT EXISTS
[base].[DeltaLoad_Table1]
        (
        @LocalRunDate string
        )
AS
BEGIN

DECLARE @InputFilePath string =
"/RAW/SourceSystem/{yyyy}/{mm}/{dd}/FileName.csv";

@Extracted =
    EXTRACT
        [Columns] string,
        //virtual columns
        [yyyy] string,
        [mm] string,
        [dd] string
    FROM
        @InputFilePath
    USING
        Extractors.Tsv(skipFirstNRows:1, quoting : true);

@Delta =
    SELECT
        *
    FROM
        @Extracted
    WHERE
        [yyyy] + [mm] + [dd] == @LocalRunDate;

@Merged =
    SELECT
        (string)([checkFlag] ? [src_Field1] : [tgt_Field1]) AS Field1,
(string)([checkFlag] ? [src_Field2] : [tgt_Field2]) AS Field2
    FROM
        (
        SELECT
            (
                ([source].[PK] == [target].[PK] & [source].[PK] != null)
                || ([PK].[PK] == null) ? true : false
            ) AS checkFlag,
            //source data
            source.[Field1] AS src_Field1,
source.[Field2] AS src_Field2,
            //target data
            target.[Field1] AS tgt_Field1,
target.[Field2] AS tgt_Field2
        FROM
            @Delta AS source
            FULL OUTER JOIN [base].[Table1] AS target
                ON [source].[PK] == [target].[PK]
        ) AS dataMerge;

TRUNCATE TABLE [base].[Table1];
INSERT INTO [base].[Table1] ([Field1],[Field2]) SELECT [Field1],[Field2] FROM
@Merged;

END;
```



2. BASE to ENRICHED

```
/*
INSERT
CODE
HERE
*/
```

Using Azure Data Lake Analytics in Production

2. BASE to ENRICHED

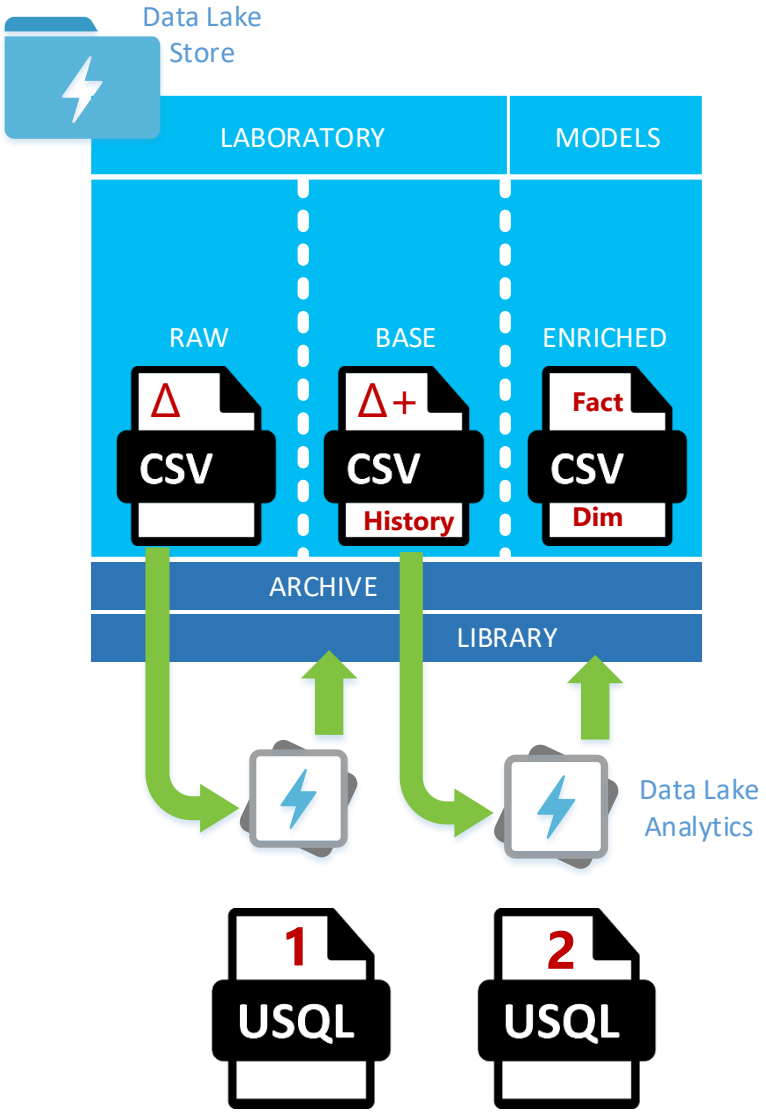
```
@AllSourceData =
    SELECT
        *
    FROM
        [SourceSystem1].[base].[Sales]

    UNION ALL

    SELECT
        *
    FROM
        [SourceSystem2].[base].[Sales];


@Transformation =
    /* INSERT CODE HERE */

@Lookups =
    /* INSERT CODE HERE */

@Aggregates =
    /* INSERT CODE HERE */
```

```
@OutputDataset =
    SELECT
        *,
        [SourceSystemKey],
        [LastUpdatedDate]
    FROM
        @Aggregates,@Lookups;


DECLARE @OutputPath = "/ENRICHED/Warehouse/Fact/Sales.csv";

OUTPUT @OutputDataset
TO @outputLocation
USING Outputters.Csv();
```

Using Azure Data Lake Analytics in Production

1. RAW to BASE

```
CREATE PROCEDURE IF NOT EXISTS
[base].[DeltaLoad_Table1]
        (
        @LocalRunDate string
        )
AS
BEGIN

DECLARE @InputFilePath string =
"/RAW/SourceSystem/{yyyy}/{mm}/{dd}/FileName.csv";

@Extracted =
    EXTRACT
        [Columns] string,
        //virtual columns
        [yyyy] string,
        [mm] string,
        [dd] string
    FROM
        @InputFilePath
    USING
        Extractors.Tsv(skipFirstNRows:1, quoting : true);

@Delta =
    SELECT
        *
    FROM
        @Extracted
    WHERE
        [yyyy] + [mm] + [dd] == @LocalRunDate;

@Merged =
    SELECT
        (string)([checkFlag] ? [src_Field1] : [tgt_Field1]) AS Field1,
(string)([checkFlag] ? [src_Field2] : [tgt_Field2]) AS Field2
    FROM
        (
        SELECT
            (
                ([source].[PK] == [target].[PK] & [source].[PK] != null)
                || ([PK].[PK] == null) ? true : false
            ) AS checkFlag,
            //source data
            source.[Field1] AS src_Field1,
source.[Field2] AS src_Field2,
            //target data
            target.[Field1] AS tgt_Field1,
target.[Field2] AS tgt_Field2
        FROM
            @Delta AS source
            FULL OUTER JOIN [base].[Table1] AS target
                ON [source].[PK] == [target].[PK]
        ) AS dataMerge;

TRUNCATE TABLE [base].[Table1];
INSERT INTO [base].[Table1] ([Field1],[Field2]) SELECT [Field1],[Field2] FROM
@Merged;

END;
```



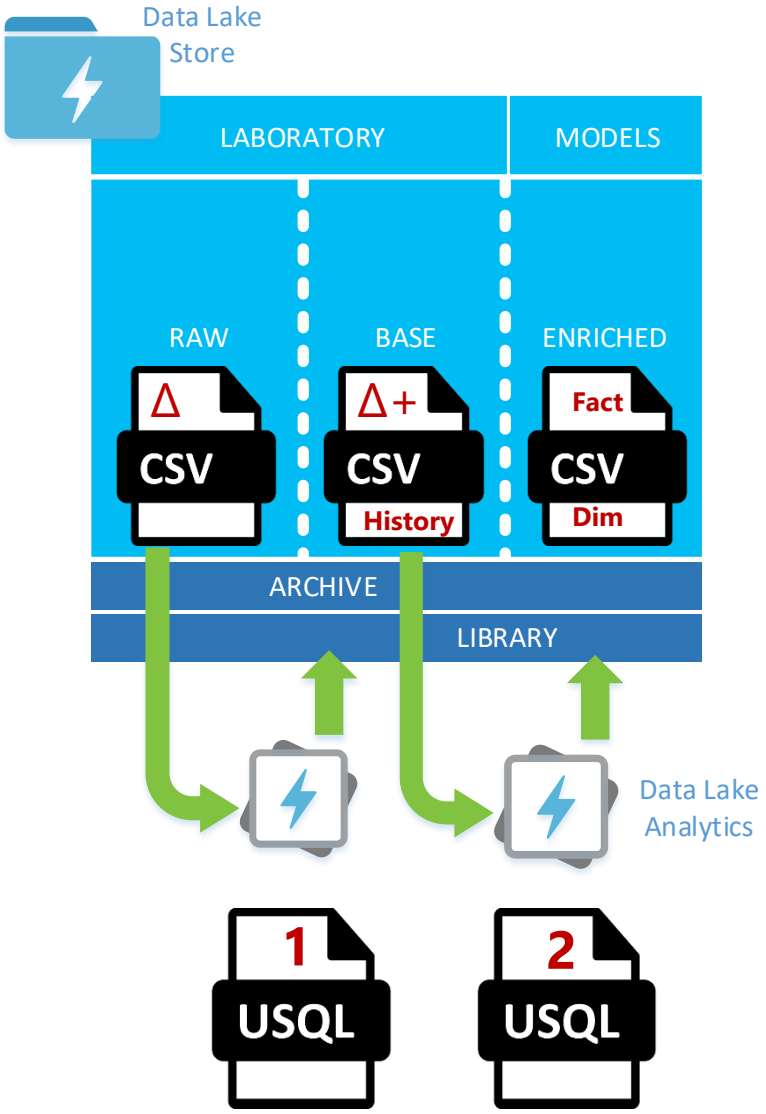
2. BASE to ENRICHED

```
CREATE PROCEDURE IF NOT EXISTS [fact].[Sales]
AS
BEGIN


    @AllSourceData =
        SELECT
            *
        FROM
            [SourceSystem1].[base].[Sales]

        UNION ALL

        SELECT
            *
        FROM
            [SourceSystem2].[base].[Sales];

    @Transformation =
        /* INSERT CODE HERE */

    @Lookups =
        /* INSERT CODE HERE */


    @OutputDataset =
        SELECT
            *,
            [SourceSystemKey],
            [LastUpdatedDate]
        FROM
            @Aggregates,@Lookups;

    DECLARE @OutputPath = "/ENRICHED/Warehouse/Fact/Sales.csv";

    OUTPUT @OutputDataset
    TO @outputLocation
    USING Outputters.Csv();


END;
```

Using Azure Data Lake Analytics in Production

1. RAW to BASE

```
CREATE PROCEDURE IF NOT EXISTS
[base].[DeltaLoad_Table1]
        (
        @LocalRunDate string
        )
AS
BEGIN

DECLARE @InputFilePath string =
"/RAW/SourceSystem/{yyyy}/{mm}/{dd}/FileName.csv";

@Extracted =
    EXTRACT
        [Columns] string,
        //virtual columns
        [yyyy] string,
        [mm] string,
        [dd] string
    FROM
        @InputFilePath
    USING
        Extractors.Tsv(skipFirstNRows:1, quoting : true);

@Delta =
    SELECT
        *
    FROM
        @Extracted
    WHERE
        [yyyy] + [mm] + [dd] == @LocalRunDate;

@Merged =
    SELECT
        (string)([checkFlag] ? [src_Field1] : [tgt_Field1]) AS Field1,
(string)([checkFlag] ? [src_Field2] : [tgt_Field2]) AS Field2
    FROM
        (
        SELECT
            (
                ([source].[PK] == [target].[PK] & [source].[PK] != null)
                || ([PK].[PK] == null) ? true : false
            ) AS checkFlag,
            //source data
            source.[Field1] AS src_Field1,
source.[Field2] AS src_Field2,
            //target data
            target.[Field1] AS tgt_Field1,
target.[Field2] AS tgt_Field2
        FROM
            @Delta AS source
            FULL OUTER JOIN [base].[Table1] AS target
                ON [source].[PK] == [target].[PK]
        ) AS dataMerge;

TRUNCATE TABLE [base].[Table1];
INSERT INTO [base].[Table1] ([Field1],[Field2]) SELECT [Field1],[Field2] FROM
@Merged;

END;
```

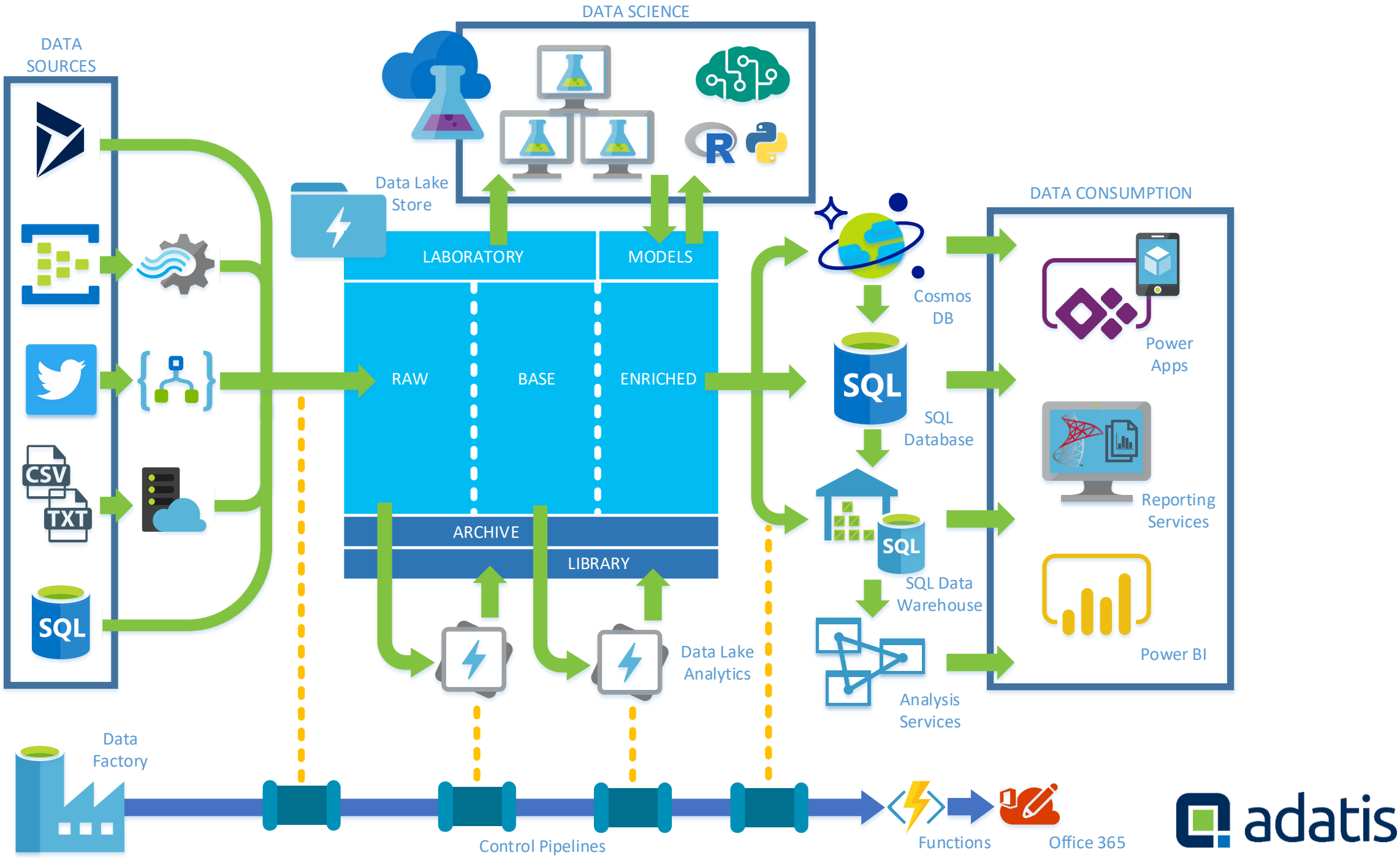


2. BASE to ENRICHED

```
CREATE PROCEDURE IF NOT EXISTS [fact].[Sales]
AS
BEGIN


    @AllSourceData =
        SELECT
            *
        FROM
            [SourceSystem1].[base].[Sales]

        UNION ALL

        SELECT
            *
        FROM
            [SourceSystem2].[base].[Sales];

    @Transformation =
        /* INSERT CODE HERE */

    @Lookups =
        /* INSERT CODE HERE */


    @OutputDataset =
        SELECT
            *,
            [SourceSystemKey],
            [LastUpdatedDate]
        FROM
            @Aggregates,@Lookups;

    DECLARE @OutputPath = "/ENRICHED/Warehouse/Fact/Sales.csv";

    OUTPUT @OutputDataset
    TO @outputLocation
    USING Outputters.Csv();


END;
```


adatis

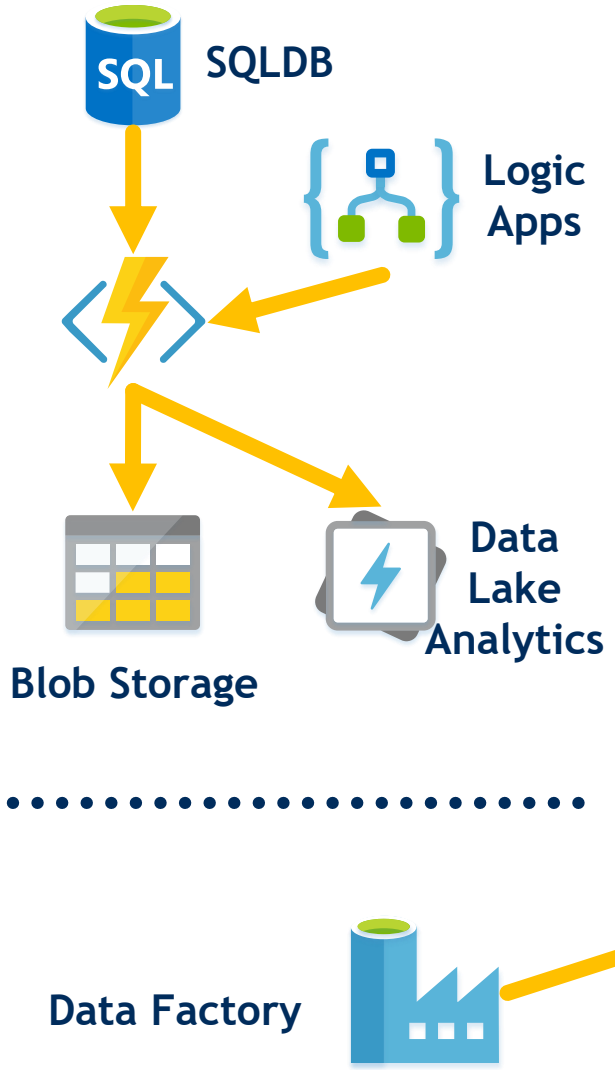
The Modern Data Analytics Platform

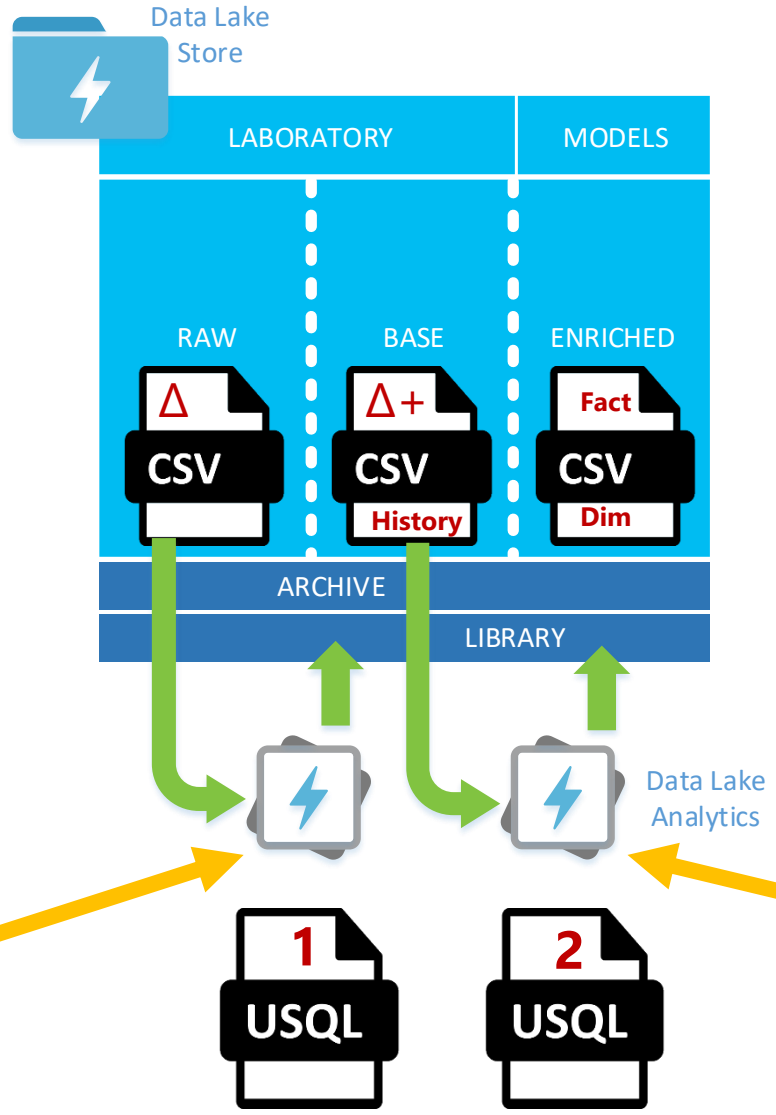


adatis

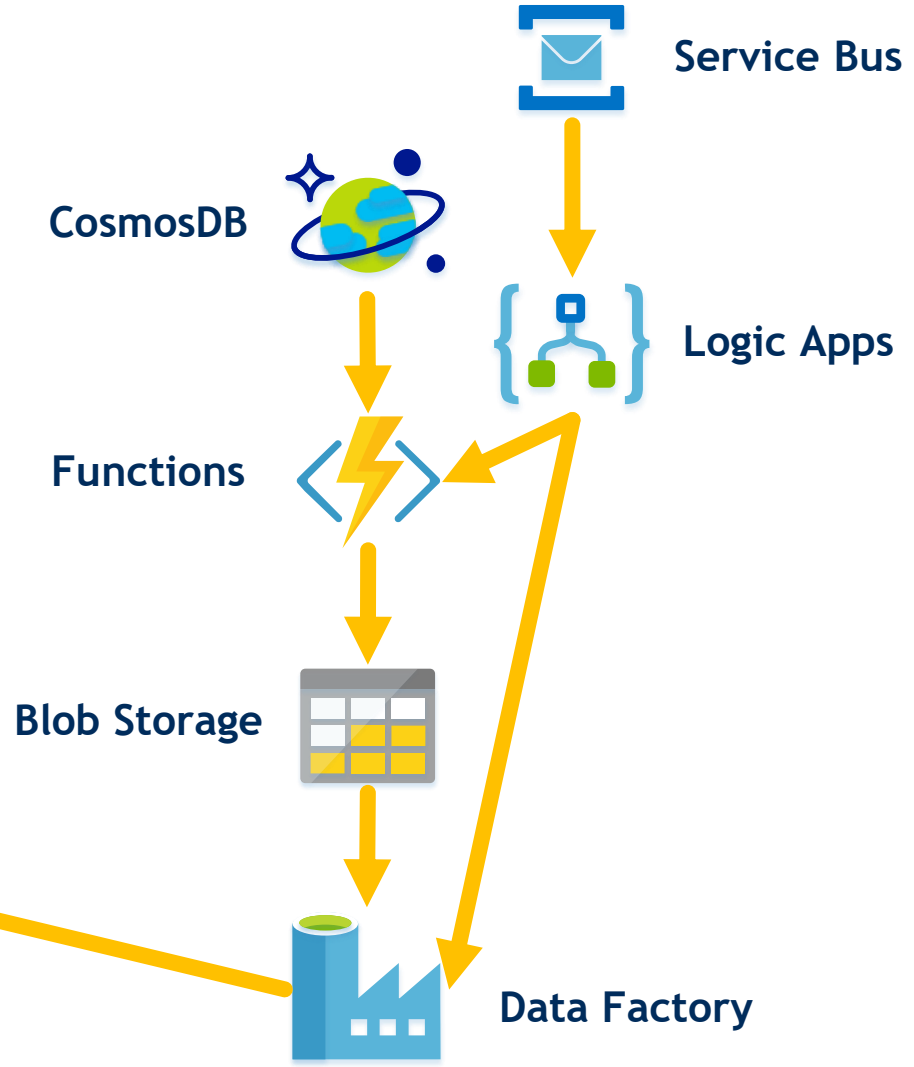
How do we generate our U-SQL?

Development Time





Run Time (Event Driven Loading)




adatis

U-SQL Further Reading

Microsoft U-SQL Language Reference Guide
https://msdn.microsoft.com/en-us/azure/data-lake-analytics/u-sql/u-sql-language-reference

SQL Server Central Stairway (21 chapters)
http://www.sqlservercentral.com/stairway/142480/

Stack Overflow U-SQL Tag
http://stackoverflow.com/questions/tagged/u-sql

MrPaulAndrew.com
https://mrpaulandrew.com/tag/u-sql/

Adatis Blogs
http://blogs.adatis.co.uk/search?q=U-SQL

Thanks for Listening

Paul Andrew

@MrPaulAndrew

Microsoft® Most Valuable Professional

adatis

Blog: http://mrpaulandrew.com
Email: paul@mrpaulandrew.com