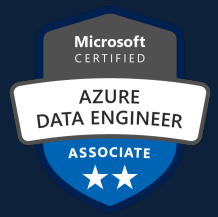
Azure Synapse Analytics For Data Engineers

An Introduction

About Me

Ramesh Retnasamy

Data Engineer/ Machine Learning Engineer



LinkedIn

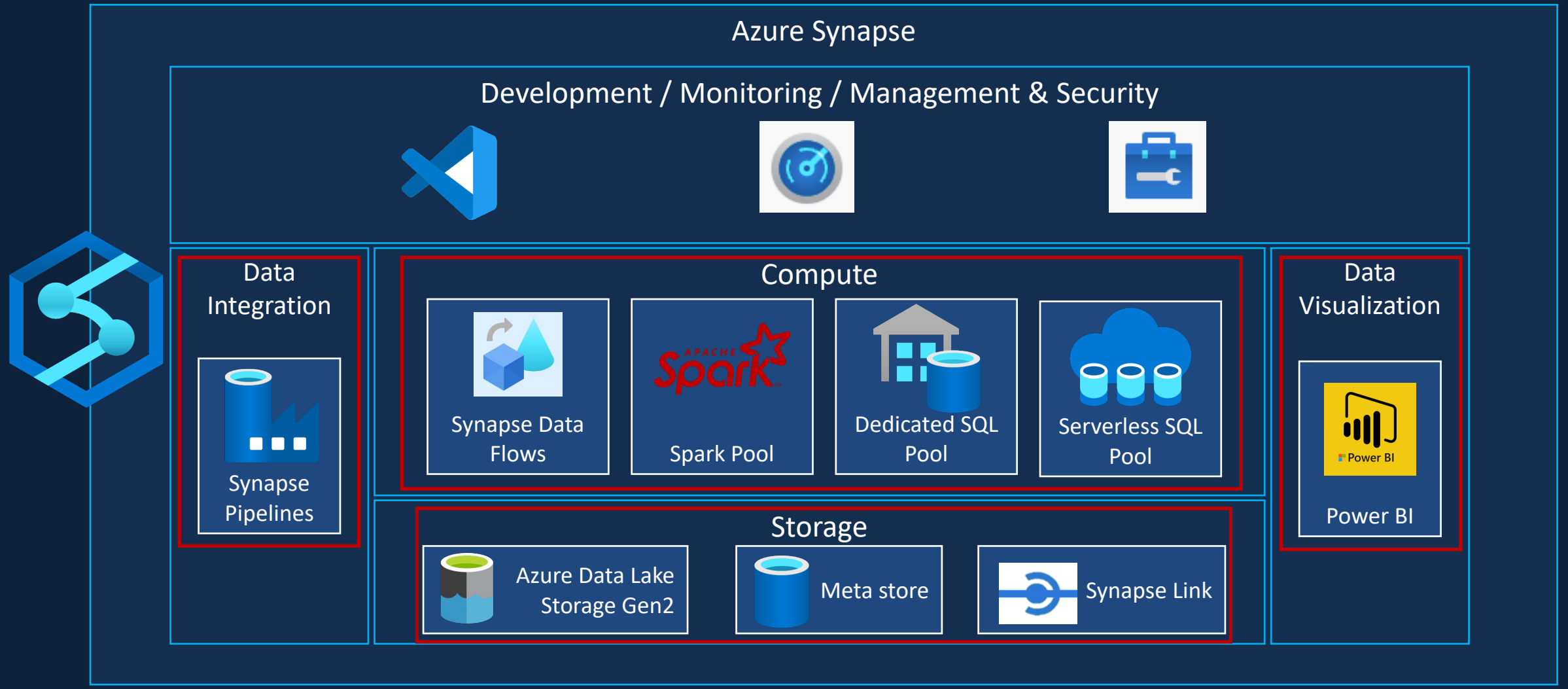
https://www.linkedin.com/in/ramesh-retnasamy/

About this course

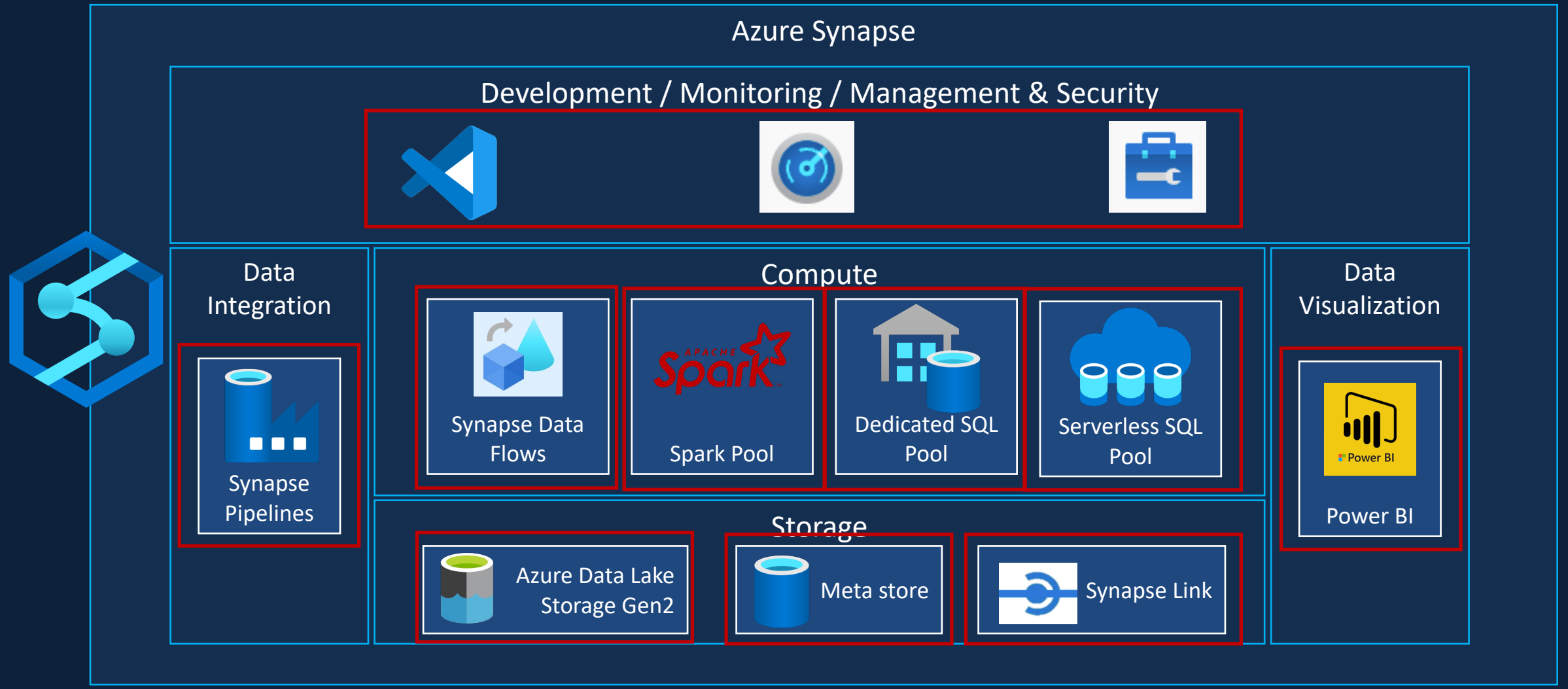
Azure Synapse Analytics

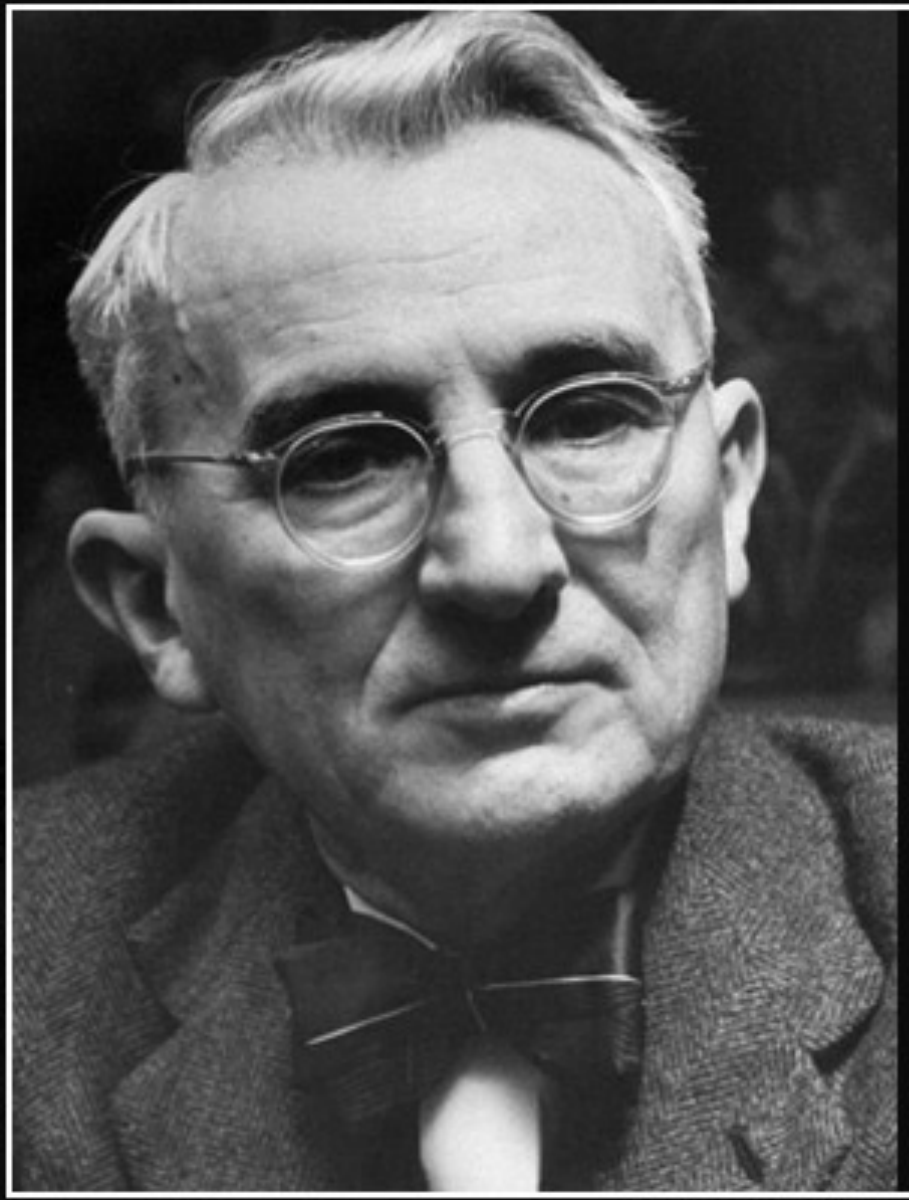
Limitless analytics service that brings together data integration, enterprise data warehousing and big data analytics.

Azure Synapse Analytics



Azure Synapse Analytics





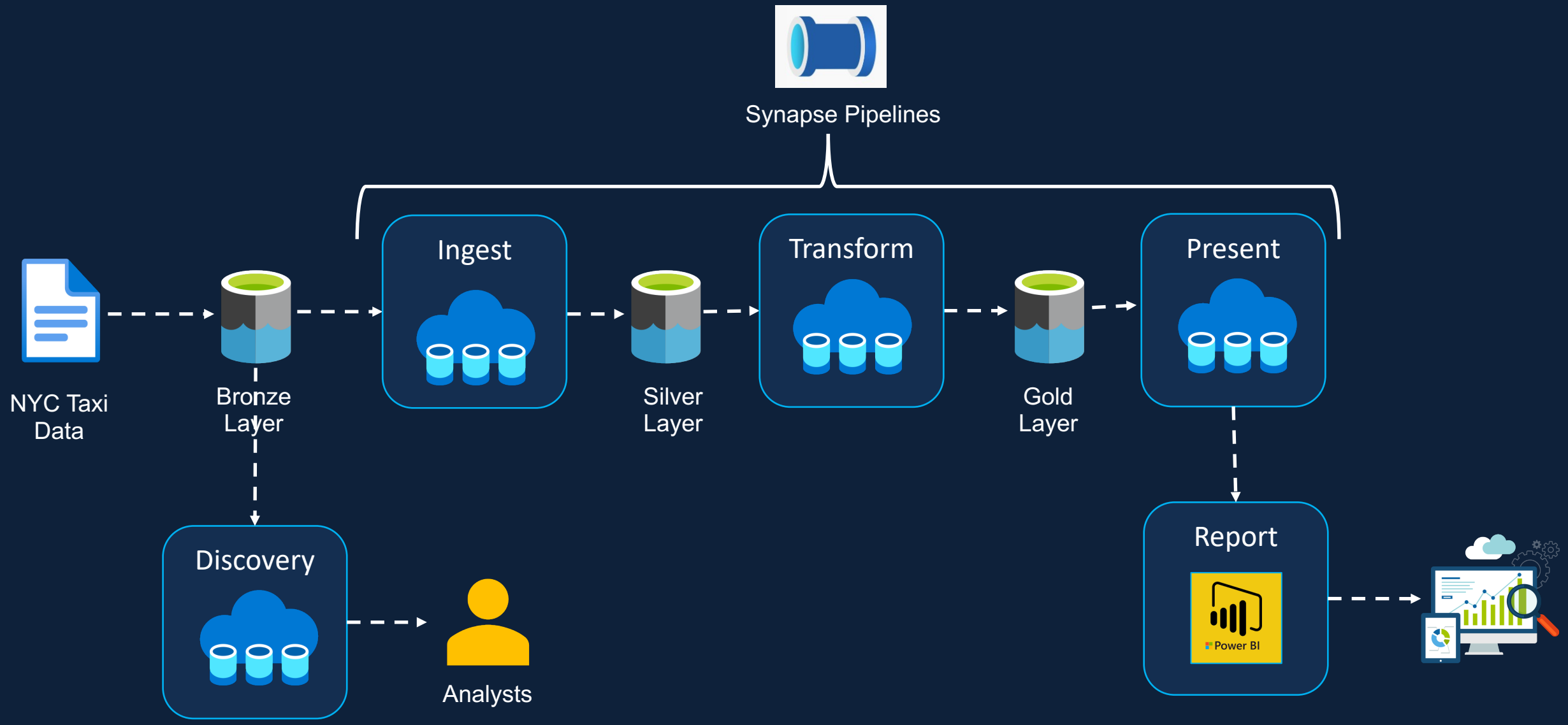
Learning is an active process. We learn by doing.. Only knowledge that is used sticks in your mind.

— Dale Carnegie —

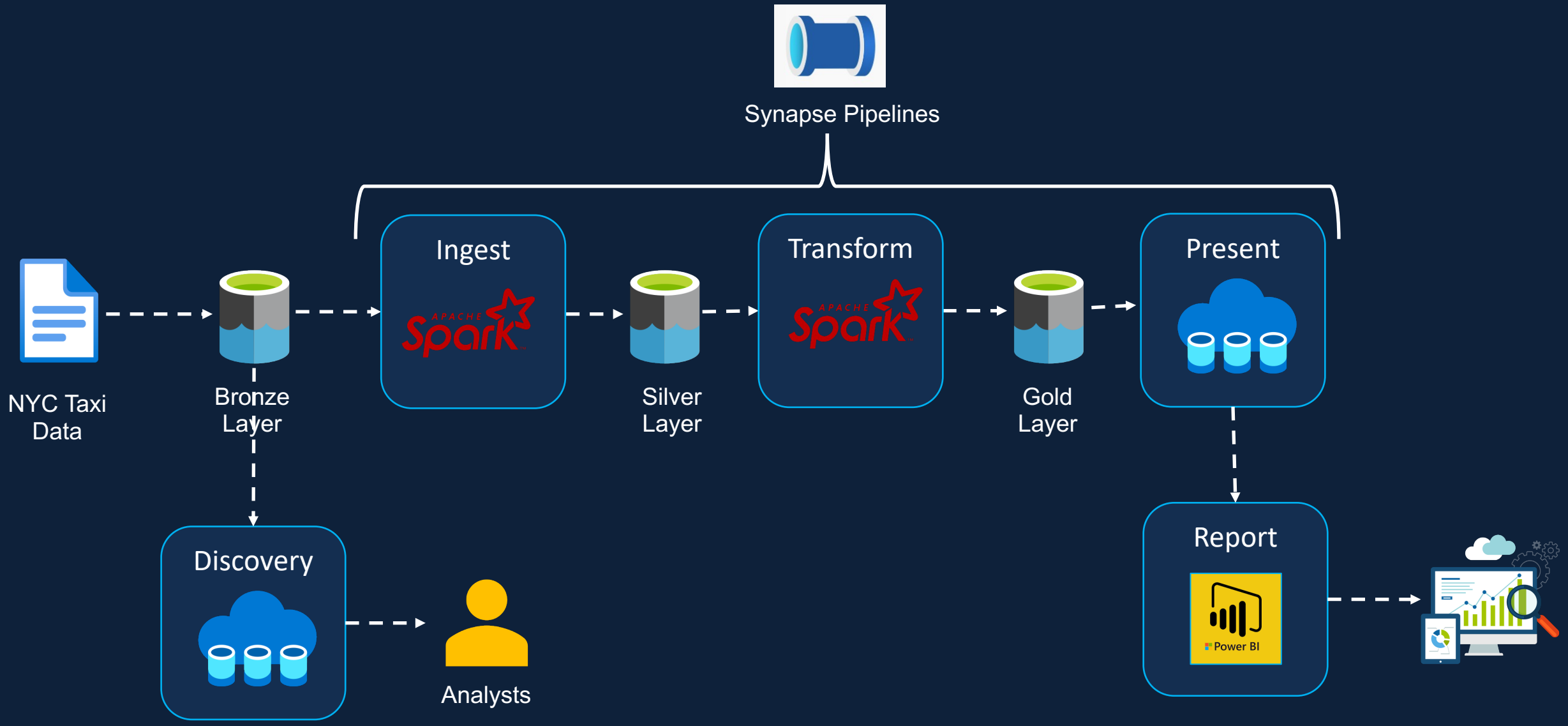
AZ QUOTES

NYC Taxi
Cloud Data Platform

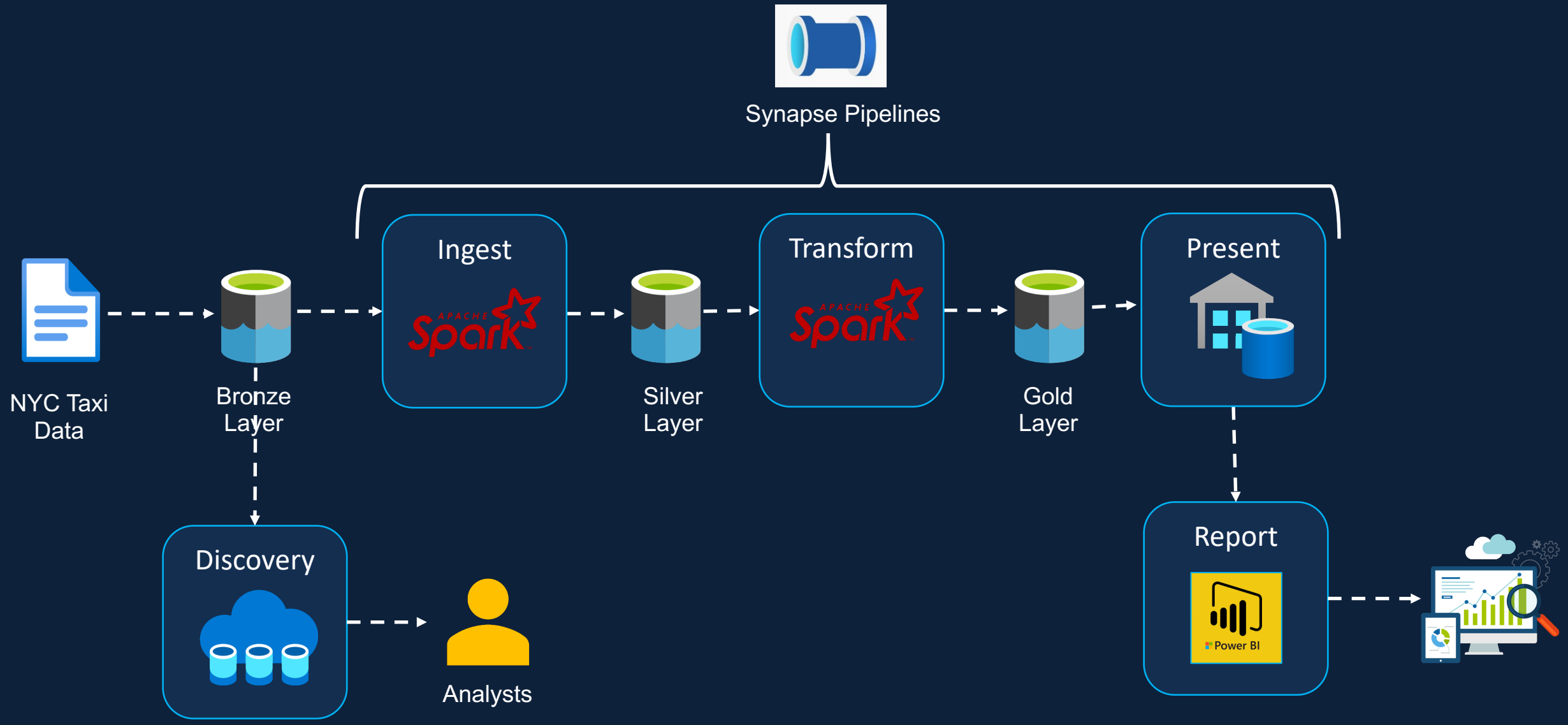
Solution Architecture – Serverless SQL Pool



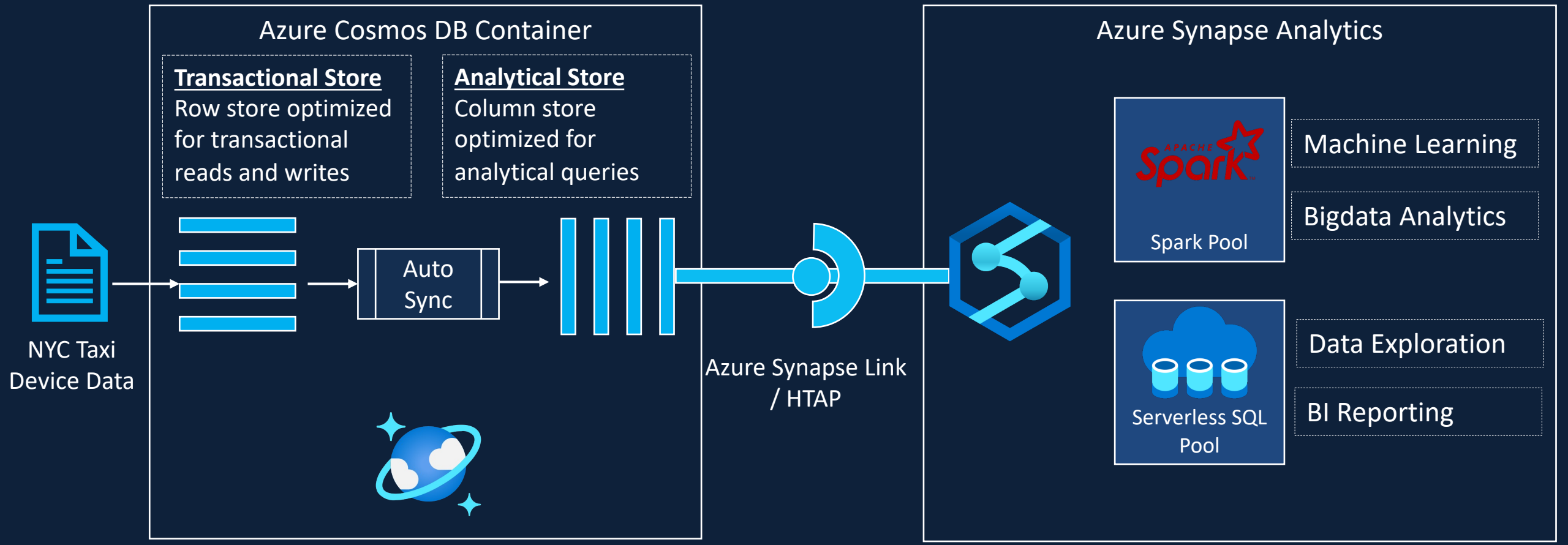
Solution Architecture – Spark Pool



Solution Architecture – Dedicated SQL Pool



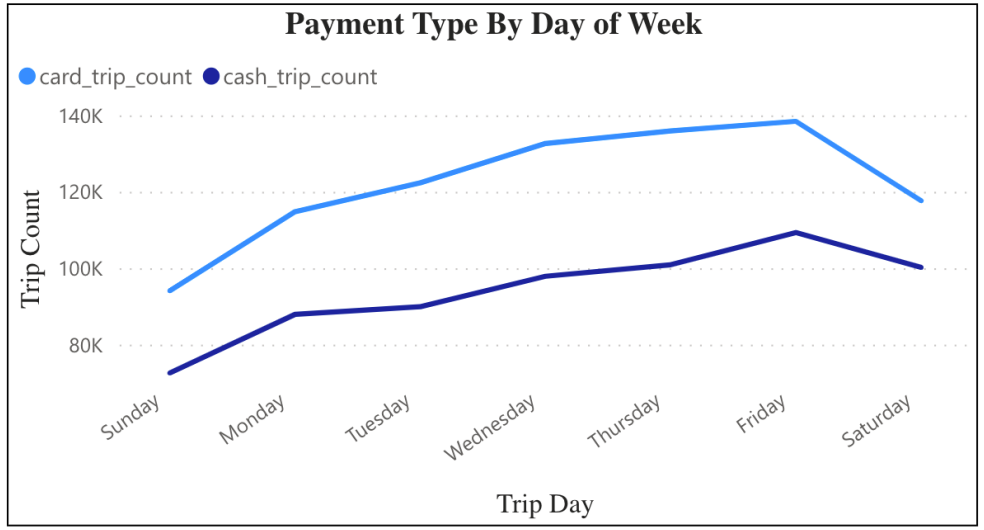
Solution Architecture - Synapse Link

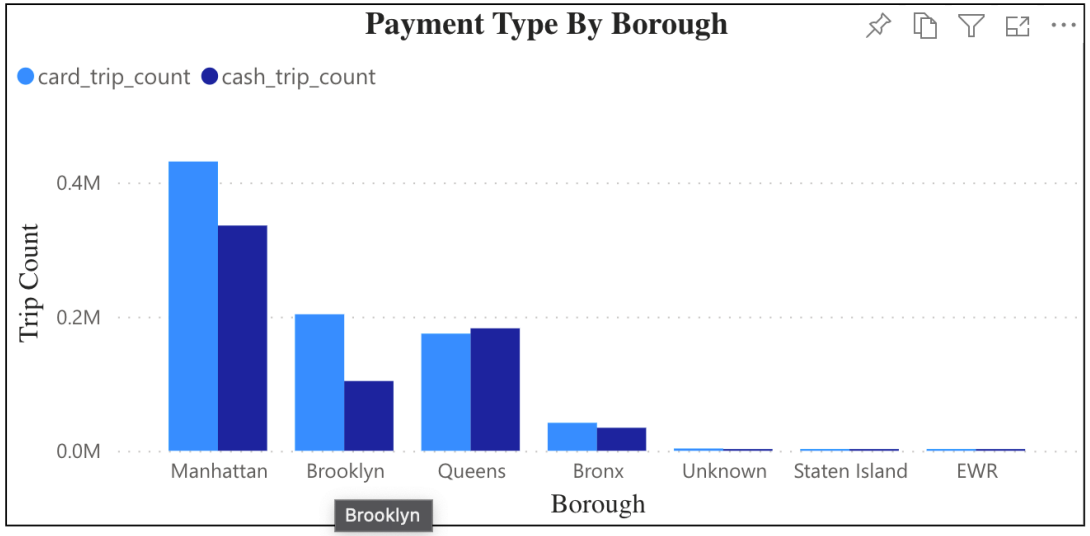


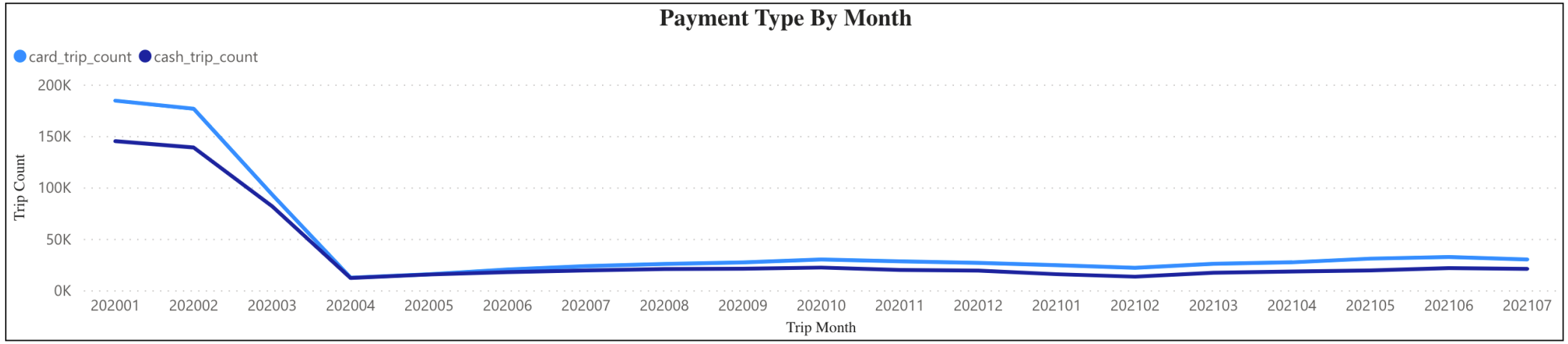
NYC Taxi Cloud Data Platform

NYC Taxi & Limousine Commission

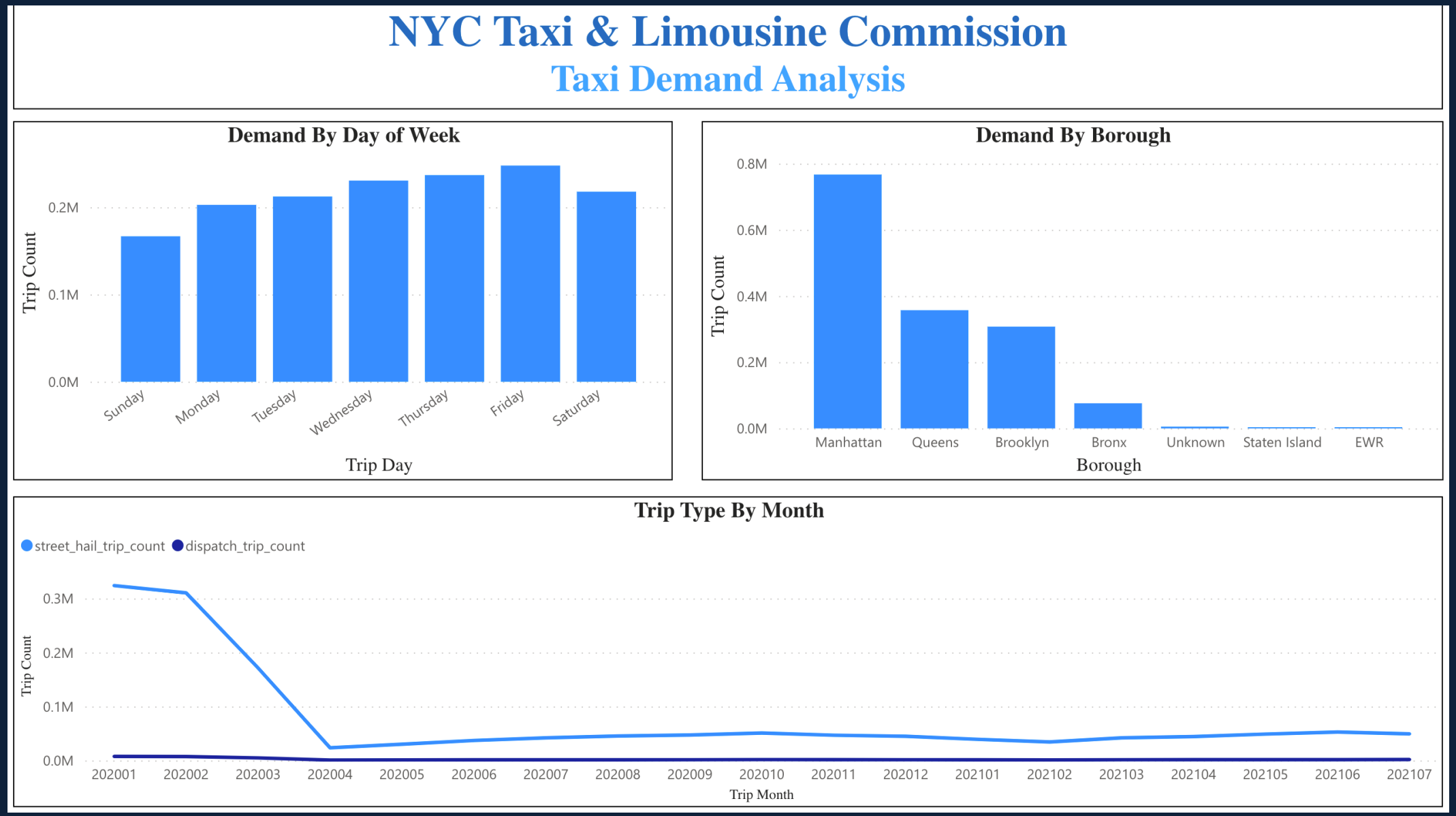
Card Payment Campaign Analysis







NYC Taxi Cloud Data Platform



Who is this course for

- University students
- IT Developers from other disciplines
- AWS/ GCP/ On-prem Data Engineers
- Data Architects

Who is this course not for

- You are not interested in hands-on learning approach
- Your only focus is Azure Data Engineering Certification
- You want to learn Data Flows
- You are looking for in-depth knowledge of dedicated SQL pool
- You are looking to learn dimensional data modelling & implementation

Pre-requisites

- All code and step-by-step instructions provided
- Basic SQL knowledge required
- Basic Python programming knowledge required
- Cloud fundamentals will be beneficial, but not mandatory
- Azure Account

Our Commitments

- Ask Questions, we will answer ☺
- Keeping the course up to date
- Udemy life time access
- Udemy 30 day money back guarantee

Introduction to Azure Synapse Analytics

Azure Synapse Analytics - Introduction

Azure Synapse Analytics is a limitless analytics service that brings together data integration, enterprise data warehousing and big data analytics.

Azure Synapse Analytics - Introduction

Azure Synapse Analytics is a limitless analytics service that brings together data integration, enterprise data warehousing and big data analytics.

Azure Synapse Analytics - Introduction

Azure Synapse Analytics is a limitless analytics service that brings together data integration, enterprise data warehousing and big data analytics.

Azure Synapse Analytics - Introduction

Data Warehouse

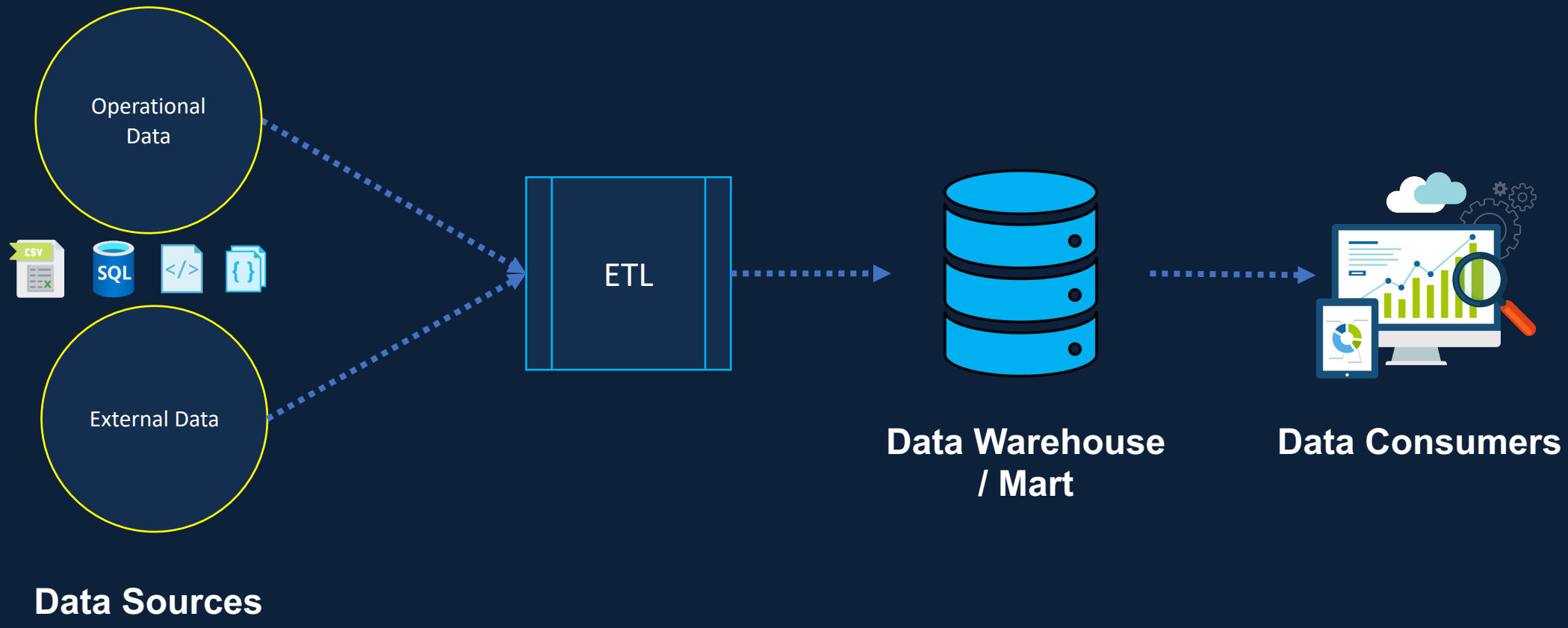
Emergence of Data Lakes

Modern Data Warehouse Architecture

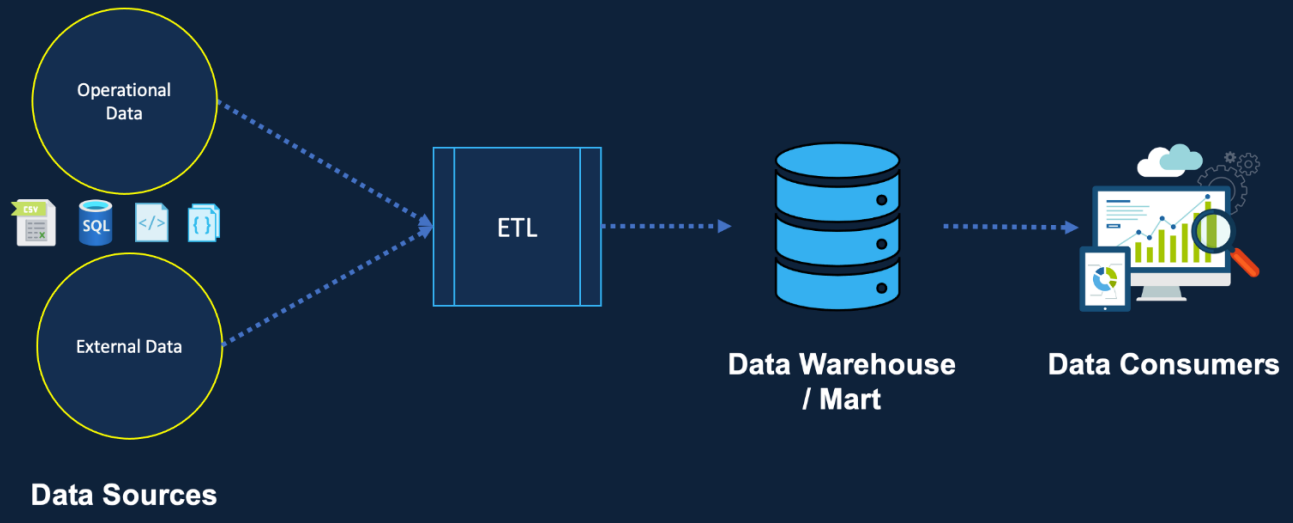
Modern Data Warehouse – Without Azure Synapse Analytics

Modern Data Warehouse - With Azure Synapse Analytics

Data Warehouse



Data Warehouse



Lack of support for unstructured data

Longer to ingest new data

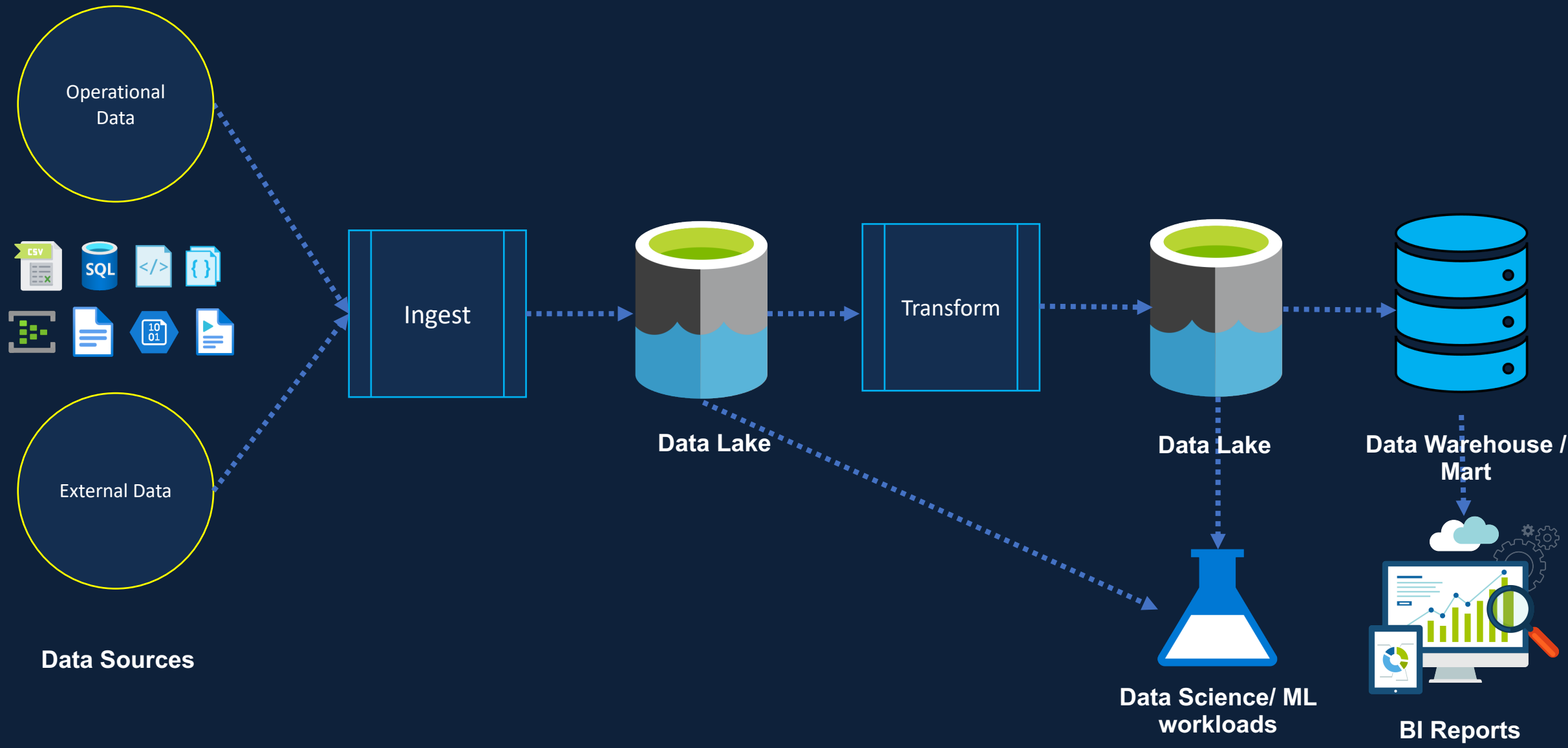
Proprietary data formats

Scalability

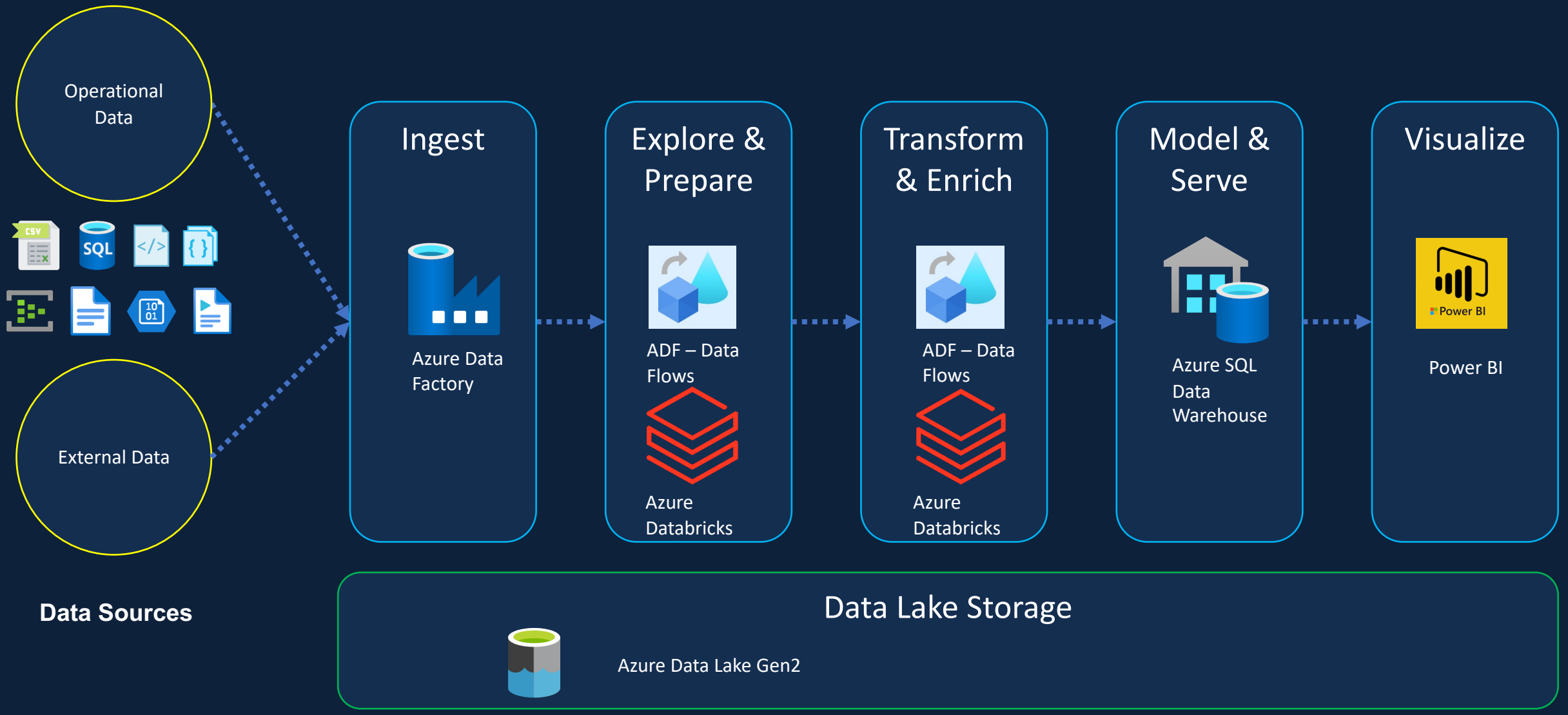
Expensive to store data

Lack of support for ML/ AI workloads

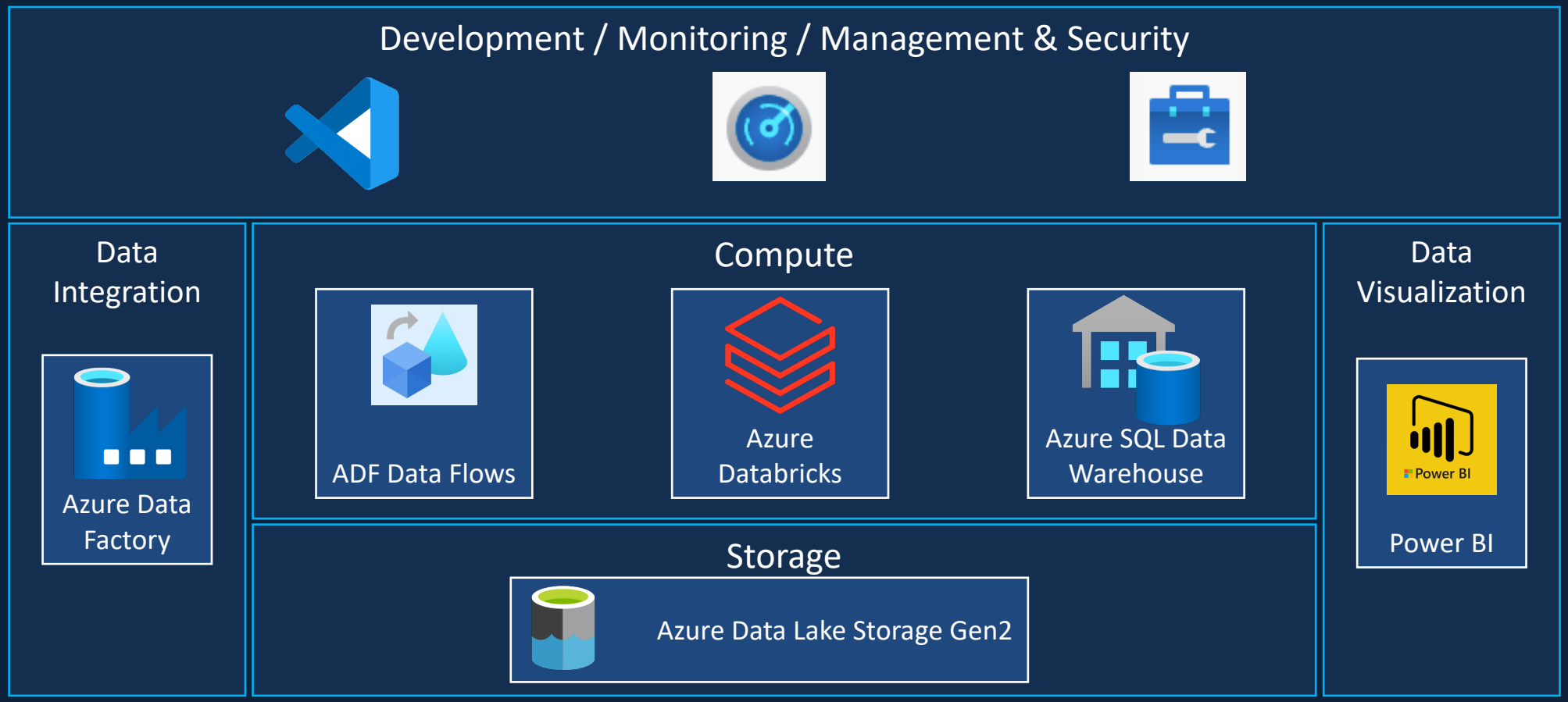
Data Lake


Operational Data
CSV
SQL
External Data
Data Sources
Ingest
Data Lake
Transform
Data Lake
Data Warehouse / Mart
Data Science/ ML workloads
BI Reports

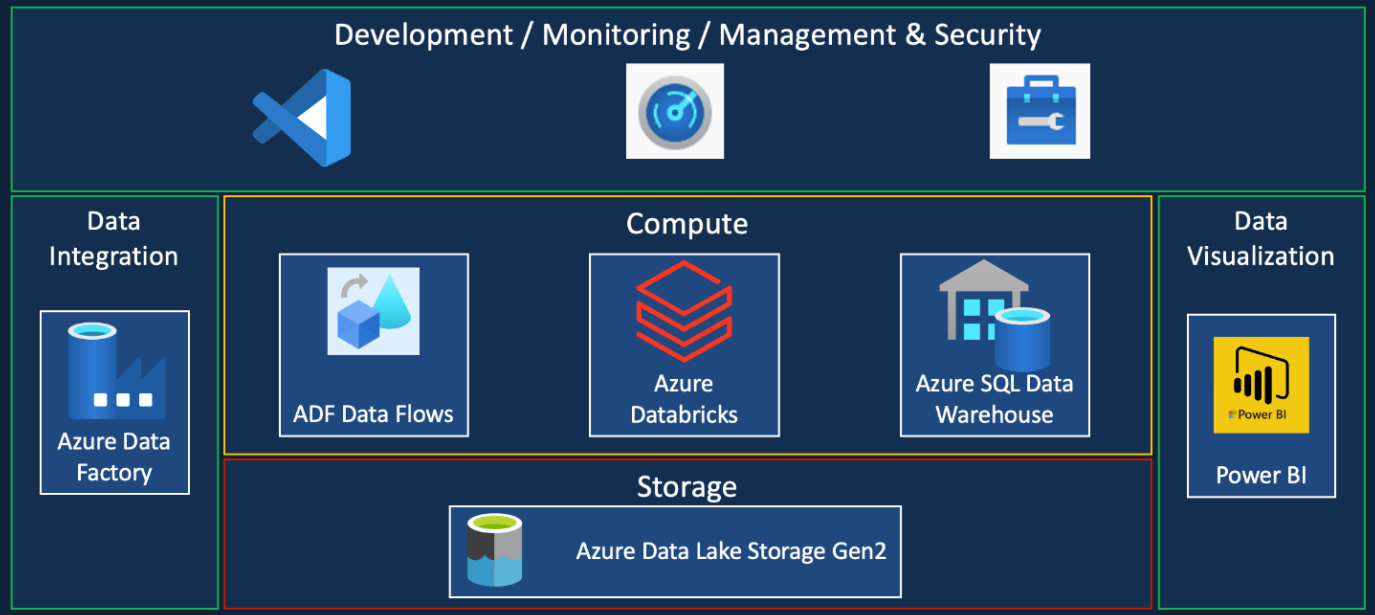
Modern Data Warehouse



Modern Data Warehouse



Modern Data Warehouse



Too many services/ workspaces

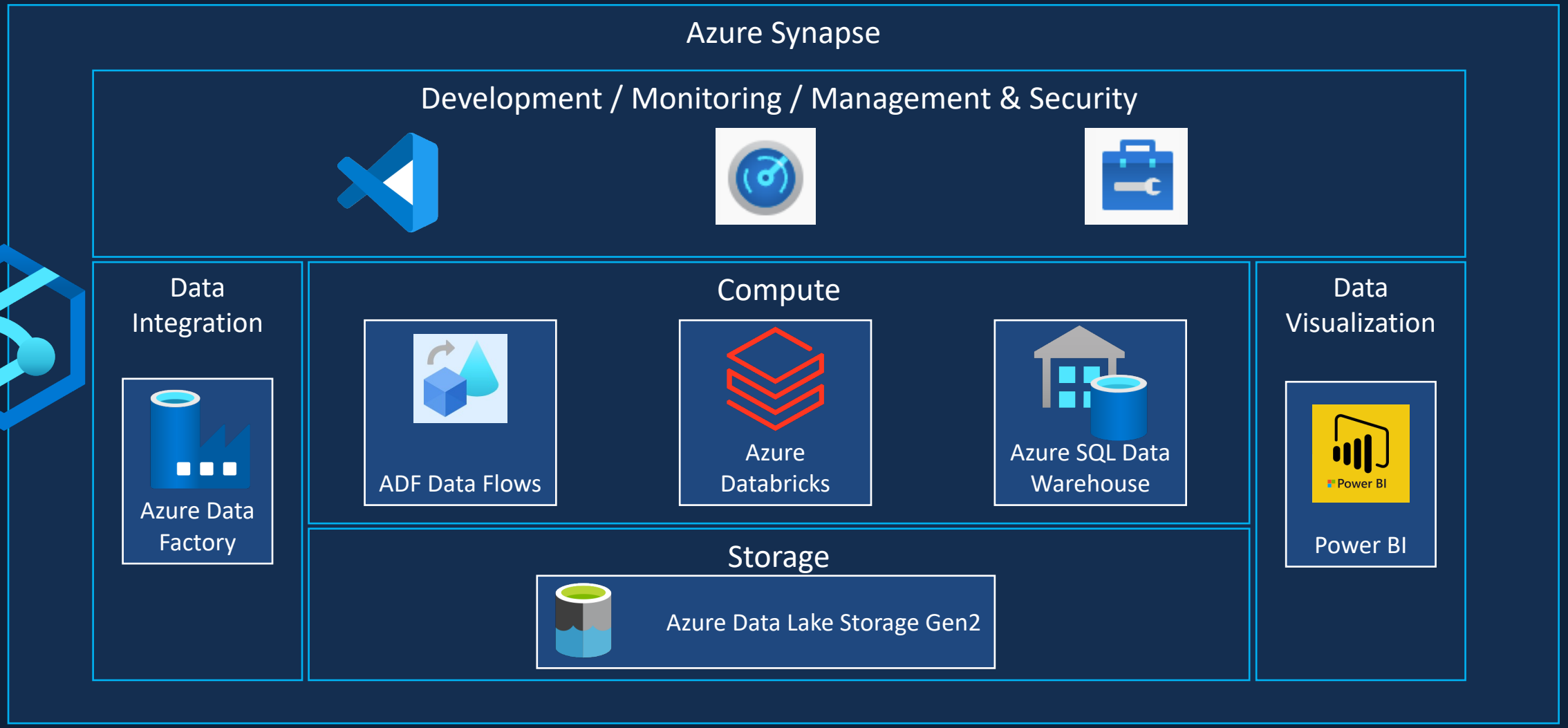
Difficult to monitor

Management & Security overhead

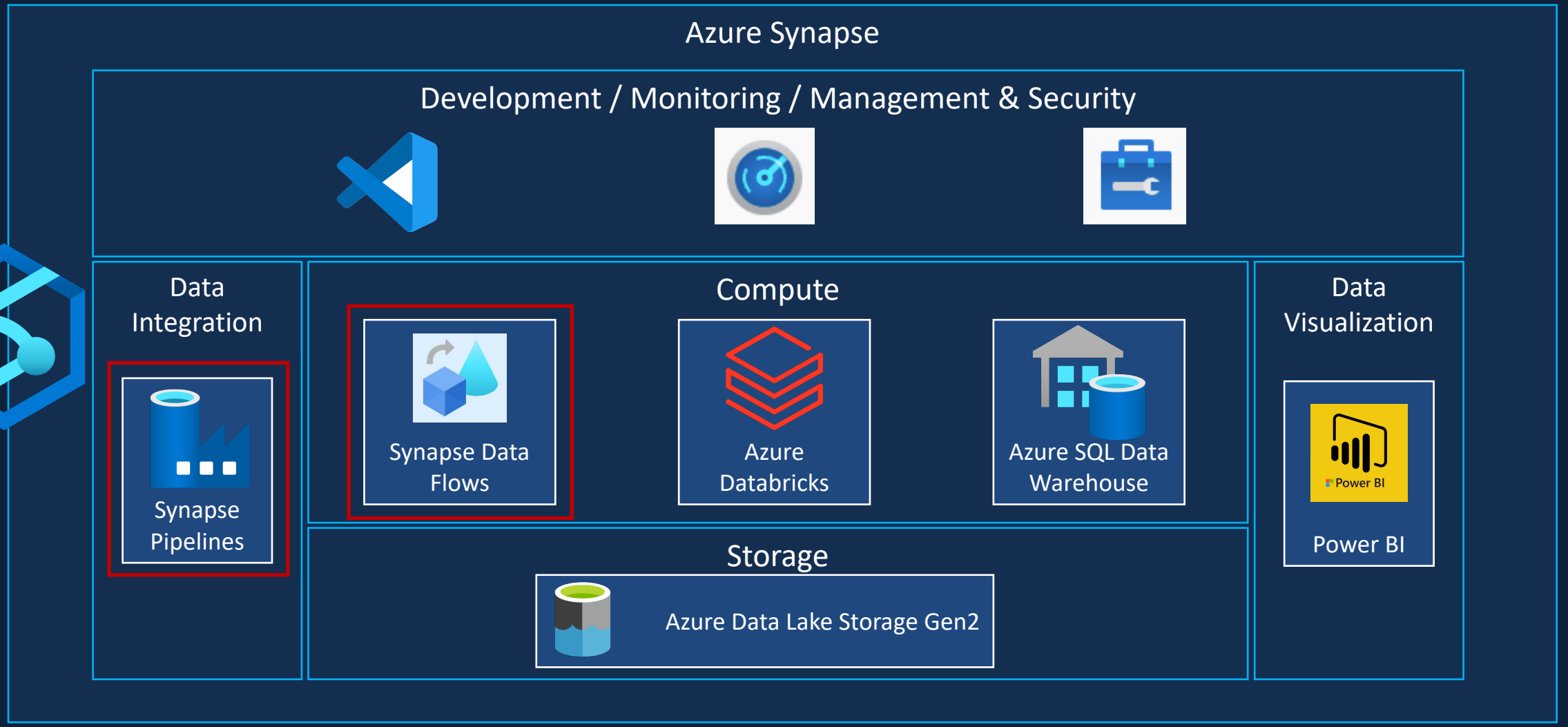
No Serverless option

Metadata not shared

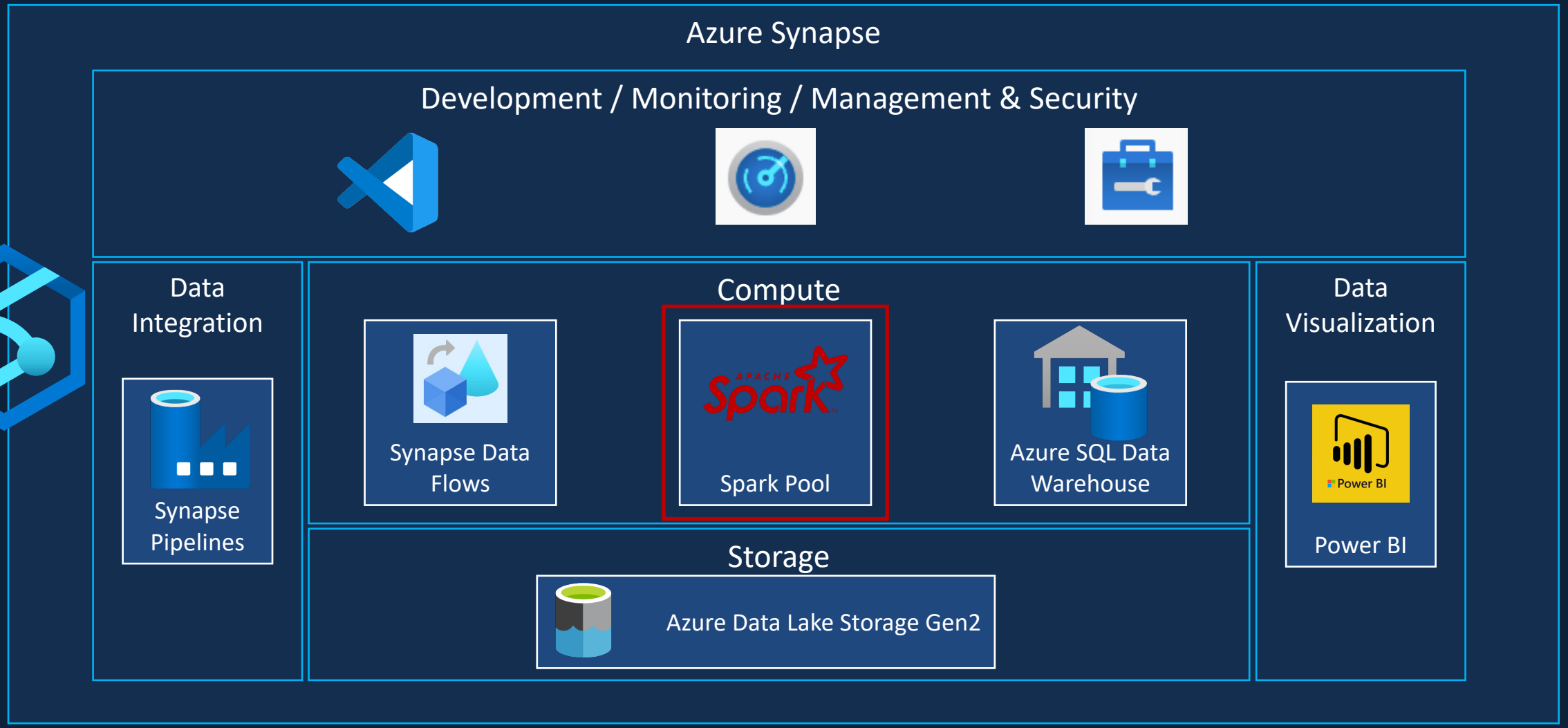
Azure Synapse Analytics



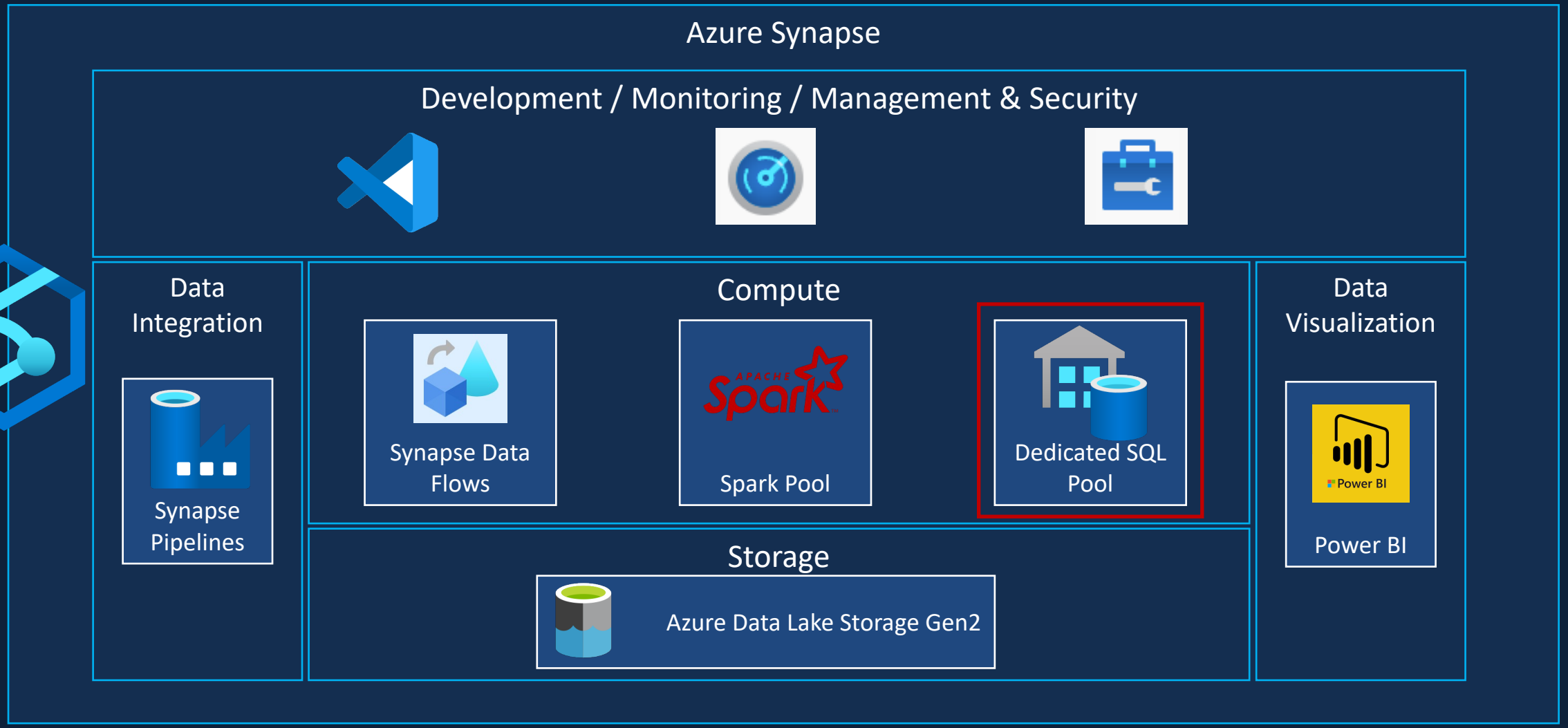
Azure Synapse Analytics



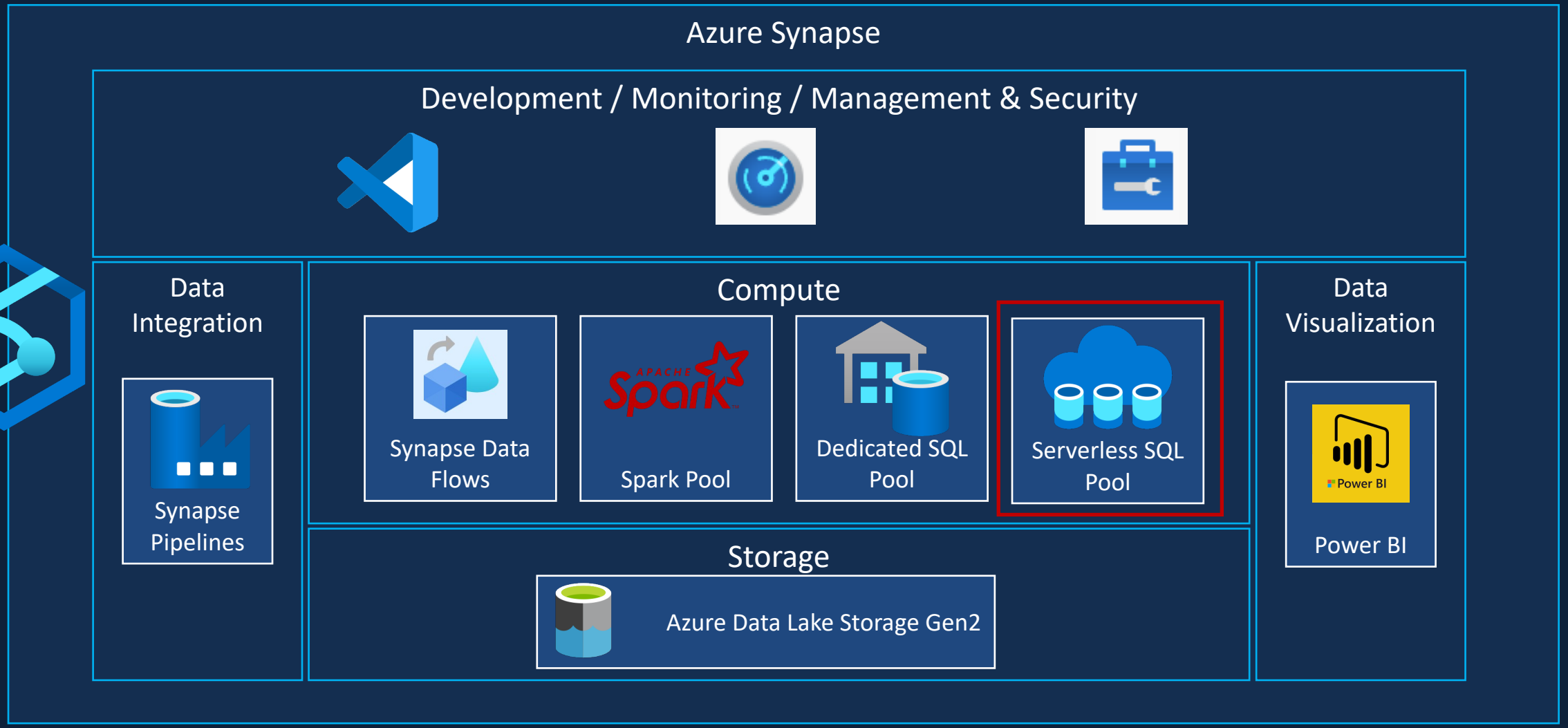
Azure Synapse Analytics



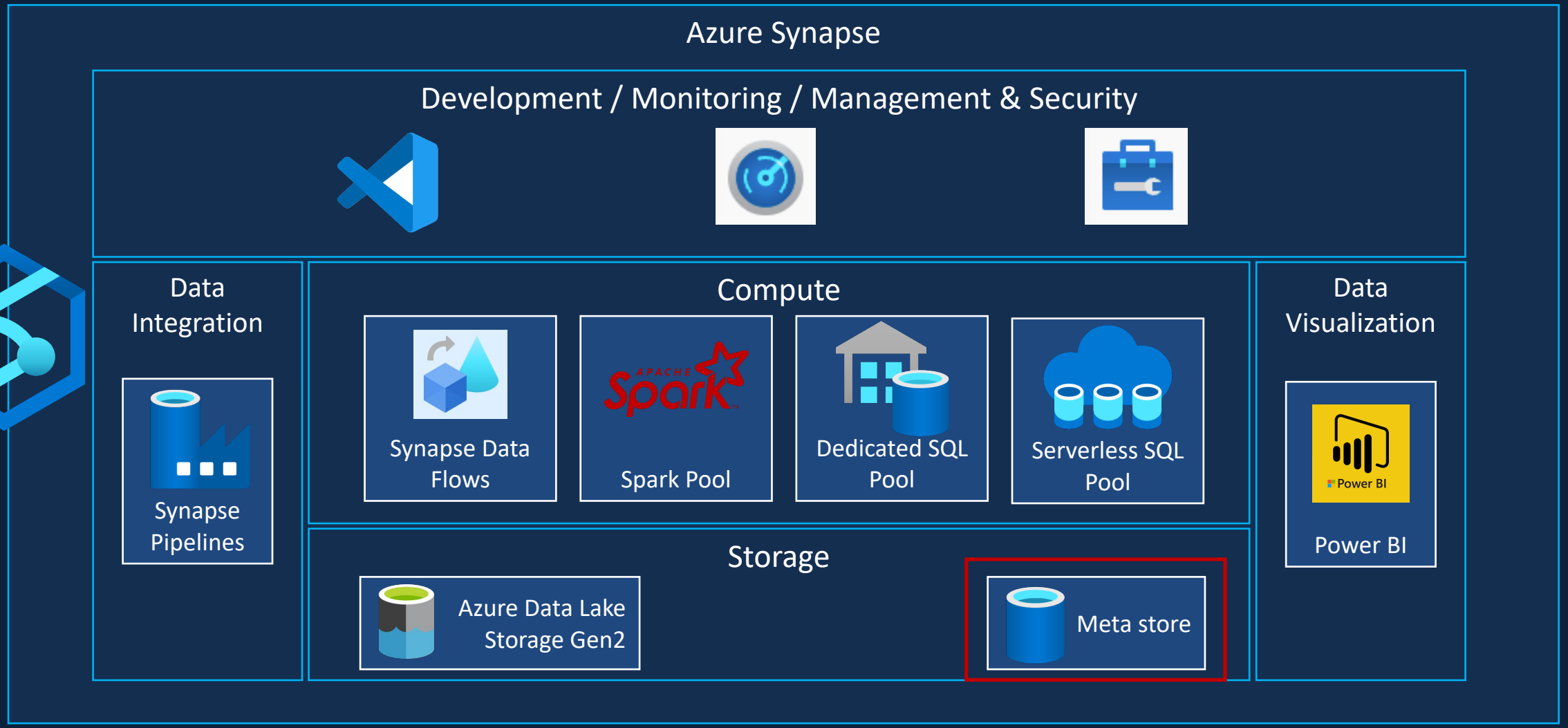
Azure Synapse Analytics



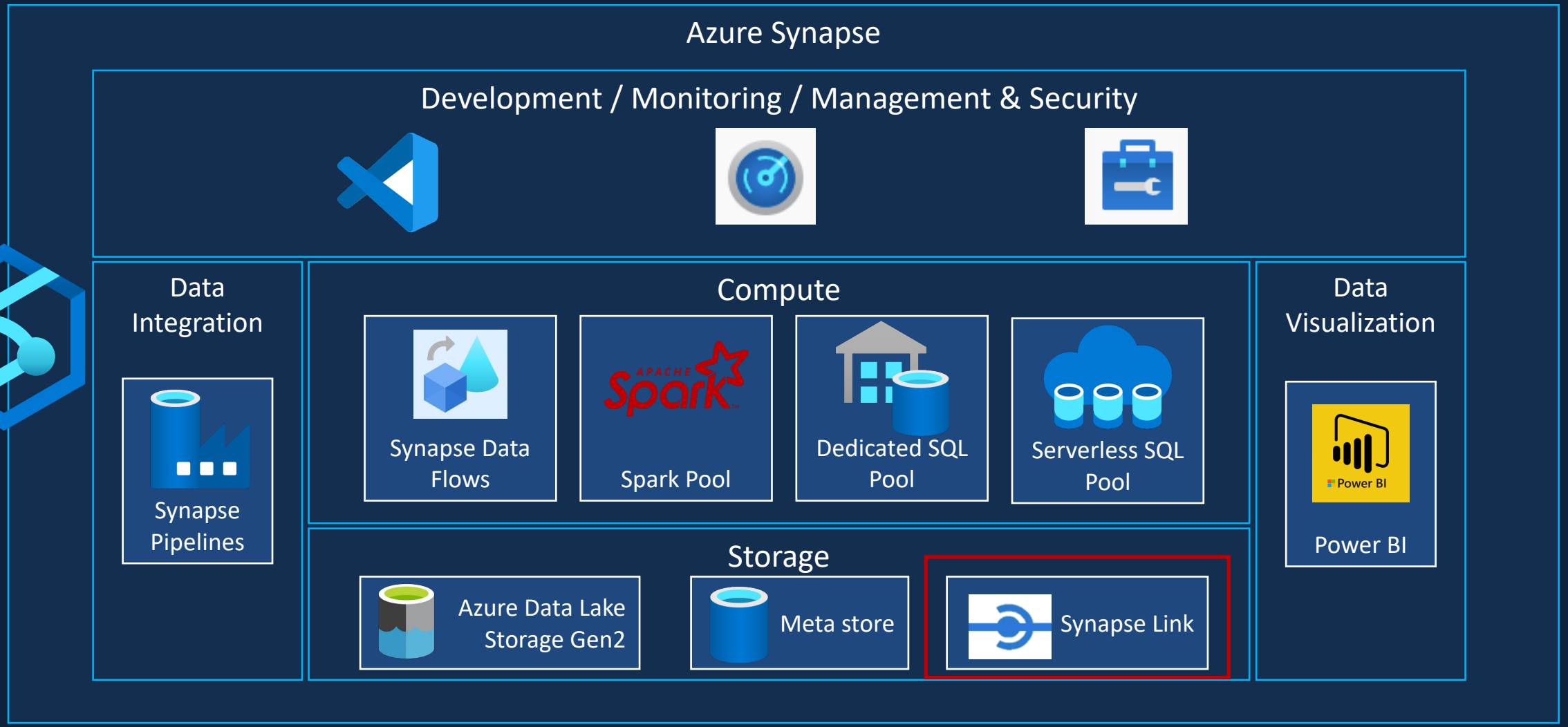
Azure Synapse Analytics



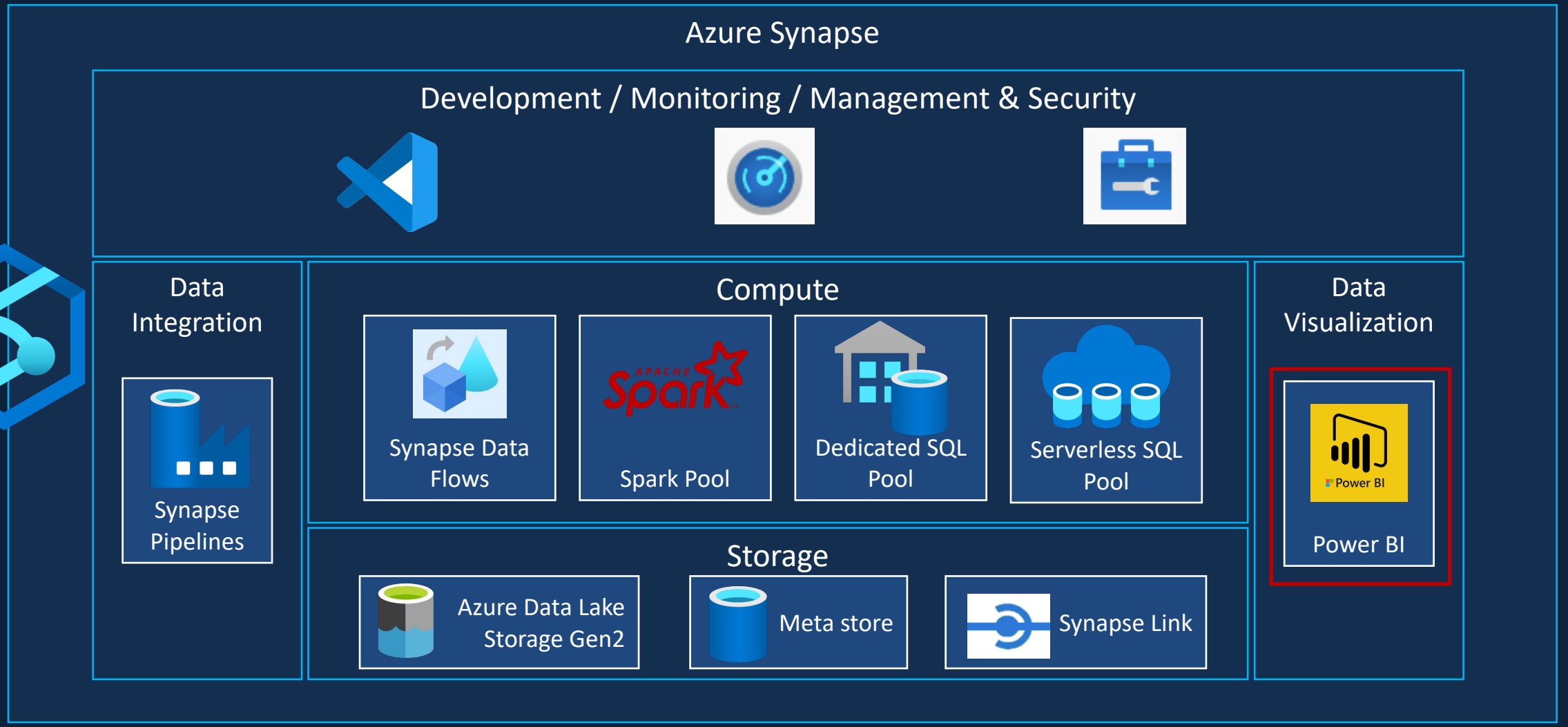
Azure Synapse Analytics



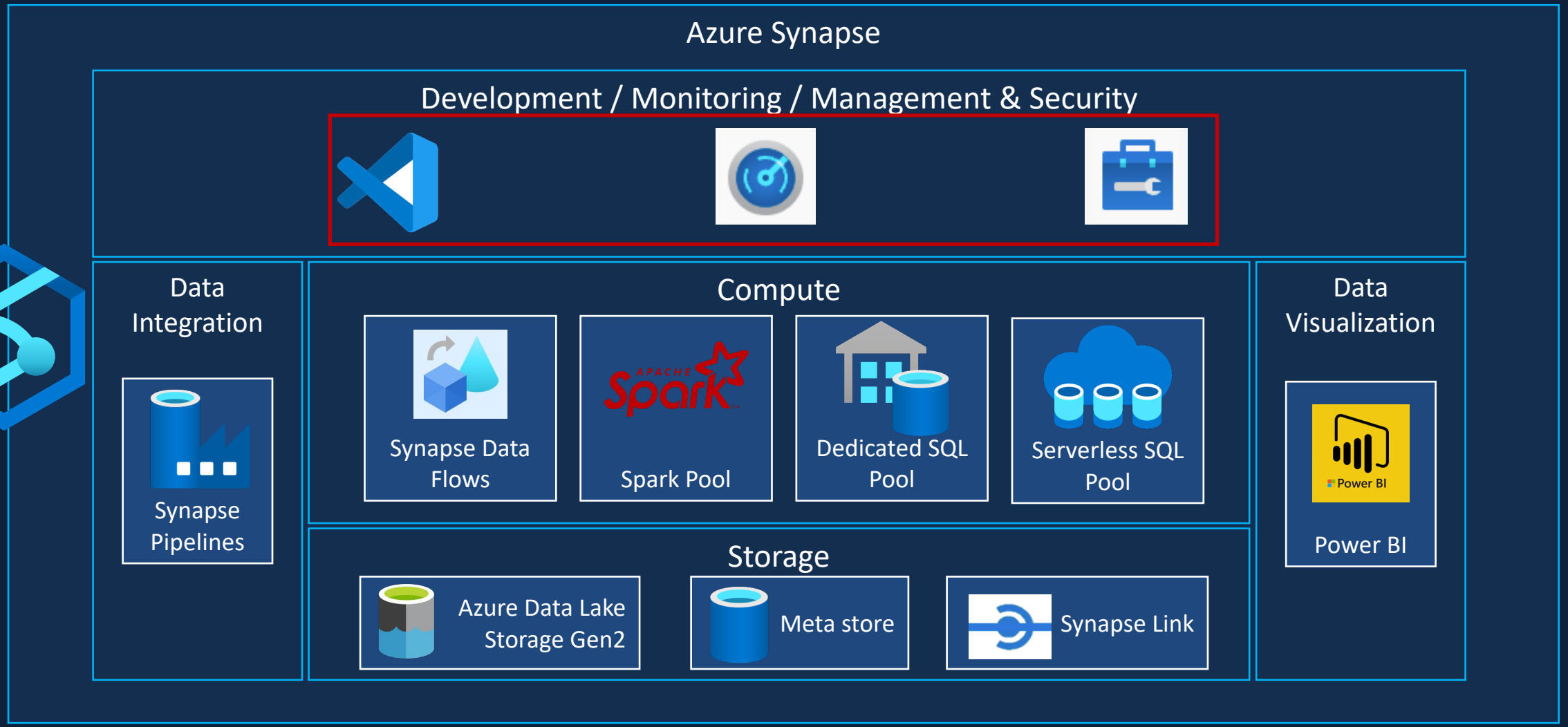
Azure Synapse Analytics



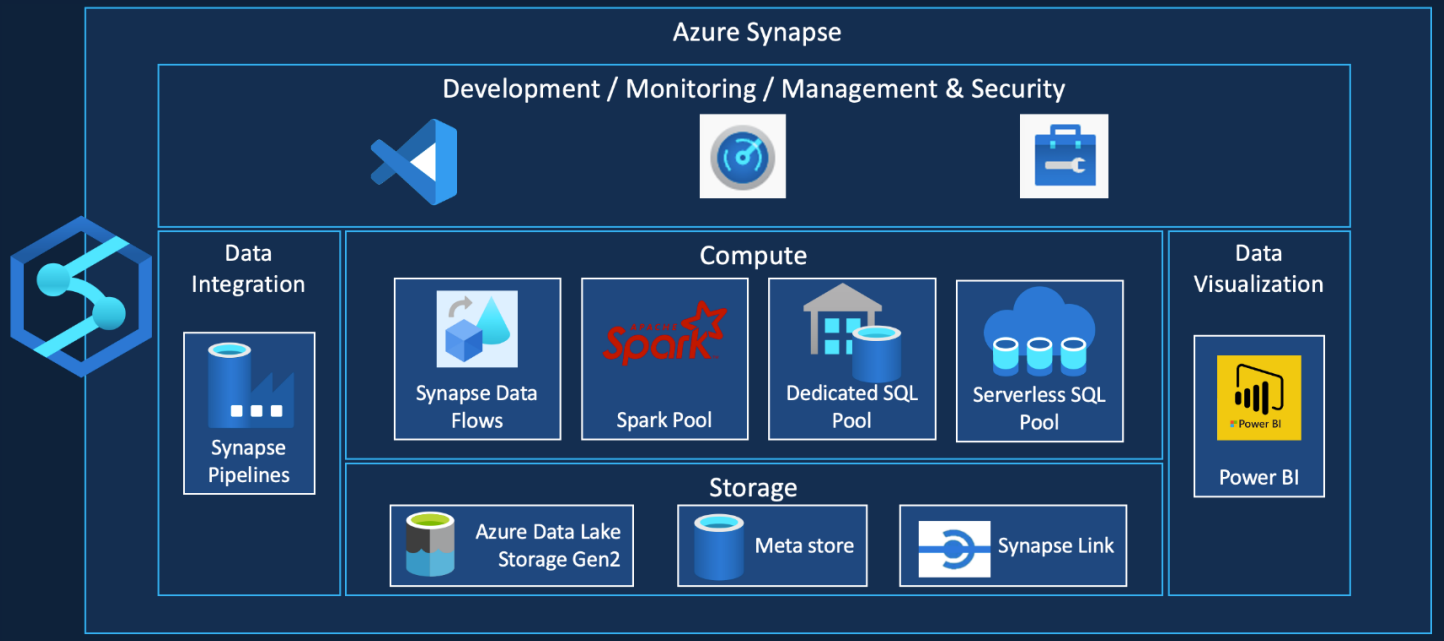
Azure Synapse Analytics



Azure Synapse Analytics



Azure Synapse Analytics – Preview Features



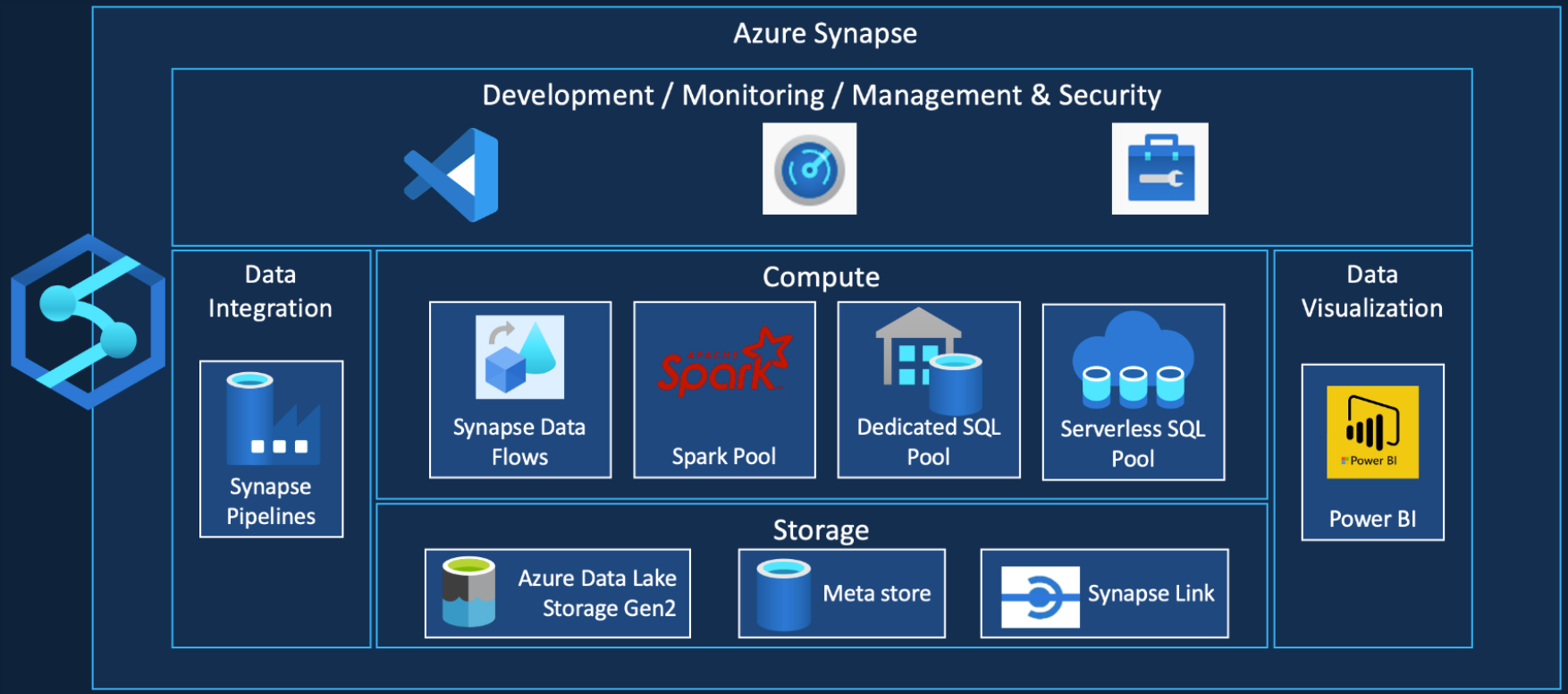
Data Explorer Pool

Synapse Link for SQL Server 2022

Many more features

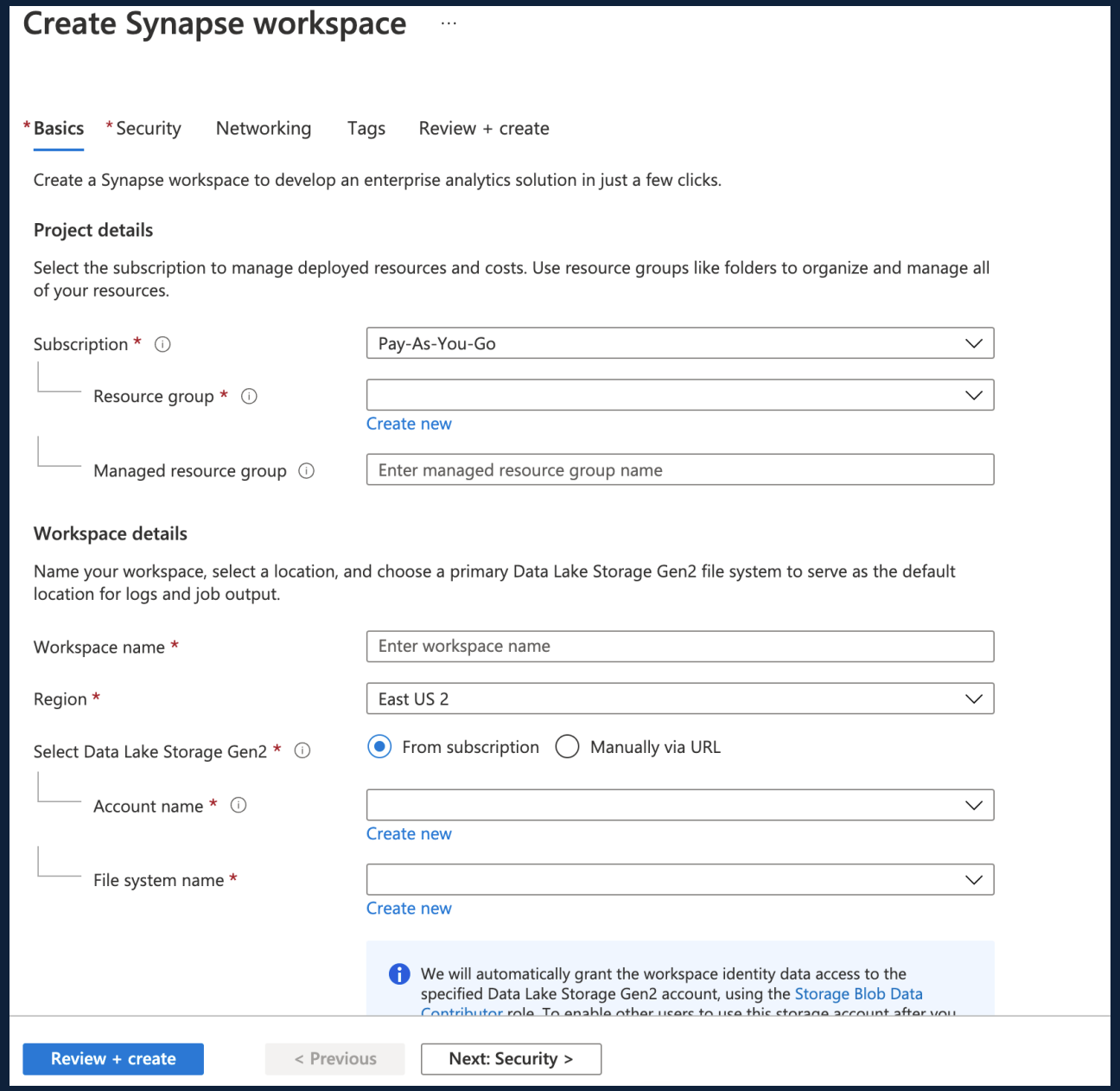
Azure Synapse Analytics

Azure Synapse Analytics is a limitless analytics service that brings together data integration, enterprise data warehousing and big data analytics.

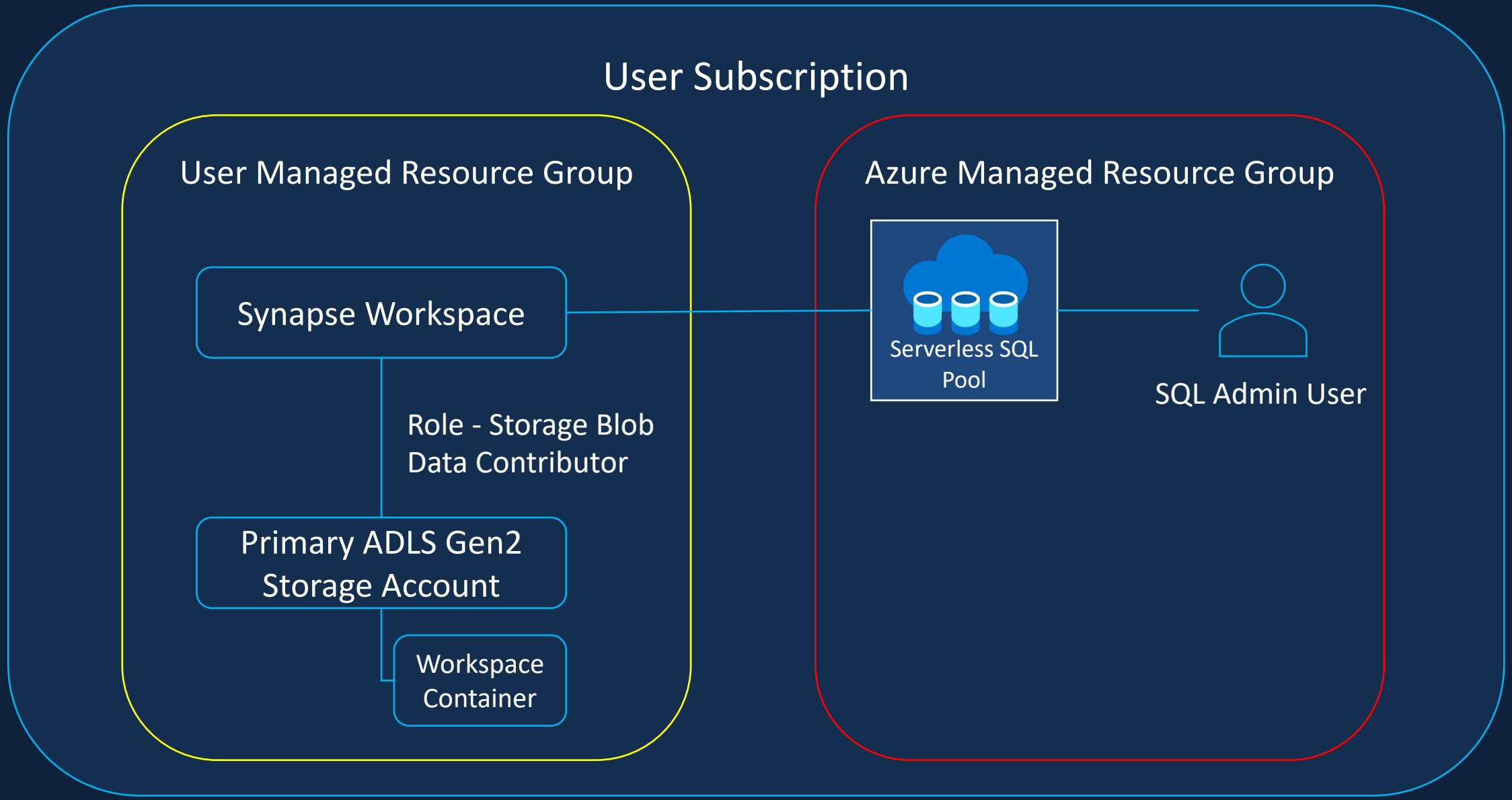


Create Synapse Analytics Workspace Lab

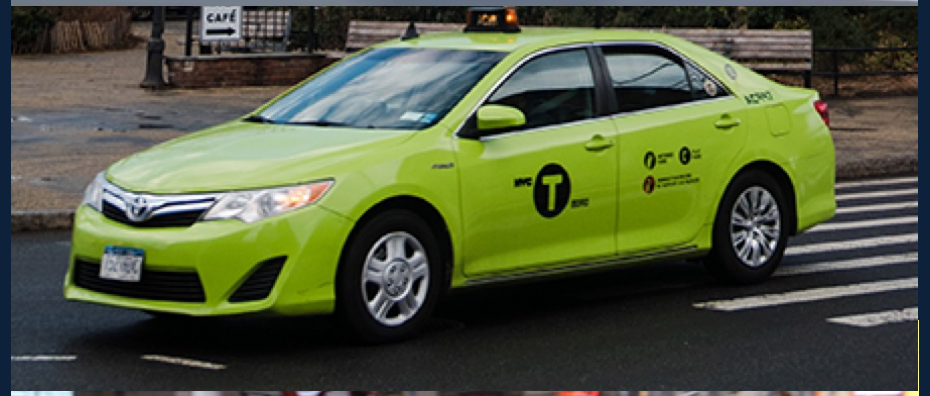
Create Synapse Analytics Workspace



Create Synapse Analytics Workspace



Project Overview



- What is NYC Taxi
- NYC Taxi data source & datasets
- Prepare the data for the project
- Project Requirements
- Solution Architecture

Data Overview – NYC Taxi Trips

Data Overview – NYC Taxis

Yellow Taxis

Data Overview – NYC Taxis

Yellow Taxis

Green Taxis

Data Overview – NYC Taxis

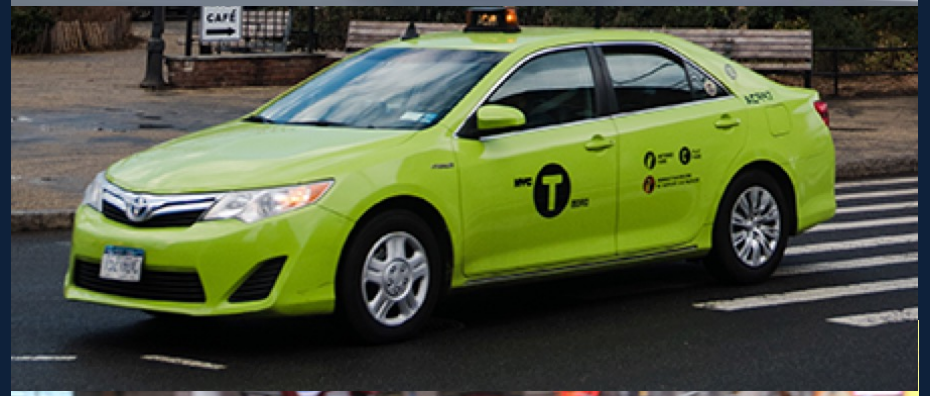
Yellow Taxis

Green Taxis

For Hire Vehicles

High Volume For Hire Vehicles

Data Overview – NYC Taxis



Yellow Taxis

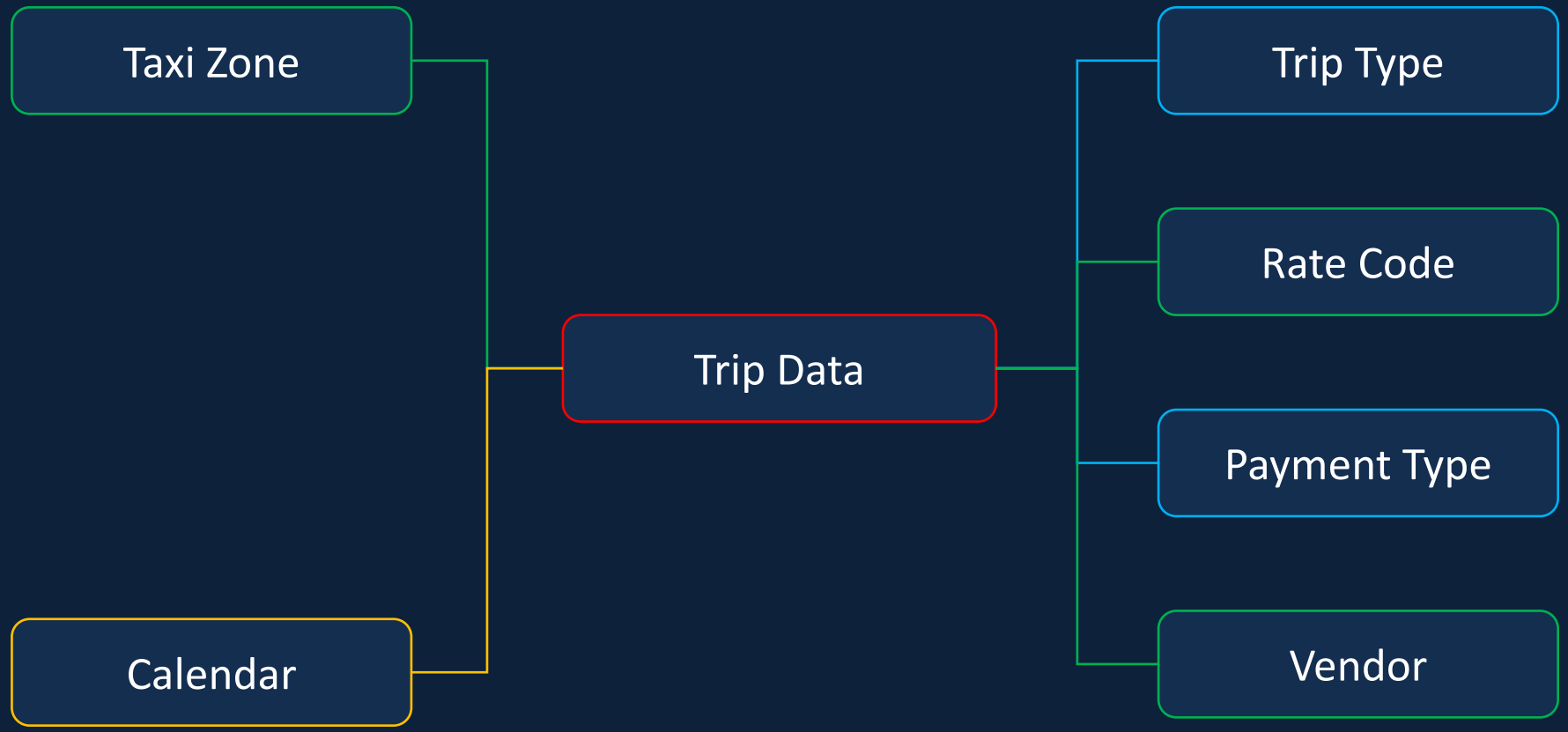
Green Taxis

For Hire Vehicles

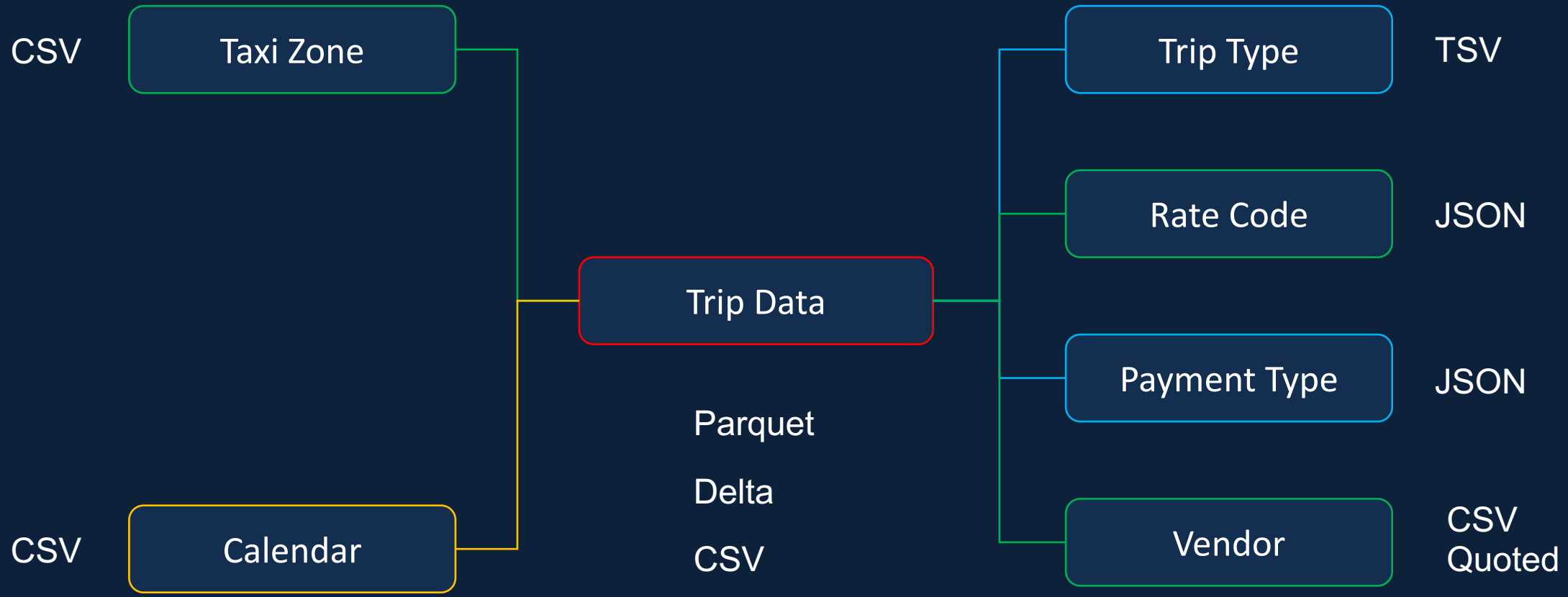
High Volume For Hire Vehicles

https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

NYC Taxi Data Files Overview



NYC Taxi Data Files Overview



Import NYC Taxi Data to Data Lake

Project Requirements

Data Discovery

- Data exploration capability on the raw data
- Schema applied to the raw data
- Discovery using T-SQL
- Discovery using pay-per-query model

Data Ingestion

- Ingested data to be stored as Parquet
- Ingested data to be stored as tables/ views
- Ability to query the ingested data using SQL
- Ingestion using pay-per-query model

Data Transformation

Join the key information required for reporting to create a new table.

Join the key information required for Analysis to create a new table.

Must be able to analyze the transformed data via T-SQL

Transformed data must be stored in columnar format (i.e., Parquet)

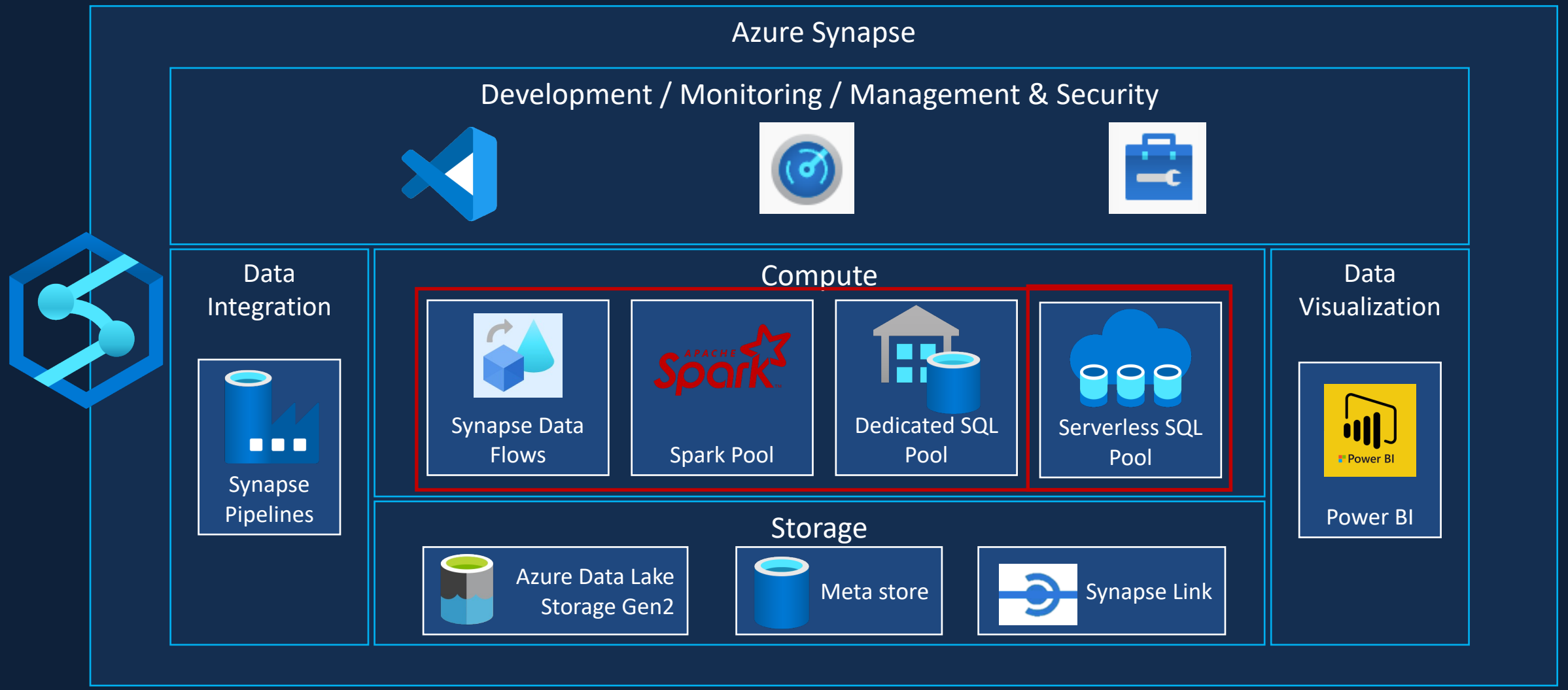
Reporting Requirements

- Taxi Demand
- Credit Card Campaign
- Operational Reporting

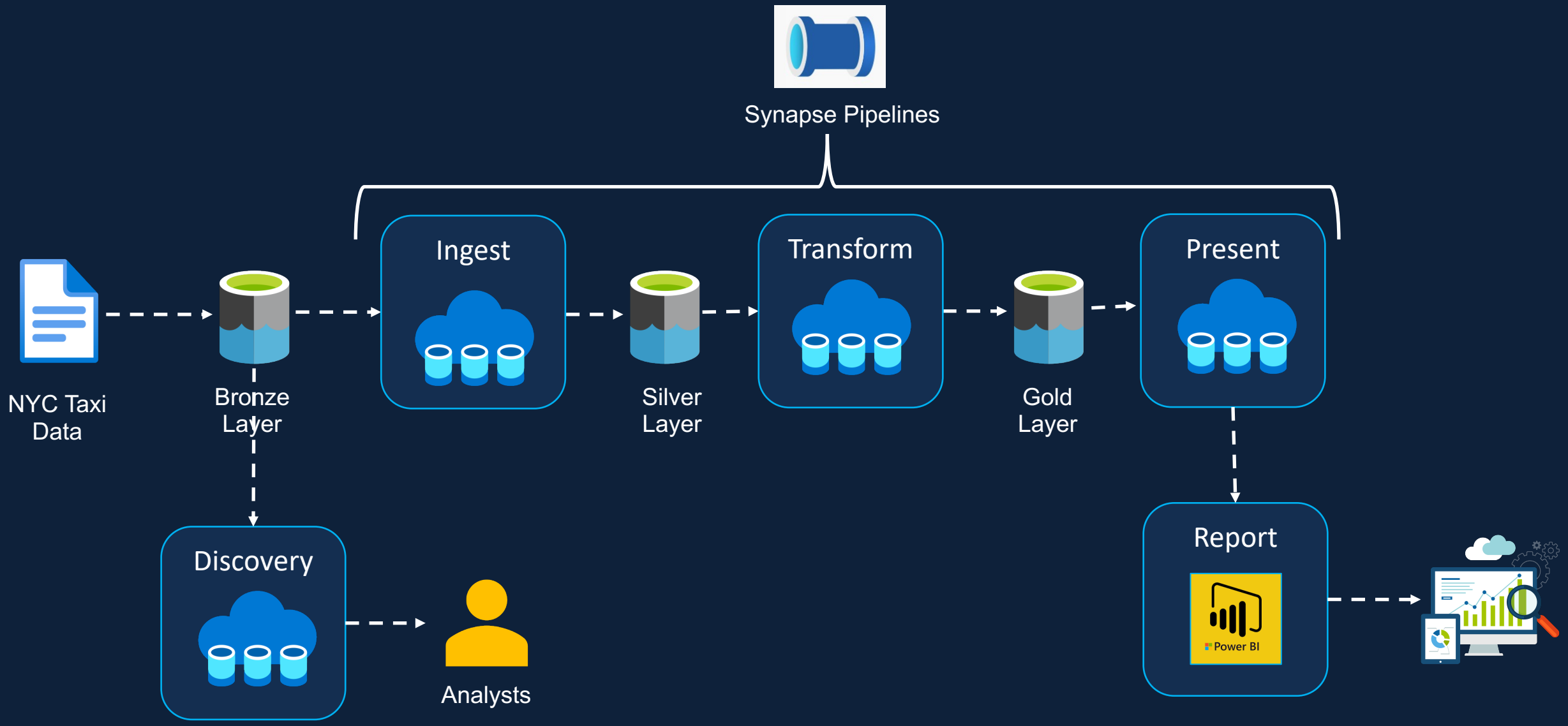
Scheduling Requirements

- Scheduled to run at regular interval
- Ability to monitor pipelines
- Ability to re-run failed pipelines
- Ability to set-up alerts on failures

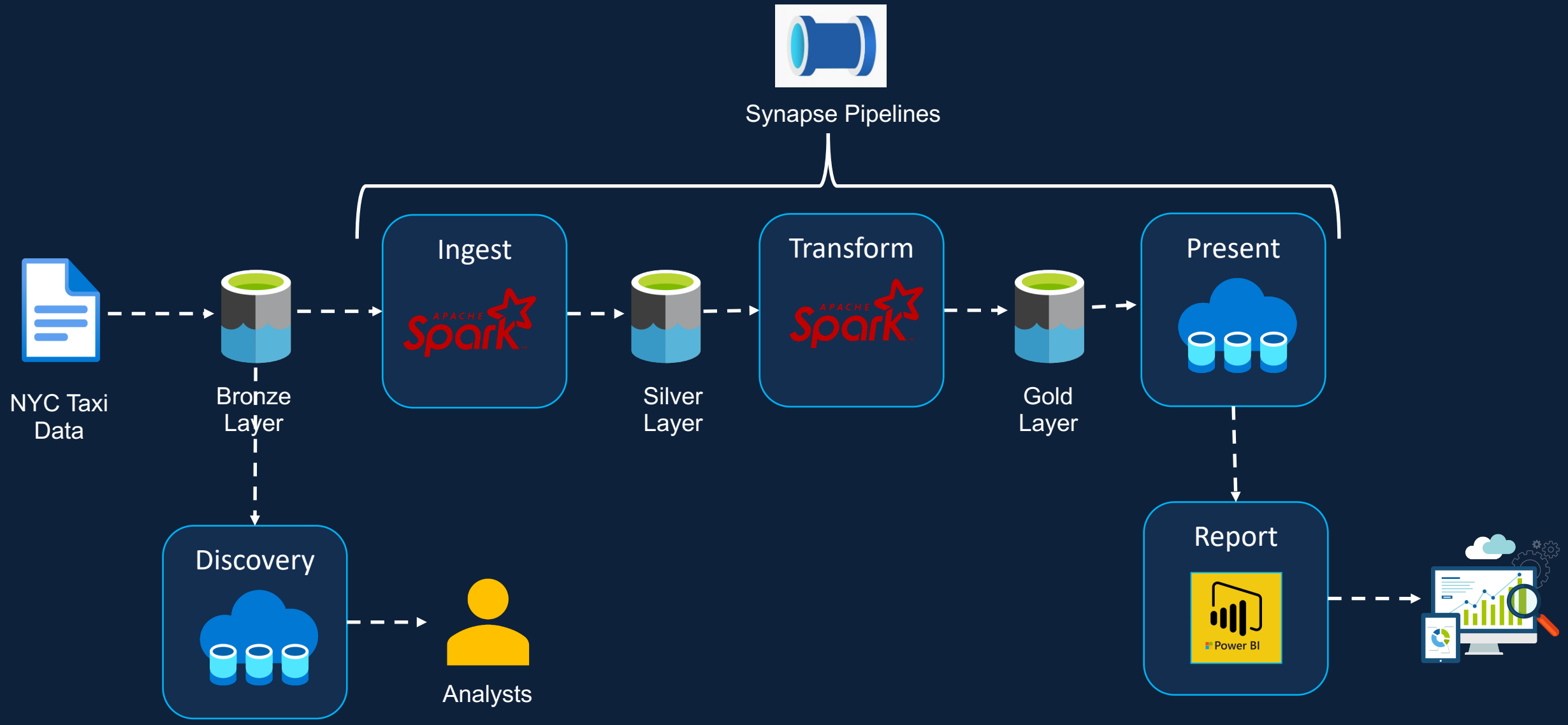
Solution Architecture



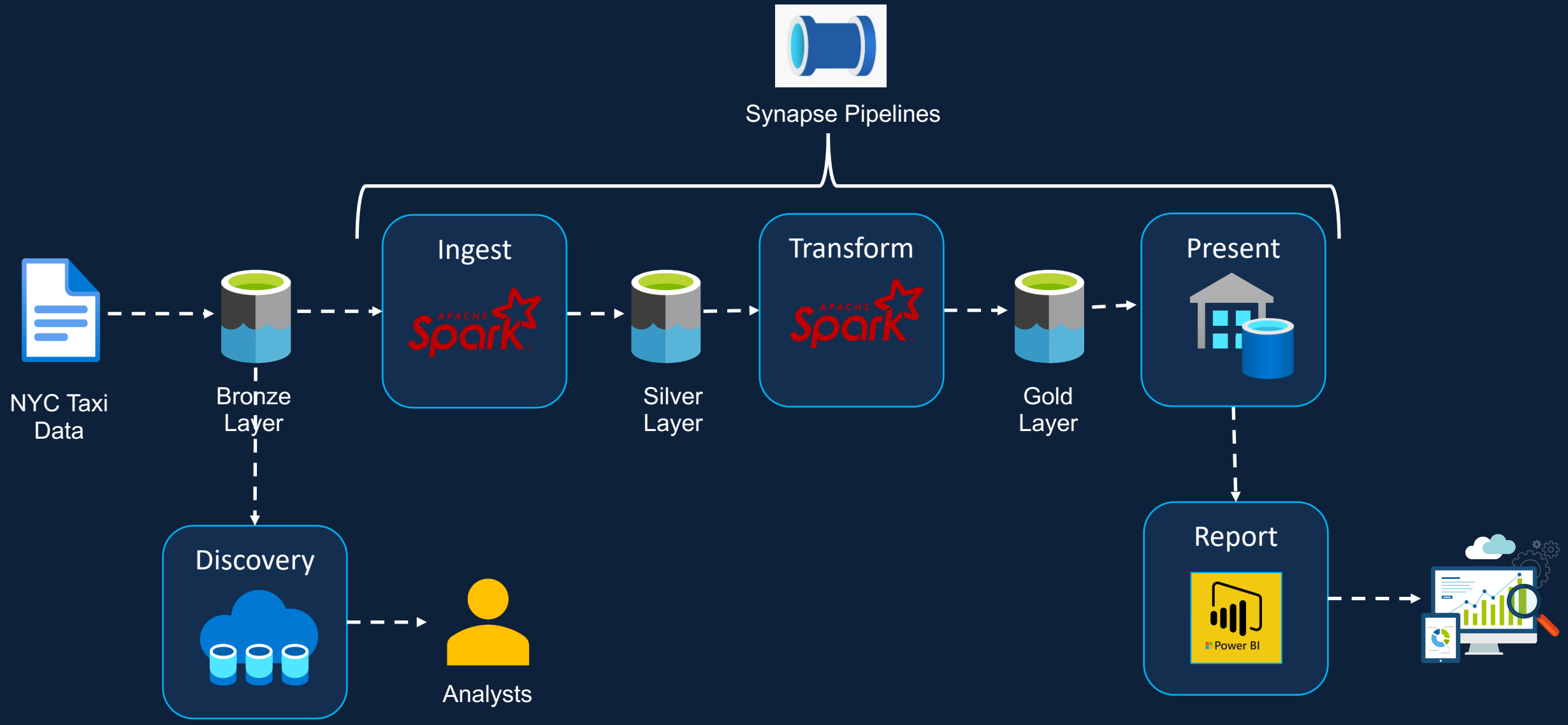
Solution Architecture – Serverless SQL Pool



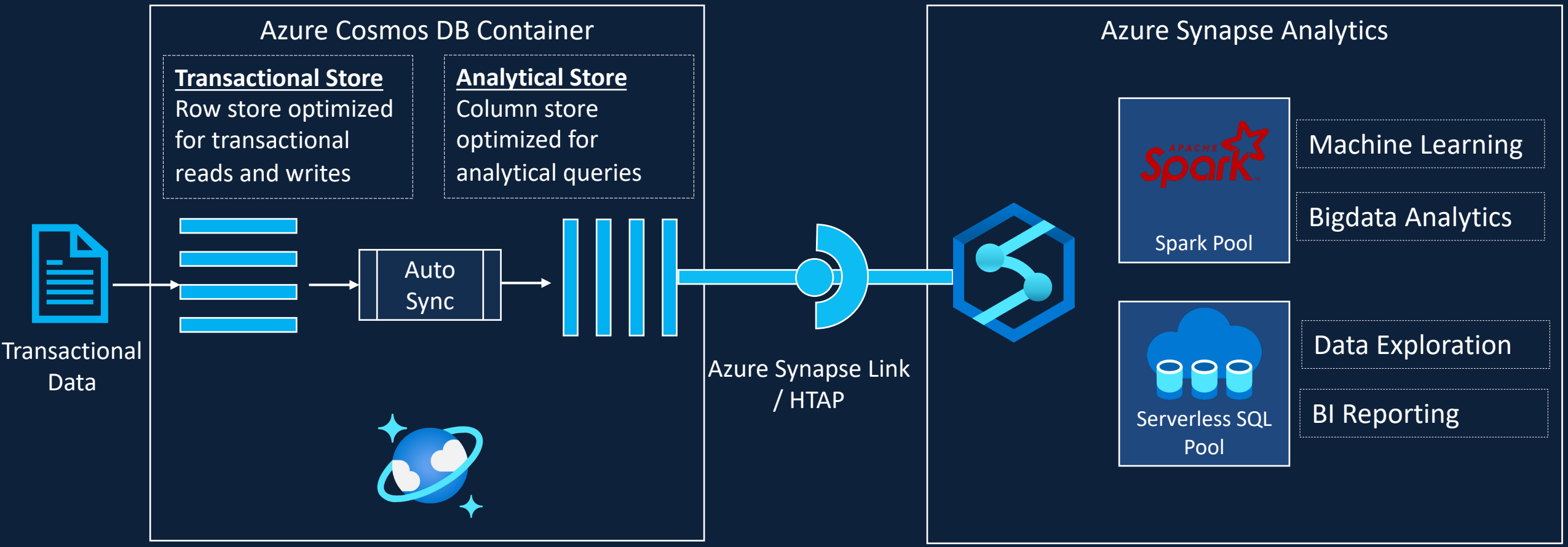
Solution Architecture – Spark Pool



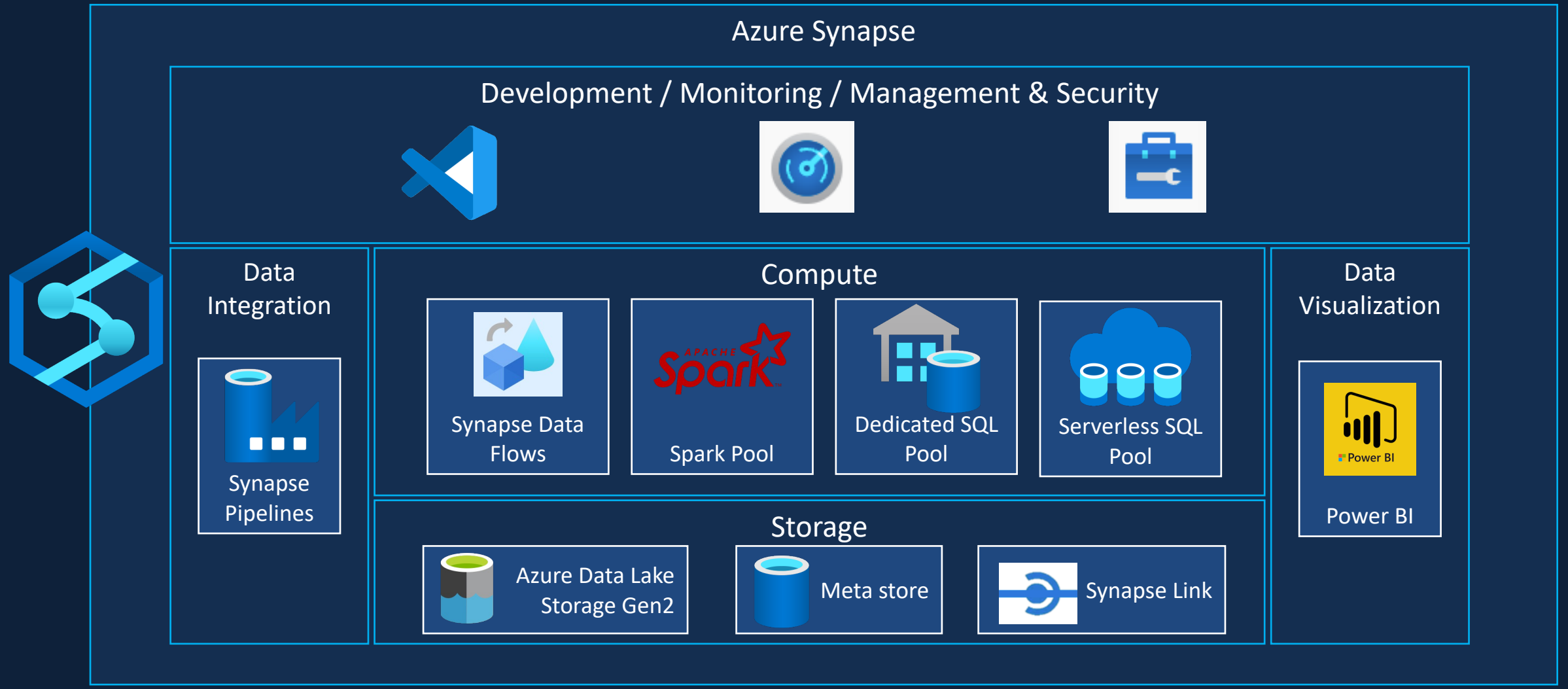
Solution Architecture – Dedicated SQL Pool



Solution Architecture - Synapse Link



Azure Synapse Analytics

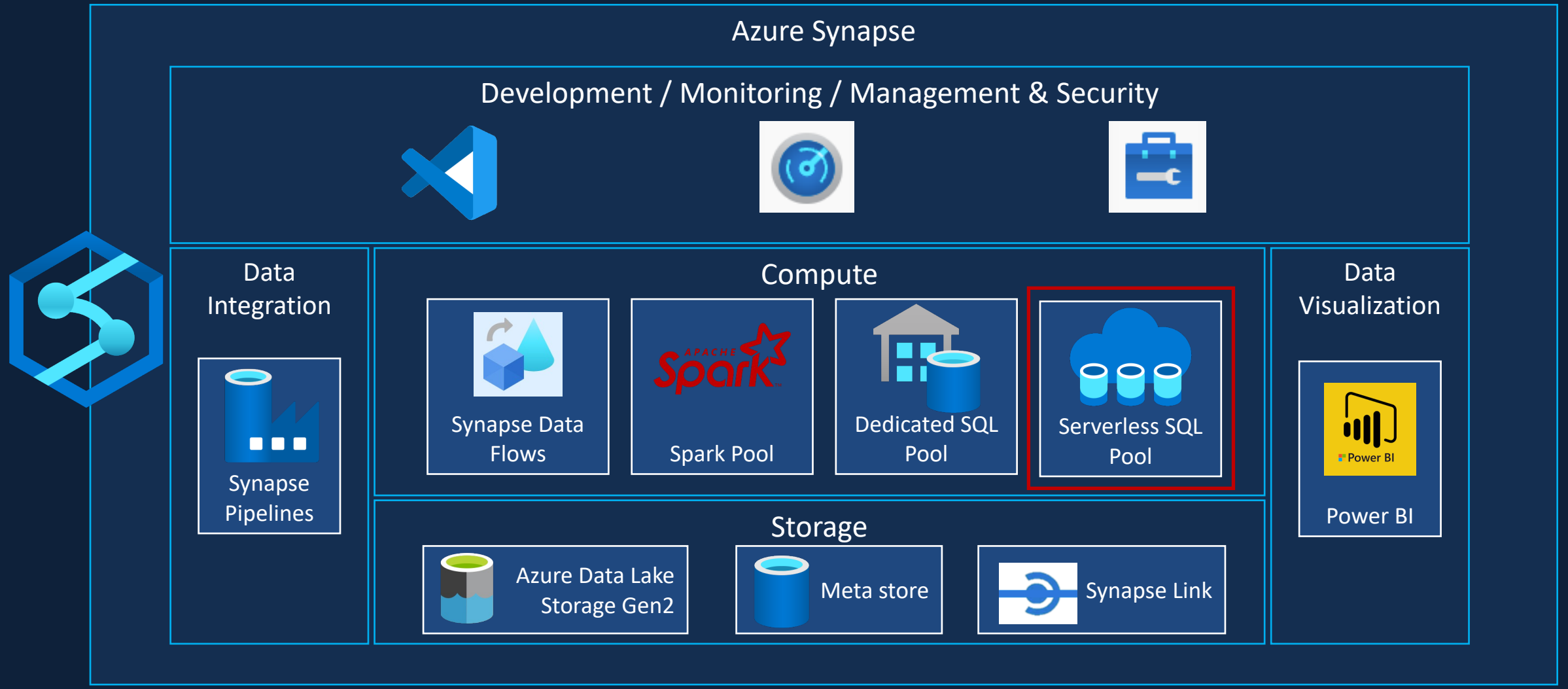


Section Overview – Serverless SQL Pool

- Serverless SQL Pool Architecture
- Features & Use Cases
- Cost Control
- Connecting to Azure Data Studio
- T-SQL Support

Serverless SQL Pool

Serverless SQL Pool



Serverless SQL Pool

Serverless SQL pool is a serverless distributed query engine that you can use to query data over your data lake using T-SQL.

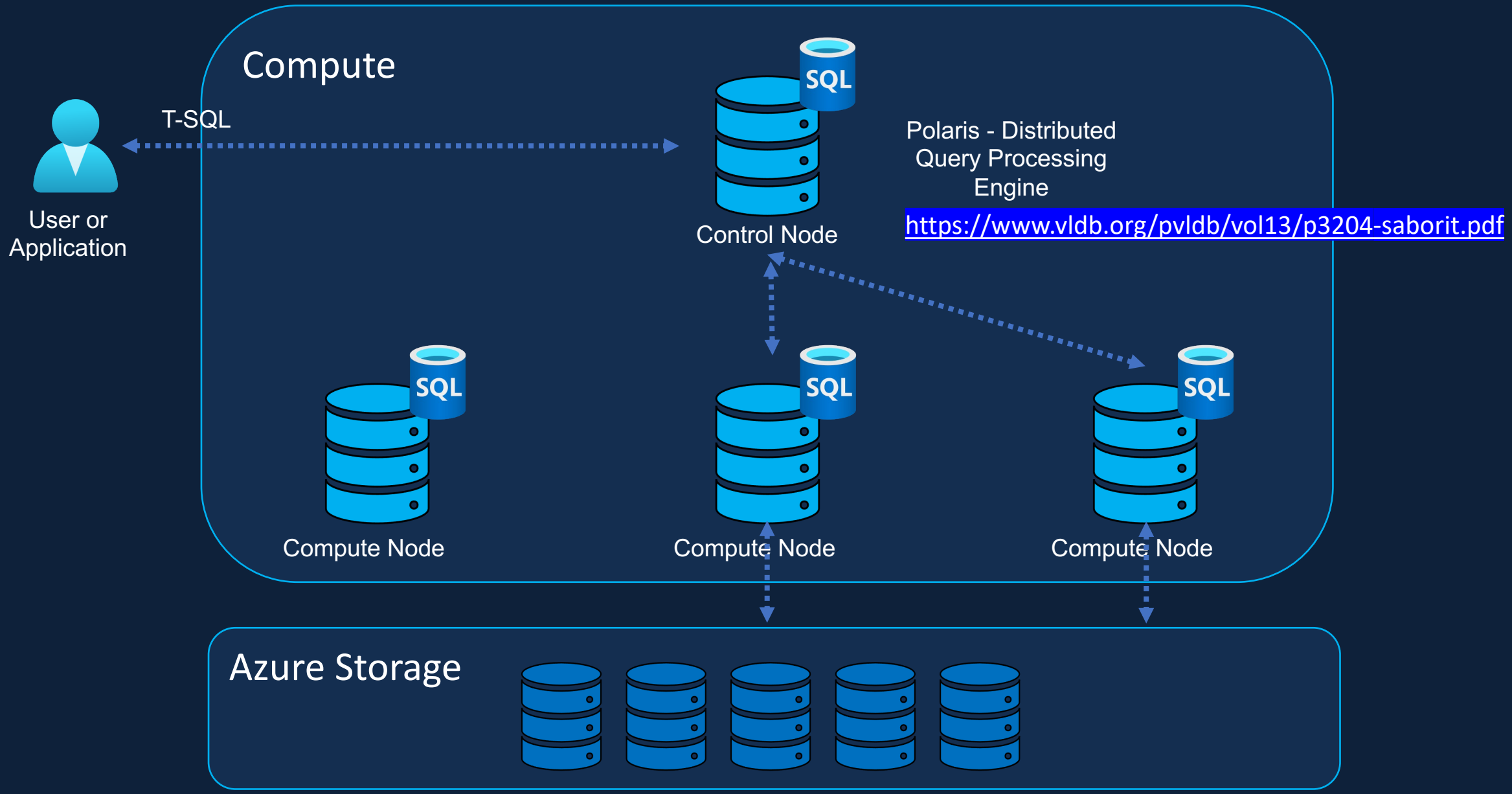
Serverless SQL Pool

Serverless SQL pool is a serverless distributed query engine that you can use to query data over your data lake using T-SQL.

Serverless SQL Pool

Serverless SQL pool is a serverless distributed query engine that you can use to query data over your data lake using T-SQL.

Serverless SQL Pool - Architecture



Serverless SQL Pool – Key Features

- Serverless
- Distributed query engine
- Robust
- Query using T-SQL
- Pay-per-query pricing model
- Not a storage
- Synapse Link
- Query spark tables

Serverless SQL Pool – Supported Data Sources

- Azure Storage Account
  - Delimited – CSV, TSV etc
  - JSON
  - Parquet
  - Delta Lake
- Cosmos
  - SQL API
  - MongoDB API
- Dataverse
- SQL Server 2022 (Preview)

Serverless SQL Pool – Use Cases

Discovery & Exploration

Logical Data Warehouse

Data Transformation

Serverless SQL Pool – Who is it far?

Data Engineers

Data Scientists

Data Analysts / BI Developers

Serverless SQL Pool

Cost Management

Cost Calculation

Billed for Data Processed

Amount of data read from storage

Amount of data in intermediate results

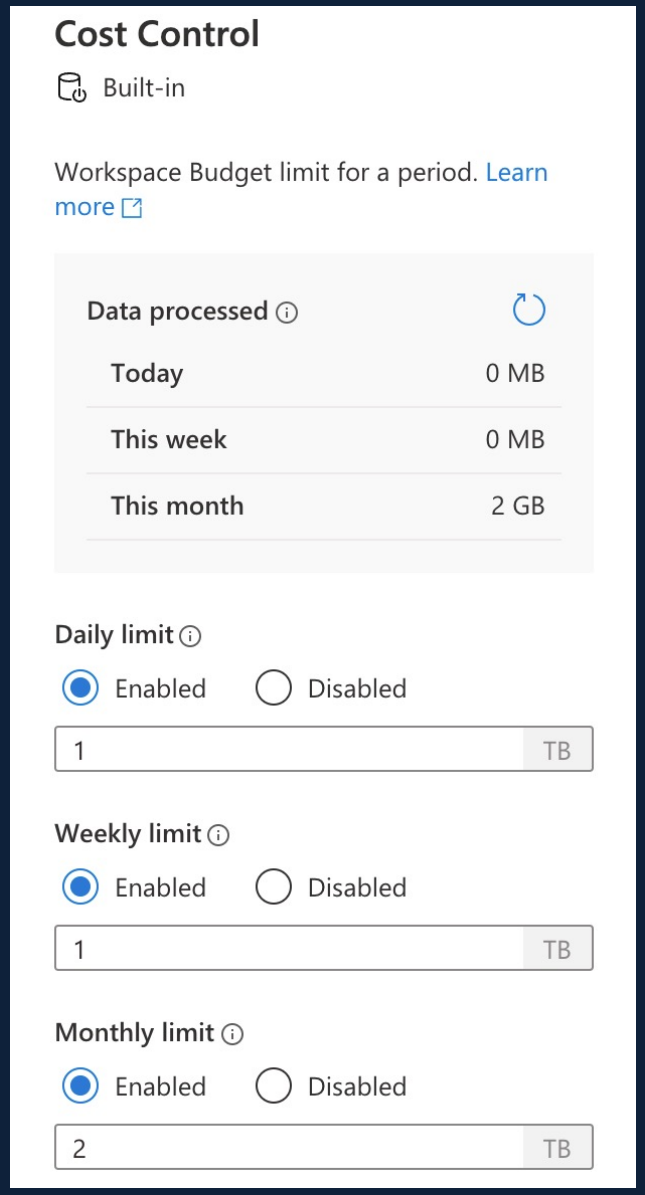
Amount of data written to storage

Data Processed rounded to the nearest MB

Minimum of 10MB per query

Currently $6.25 per 1TB

Cost Control



Serverless SQL Pool

Cost Management

Lab

Azure Data Studio

Section Overview - Query CSV Files

Header Row

Field Terminator

Row Terminator

Quoted Files and escaping characters

Query CSV Files

OPENROWSET function overview

Query using OPENROWSET function

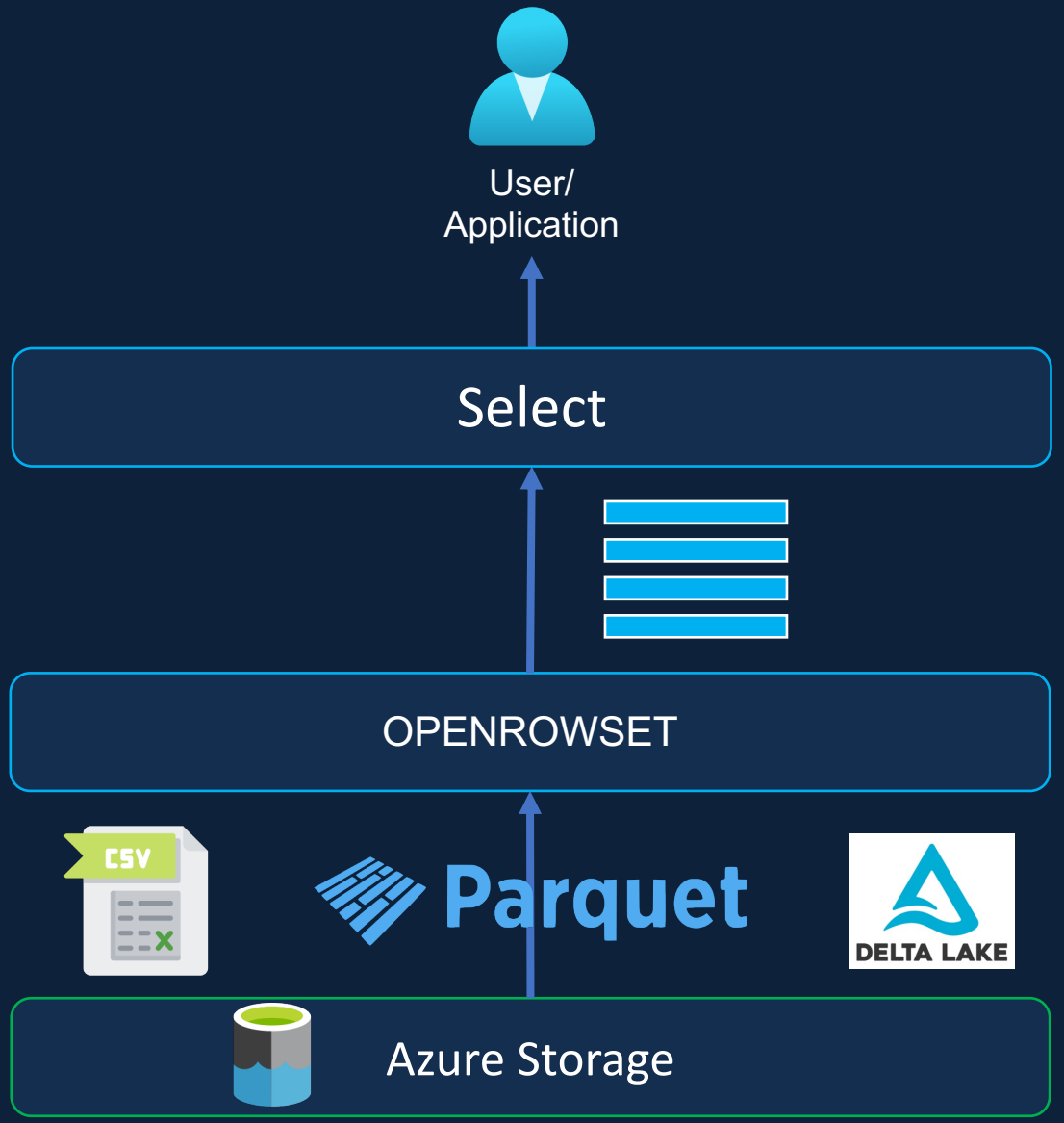
Data Types & Collations

Query subset of columns

Quoted strings & Escape Char

Query Tab Separated Values (TSV) file

OPENROWSET Function




```
SELECT *
FROM OPENROWSET(BULK ‘blob file path‘,
                FORMAT = [‘CSV’ |
                          'PARQUET’ |
                          ‘DELTA’]
                ) AS [file]
```

Section Overview - Query CSV Files

Line Delimited JSON

payment_type.json

Scroll Tabs

```
1 {"payment_type":1,"payment_type_desc":"Credit card"}
2 {"payment_type":2,"payment_type_desc":"Cash"}
```

Section Overview - Query CSV Files

Single Line JSON

Standard JSON

rate_code.json

```
[
 {"rate_code_id":1,"rate_code":"Standard rate"},
 {"rate_code_id":2,"rate_code":"JFK"}
]
```

Section Overview - Query CSV Files

Single Line JSON

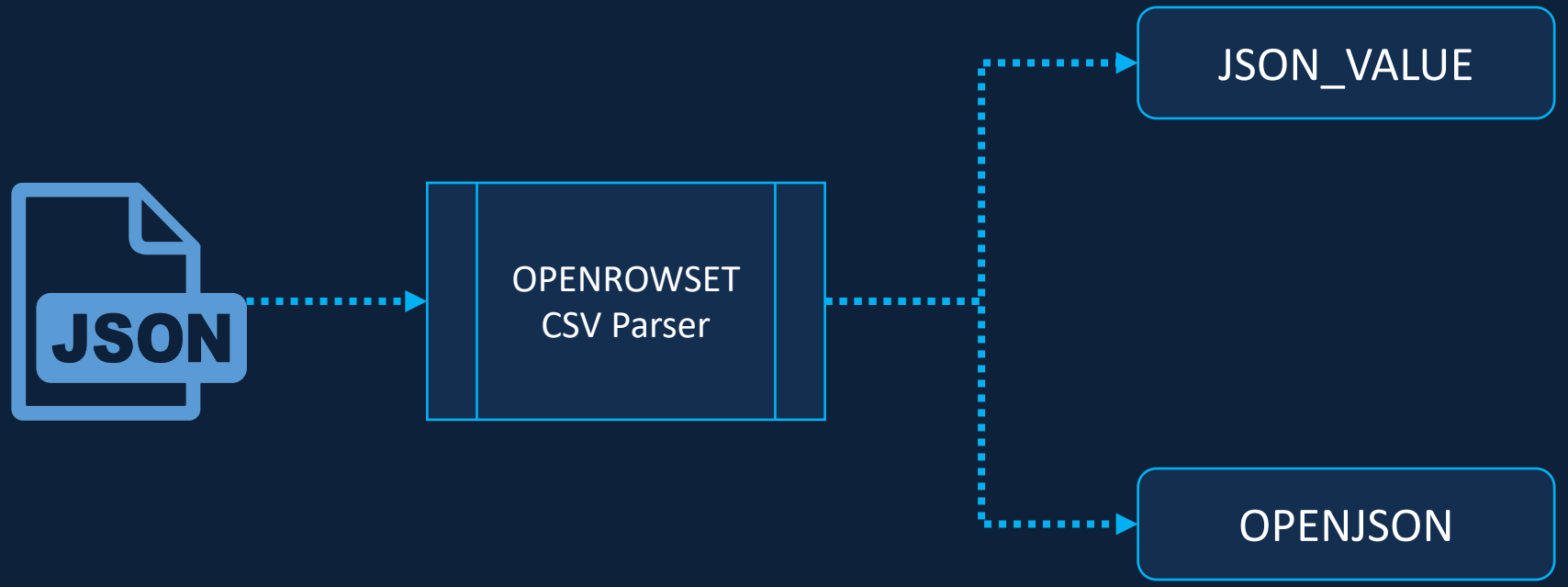
Standard JSON

Classic JSON

rate_code_multi_line.json

```
[
  {
    "rate_code_id": 1,
    "rate_code": "Standard rate"
  },
  {
    "rate_code_id": 2,
    "rate_code": "JFK"
  },
]
```

Query JSON Files



Line-delimited JSON

FIELDTERMINATOR – 0x0b
FIELDQUOTE – 0x0b

Standard JSON

FIELDTERMINATOR – 0x0b
FIELDQUOTE – 0x0b
ROWTERMINATOR – 0x0b

Serverless SQL Pool – T-SQL Support

Supported

- Databases
- Schemas
- Views
- Stored Procedures
- Inline table value functions
- External Resources – data sources, file formats and tables

Serverless SQL Pool – T-SQL Support

Not Supported

Tables

Triggers

Materialized views

DML statements

DDL statements other than ones related to views and security

Serverless SQL Pool – T-SQL Support

Security

Logins and users

Credentials to control access to storage accounts

Grant, deny, and revoke permissions per object level

Azure Active Directory integration

Serverless SQL Pool – T-SQL Support

Additional Features

CETAS - CREATE EXTERNAL TABLE AS SELECT

Extension to OPENROWSET to aid querying data in data lake

Serverless SQL Pool – Monitoring Queries Lab

Discovery & Exploration Lab

Serverless SQL – Data Discovery

- Identify the volume of the data
  - Total record count
  - Record count per day/ Week/ Month
- Qualify of data
  - Duplicates
  - Missing values
  - Invalid data
- Ability join datasets (e.g. keys exist)
- Ability to get business value
  - Right columns exist
  - Transformations
  - Aggregations
  - Identify additional data required

Serverless SQL – Data Discovery

Identify duplicates in data

Check for missing data values

Invalid/ Unexpected data in columns

Join data from multiple files

Summarize/ Aggregate data

Apply some transforms

Data Virtualization

Data virtualization is a logical data layer that allows us to combine data from multiple sources at query time without having to write complex ETL pipelines to load the data.

NYC Taxi Data Files Overview

Parquet

DELTA LAKE

CSV Taxi Zone

CSV Calendar

Trip Data

Parquet

Delta

CSV

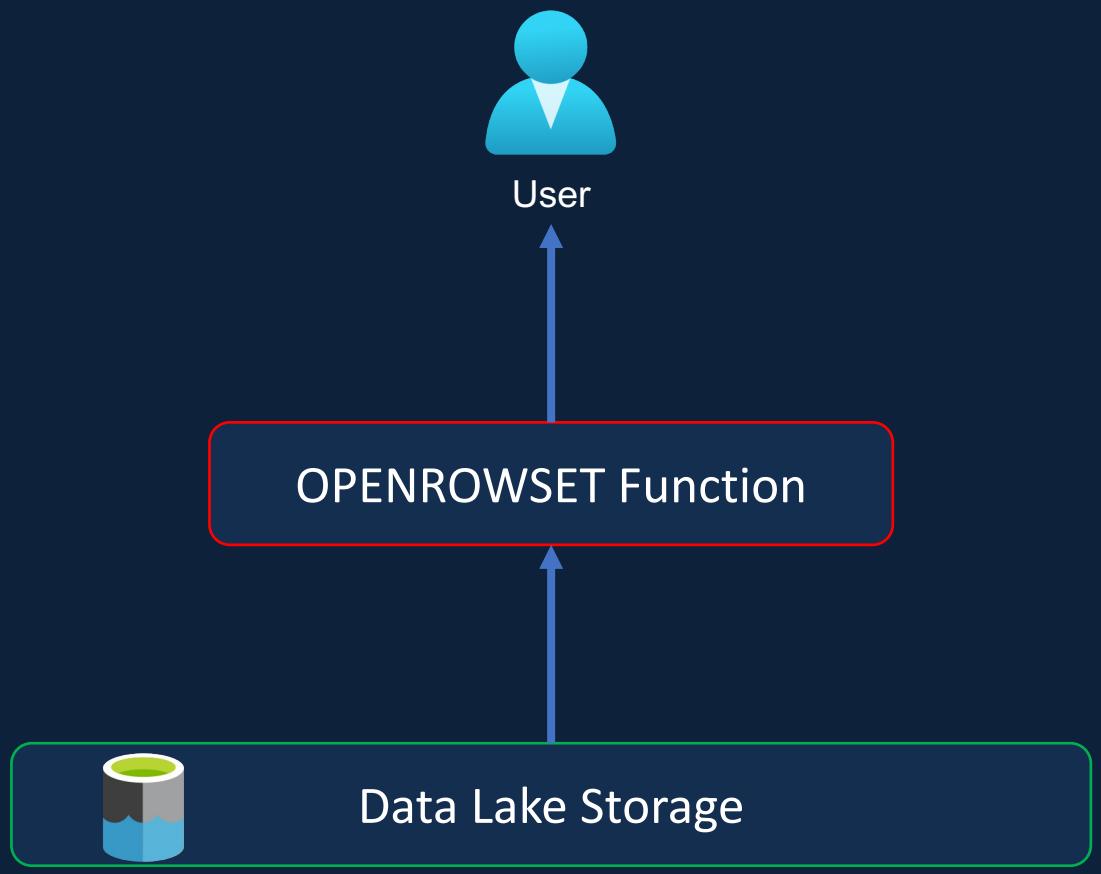
Trip Type — TSV

Rate Code — JSON

Payment Type — JSON

Vendor — CSV Quoted

Database Objects – Why?



Storage Account and File details

File Types

Column Names and Data Types

Code Repetition

Ability to query from BI tools

NYC Taxi Data Files Overview

Parquet

DELTA LAKE

CSV Taxi Zone

CSV Calendar

Trip Data

Parquet

Delta

CSV

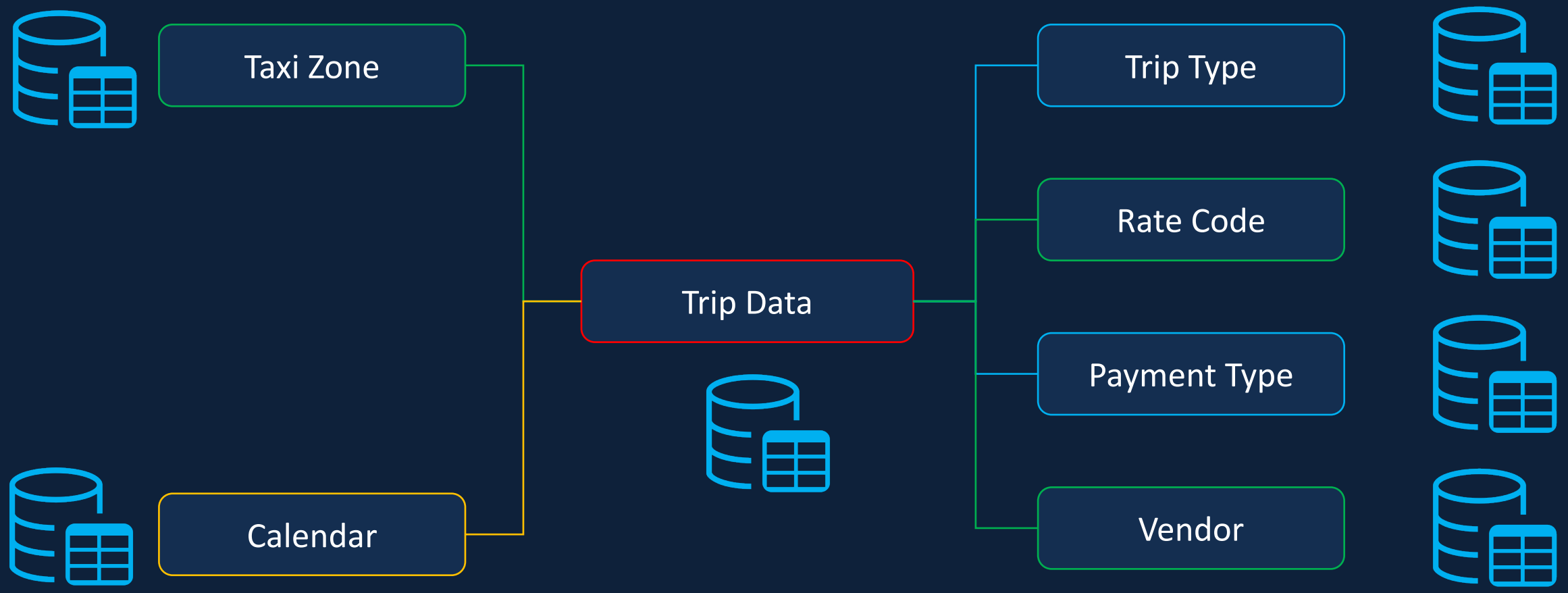
Trip Type — TSV

Rate Code — JSON

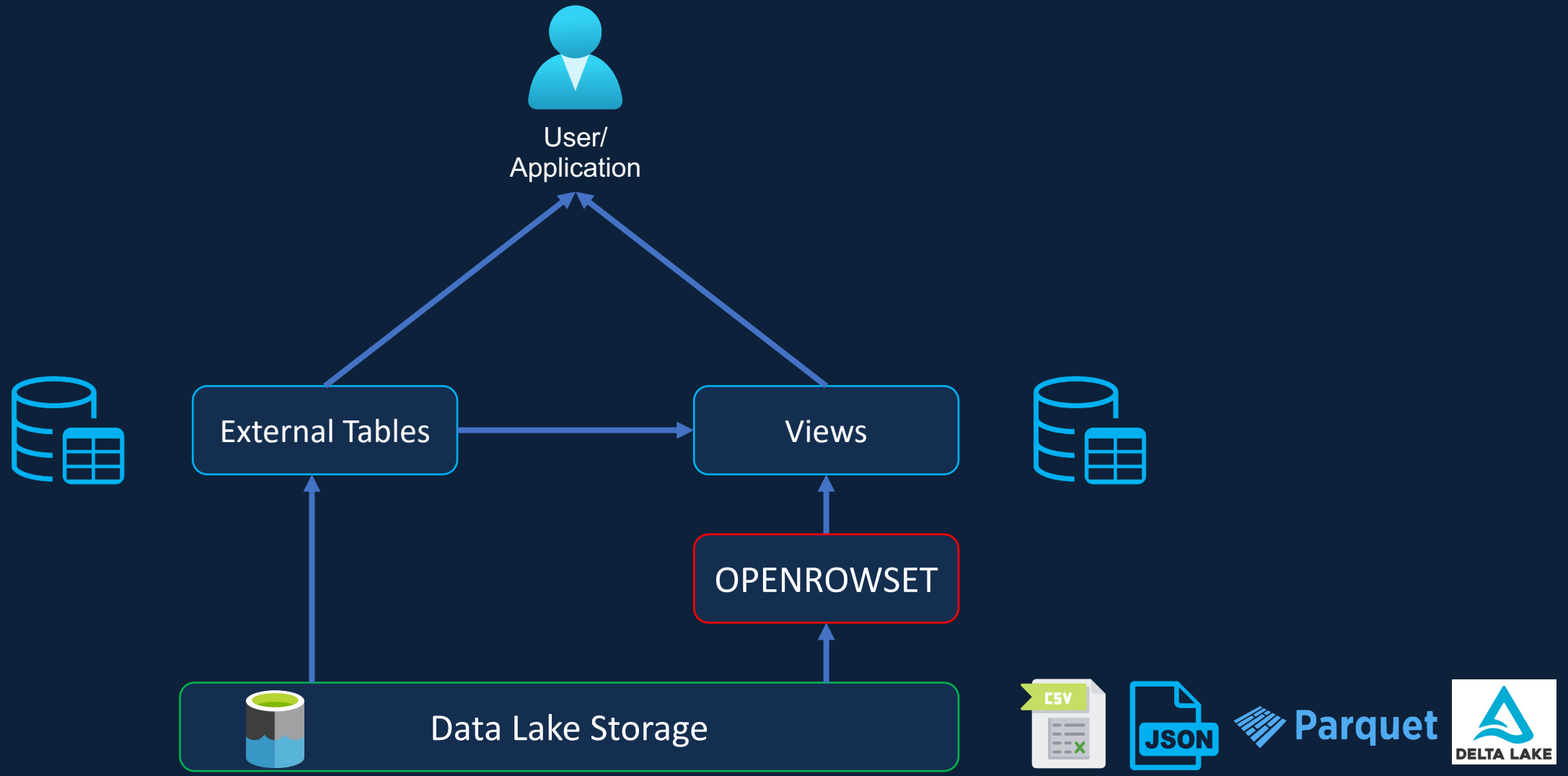
Payment Type — JSON

Vendor — CSV Quoted

NYC Taxi Data Files Overview



Database Objects - Types



External Table

Create External Table

Creates an external table on the data already present in the storage.

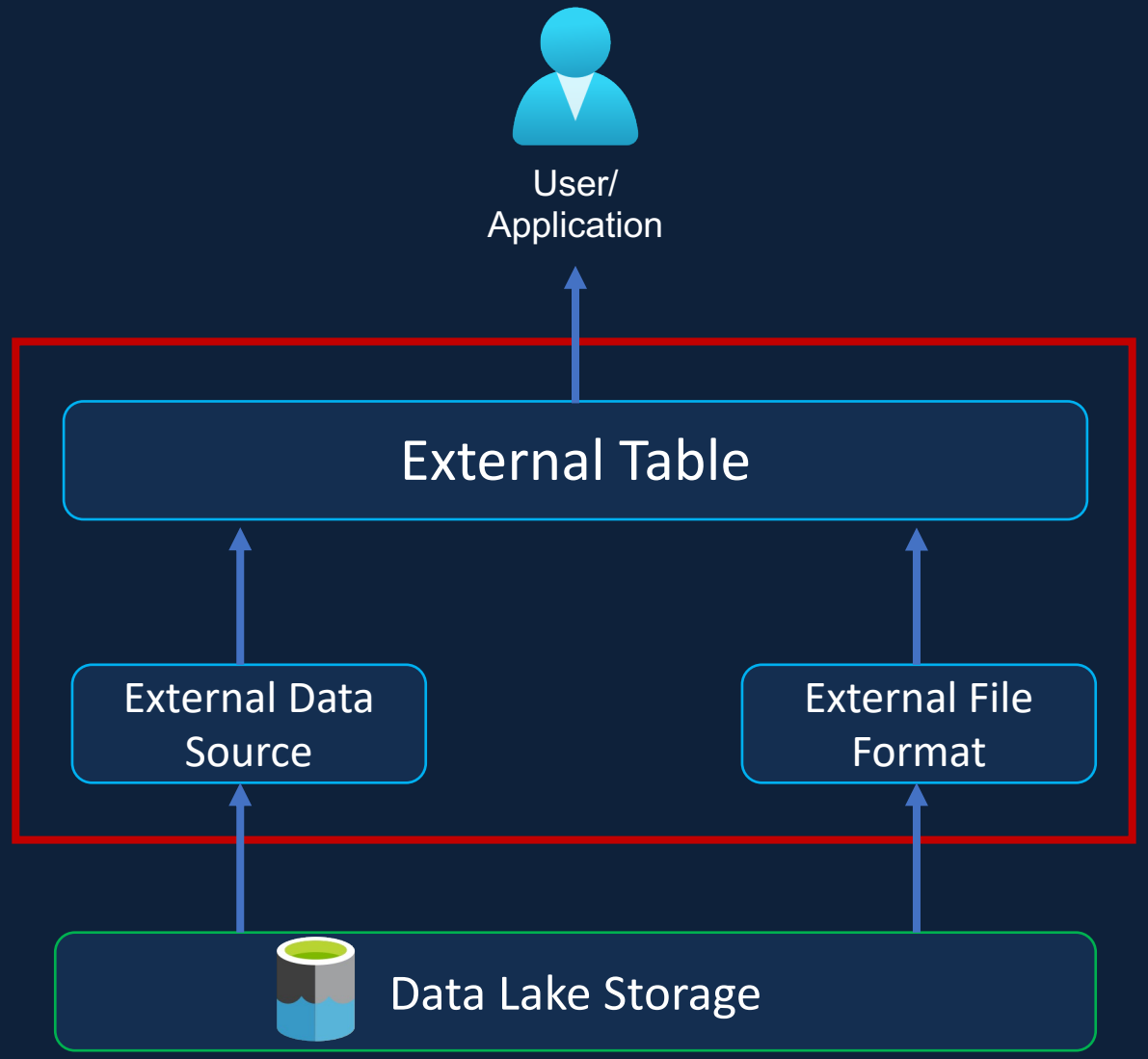
Metadata only change

Create External Table As Select (CETAS)

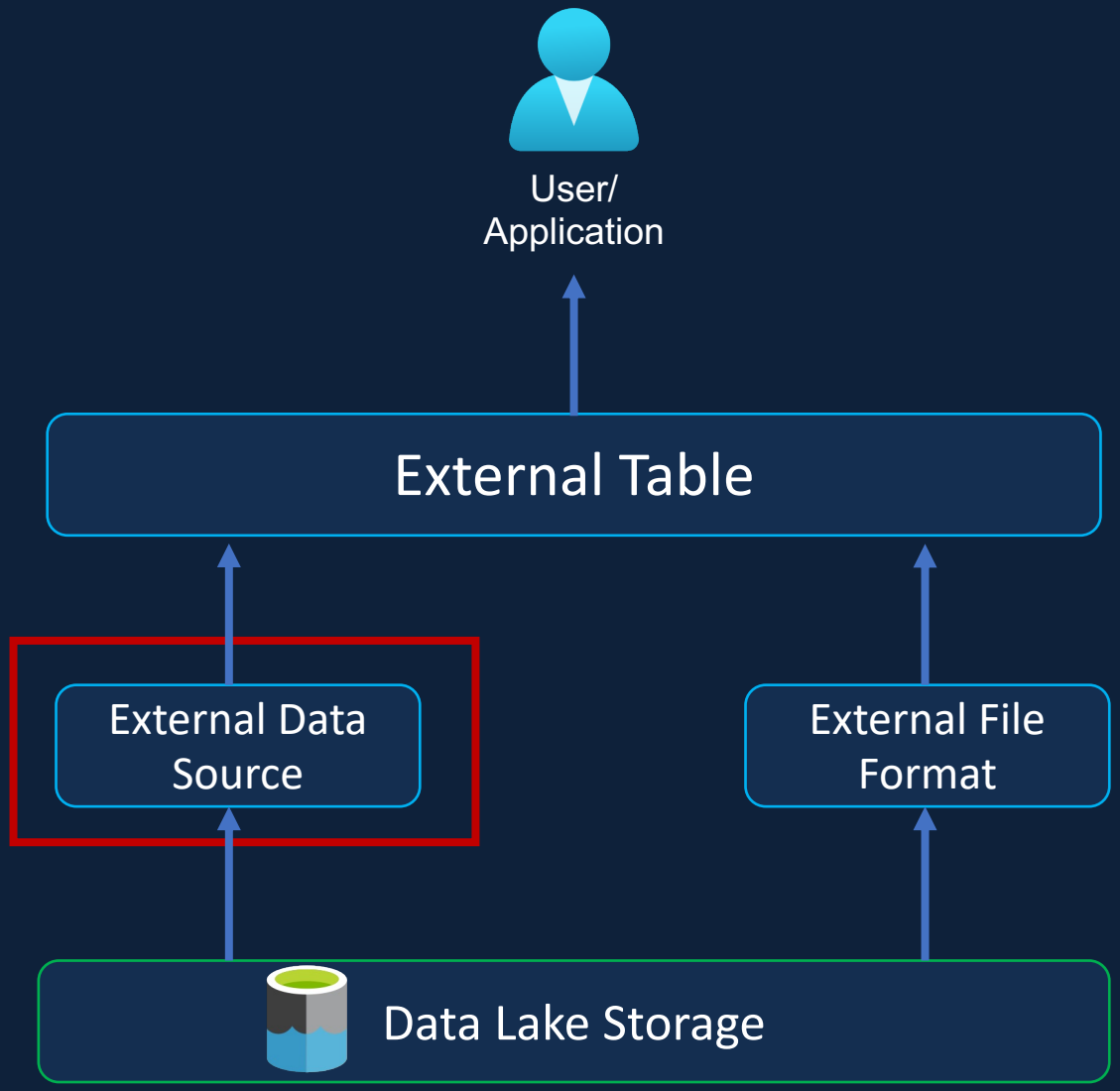
Selected data will copied to the location specified in the table definition.

Metadata change + Data copied

External Table



External Data Source

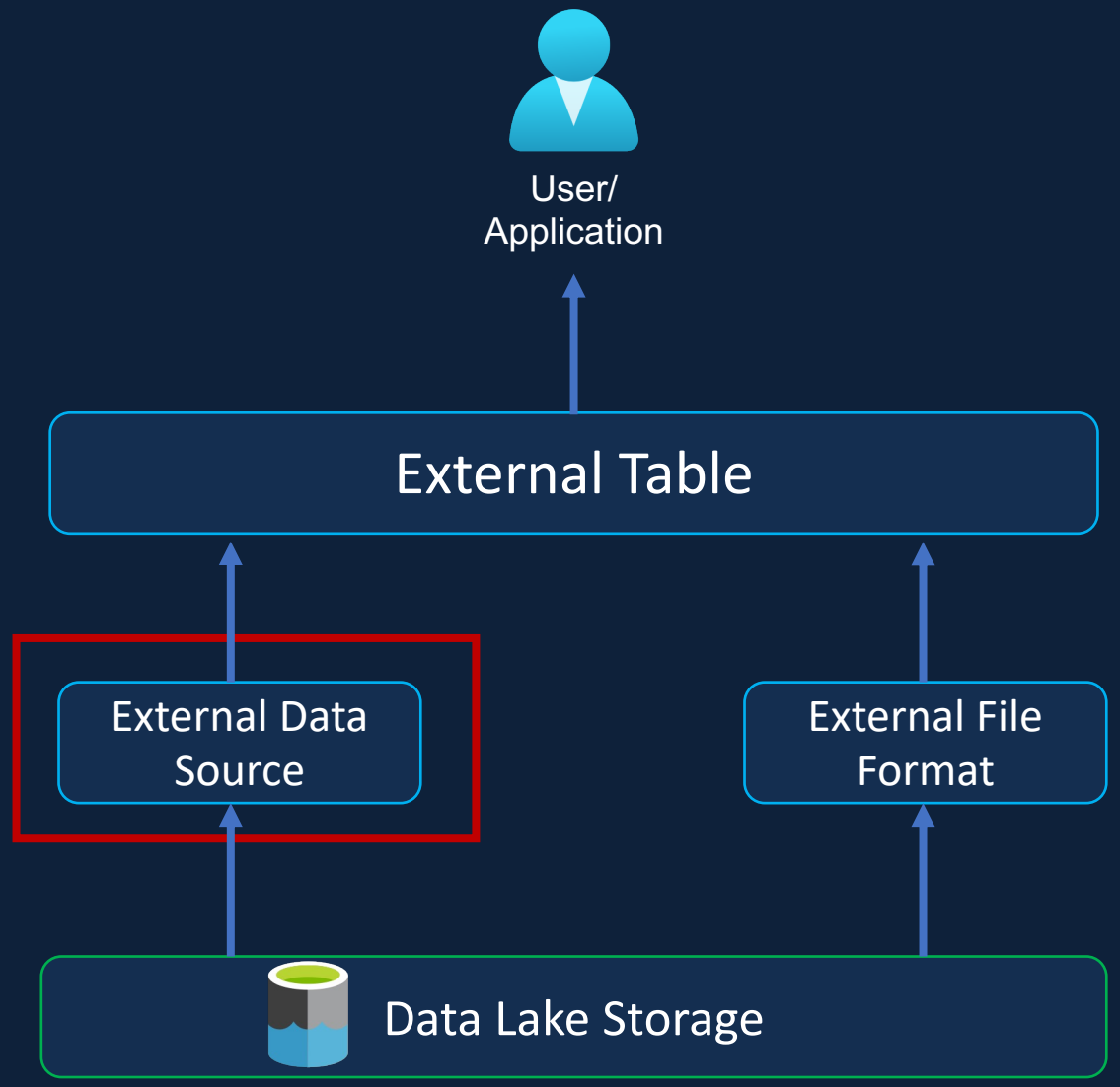


Location of the storage

Credential

Data source type

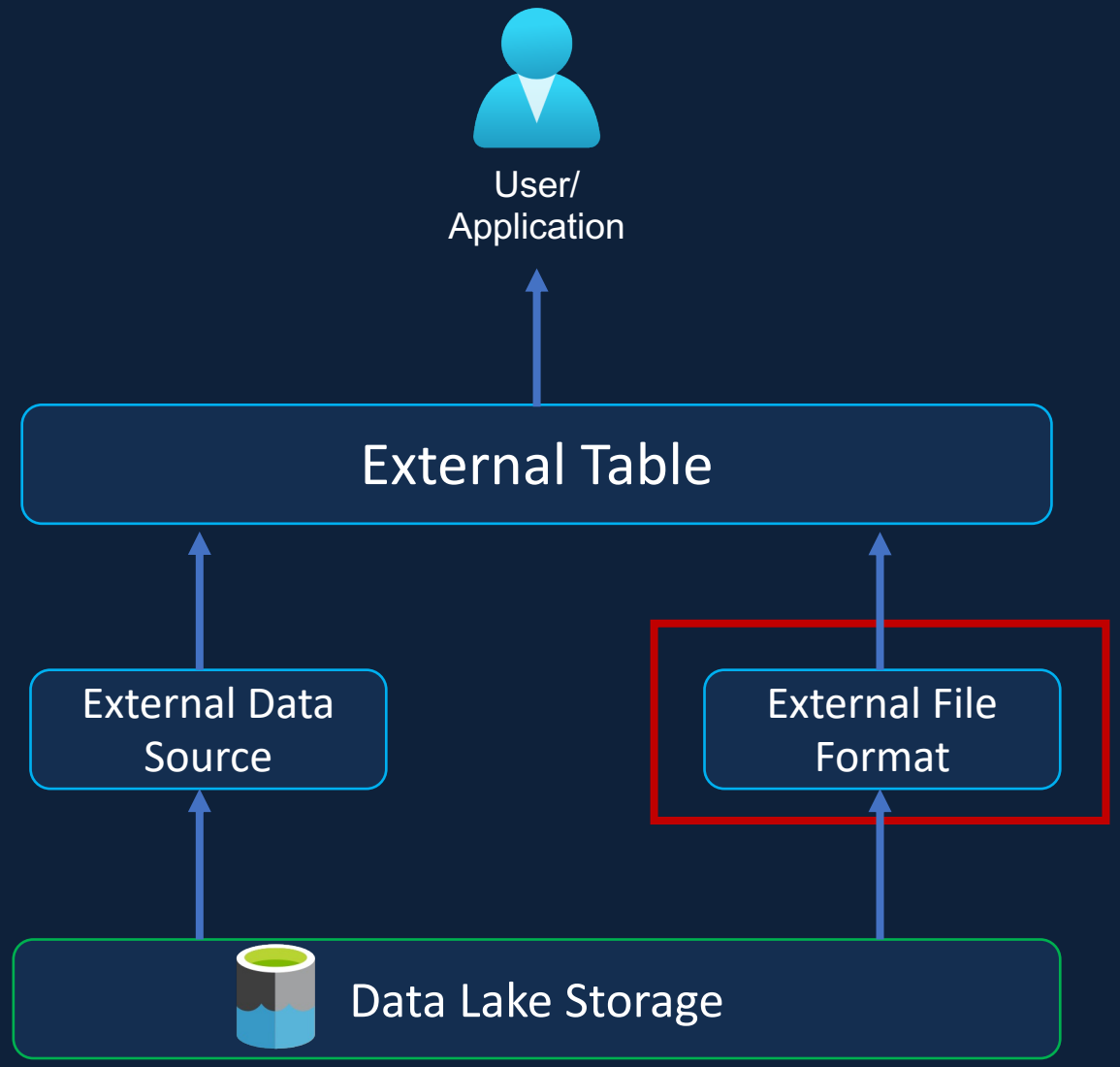
External Data Source




```
CREATE EXTERNAL DATA SOURCE <Data Source Name>
WITH
(
    LOCATION = <folder path URI> ,
    CREDENTIAL = <Credential Name> ,
    TYPE = {HADOOP}
) ;
```

```
CREATE EXTERNAL DATA SOURCE nyc_taxi_data
WITH
(
    LOCATION = 'abfss://nyc-taxi-
    data@synapsecoursedl.dfs.core.windows.net',
    CREDENTIAL = nyc_taxi_data_cred
) ;
```

External File Format

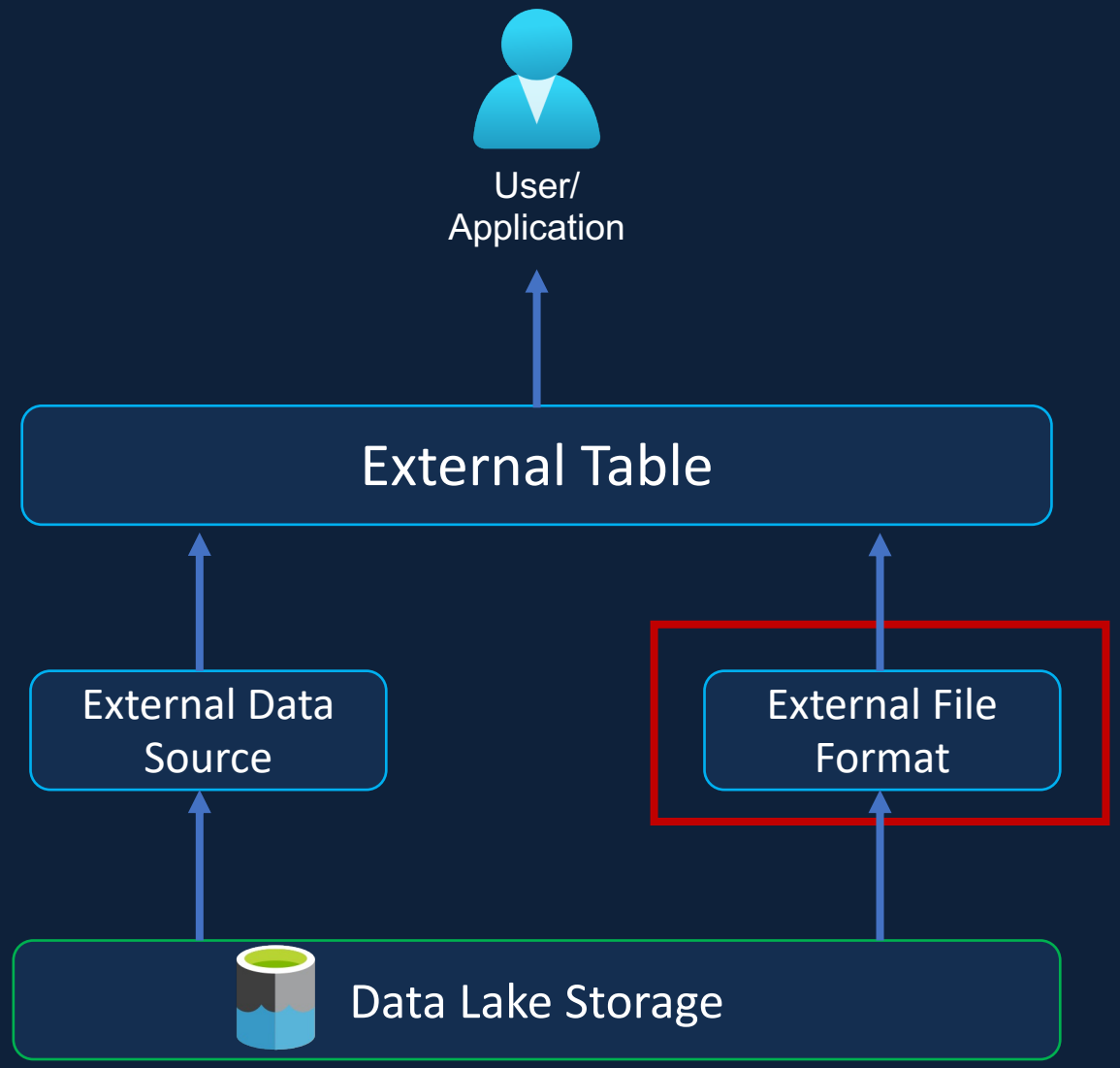


Format Type

Data Compression

Format Options

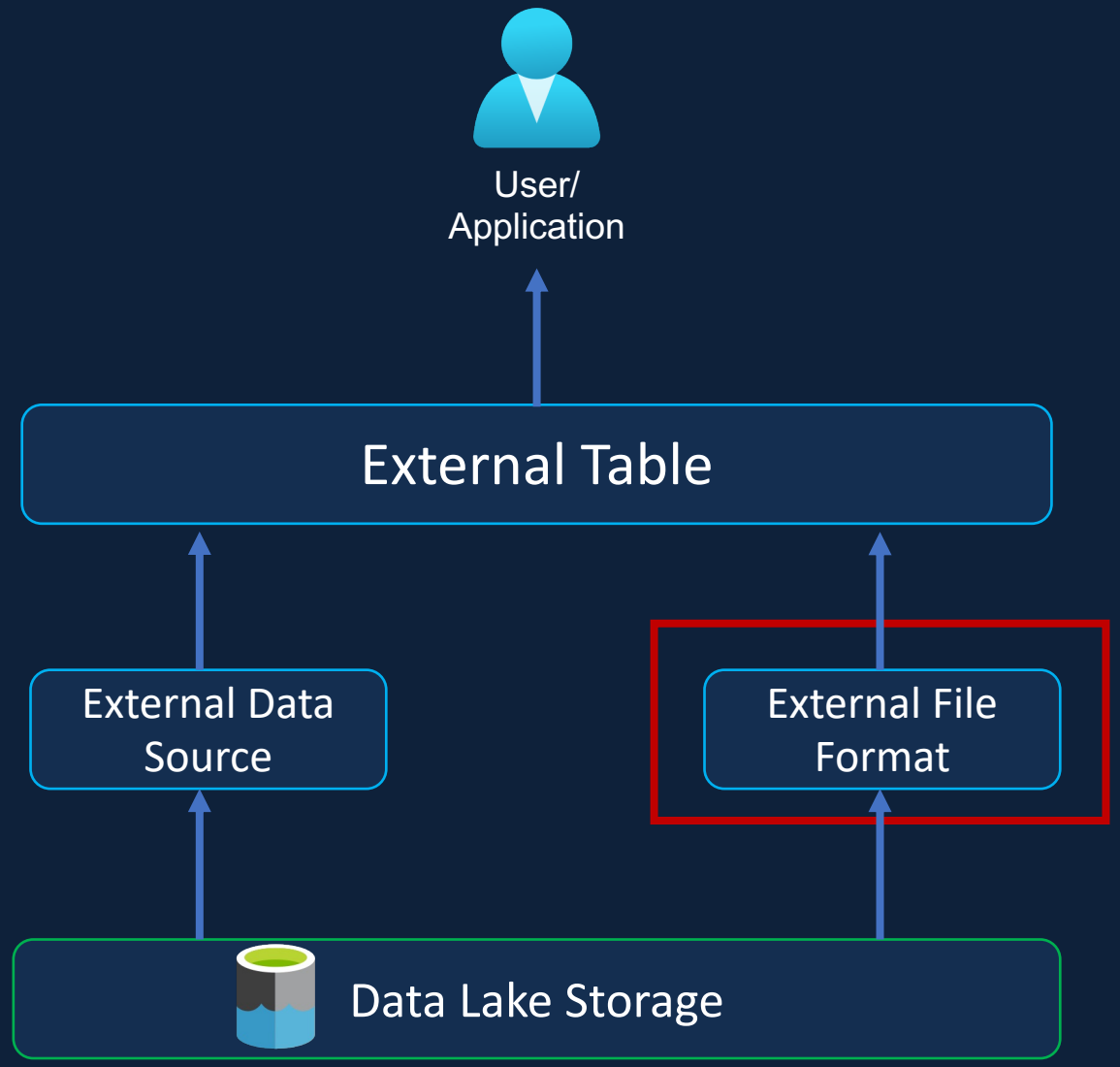
External File Format




```
CREATE EXTERNAL FILE FORMAT <file format name>
WITH
(
      FORMAT_TYPE = {DELIMITEDTEXT | PARQUET| DELTA}
      [ , DATA_COMPRESSION =
      'org.apache.hadoop.io.compress.GzipCodec'
      | 'org.apache.hadoop.io.compress.SnappyCodec']
      [ , FORMAT_OPTIONS ( <format_options> [ ,...n ] ) ]
);
```

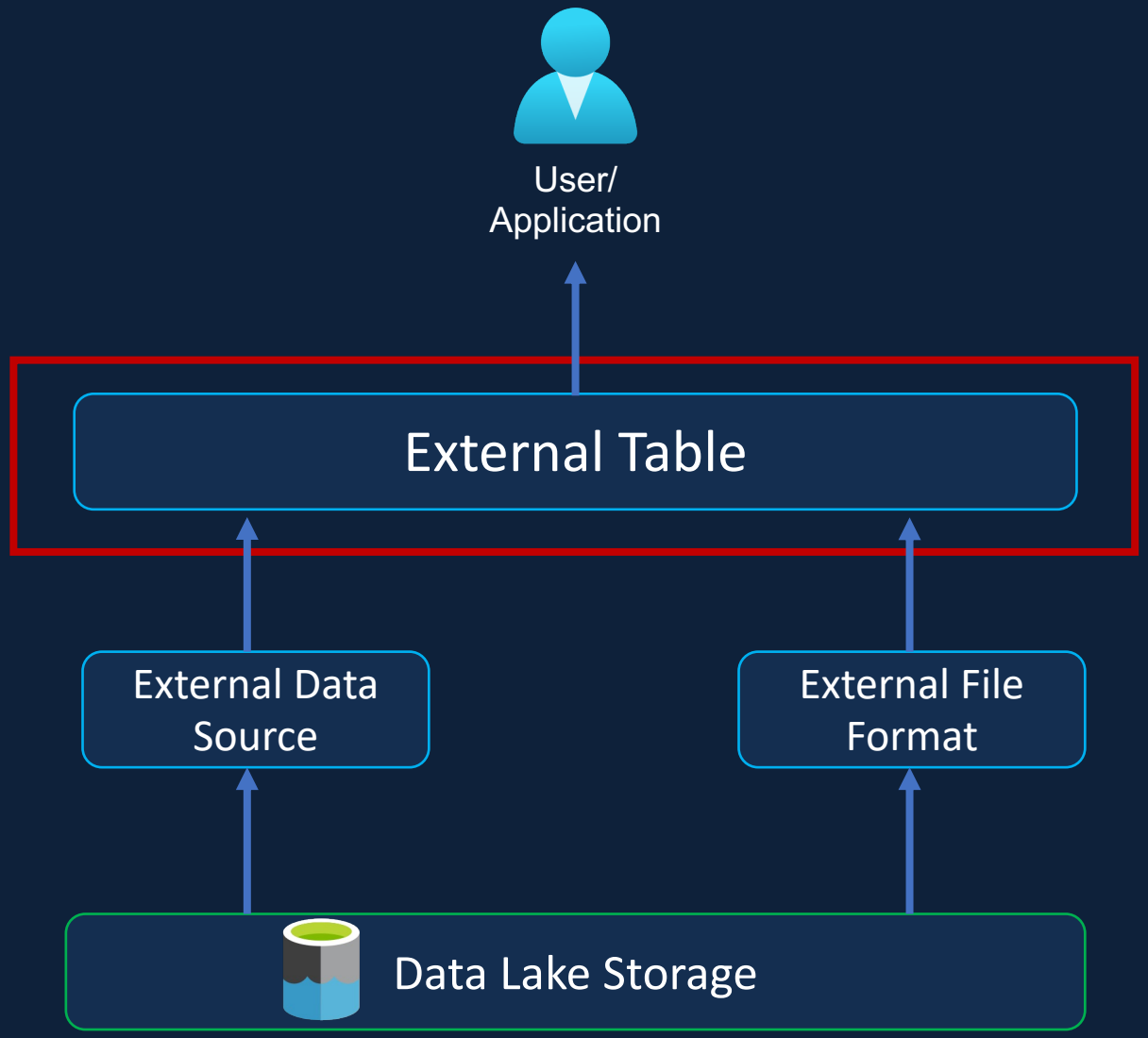
```
<format_options> ::=
{
      FIELD_TERMINATOR = field_terminator
      | STRING_DELIMITER = string_delimiter
      | First_Row = integer
      | USE_TYPE_DEFAULT = { TRUE | FALSE }
      | Encoding = {'UTF8' | 'UTF16'}
      | PARSER_VERSION = {'parser_version'}
}
```

External File Format (Example)




```
CREATE EXTERNAL FILE FORMAT csv_file_format
WITH
(
      FORMAT_TYPE = DELIMITEDTEXT,
      FORMAT_OPTIONS
                        (
                              FIELD_TERMINATOR = ',',
                              STRING_DELIMITER = '"',
                              FIRST_ROW = 2,
                              USE_TYPE_DEFAULT = False,
                              ENCODING = 'UTF8',
                              PARSER_VERSION = '2.0’
                        )
);
```

External Table



External Data Source Name

File Locations name

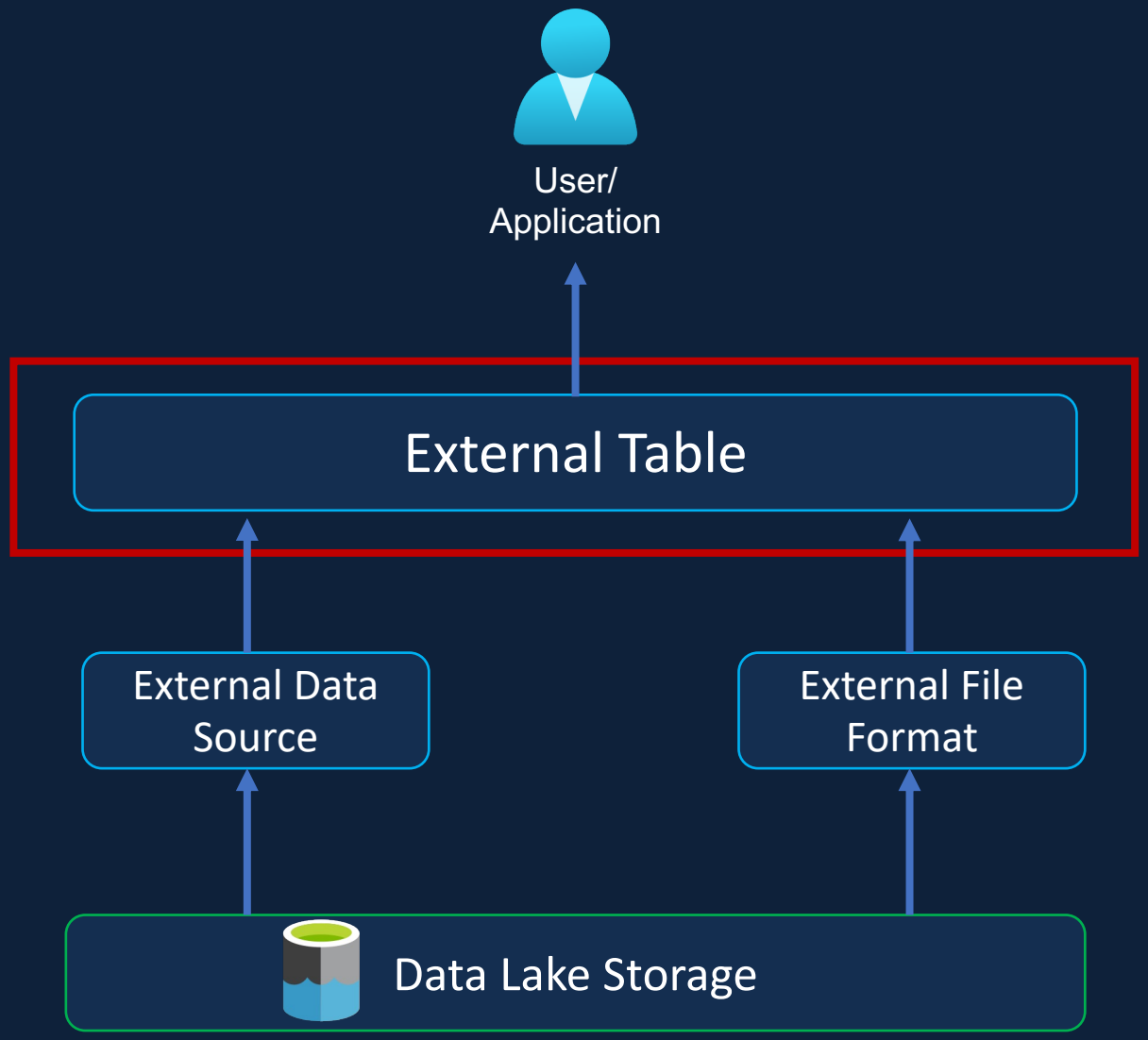
External File Format Name

Column Name/ Data Type

Read Options

Reject Options

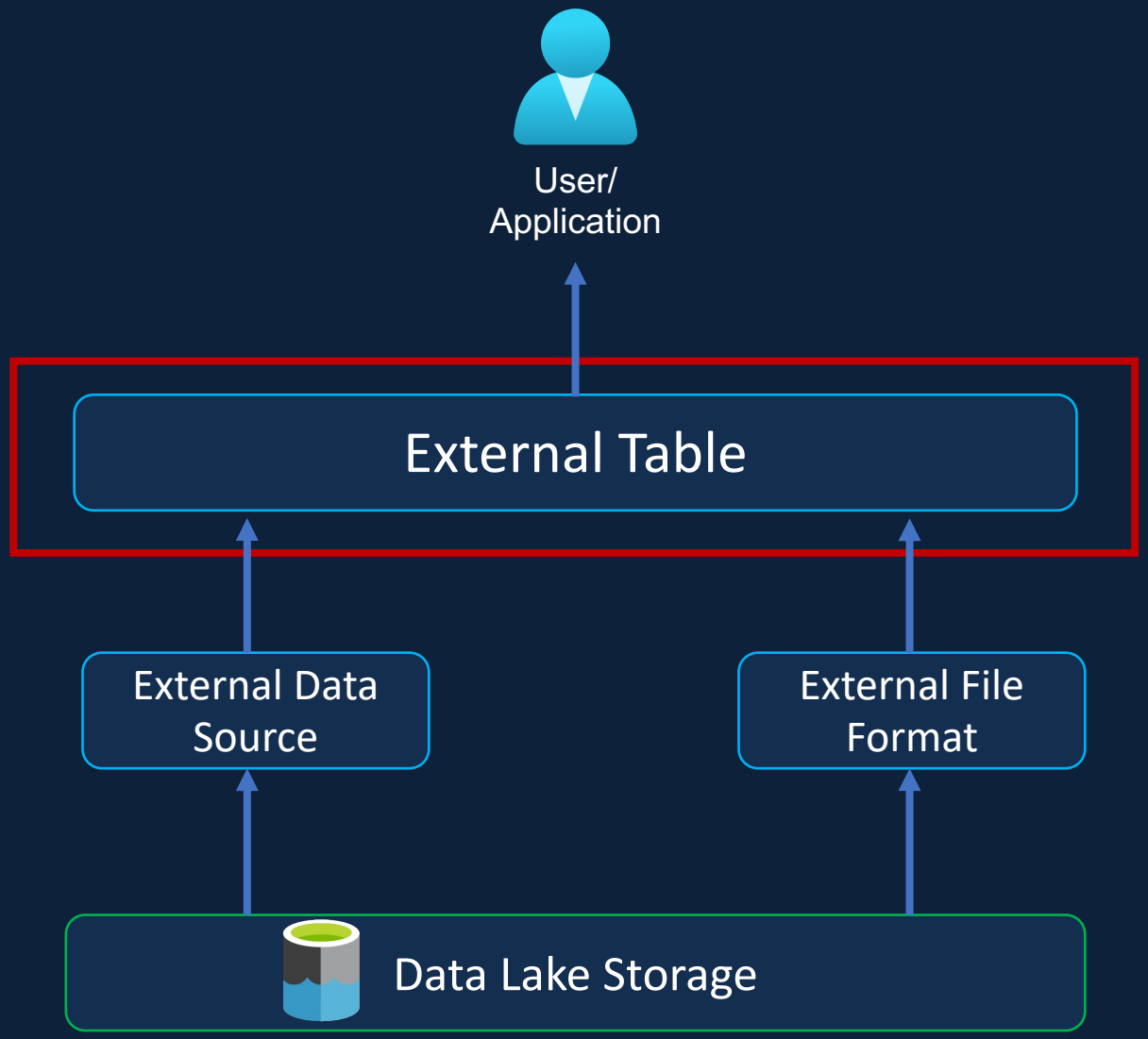
External Table




```
CREATE EXTERNAL TABLE
<[database_name].[schema_name].table_name>
( <column name>  <data type> )
WITH (
        LOCATION = ‘<folder_or_filepath>',
        DATA_SOURCE = <external_data_source_name>,
        FILE_FORMAT = <external_file_format_name>
        [, TABLE_OPTIONS =
        N'{"READ_OPTIONS":["ALLOW_INCONSISTENT_READS"]}
        ' ]
        [, <reject_options>]
)
```

```
<reject_options> ::=
{
| REJECT_TYPE = value,
| REJECT_VALUE = reject_value,
| REJECT_SAMPLE_VALUE = reject_sample_value,
| REJECTED_ROW_LOCATION = '/REJECT_Directory'
}
```

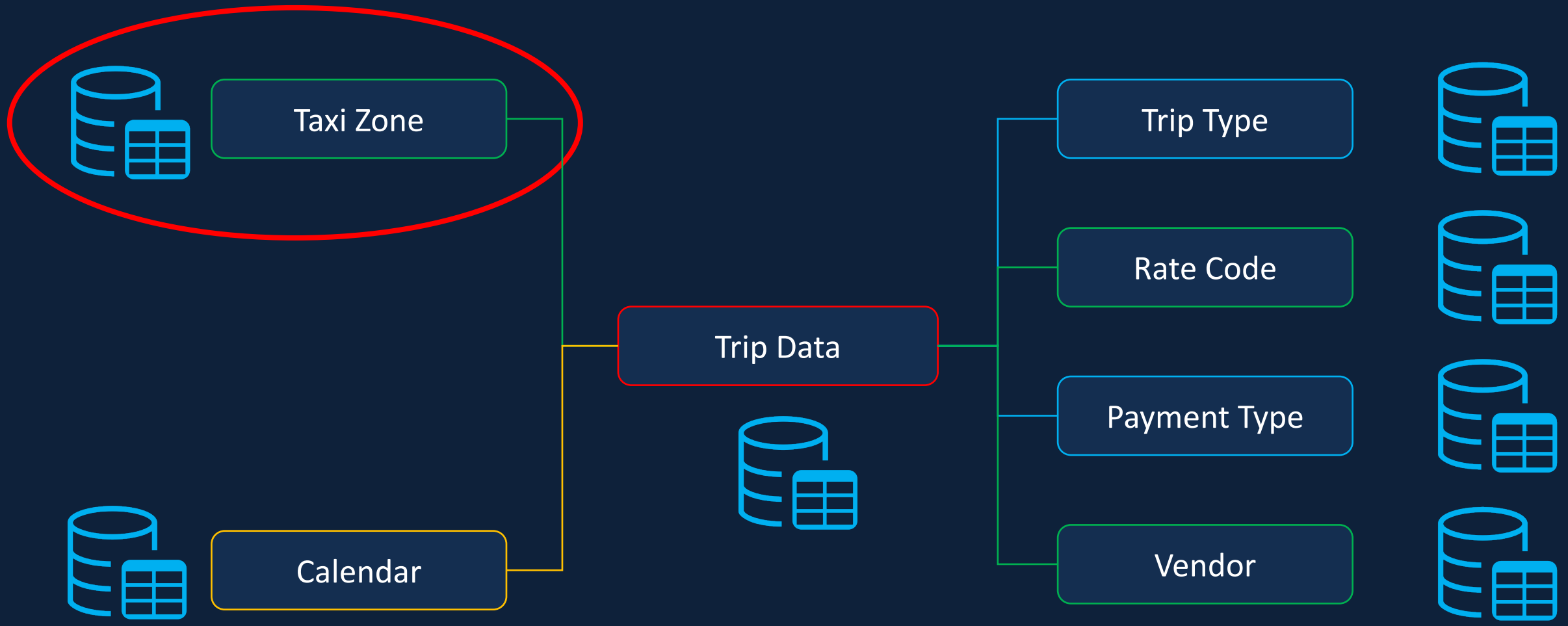
External Table



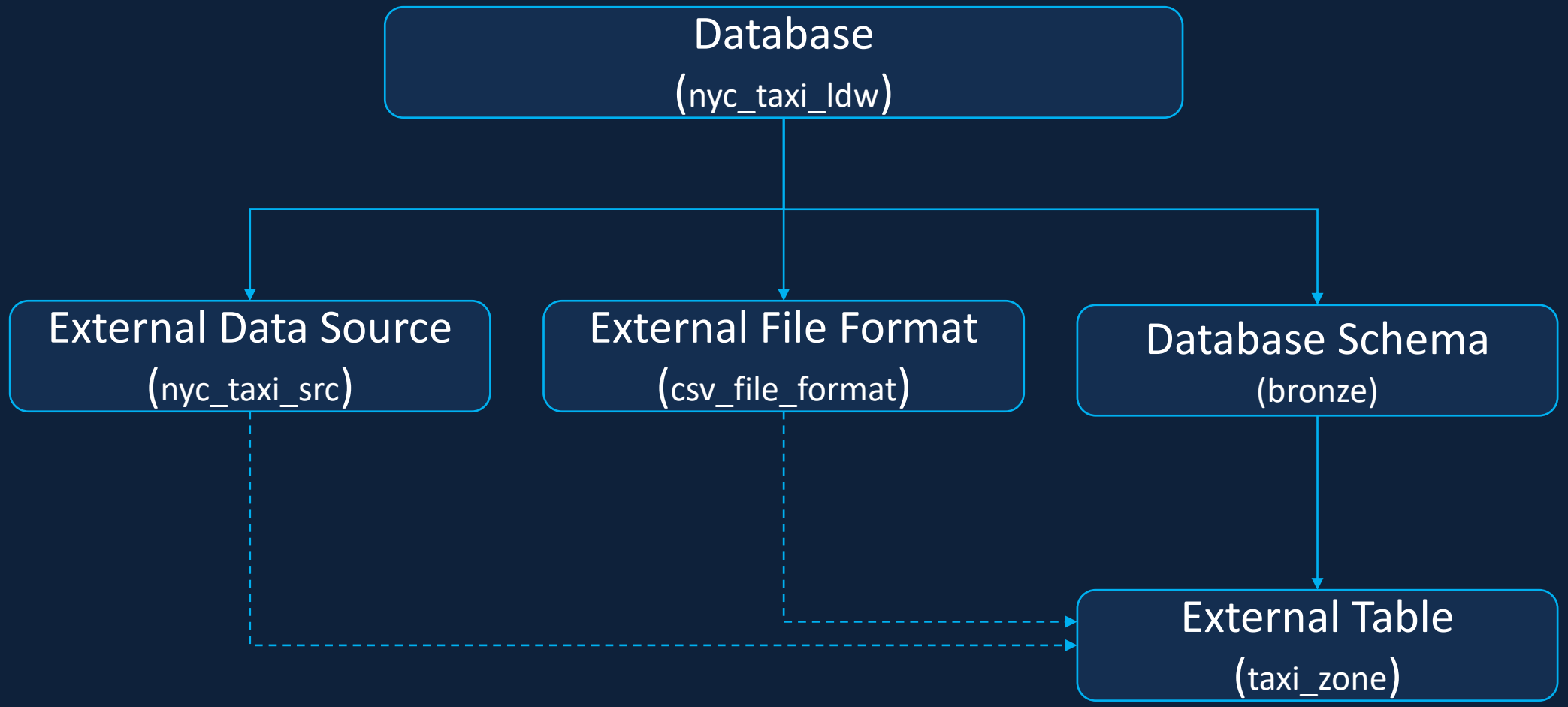

```
CREATE EXTERNAL TABLE demo.ldw.taxi_zone
(
        location_id SMALLINT,
        borough VARCHAR(1),
        zone VARCHAR(50) ,
        service_zone VARCHAR(15)
)
WITH (
        LOCATION = 'raw/taxi_zone.csv',
        DATA_SOURCE = nyc_taxi_data,
        FILE_FORMAT = csv_file_format,
        REJECT_VALUE = 10,
        REJECTED_ROW_LOCATION = 'rejections/ldw/taxi_zone'
);
```

Create External Table Lab

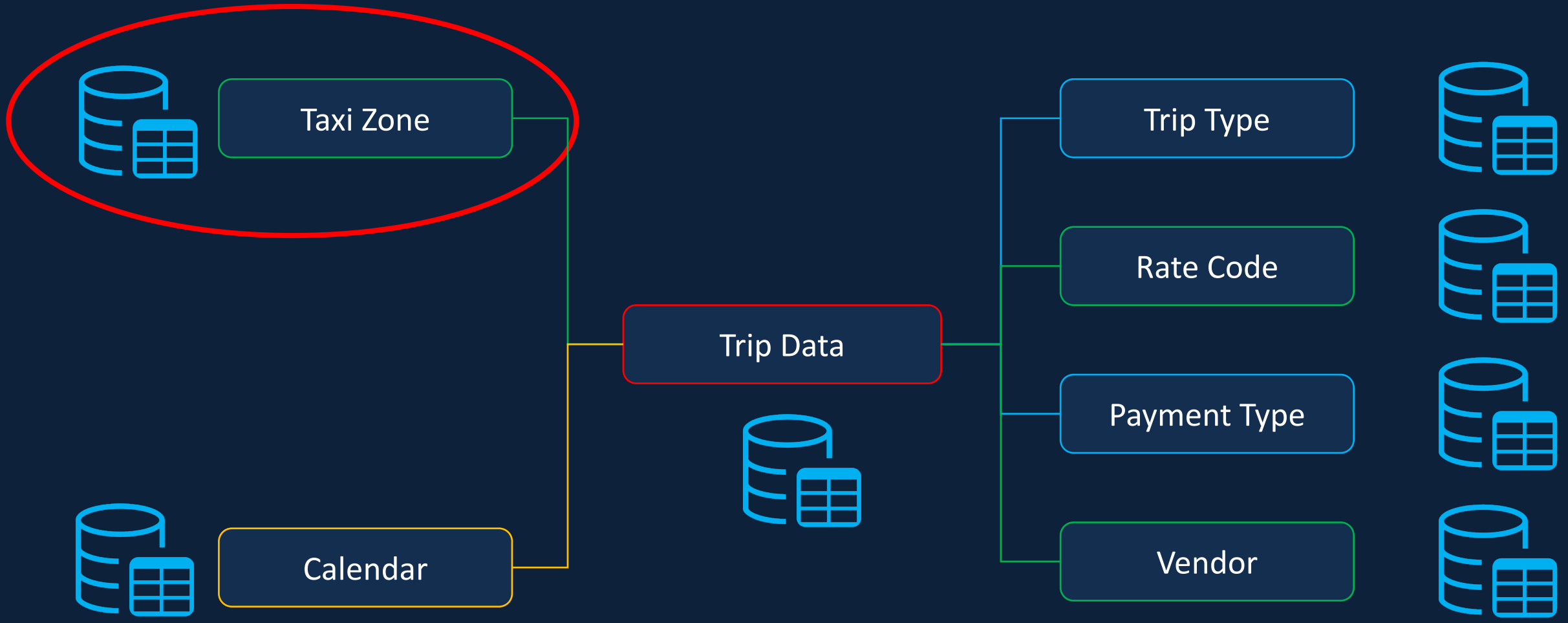
NYC Taxi Data Files Overview



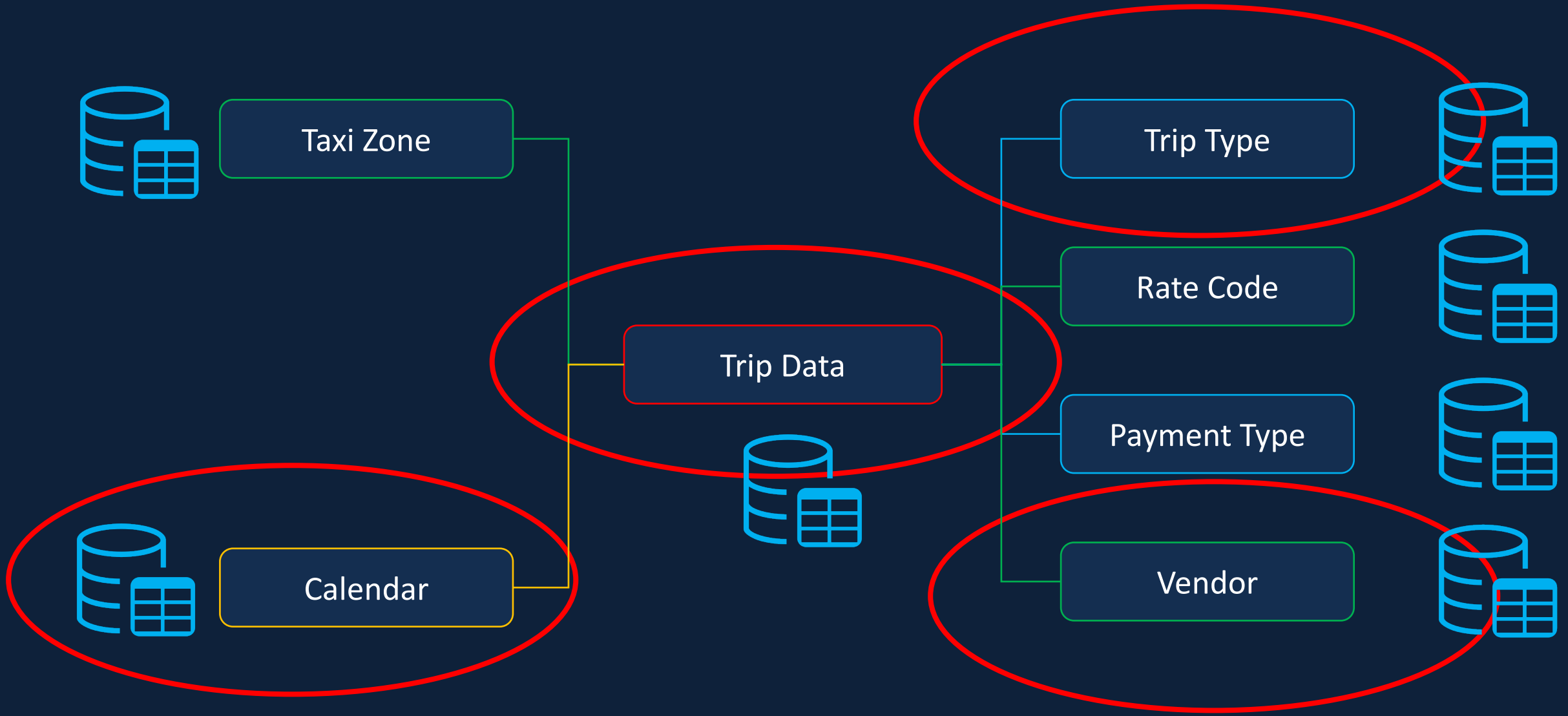
Create External Table



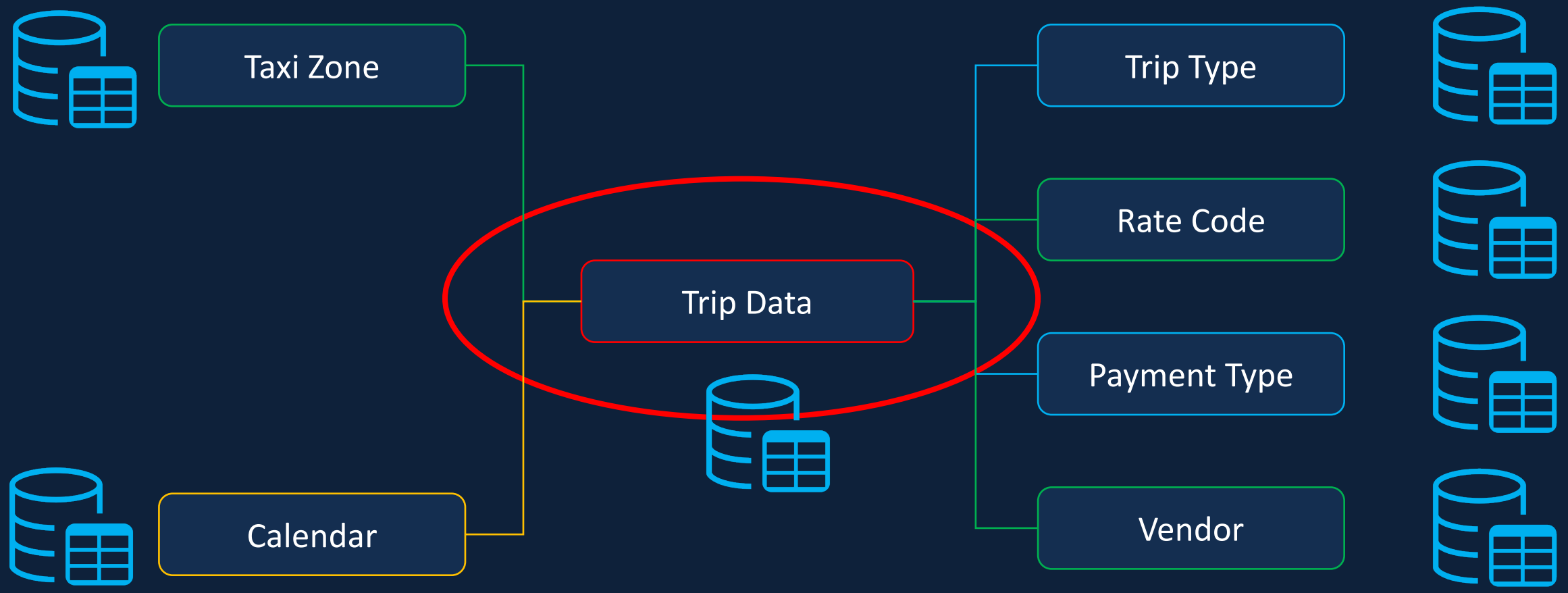
Create External Table – Delimited Text



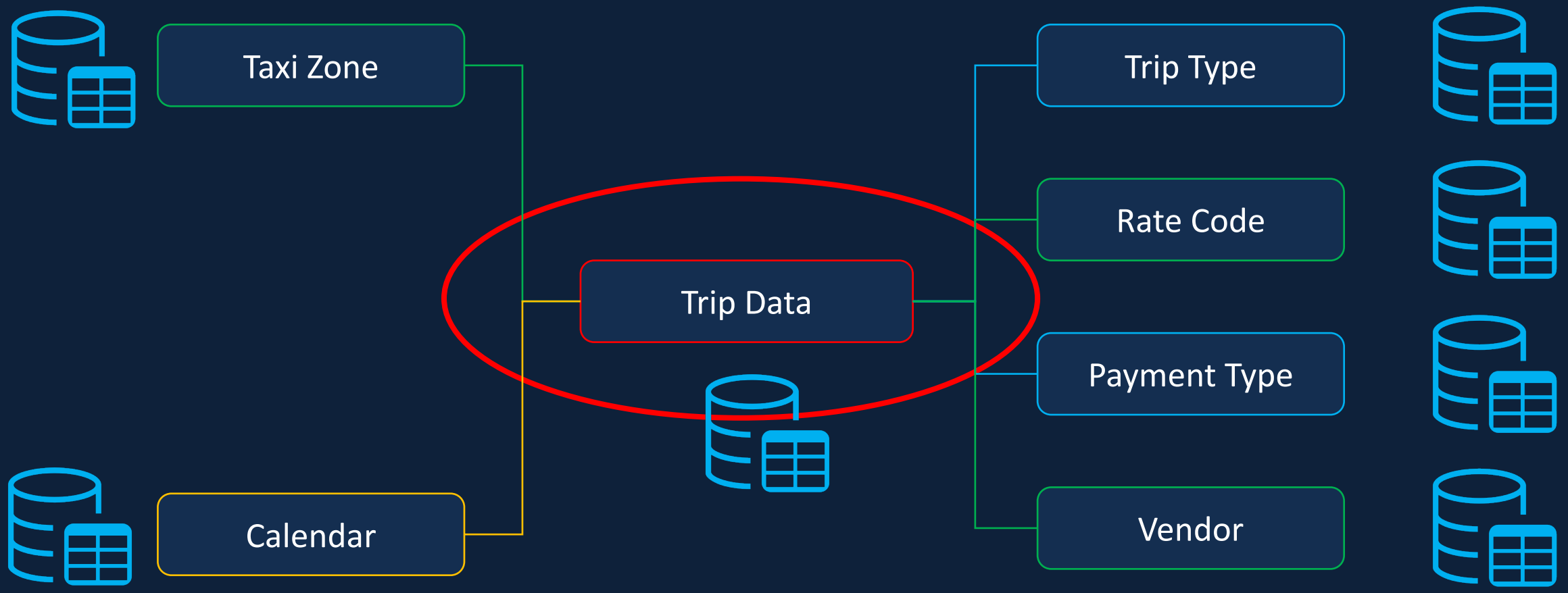
Create External Table – Delimited Text (Assignment)



Create External Table – Parquet



Create External Table – Delta (Assignment)

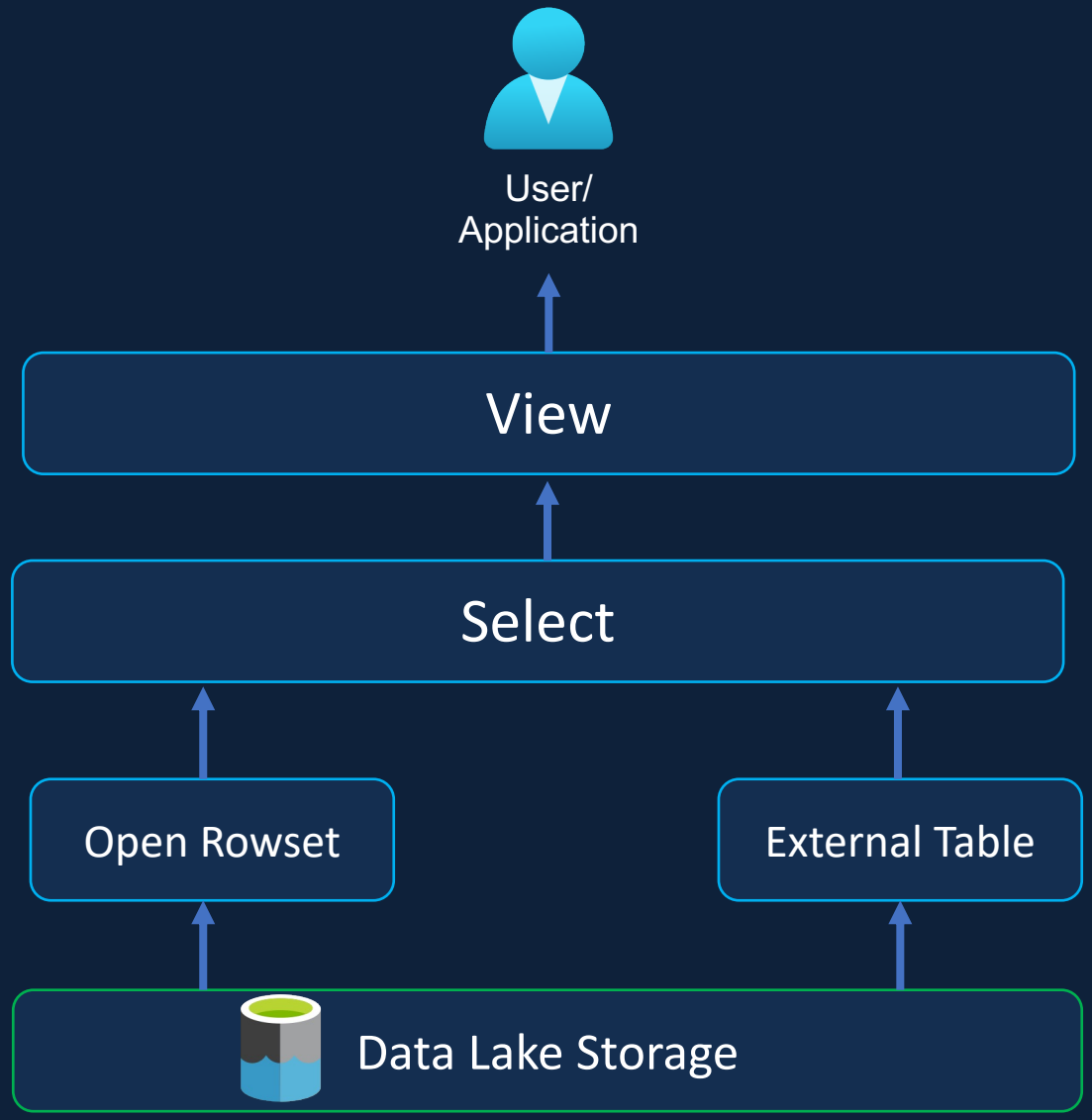


Views

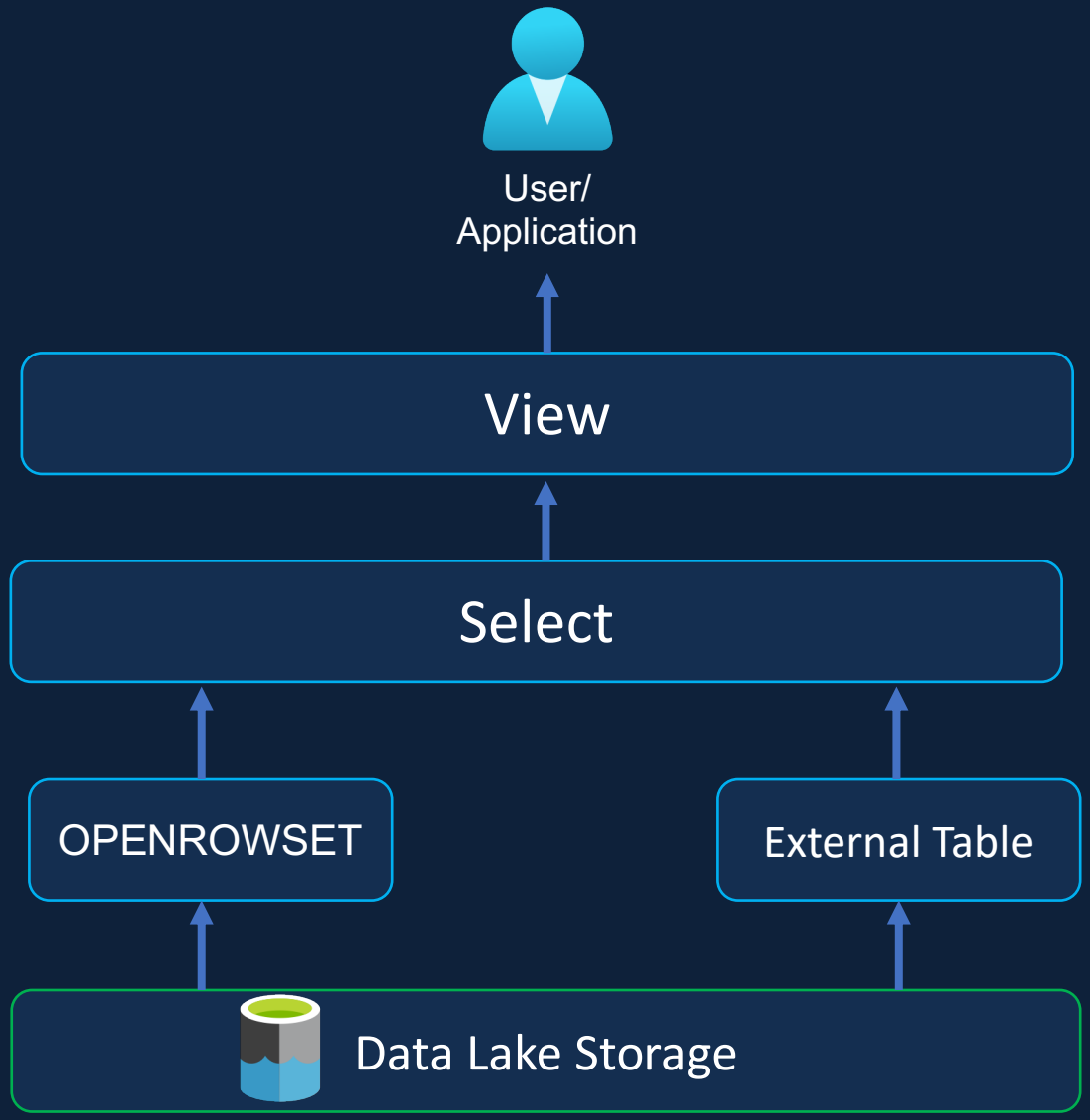
Virtual tables with rows and columns

Data in a view is defined by a SELECT statement

Views

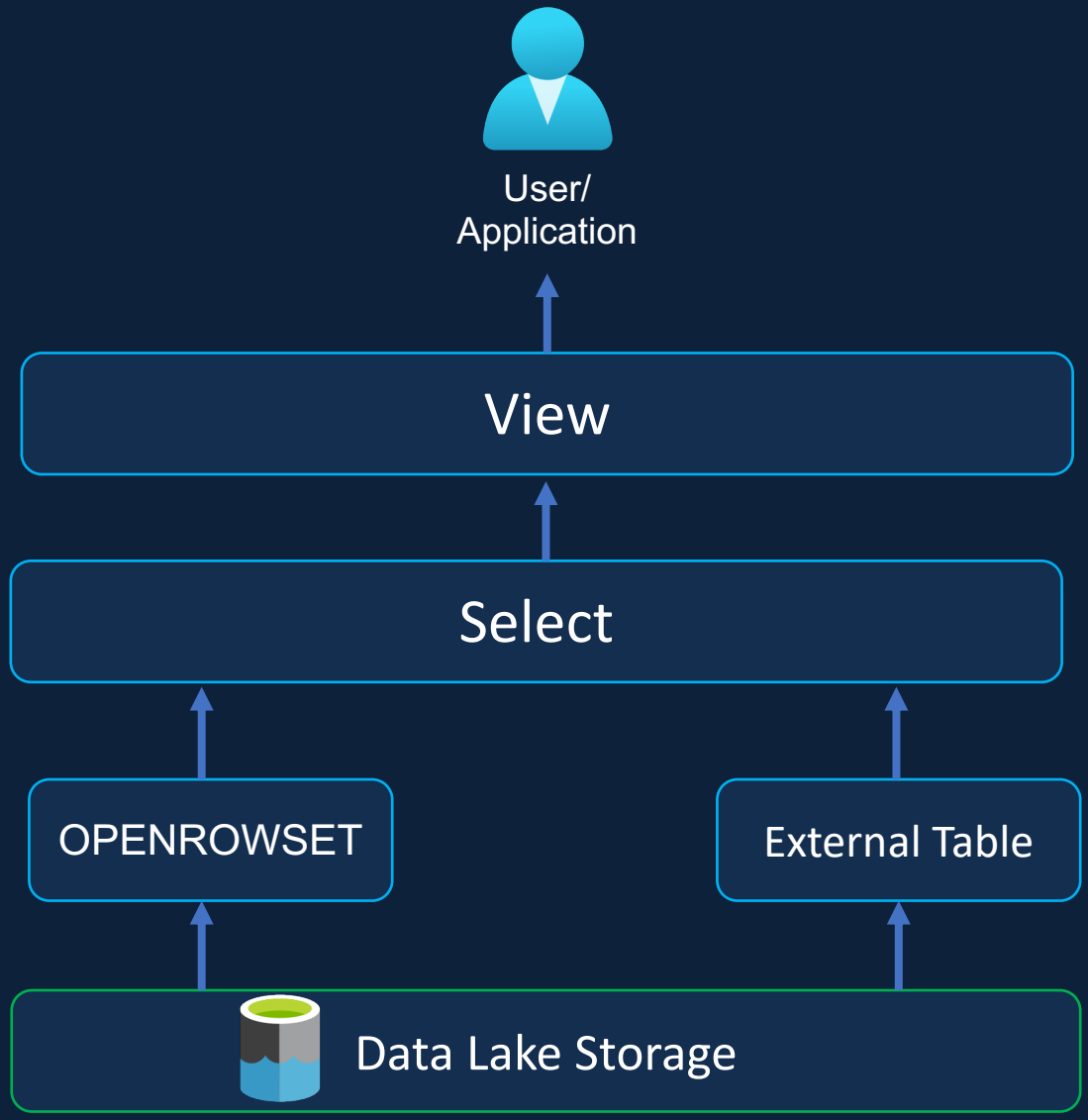

User/
Application
View
Select
Open Rowset
External Table
Data Lake Storage

Create View



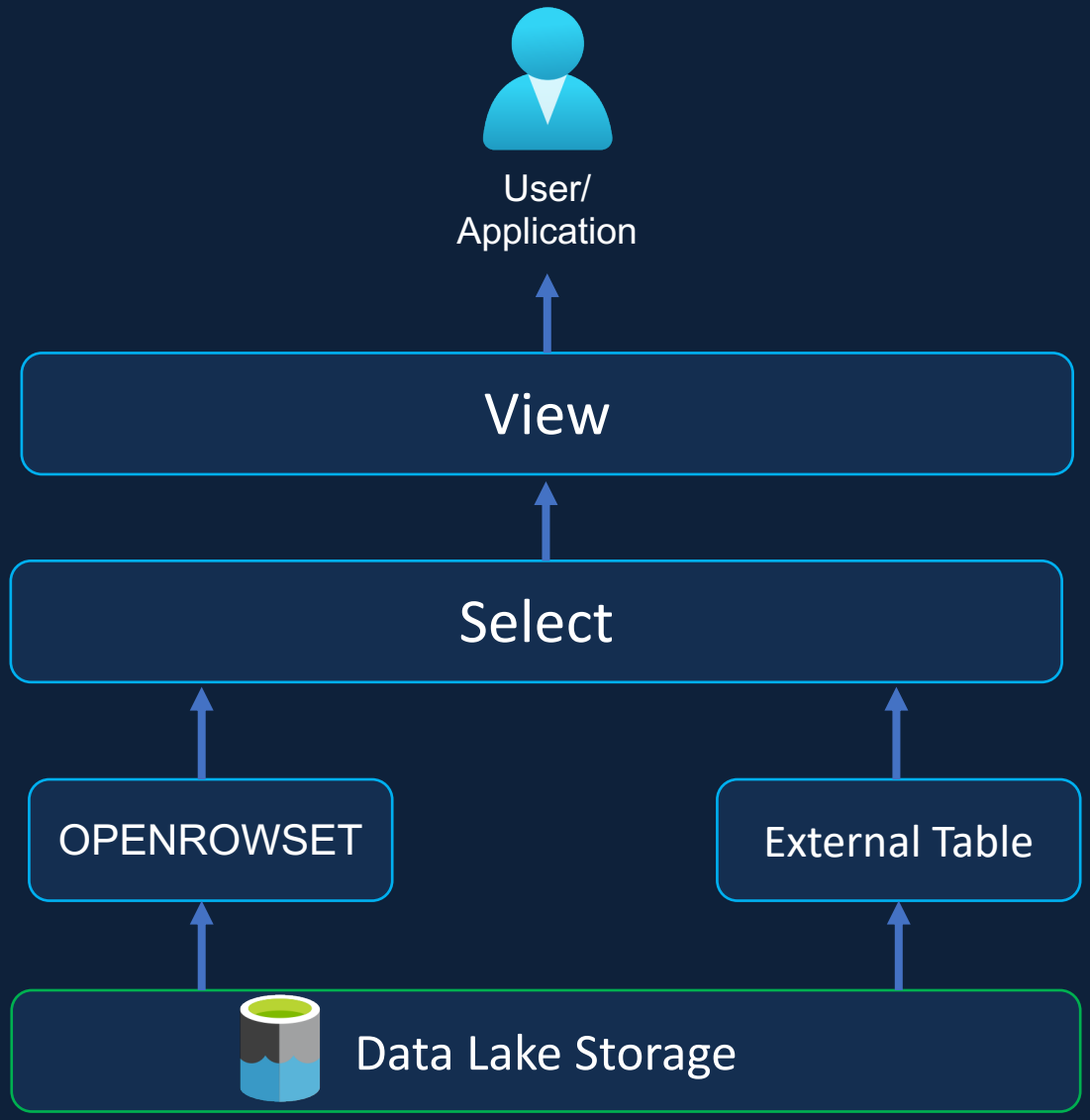

```
CREATE VIEW [ schema_name . ] view_name
                    [ ( column_name [ ,...n ] ) ]
AS
<select_statement> [;]
```

Create View - OPENROWSET




```
CREATE VIEW bronze.vw_vendor
AS
SELECT *
FROM OPENROWSET(
                    BULK 'vendor.csv',
                    DATA_SOURCE = 'nyc_taxi_data_raw',
                    FORMAT = 'CSV',
                    PARSER_VERSION = '2.0',
                    HEADER_ROW = TRUE
                ) AS vendor;
```

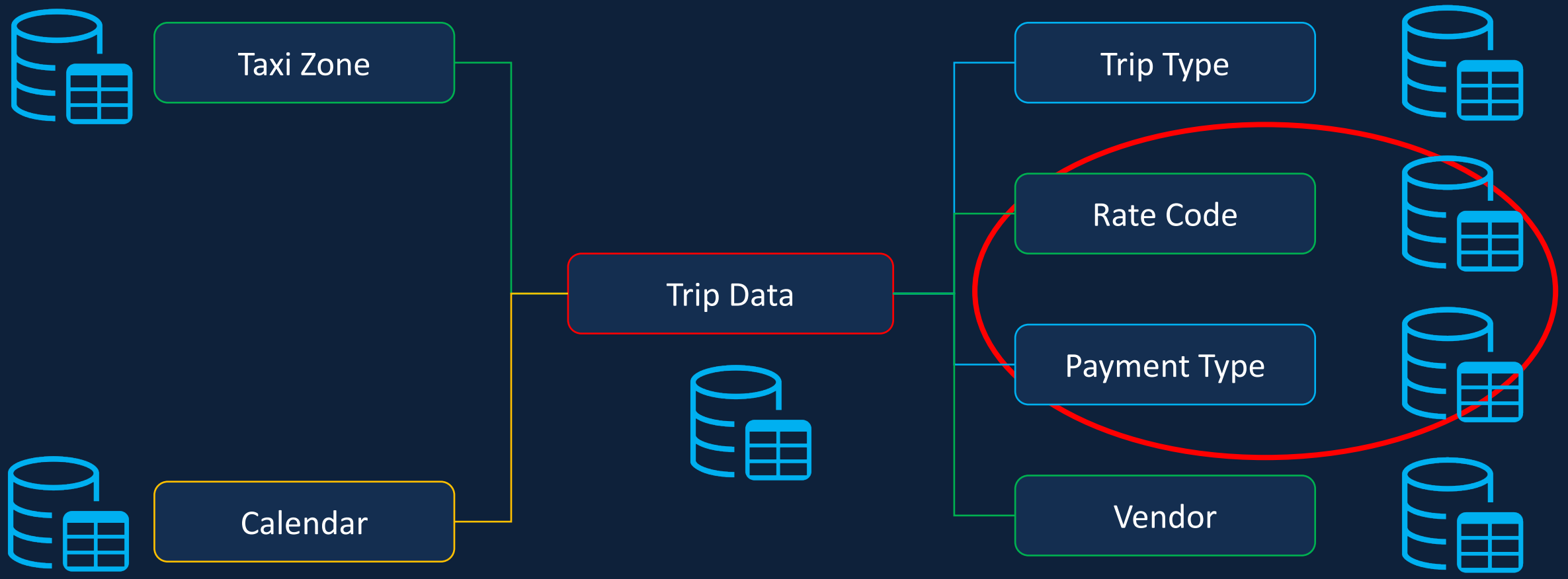
Create View – External Table



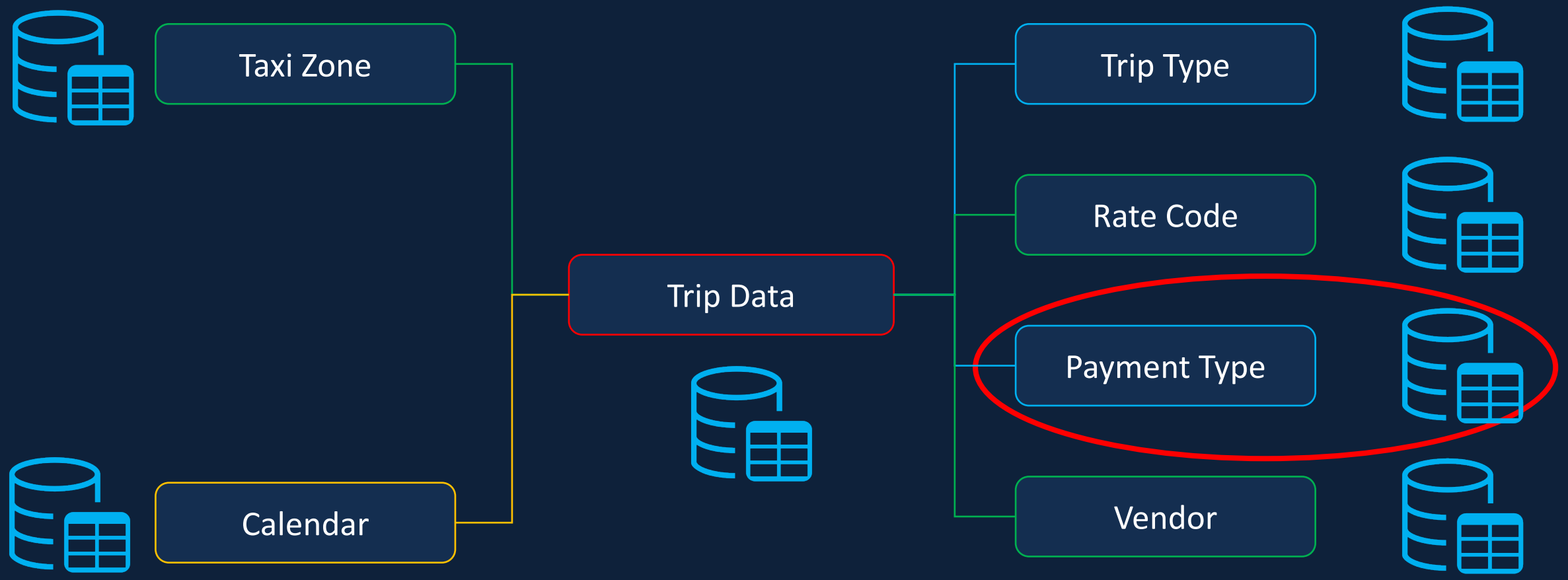

```
CREATE VIEW bronze.vw_taxi_zone_brooklyn
AS
SELECT location_id, zone, service_zone
FROM bronze.taxi_zone
WHERE borough = 'Brooklyn';
```

Create View Lab

NYC Taxi Data Files Overview



Create View – Assignment



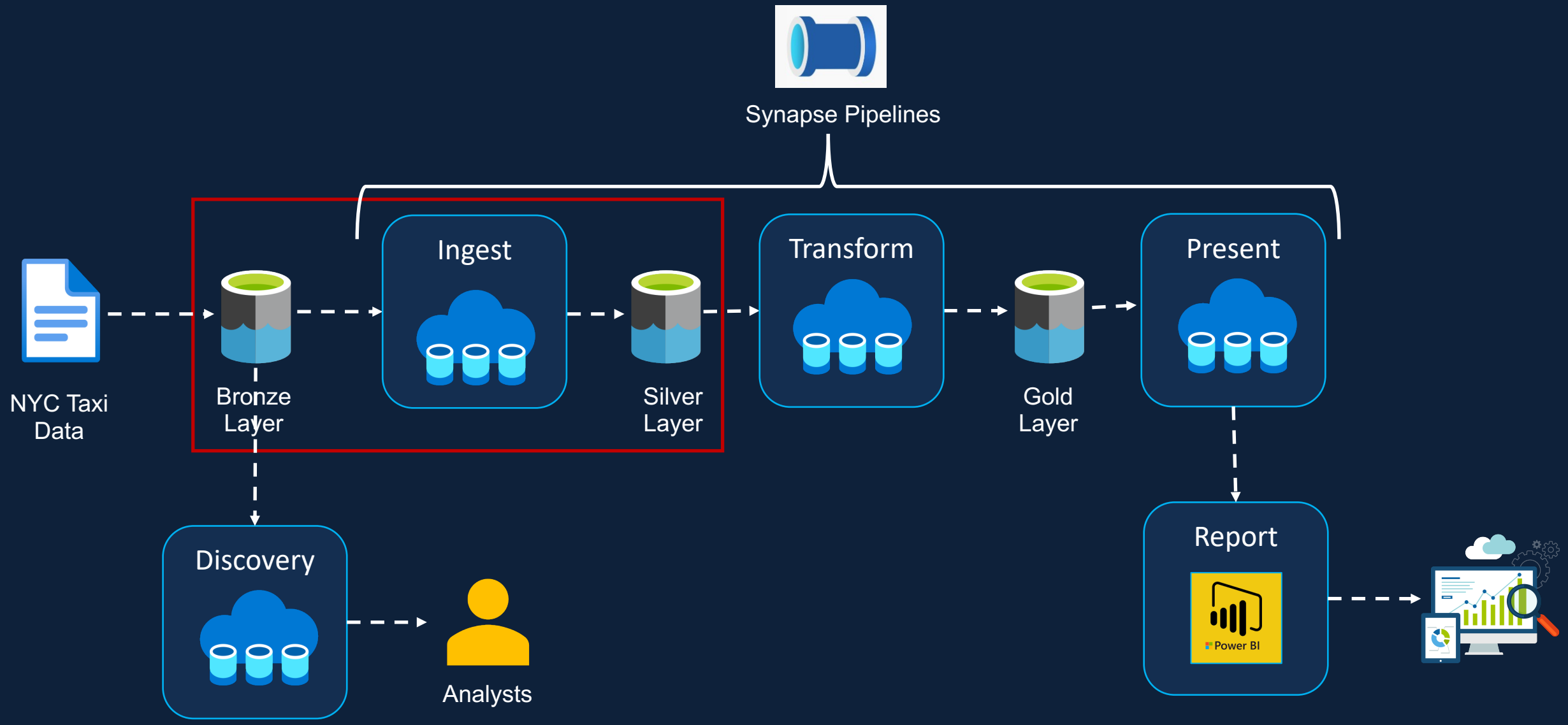
Partition Pruning

External Tables cannot prune partitions.

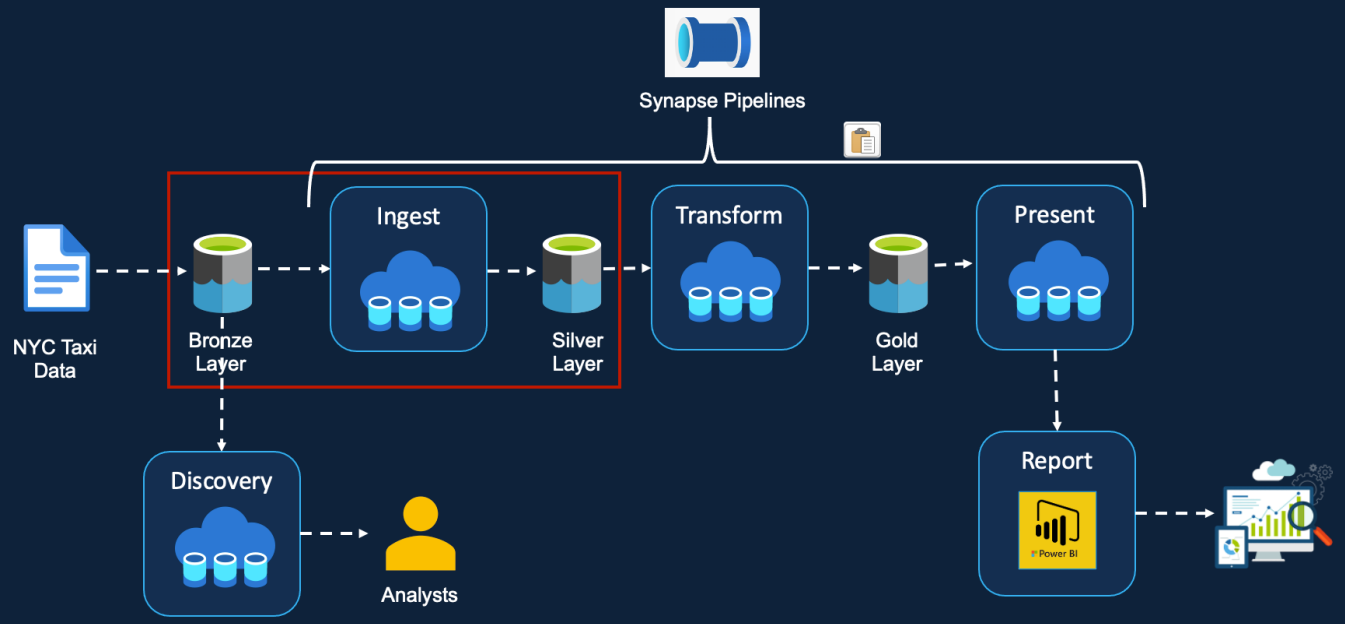
Combination of Views & OpenRowset can be used to prune partitions

External Tables vs Views

Section Overview - Data Ingestion



Section Overview - Data Ingestion



- CETAS Statement Overview
- Transform Delimited to Parquet
- Transform JSON to Parquet
- Challenges Processing Partitioned Data
- Stored Procedures Introduction
- Transform Partitioned Data
- Create Views

Data Transformation

Write the data in Columnar format

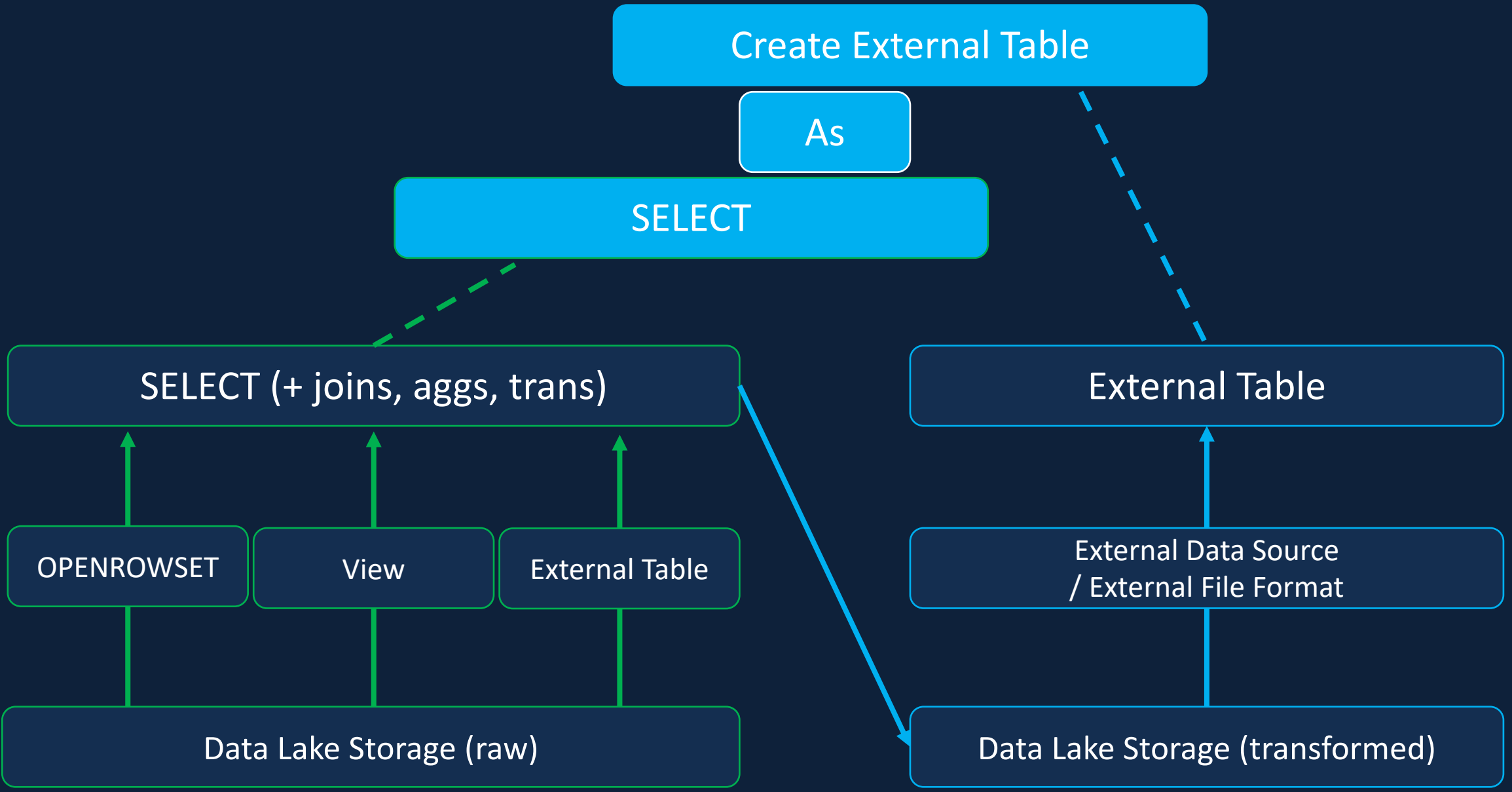
Remove unwanted columns

Transform semi-structured data (e.g. JSON)

Store pre-aggregated data

Build a traditional data warehouse

Create External Table As Select (CETAS)



CREATE EXTERNAL TABLE AS SELECT (CETAS)

```
CREATE EXTERNAL TABLE {[database_name .] [ schema_name . ] table_name }
        [(column_name [,...n ] ) ]
WITH (
        LOCATION = 'path_to_folder',
        DATA_SOURCE = external_data_source_name,
        FILE_FORMAT = external_file_format_name
    )
AS
<select_statement>
[;]
```

CREATE EXTERNAL TABLE AS SELECT (CETAS)

```
CREATE EXTERNAL TABLE transformed.taxi_zone
WITH (
            LOCATION = 'transformed/taxi_zone',
            DATA_SOURCE = nyc_taxi_data,
            FILE_FORMAT = parquet_file_format
        )
AS
SELECT *
 FROM raw.taxi_zone ;
```

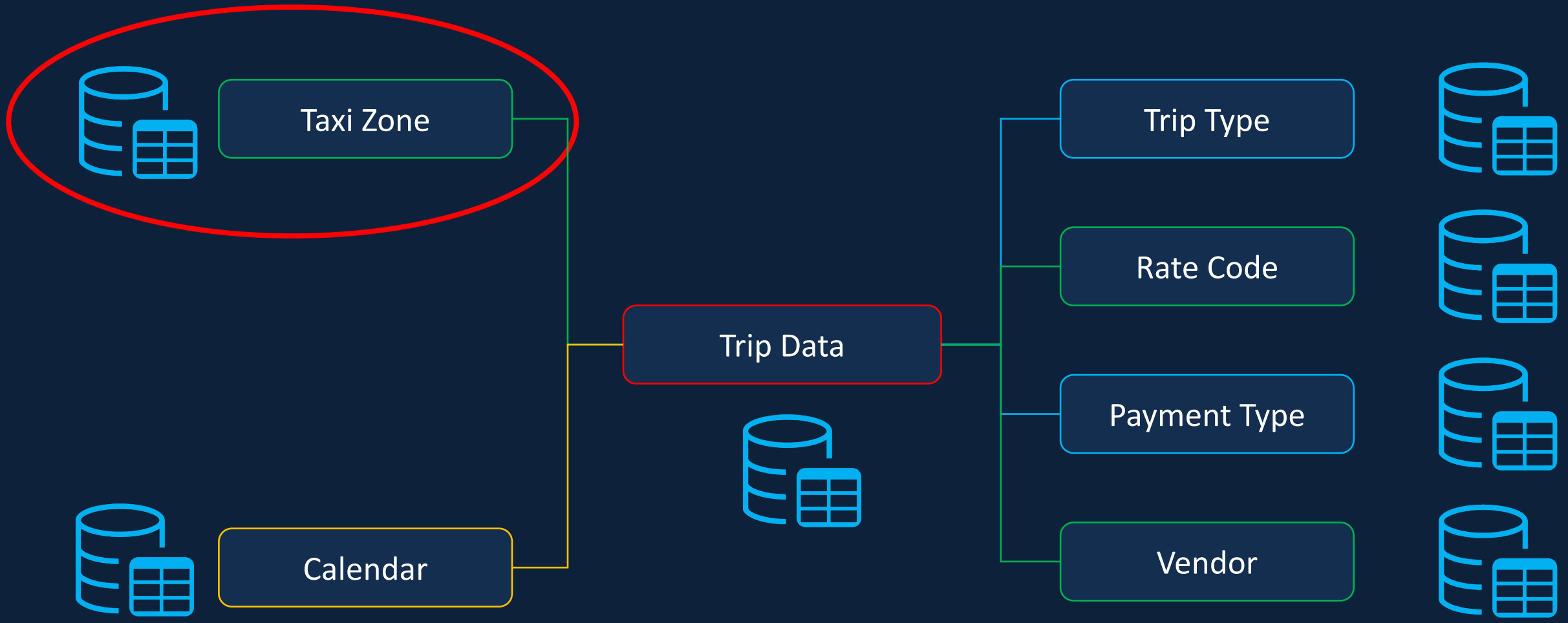
CREATE EXTERNAL TABLE AS SELECT (CETAS)

```
CREATE EXTERNAL TABLE transformed.taxi_zone
WITH (
        LOCATION = 'transformed/taxi_zone',
        DATA_SOURCE = nyc_taxi_data,
        FILE_FORMAT = parquet_file_format
    )
AS
SELECT *
 FROM
OPENROWSET(
                BULK 'abfss://nyc-taxi-data@synapsecoursedl.dfs.core.windows.net/raw/taxi_zone.csv',
                FORMAT = 'CSV',
                    PARSER_VERSION = '2.0',
                FIRSTROW = 2
            )
    WITH (
            location_id SMALLINT 1,
            borough VARCHAR(15) 2,
            zone VARCHAR(50) 3,
            service_zone VARCHAR(15) 4
        ) AS [result];
```

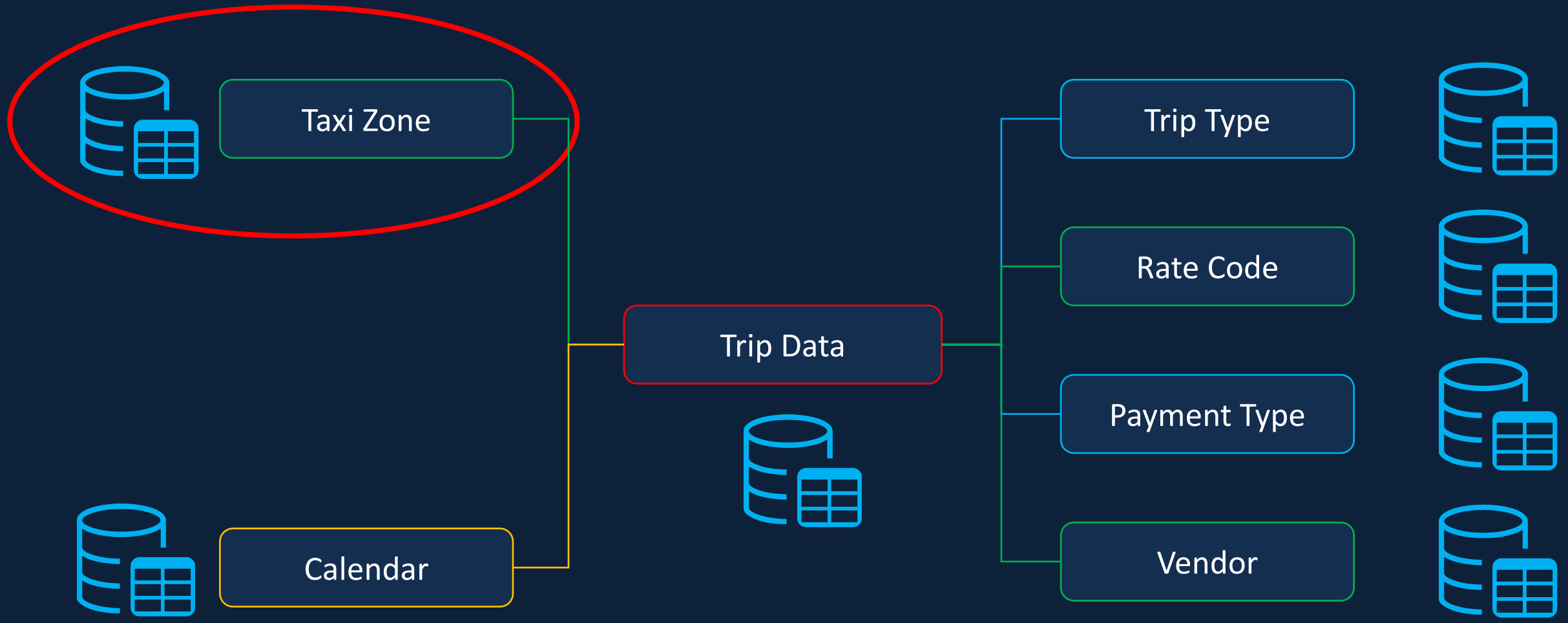
CREATE EXTERNAL TABLE AS SELECT (CETAS)

```
CREATE EXTERNAL TABLE transformed.taxi_borough
WITH (
            LOCATION = 'transformed/taxi_borough',
            DATA_SOURCE = nyc_taxi_data,
            FILE_FORMAT = parquet_file_format
      )
AS
SELECT borough, COUNT(1) AS number_of_zones
 FROM raw.taxi_zone
GROUP BY borough;
```

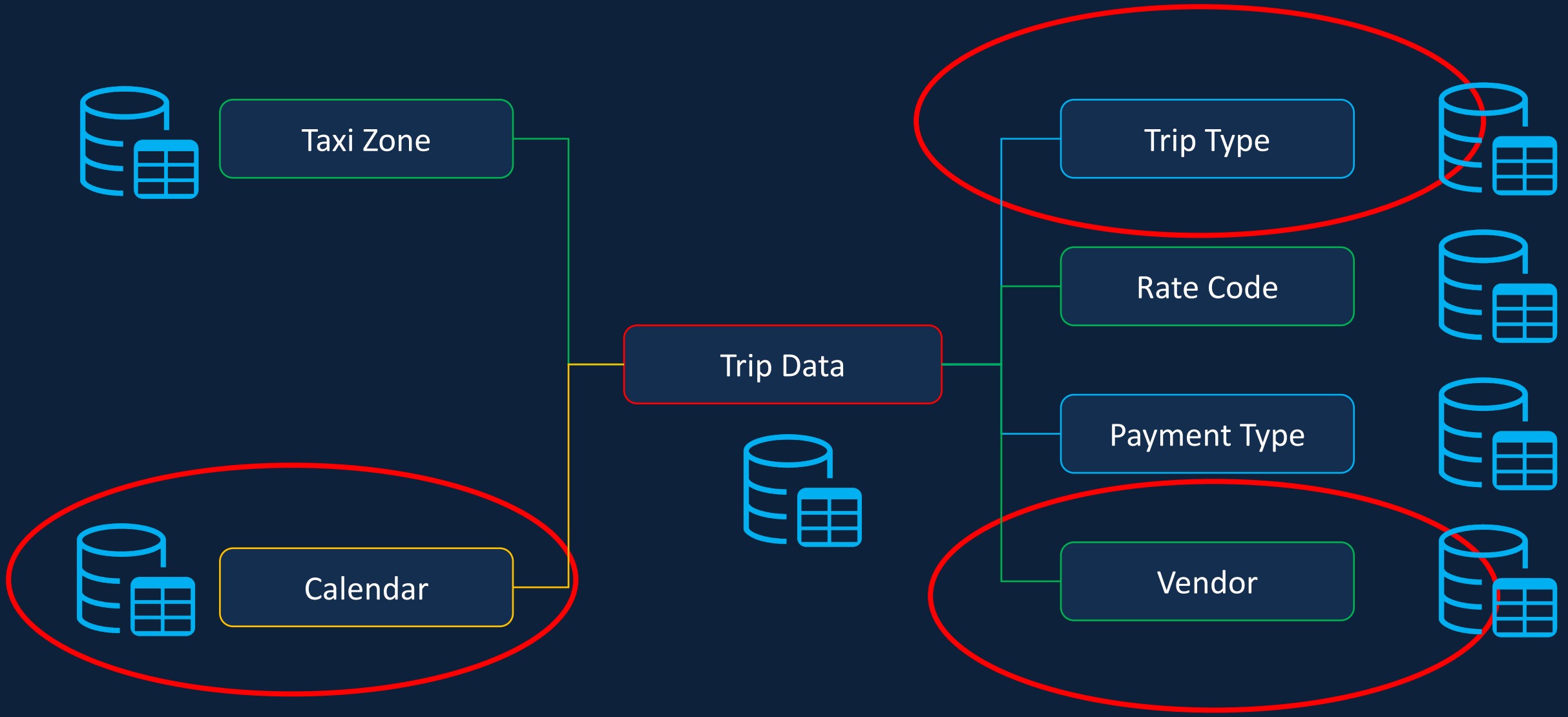
Transform to Parquet Format



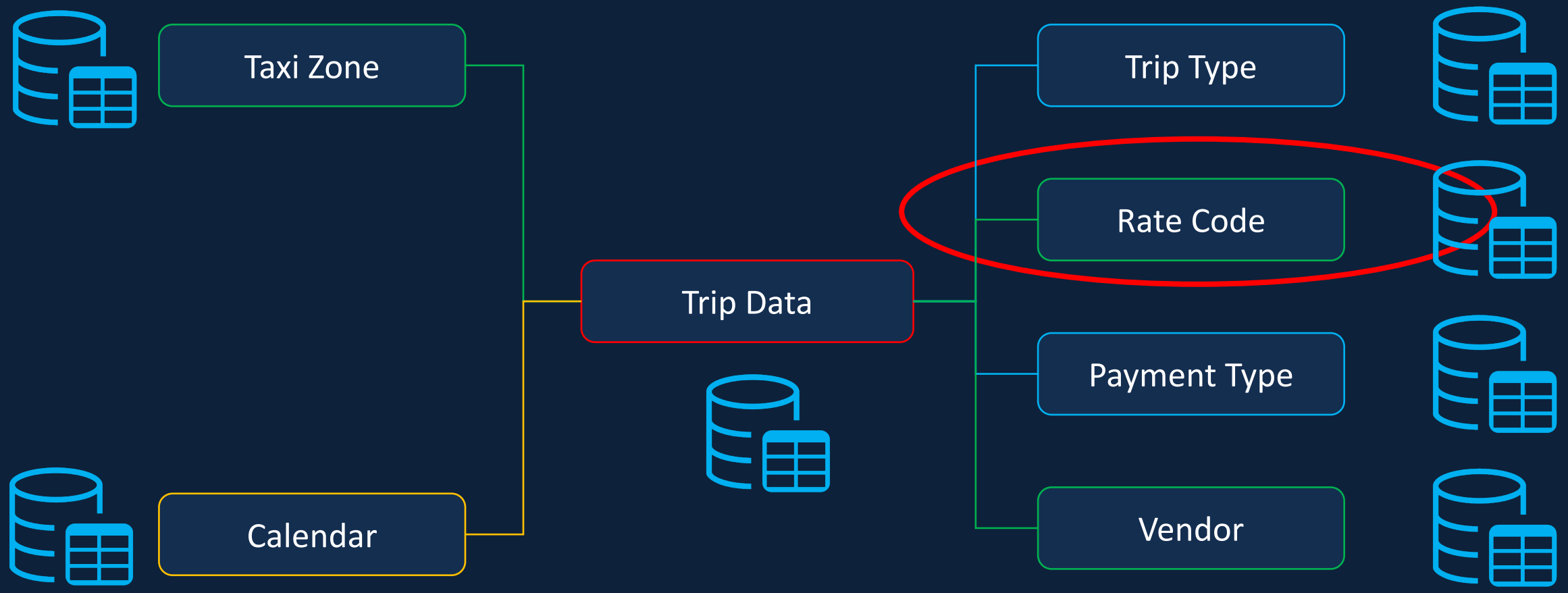
Transform to Parquet Format (Assignment)



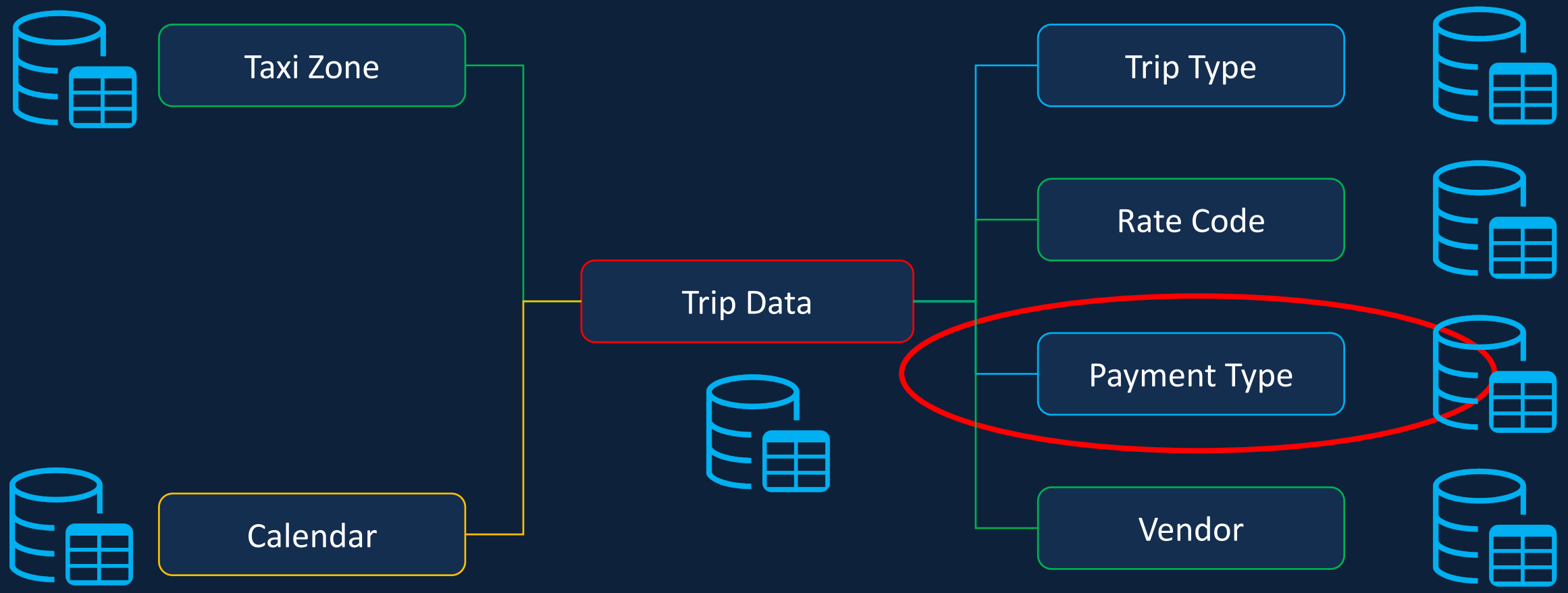
Transform to Parquet Format (Assignment)



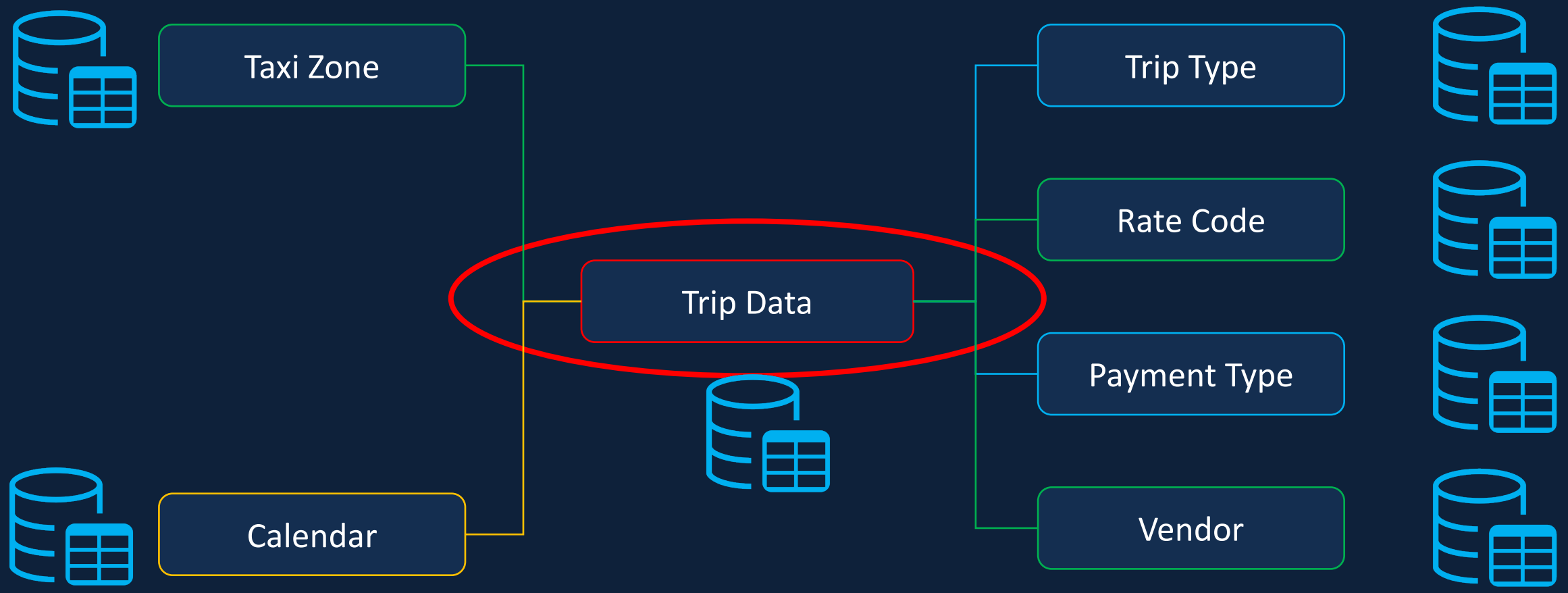
Transform to Parquet Format from JSON



Transform to Parquet Format from JSON (Assignment)



Transform to Parquet Format – Partitioned Files



Stored Procedures

Group of T-SQL statements stored in the database

Accept input parameters & Return output parameters

Can execute another stored procedure

Stored Procedures - Example

```
CREATE PROCEDURE usp_test @borough
AS
BEGIN
	SELECT *
	 FROM bronze.taxi_zone
	WHERE borough = @borough;
END;
```

```
EXEC usp_test @borough = ' Queens'
```

Stored Procedures - Benefits

Reuse of code

Easier maintenance

Improved security

Stored Procedures - Limitations

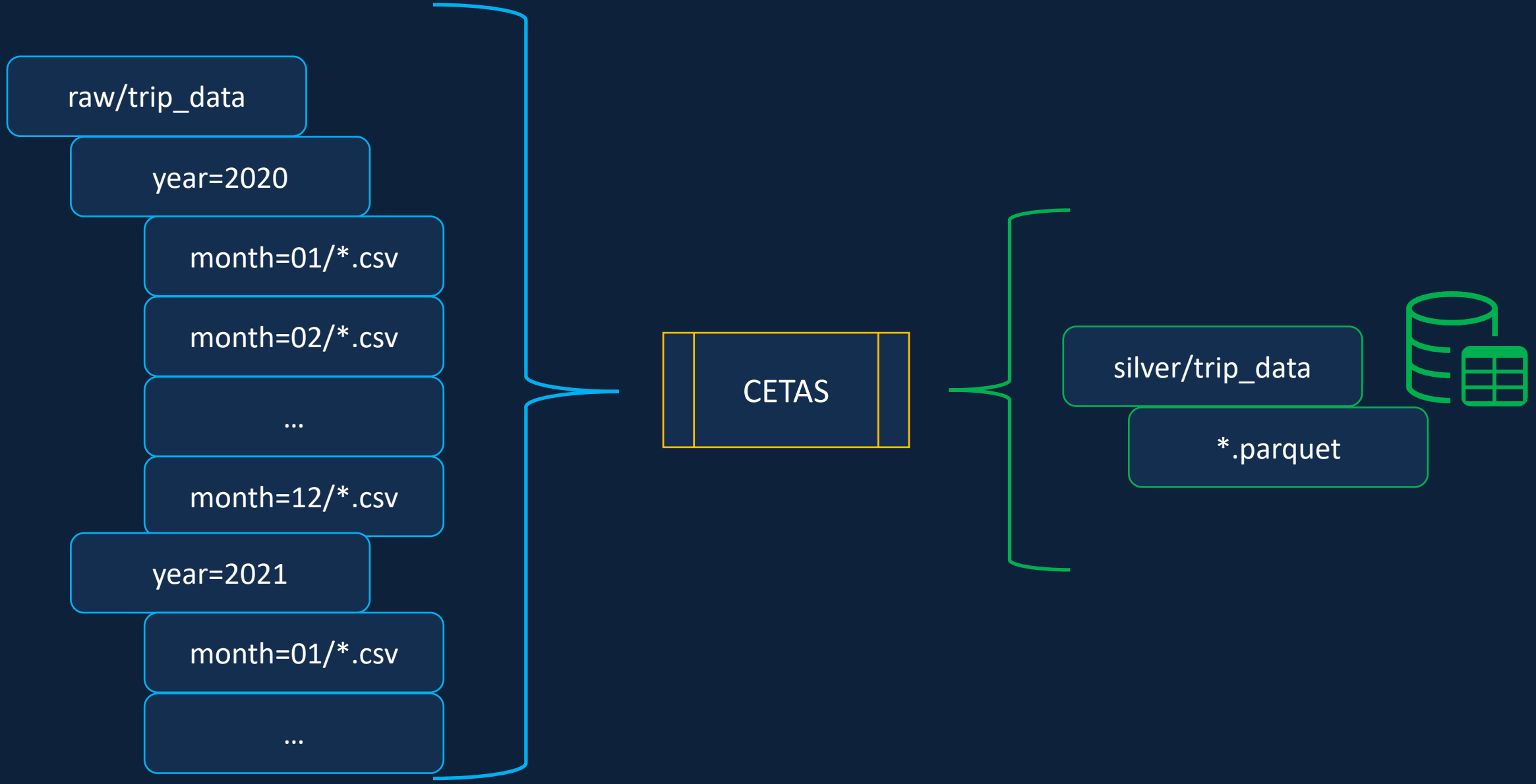
T-SQL support

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/on-demand-workspace-overview#t-sql-support

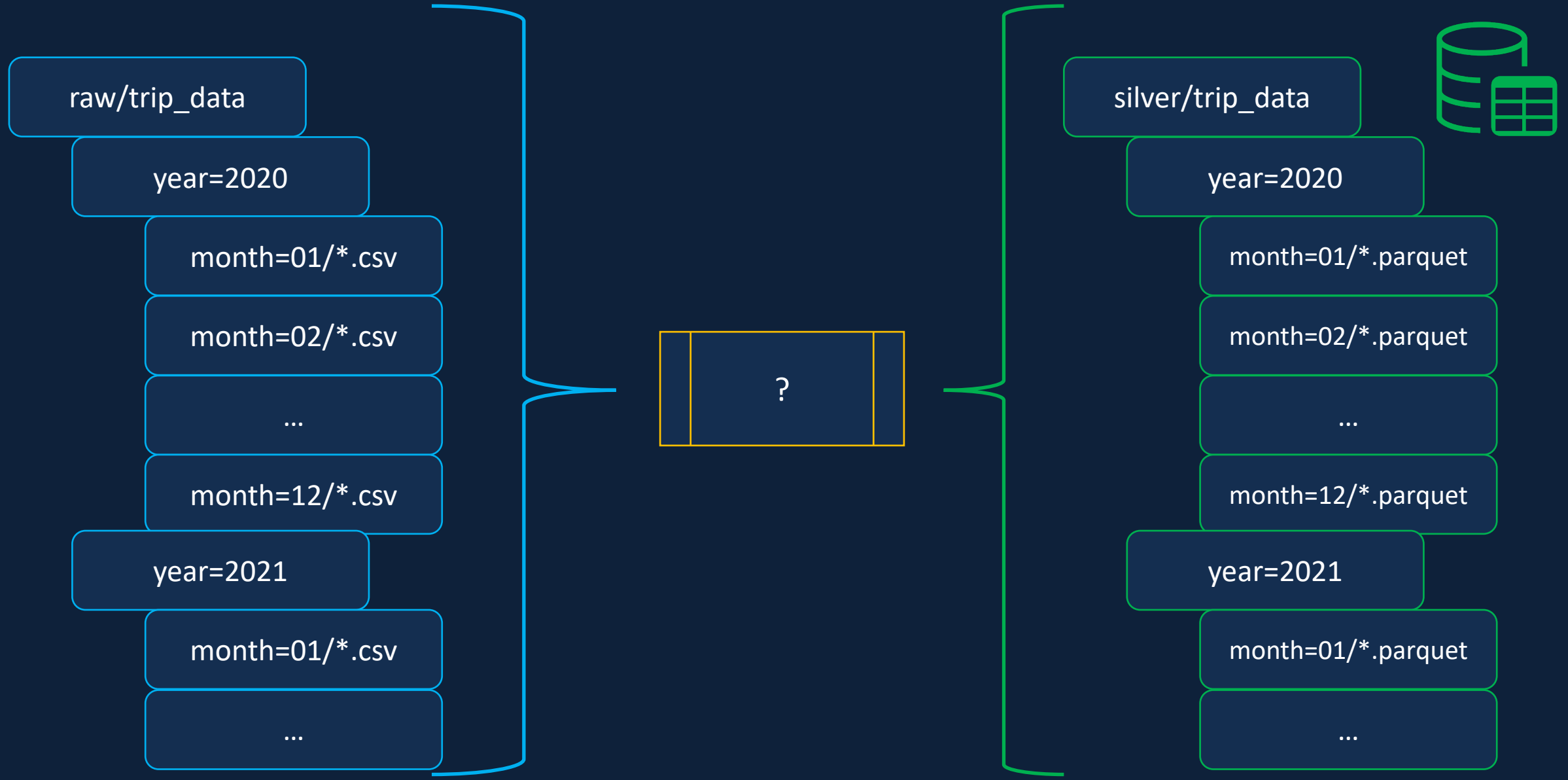
Reduced implementation

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-stored-procedures#limitations

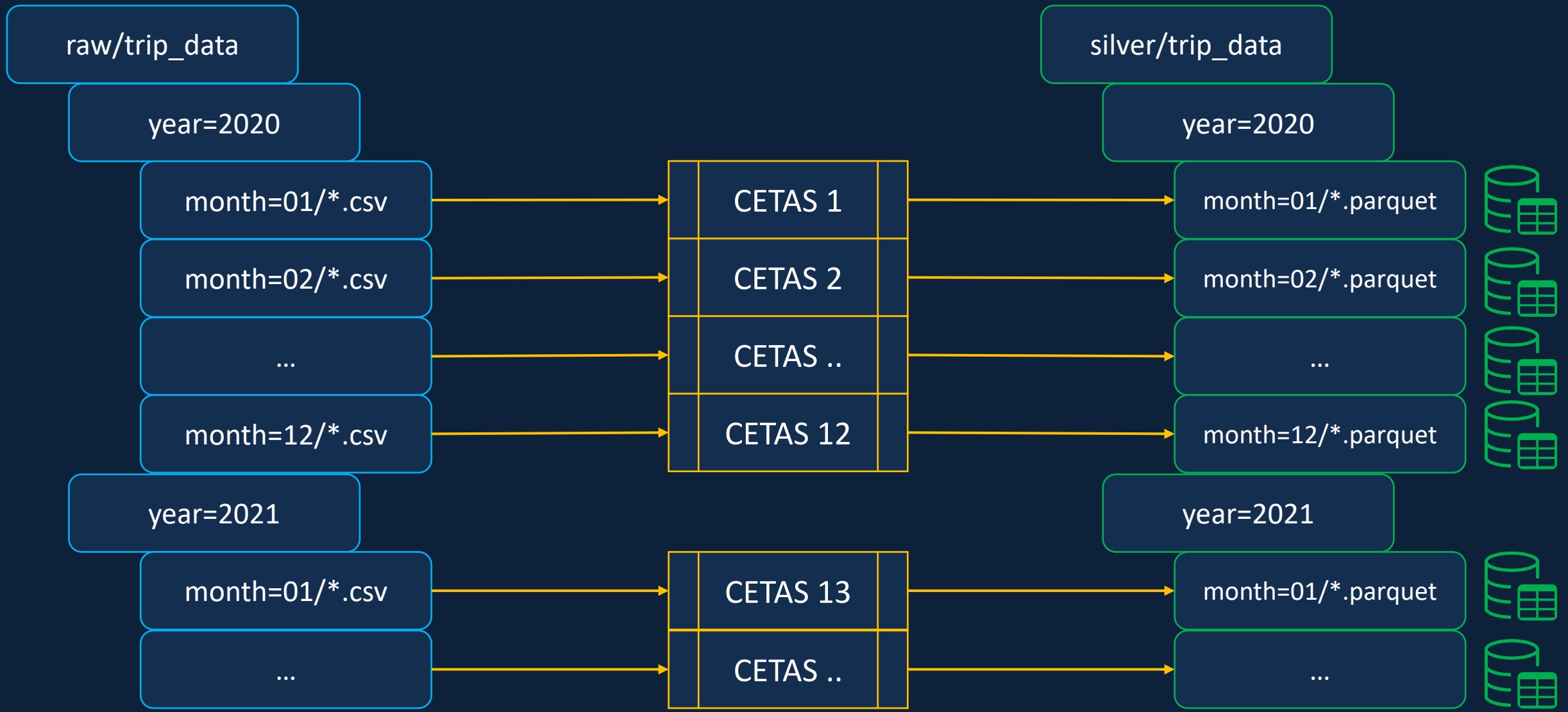
Transform to Parquet Format – Partitioned Files



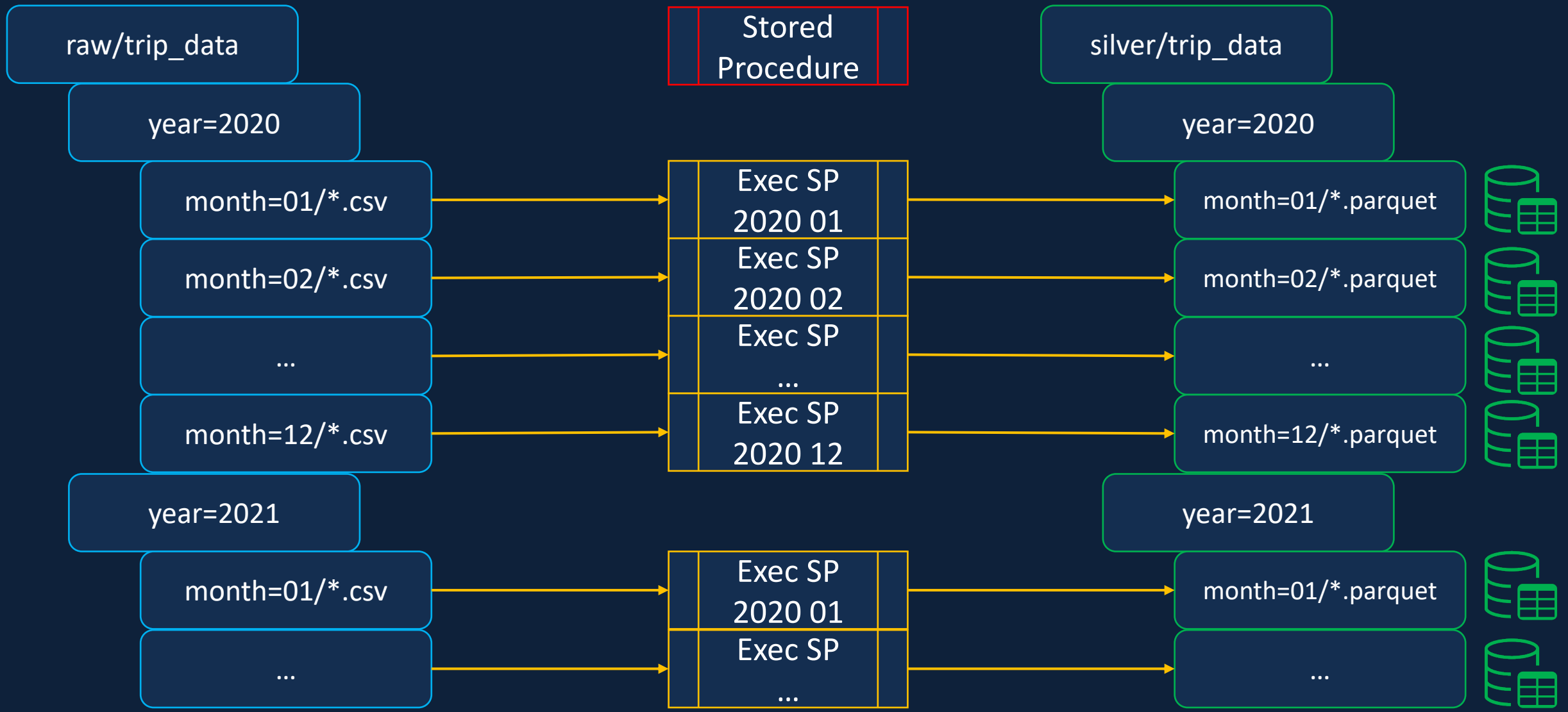
Transform to Parquet Format – Partitioned Files



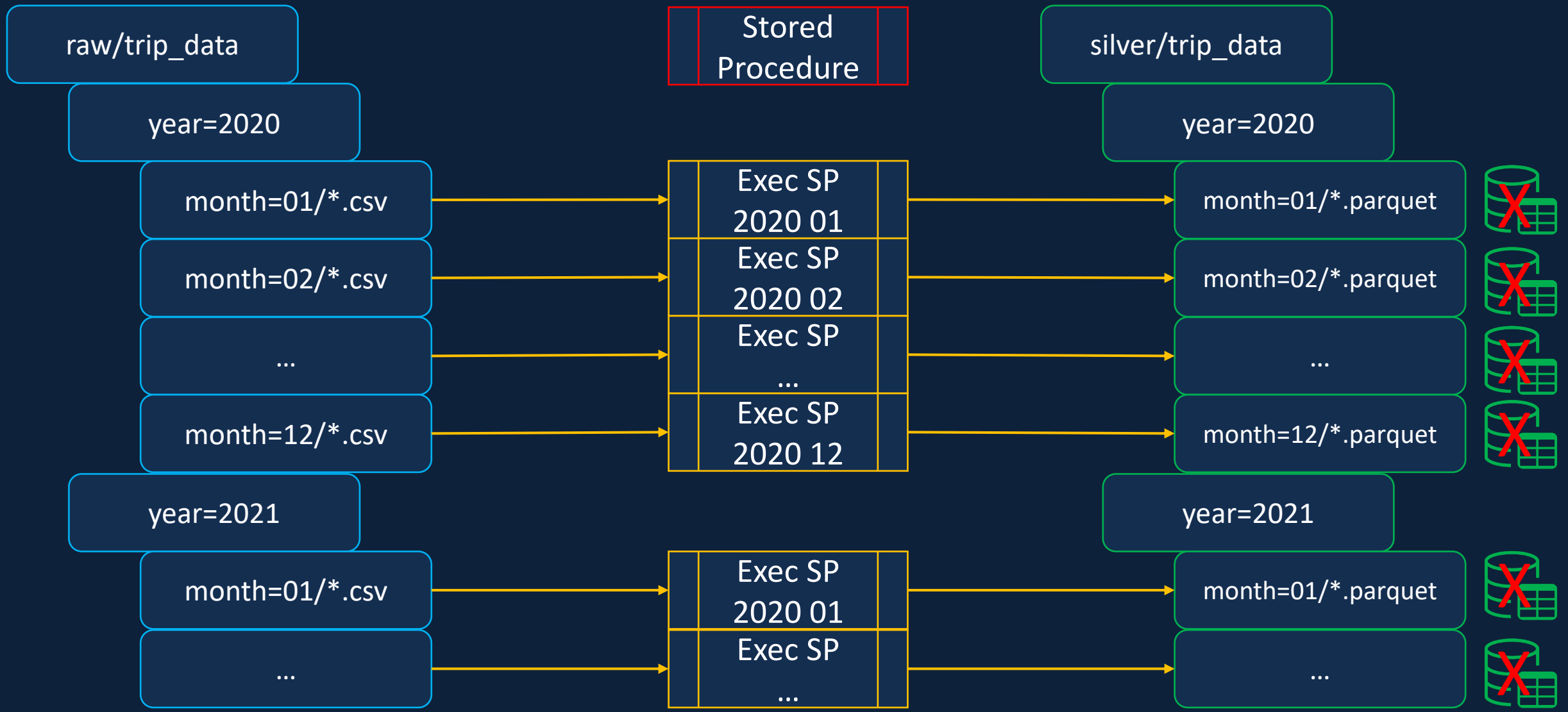
Transform to Parquet Format – Partitioned Files



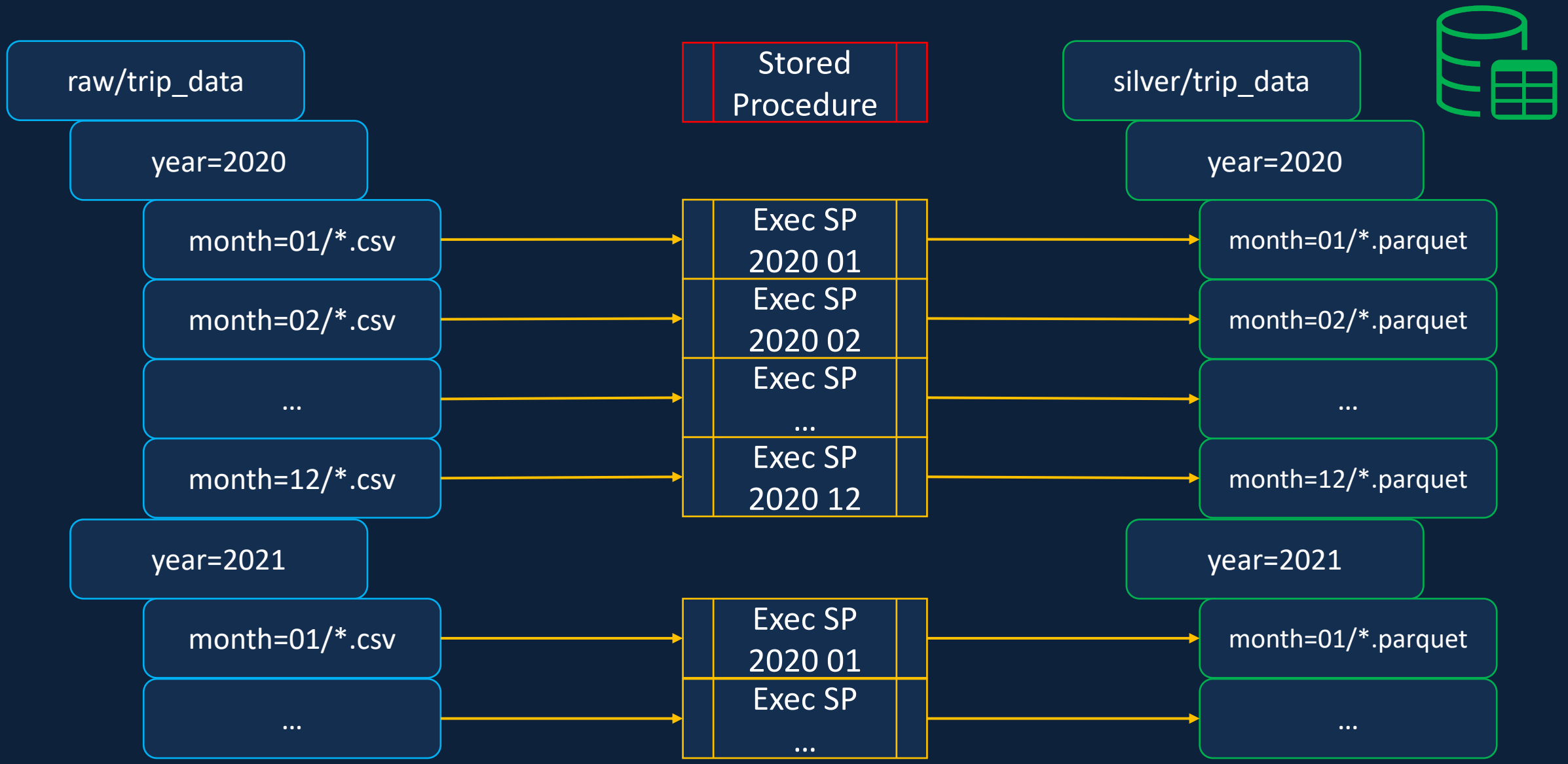
Transform to Parquet Format – Partitioned Files



Transform to Parquet Format – Partitioned Files



Transform to Parquet Format – Partitioned Files



Transform to Parquet Format – Partitioned Files

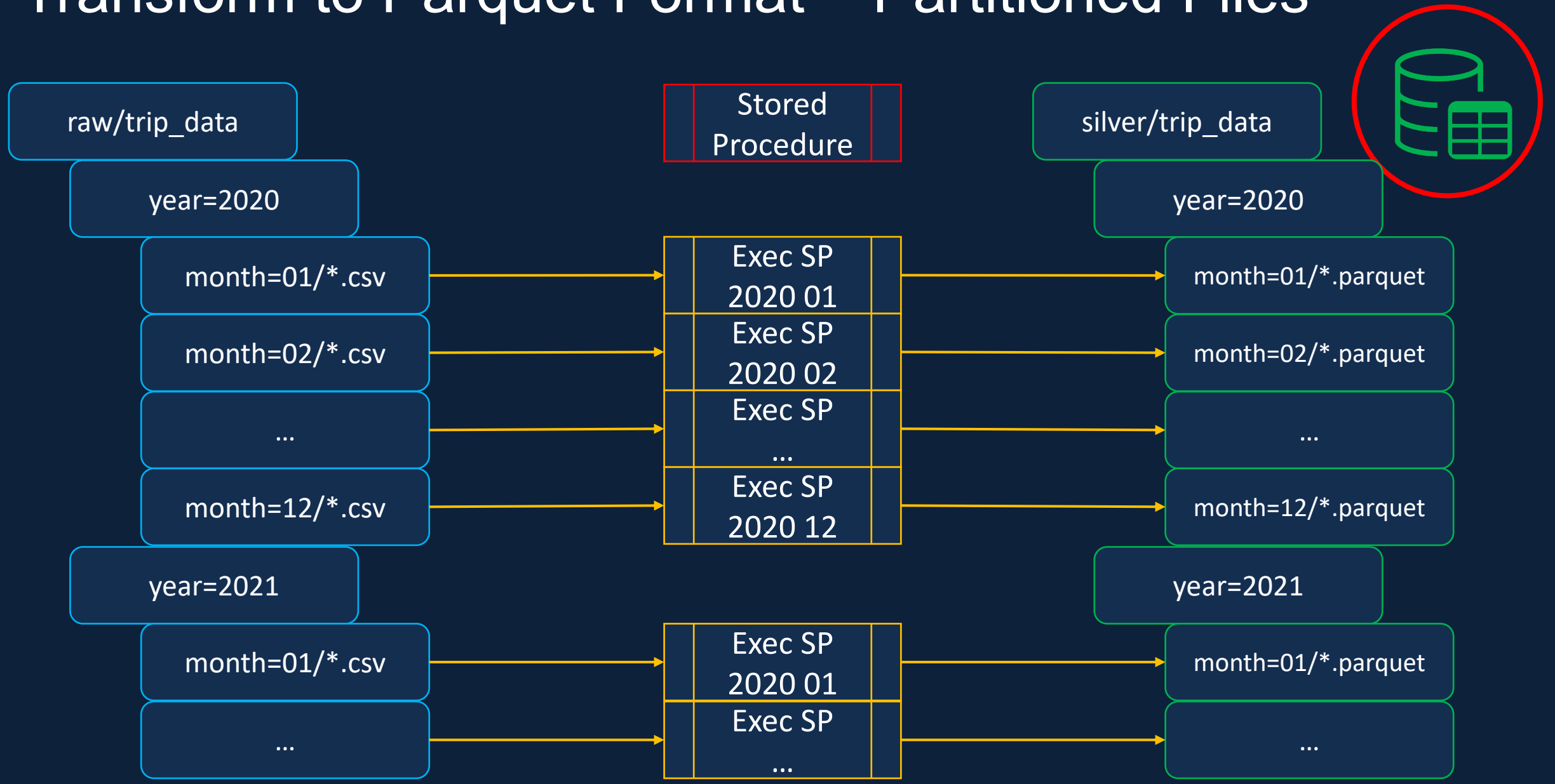
Stored Procedure – CETAS to transform data

Stored Procedure – DROP External Tables

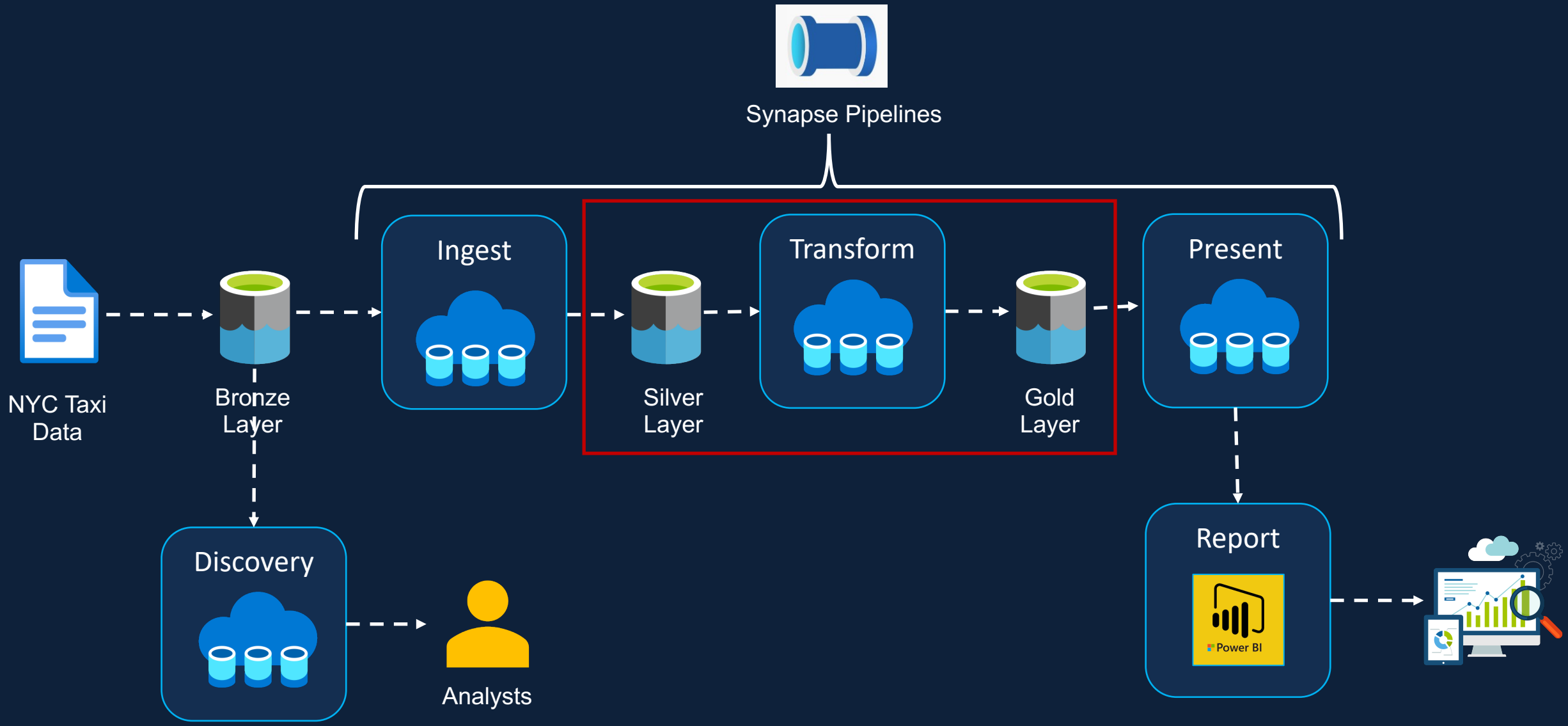
Execute store procedure for each partition

Create view with partitioned columns

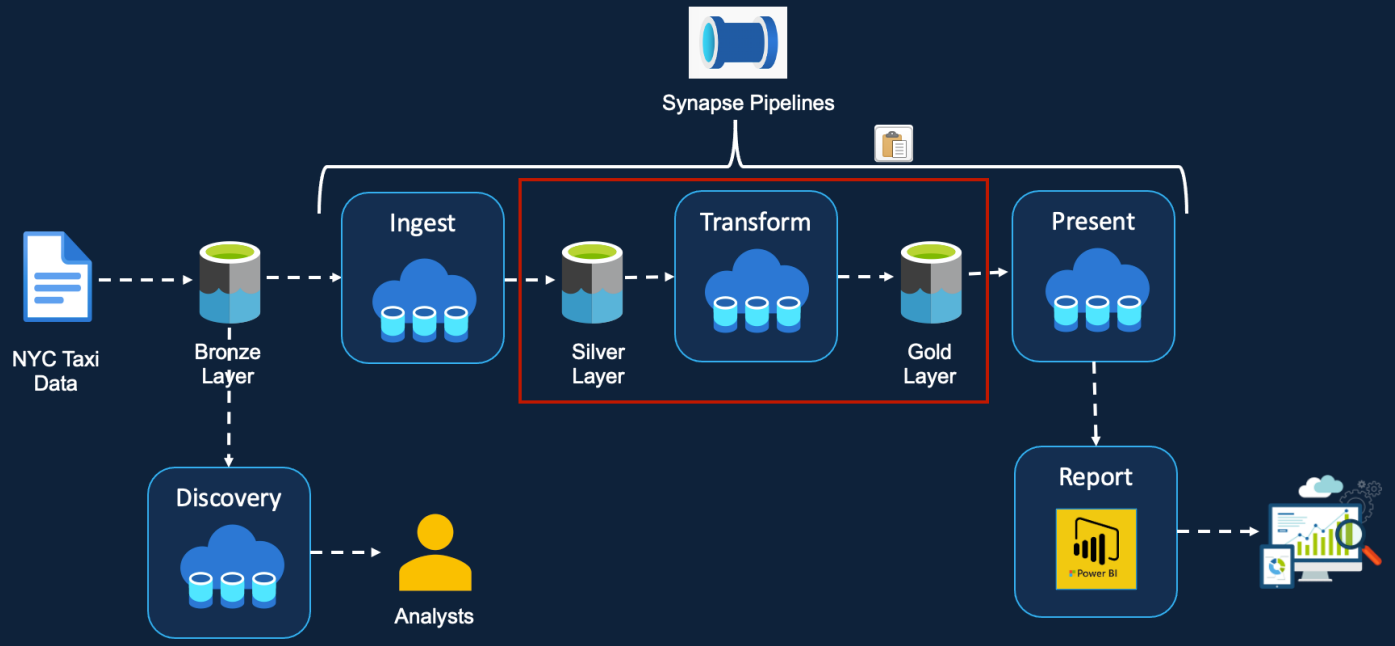
Transform to Parquet Format – Partitioned Files



Section Overview - Data Transformation



Section Overview - Data Transformation



- Project Requirements
- Data for Campaign Analysis
- Data for Taxi Demand
- Create View

Business Requirements (1)

Campaign to encourage credit card payments

Trips made using credit card/ cash payments

Payment behaviour during days of the week/ weekend

Payment behaviour between boroughs

Non Functional Requirements

Reporting data to be pre-aggregated for better performance

Pre-aggregate data for each year/ month partition in isolation

Able to read data efficiently for specific months from aggregated data

Minimize the number of aggregated tables created

Business Requirements

Campaign to encourage credit card payments

Trips made using credit card/ cash payments

Payment behaviour during days of the week/ weekend

Payment behaviour between boroughs

Cash Trip Count Card Trip Count Trip Date

Trip Day Trip Day Week End Ind Borough

Non Functional Requirements

Reporting data to be pre-aggregated for better performance

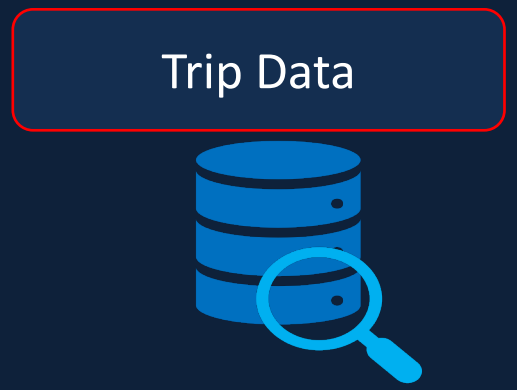
Pre-aggregate data for each year/ month partition in isolation

Able to read data efficiently for specific months from aggregated data

Minimize the number of aggregated tables created

Year Month

Gold Layer



Year

Month

Borough

Trip Date

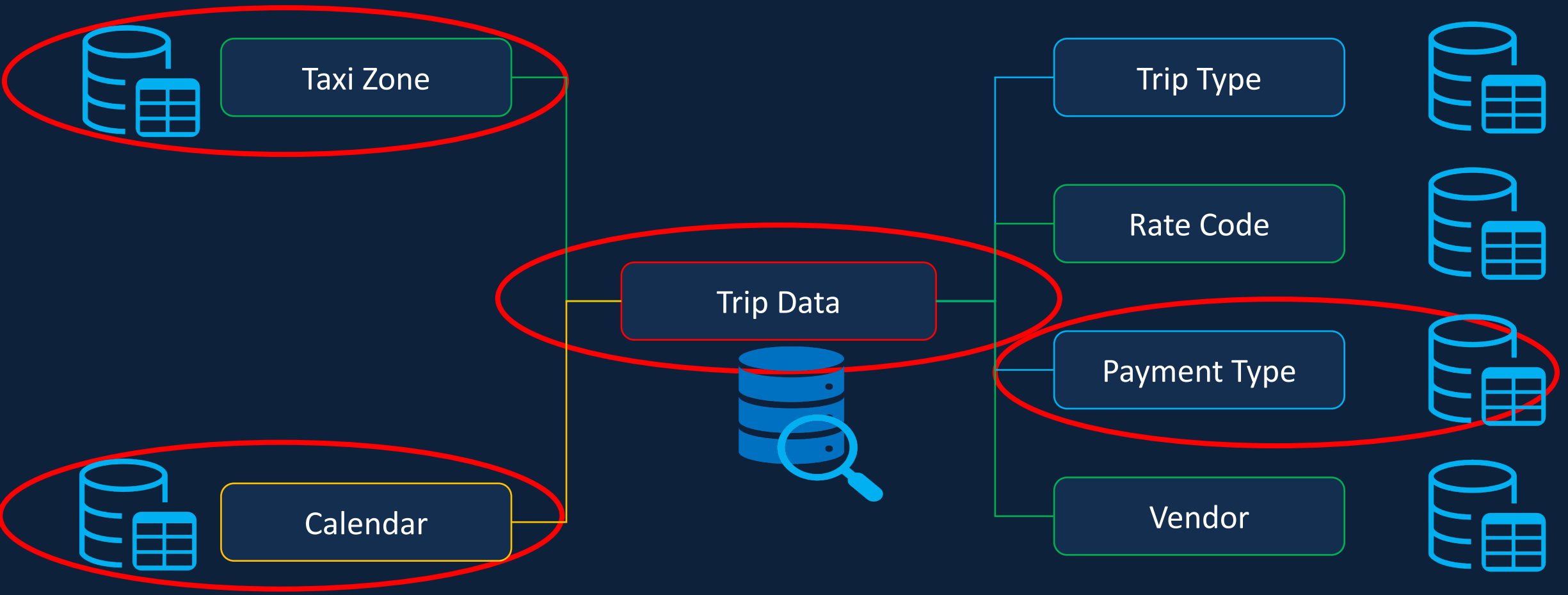
Trip Day

Trip Day Week End Ind

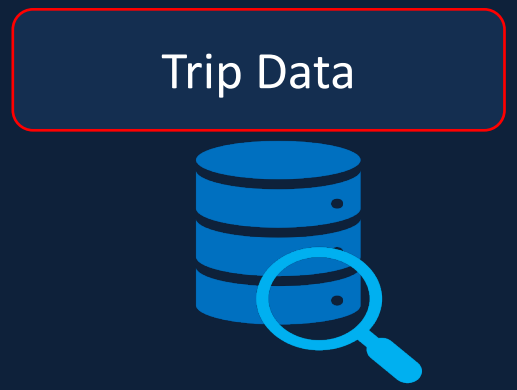
Cash Trip Count

Card Trip Count

Silver Layer

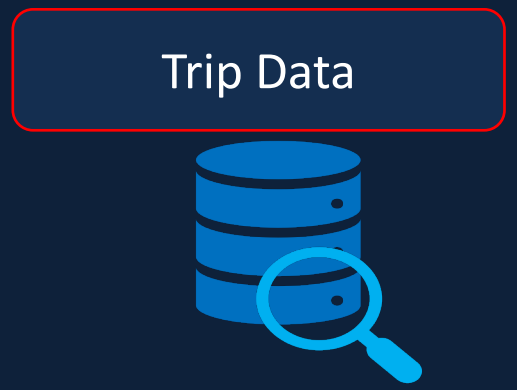


Gold Layer



Year	Trip Data
Month	Trip Data
Borough	Taxi Zone
Trip Date	Trip Data
Trip Day	Calendar
Trip Day Week End Ind	Derived from Trip Day
Cash Trip Count	Payment Type + Trip Data
Card Trip Count	Payment Type + Trip Data

Gold Layer



Year	Trip Data
Month	Trip Data
Borough	Taxi Zone
Trip Date	Trip Data
Trip Day	Calendar
Trip Day Week End Ind	Derived from Trip Day
Cash Trip Count	Payment Type + Trip Data
Card Trip Count	Payment Type + Trip Data

Business Requirements (2)

Identify taxi demand

Demand based on borough

Demand based on day of the week/ weekend

Demand based on trip type (i.e., Street hail/ Despatch)

Trip distance, trip duration, total fare amount etc per day/ borough

Non Functional Requirements

Reporting data to be pre-aggregated for better performance

Pre-aggregate data for each year/ month partition in isolation

Able to read data efficiently for specific months from aggregated data

Minimize the number of aggregated tables created

Business Requirements (2)

Identify taxi demand

Demand based on borough

Demand based on day of the week/ weekend

Demand based on trip type (i.e., Street hail/ Despatch)

Trip distance, trip duration, total fare amount etc per day/ borough

Borough Trip Date Trip Day Trip Day Week End Ind Despatch Trip Count

Street-hail Trip Count Trip Distance Trip Duration Total Fare Amount

Business Requirements (2)

Identify taxi demand

Demand based on borough

Demand based on day of the week/ weekend

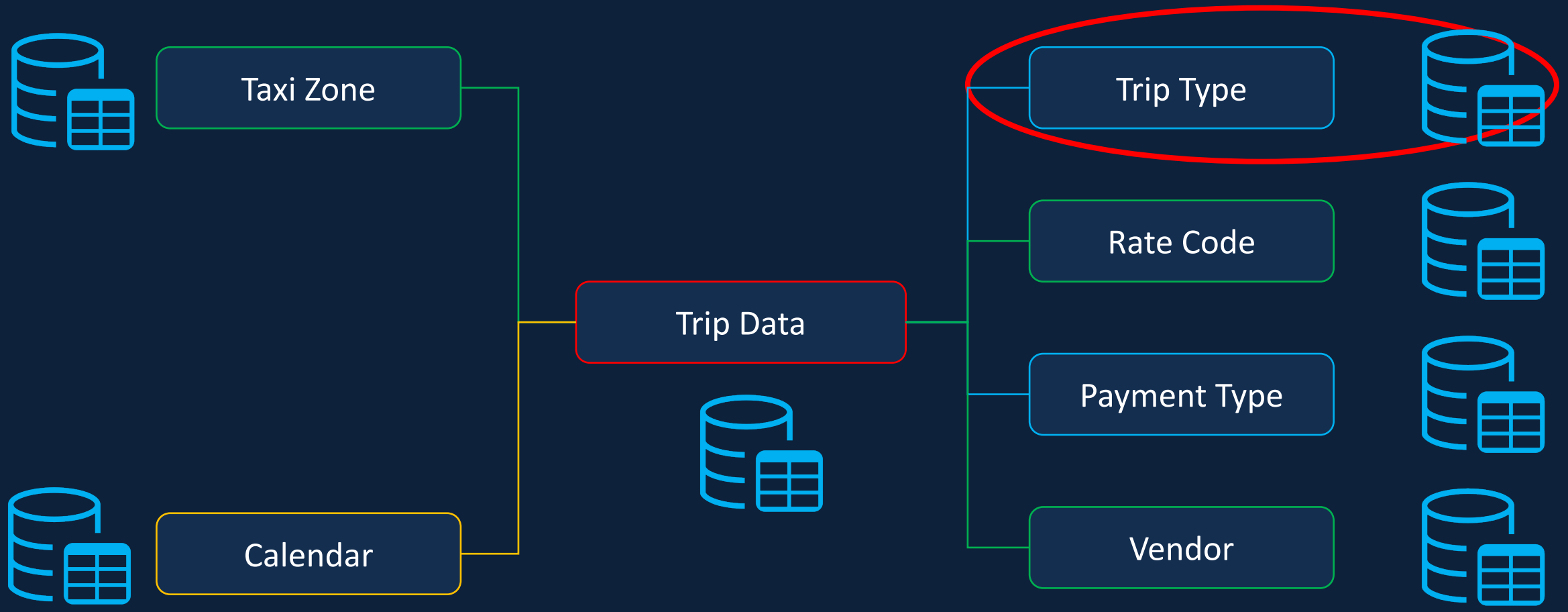
Demand based on trip type (i.e., Street hail/ Despatch)

Trip distance, trip duration, total fare amount etc per day/ borough

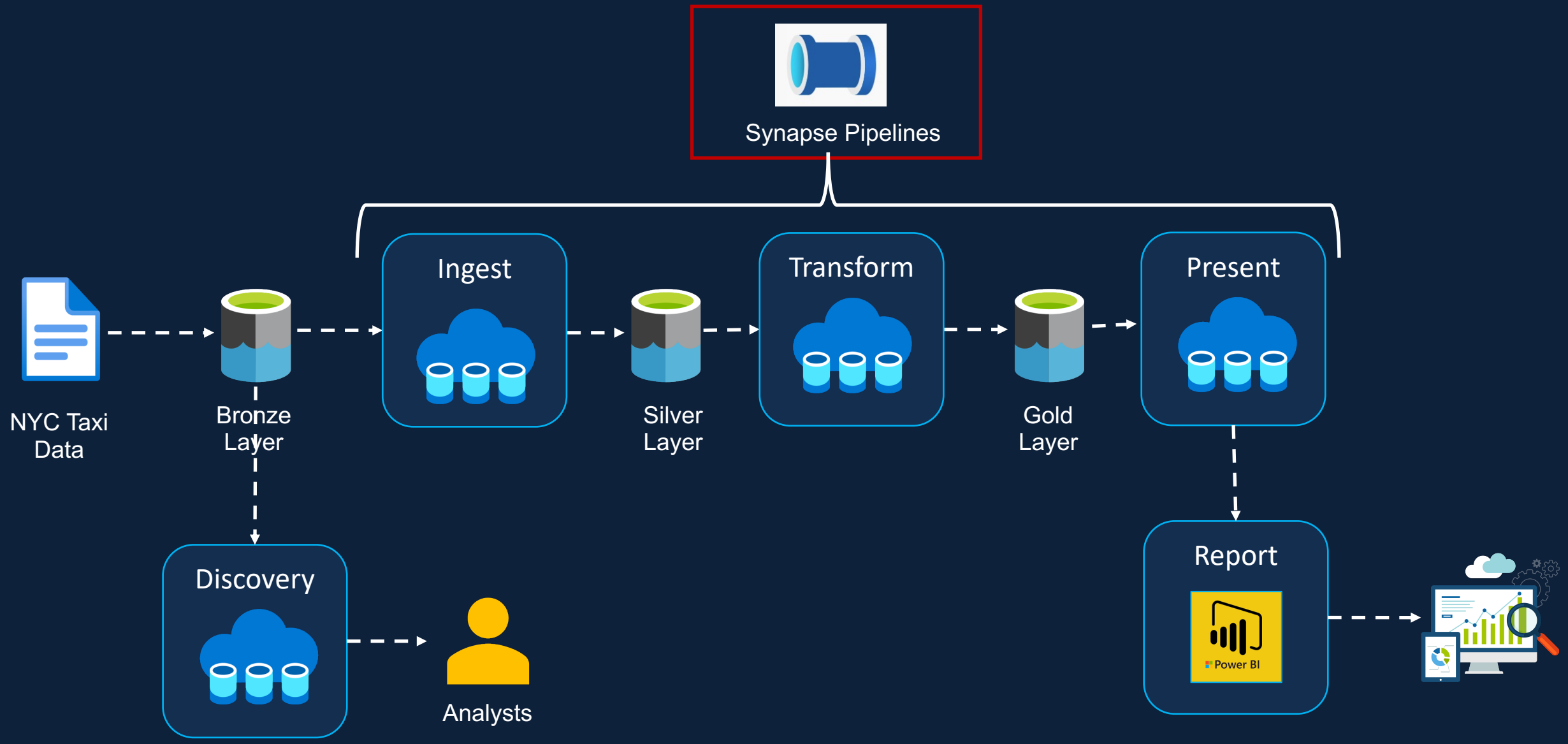
Borough Trip Date Trip Day Trip Day Week End Ind Despatch Trip Count

Street-hail Trip Count Trip Distance Trip Duration Total Fare Amount

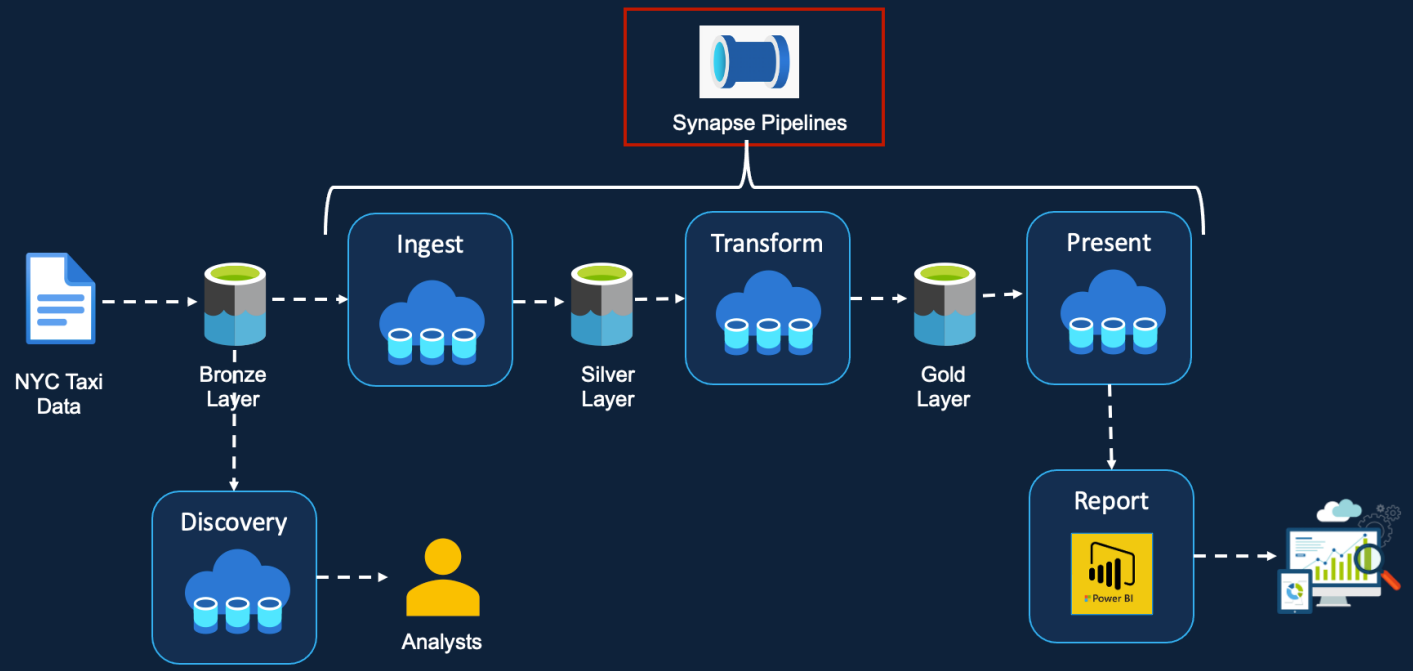
Silver Layer



Section Overview – Synapse Pipelines



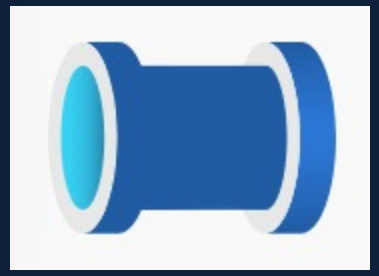
Section Overview - Synapse Pipelines



- Overview
- Components
- Creating Pipelines
- Variables & Parameters
- Dynamic Pipelines
- Pipeline Dependencies
- Creating Triggers

Synapse Pipelines Overview

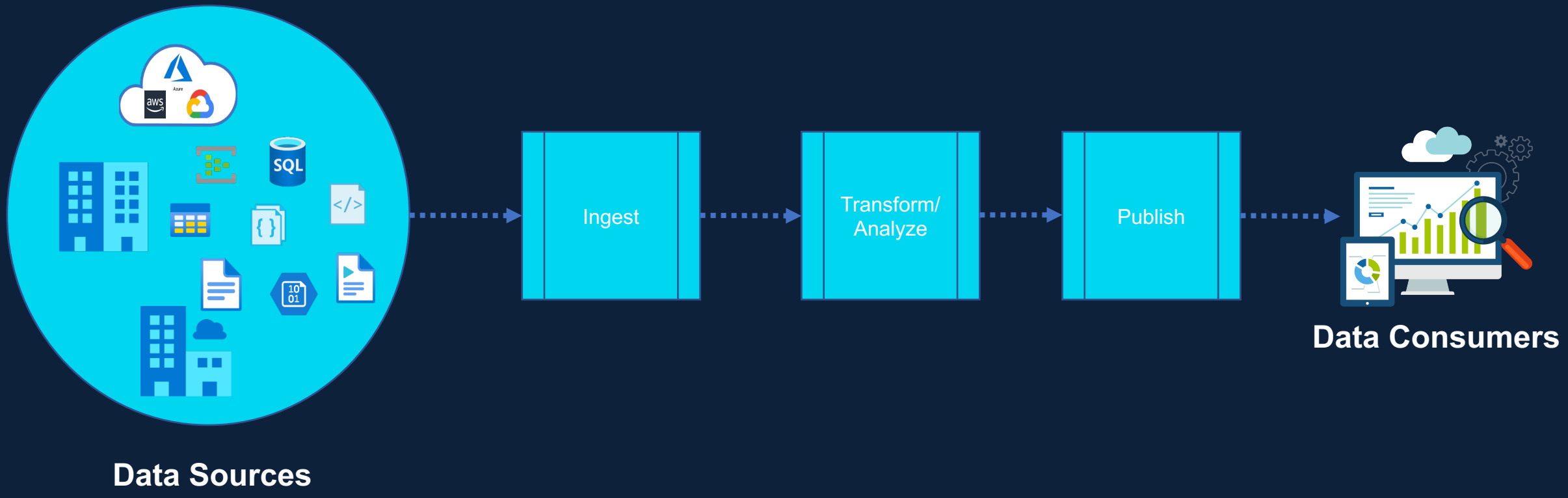
What are Synapse Pipelines



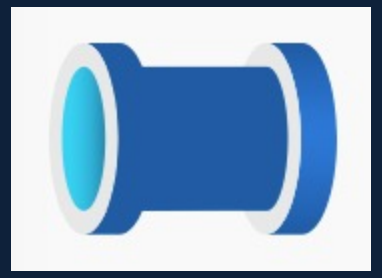
A fully managed, serverless data integration & orchestration service

The Data Problem

The Data Problem



Data Integration



100+ Native connectors

SaaS connectors

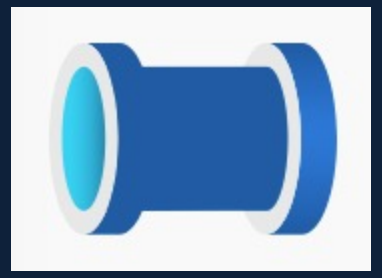
Multi-cloud support

On-premises support

Serverless & Auto Scale

Control flow activities

Data Orchestration



Synapse Data Flows

Synapse Dedicated SQL Pool Scripts

Synapse Serverless SQL Pool Scripts

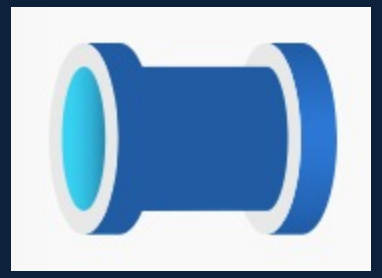
Synapse Spark Notebooks

Azure Databricks Notebooks

Azure HDInsight Scripts

Azure Machine Learning Pipelines

Schedule & Monitor



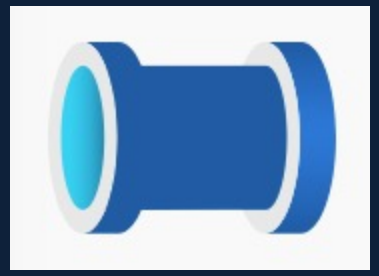
Schedule to run pipelines

Monitoring within the synapse

Alerting within the synapse

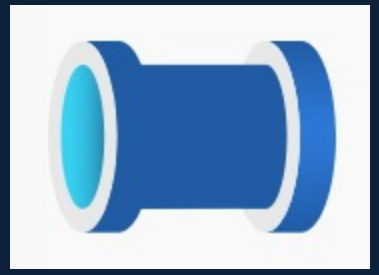
Ability to monitor/ alert from outside of Synapse

What are Synapse Pipelines



A fully managed, serverless data integration & orchestration service

Azure Data Factory vs Synapse Pipelines



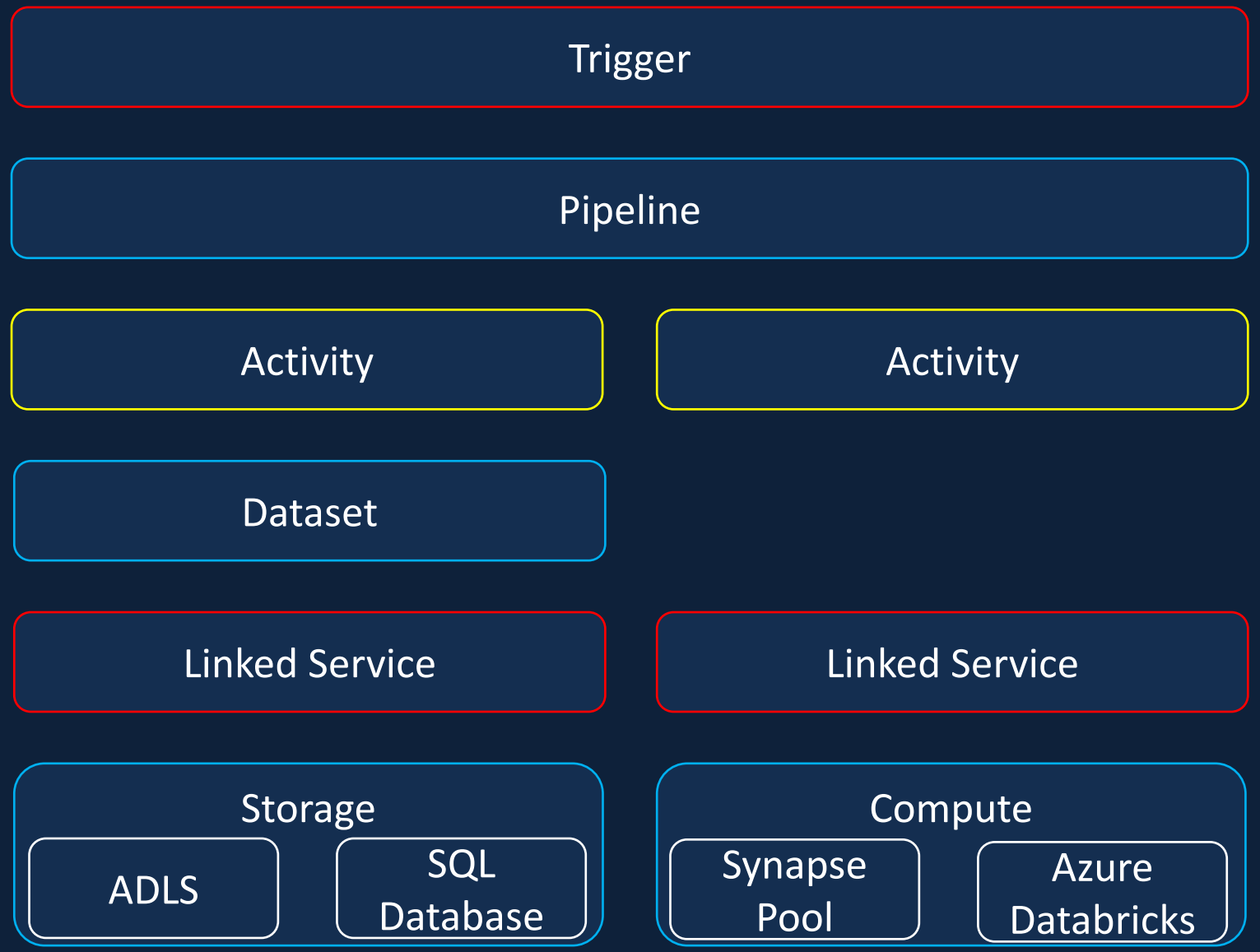
Shares the same codebase, with minor differences

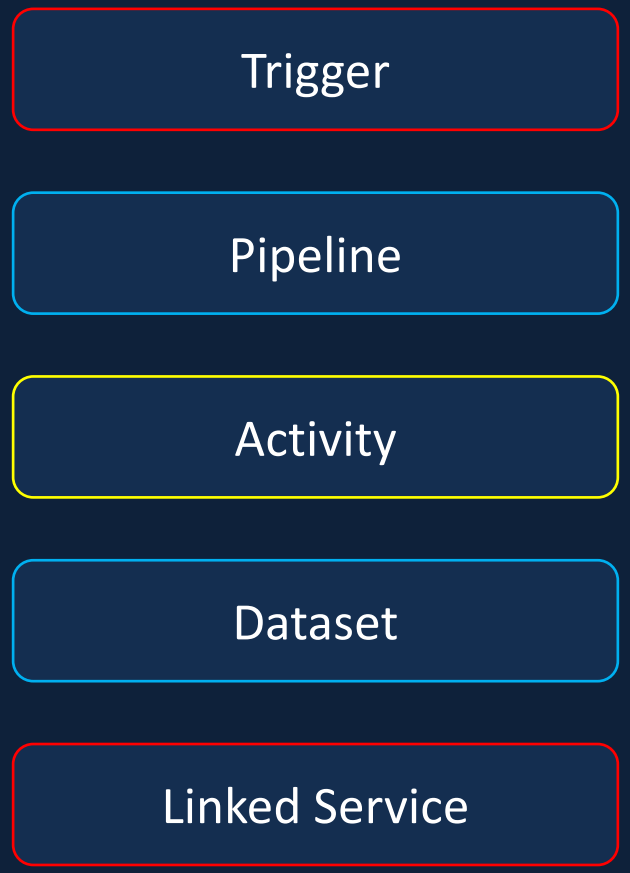
Azure Data Factory vs Synapse Pipelines

Category	Feature	Azure Data Factory	Azure Synapse Analytics
Integration Runtime	Using SSIS and SSIS Integration Runtime	✓	✓ *Public preview*
	Support for Cross-region Integration Runtime (Data Flows)	✓	✗
	Integration Runtime Sharing	✓ *Can be shared across different data factories*	✗
Pipelines Activities	SSIS Package Activity	✓	✓ *Public preview*
	Support for Power Query Activity	✓	✗
Template Gallery and Knowledge center	Solution Templates	✓ *Azure Data Factory Template Gallery*	✓ *Synapse Workspace Knowledge center*
GIT Repository Integration	GIT Integration	✓	✓
Monitoring	Monitoring of Spark Jobs for Data Flow	✗	✓ *Leverage the Synapse Spark pools*

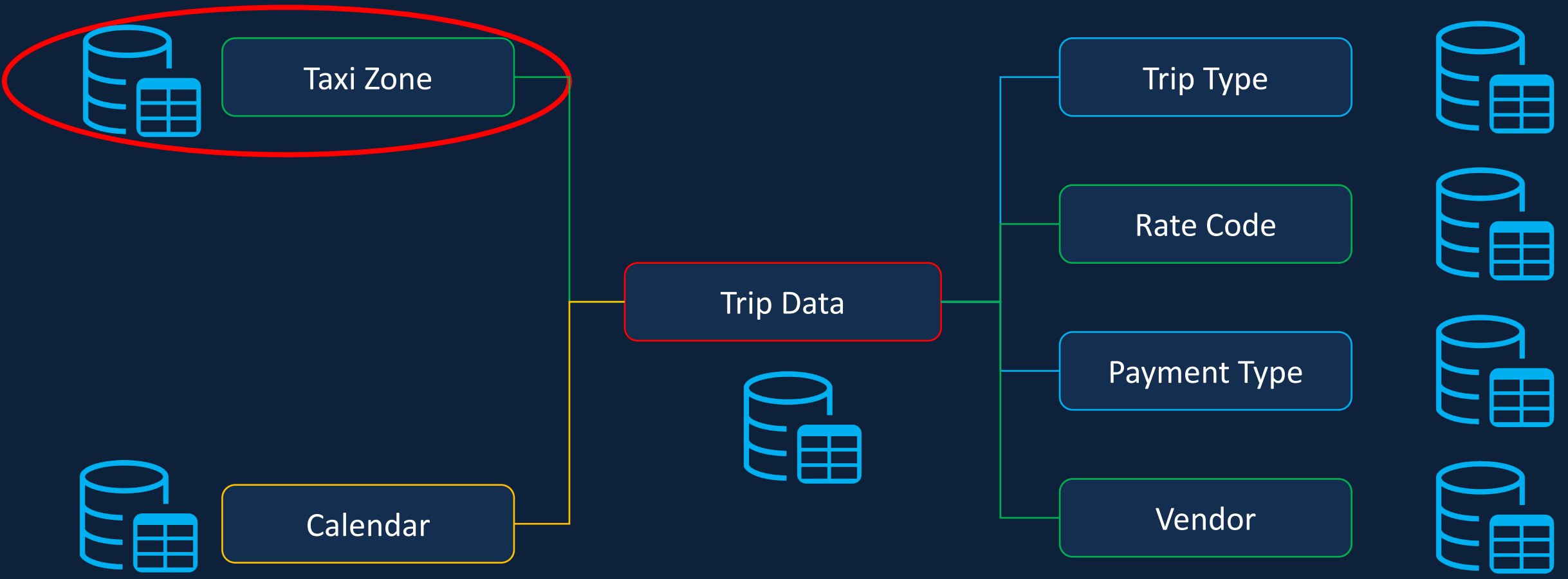
https://docs.microsoft.com/en-us/azure/synapse-analytics/data-integration/concepts-data-factory-differences

Synapse Integration Pipeline Components





Bronze to Silver Layer Transformation

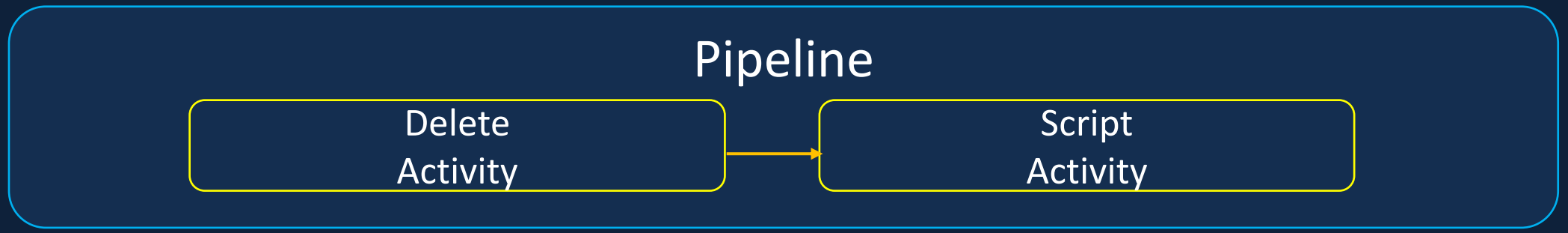


Transform to Parquet Format – Taxi Zone

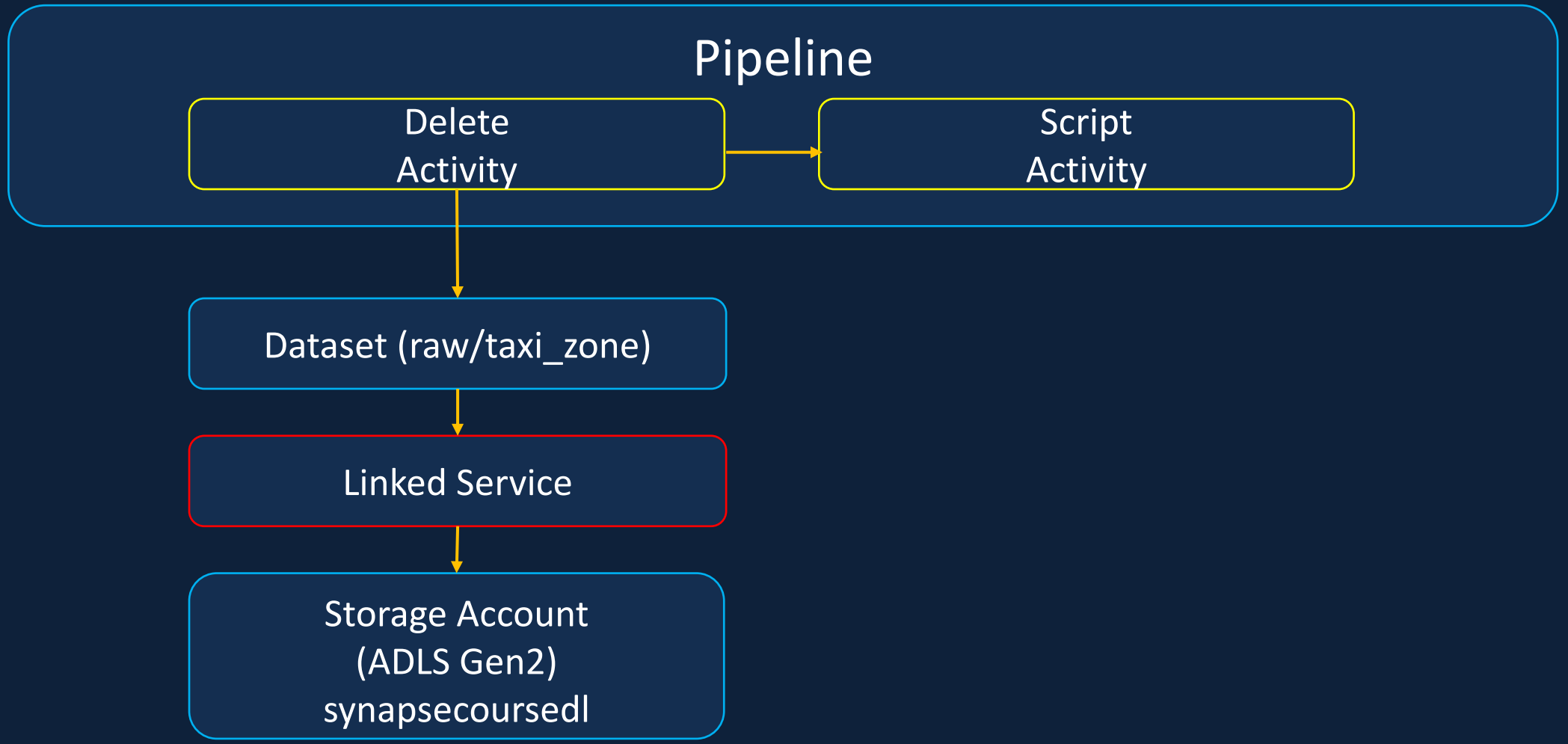
Transform to Parquet Format – Taxi Zone

Transform to Parquet Format – Taxi Zone

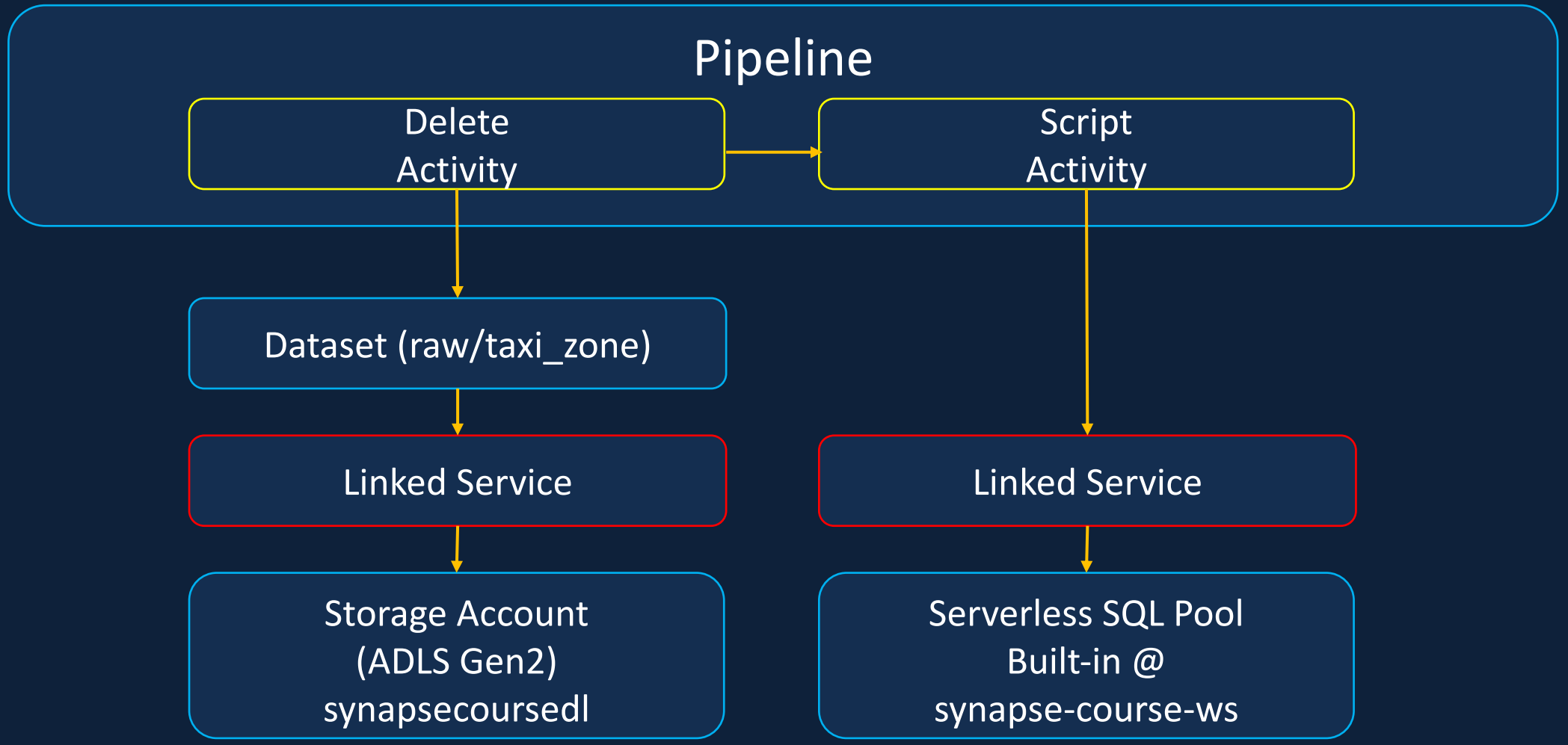
Transform to Parquet Format – Taxi Zone



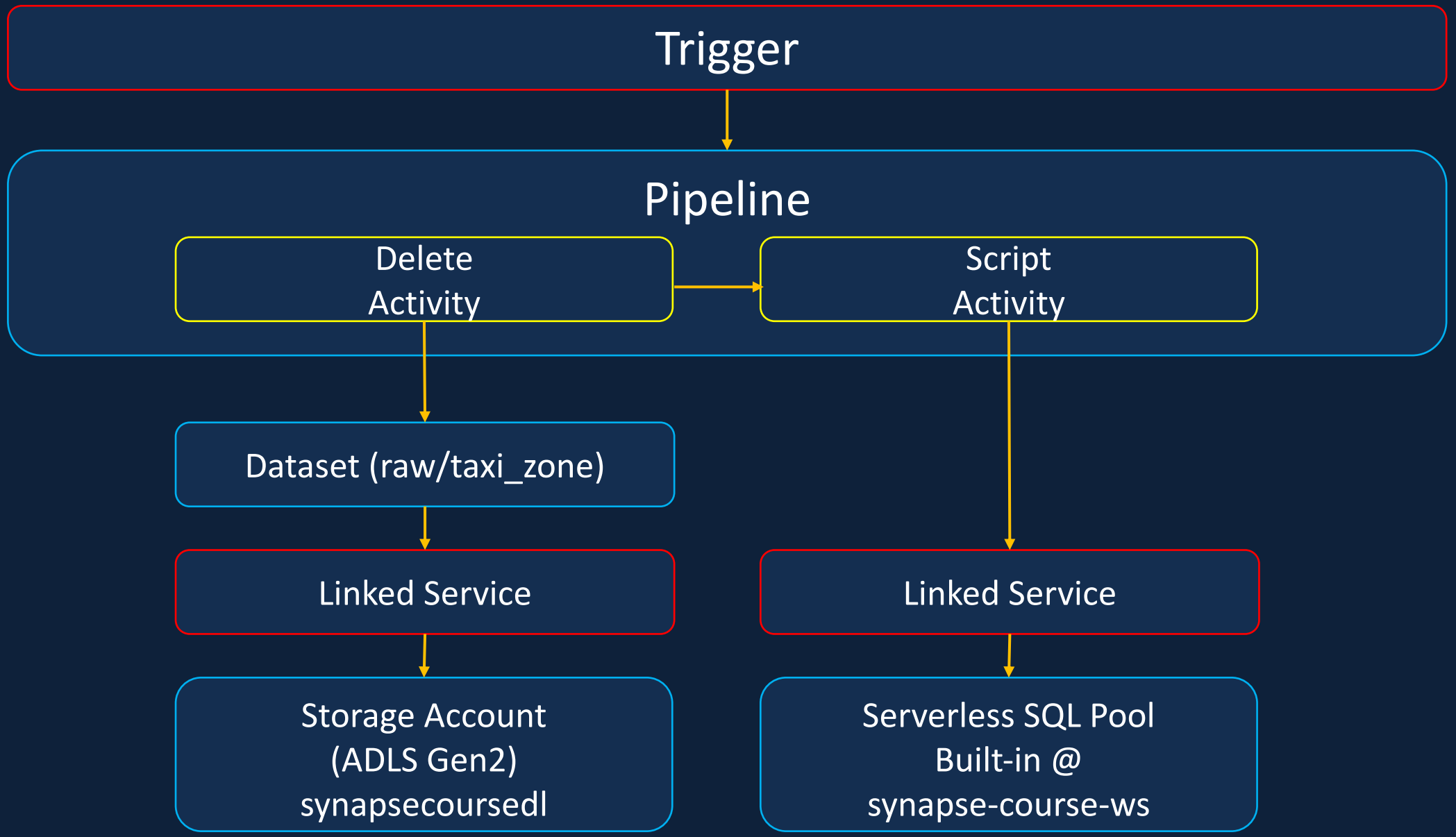
Delete Activity



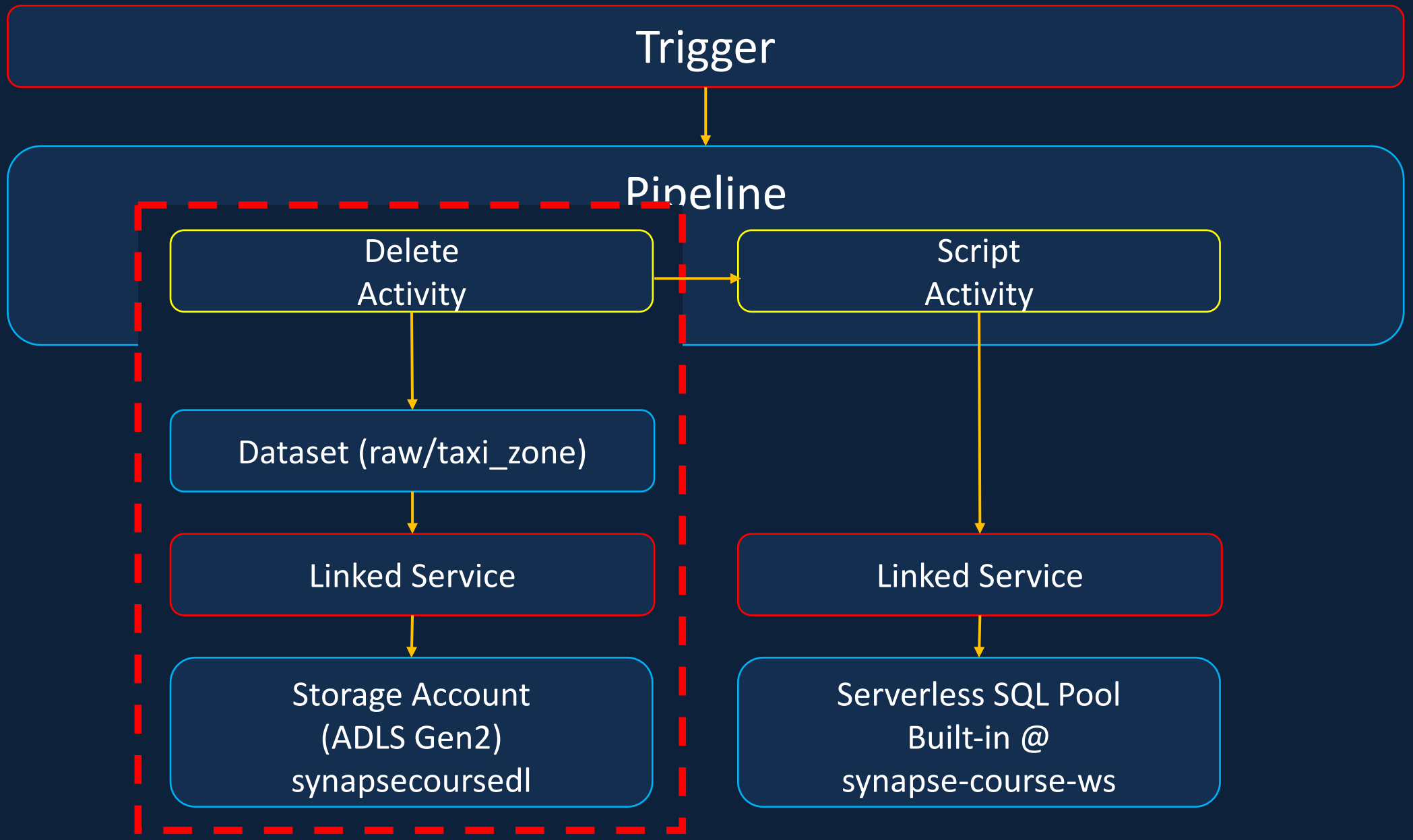
Script Activity



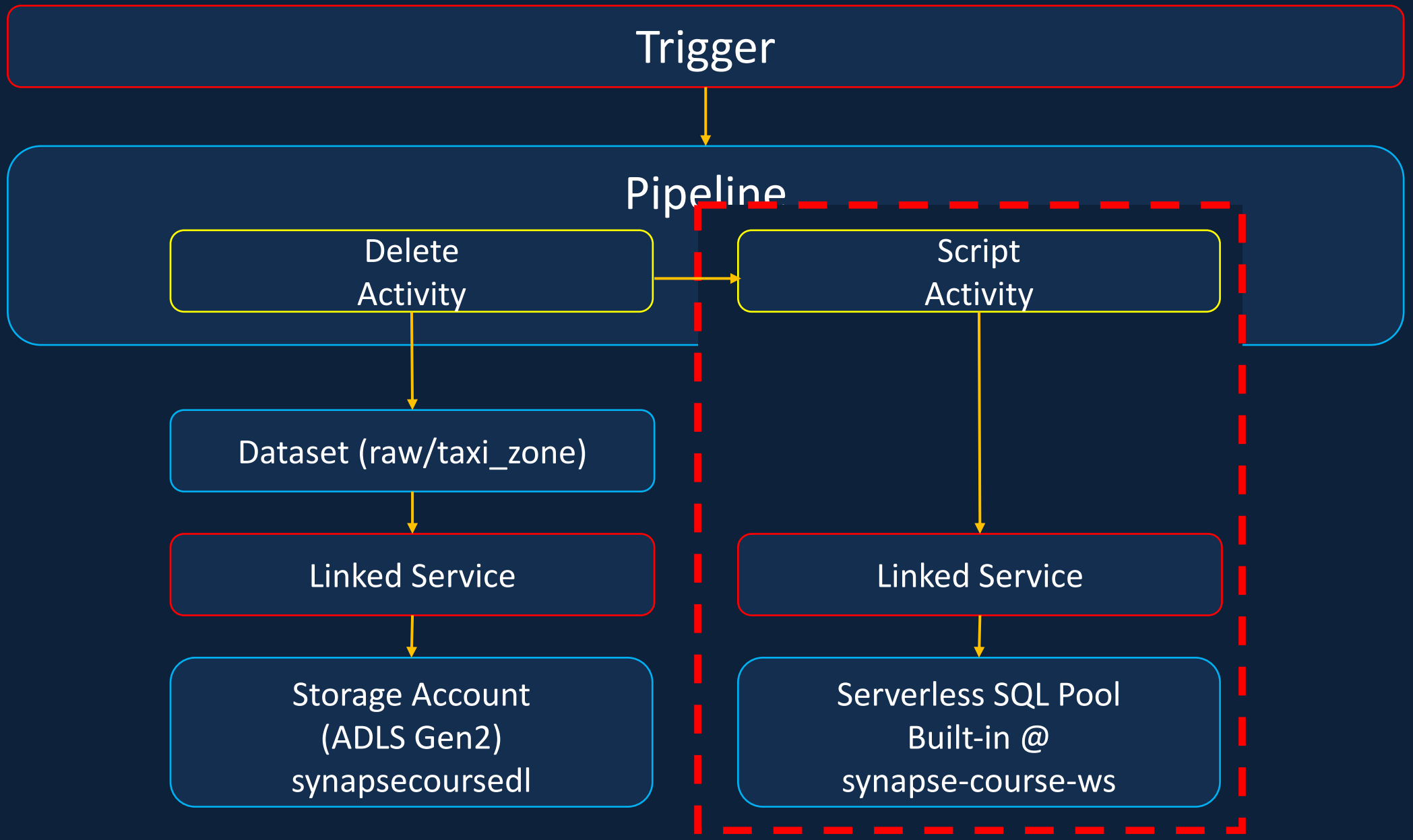
Script Activity



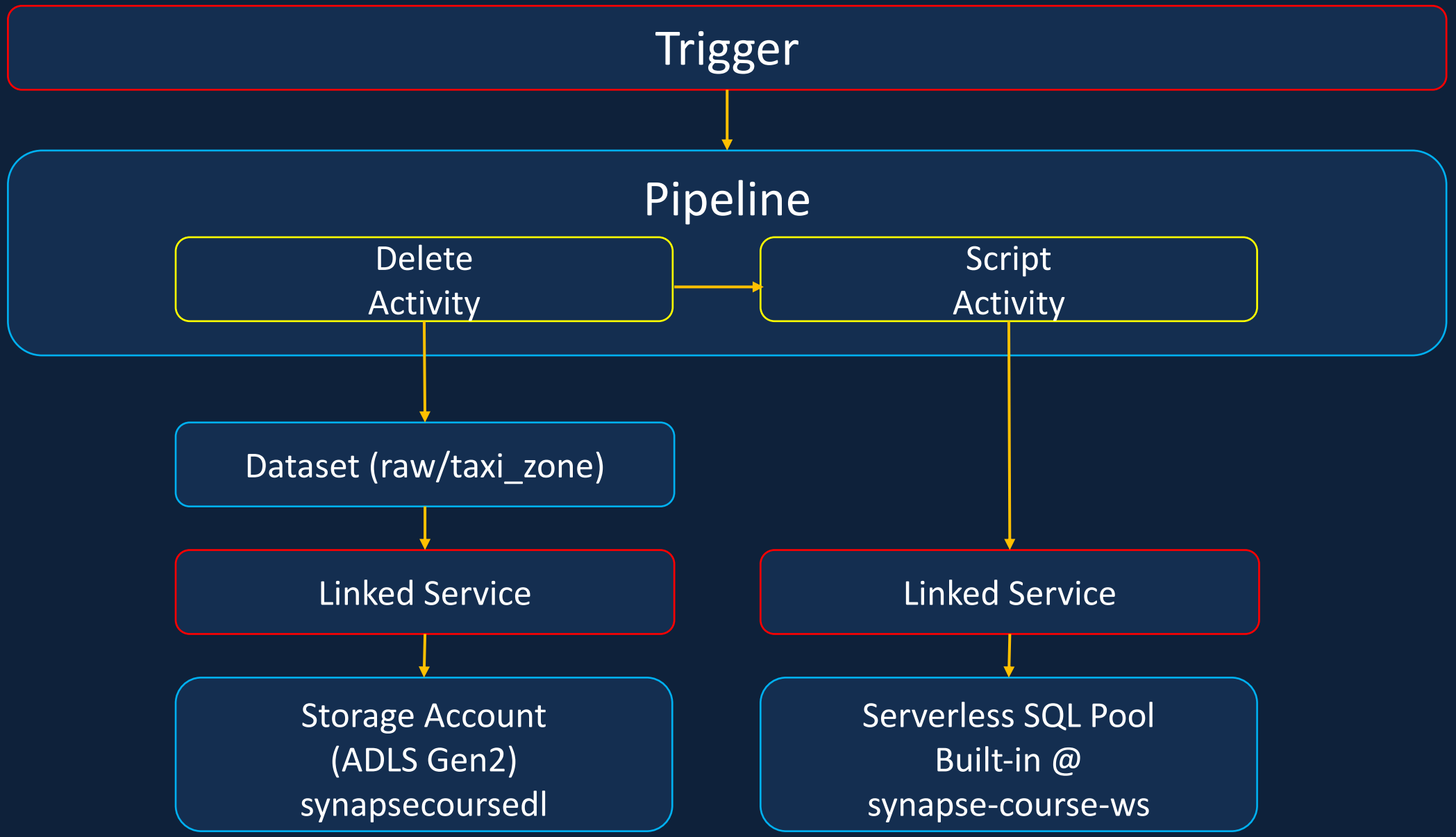
Transformation Pipeline – Taxi Zone



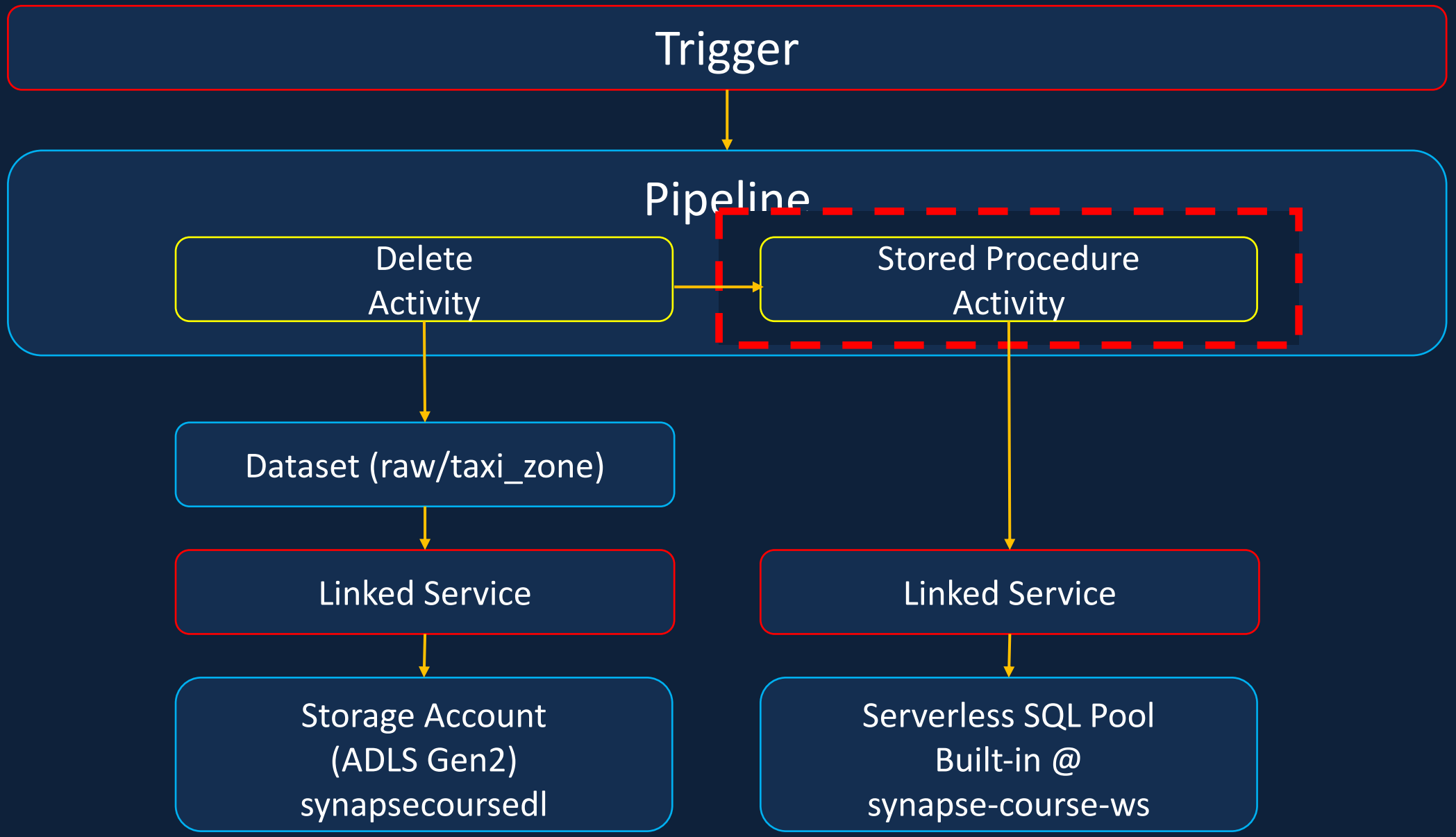
Transformation Pipeline – Taxi Zone



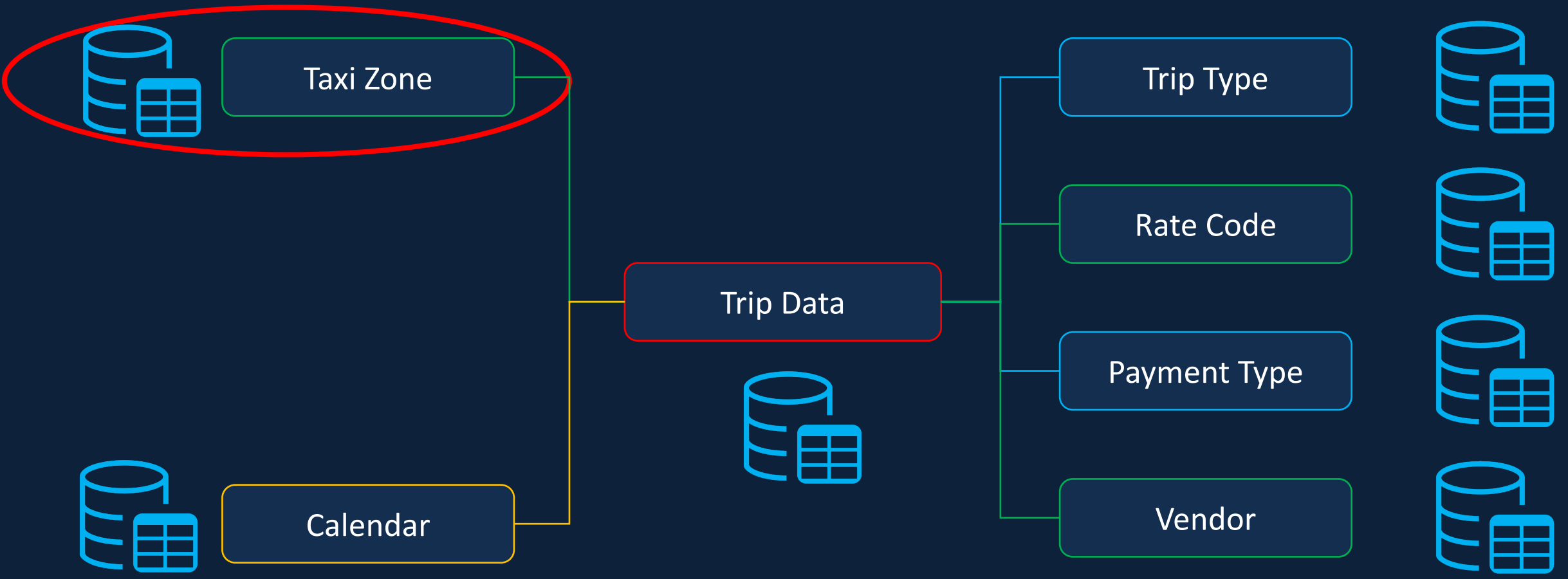
Transformation Pipeline – Taxi Zone



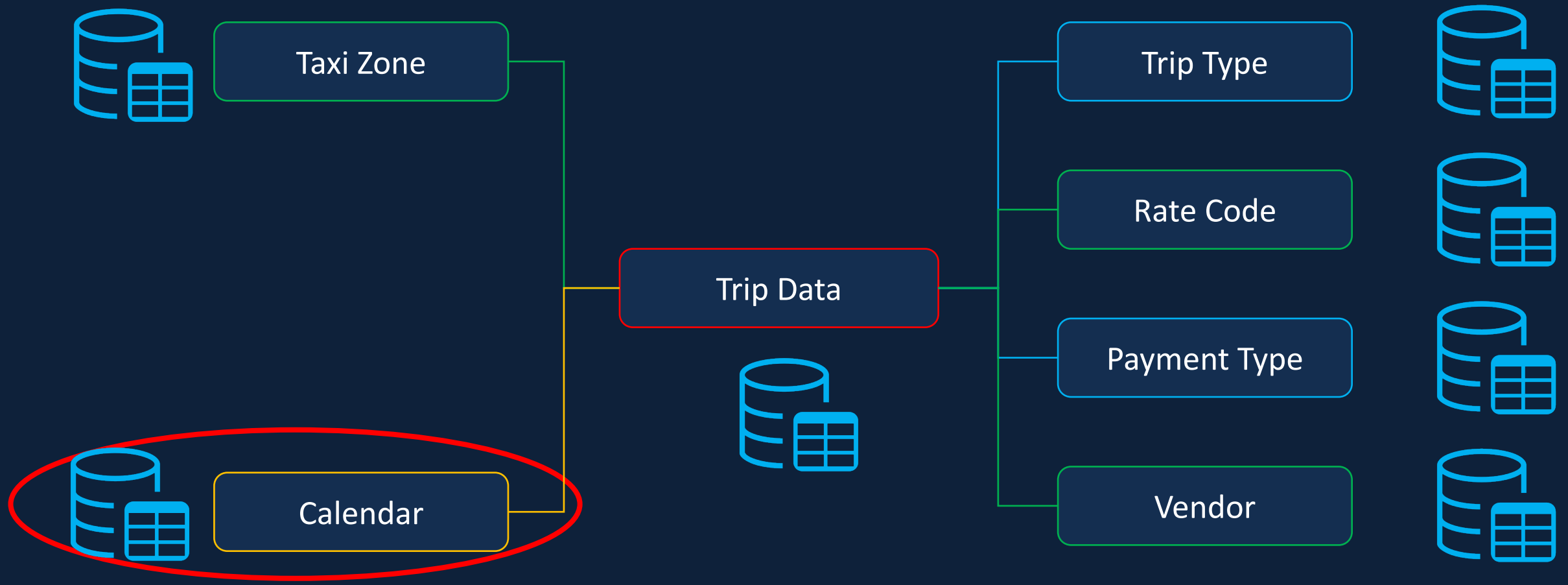
Transformation Pipeline – Taxi Zone



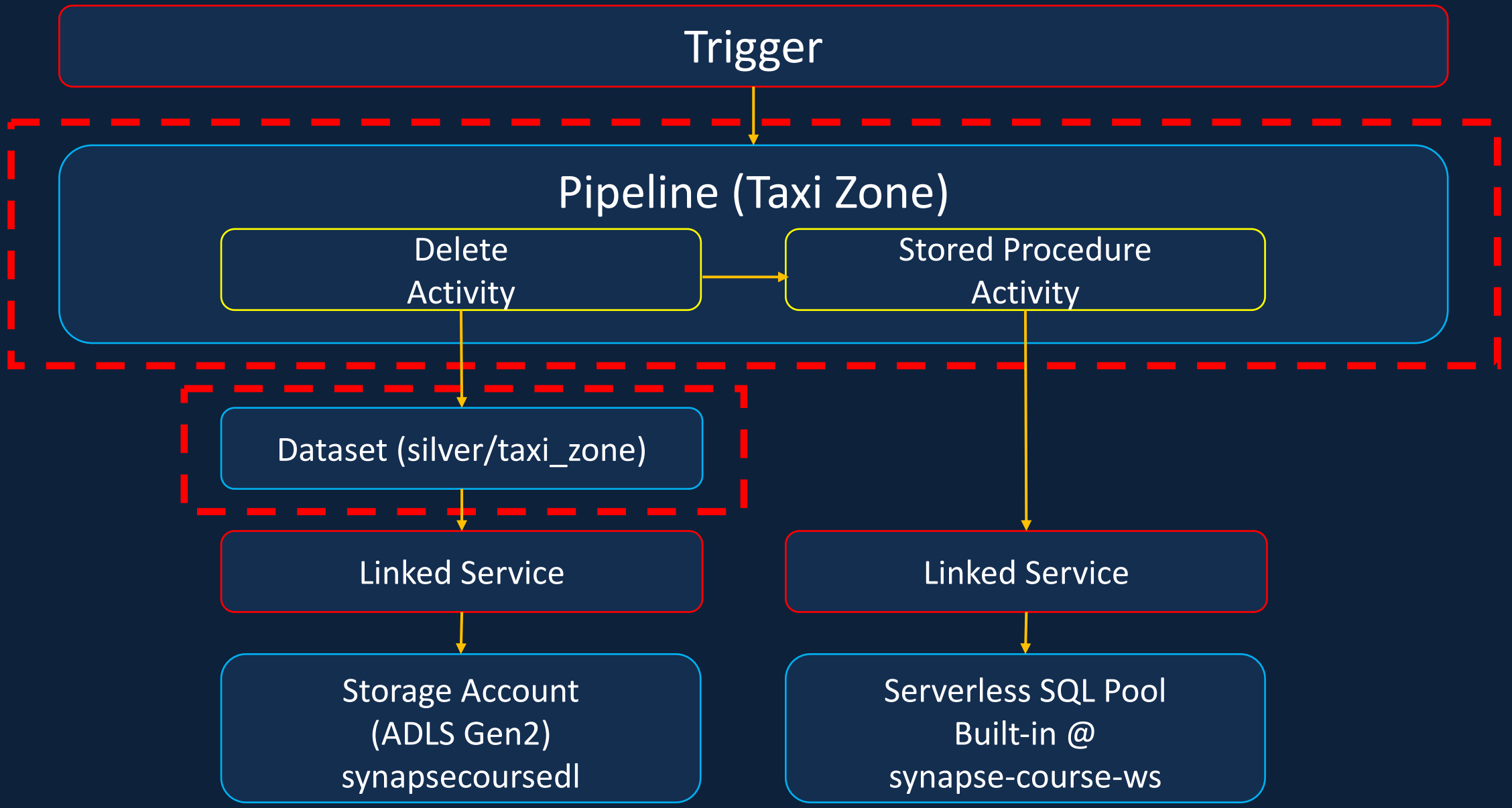
Bronze to Silver Layer Transformation



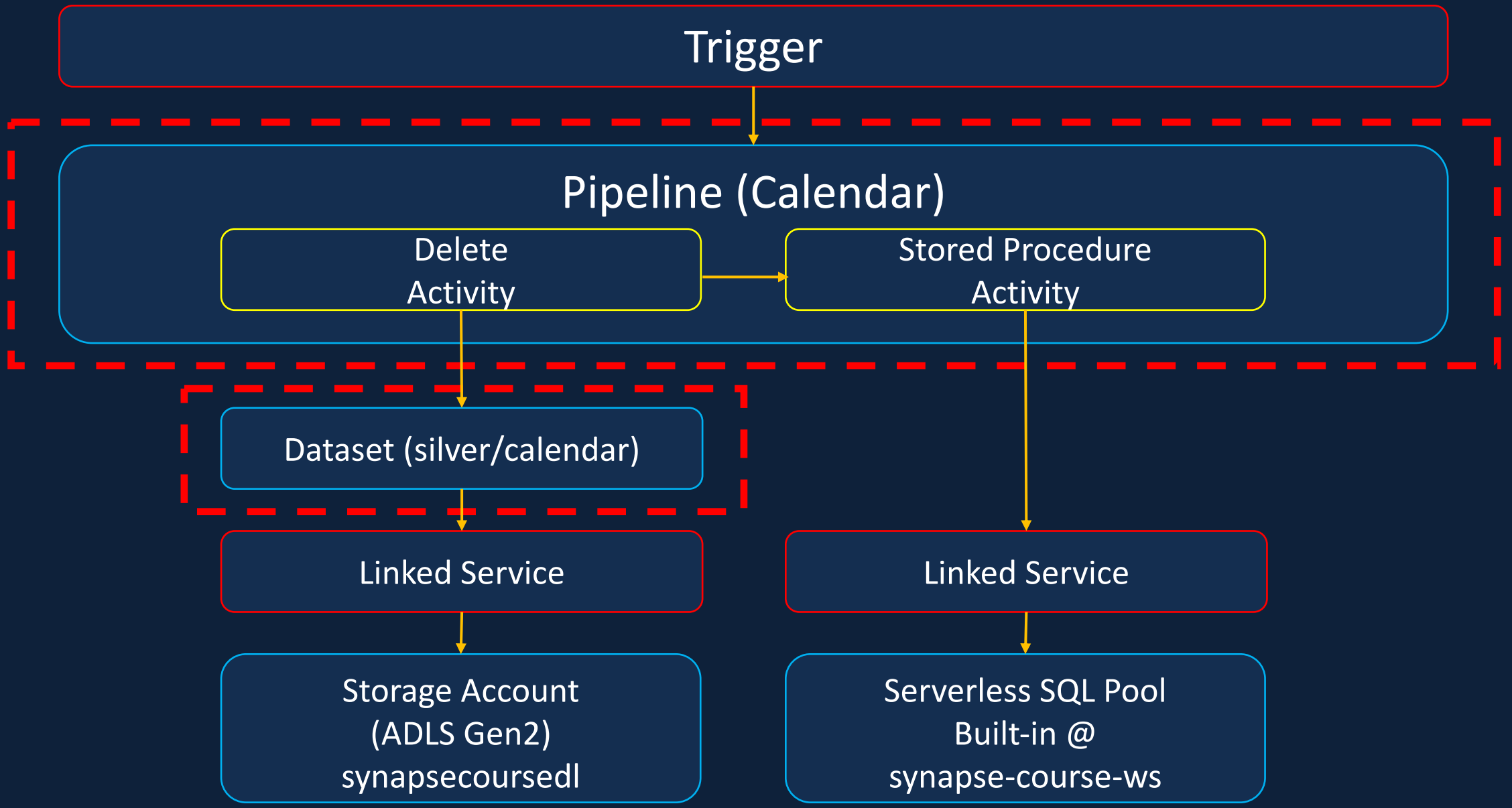
Bronze to Silver Layer Transformation



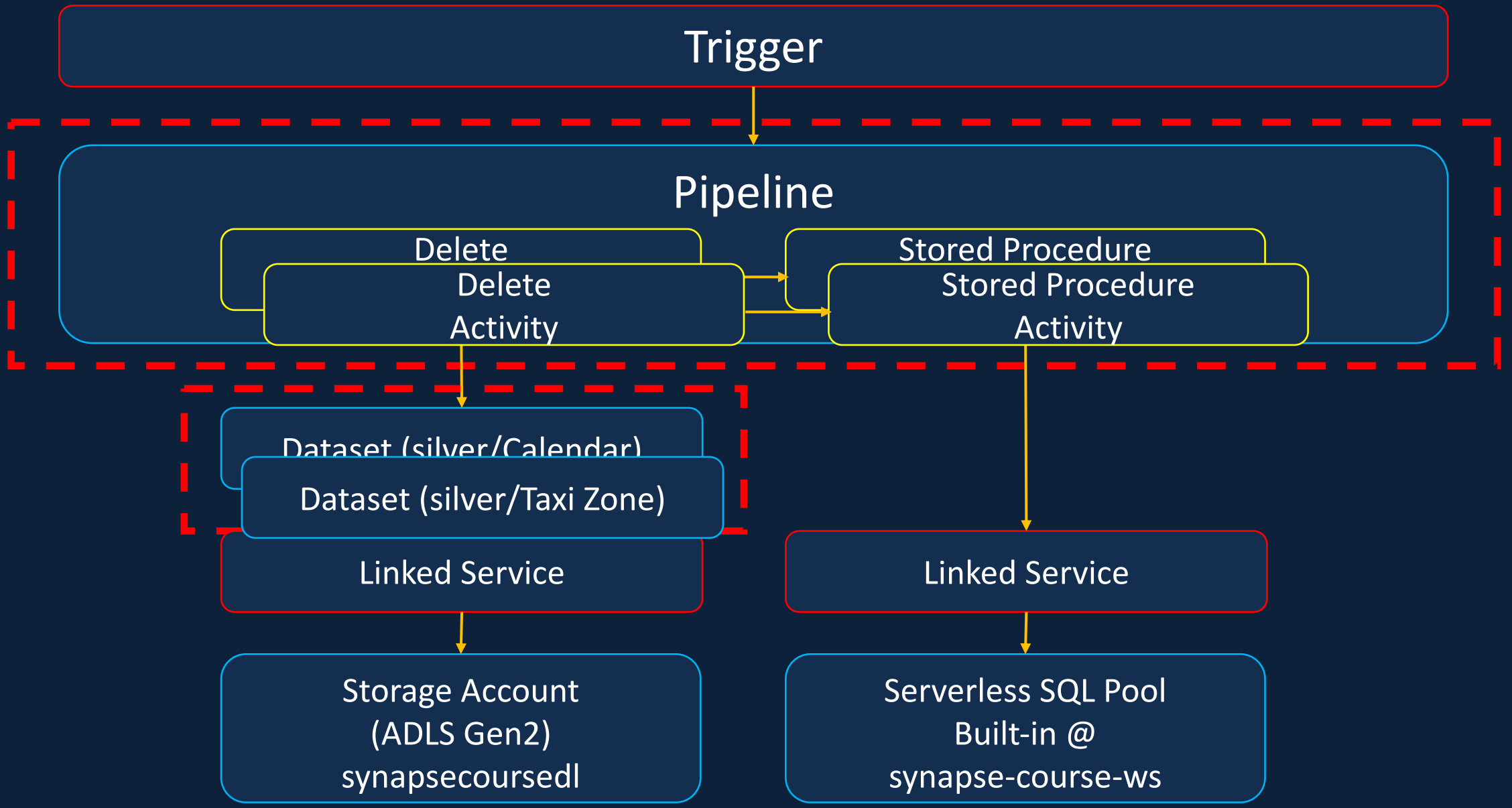
Transformation Pipeline – Taxi Zone



Transformation Pipeline – Calendar



Transformation Pipeline



Dynamic Pipeline - Parameters

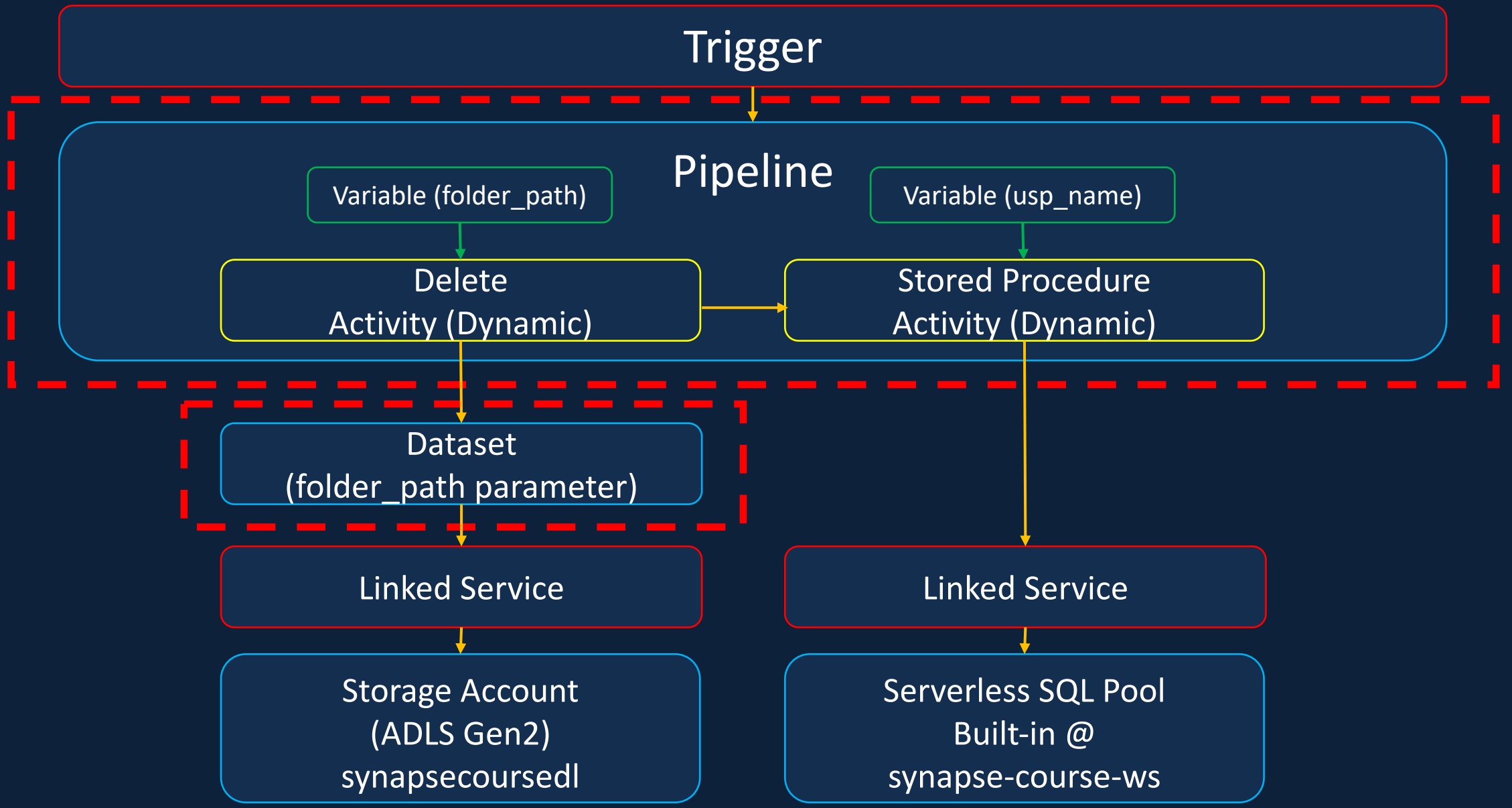
Available in Pipelines, Datasets, Linked Services & Data Flows

Pass external values from one component to the other

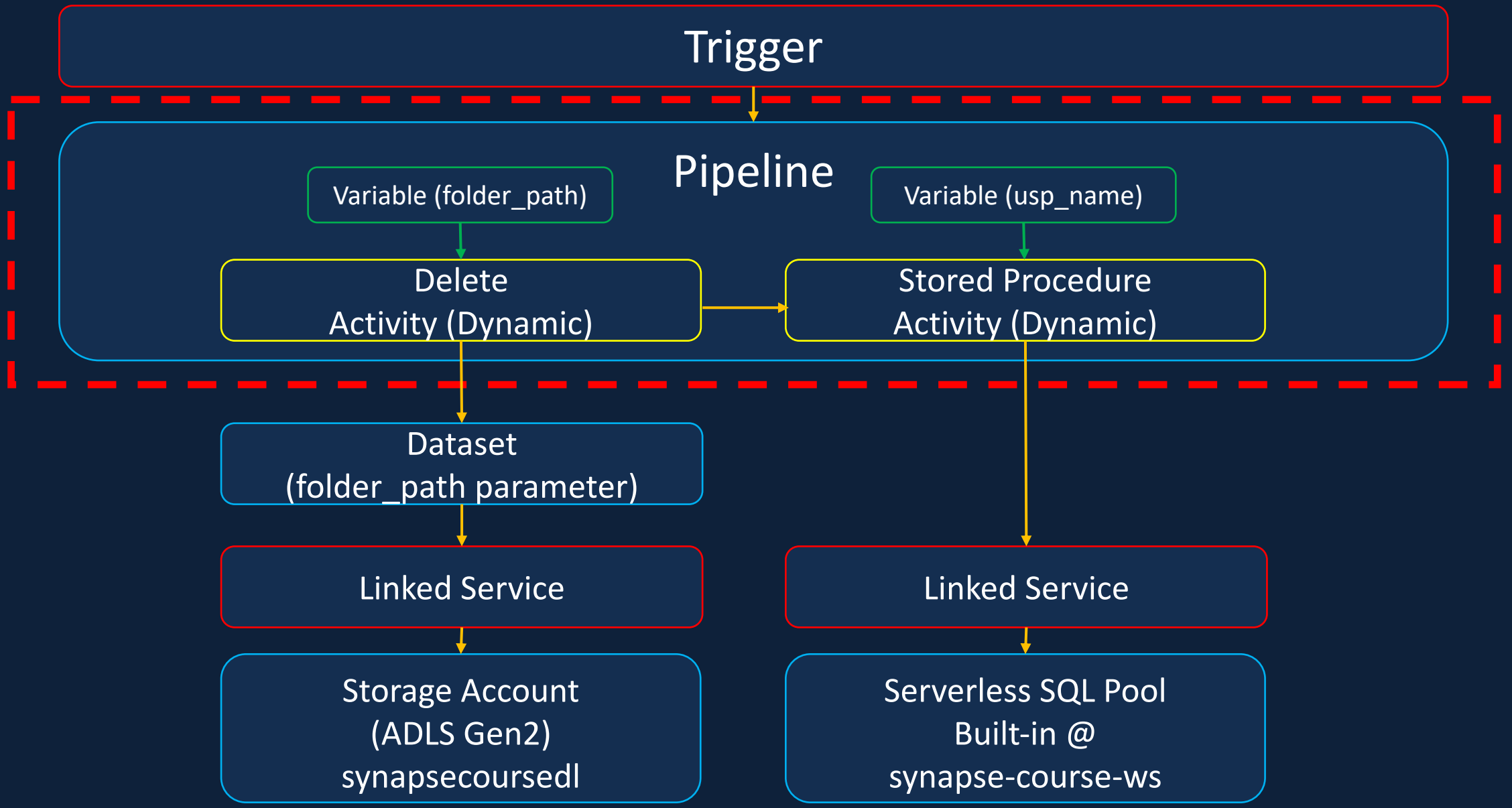
The value cannot be changed inside the component

Unlocks reusability of a component

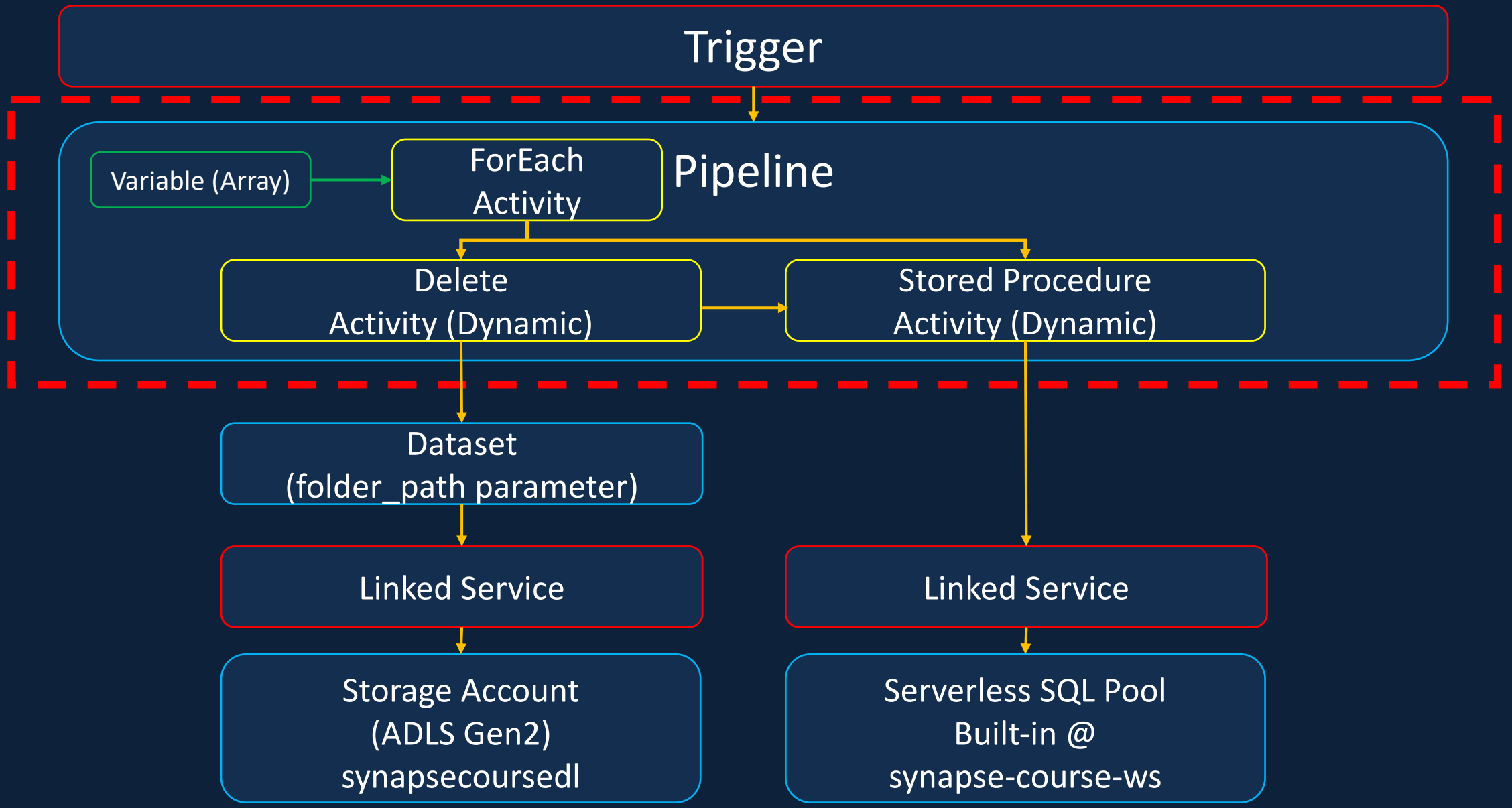
Dynamic Pipeline - Parameters & Variables



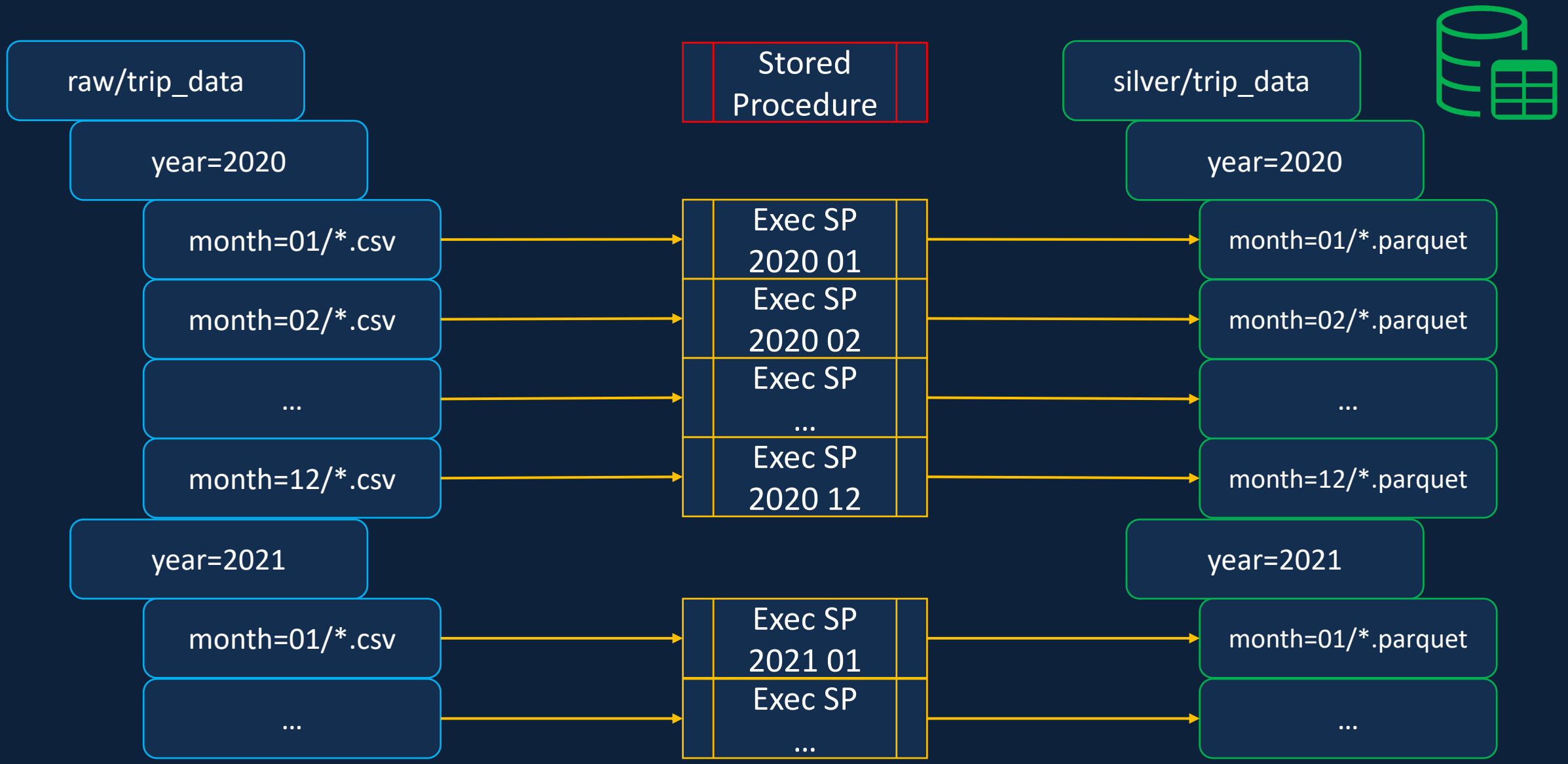
Dynamic Pipeline - Parameters & Variables



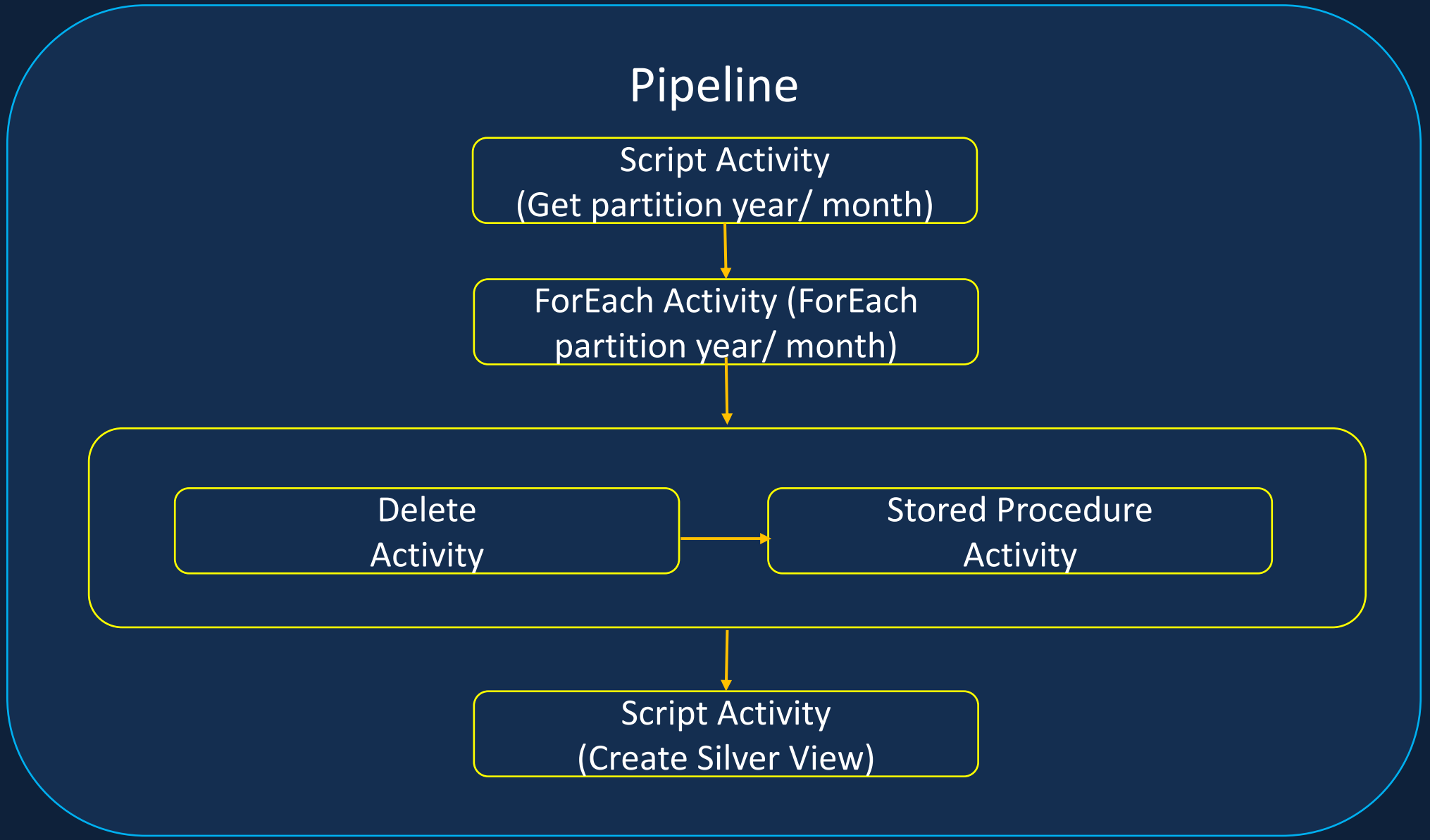
Dynamic Pipeline - Parameters & Variables



Transform to Parquet Format – Partitioned Files



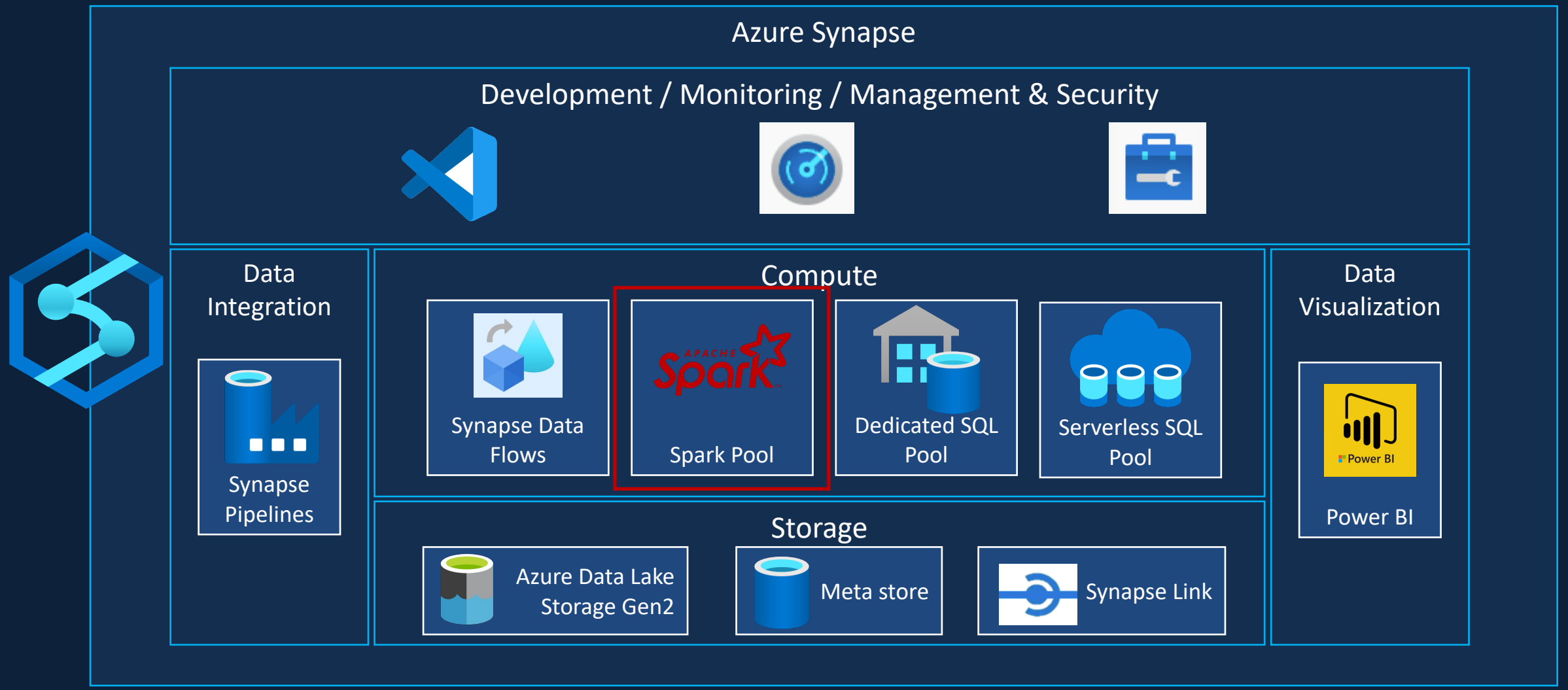
Transform to Parquet Format – Partitioned Files



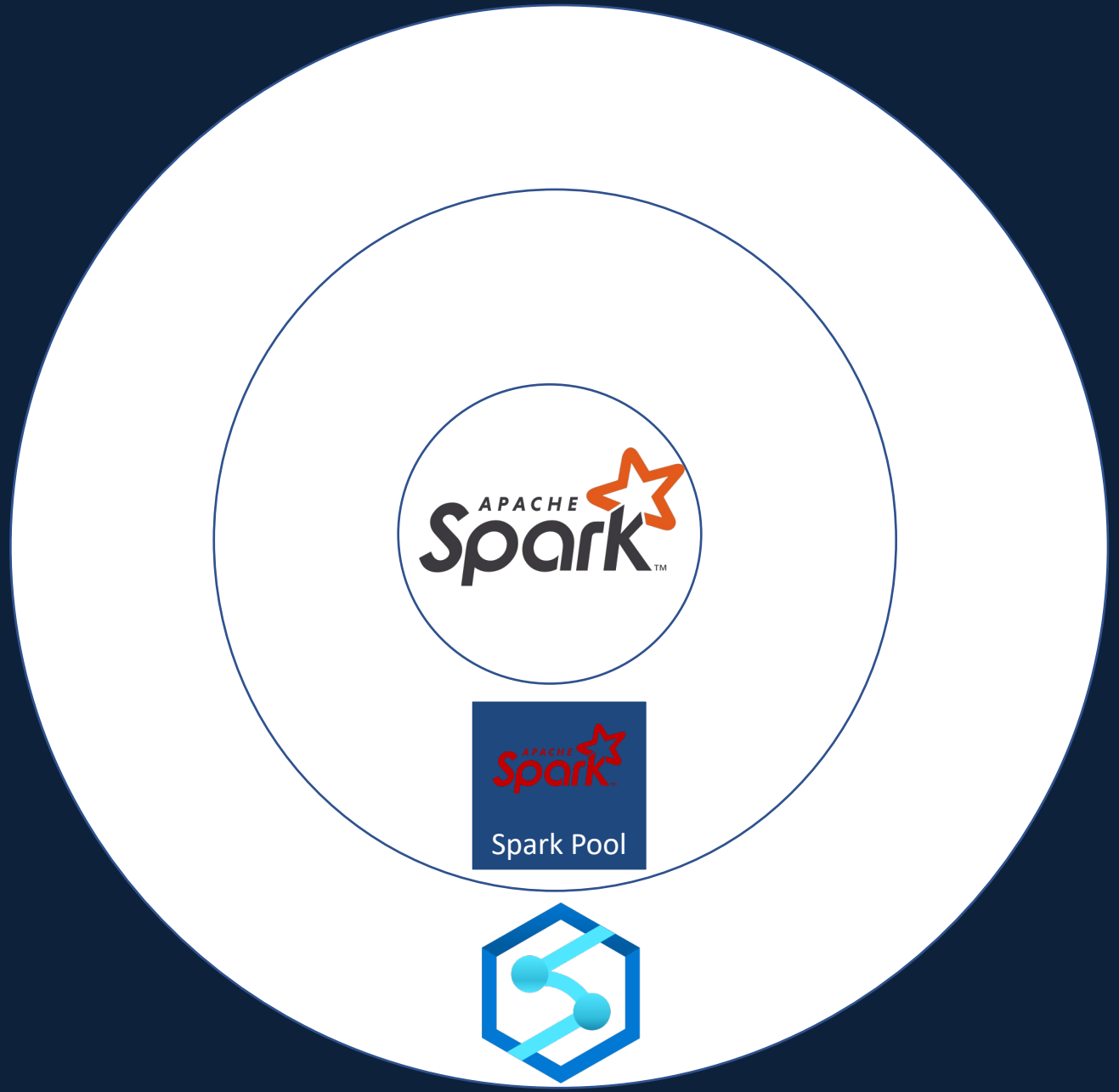
Section Overview – Spark Pool

- Spark Pool Overview
- Create Spark Pool
- Notebooks Overview
- Integration with Serverless SQL Pool
- Create Spark table
- Create Synapse Pipeline

Azure Synapse Analytics – Spark Pool



Azure Synapse Analytics – Spark Pool



Apache Spark

Apache Spark is a lightning-fast unified analytics engine for big data processing and machine learning

- 100% Open source under Apache License
- Simple and easy to use APIs
- In-memory processing engine
- Distributed computing Platform
- Unified engine which supports SQL, streaming, ML and graph processing

Apache Spark Architecture

Spark SQL

Spark Streaming

Spark ML

Spark Graph

DataFrame / Dataset APIs

Spark SQL Engine

Catalyst Optimizer

Tungsten

Spark Core

Scala

Python

Java

R

Resilient Distributed Dataset (RDD)

Spark Standalone, YARN, Apache Mesos, Kubernetes

Azure Synapse Analytics – Spark Pool

Ease of pool creation

Use of notebooks

Delta Lake

Scalability

Pre-loaded libraries

Integration with 3rd party IDEs

Support for C#

Integration with Serverless SQL Pool

Azure Synapse Analytics – Spark Pool Use Cases

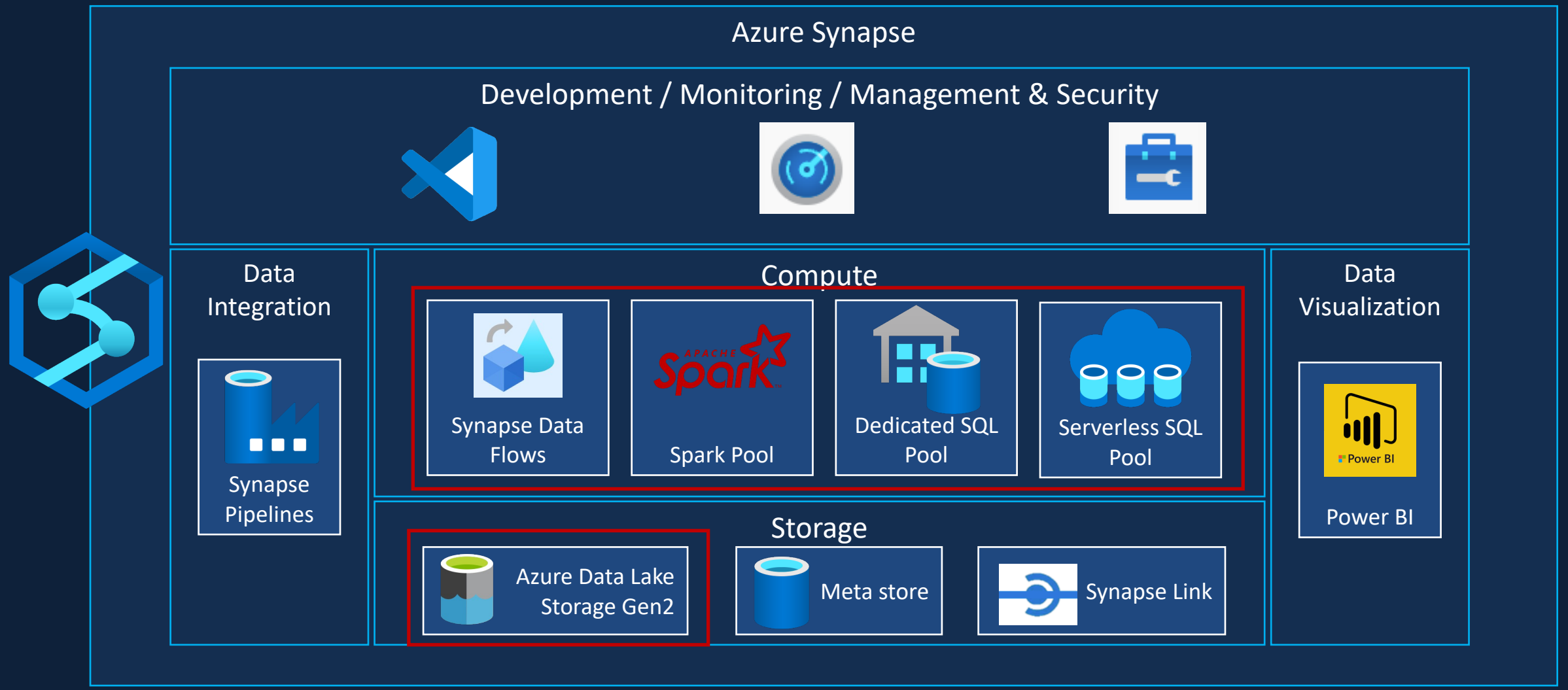
Data Preparation/ Data Transformation

Machine Learning

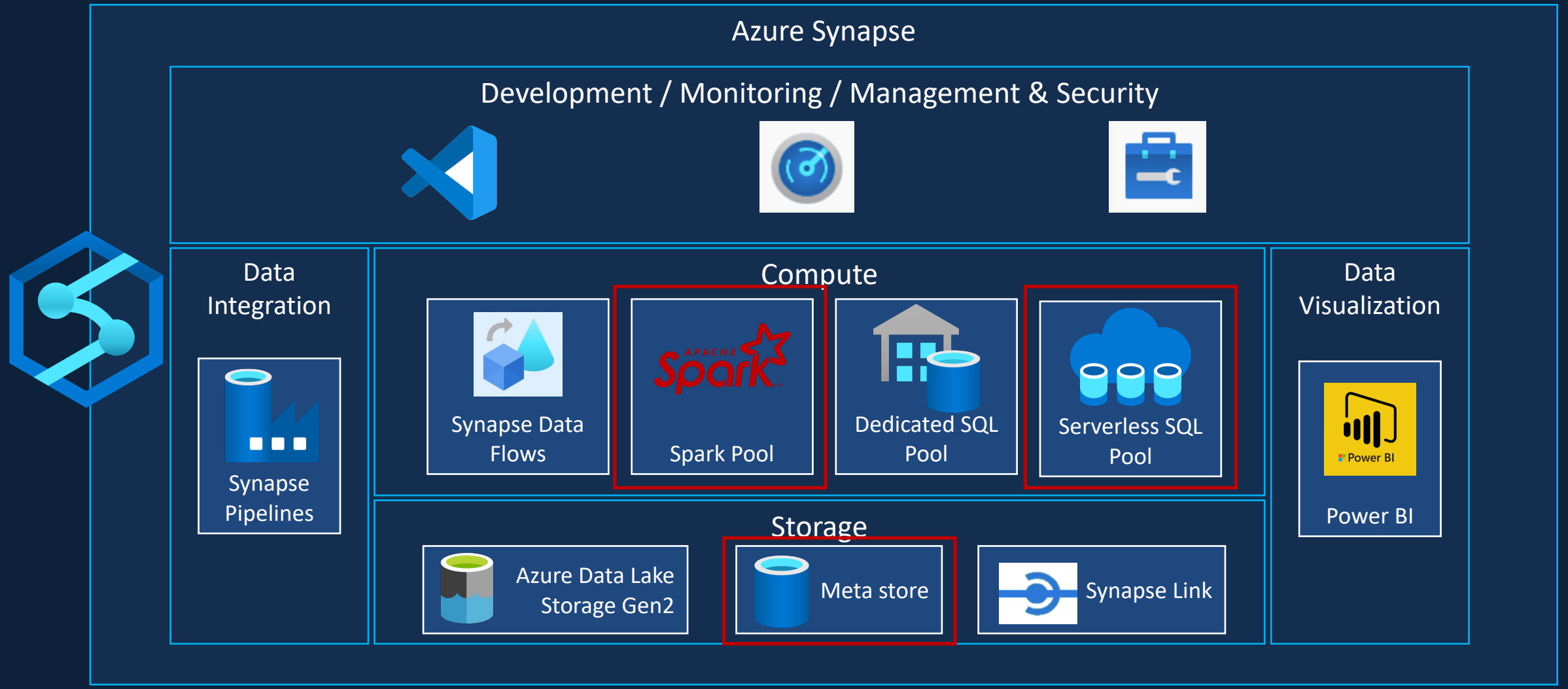
Create Spark Pool Lab

Notebooks Overview Lab

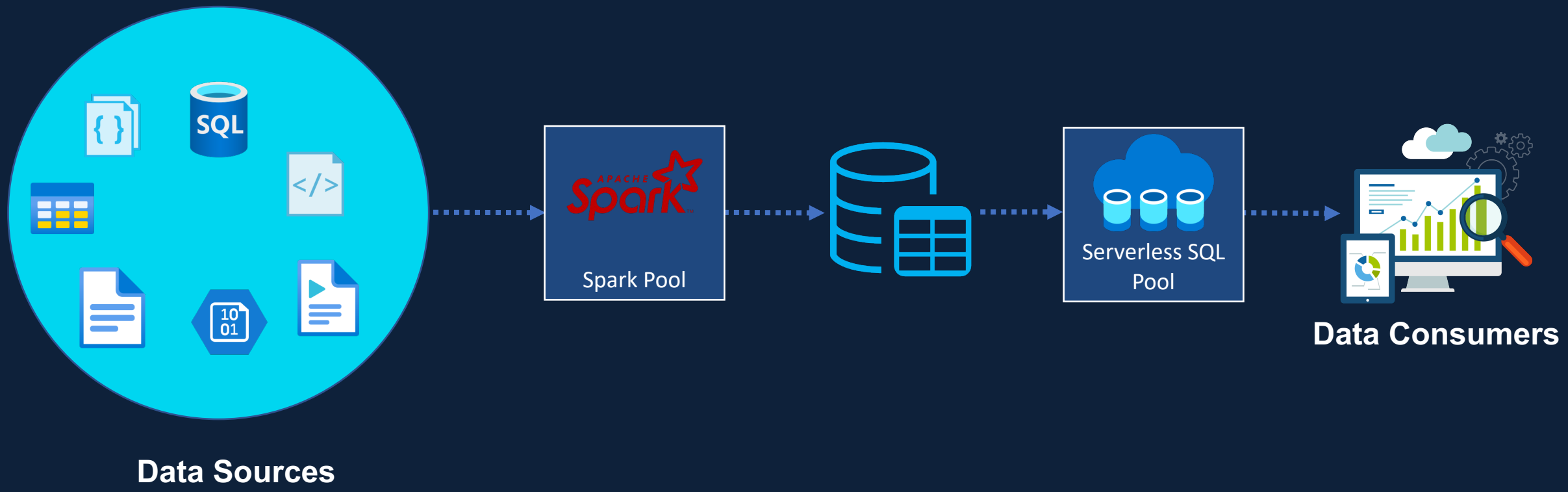
Azure Synapse Analytics



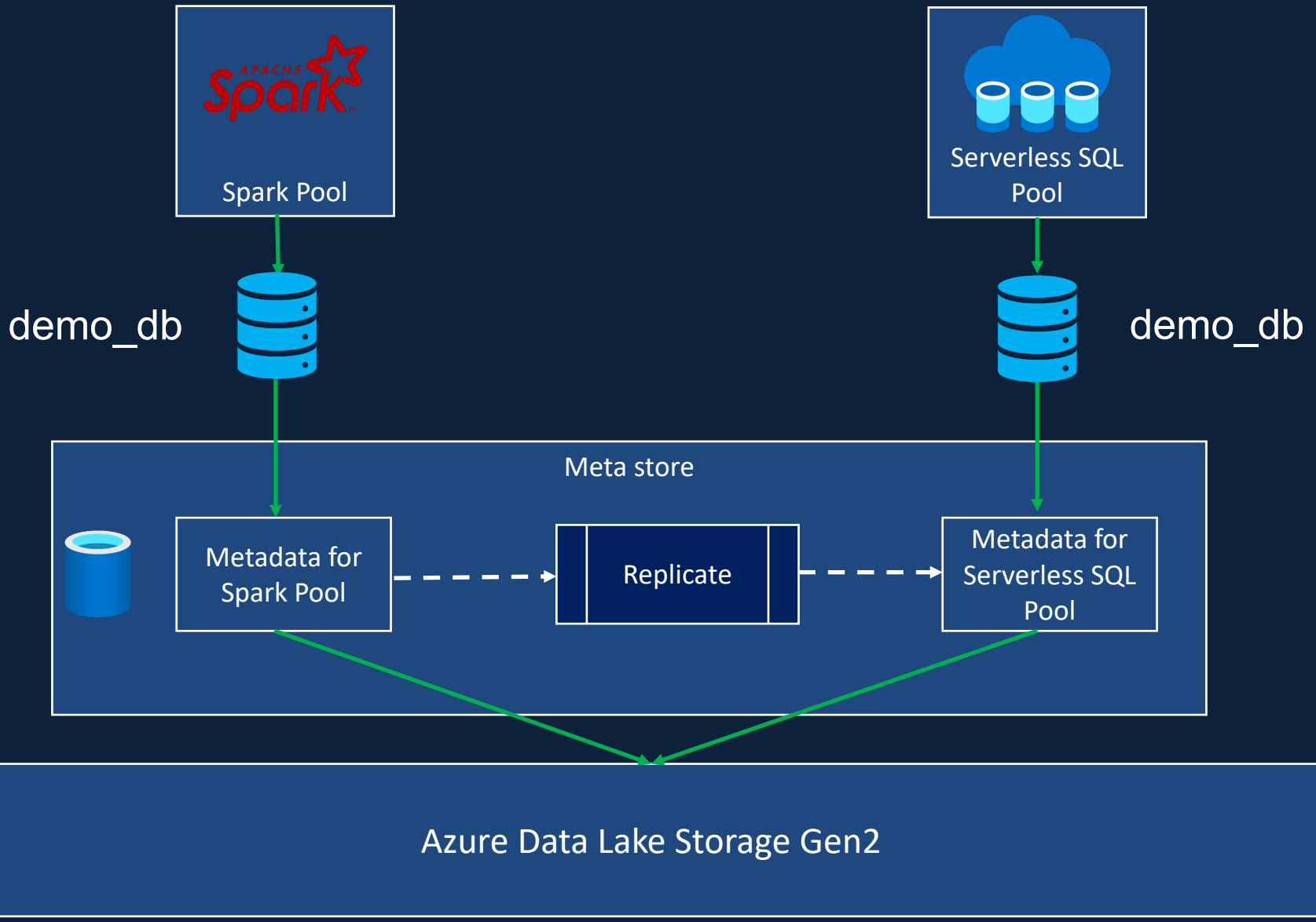
Azure Synapse Analytics



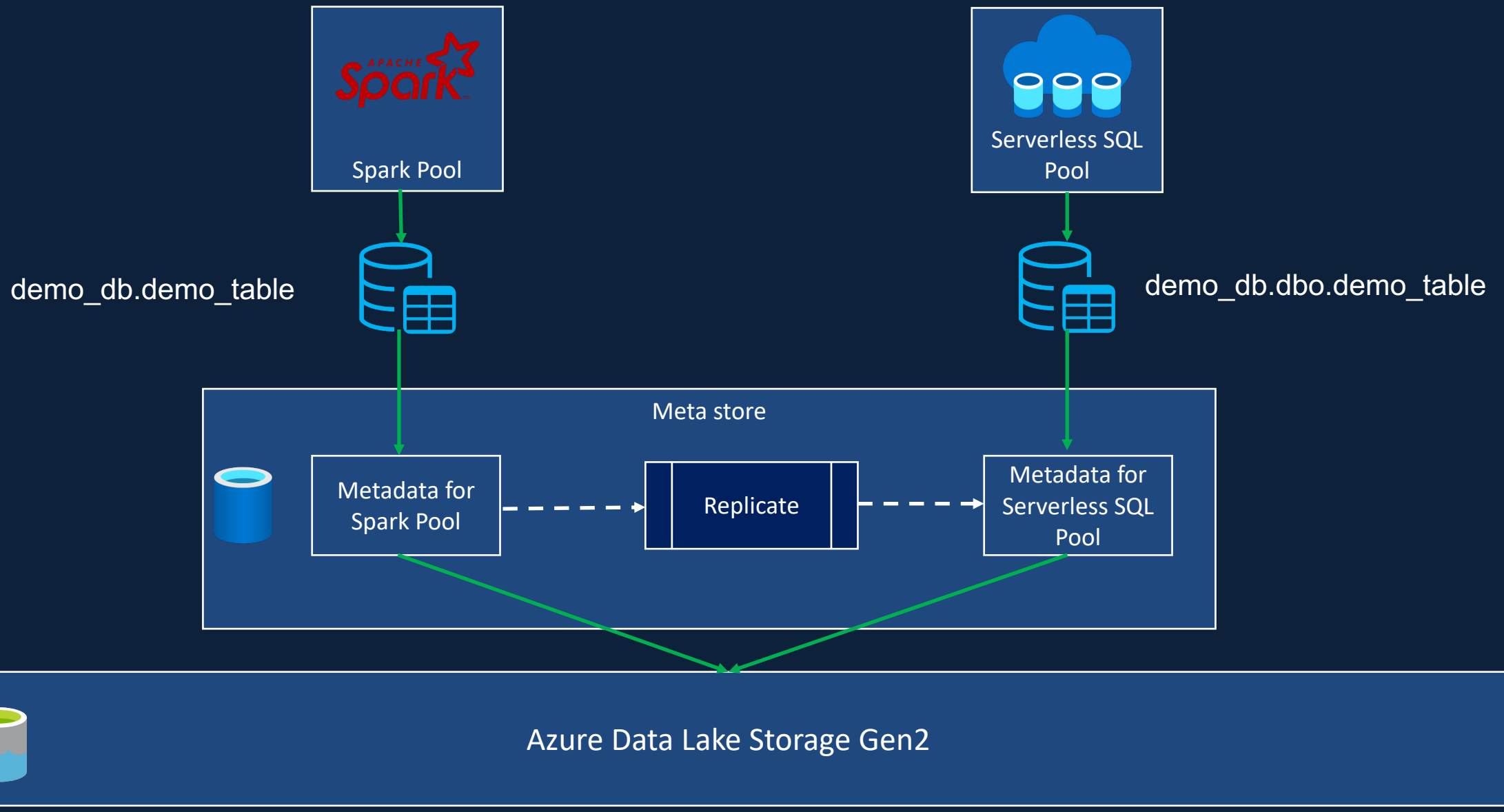
Azure Synapse Analytics



Metadata Replication - Database



Metadata Replication - Table



Metadata Replication

Metadata replicated asynchronously

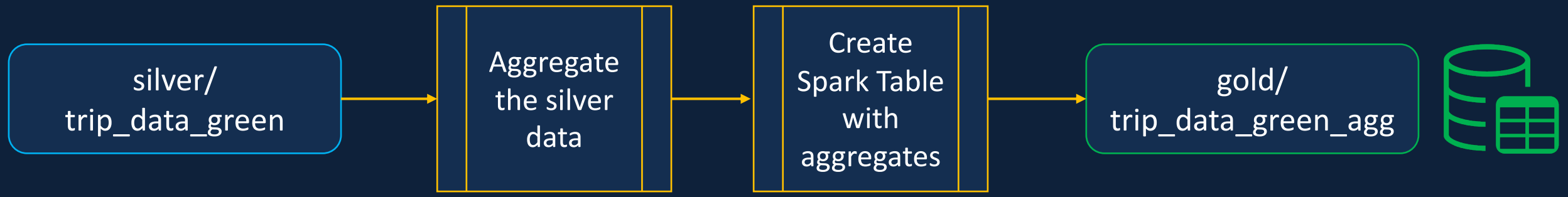
Supports only Parquet & CSV backed tables

Replicated tables cannot be updated by Serverless SQL Pool

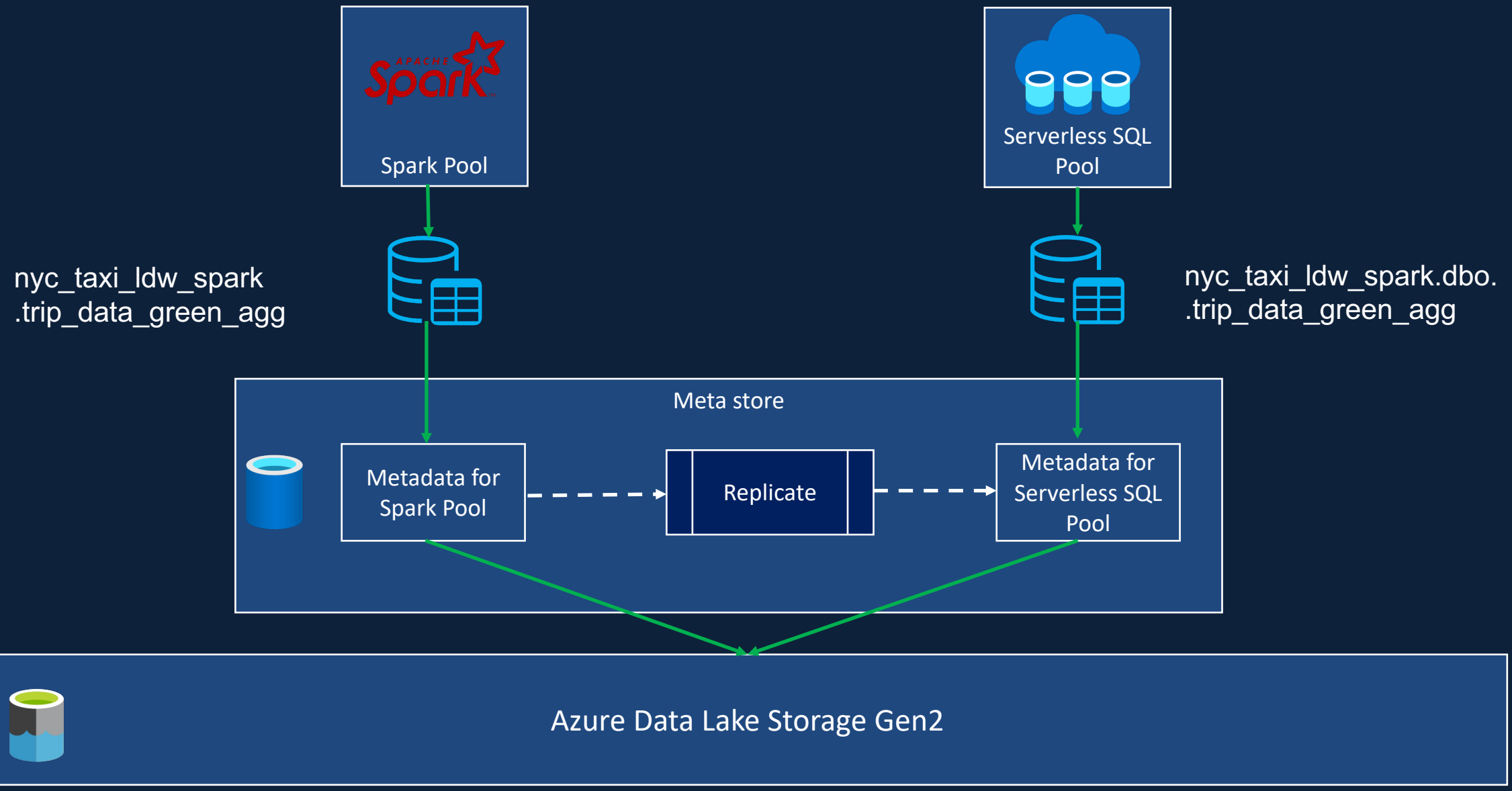
Secured at the underlying storage level

Database names have to be unique across Spark Pools

Integration Between Spark Pool & Serverless SQL Pool



Integration Between Spark Pool & Serverless SQL Pool

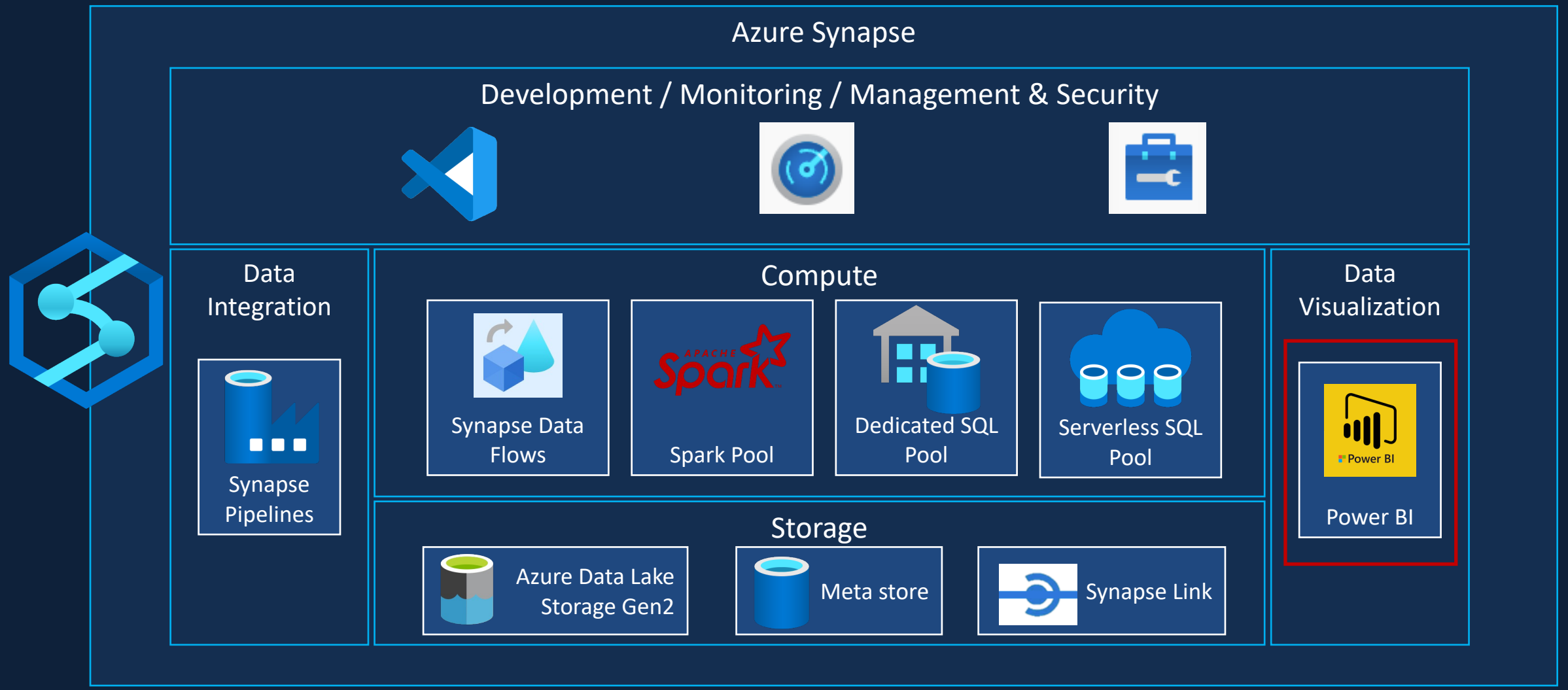


Synapse – Power BI Integration

Section Overview – Power BI Integration

- Power BI Integration Overview
- Connecting from Power BI Desktop
- Publishing to Power BI Workspace
- Power BI Integration within Synapse Studio
- Create Reports

Synapse – Power BI Integration



Pre-requisites

Power BI Desktop

Power BI Workspace

Power BI Workspace

Azure Active Directory User

Power BI User

Power BI Workspace

Access to the Synapse resource group

Access to Synapse Studio

Access to the Synapse Primary Storage Account

Project Requirements

Business Requirements (1)

Campaign to encourage credit card payments

Trips made using credit card/ cash payments

Payment behaviour during days of the week/ weekend

Payment behaviour between boroughs

Business Requirements (2)

Identify taxi demand

Demand based on borough

Demand based on day of the week/ weekend

Demand based on trip type (i.e., Street hail/ Despatch)

Trip distance, trip duration, total fare amount etc per day/ borough

Section Overview – Synapse Link

- Synapse Link Overview
- Synapse Link for Cosmos DB Overview
- Create Cosmos DB Service
- Query using Serverless SQL Pool
- Query using Spark Pool

Synapse Link

Why do we need Synapse Link?

Online Transactional Processing (OLTP)

Online Analytical Processing (OLAP)

Synapse Link

Online Transactional Processing (OLTP)

Online Analytical Processing (OLAP)

Row Oriented

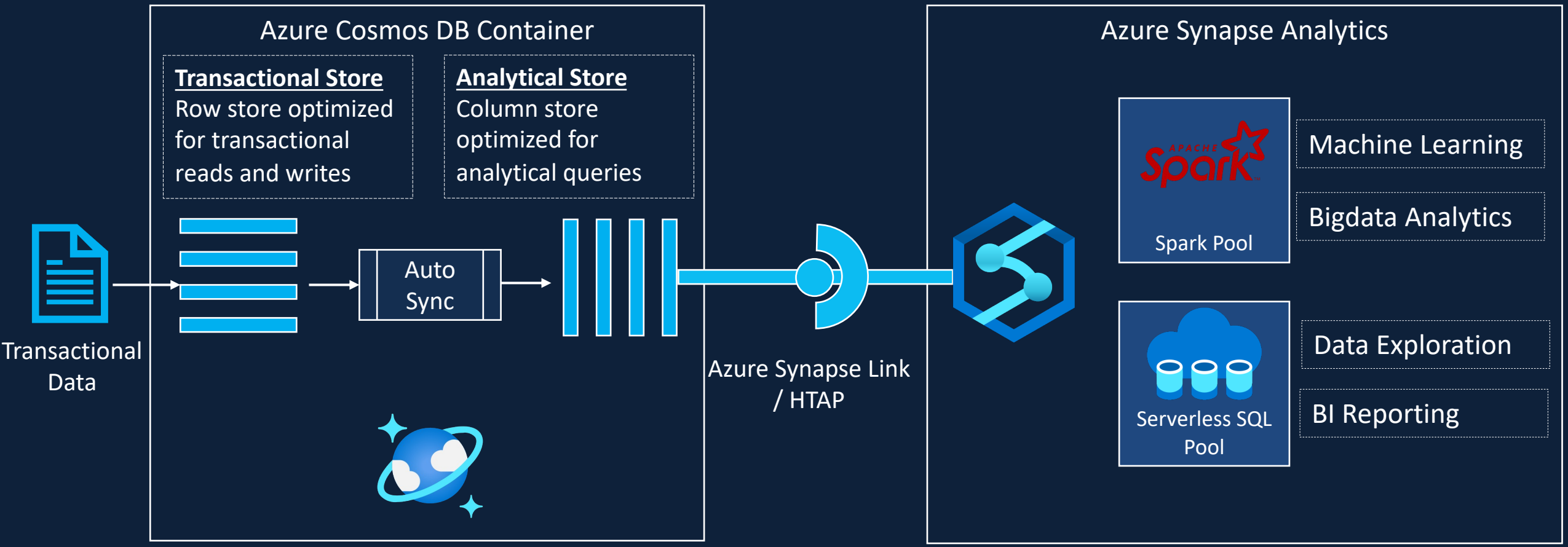
Column Oriented

Cosmos DB - GA

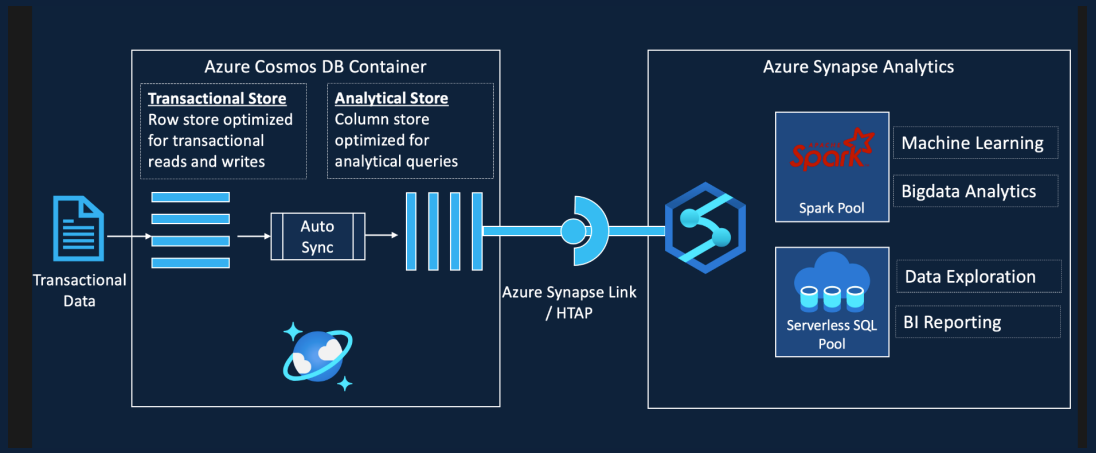
Dataverse - GA

SQL Server 2022 – Private Preview

Synapse Link for Cosmos DB

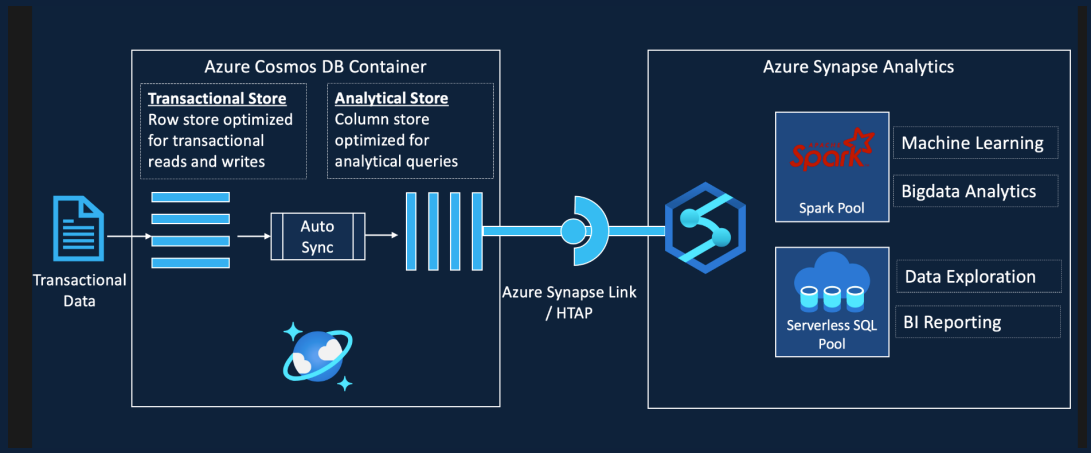


Synapse Link for Cosmos DB - Benefits



- Less Complex Solution
- Fully managed service
- Near Real Time Analytics
- No impact to Transactional System
- Automatic Schema Inference
- Native Integration between Synapse and Cosmos

Synapse Link for Cosmos DB - Limitations

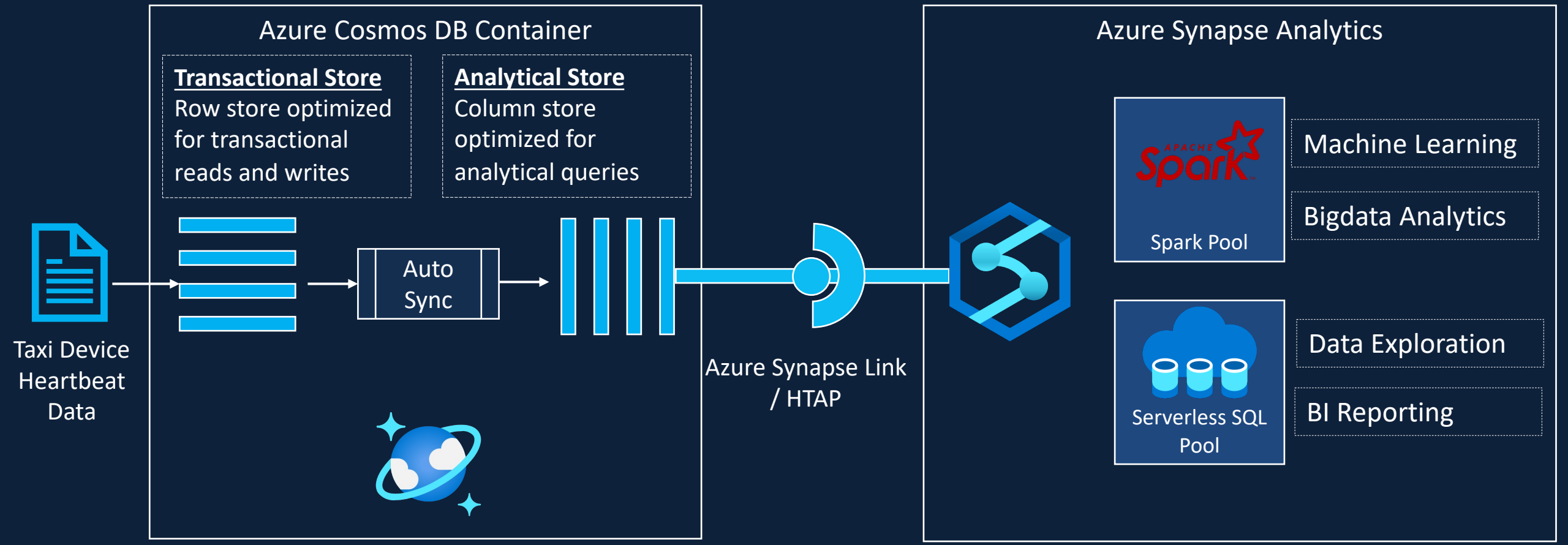


- Only SQL API and Mongo DB API supported
- Dedicated SQL Pool – Not Supported
- Limited support for existing Cosmos DB containers
- Backup and Restore of Analytical Store – Not Available
- Custom Partitioning – In Preview

Project Overview

- NYC Taxis are fitted with a device to manage taxi hires
- Device sends data to Cosmos DB every minute
- Create a Cosmos DB analytic store using Synapse Link
- Ability to query/ process the analytic store using Spark Pool
- Ability to query and make data available to PowerBI using Serverless SQL Pool

Project Solution



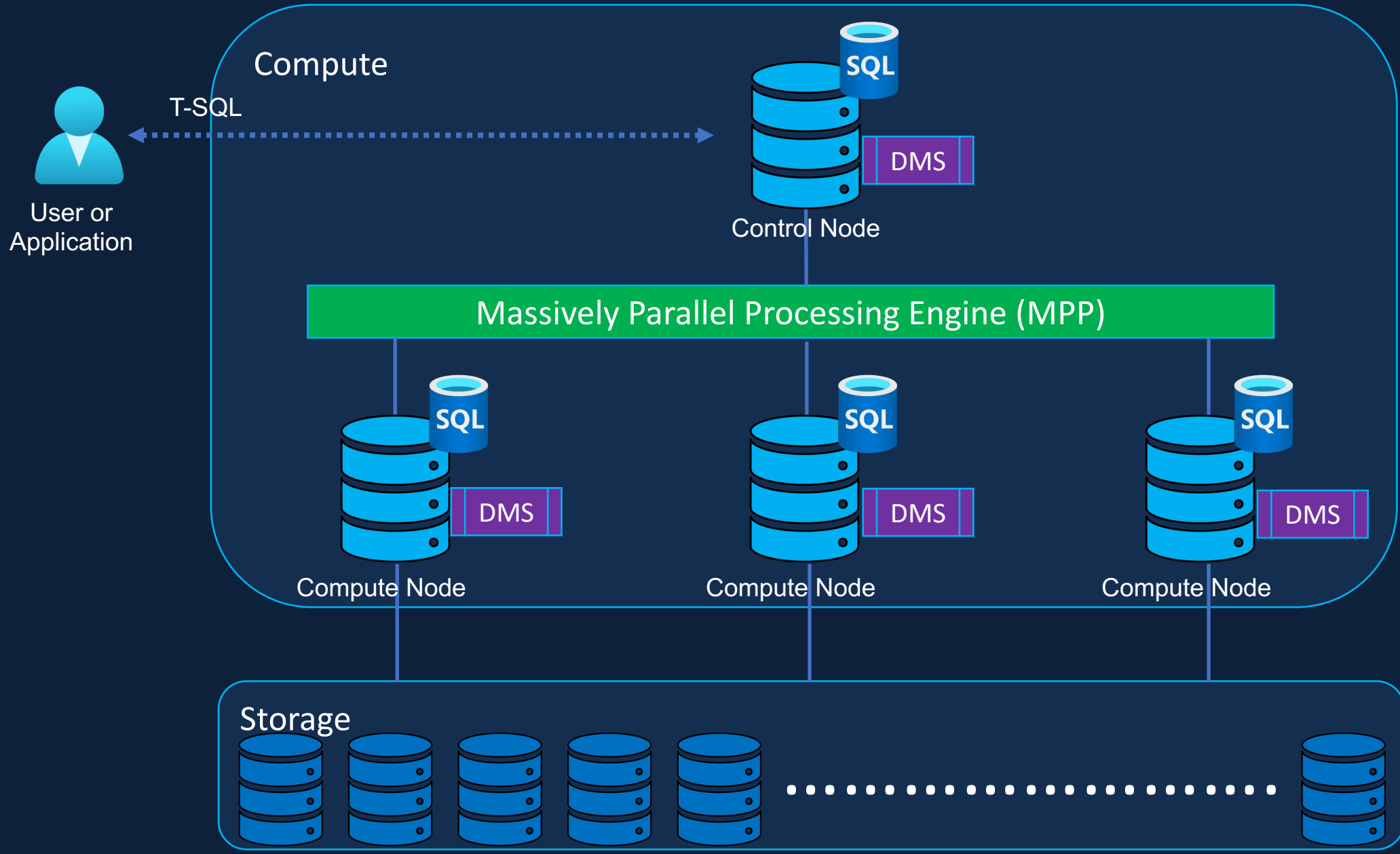
Section Overview – Dedicated SQL Pool

- Overview
- Create Dedicated SQL Pool
- External Tables
- Copy Command
- Connecting from Azure Data Studio
- Connecting from Power BI

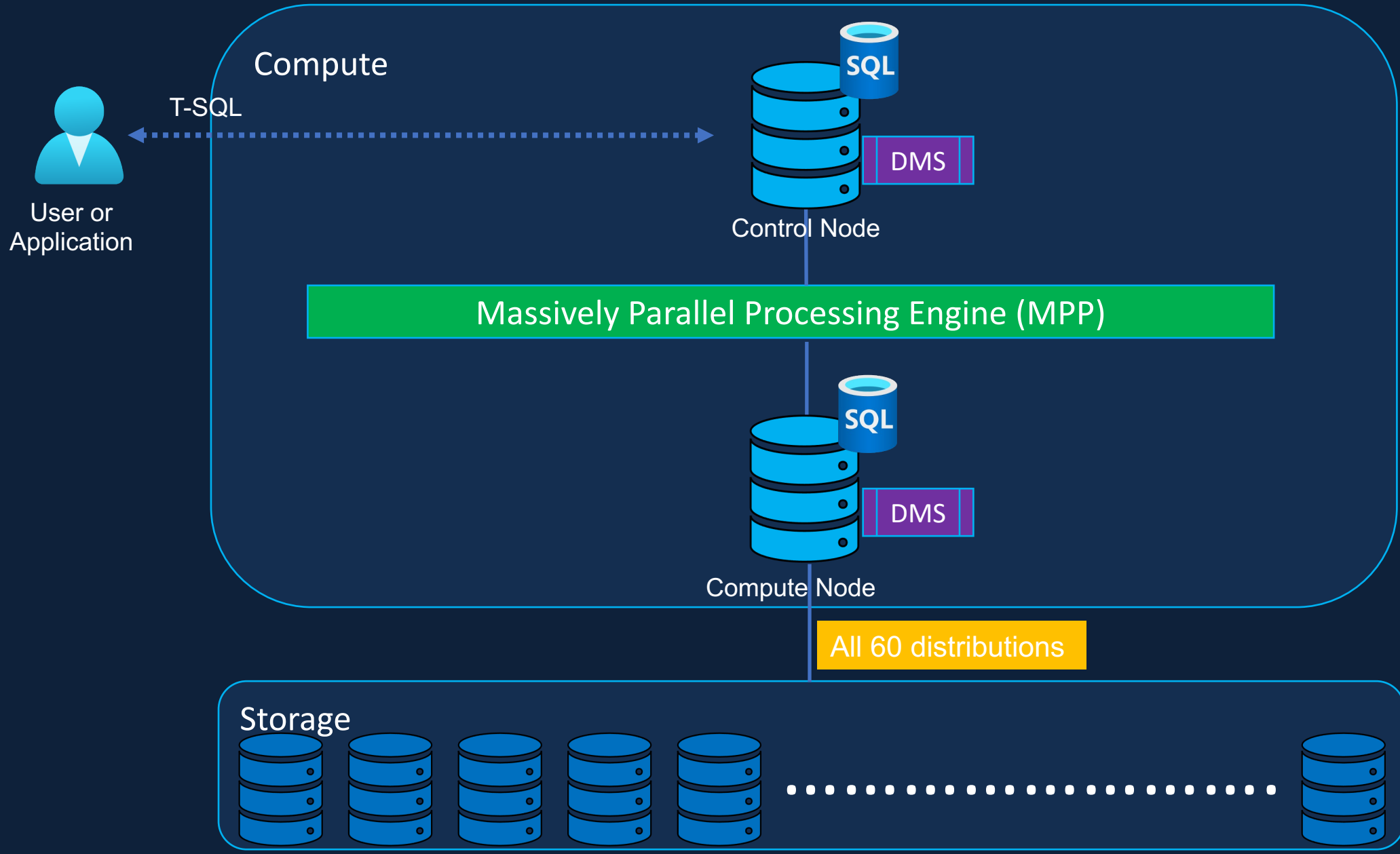
Dedicated SQL Pool

Dedicated SQL pool (formerly SQL DW) is a distributed query engine that you can use to perform high performance big data analytics using T-SQL.

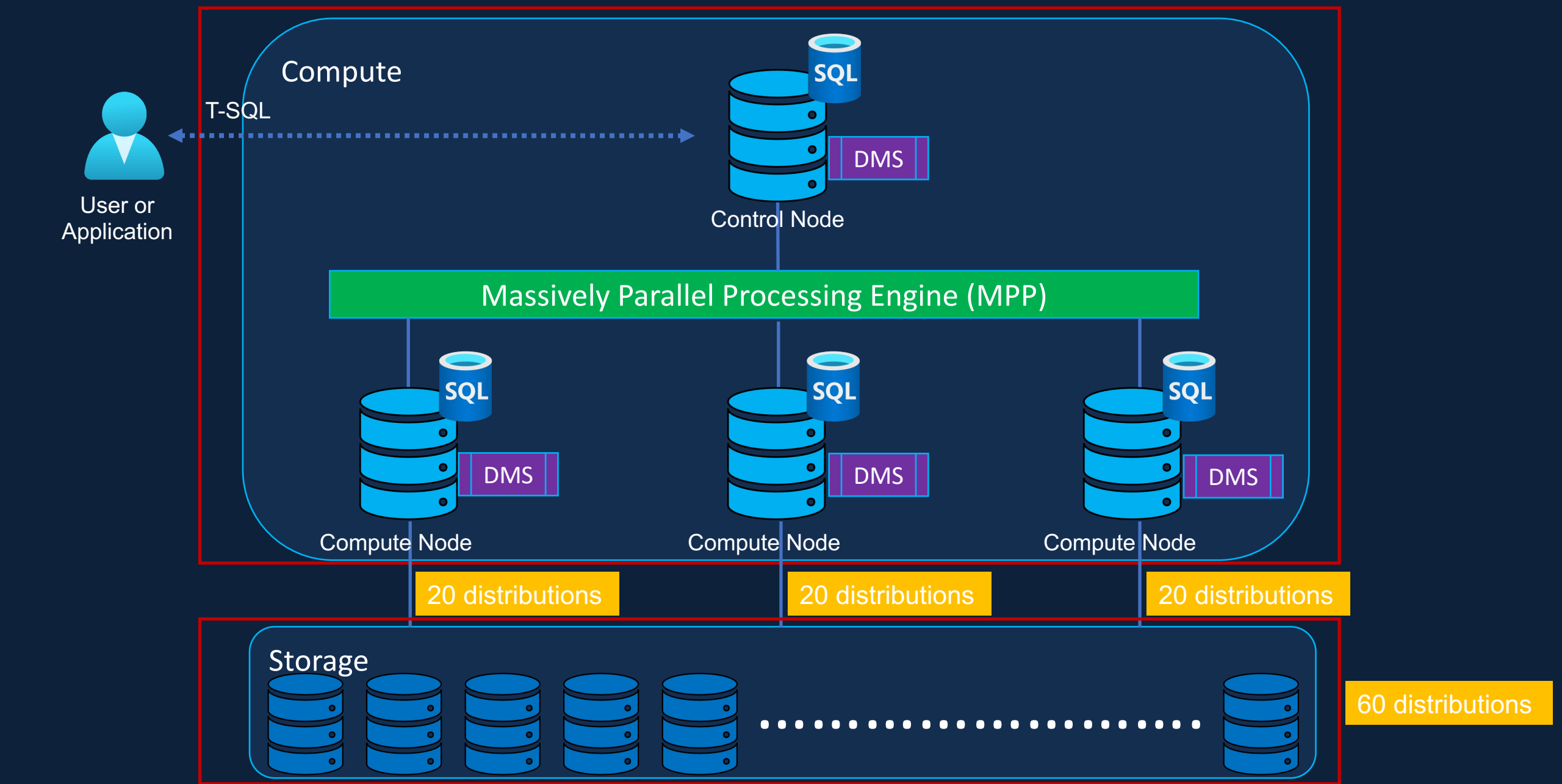
Dedicated SQL Pool - Architecture



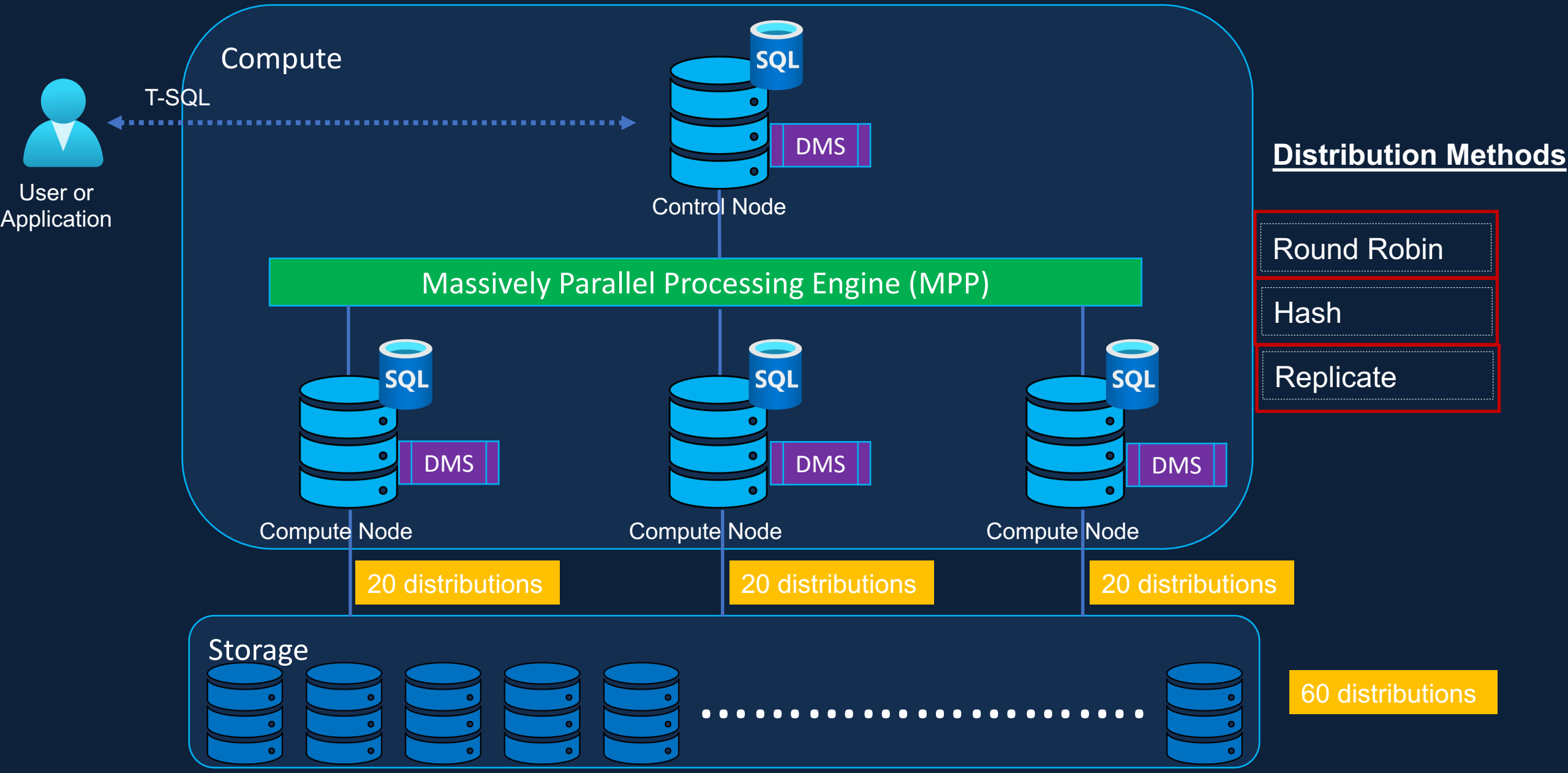
Dedicated SQL Pool - Architecture



Dedicated SQL Pool - Architecture



Dedicated SQL Pool - Architecture



Dedicated SQL Pool – Use Cases

Traditional Data Warehouses (Facts/ Dims)

Larger Data Warehouses (Greater than 1TB)

Instant response times

Predictable Performance

Dedicated SQL Pool – Price & Performance

Based on Data Warehouse Units (DWUs)

DWU relates to CPU, memory, and IO

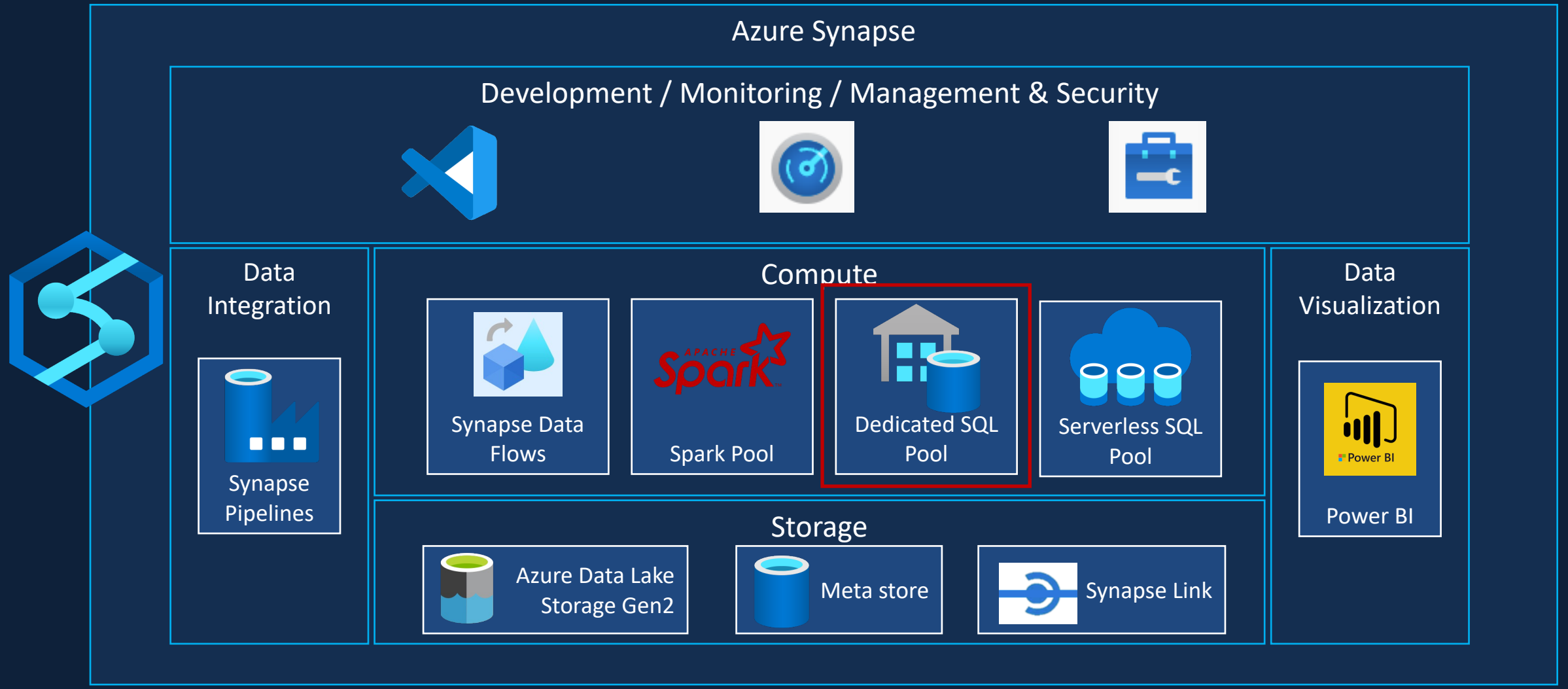
Dedicated SQL Pool - DWU

Performance level (DWU)	Compute nodes	Distributions per Compute node	Memory per data warehouse (GB)
DW100c	1	60	60
DW200c	1	60	120
DW300c	1	60	180
DW400c	1	60	240
DW500c	1	60	300
DW1000c	2	30	600
DW1500c	3	20	900
DW2000c	4	15	1200
DW2500c	5	12	1500
DW3000c	6	10	1800
DW5000c	10	6	3000
DW6000c	12	5	3600
DW7500c	15	4	4500
DW10000c	20	3	6000
DW15000c	30	2	9000
DW30000c	60	1	18000

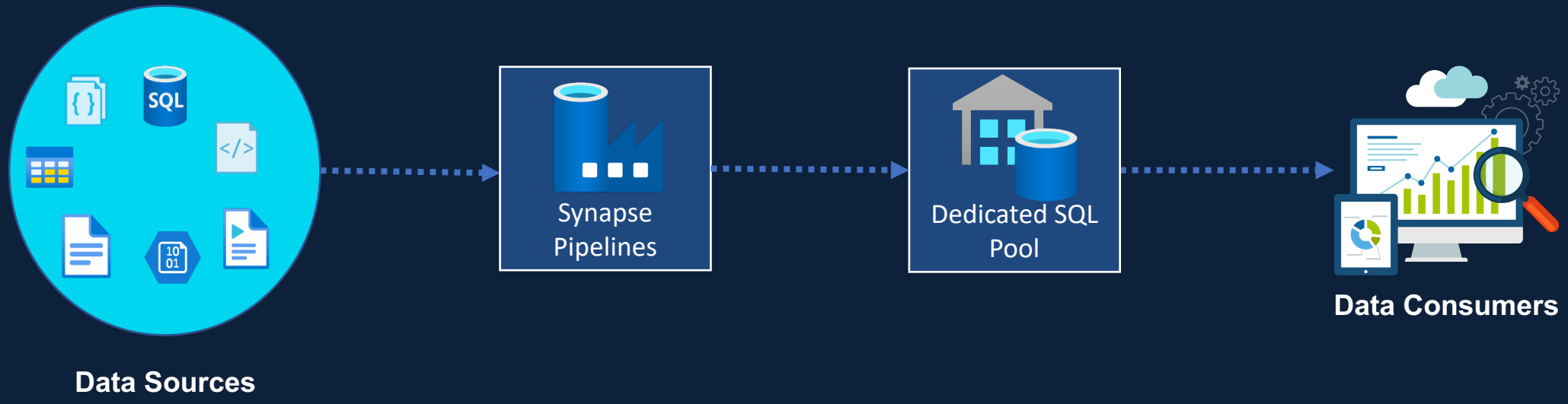
Synapse – Dedicated SQL Pool

Use Cases

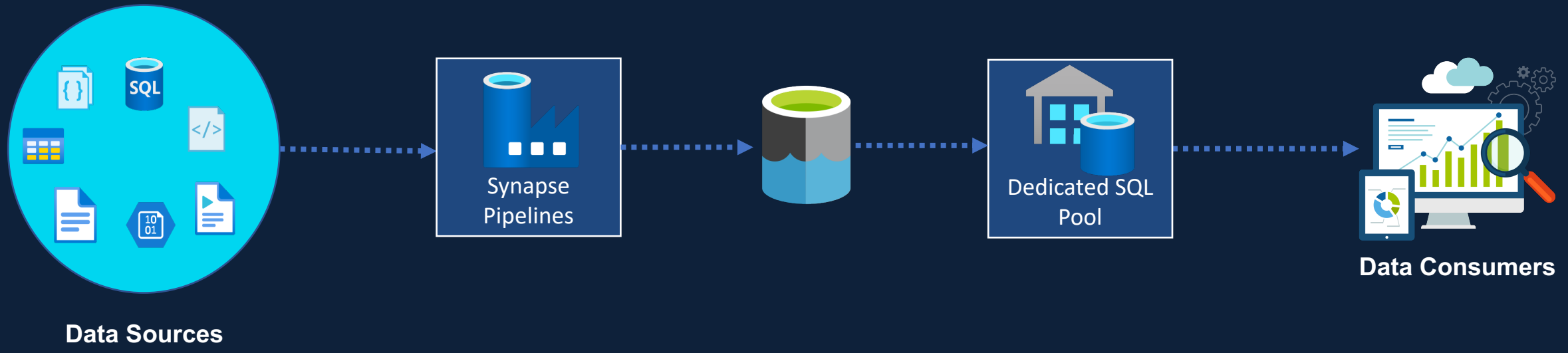
Synapse – Dedicated SQL Pool



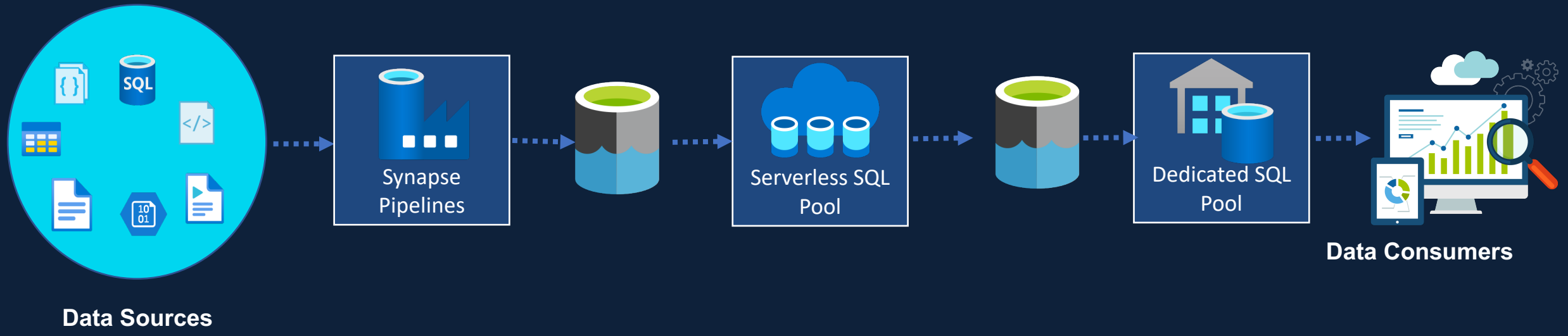
Azure Synapse Analytics



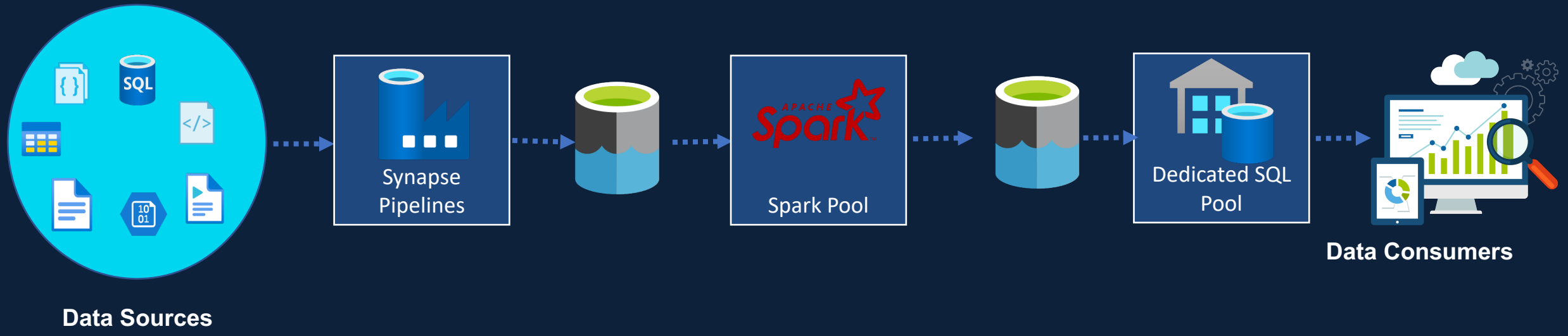
Azure Synapse Analytics



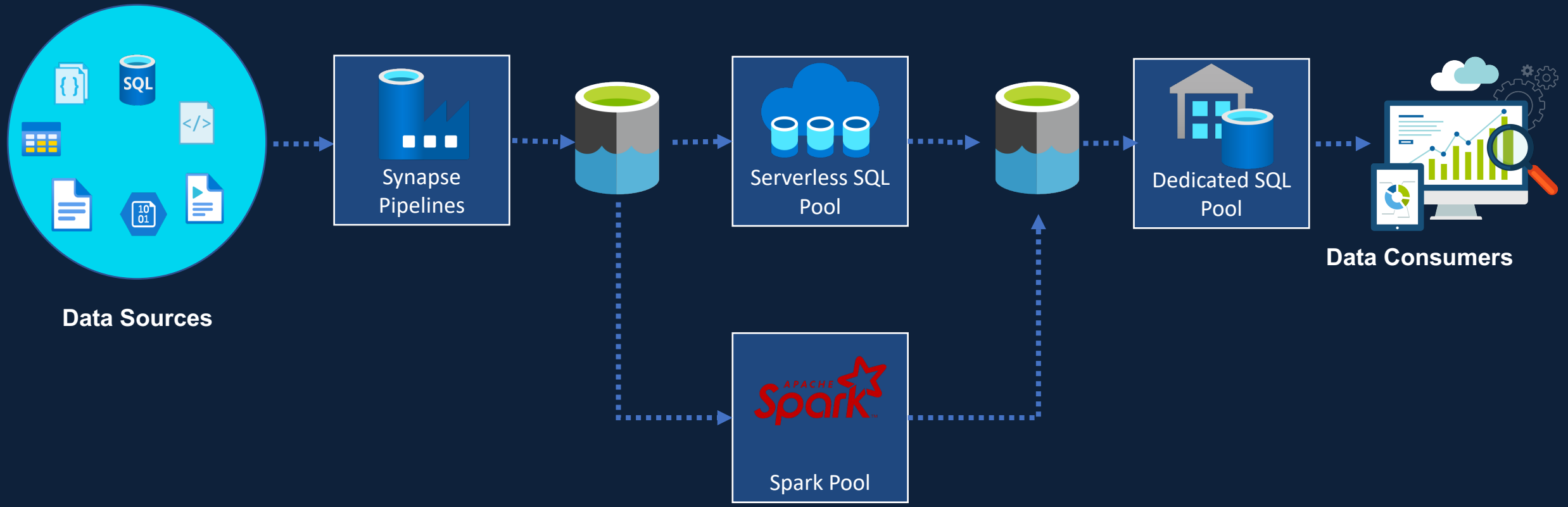
Azure Synapse Analytics



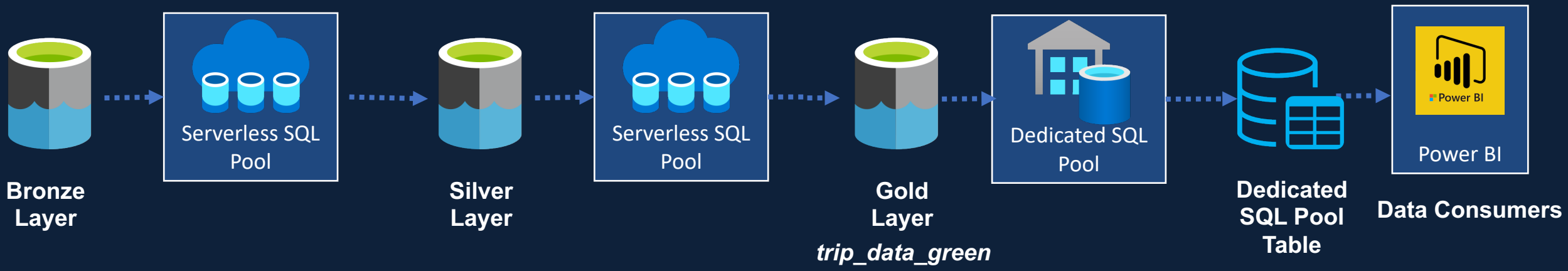
Azure Synapse Analytics



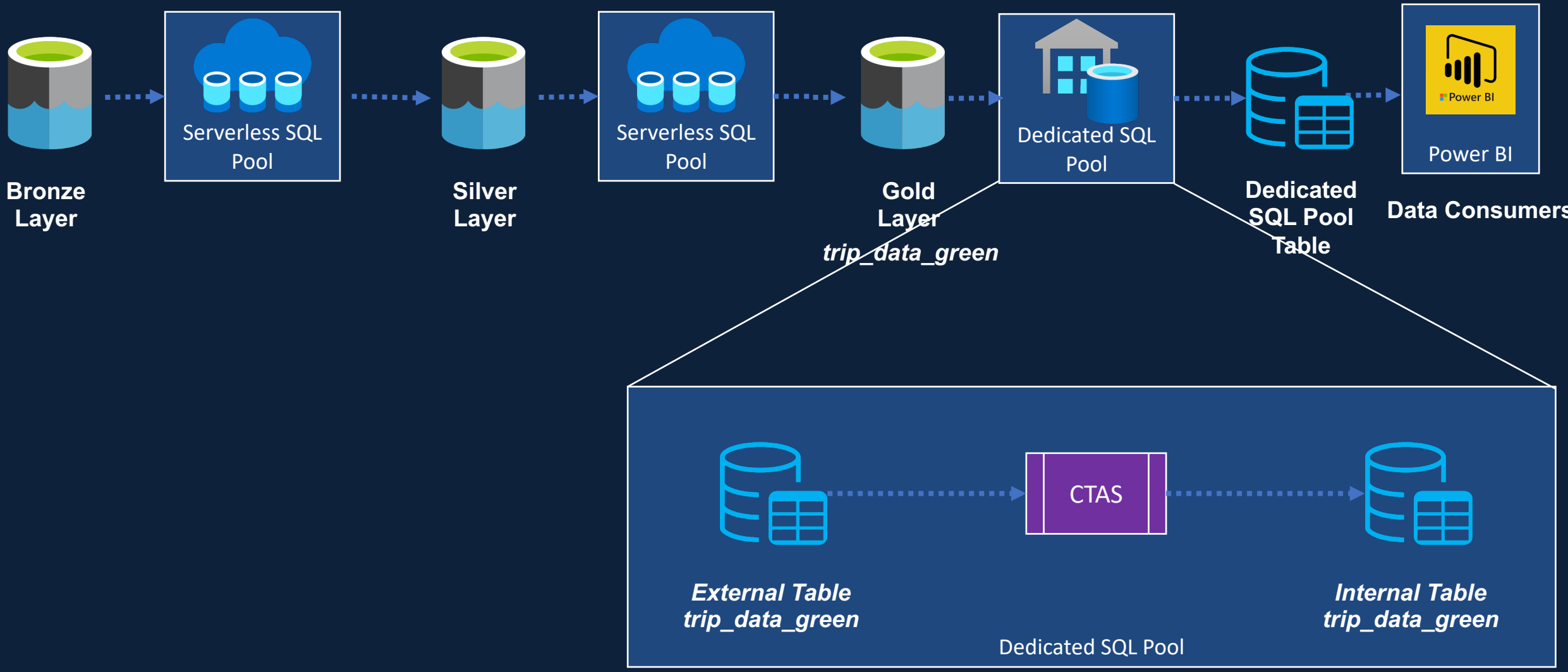
Azure Synapse Analytics



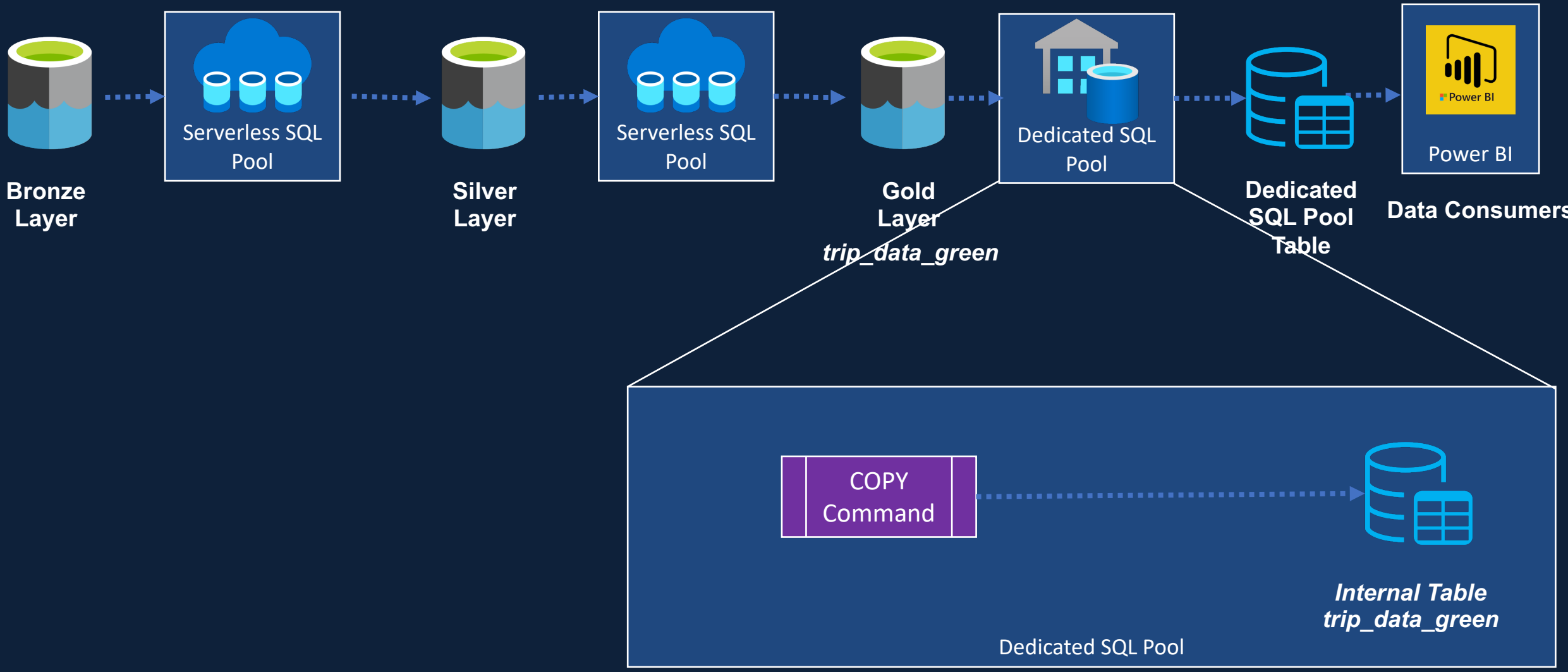
Requirement



Solution


Serverless SQL Pool
Serverless SQL Pool
Dedicated SQL Pool
Power BI
Power BI
Bronze Layer
Silver Layer
Gold Layer
trip_data_green
Dedicated SQL Pool Table
Data Consumers
CTAS
External Table
trip_data_green
Internal Table
trip_data_green
Dedicated SQL Pool

Solution


Serverless SQL Pool
Serverless SQL Pool
Dedicated SQL Pool
Power BI
Power BI
Bronze Layer
Silver Layer
Gold Layer
trip_data_green
Dedicated SQL Pool Table
Data Consumers
COPY Command
Internal Table
trip_data_green
Dedicated SQL Pool

Congratulations!
&
Thank you

Feedback

Ratings & Review

Thank you
&
Good Luck!