Microsoft

Enterprise Data Lake

Benefits and best practices to setup an EDL

January 30, 2020

Sarath Sasidharan (sasasid@microsoft.com)
René Bremer (rebremer@microsoft.com)

Cloud Architect ALLIANCE

Context

- Enterprise Data Lake benefits
- Deep dive: Data access publisher-subscriber
- Deep dive: Metadata
- Conclusion

Enterprise Data Lake (EDL) benefits

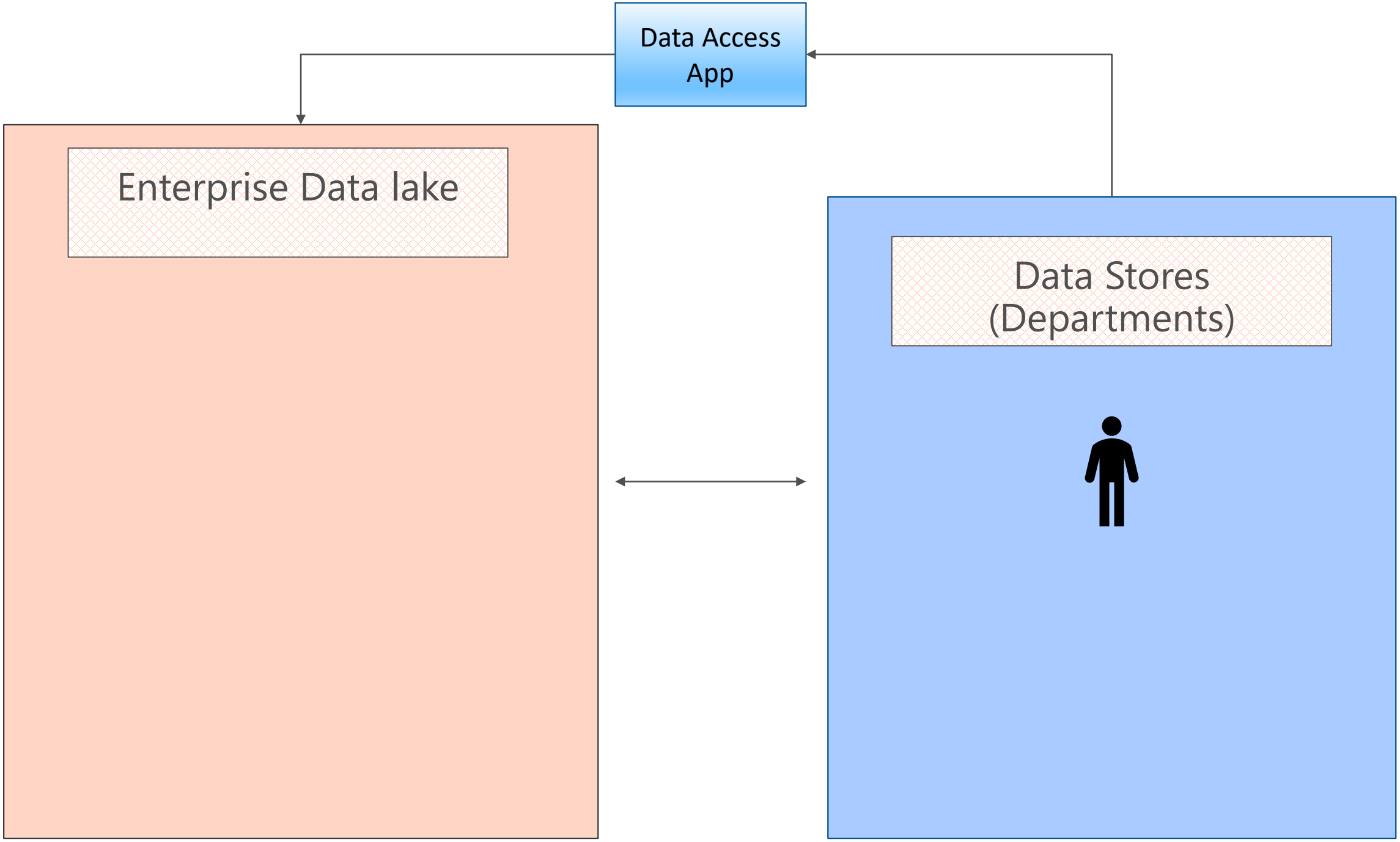
- Business goal of data lake
  - Get more business value out of data in enterprise
  - Scalability and to accommodate data variety and velocity

- Properties of successful data lake
  1. Fine grained data access control
  2. Store data in centralized repository
  3. Facilitate early exploration and innovation

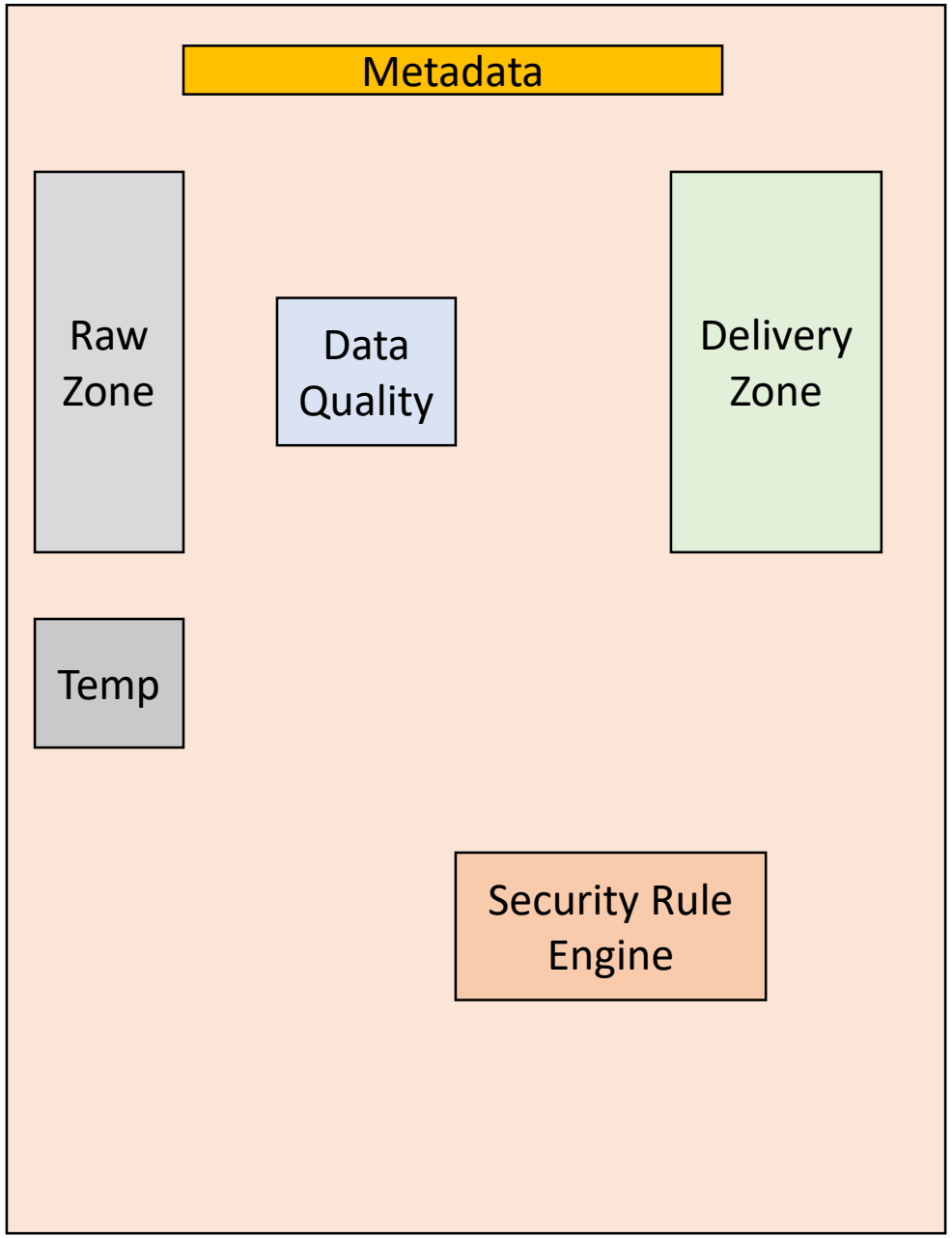
- Architecture patterns:
  - Data access publisher-subscriber model (supporting property 1)
  - Metadata (supporting property 2,3)

Context

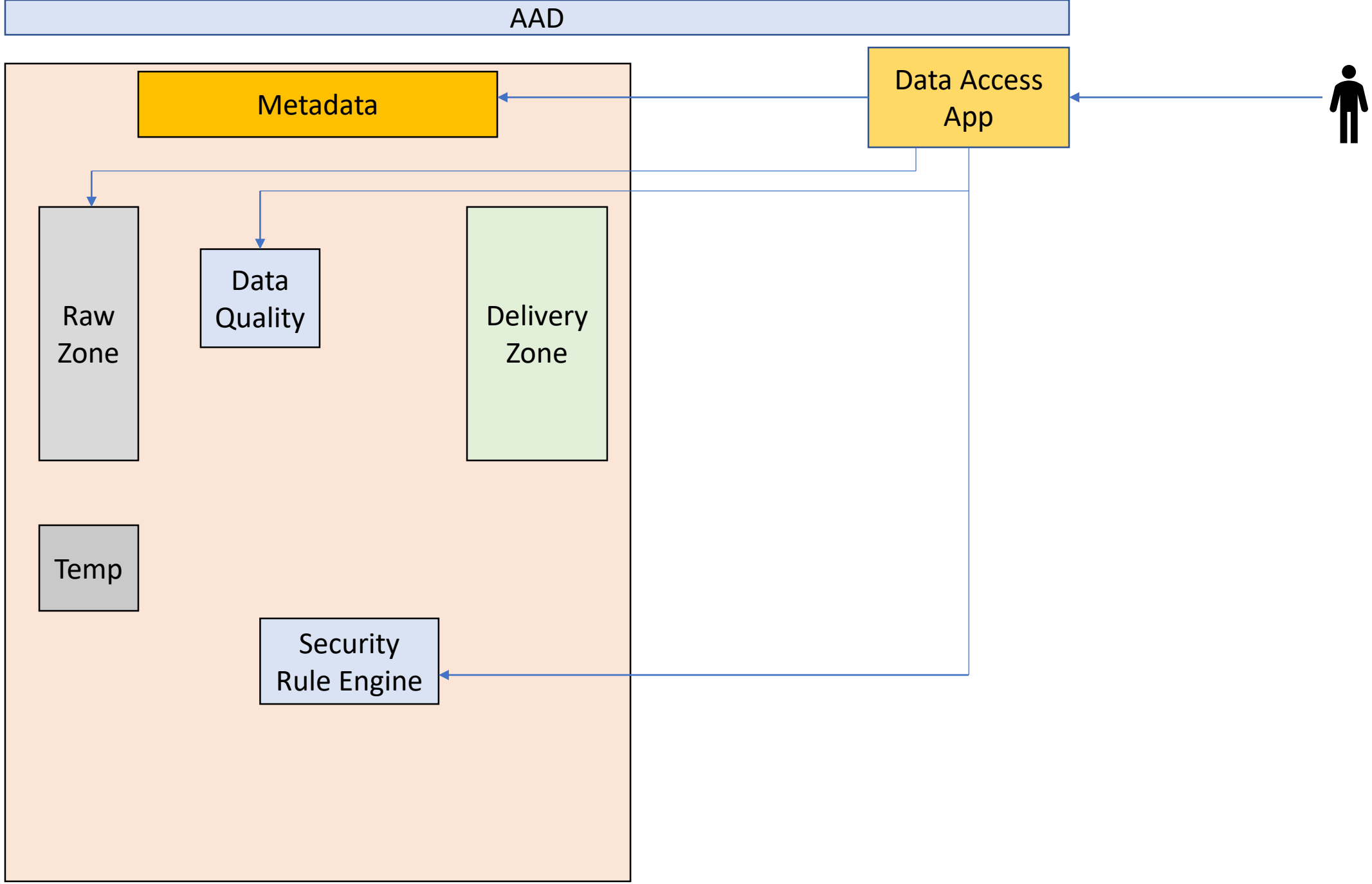
- Enterprise Data Lake benefits
- **Deep dive: Data access publisher-subscriber**
- Deep dive: Metadata
- Conclusion


Data Access App
Enterprise Data lake
Data Stores (Departments)

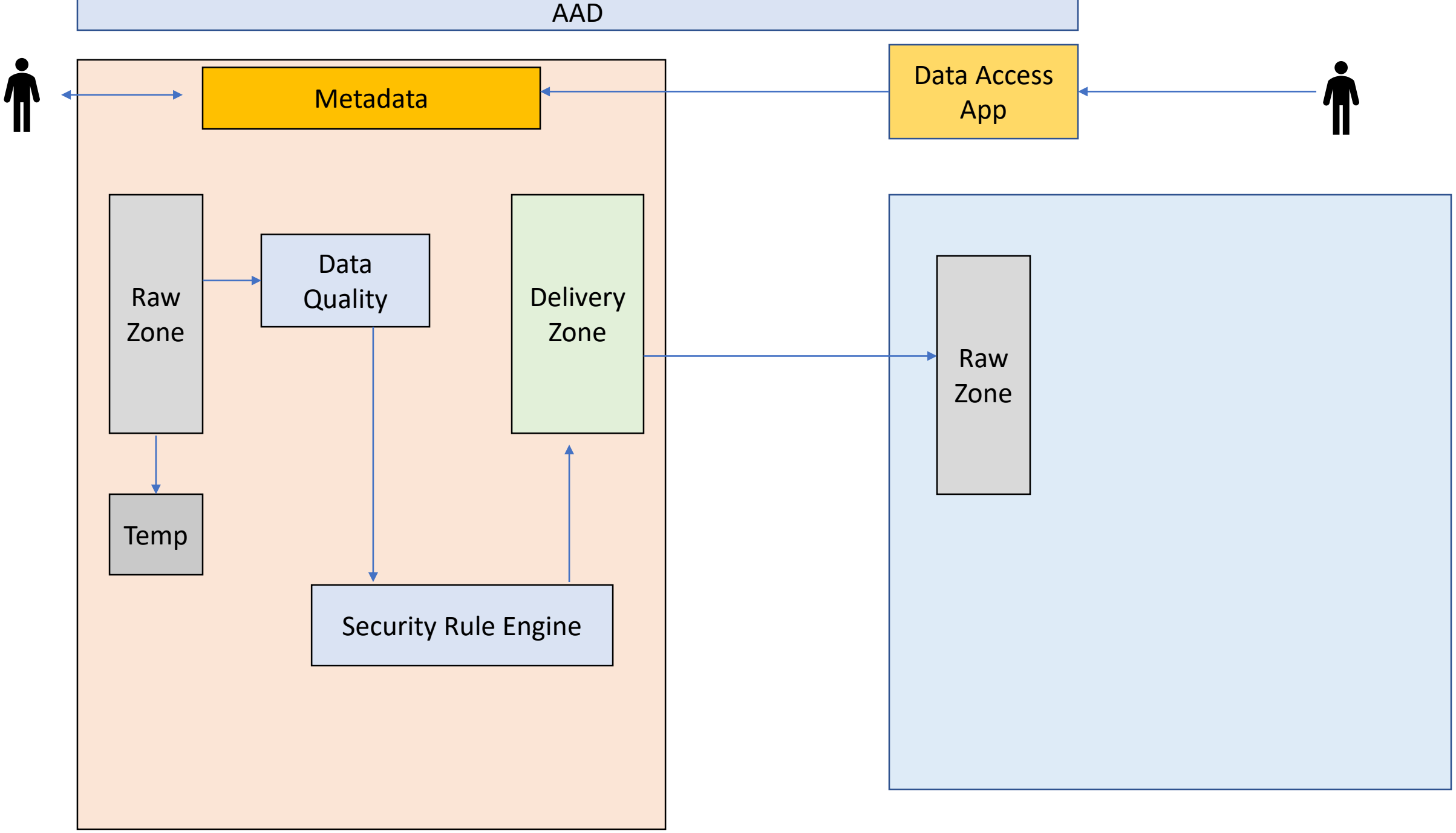
Components In EDL



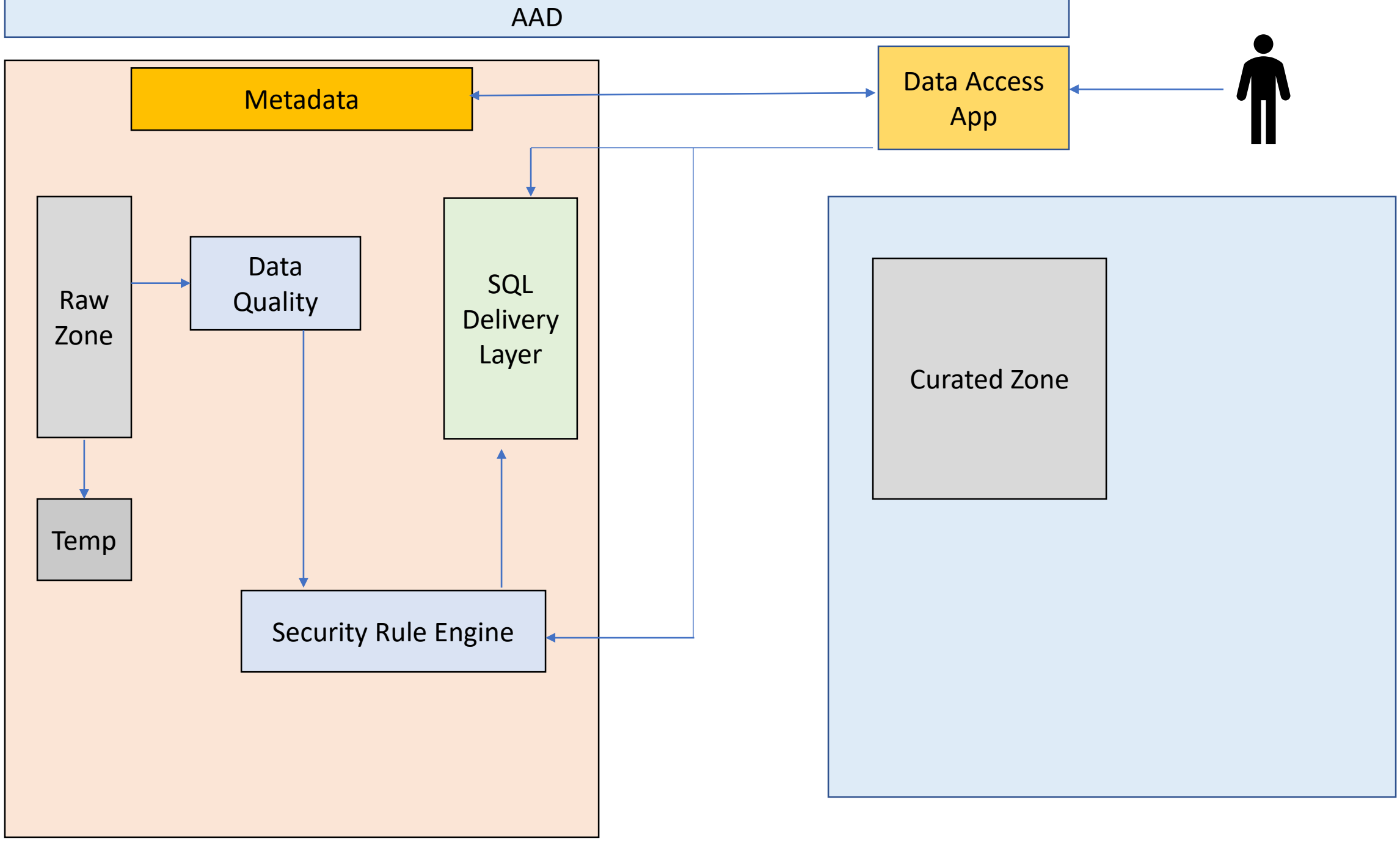
Producer Flow


AAD
Metadata
Data Access App
Raw Zone
Data Quality
Delivery Zone
Temp
Security Rule Engine

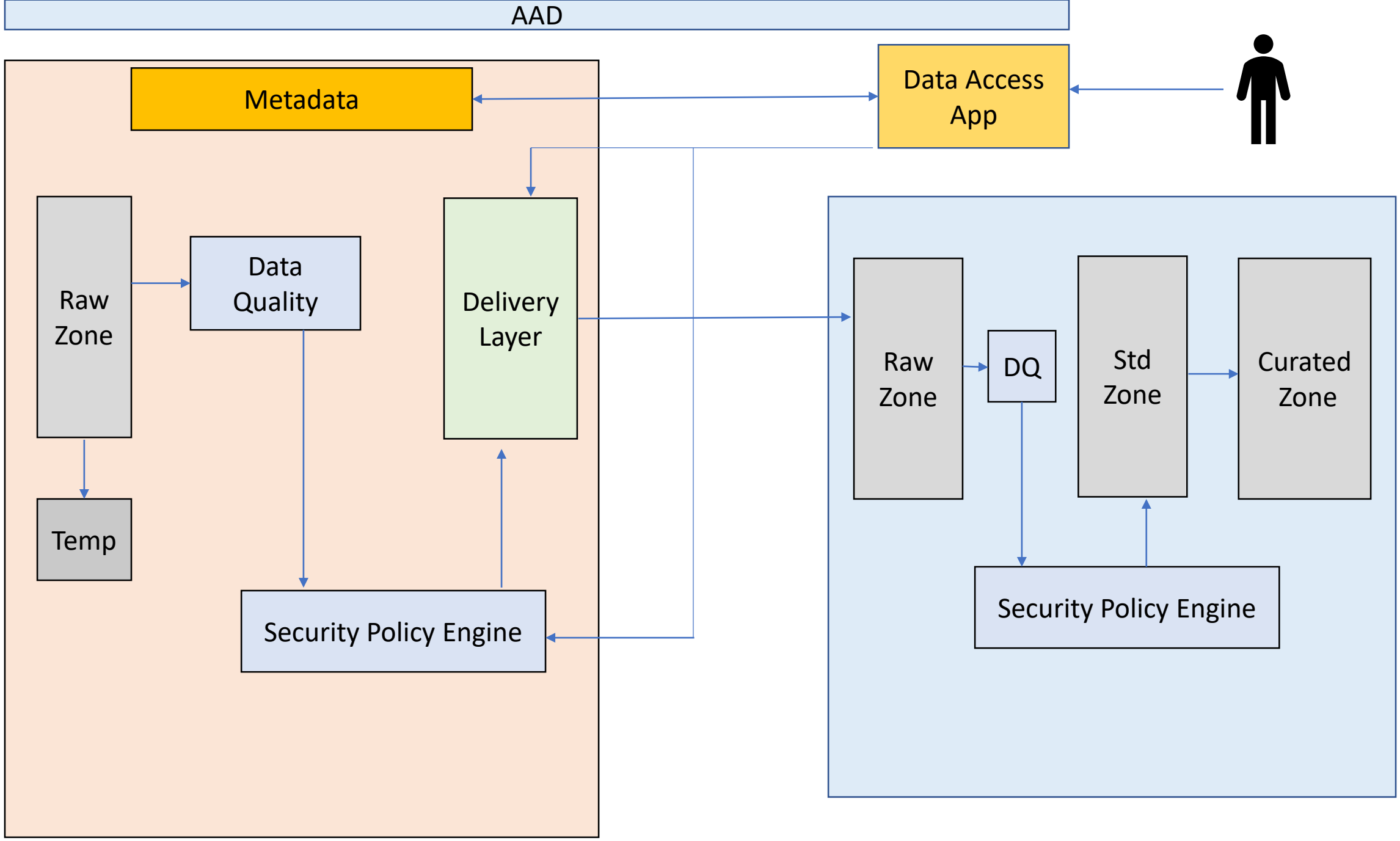
Consumer Flow


AAD
Metadata
Data Access App
Raw Zone
Data Quality
Delivery Zone
Temp
Security Rule Engine
Raw Zone

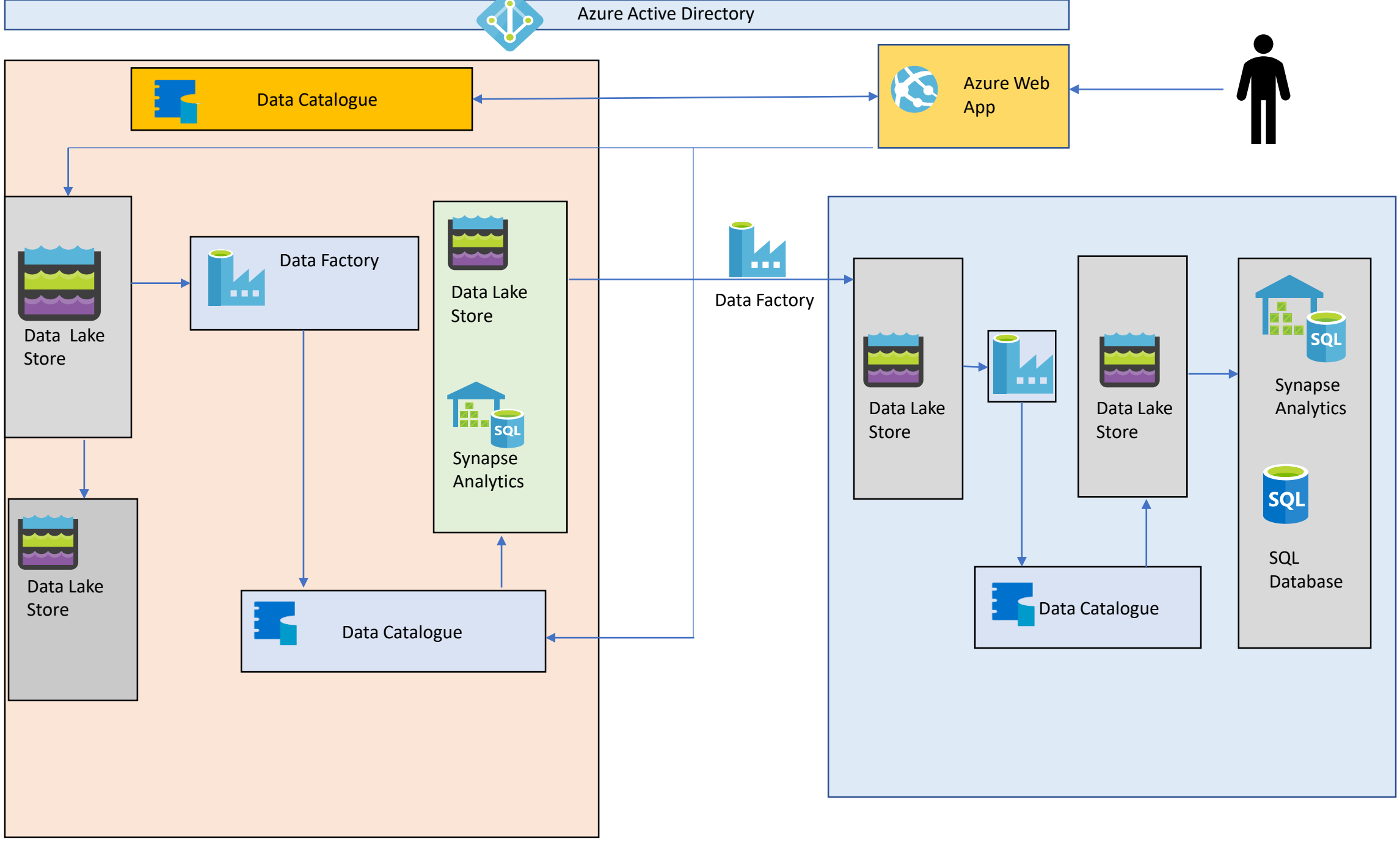
Consumer Flow II


AAD
Metadata
Data Access App
Raw Zone
Data Quality
SQL Delivery Layer
Temp
Security Rule Engine
Curated Zone

The Whole Picture


AAD
Metadata
Data Access App
Raw Zone
Data Quality
Delivery Layer
Temp
Security Policy Engine
Raw Zone
DQ
Std Zone
Curated Zone
Security Policy Engine

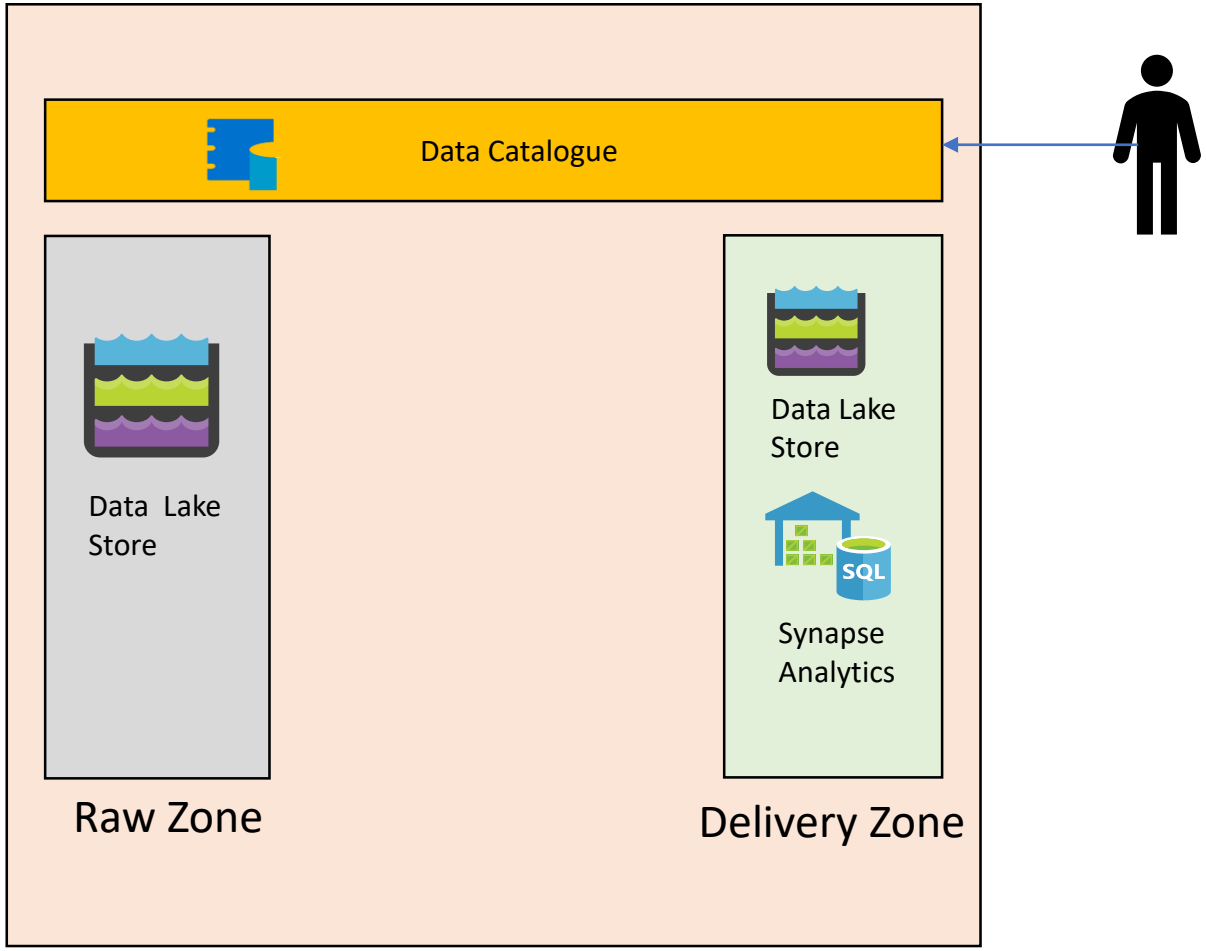
Azure Mapping for EDL / DS


Azure Active Directory
Data Catalogue
Azure Web App
Data Lake Store
Data Factory
Data Lake Store
Synapse Analytics
SQL
Data Lake Store
Data Catalogue
Data Factory
Data Lake Store
Data Lake Store
Synapse Analytics
SQL
SQL Database
Data Catalogue

Context

- Enterprise Data Lake benefits
- Deep dive: Data access publisher-subscriber
- **Deep dive: Metadata**
- Conclusion

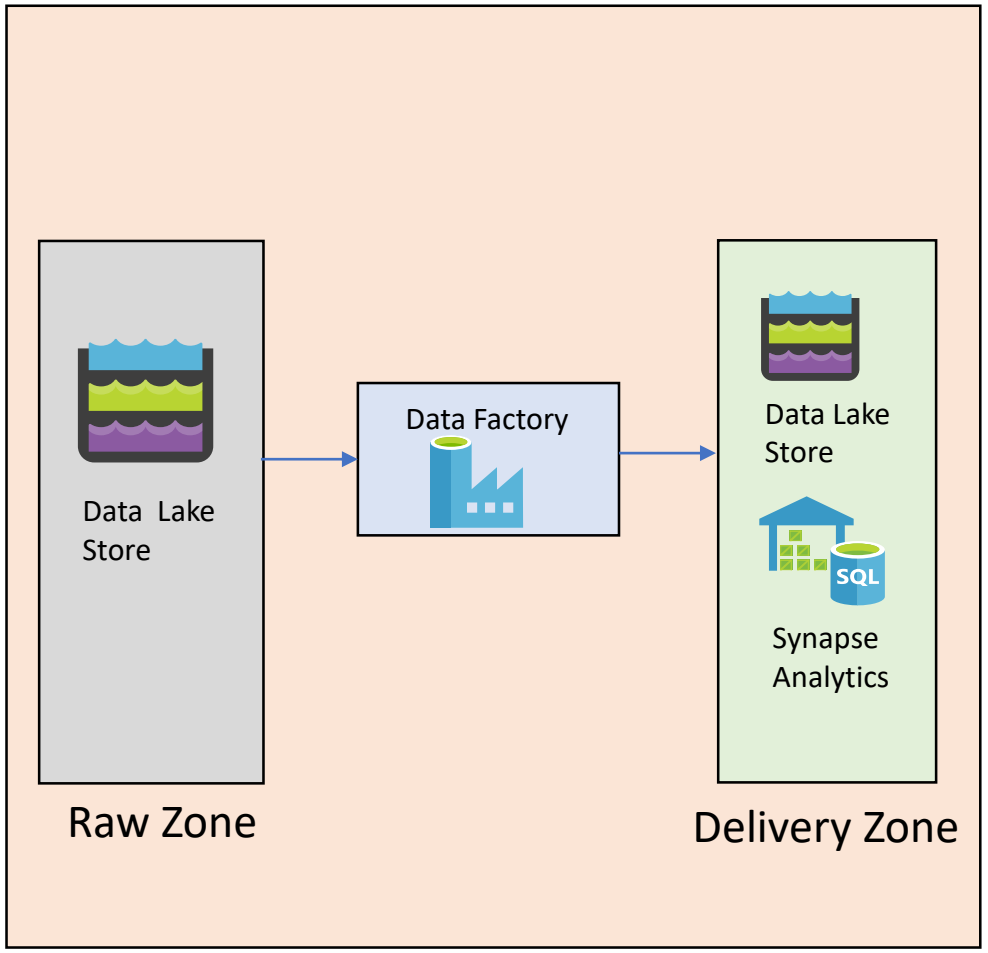
Why metadata-I?



Suppose a user finds a mortgage data set in the catalog. Then the following shall be clear:

- Data owner
- Privacy sensitivity of data
- Retention period
- Freshness of data
- Format/size of data
- Etc

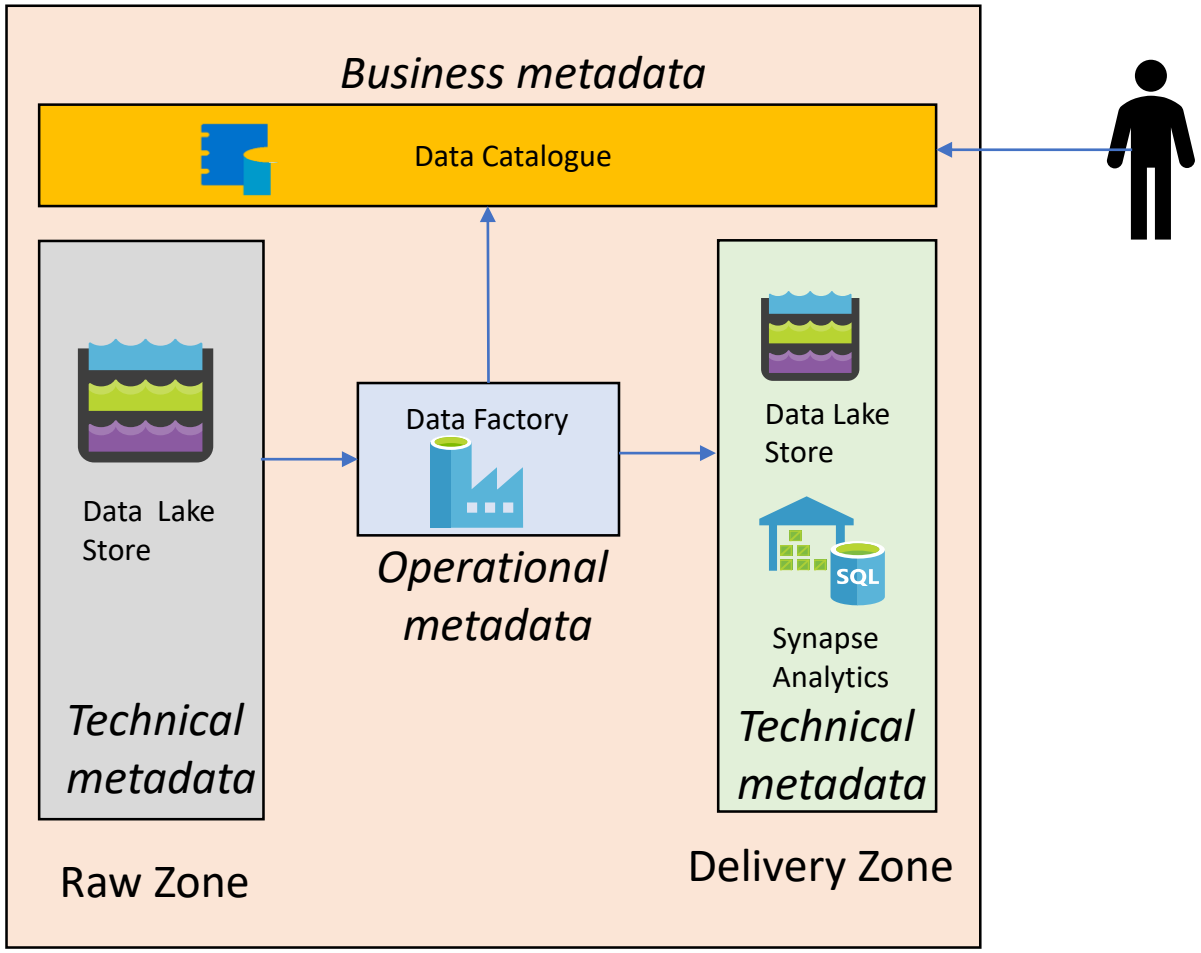
Why metadata-II?



Suppose data is cleansed from Raw to Delivery. Then the following shall be clear:

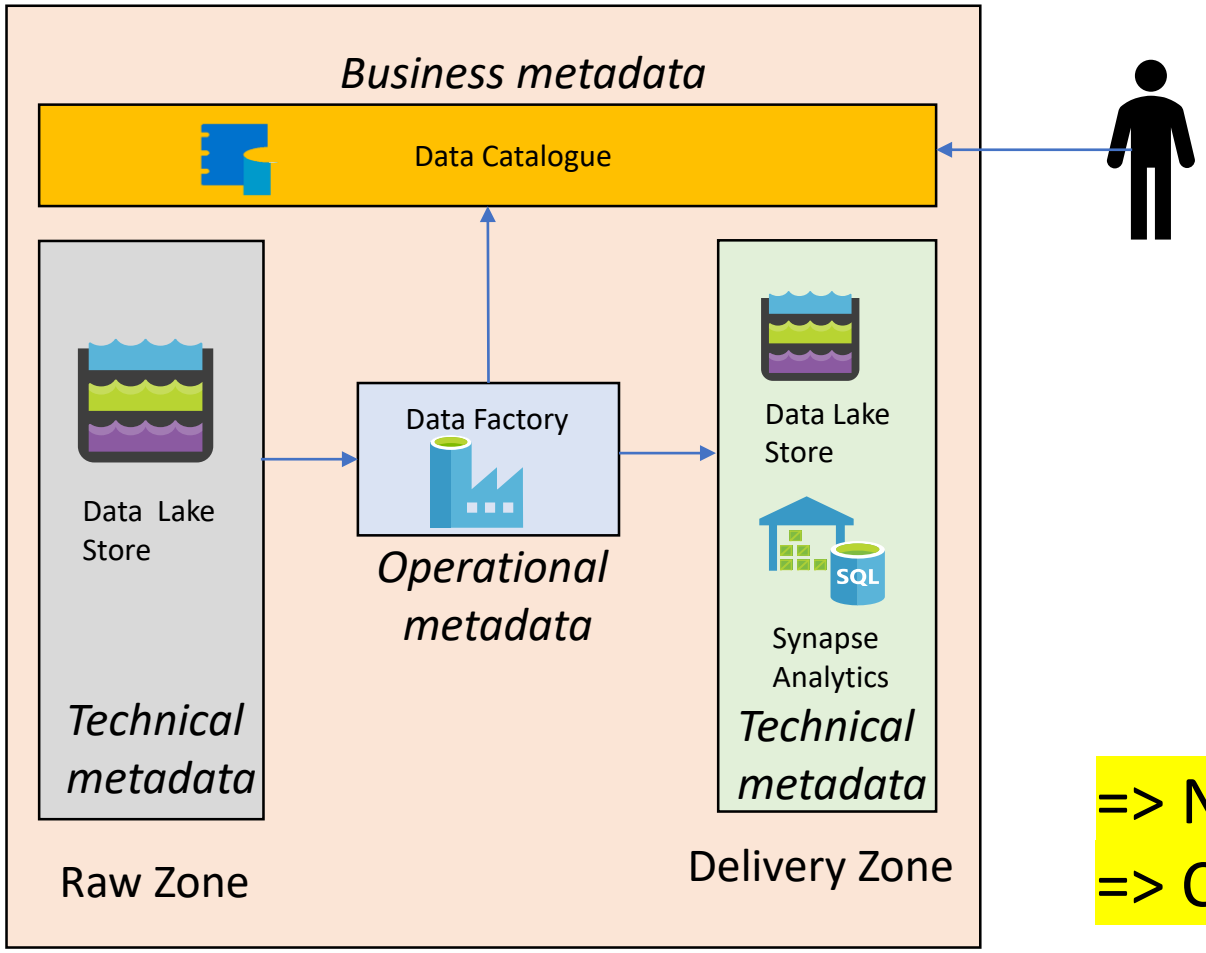
- Schema name, table name, field name
- Field type (string, int)
- Relationships between data
- Lineage of data
- Etc

Forms of metadata



- Business metadata
  - Data owner, data source, privacy level
- Technical metadata:
  - Schema name, table name, field name/type
- Operational metadata:
  - Timestamp, size of data, lineage

Forms of metadata



- Business metadata
  - Data owner, data source, privacy level
- Technical metadata:
  - Schema name, table name, field name/type
- Operational metadata:
  - Timestamp, size of data, lineage

=> Need for a uniform format of metadata
=> Common Data Model (CDM) can help

Common Data Model (CDM)

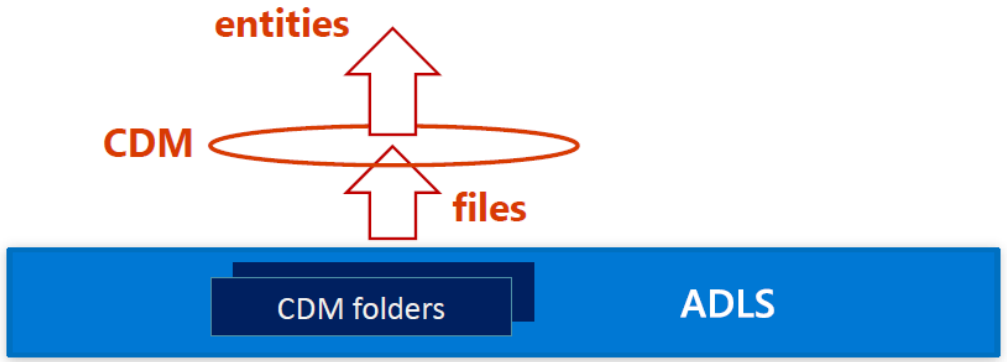
- Shared data language for business and analytical applications

- Includes standard descriptions for industry data models for analytics
  - Account
  - Product
  - …

- Part of Open Data Initiative founded by Adobe, Microsoft and SAP to encourage a date exchange ecosystem
  - Core data models can be extended by customers and partners over time

Common Data Model (CDM) – logical lens

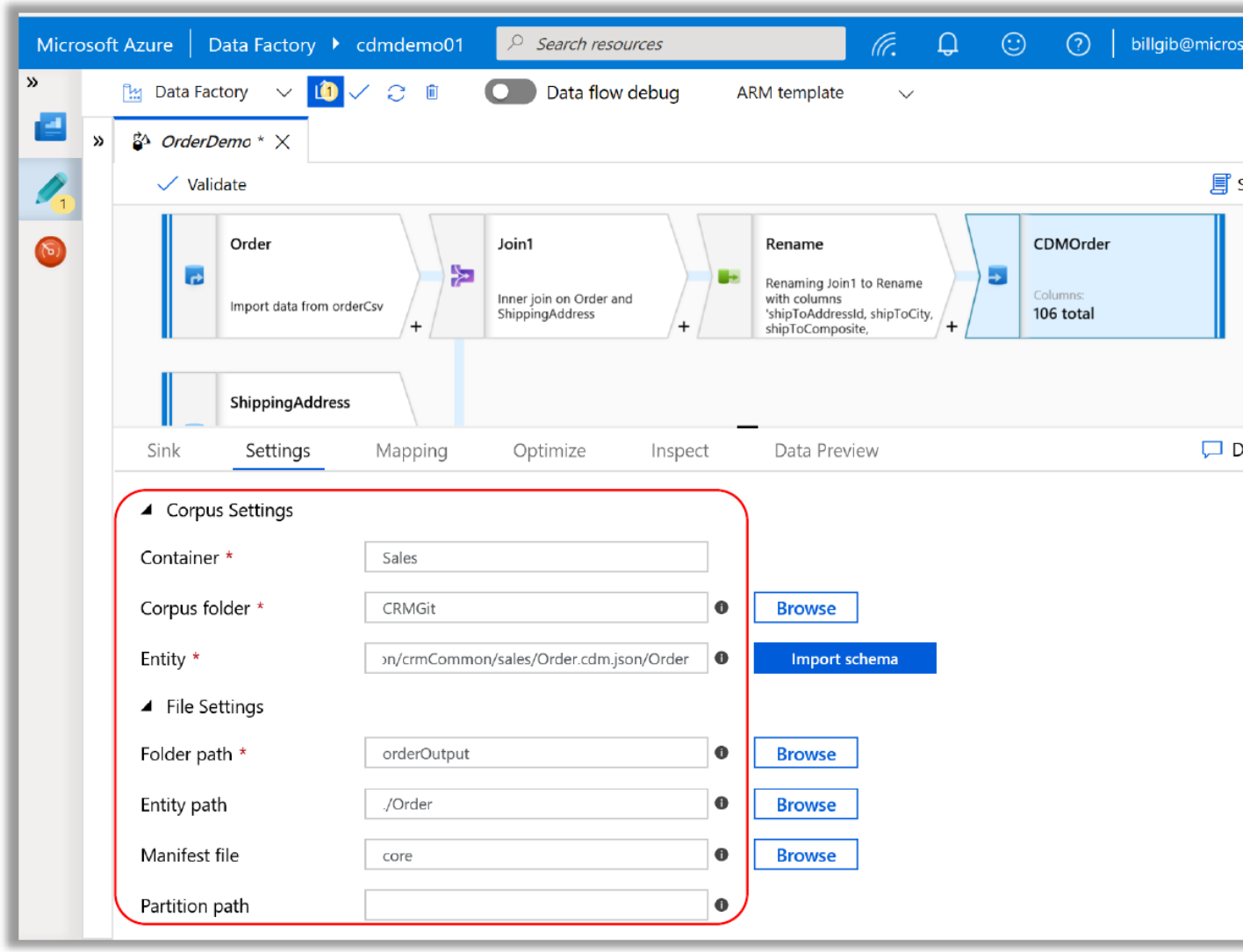
- CDM experiences must project a logical lens over the data



- Next: Example flow how to produce, store and consume CDM

Produce CDM – Azure Data Factory

- CDM entities can be used as a source and/or sink, with 80+ other data connectors
- Full scope of visual transform language is usable
- Data flow is deployed to a cluster for scalable execution
- Can be orchestrated within a pipeline

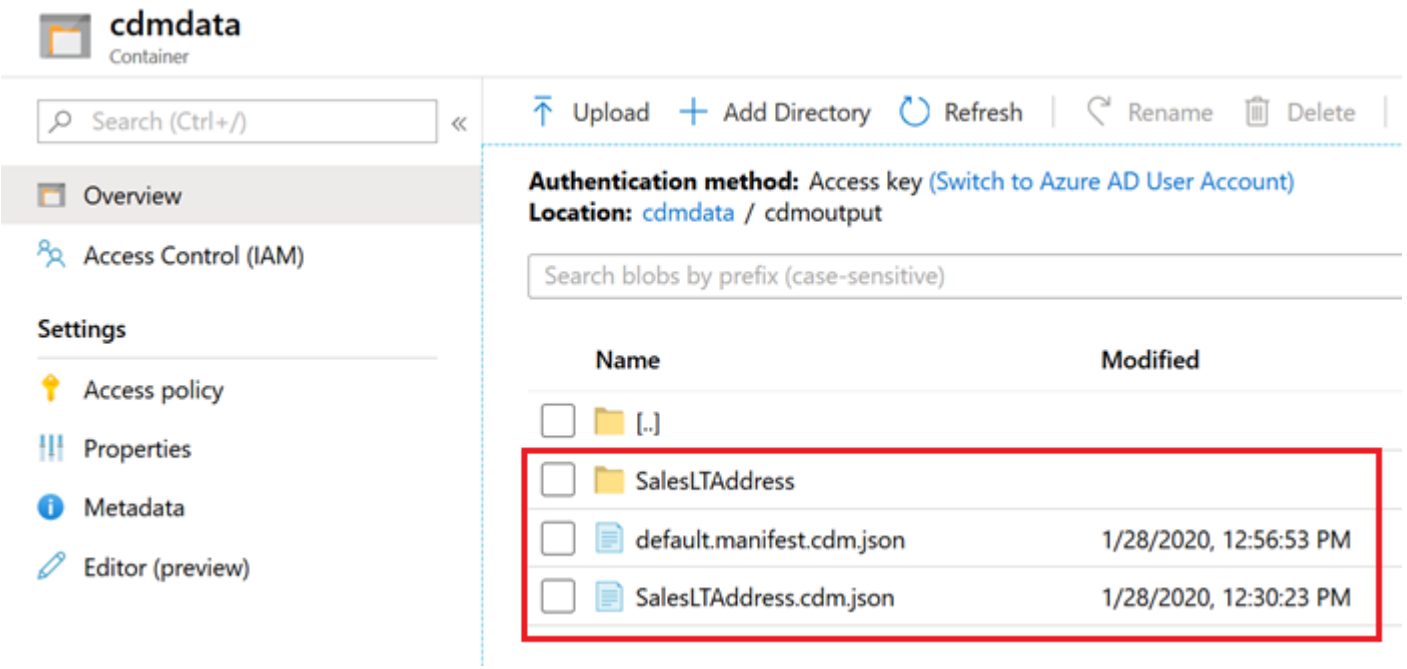


Data flow logical view = entity dataset

Store CDM – ADLSgen2, CDM folder

Most important artifacts of CDM

- default.manifest.cdm.json
- SalesLTAddress.cdm.json
- SalesLTAddress



Store CDM – ADLSgen2, manifest

Upload Add Directory ···

Authentication method: Access key (Switch to Azure AD User Account)
Location: cdmdata / cdmoutput

Search blobs by prefix (case-se...

Name
[..]
SalesLTAddress
default.manifest.cdm.json
SalesLTAddress.cdm.json

cdmoutput/default.manifest.cdm.json
Blob

Save Discard Download Refresh Delete

Overview Edit Generate SAS


```
1   {
2     "jsonSchemaSemanticVersion" : "0.9.0",
3   > "imports" : [ ···
7     ],
8     "manifestName" : "default",
9     "entities" : [
10      {
11        "type" : "LocalEntity",
12        "entityName" : "SalesLTAddress",
13        "exhibitsTraits" : [
14          {
15            "traitReference" : "is.extension.annotation",
16            "arguments" : [{"name":"retention","value":"62"},{"name":"owner","value" :"bremer"}
17          }
18        ],
19        "entityPath" : "SalesLTAddress.cdm.json/SalesLTAddress",
20        "dataPartitionPatterns" : [
21          {
22            "name" : "NewPartition0",
23            "rootLocation" : "SalesLTAddress/2020-01-27/",
24            "regularExpression" : "SalesLTAddress-b4c8a0cab0a54b02ba49c3019dbf45a5*.csv",
25  >         "exhibitsTraits" : [ ···
39            ]
40          },
```

Store CDM – ADLSgen2, entity

Upload Add Directory ···

Authentication method: Access key (Switch to Azure AD User Account)
Location: cdmdata / cdmoutput

Search blobs by prefix (case-se...

Name	
[..]	···
SalesLTAddress	···
default.manifest.cdm.json	···
SalesLTAddress.cdm.json	···

cdmoutput/SalesLTAddress.cdm.json
Blob

Save Discard Download Refresh Delete

Overview Edit Generate SAS

```
1   {
2     "jsonSchemaSemanticVersion" : "0.9.0",
3   > "imports" : [ ···
8     ],
9     "definitions": [
10      {
11        "entityName": "SalesLTAddress",
12        "hasAttributes": [
13          {
14            "name": "AddressID",
15            "dataFormat": "Int32"
16          },
17          {
18            "name": "AddressLine1",
19            "dataFormat": "String"
20          },
21          {
22            "name": "AddressLine2",
23            "dataFormat": "String"
24          },
```

Store CDM – ADLSgen2, data

Upload | Add Directory | Refresh | Rename | Delete | Change tier | Acquire lease | Break lease

Authentication method: Access key (Switch to Azure AD User Account)
Location: cdmdata / cdmoutput / SalesLTAddress / 2020-01-27

Search blobs by prefix (case-sensitive)

Name	Modified	Access tier	Blob type
[..]			
SalesLTAddress-6d0b88c646014036864c5b1d792ba016-00001.csv	1/27/2020, 5:49:00 PM	Hot (Inferred)	Block blob
SalesLTAddress-b4c8a0cab0a54b02ba49c3019dbf45a5-00001.csv	1/27/2020, 5:44:26 PM	Hot (Inferred)	Block blob
SalesLTAddress-f56290543e284152ba84525de34868bb-00001.csv	1/27/2020, 5:47:53 PM	Hot (Inferred)	Block blob

Consume CDM – Spark

- Read and write entity data and metadata via a Spark dataframe
- Dataframe is configured from CDM metadata on read
- CDM metadata is written/updated on write based on dataframe schema
- Scala library, uses Java CDM SDK
- Can be used in Azure Databricks

Notebook 1 *

Cell | Run all | Publish | Attach to pool9 | Language PySpark (Python)

Cell 2

[52]

```python
# Read PackageTypes entity from an ODI folder
orig_df = spark.read.format("com.microsoft.cdm")\
    .option("storage", "demostoragecdm.dfs.core.windows.net")\
    .option("container", "/product-folder")\
    .option("entity", "PackageTypes")\
    .option("appId", appId)\
    .option("appKey", appKey)\
    .option("tenantId", tenantId)\
    .load()
orig_df.select("*").show(10)
```

Command executed in 6s 158ms by tibisso on 11-05-2019 07:03:07.308 -08:00

▶ **Job execution** Succeeded **Spark** 2 executors 8 cores Spark

```
+-------------+---------------+-----------+-------------------+-------------------+
|PackageTypeID|PackageTypeName|LastEditedBy|          ValidFrom|            ValidTo|
+-------------+---------------+-----------+-------------------+-------------------+
|            1|            Bag|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            2|          Block|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            3|         Bottle|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            4|            Box|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            5|            Can|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            6|         Carton|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            7|           Each|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            8|             Kg|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|            9|         Packet|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
|           10|           Pair|          1|2013-01-01 00:00:00|9999-12-31 23:59:59|
+-------------+---------------+-----------+-------------------+-------------------+
only showing top 10 rows
```

Spark logical view – entity dataframe

Consume CDM 2 – SQL

- On-demand or pool
- Query analytics data without loading to a DW
- High-performance query with Parquet
- Use with any SQL tools, including Excel and Power BI

```
CREATE EXTERNAL DATA SOURCE <data_source_name>
WITH
(
     TYPE                          =  CDM,
     LOCATION                      = '<prefix>://<manifestfilepath>[:<port>'
)

CREATE EXTERNAL TABLE { database_name.schema_name.table_name |
schema_name.table_name | table_name }
    ( <column_definition> [ ,...n ] )
    WITH (
        LOCATION = '<entity name>',
        DATA_SOURCE = external_data_source_name
    )
```

Context

- Enterprise Data Lake benefits
- Deep dive: Data access publisher-subscriber
- Deep dive: Metadata
- Conclusion

Summary

- Properties of successful data lake
  1. Fine grained data access control
  2. Store data in centralized repository
  3. Facilitate early exploration and innovation

- Data access pub-sub
  - Fine grained access to support Producers/Consumers

- Metadata: Business, technical and operation
  - Single source of truth, prevent a Data Swamp
  - CDM to provide logical lens over data, facilitate early exploration