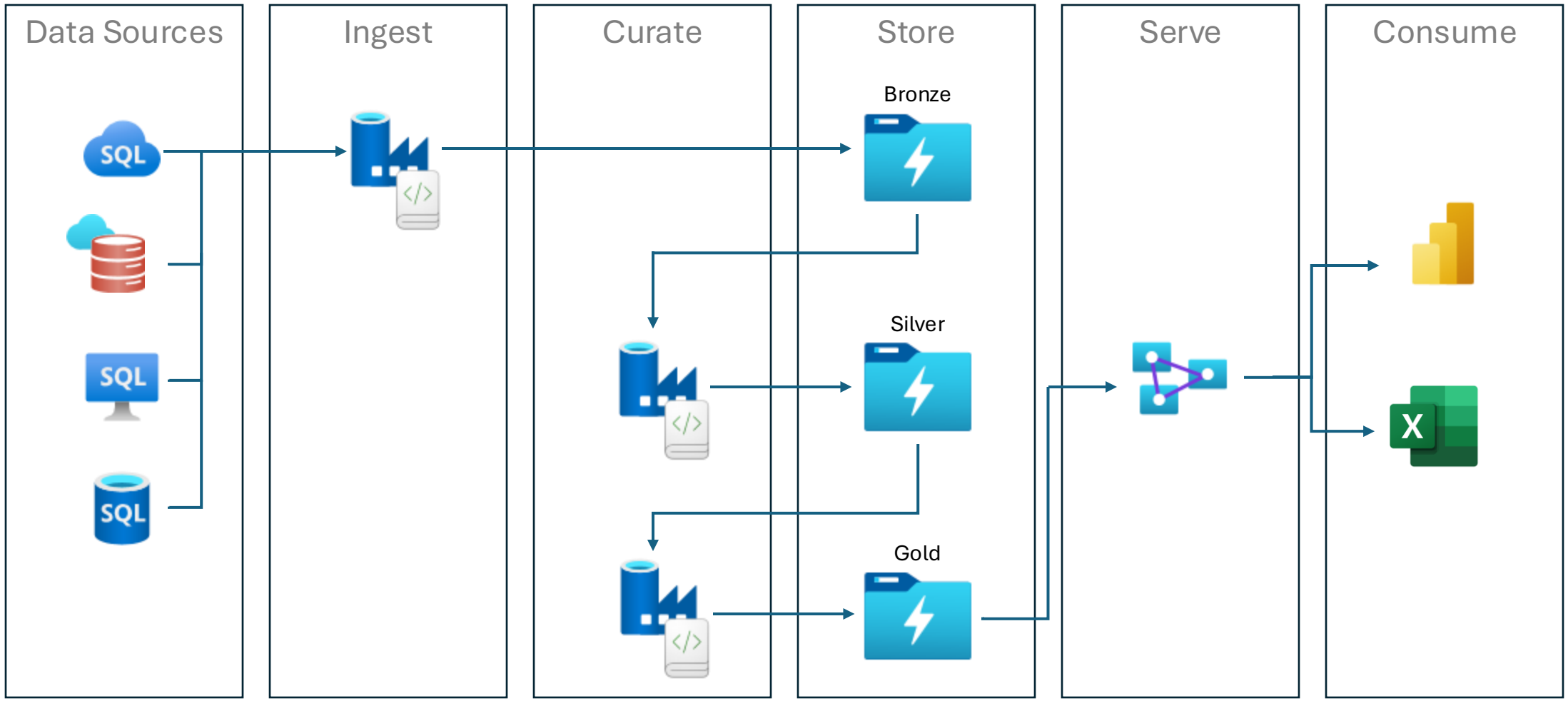
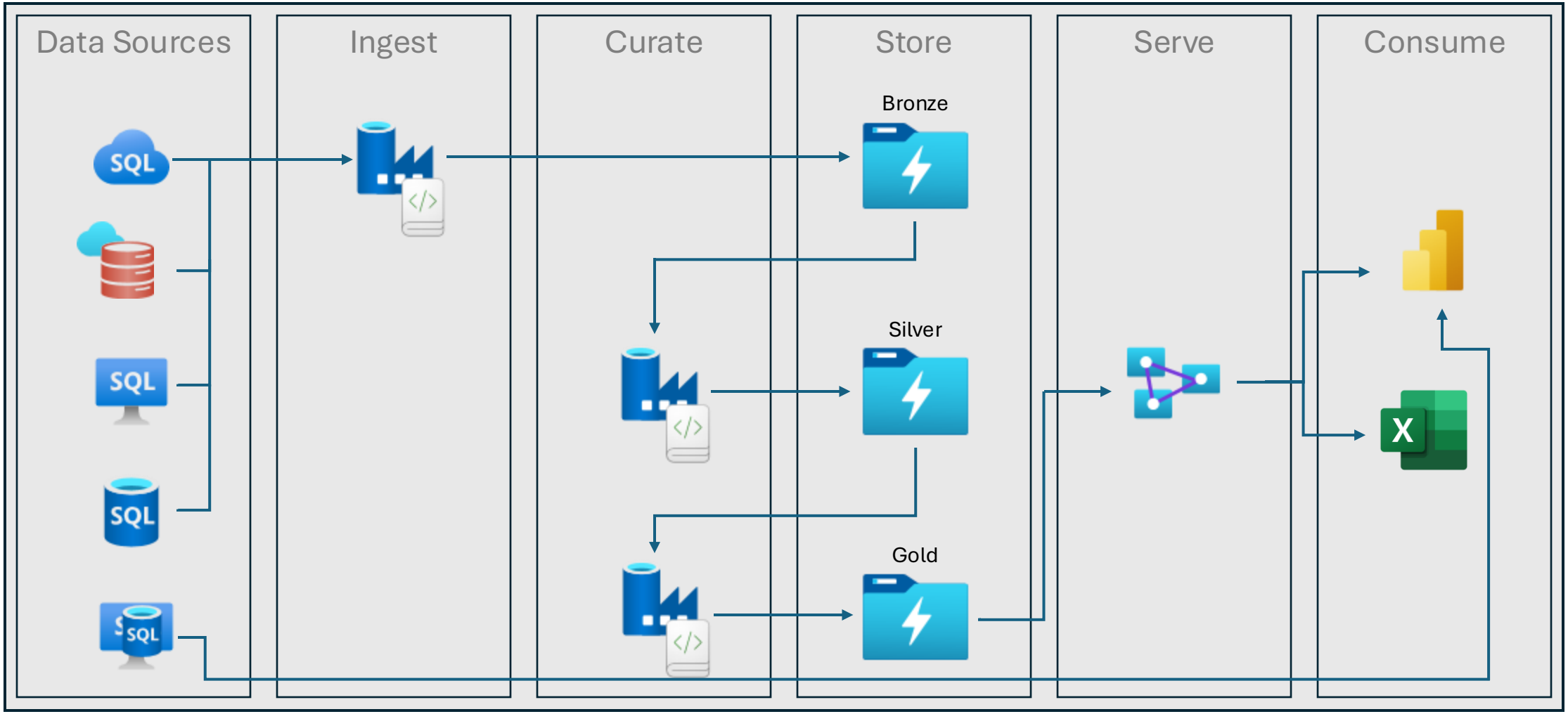
Medallion Architecture - Data Flow



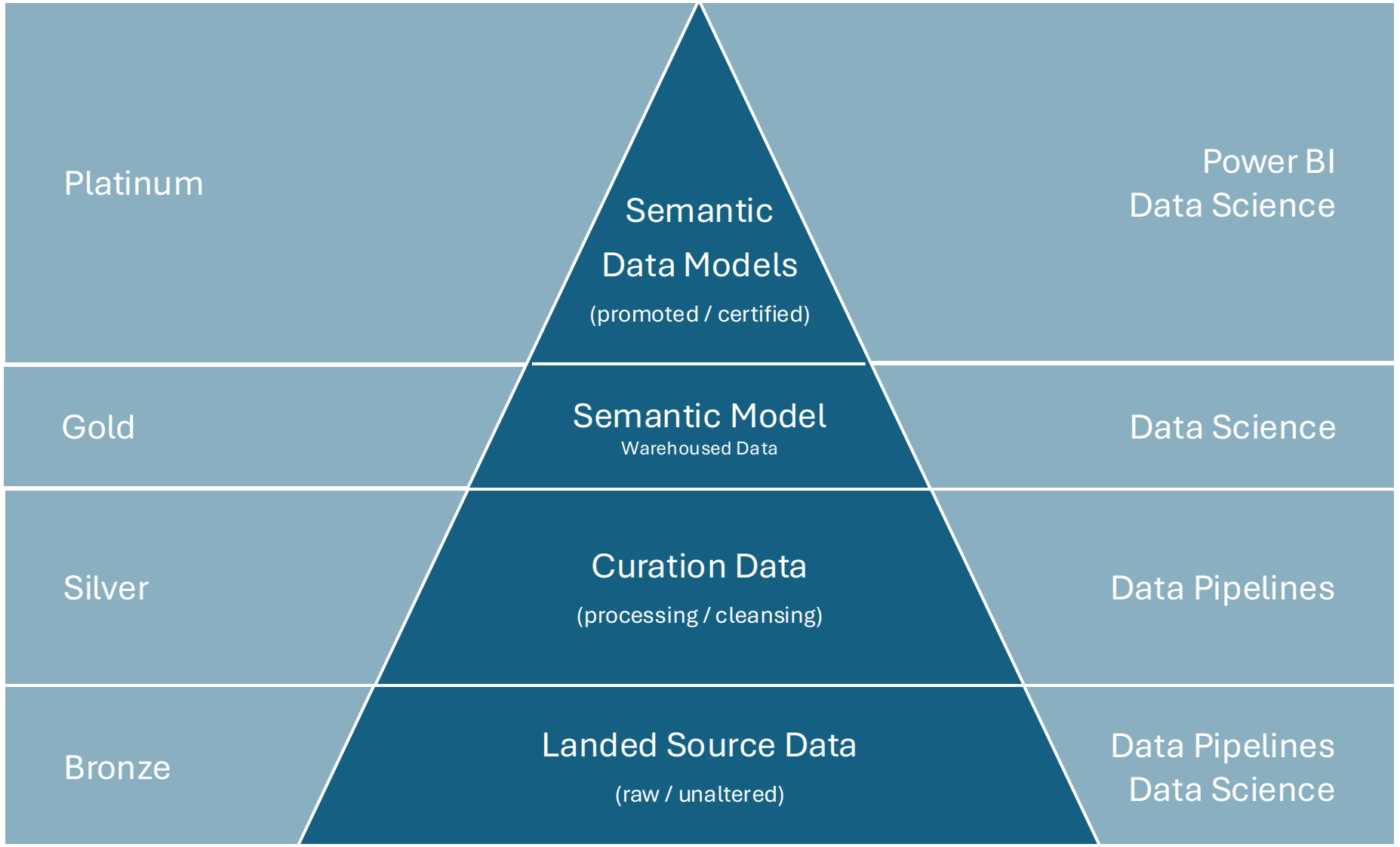
CAF - Data Products

* When ingesting data from operational systems into a read data source. Apart from data quality checks and other applied data, the data should avoid having other data transformations applied to it. This drives reusability of the data product and allow other domains to consume, subject to access, for their use cases as opposed to having multiple extractions from the same operational system.

Medallion Flow - Adopting (temp) Direct Query as Non-Medallion Architecture

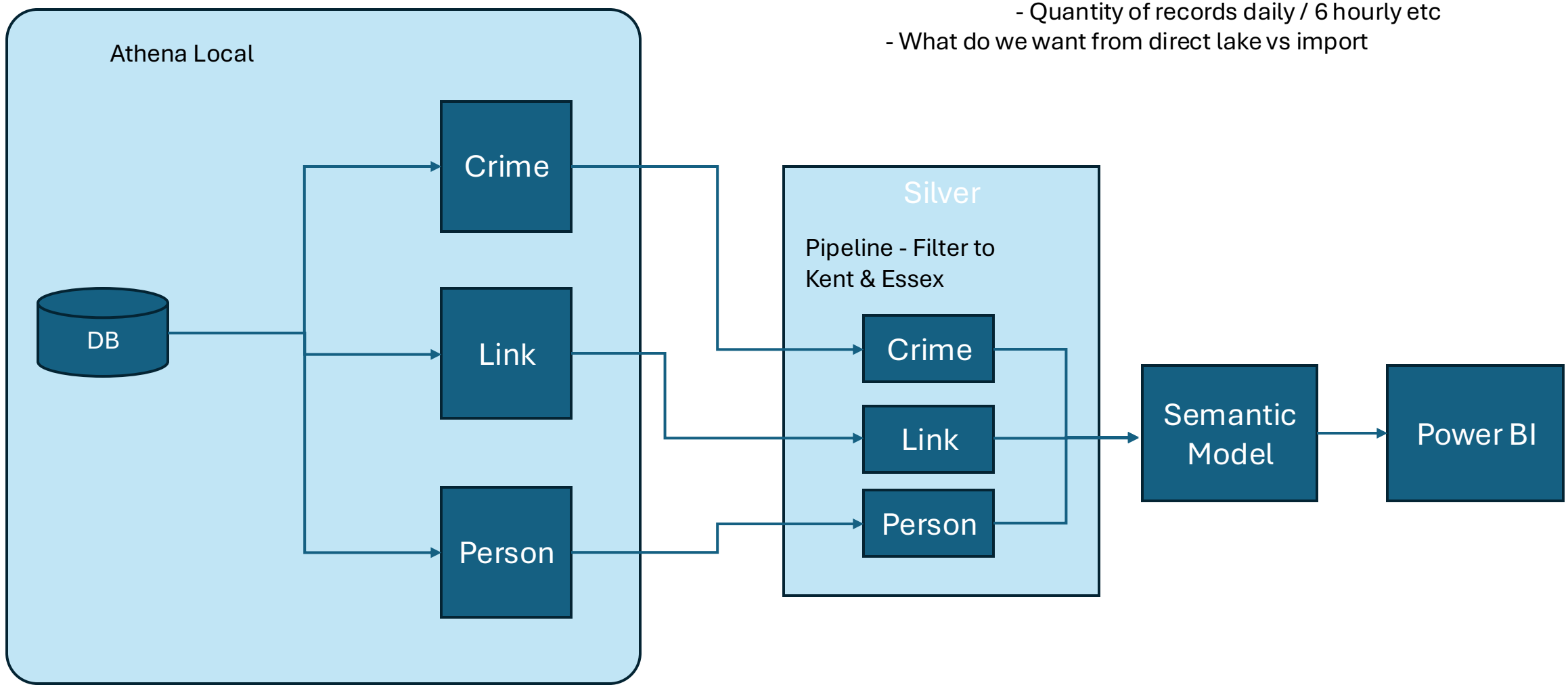


Controlled Data - Accessibility



Athena - Test Architecture in UAT Domain

Once running
- How many records are pulled in from Each Refresh
  - Quantity of records daily / 6 hourly etc
- What do we want from direct lake vs import



MI Reporting - Data / Infra Metrics

Capture from SolarWinds, Tenable, Azure and stored in Log Analytics
Specify what from these systems would be of greatest benefit when presented together (without replacing the main tools' logs / reports

How do we calculate projected processing costs that may require upgrade to F128 - eg if we expand athena ingest from K&E to wider

Infra

Capture from SolarWinds, Tenable, Azure and stored in Log Analytics

Specify what from these systems would be of greatest benefit when presented together (without replacing the main tools' logs / reports

Azure
- Processor usage
- Capacity utilisation

Data Processing

Capture run data from Metadata Framework ingest process

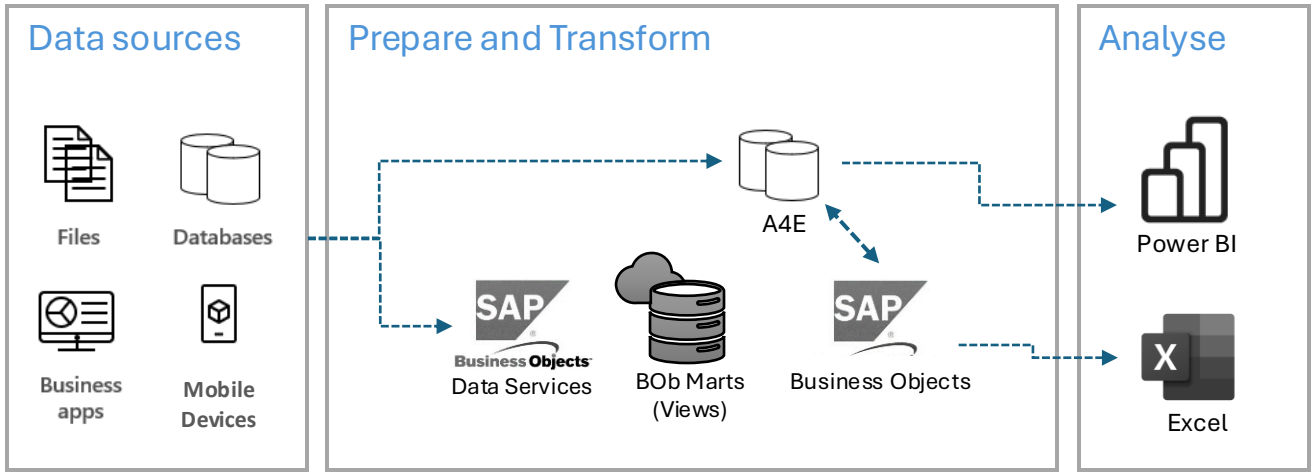
- No of sources processed
- No of pipelines run - success / failed
- No of records ingested / processed / warehoused
- No of records weeded

Button to re-run pipeline if fails

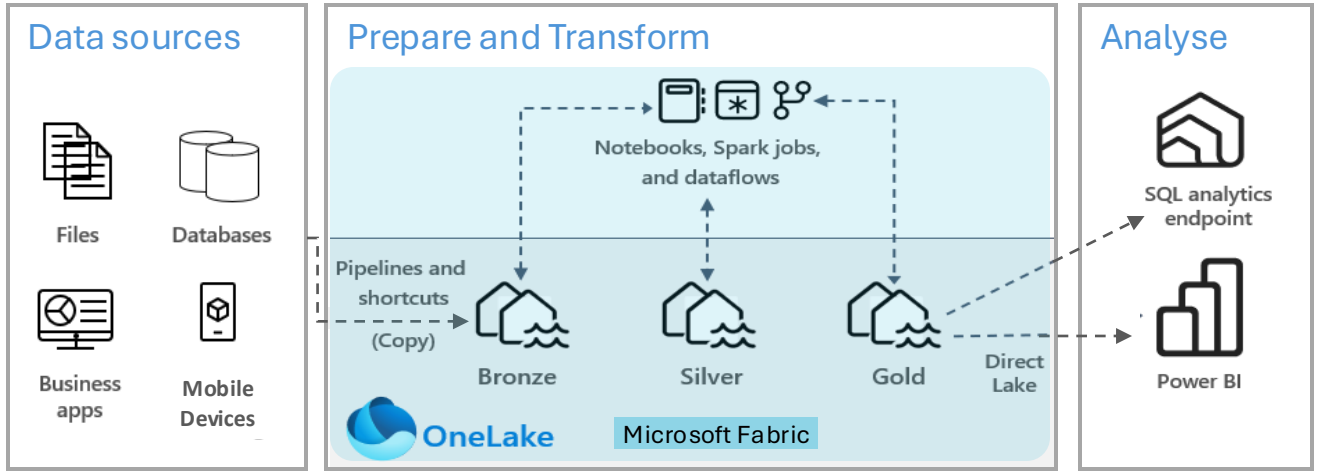
Billing

Existing and To-Be Data / BI Architecture

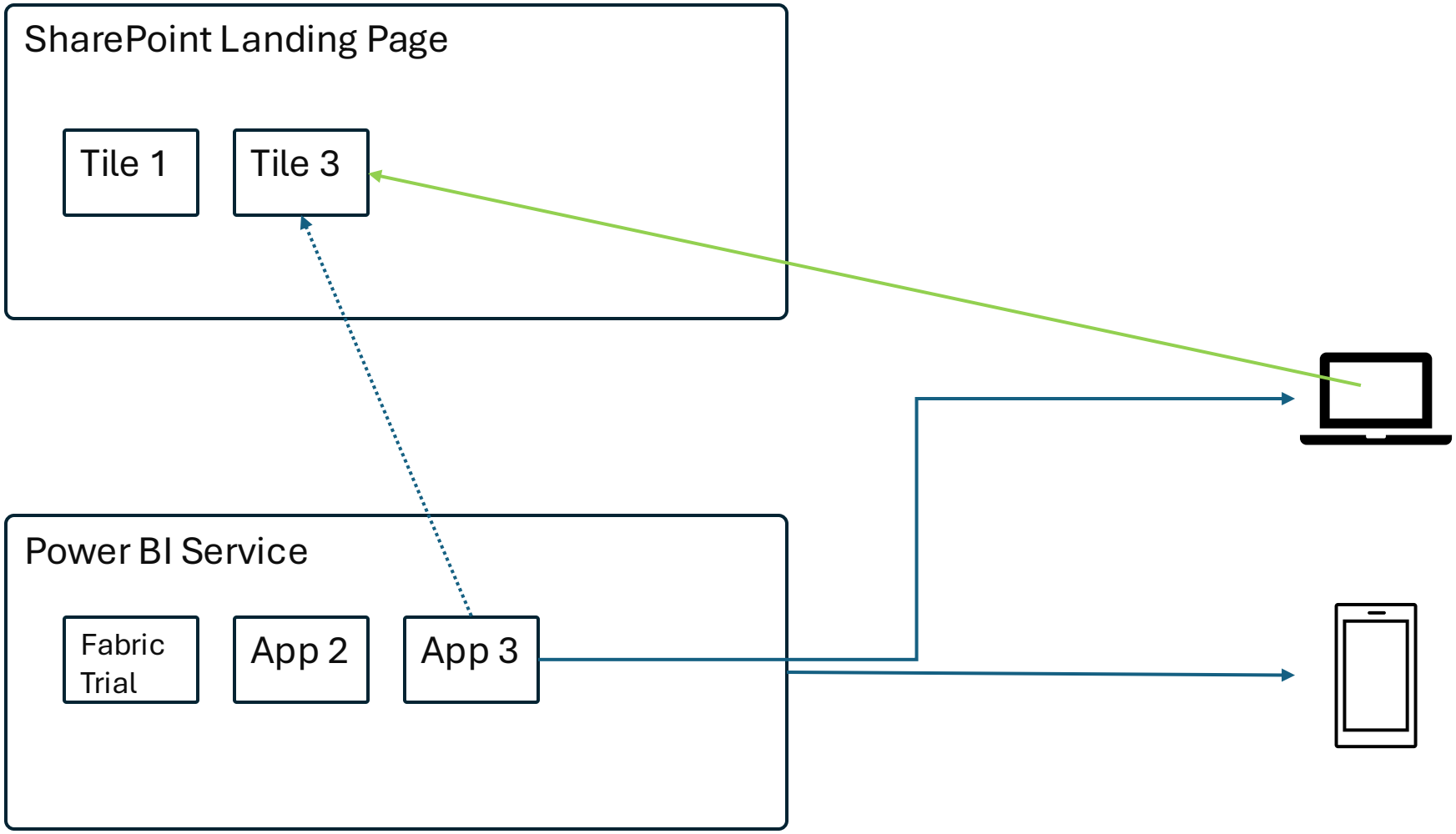
Existing



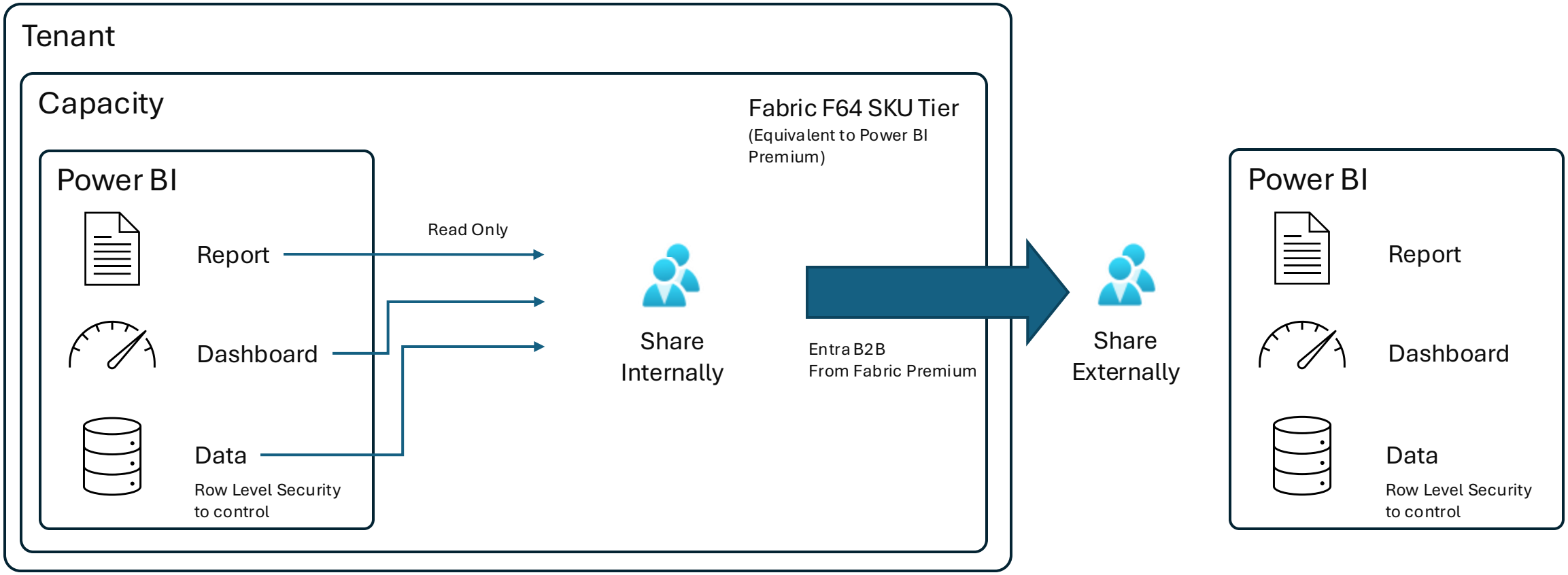
To-Be



Power BI Navigation



Sharing Reports, Dashboards and Data

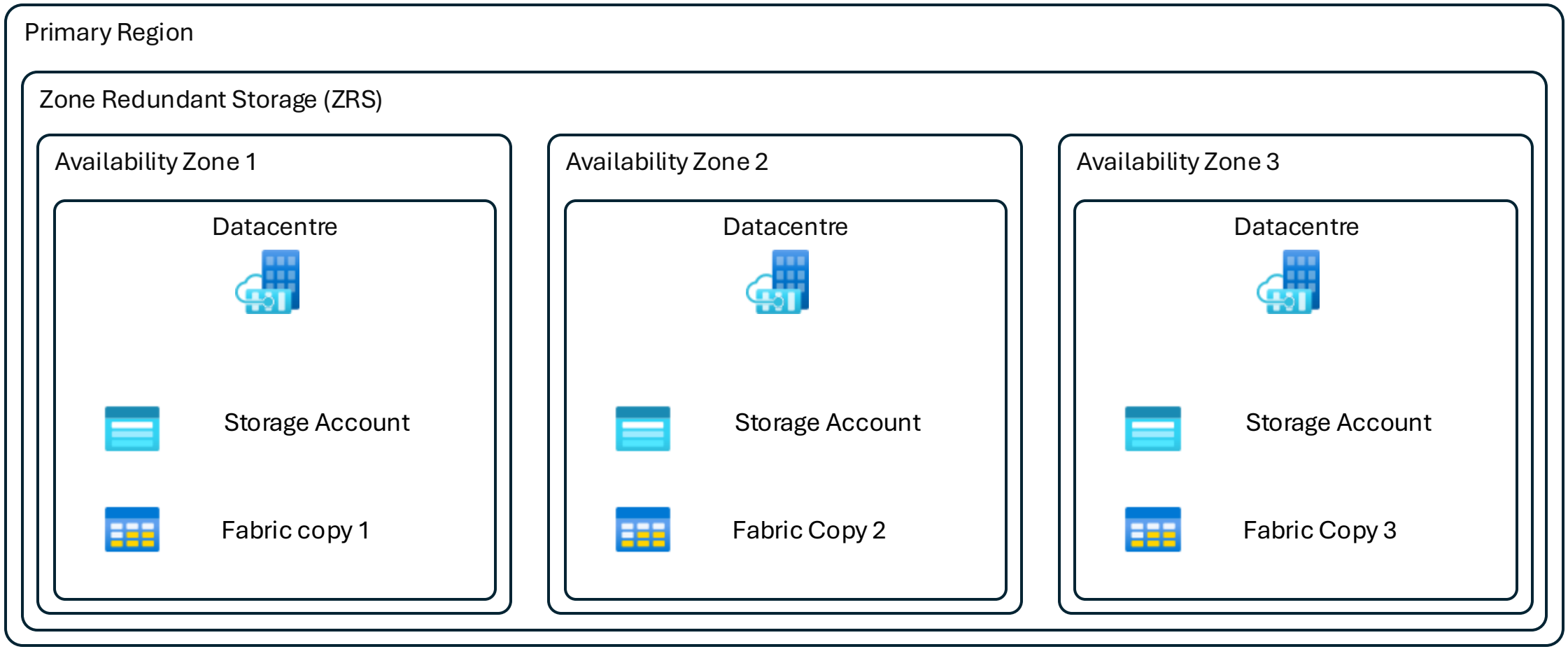


When a report or dashboard is shared, those it is shared with can view it and interact with it but can't edit it.
The recipients see the same data in their reports and dashboards. They also get access to the entire underlying semantic model, unless row-level security (RLS) is applied to it.
The coworkers shared with can reshare with their coworkers if this permission is enabled.

Free licence and above required to view content from K&E Premium capacity

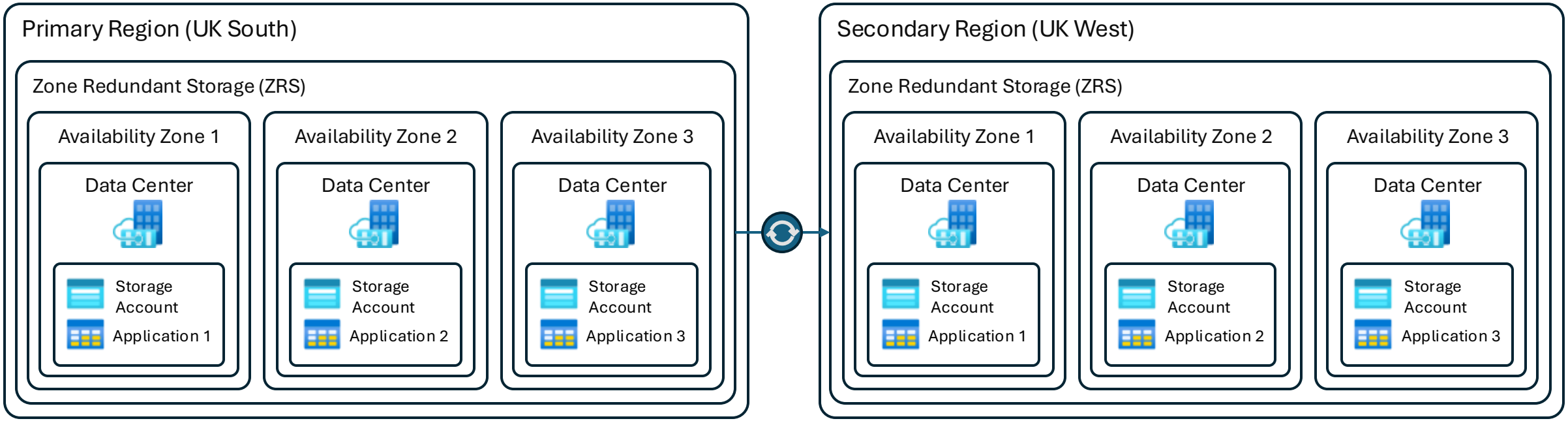
Availability Zones - Zone Redundant Storage (ZRS)

Replication of data across availability zones in the primary region

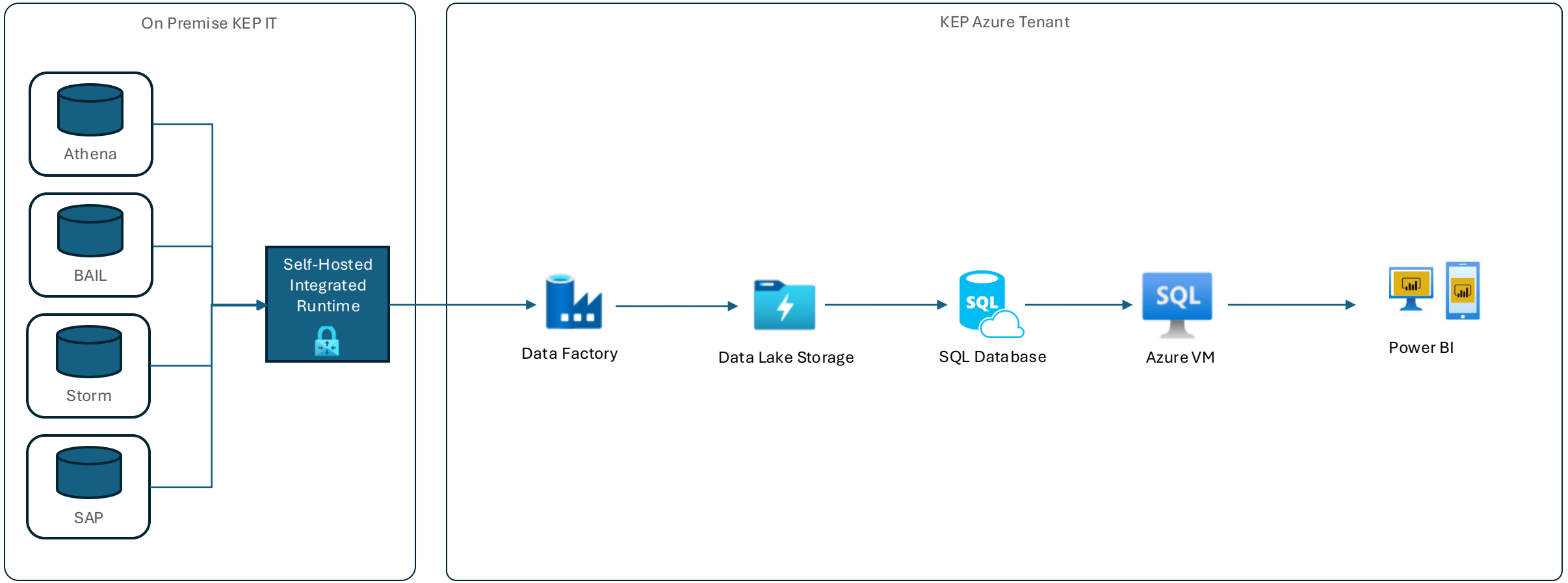


Availability Zones - Zone Redundant Storage (ZRS) with High Availability (HA)

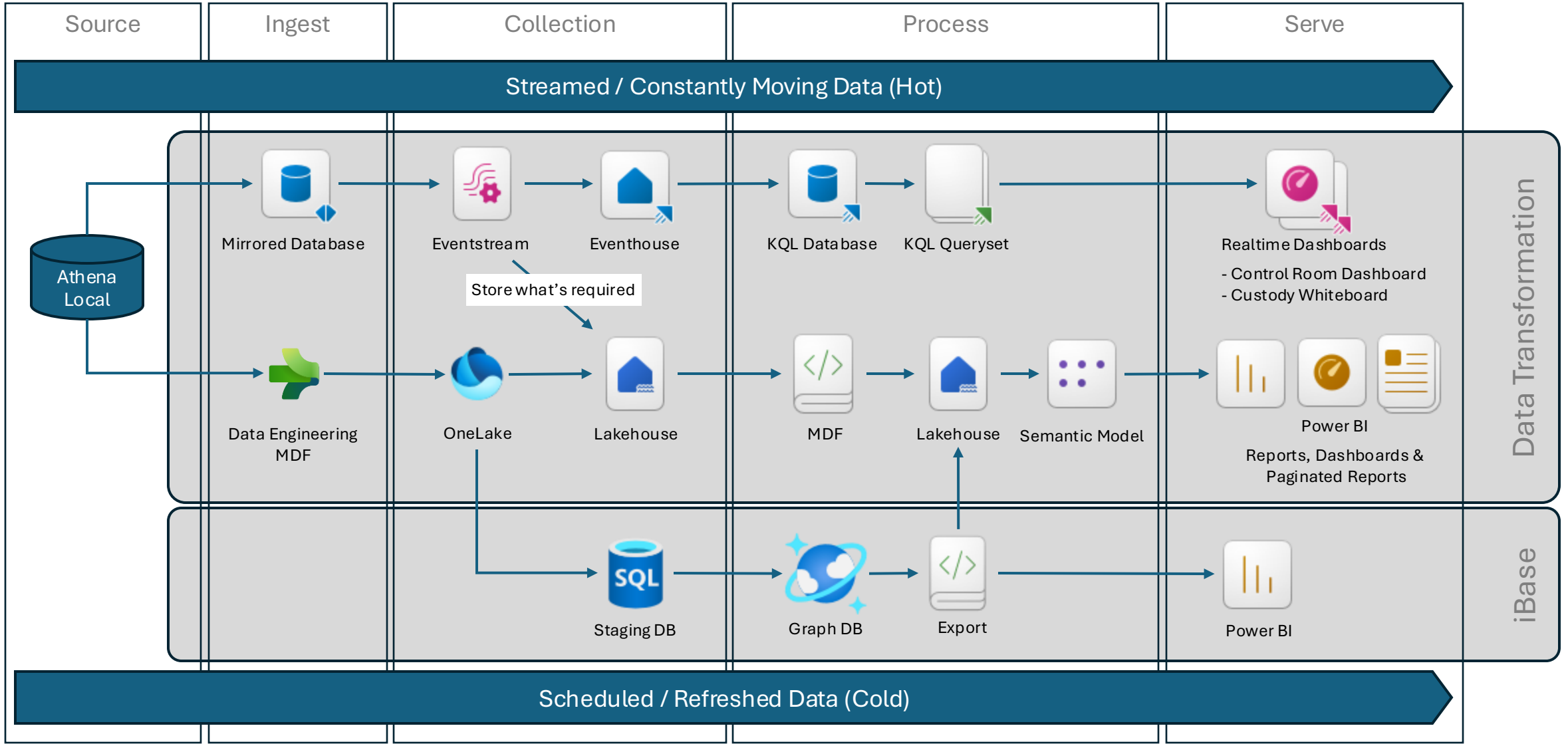
High Availability replication of data across availability zones in a secondary region for Disaster Recovery



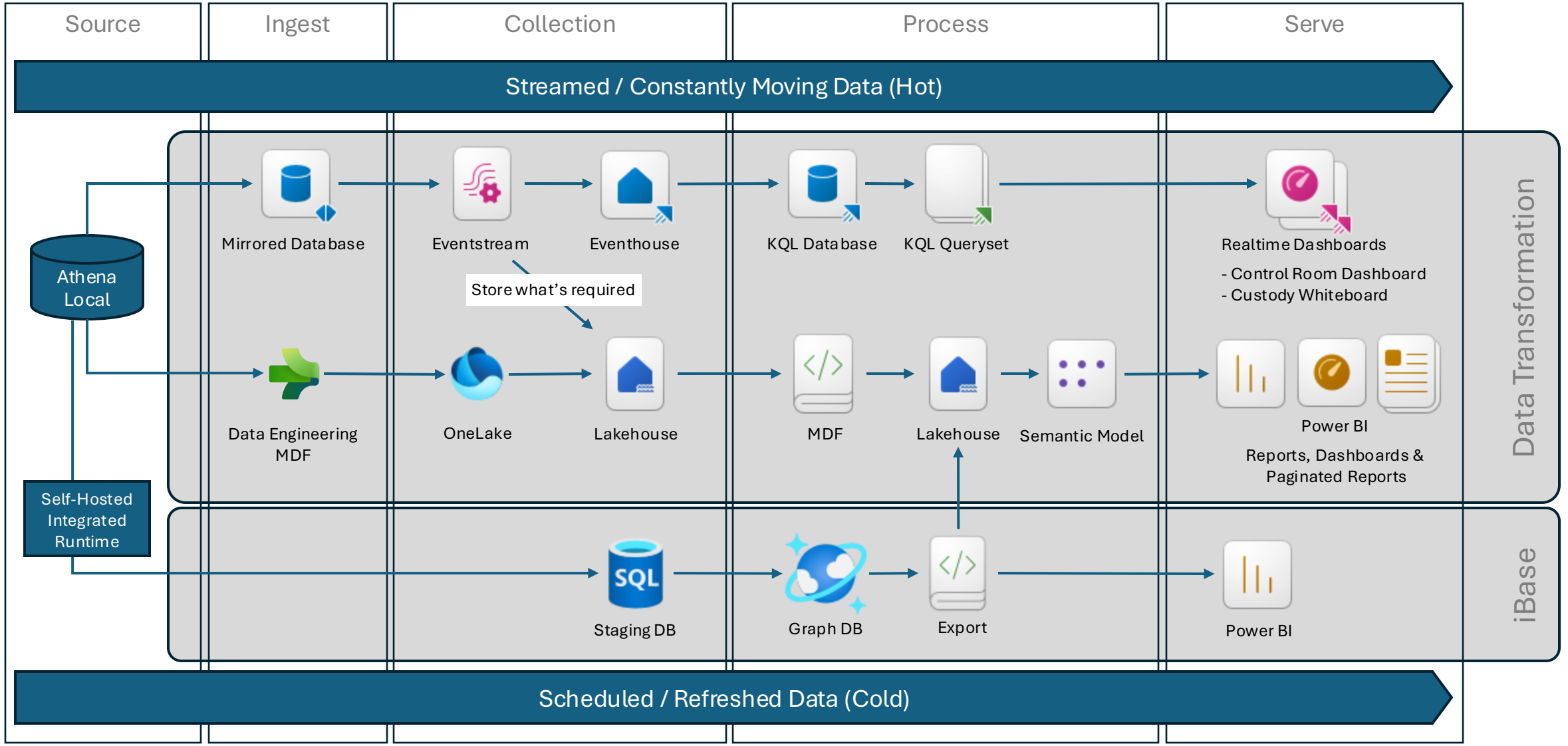
Data Flows - A4E



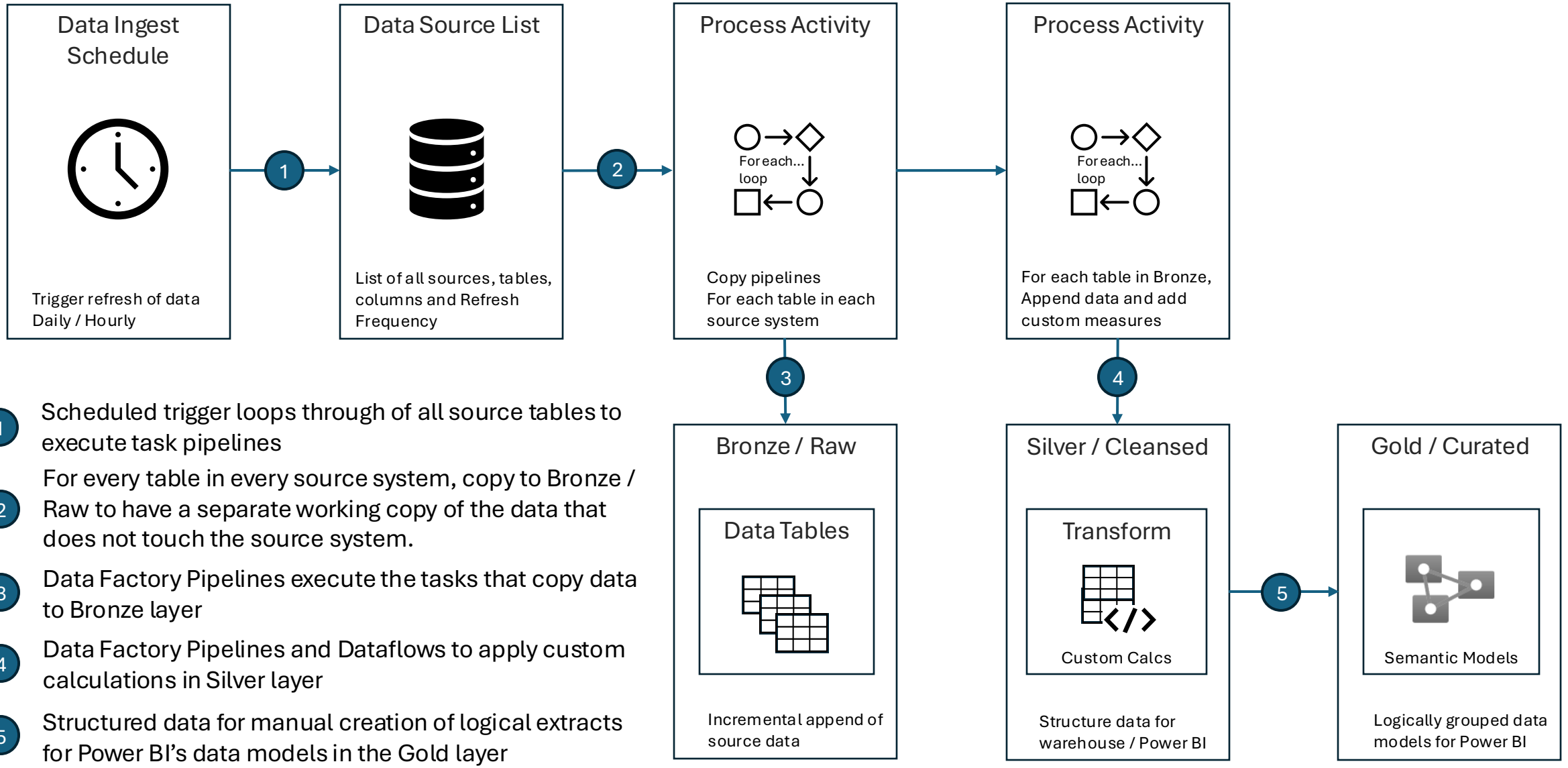
Athena - Scheduled / Streamed Data Across Shared Subscriptions



Athena - Scheduled / Streamed Data Across Separate Subscriptions



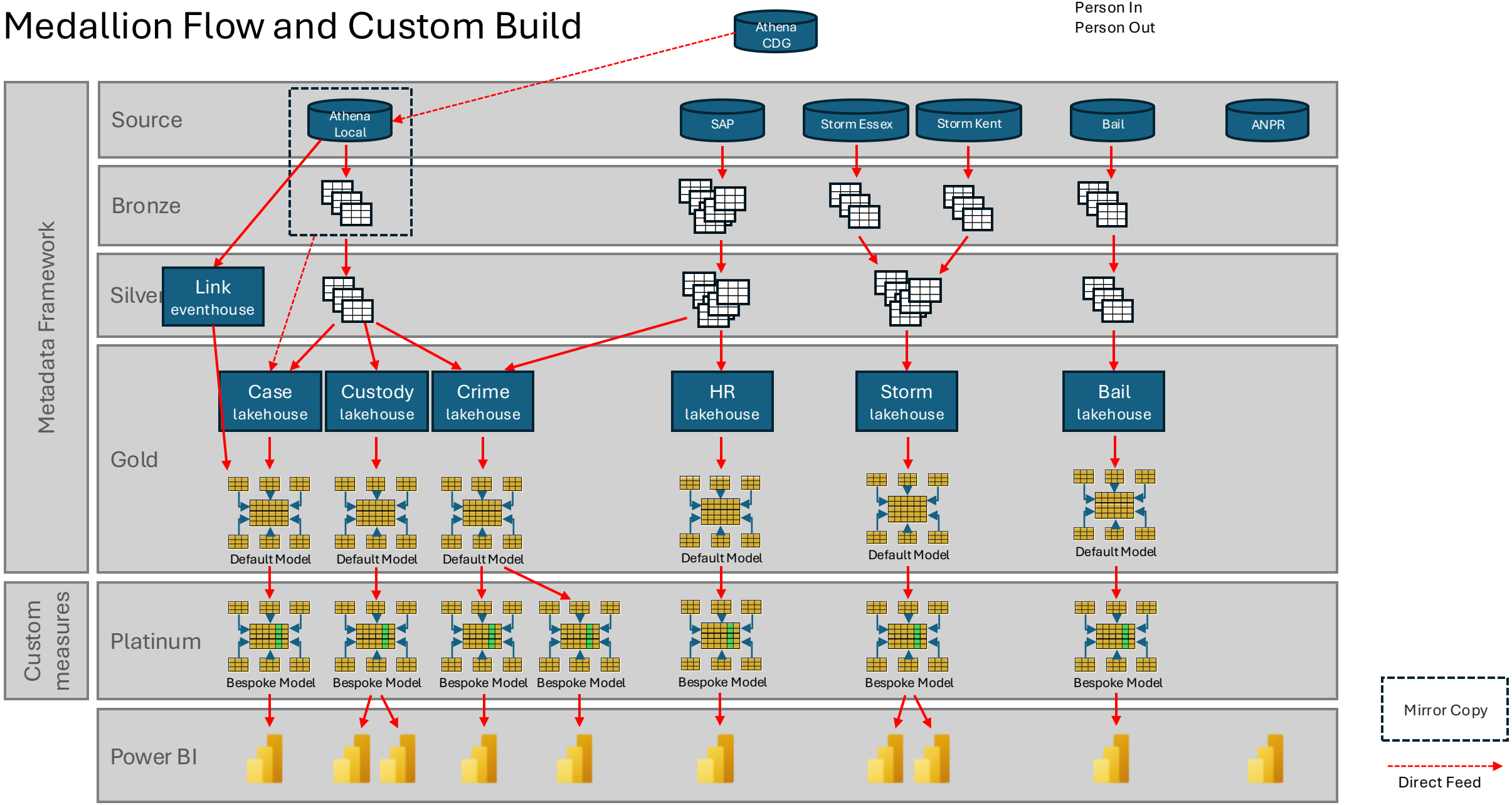
Metadata Framework



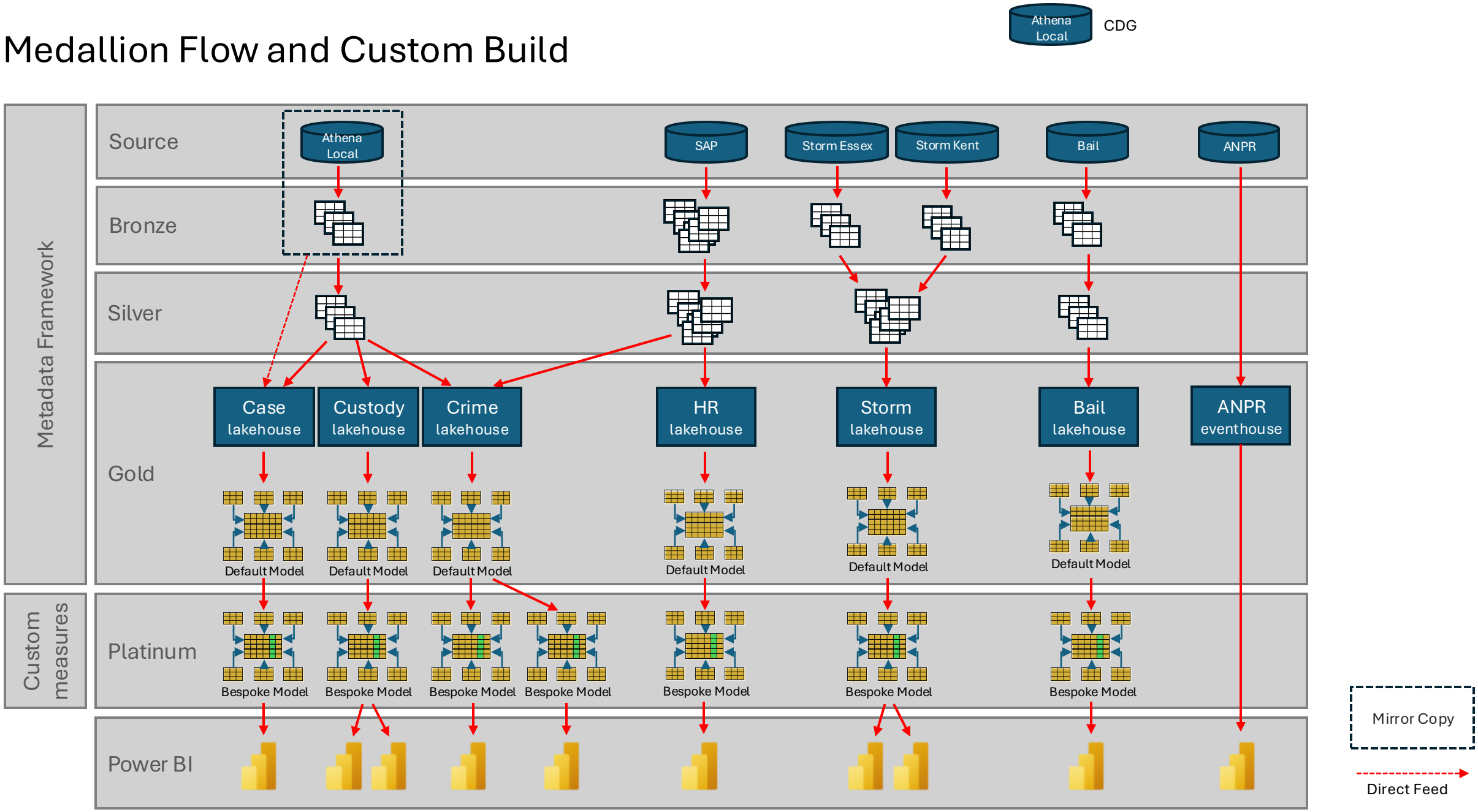
1. Scheduled trigger loops through of all source tables to execute task pipelines
2. For every table in every source system, copy to Bronze / Raw to have a separate working copy of the data that does not touch the source system.
3. Data Factory Pipelines execute the tasks that copy data to Bronze layer
4. Data Factory Pipelines and Dataflows to apply custom calculations in Silver layer
5. Structured data for manual creation of logical extracts for Power BI's data models in the Gold layer

Medallion Flow and Custom Build

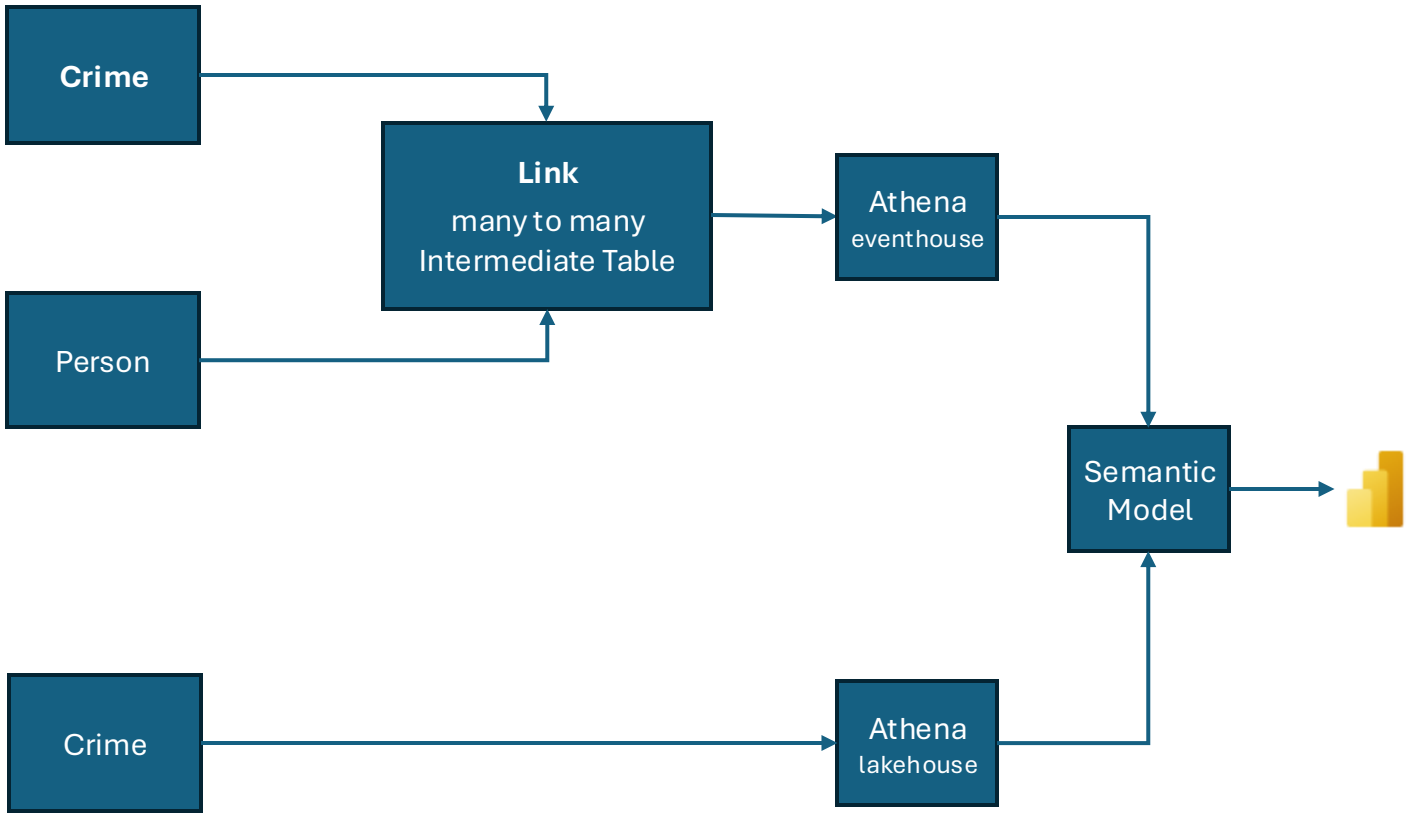
Local Audit Delta
Person In
Person Out



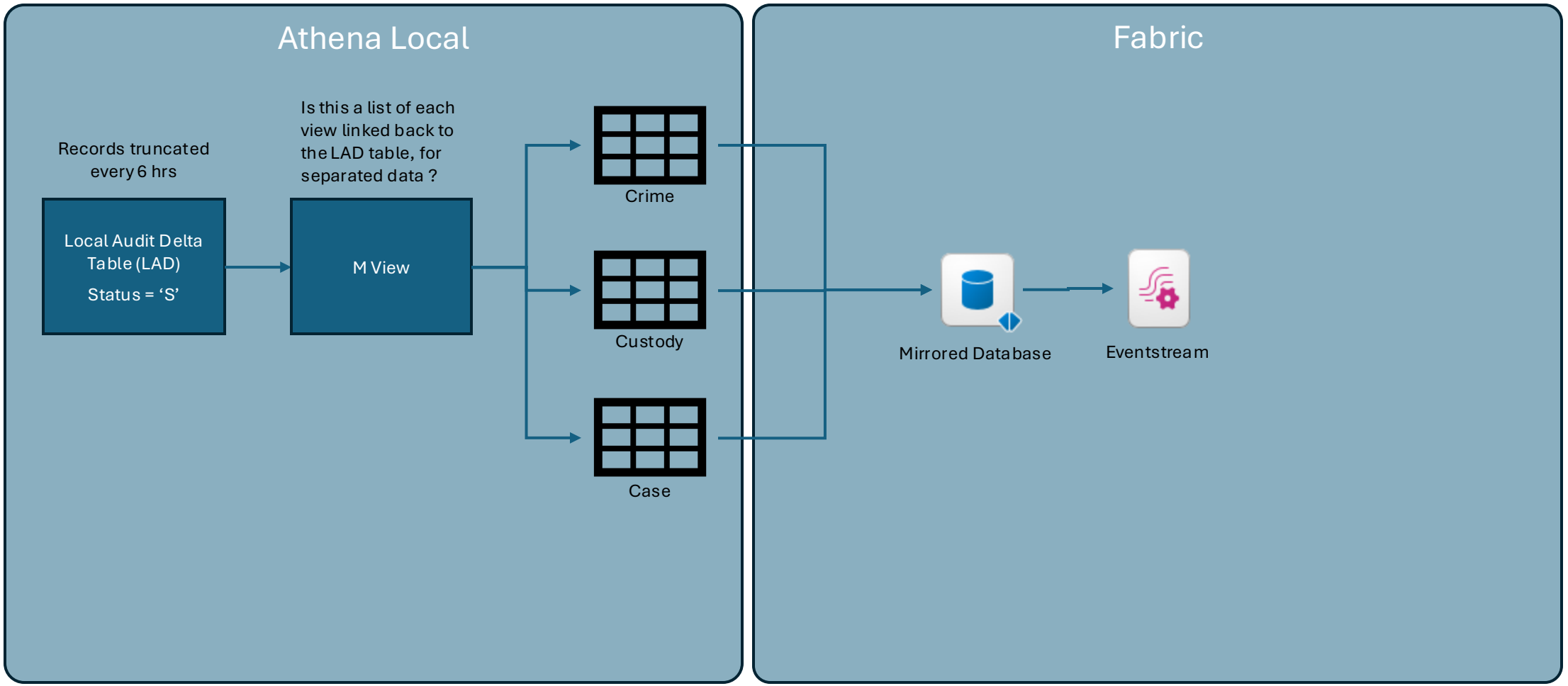
Medallion Flow and Custom Build



Athena


Crime
Link
many to many
Intermediate Table
Athena
eventhouse
Person
Semantic
Model
Crime
Athena
lakehouse

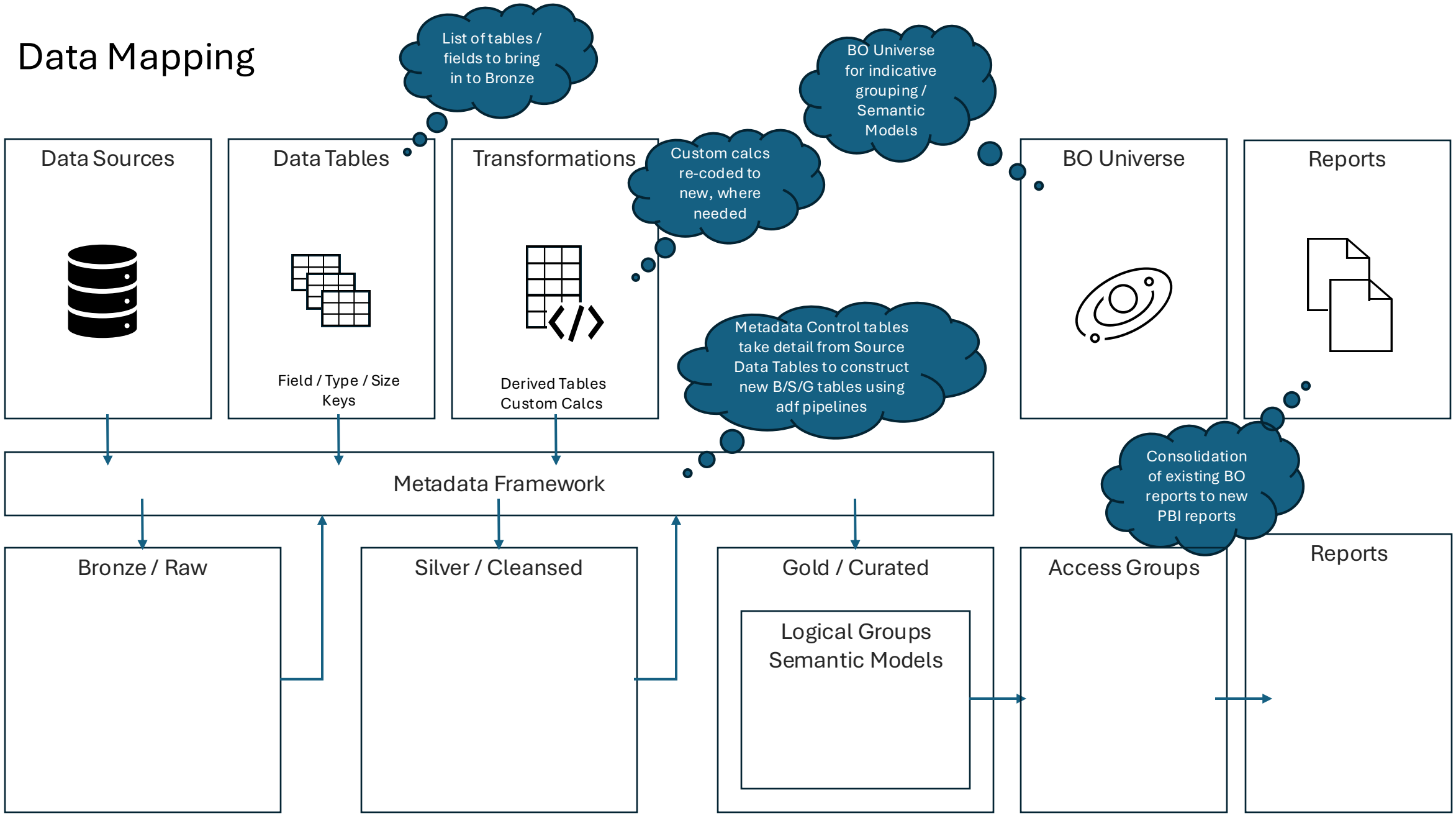
Athena Ingest



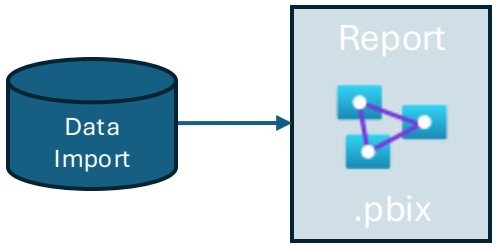
Refresh Plan

Data Source	Refresh Frequency	00:00				01:00				02:00				03:00				04:00				05:00				06:00				07:00				08:00				09:00				10:00				11:00			
		00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45	00	15	30	45
Athena	Hourly		■				■				■				■				■				■				■				■				■				■				■				■		
Storm	Hourly			■				■				■				■				■				■				■				■				■				■				■				■	
Bail	Daily	■	■																																													
SAP	Daily	■	■																																												

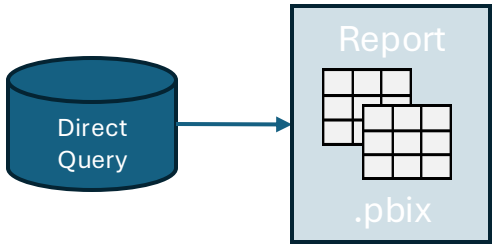
Data Mapping



Semantic Model - Power BI

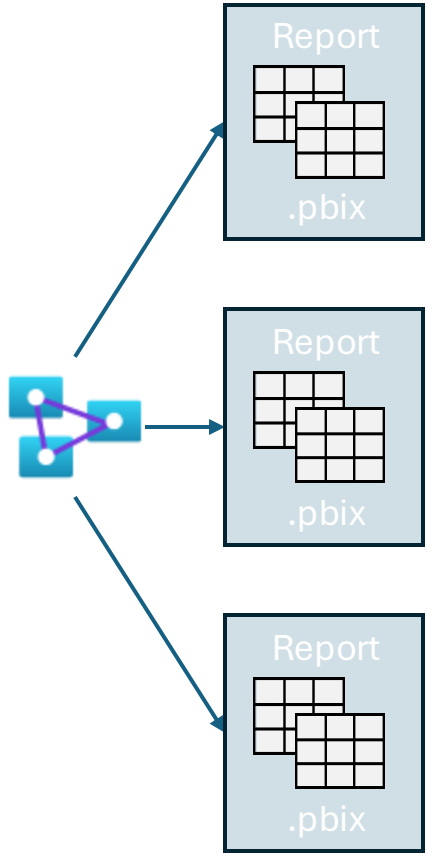


Data Import
- Report file has its own Semantic data model embedded
- Multiple reports have duplicated data models when reporting from the same data source
- Data model is efficient when reading large datasets
- Updates to the data carried out in each report



Direct Query
- Report file linked to and served by source database
- Multiple reports share the same data source
- Data model is structured to the source system, not Power BI's and can be inefficient when reading large datasets
- Updates to the data carried out centrally on the data source

Semantic Model
- Multiple report files linked to a semantic data model on the gold layer
- Multiple reports share the same data model
- Data model efficient when reading large datasets
- Updates to the data carried out centrally on gold layer's semantic model



Key

Data Table

Semantic Model (Related Tables)

Access Control - Hierarchies

Management groups

Billing,
Governance,
Security

Subscriptions

Sandbox
Pre-Prod
Production

Resource groups

rg-kpep-uks-s-fabric-001
rg-kpep-uks-d-fabric-001
rg-kpep-uks-t-fabric-001
rg-kpep-uks-u-fabric-001
rg-kpep-uks-p-fabric-001

Resources

Fabric F2 (sandbox)
Fabric F4 (pre-prod)
Fabric F64 (prod)

Azure / Fabric Container Hierarchy

Tenant

Subscription (Pre-Prod / Prod)

Domain (Dev / QA / UAT / Live)

Workspace

Items

Data Warehouses
Data Pipelines
Semantic Models
Reports
Dashboards

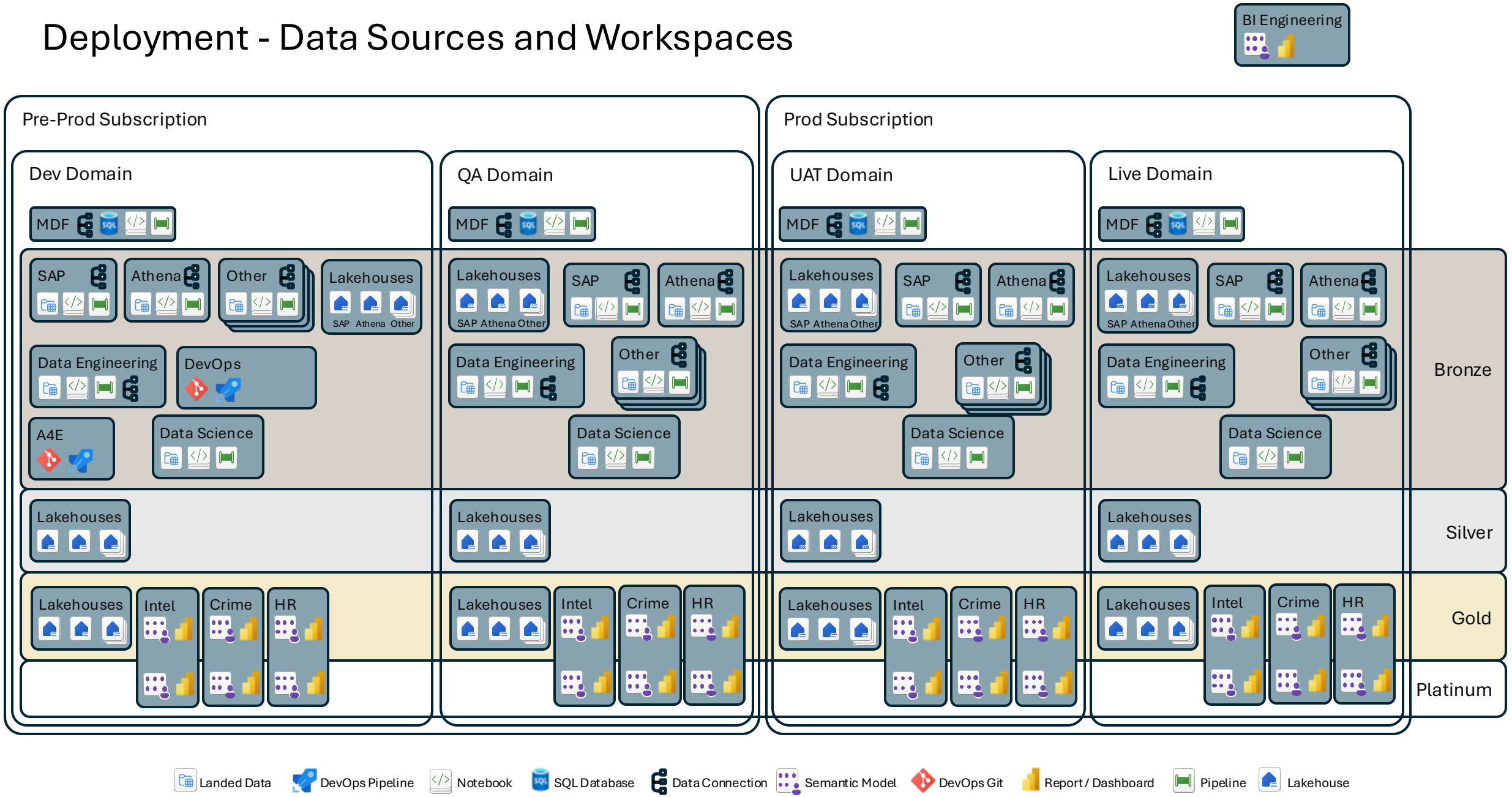
Feature
Development
QA
Production

Logical groups
(as BO Universes)

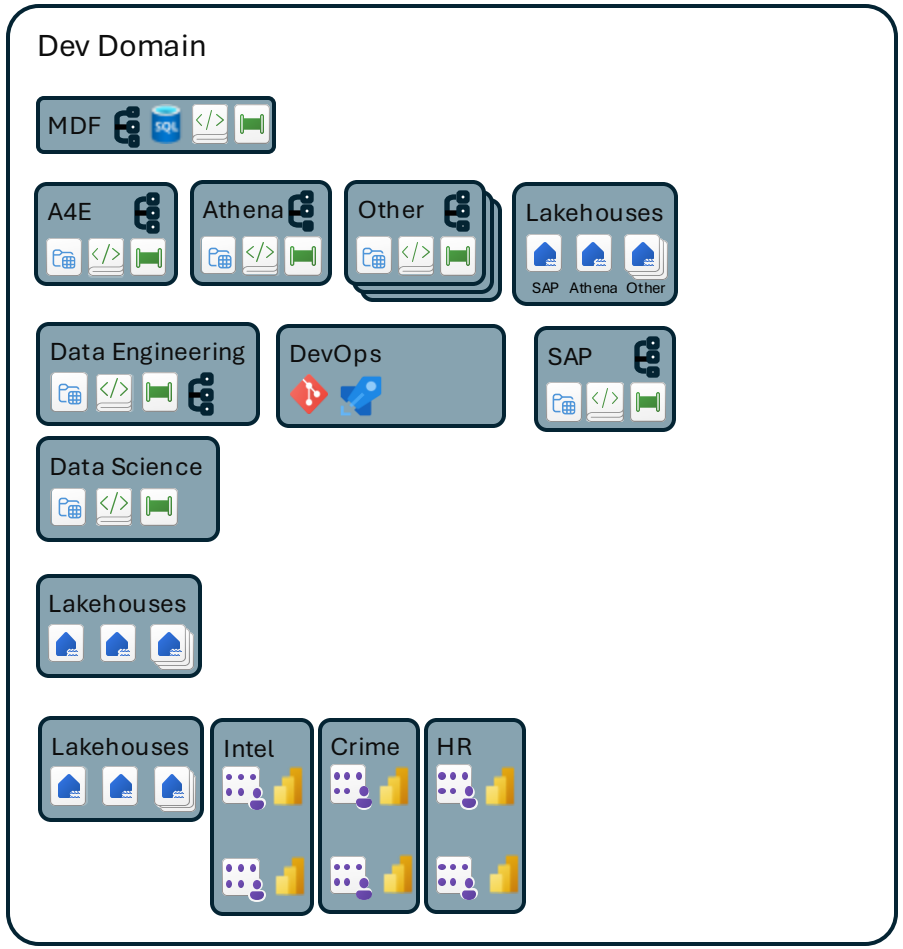
F64 SKU Tier

Azure Cost Control
DevOps Git Repo

Deployment - Data Sources and Workspaces



Deployment - Data Sources and Workspaces



ADF
adf-mdf-dev
adf-dataengineering-dev
adf-dtran-dev

Workspaces
ws-athena-dev
ws-dataengineering-dev
ws-datascience-dev
ws-a4e-dev
ws-loganalytics

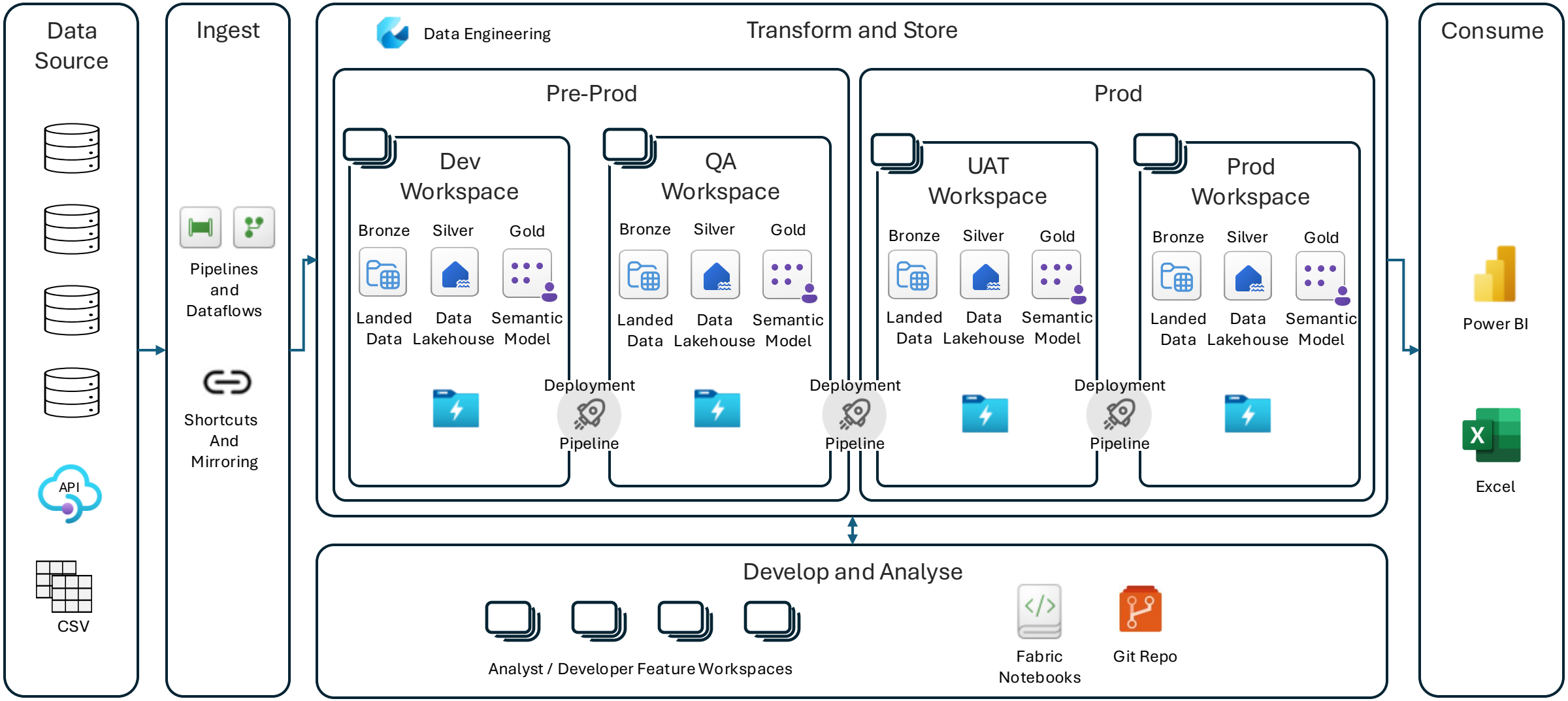
Purview
pview-dataengineering-dev

Lakehouses
LH_100_Bronze
LH_200_Silver
LH_300_Gold
LH_400_Platinum

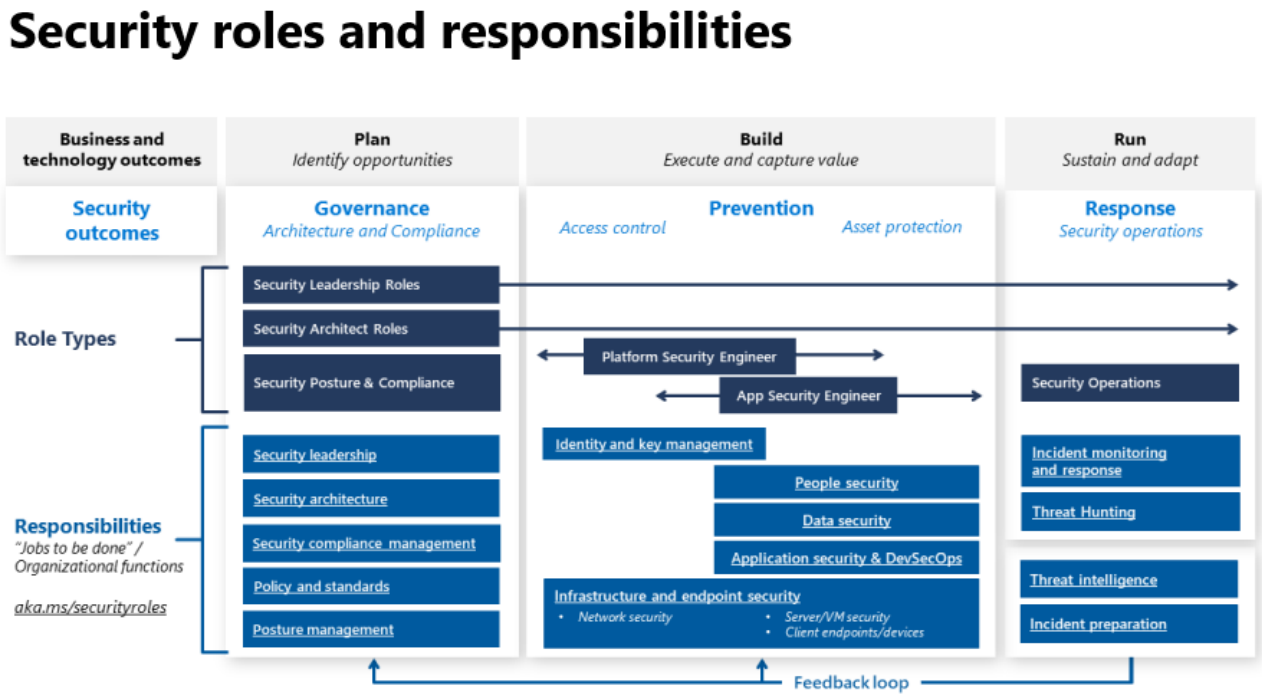
Pipelines - folder structure
PL_Metadata_Grandparent
PL_Metadata_Parent

Notebooks
NB_05_Bail_Bronze

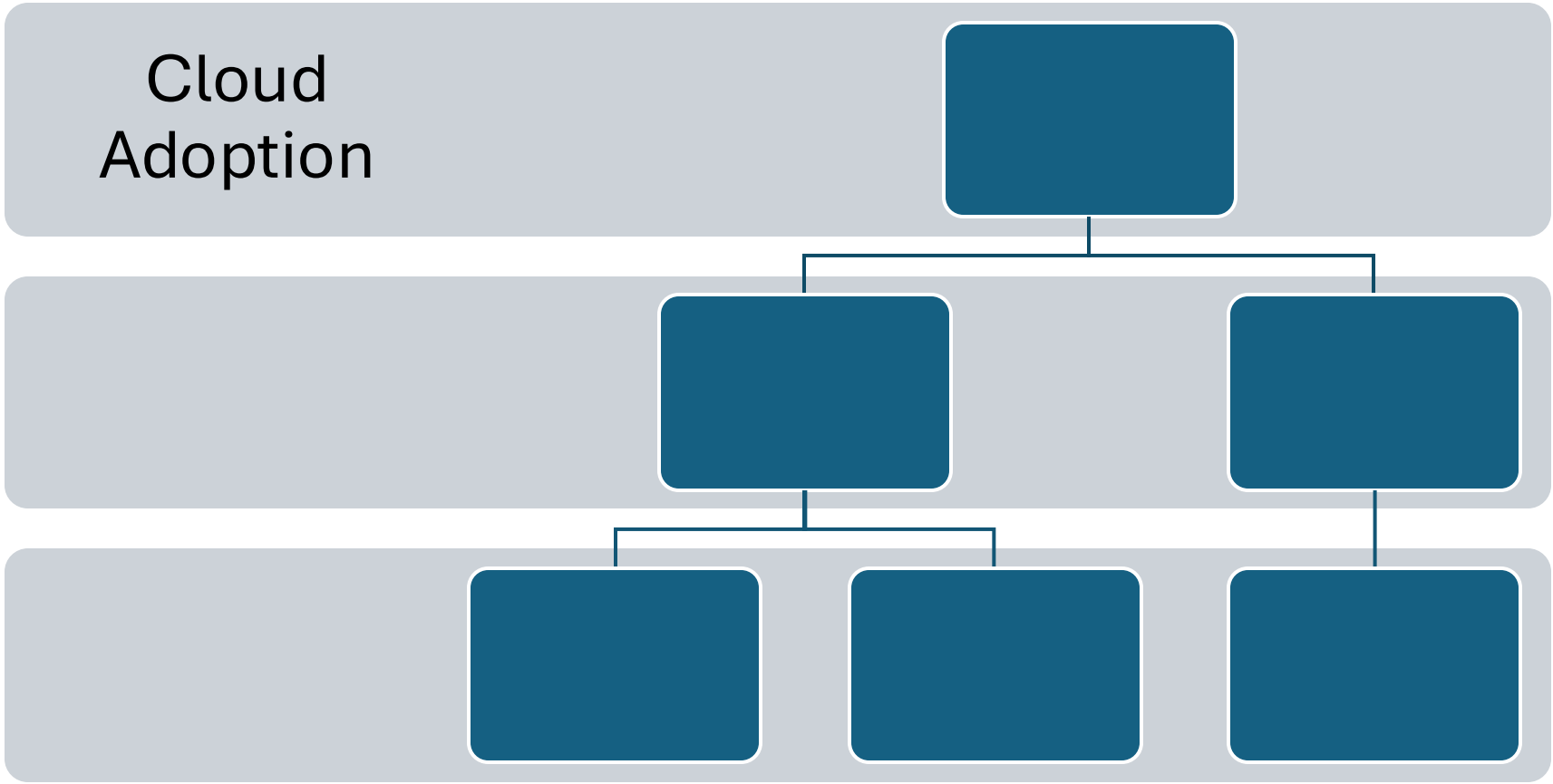
Deployment Across Workspaces - DevOps and Fabric



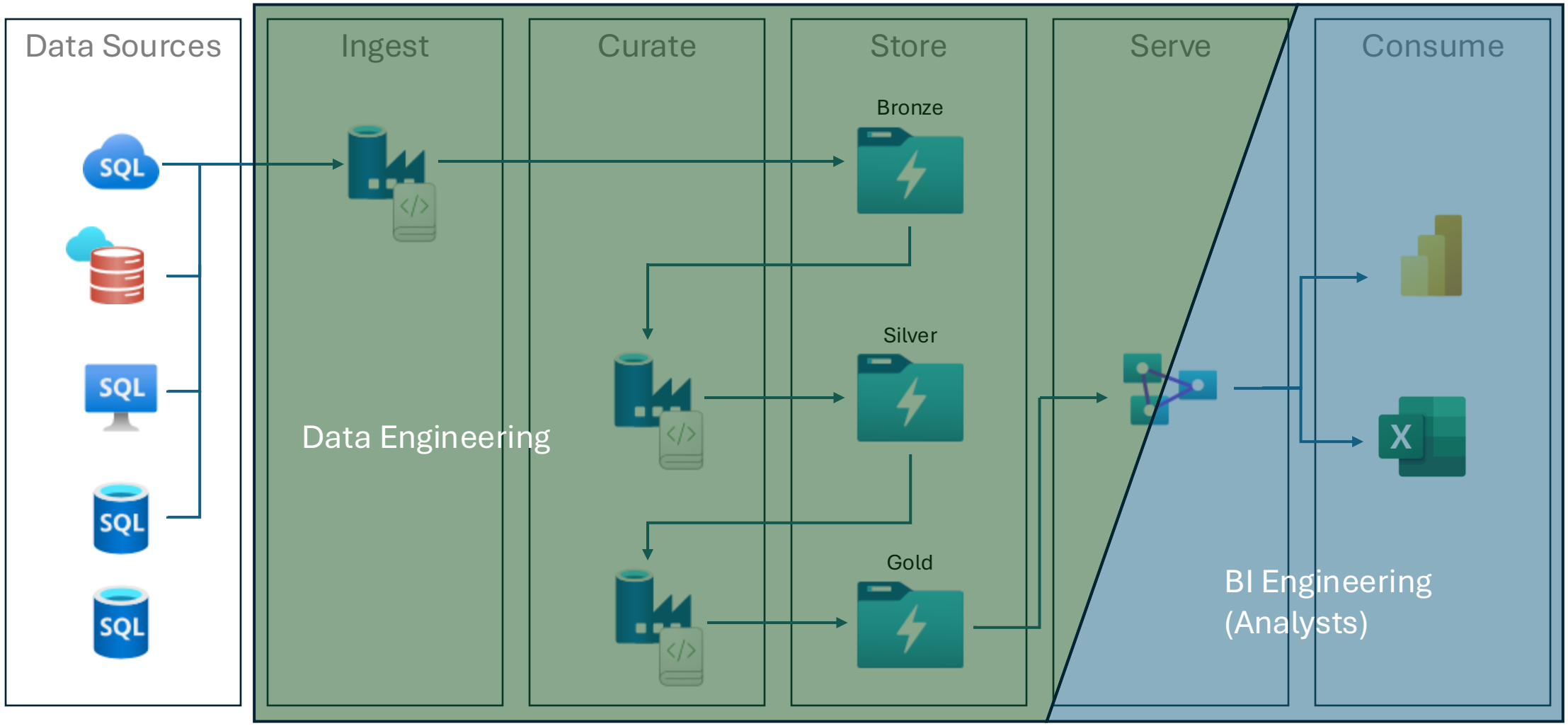
Security Roles



Proposed Team Collaboration and Structure



Medallion Architecture - Data Engineering / BI (Analyst) Team Split



Testing Strategy

1. Create Test Strategy
Design what is to be tested
Select appropriate schedule
Allocate test resource(s)

2. Task from DevOps
Design what is to be tested
Select appropriate schedule
Allocate test resource(s)

3. Develop the Test(s)
Create test plans
- Cyclic / on demand (data)
- One-off / scheduled (infra)

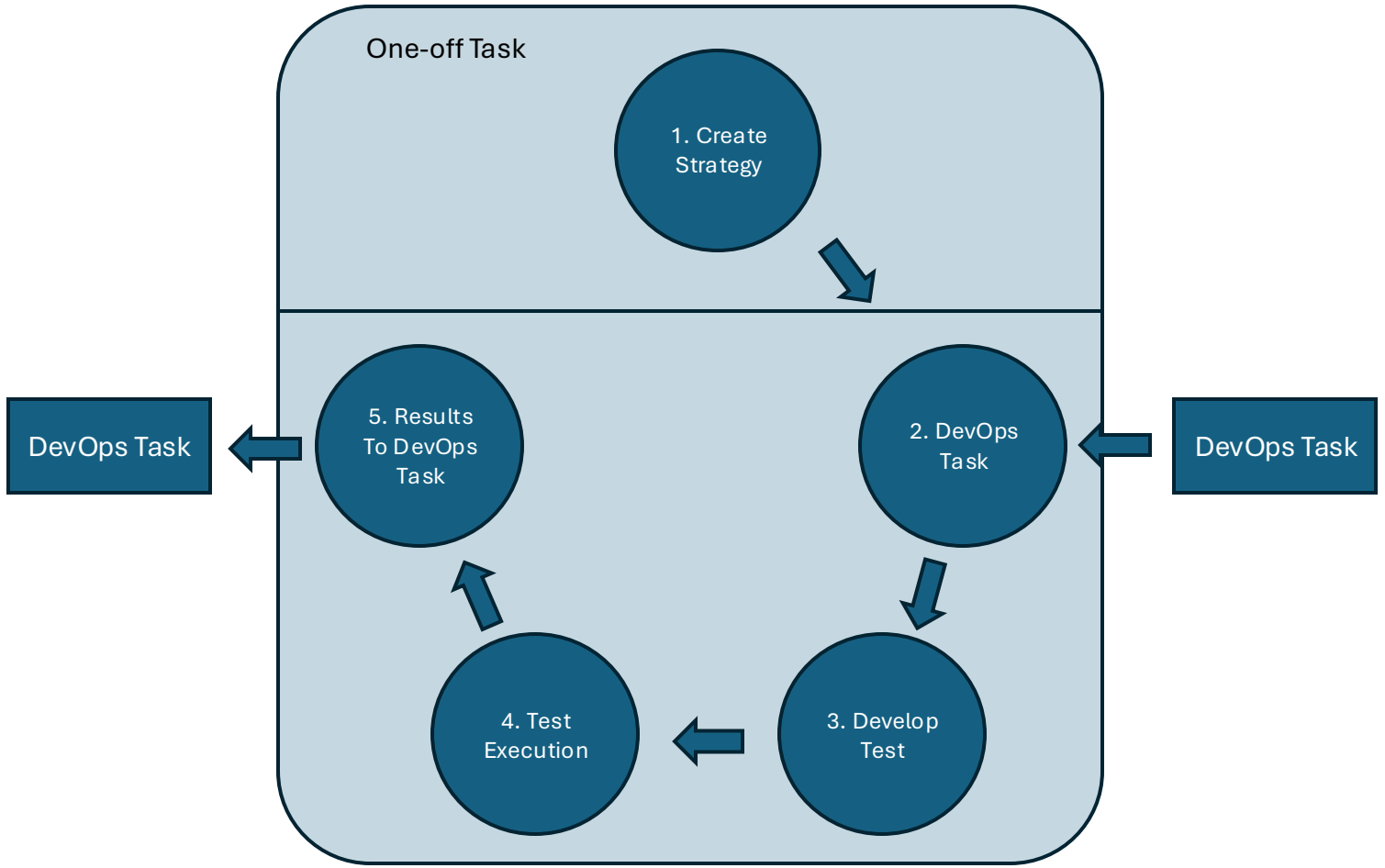
Create test scripts
- Azure monitoring (infra / platform)
- Requirements from business (data / BI)
- Existing report comparison (BO to PBI)

Select appropriate data
- Test data (Pre-Prod)
- Live data (UAT)

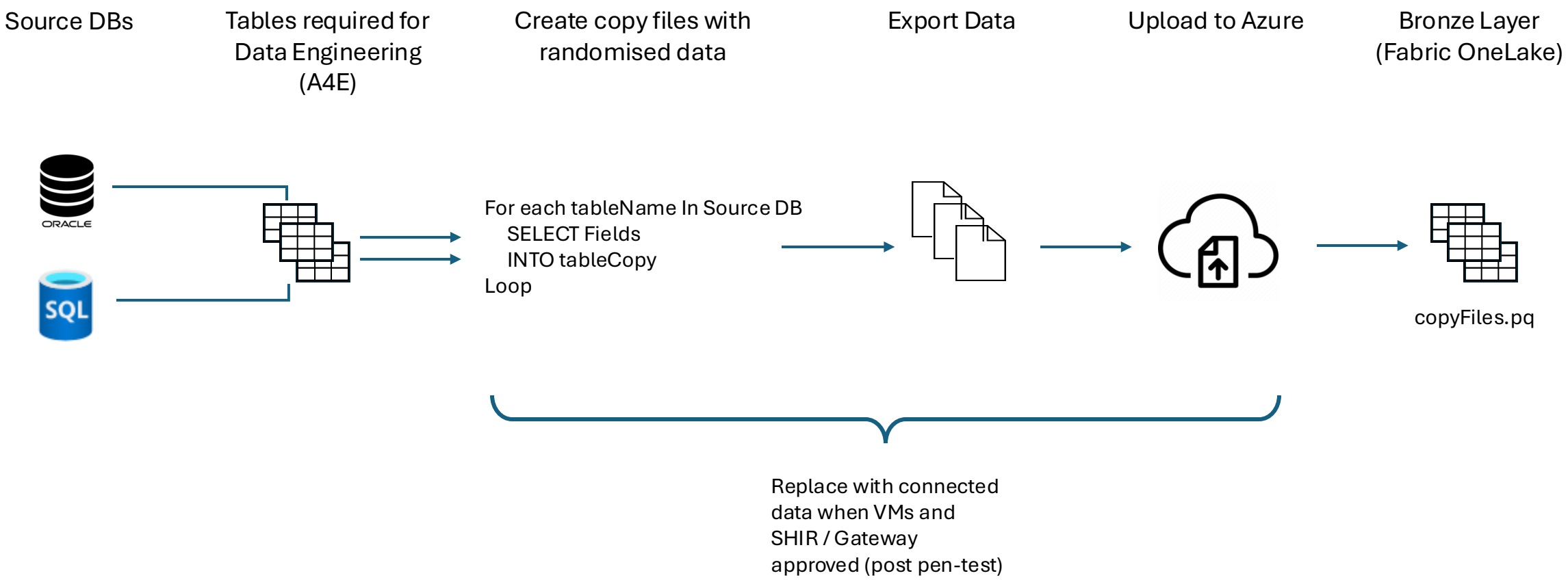
4. Execute the Test(s)
Record results of items on test scripts

5. Apply Test Results to the DevOps Task
If test criteria met
- Update in DevOps and apply push to next environment

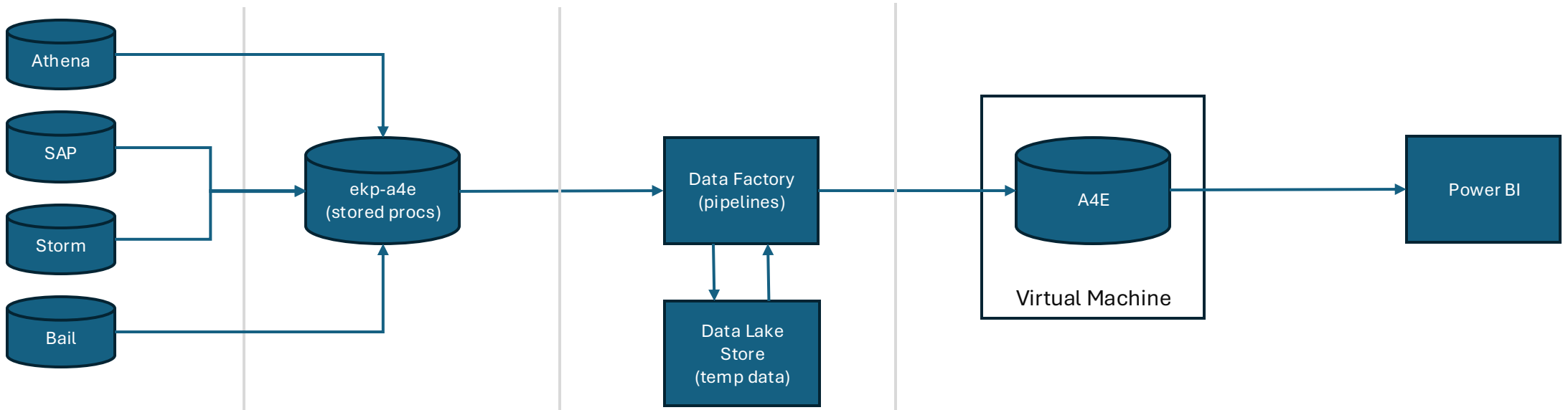
Else
- Feed back to devs to apply corrections



Development Data for use in Pre-Prod



A4E - As-is Architecture

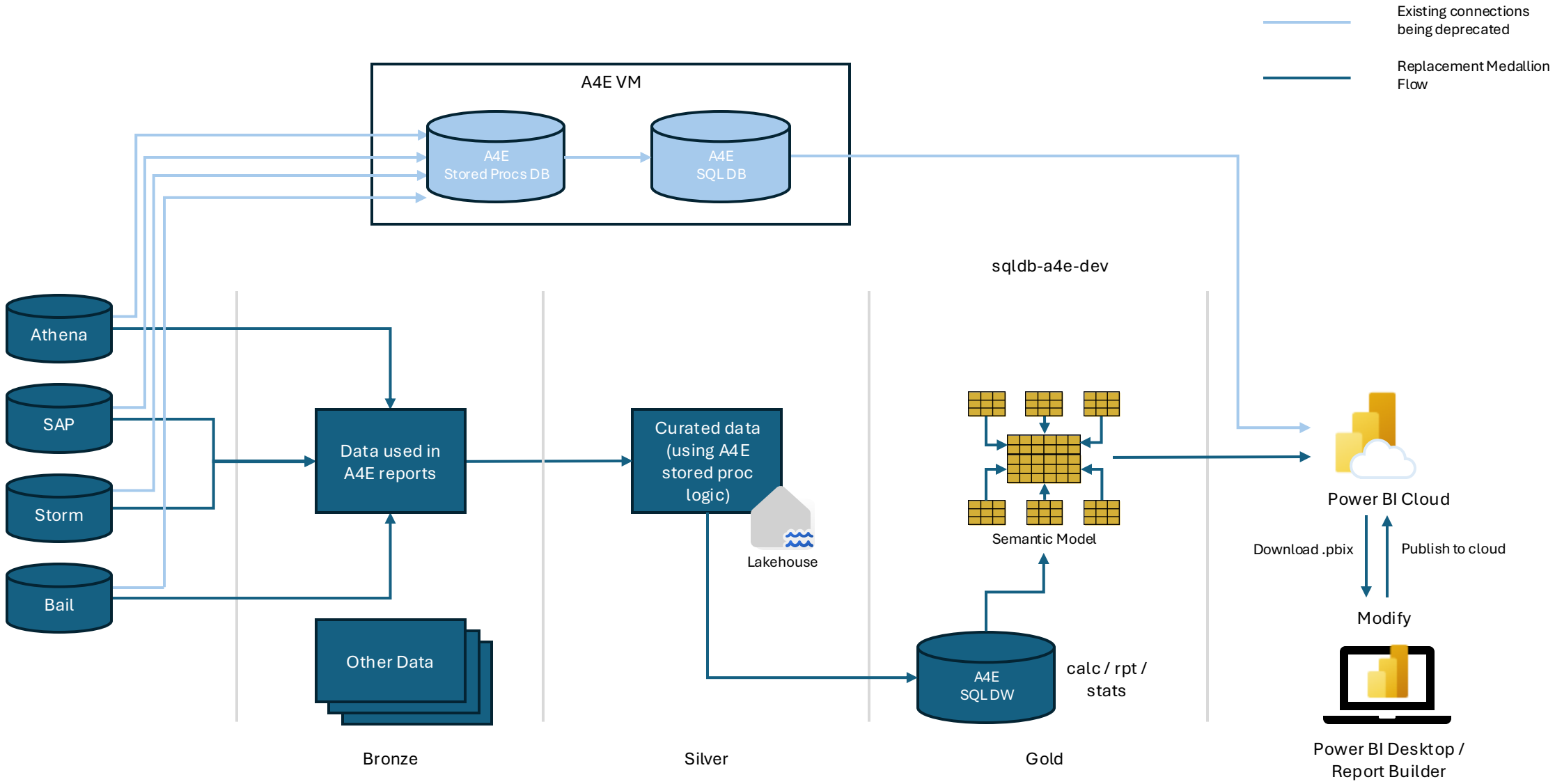


Multiple stored procedures including nested procedures running inside
All scripted (MS-BI pattern)

Pipelines execute stored procedures
Temp data written to data lake

Power BI Reports served directly from A4E database via gateway connection

A4E - Replacement Data Ingest to Medallion



Transformation Process – How does it work ?

Ingredients

Extract
Source systems' data

Source Data

Combine

Transform
Data
Factory / Notebooks

Transform

Assemble

Load
Data structured for
reporting / analytics

Warehouse

Prepare

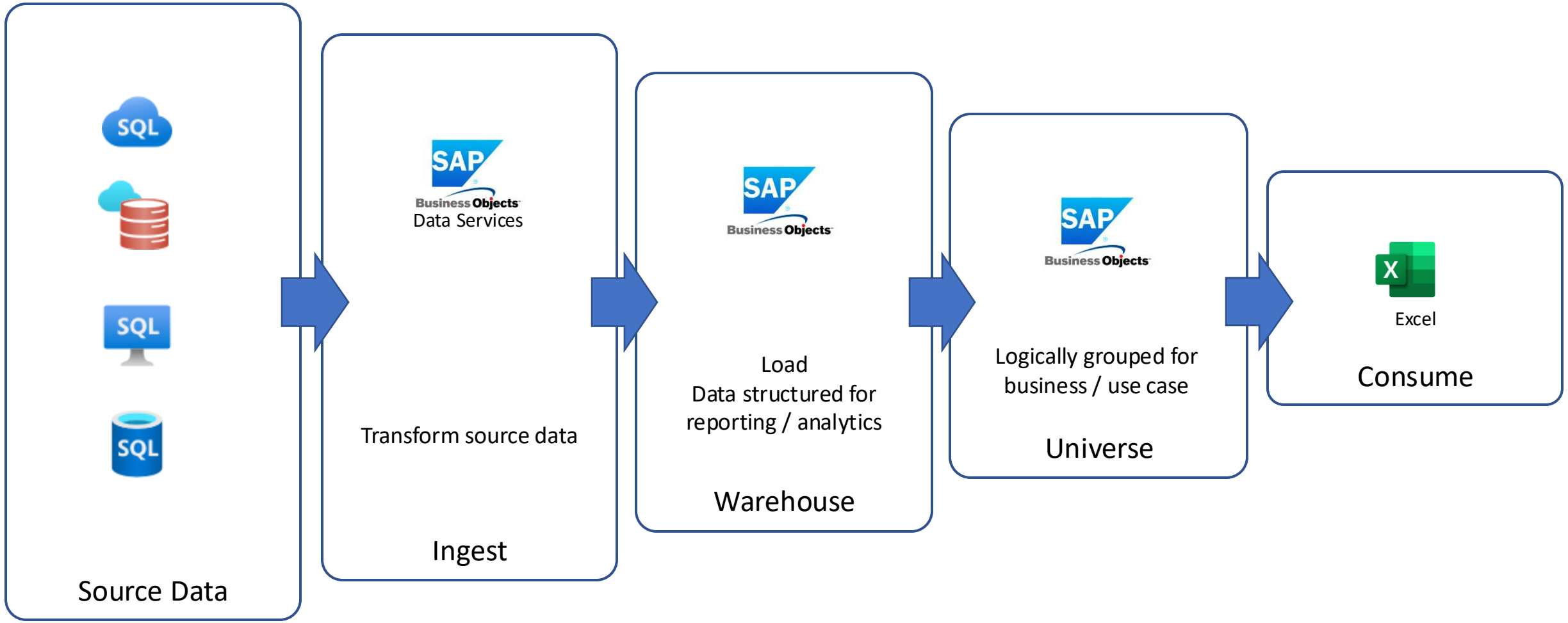
Logically grouped by
domain / business unit

Semantic Model

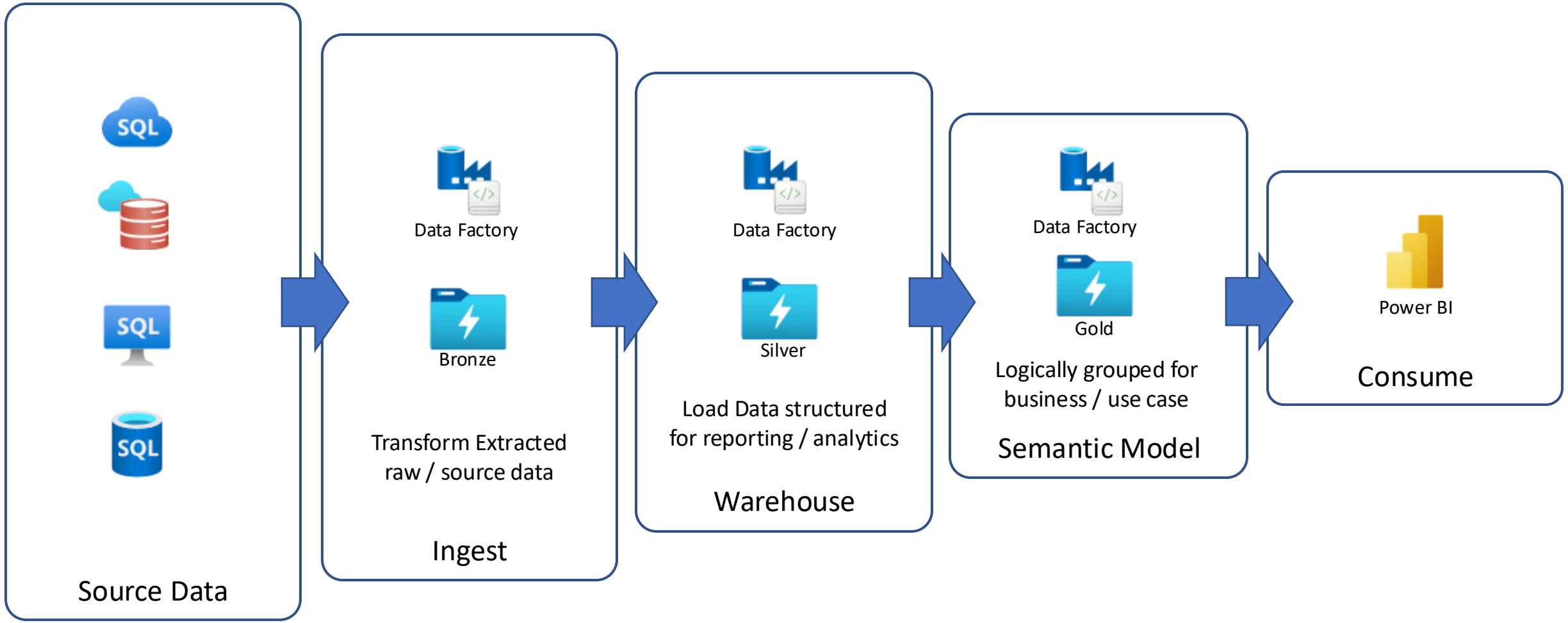
Serve

Power BI

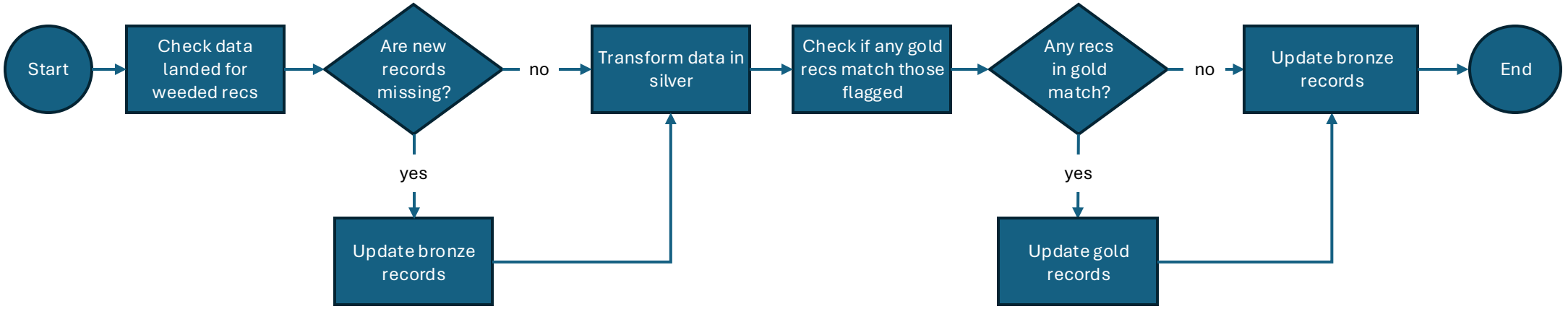
Transformation Process – How it currently works



Transformation Process – How it will work



Data Weeding Process - Medallion Flow



Compare records from latest bronze ingest against previously loaded bronze records

Update records with weeded flag

Silver process as normal, maintain the weeded flag

For each record in gold check for matches with the weeded flagged recs

Update the pii part of each matched record with 'Removed'

Delete / Update to 'Removed' all weeded recs

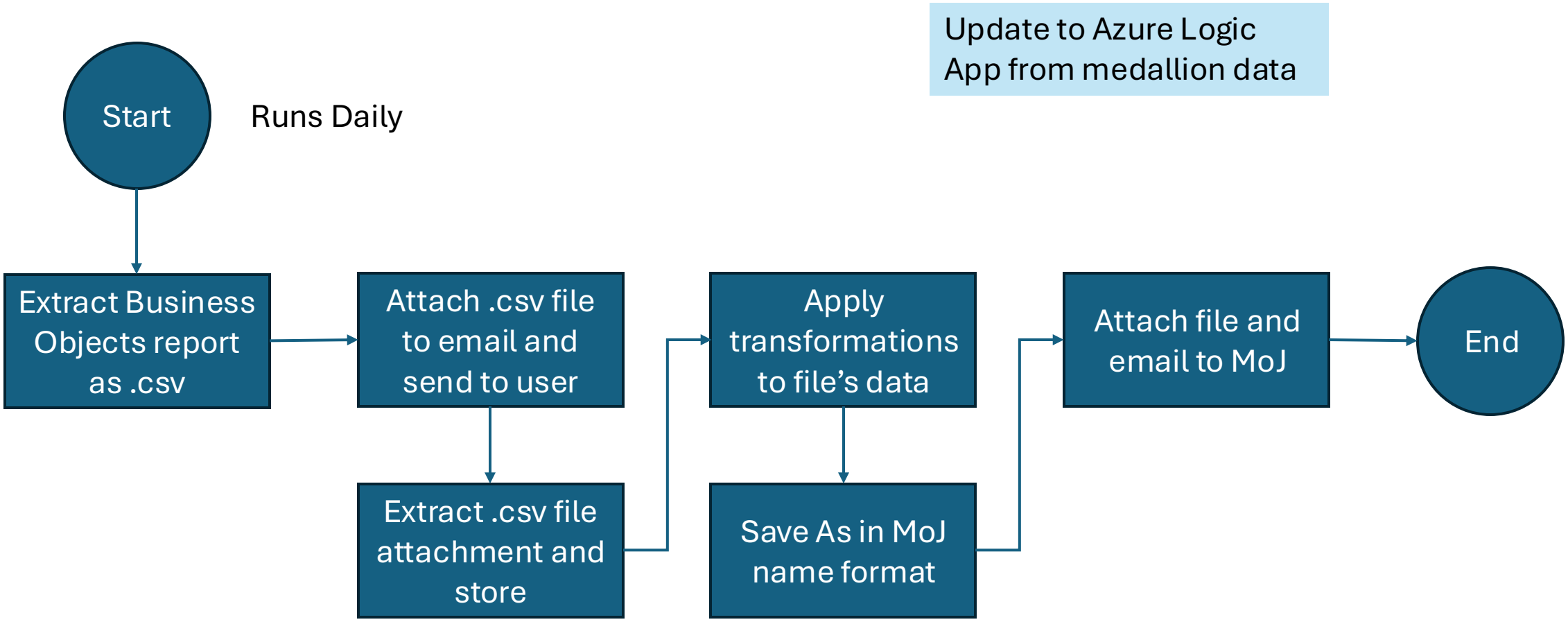
Regular process in the gold layer
Should be in line with current process
Ultimately marking a record as weeded - should sit on the main tables
SCD Type 2 - Historic is what we have now
SCD Type 4 - separate table of deleted records - possible inclusion

Do we move recs over 7years old to history table
Any rec ingested that is older than 7 years, what do we need to do ?

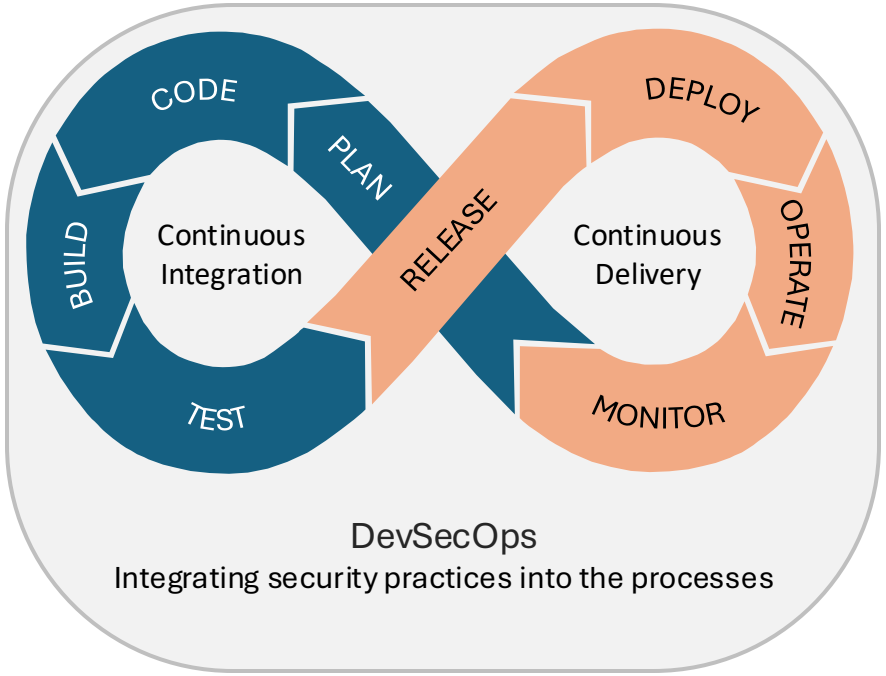
Source system data removed
- Check against previous refresh - bronze
- Update pii rec and overwrite

Flag the record as removed in bronze
- Process through to gold to update
- Check for SCD 4 historical recs and overwrite the pii

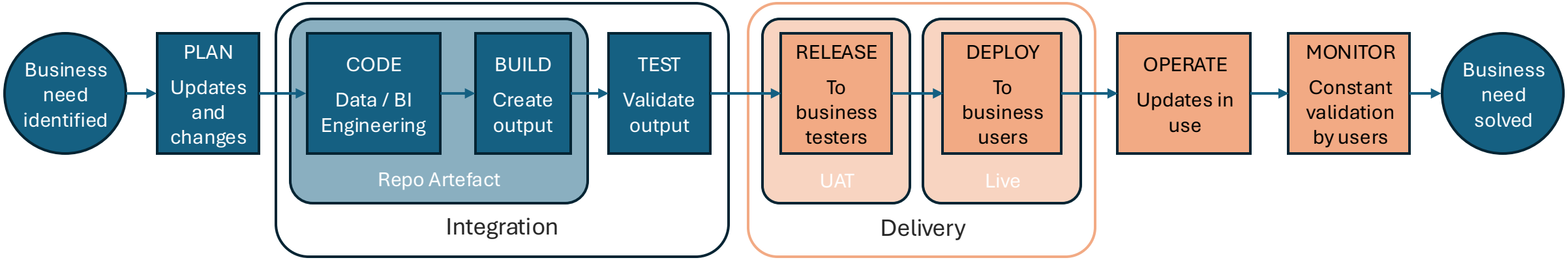
Logic App Flow - MoJ Daily Submission



DevOps Flow - CI/CD Process



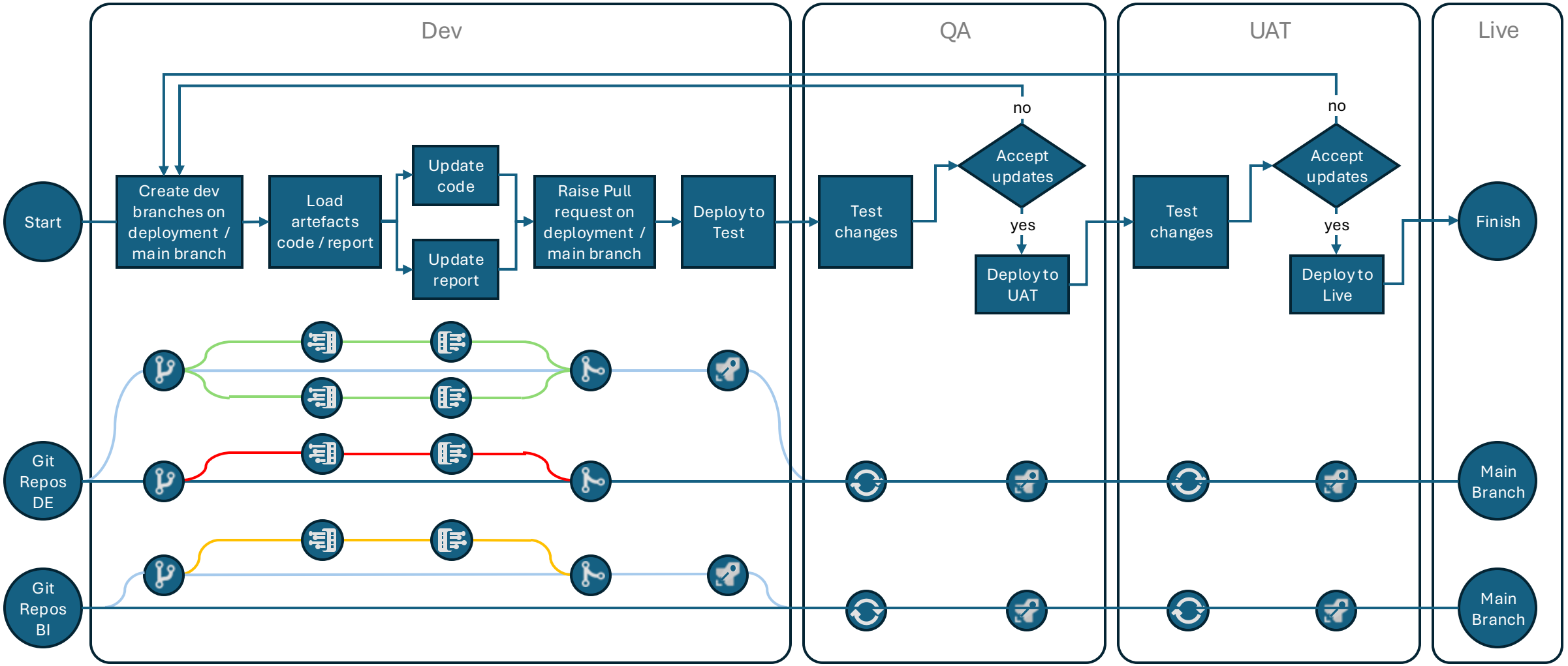
CI - Continuous Integration
CD* - Continuous Delivery



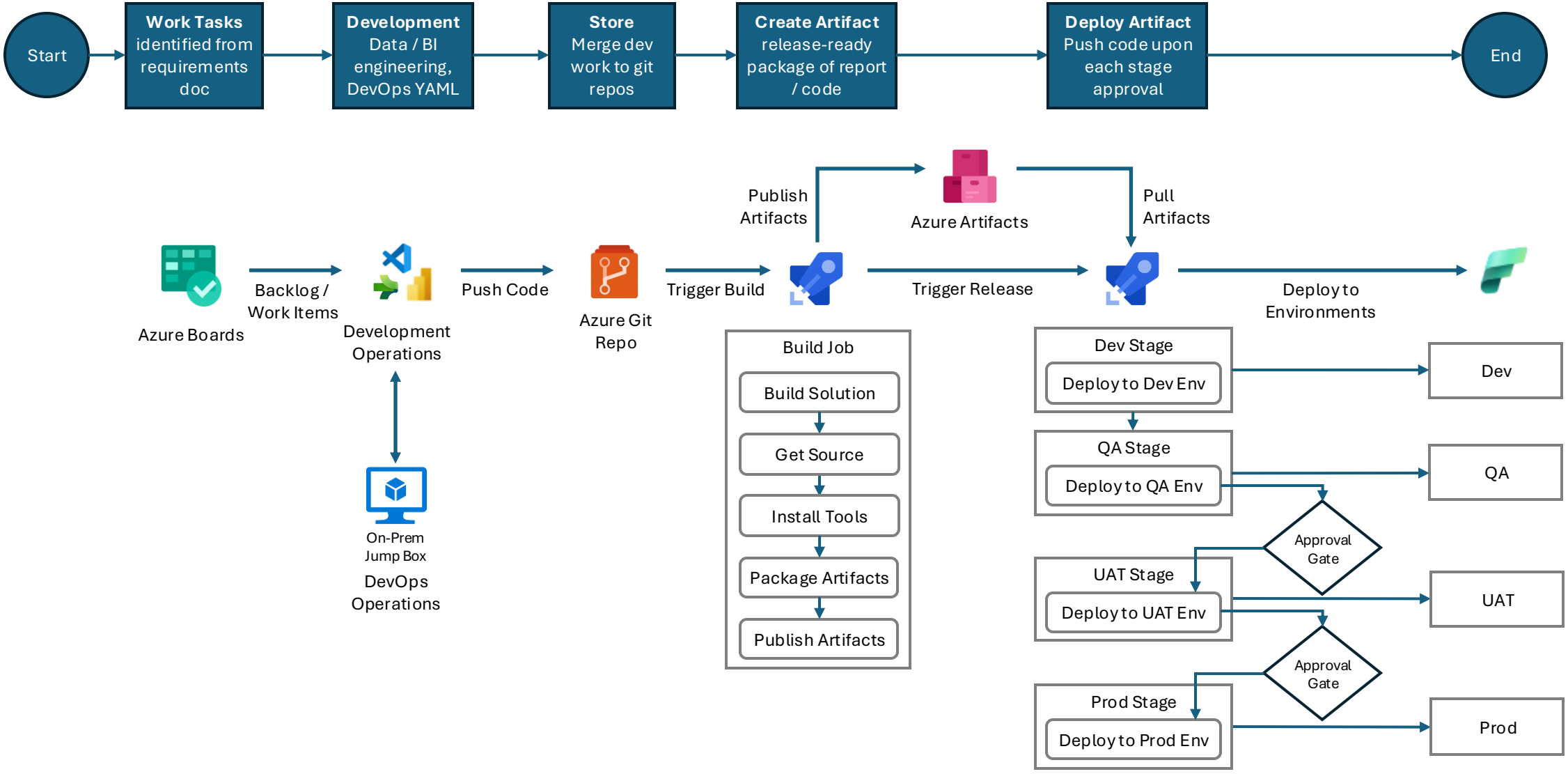
*Not Continuous Deployment, which is fully automated

Development Flow - Process and DevOps CI/CD

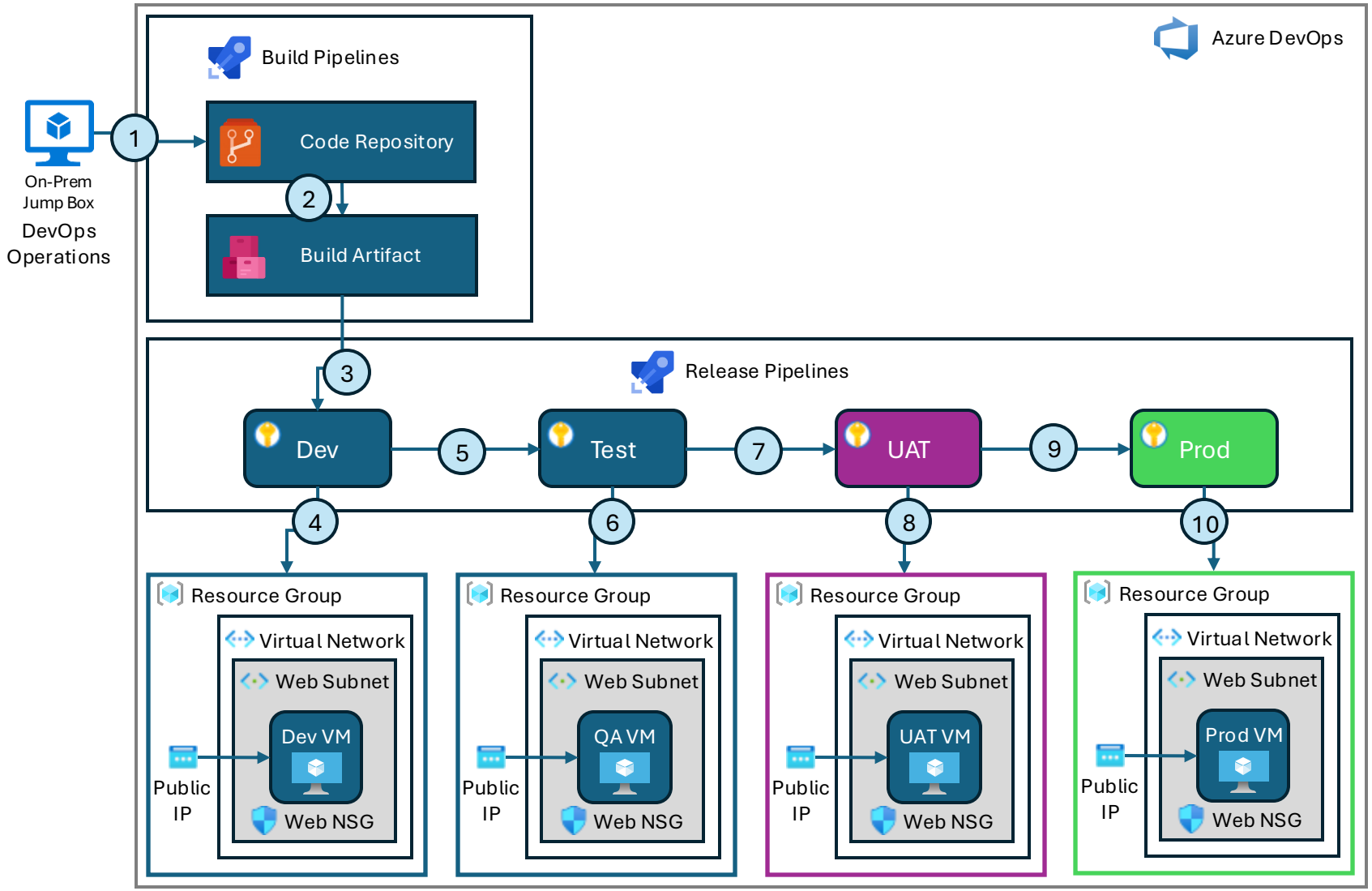
Staged Branching Strategy - delete feature (dev) branches once merged



Deployment Flow from Dev Environment - DevOps CI/CD

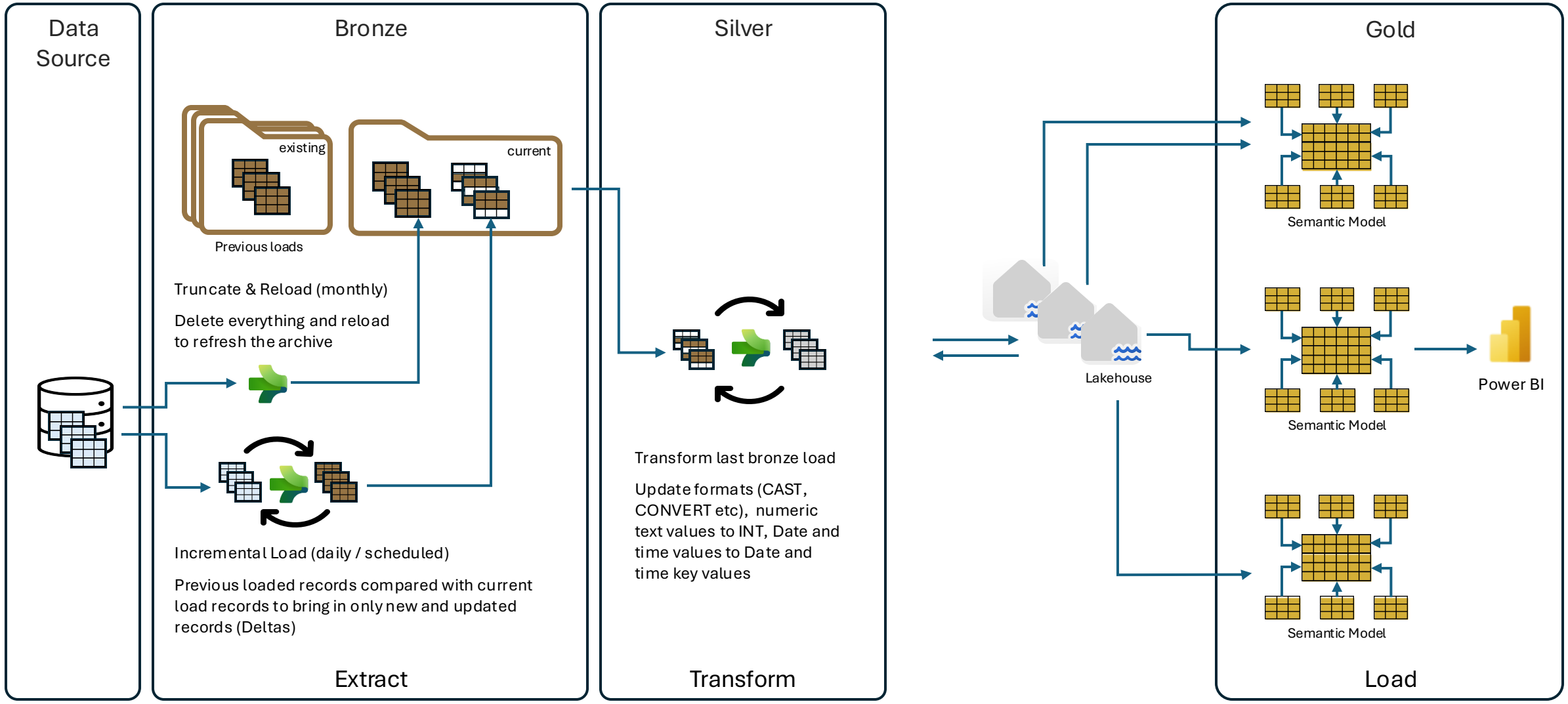


Development Flow - DevOps CI/CD Server Interaction



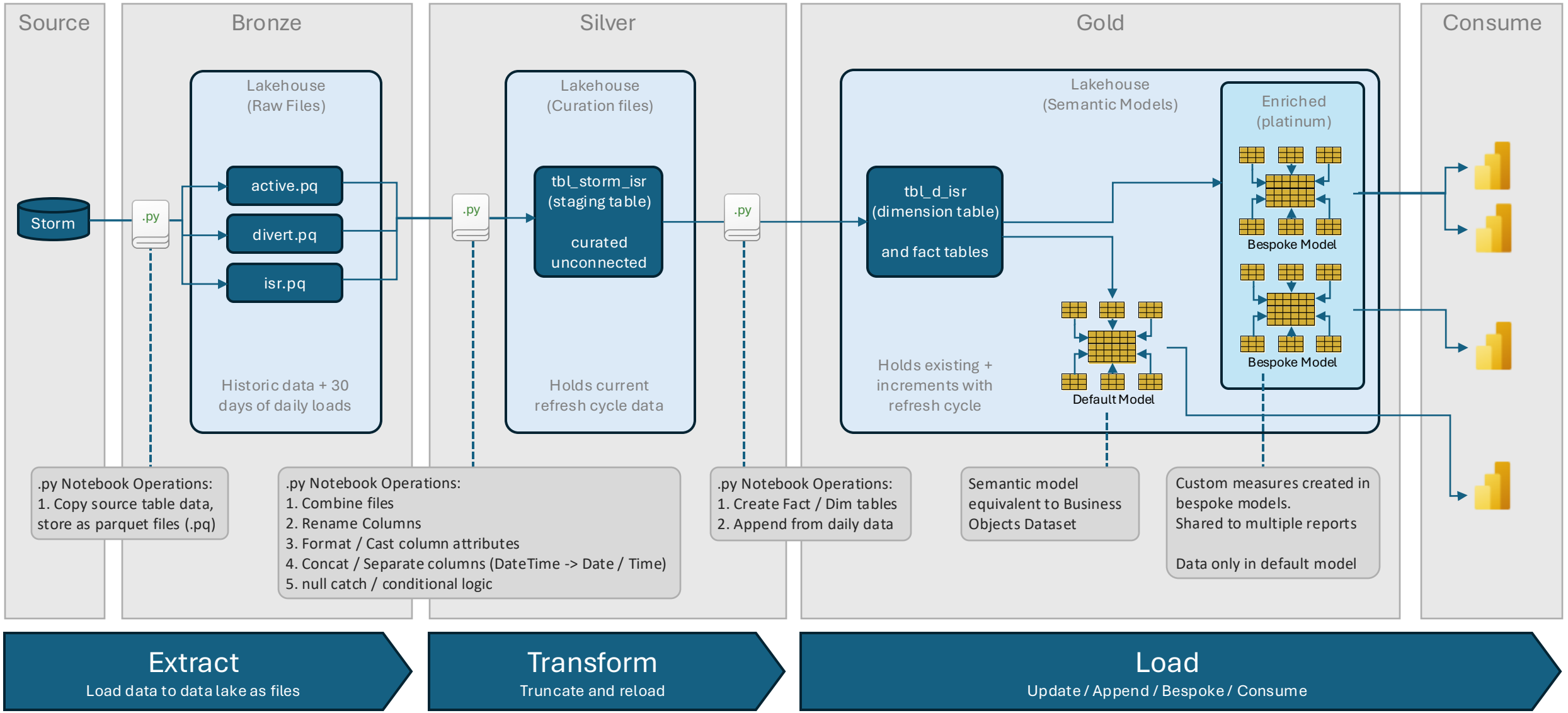
1. Code Development - Developers create feature branches, implement changes, commit, and push code, via Jump Box where VS Code is installed
2. Merge & Build - Code is peer-reviewed, merged into main/master, and the pipeline prepares build artifacts (packages, Fabric items, SQL scripts, etc.)
3. Deploy to Dev - Build artifacts are automatically deployed to the Dev environment
4. Validate in Dev: Deployed to Dev and developers perform unit tests / smoke tests in Dev to ensure functionality
5. Deploy to Test - If Dev validation passes, code is promoted and deployed to the Test environment
6. Verify in Test - The test team (or automated regression tests) validate the solution against requirements
7. Deploy to UAT - Once Test verification succeeds code is promoted and deployed to the UAT environment
8. Validate in UAT - Business users (UAT team) validate functionality against requirements and sign off
9. Deploy to Live (Prod) - After UAT acceptance, the release is promoted and deployed to the Live environment
10. Post-Deployment Verification - Smoke tests, monitoring, and validation are performed in Live to confirm stability

Medallion Data Load Pattern - ETL



Semantic Model Subset Equivalent to Business Objects Dataset)
Default semantic model on each lakehouse

Medallion Warehouse Load Pattern



Extract
Load data to data lake as files

Transform
Truncate and reload

Load
Update / Append / Bespoke / Consume

Flat File to Normalised Structure - Creating Lookups

Vehicles

VEHICLE_ID	MAKE_ID	MODEL	COLOUR	FUEL	REG_NO	REG_DATE
1	BMW	3 Series	Jet Black	Petrol	RG 75 YUM	1/9/25
2	Audi	EQ6	Metallic Black	Electric	MM 75 WOW	3rd September 25
3	Ford	Focus	White	Diesel	KV 67 OMG	01/01/2017

VEHICLE_ID	REG_NO	REG_DATE
1	RG 75 YUM	1/9/25
2	MM 75 WOW	3rd September 25
3	KV 67 OMG	01/01/2017

Vehicle_Details

MAKE_ID	MAKE_TEXT
100	BMW
200	Audi
300	Ford

Vehicle_Make

TYPE_ID	TYPE_TEXT
1	Petrol
2	Diesel
3	LPG
4	Electric
5	Petrol Hybrid

Vehicle_Fuel_Type

MODEL_ID	MODEL_TEXT	MAKE_ID
1000	1 Series	100
1010	2 Series	100
1020	3 Series	100
2000	A6	200
2010	Q6	200
2020	EQ6	200
3000	Fiesta	300
3010	Focus	300

Vehicle_Model

COL_ID	COL_TEXT
1	White
2	Space Grey
3	Metallic Black
4	Jet Black

Vehicle_Colour

Source - Flat File

Extract columns in source file

- De-Duplicate - Keep unique values
- Assign unique ID to each record

Query to extract in original joined format

```
SELECT
  v.VEHICLE_ID  AS [Vehicle ID]
 ,vm.MAKE_TEXT AS [Make]
 ,vk.MODEL_TEXT AS [Model]
 ,vc.COL_TEXT AS [Colour]
 ,vf.TYPE_TEXT AS [Fuel]
 ,REG_NO AS [Reg No]
 ,REG_DATE AS [Reg Date]
FROM
 Vehicles v
  INNER JOIN Vehicle_Make vm ON v.MAKE_ID = vm.MAKE_ID
  INNER JOIN Vehicle_Model vk ON v.MODEL_ID = v.MODEL_ID
  INNER JOIN Vehicle_Colour vc ON vc.COL_ID = v.COL_ID
  INNER JOIN Vehicle_Fuel_Type vf ON vf.TYPE_ID = v.TYPE_ID
WHERE
  Make = 'BMW';
```

Vehicle ID	Make	Model	Colour	Fuel	Reg No	Reg Date
1	BMW	3 Series	Jet Black	Petrol	RG 75 YUM	1/9/25

Source Data Medallion Flow - Bronze to Silver

VehicleID	MakeID	ModelID	ColourID	FuelTypeID	RegNo	RegDate
1	100	1020	4	1	RG 75 YUM	1/9/25
2	200	2020	3	4	MM 75 WOW	3rd September 2025
3	300	3010	1	5	KV 67 OMG	20170101

Source to Bronze

Extract source tables from source database

- Assign unique ID to each record

Save in Bronze as parquet file

VEHICLE_ID	REG_NO	REG_DATE
1	RG 75 YUM	1/9/25
2	MM 75 WOW	3rd September 25
3	KV 67 OMG	01/01/2017

Vehicle_Details

MODEL_ID	MODEL_TEXT	MAKE_ID
1000	1 Series	100
1010	2 Series	100
1020	3 Series	100
2000	A6	200
2010	Q6	200
2020	EQ6	200
3000	Fiesta	300
3010	Focus	300

Vehicle_Model

MAKE_ID	MAKE_TEXT
100	BMW
200	Audi
300	Ford

Vehicle_Make

COL_ID	COL_TEXT
1	White
2	Space Grey
3	Metallic Black
4	Jet Black

Vehicle_Colour

TYPE_ID	TYPE_TEXT
1	Petrol
2	Diesel
3	LPG
4	Electric
5	Petrol Hybrid

Vehicle_Fuel_Type

VEHICLE_ID	REG_NO	REG_DATE
1	RG 75 YUM	20250901
2	MM 75 WOW	20250903
3	KV 67 OMG	20170101

Vehicles

Bronze to Silver

Checking formats to standardise

- De-Duplicate - Keep unique values
- Replace dates as date key value
- Standardise formats - INT64 to BIGINT

Source Data Medallion Flow - Silver to Gold

dimVehicleMake

MakeID	VehicleMake
100	BMW
200	Audi
300	Ford

dimVehicleModel

ModelID	VehicleModel	MakeID
1000	1 Series	100
1010	3 Series	100
2000	A6	200
2010	EQ6	200
3000	Fiesta	300
3010	Focus	300

dimColour

ColourID	Colour
1	White
2	Space Grey
3	Black

dimFuelType

FuelTypeID	FuelType
1	Petrol
2	Diesel
3	Electric

factVehicles

VehicleID	MakeID	ModelID	ColourID	FuelTypeID	RegNo	RegDate	CurrentRec	DaysSinceReg
1	100	1020	2	1	RG 75 YUM	20250901	Y	40
2	200	2000	3	2	MM 75 WOW	20250903	Y	37
3	300	3010	1	4	KV 67 OMG	20170101	N	3204
3	300	3010	1	4	B 16 FOC	20250902	Y	38
4	200	2010	3	3	RG 75 YES	20251001	Y	1

dimDate

DateID	ShortDate	LongDate	Day	Month	Year	Quarter	FiscalMonth
20170101	01/01/2017	January 1st, 2017	1	Jan	2017	1	10
...	...	...	...	...	...	...	...
20250901	01/09/2025	September 1st, 2025	1	Sep	2025	3	6
20250902	02/09/2025	September 2nd, 2025	2	Sep	2025	3	6
20250903	03/09/2025	September 3rd, 2025	3	Sep	2025	3	6

Silver to Gold

Checking silver records against gold's for changes
- De-Duplicate / Changes - Add new record as current, update flag of existing record
- Update dimensions first then facts
- Associate new fact records to dimension keys

Creates a Star Schema Semantic model comprising the cleansed data, structured for BI reporting

Platinum Enrichment (measures)

Adding calculated fields to the semantic model
- KPIs
- Aggregations
- Combining fields to create composite values

Can be attached to the model to share across reports or created in the report

Date Dimension

A pre-created calendar table with a unique Date Key to align with standardised date values in silver process

Multiple permutations of a date are available to select on the report in human and abbreviated format

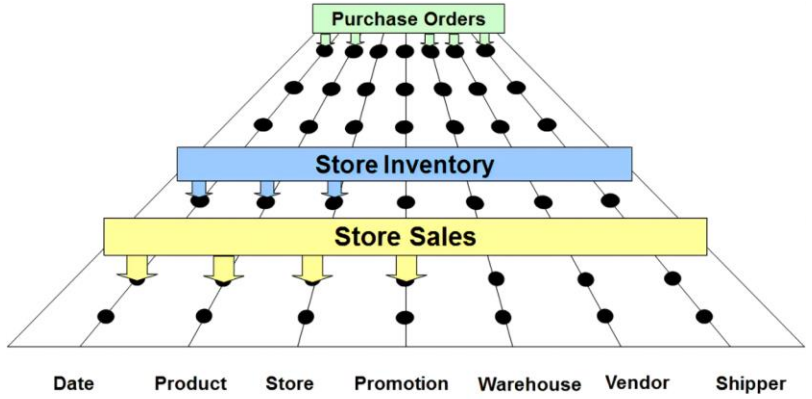
Reporting periods automatically aggregate for on-report calculations

Data Warehouse Design

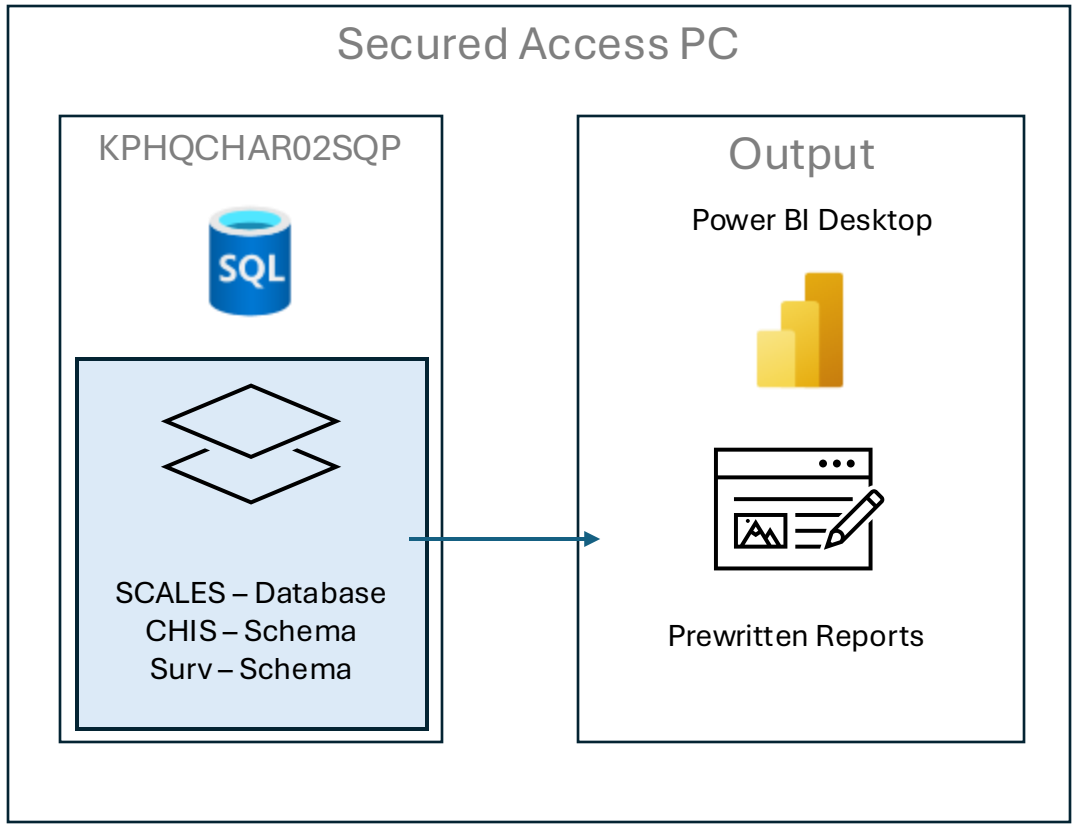
Bus Matrix

	Fact Type	Date	Position	Employee	Organization	Benefit
Hiring Processes						
Employee Position Snapshot	Periodic	X	X	Empl Mgr	X	
Employee Requisition Pipeline	Accumulating	X	X	Empl Mgr	X	
Employee Hiring	Transaction	X	X	Empl Mgr	X	
Employee "On Board" Pipeline	Accumulating	X	X	Empl Mgr	X	
Benefits Processes						
Employee Benefits Eligibility	Periodic	X		X	X	X
Employee Benefits Application	Accumulating	X		X	X	X
Employee Benefit Participation	Periodic	X		X	X	X
Employee Management Processes						
Employee Headcount Snapshot	Periodic	X		X	X	X
Employee Compensation	Transaction	X		X	X	X
Employee Benefit Accruals	Transaction	X		X	X	X
Employee Performance Review Pipeline	Accumulating	X		Empl Mgr	X	X
Employee Performance Review	Transaction	X		Empl Mgr	X	X
Employee Prof Dev Completed Courses	Transaction	X		X	X	
Employee Disciplinary Action Pipeline	Accumulating	X		Empl Mgr	X	
Employee Separations	Transaction	X		Empl Mgr	X	

Business Process / Event	Time	Customer	Service	Rate Category	Local Svc Provider	Calling Party	Called Party	Long Dist Provider	Internal Organization	Employee	Location	Equipment Type	Supplier	Item Shipped	Account Status
Customer Billing	X	X	X	X	X			X			X				X
Service Orders	X	X	X		X			X	X	X	X	X			X
Trouble Reports	X	X	X		X	X		X	X	X	X	X	X	X	X
Yellow Page Ads	X	X		X		X			X	X	X				X
Customer Inquiries	X	X	X	X	X	X		X	X	X	X				X
Promotions & Communication	X	X	X	X	X	X		X	X	X	X	X	X	X	X
Billing Call Detail	X	X	X	X	X	X	X	X	X		X	X	X	X	X
Network Call Detail	X	X	X	X	X	X	X	X	X		X	X	X	X	X
Customer Inventory	X	X	X	X	X			X	X		X	X	X	X	X
Network Inventory	X		X						X	X	X	X	X	X	
Real Estate	X								X	X	X	X			
Labor & Payroll	X								X	X	X				
Computer Charges	X	X	X		X			X	X	X	X	X	X	X	
Purchase Orders	X								X	X	X	X	X	X	
Supplier Deliveries	X								X	X	X	X	X	X	



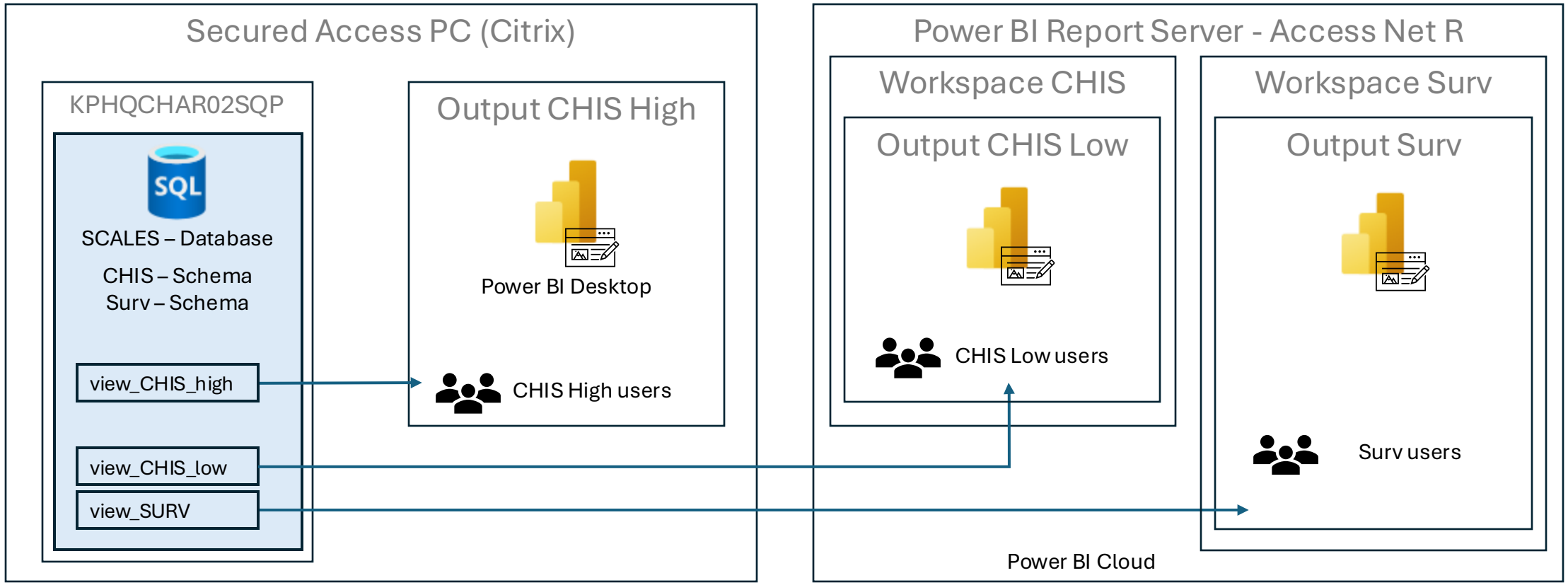
SCALES – Power BI Architecture



- Legacy data structured (ERD provided), into a new empty database on the existing KPHQCHAR02SQP server, utilising existing security.
- Legacy data split into 2 schemas. Separated data by report for
  - CHIS
  - Surveillance
- Connection via secure pipe from Power BI
- Secured-access PC running Power BI Desktop in CAB
  - Power BI Pro licence not required (not published to Power BI Service)
- Trace Events to run on the database to capture activity SQL Trace

Assumed :
Secured VM housing source DBs with schema segregated reporting tables
Single-access PC as current

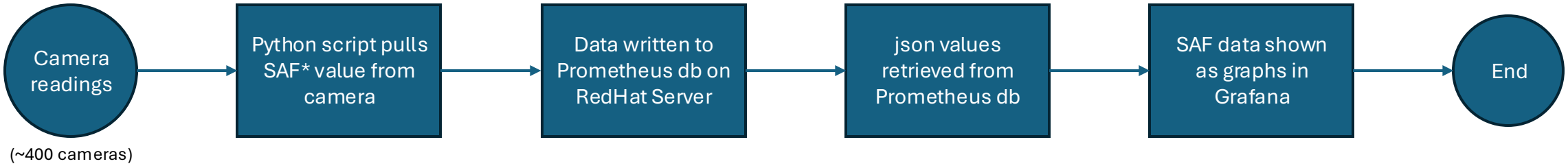
SCALES – Power BI Architecture

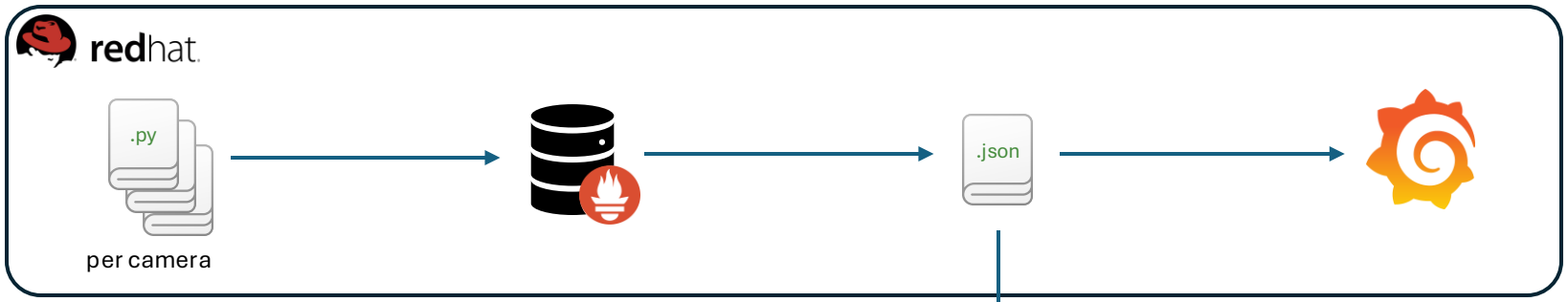


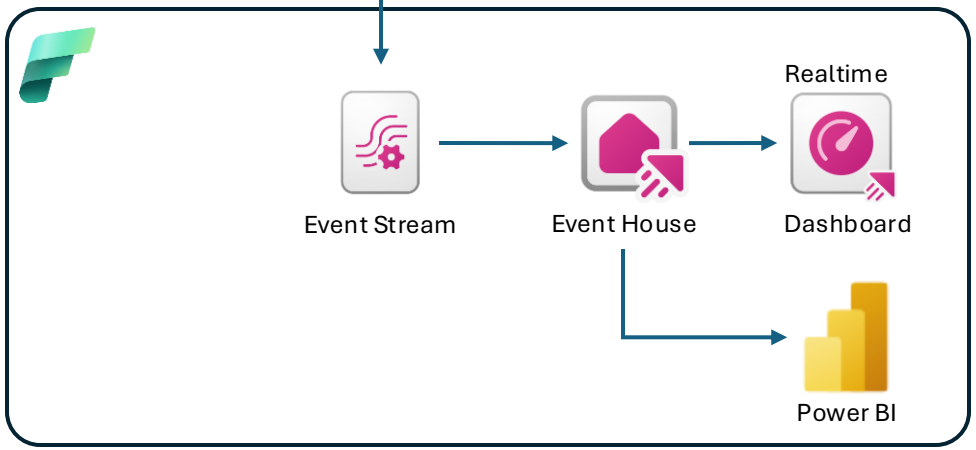
Physical Access Control:
1 - Citrix login
2 - Data segregated by schema
3 - Schema-controlled View
4 - Power BI Desktop Report

Physical Access Control:
1 - Power BI login (Net-R)
2 - Data segregated by schema
3 - Schema-controlled View
4 - Power BI Cloud Report

ANPR Camera Stats / Performance Monitoring



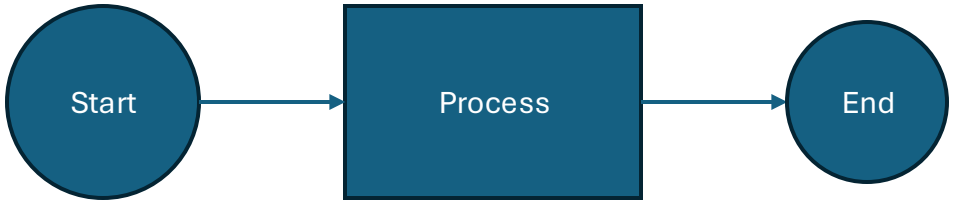


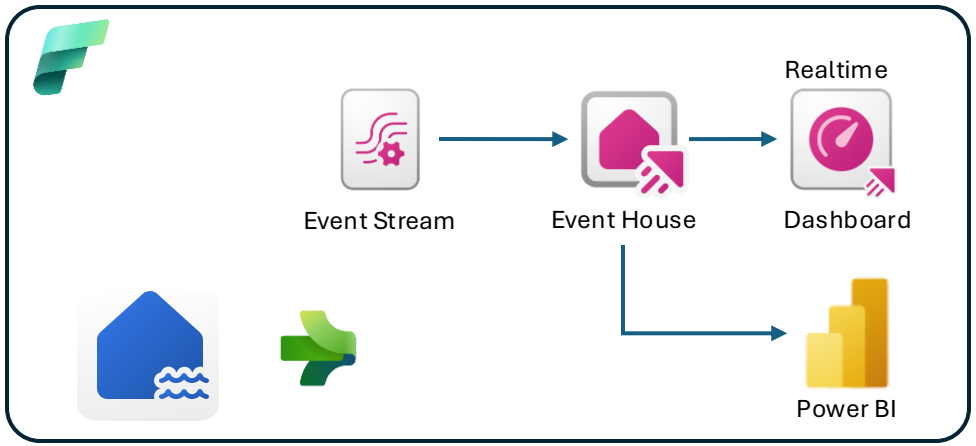


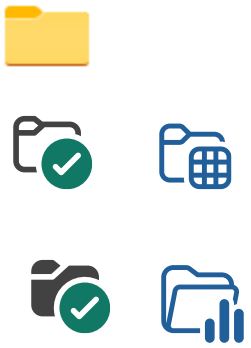
Streaming data from Prometheus database read and displayed real-time in Fabric. Can be stored for BI in medallion lakehouse and presented in Power BI

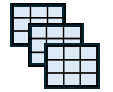
* SAF - Store And Forward

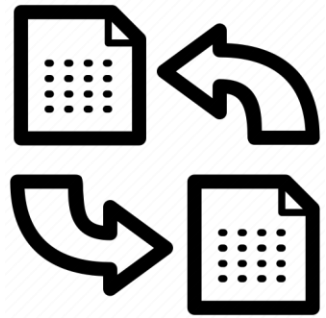
Image Library

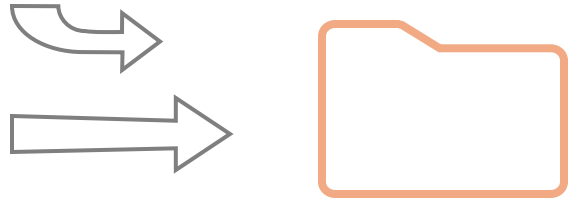






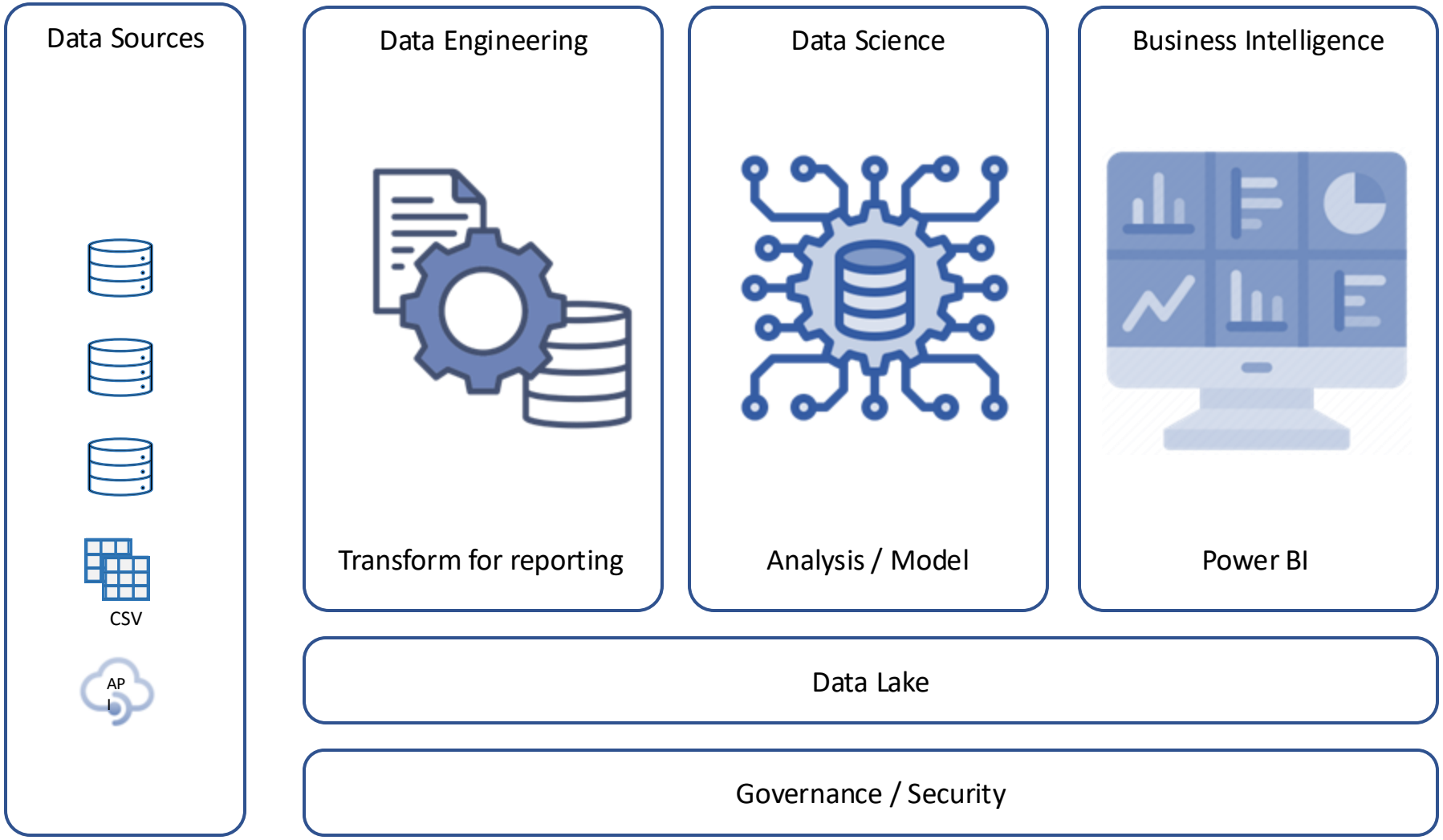






Truncate
& Reload

Microsoft Fabric - Components



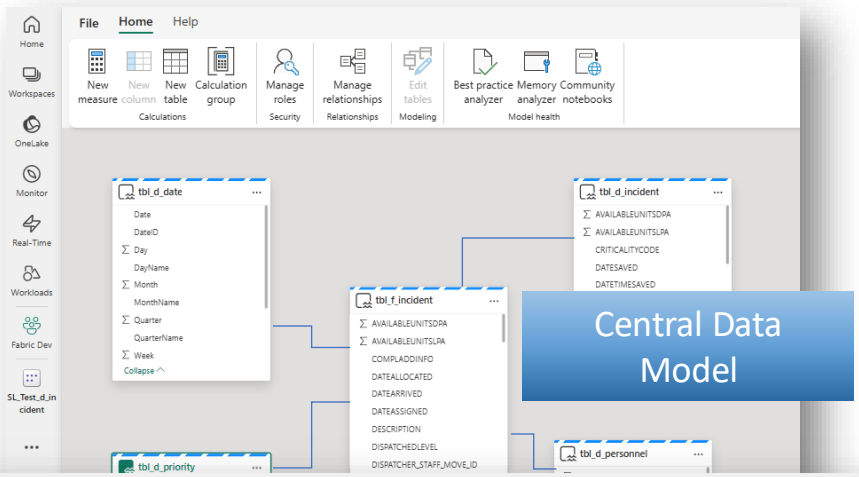
Unified SaaS solution
- Azure features integrated (database / storage / process)
- Focus on the data

Role-specific workloads
- Access controlled

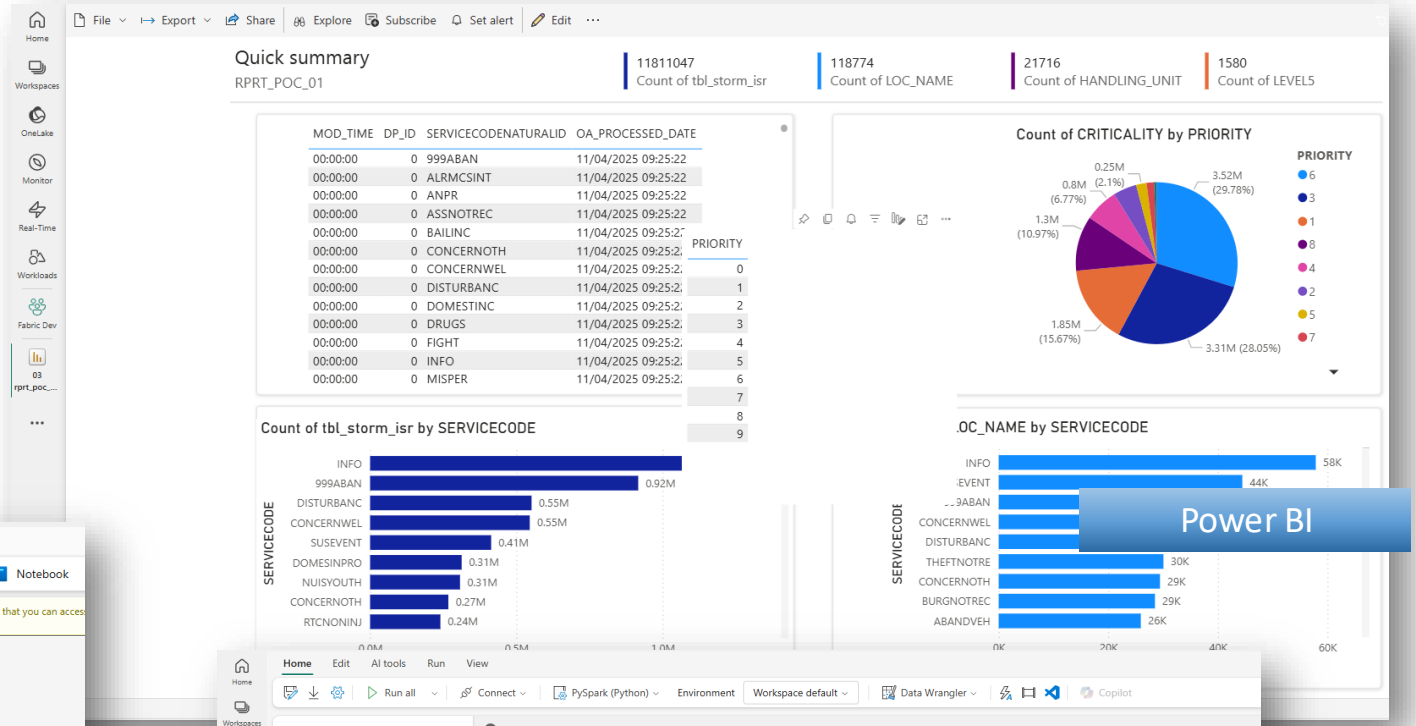
OneLake (data lake)
- Copy / mirrored data
- Unified data management

Integration with M365 / Copilot

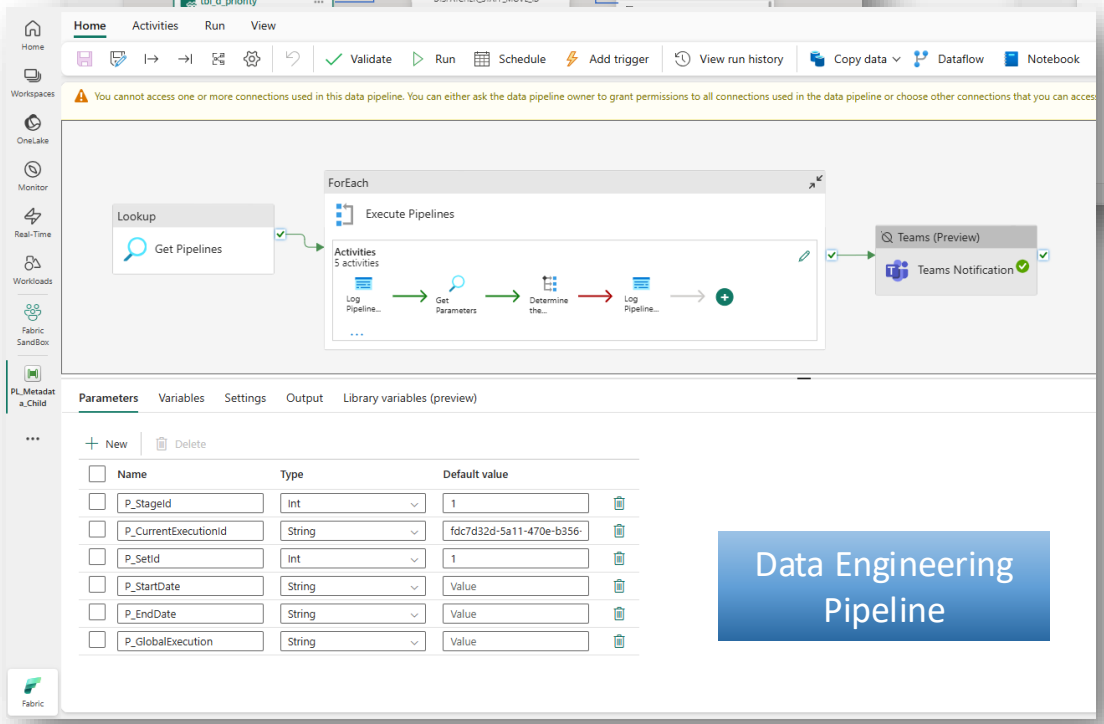
Microsoft Fabric - Screenshots



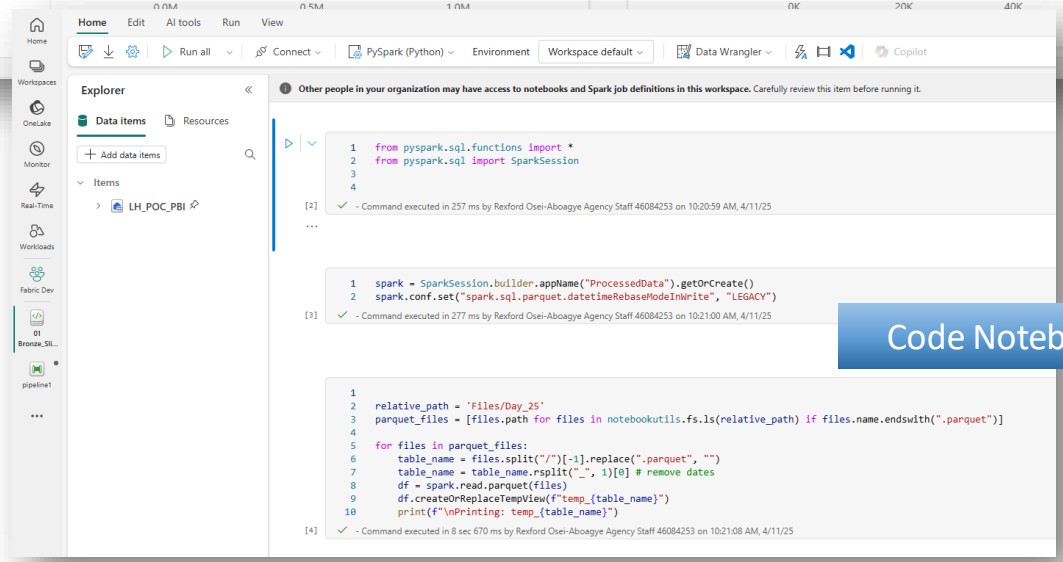
Central Data Model



Power BI

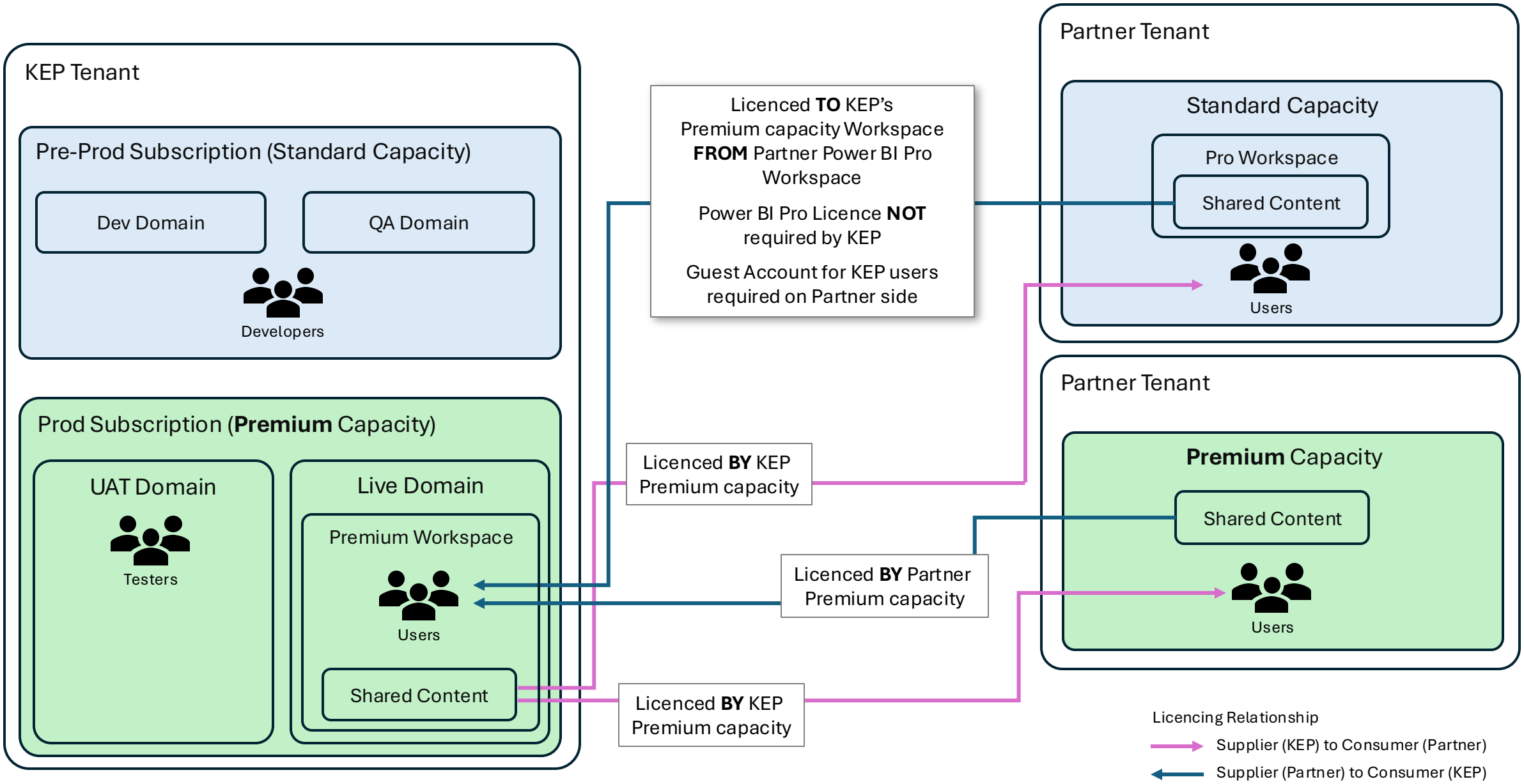


Data Engineering Pipeline

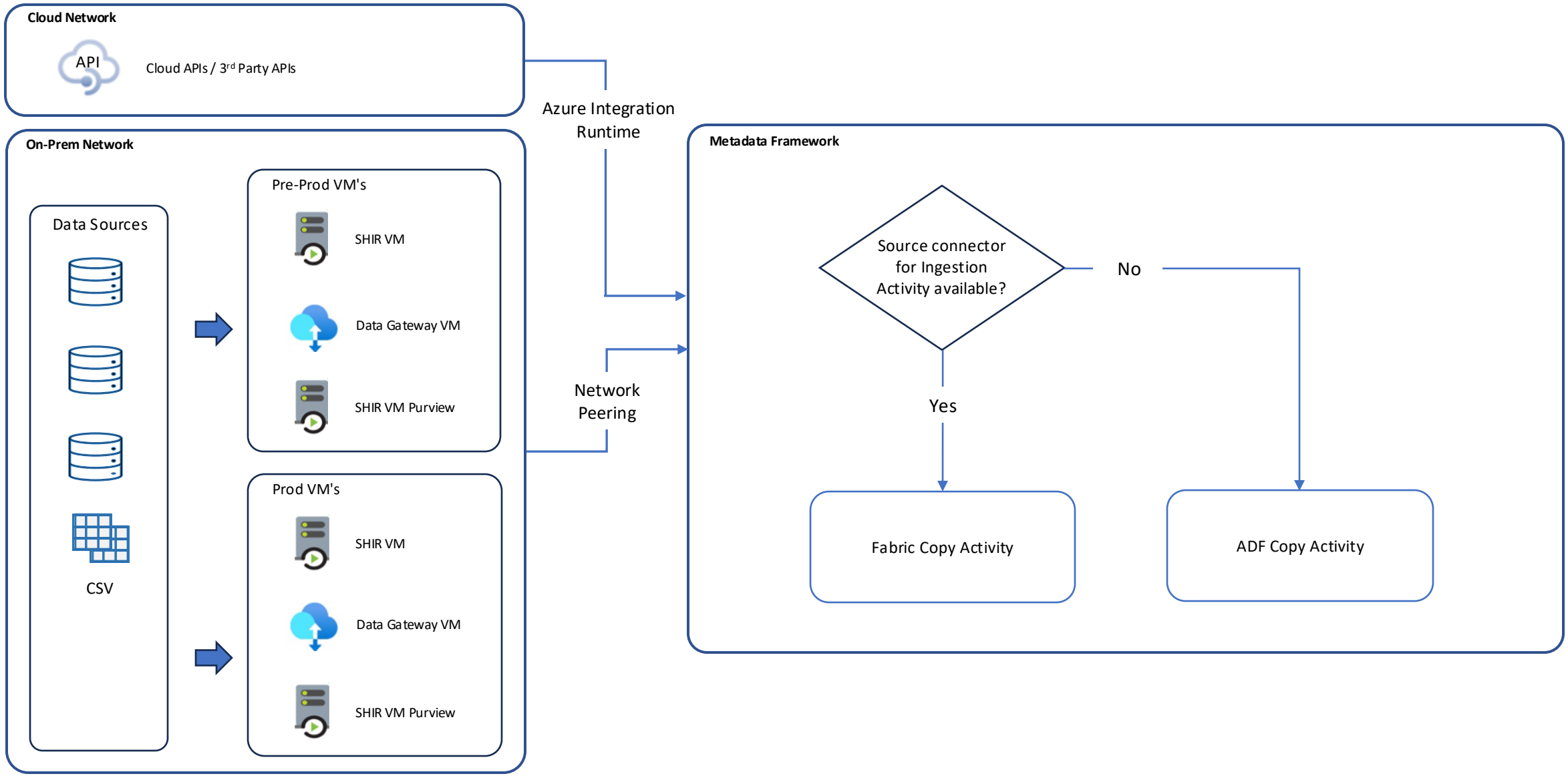


Code Notebook

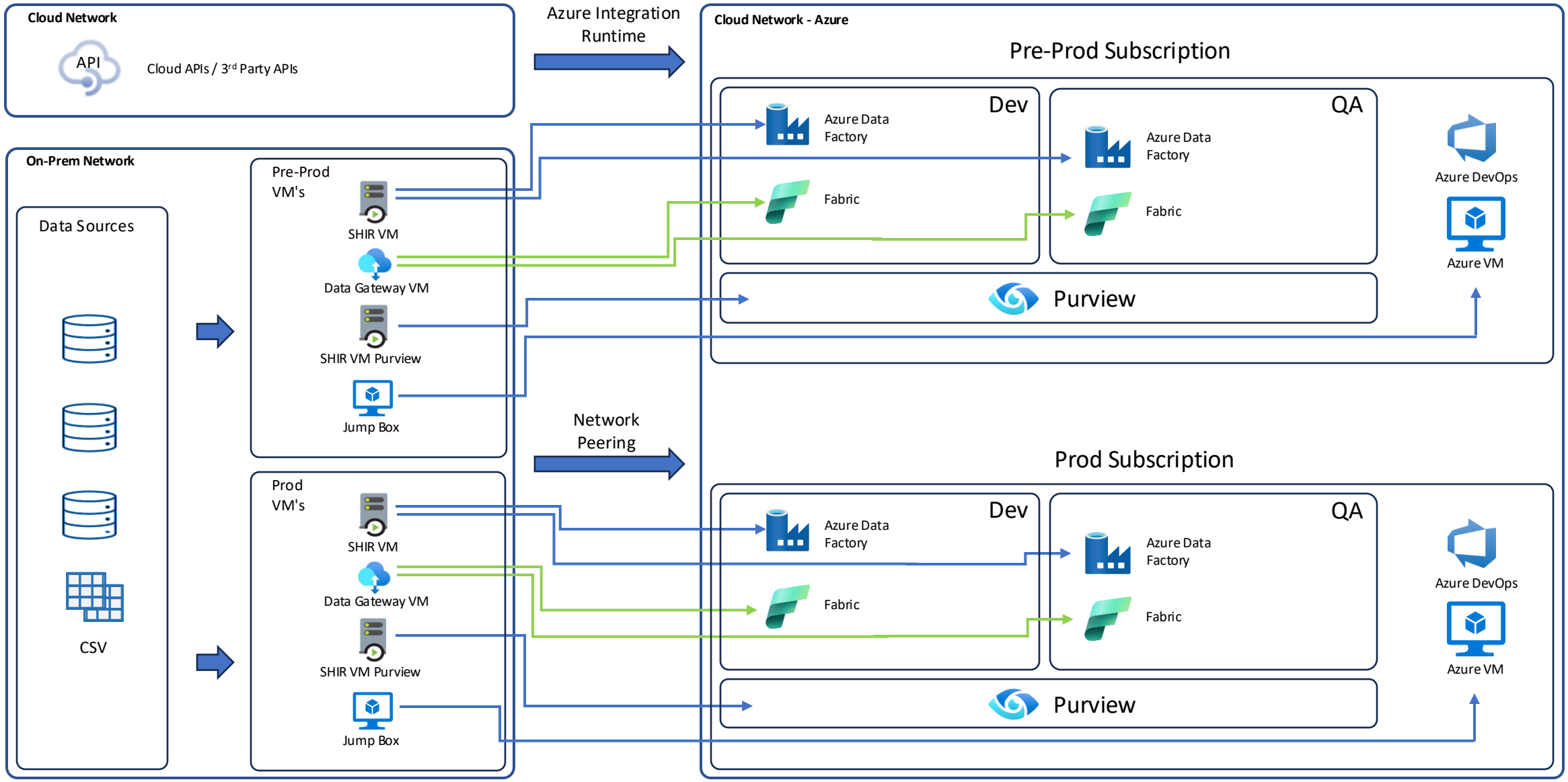
Microsoft Fabric - Licensing / Consumption



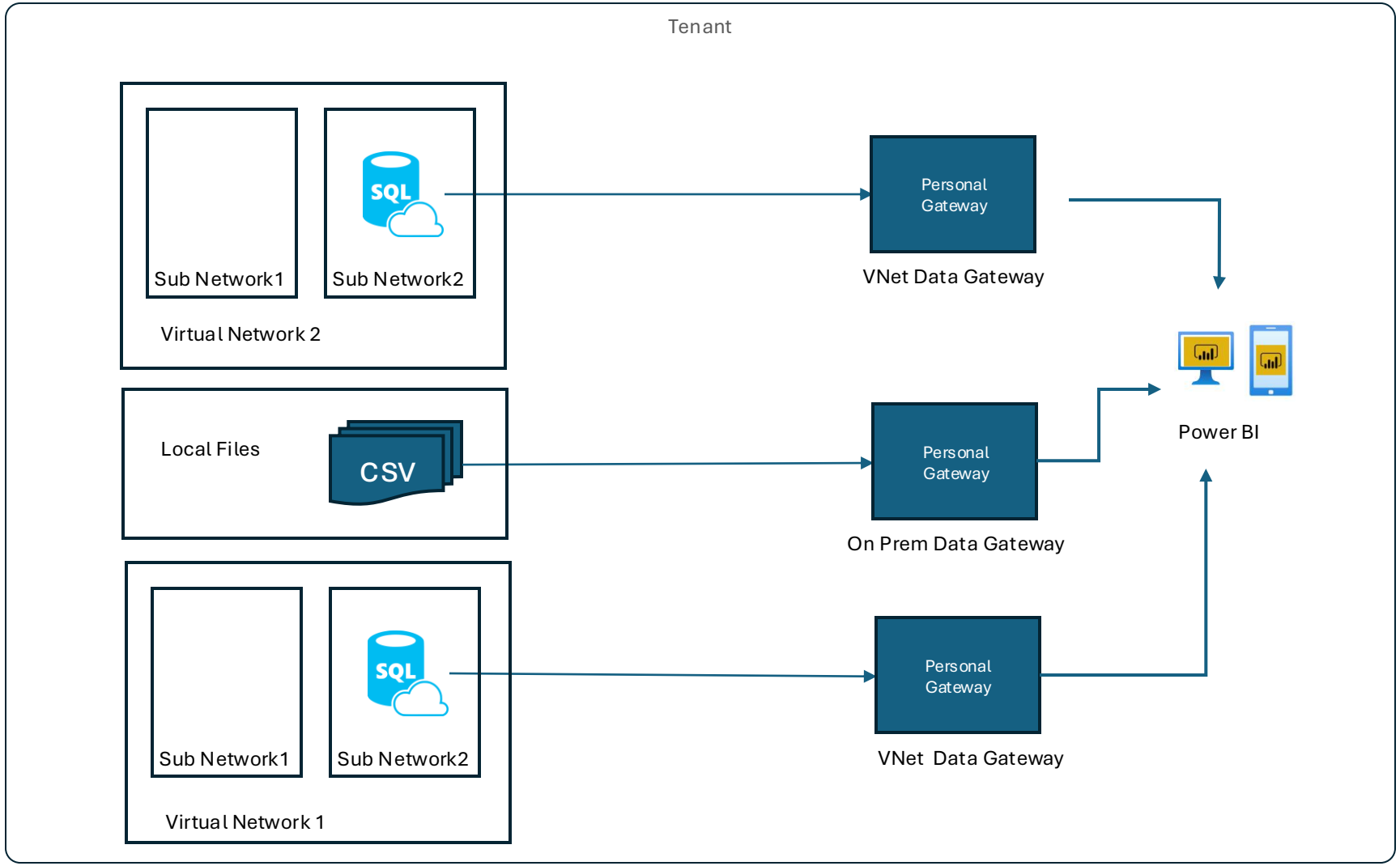
Data Pipelines Flow - Data Ingestion

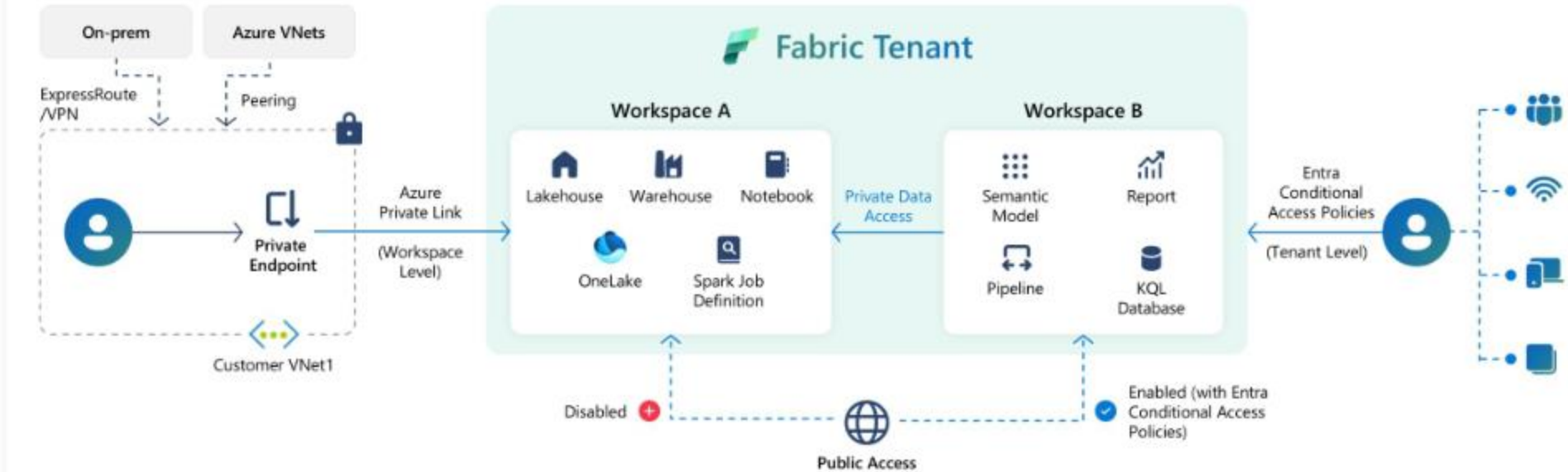


Fabric Data Connections - Data Gateways, Jump Boxes and SHIRs



Data Flows - Power BI / Virtual Network





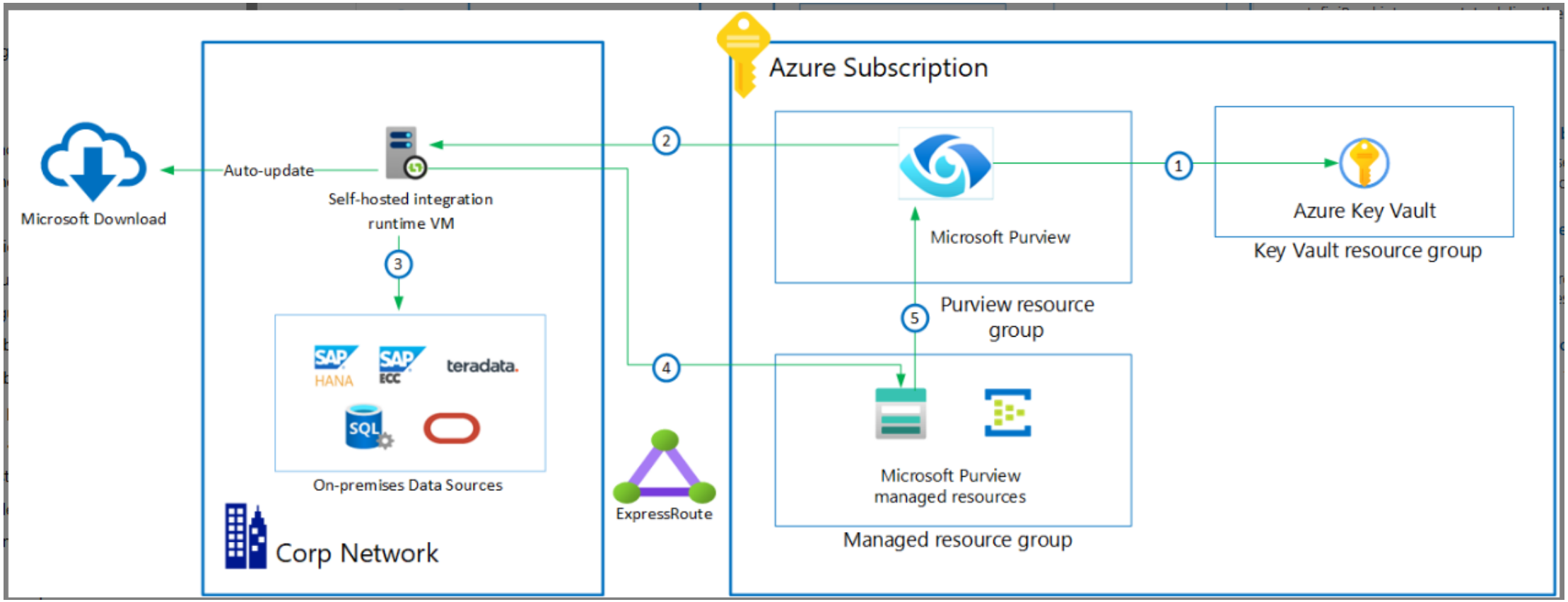
Selected workspaces can be protected using Private Links and closed from public internet.

Create a secure connection between public and private workspaces using private data access.

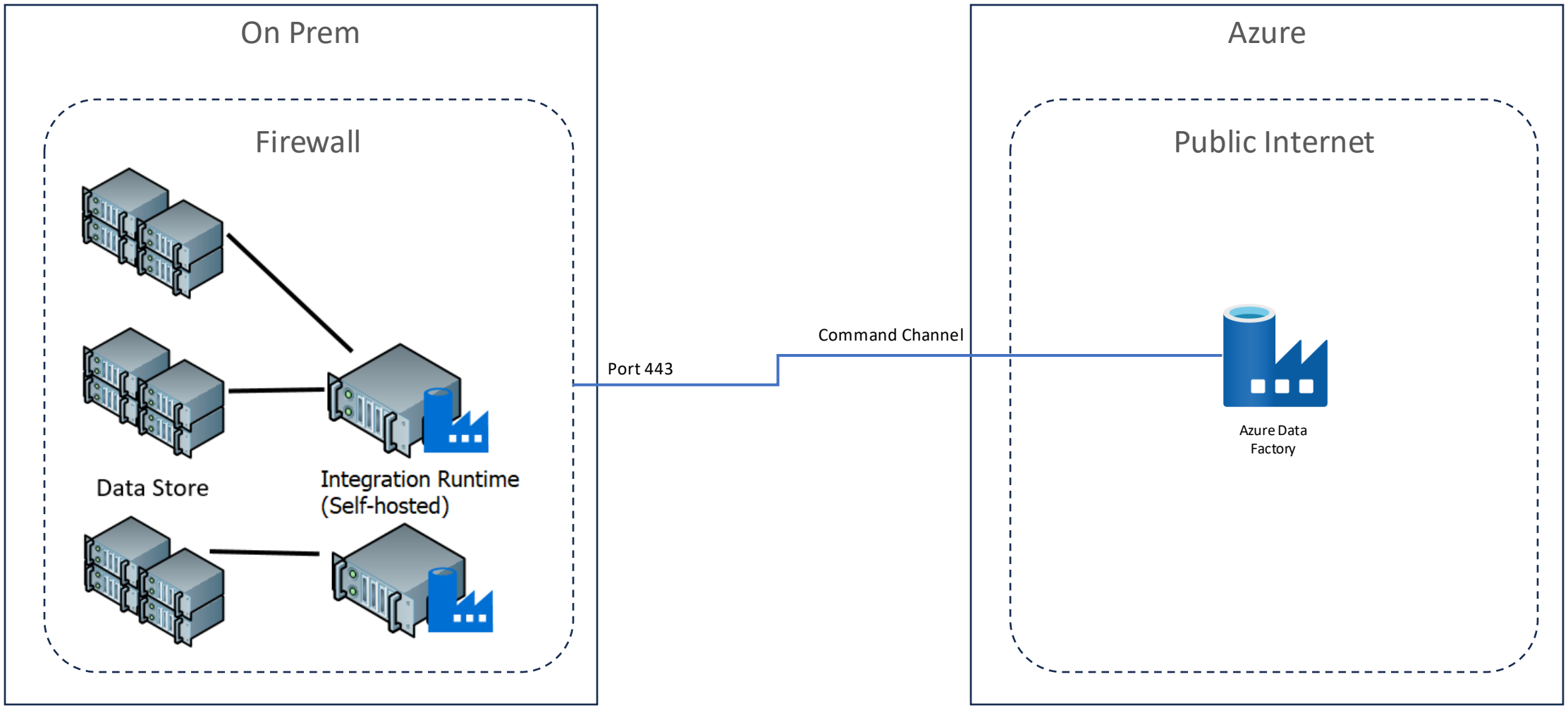
Public workspaces are secured through Entra policies for example to use Power BI.

Purview reference architecture Sample

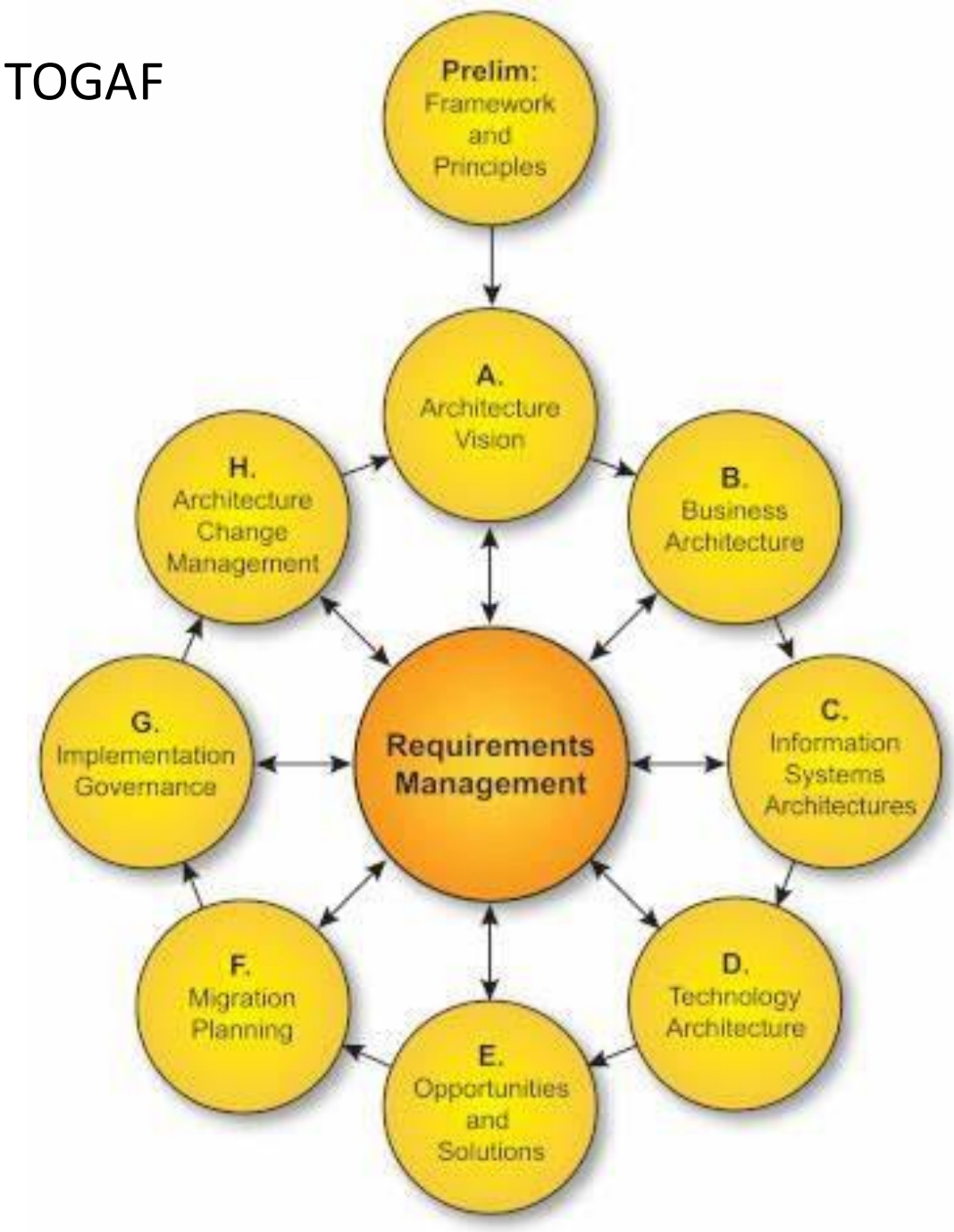
we need to implement same with minor changes pointing the sources to Fabric One Lake



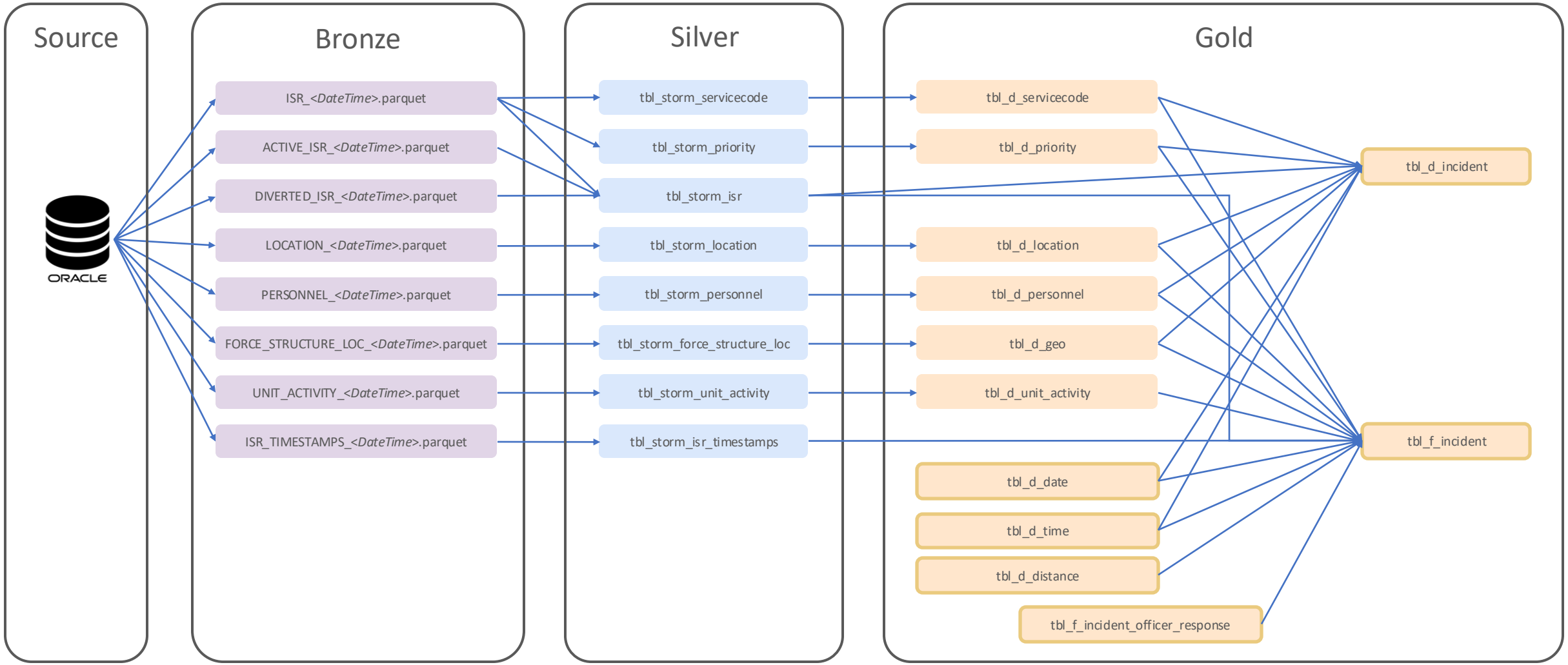
ADF to SHIR Communication



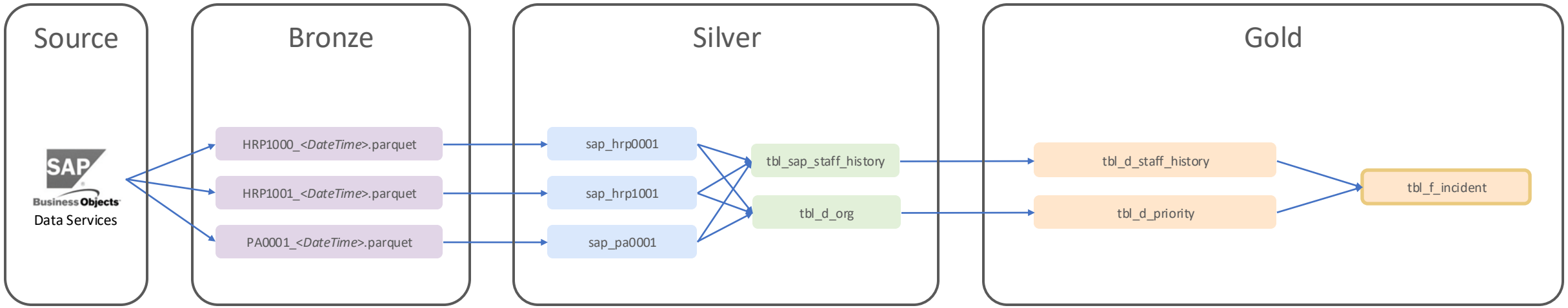
Architecture Framework - TOGAF

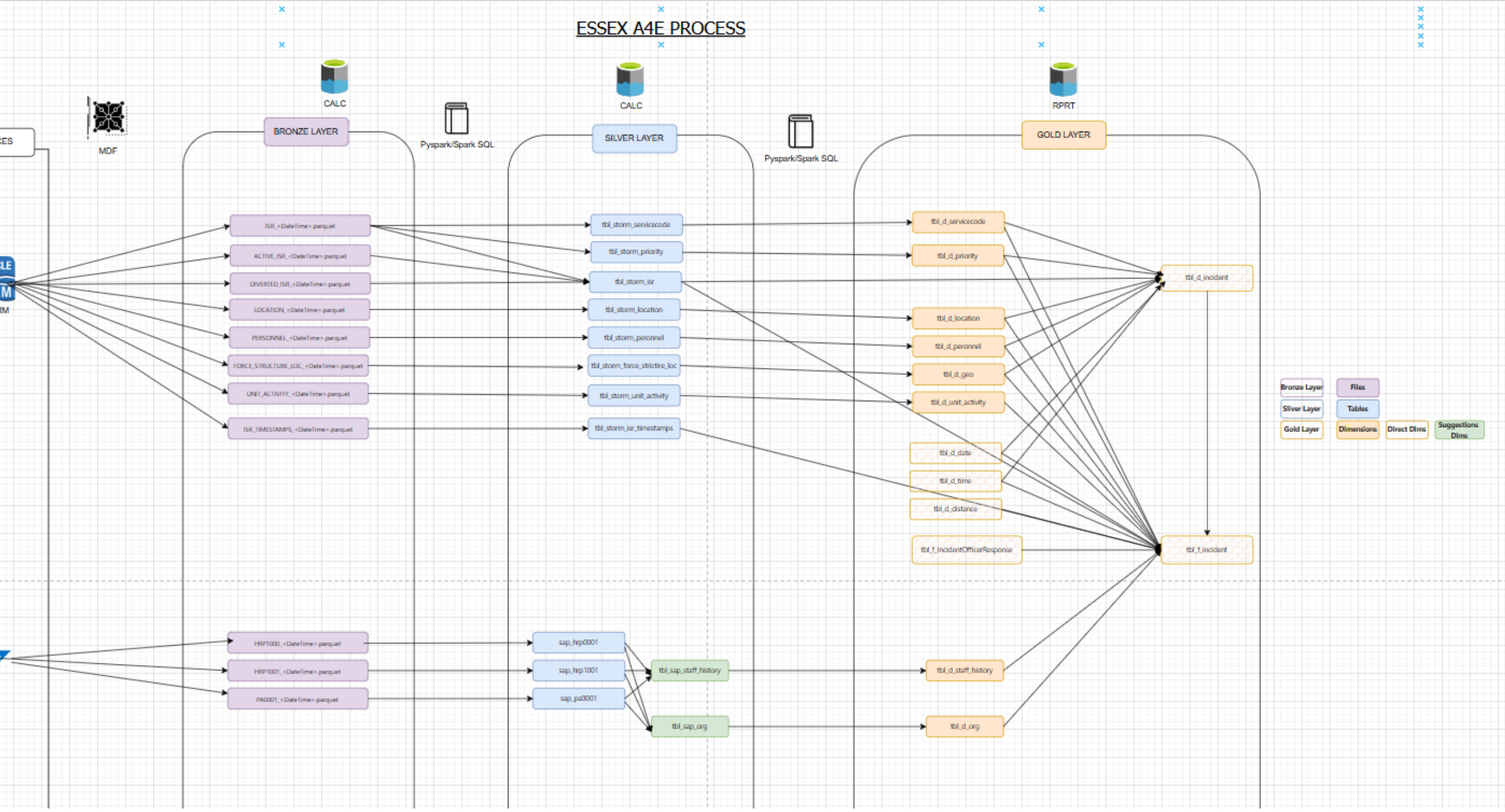


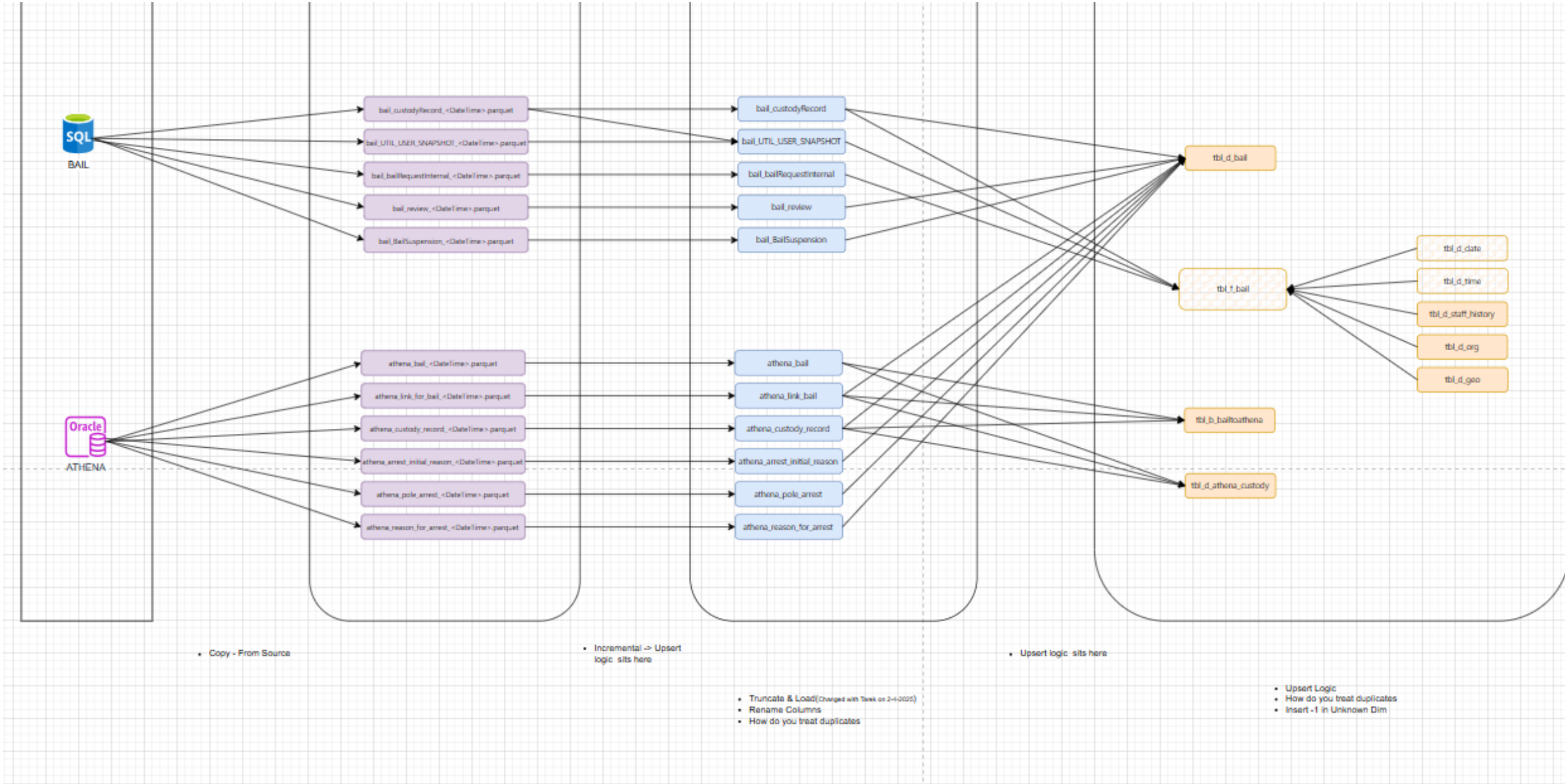
A4E Source to Medallion Target - Storm

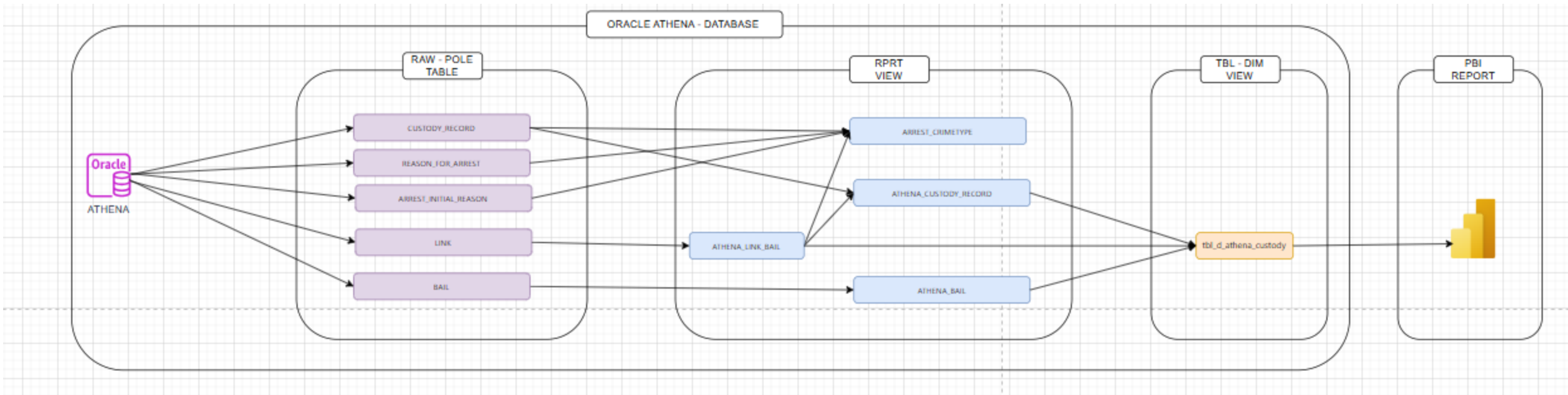


A4E Source to Medallion Target - SAP




ESSEX A4E PROCESS
CALC
CALC
RPRT
MDF
BRONZE LAYER
Pyspark/Spark SQL
SILVER LAYER
Pyspark/Spark SQL
GOLD LAYER
ISR_<DateTime>.parquet
ACTIVE_ISR_<DateTime>.parquet
DIVERTED_ISR_<DateTime>.parquet
LOCATION_<DateTime>.parquet
PERSONNEL_<DateTime>.parquet
FORCE_STRUCTURE_LOC_<DateTime>.parquet
UNIT_ACTIVITY_<DateTime>.parquet
ISR_TIMESTAMPS_<DateTime>.parquet
tbl_storm_servicecode
tbl_storm_priority
tbl_storm_isr
tbl_storm_location
tbl_storm_peronnel
tbl_storm_force_strictire_loc
tbl_storm_unit_activity
tbl_storm_isr_timestamps
tbl_d_servicecode
tbl_d_priority
tbl_d_incident
tbl_d_location
tbl_d_peronnel
tbl_d_geo
tbl_d_unit_activity
tbl_d_date
tbl_d_time
tbl_d_distance
tbl_f_IncidentOfficerResponse
tbl_f_incident
HRP1000_<DateTime>.parquet
HRP1001_<DateTime>.parquet
PA0001_<DateTime>.parquet
sap_hrp0001
sap_hrp1001
sap_pa0001
tbl_sap_staff_history
tbl_sap_org
tbl_d_staff_history
tbl_d_org
Bronze Layer
Files
Silver Layer
Tables
Gold Layer
Dimensions
Direct Dims
Suggestions Dims


SQL
BAIL
bail_custodyRecord_<DateTime>.parquet
bail_UTIL_USER_SNAPSHOT_<DateTime>.parquet
bail_bailRequestInternal_<DateTime>.parquet
bail_review_<DateTime>.parquet
bail_BailSuspension_<DateTime>.parquet
bail_custodyRecord
bail_UTIL_USER_SNAPSHOT
bail_bailRequestInternal
bail_review
bail_BailSuspension
tbl_d_bail
tbl_f_bail
tbl_d_date
tbl_d_time
tbl_d_staff_history
tbl_d_org
tbl_d_geo
Oracle
ATHENA
athena_bail_<DateTime>.parquet
athena_link_for_bail_<DateTime>.parquet
athena_custody_record_<DateTime>.parquet
athena_arrest_initial_reason_<DateTime>.parquet
athena_pole_arrest_<DateTime>.parquet
athena_reason_for_arrest_<DateTime>.parquet
athena_bail
athena_link_bail
athena_custody_record
athena_arrest_initial_reason
athena_pole_arrest
athena_reason_for_arrest
tbl_b_bailtoathena
tbl_d_athena_custody
Copy - From Source
Incremental -> Upsert logic sits here
Truncate & Load(Changed with Tarek on 2-4-2025)
Rename Columns
How do you treat duplicates
Upsert logic sits here
Upsert Logic
How do you treat duplicates
Insert -1 in Unknown Dim


ORACLE ATHENA - DATABASE
RAW - POLE TABLE
RPRT VIEW
TBL - DIM VIEW
PBI REPORT
Oracle
ATHENA
CUSTODY_RECORD
REASON_FOR_ARREST
ARREST_INITIAL_REASON
LINK
BAIL
ARREST_CRIMETYPE
ATHENA_CUSTODY_RECORD
ATHENA_LINK_BAIL
ATHENA_BAIL
tbl_d_athena_custody