Event Sourcing and Data Platform B

Cloud Community Of Practice

Purpose & Context

Event Sourcing with Data Platform B

Purpose

- An insight into Event Sourcing and a holistic view how we may apply or implement this to Data Platform B
- GDPR compliance challenge and possible another option with data privacy with Event Sourcing (or any appending data source)

Context

- Current state of Data Platform B - ETL
- Further patterns we are considering with Data Platform B - ELT and Event Sourcing
- What are events and how they capture behaviour. Which is lost when storing just current state.
- Holistic overview of the technology, over laid the proposed reference architecture for Event Sourcing
- How to may tackle GDPR and possible another option in regards to data privacy security.
- The proposed MVP and capabilities/learning we should obtain from it.

Data Platform B

Data Platform B is a data and analytics solution for Wealth. Its mission is to be able to tell **anyone, anything, anytime**.

Data Platform B Data and Analytics team

Data Platform B is a team of best in class Data Engineers and Data Scientists which support the Wealth Business by connecting various sources into Data Platform B, developing dashboards and reporting capabilities and by applying data science to business challenges.

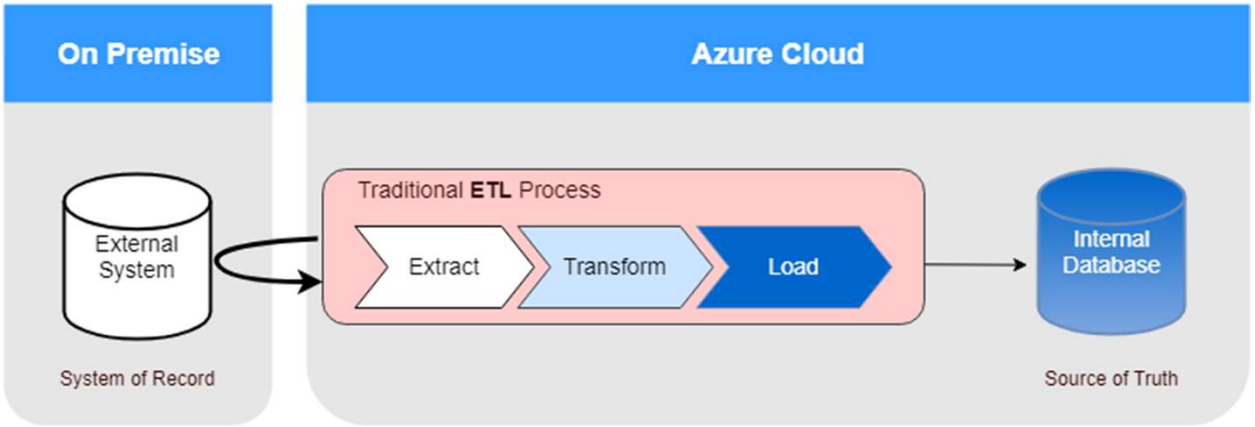
How does Data Platform B support the business:

There are 3 broad services that they provide to the business…

- **Self Serve Analytics:** Dashboards
- **Ad hoc queries**
- **Development**: build repeatable and automated solutions for the business.

ETL (Replicating on Premise to the Cloud)

Transforming of data is done on premises via ETL



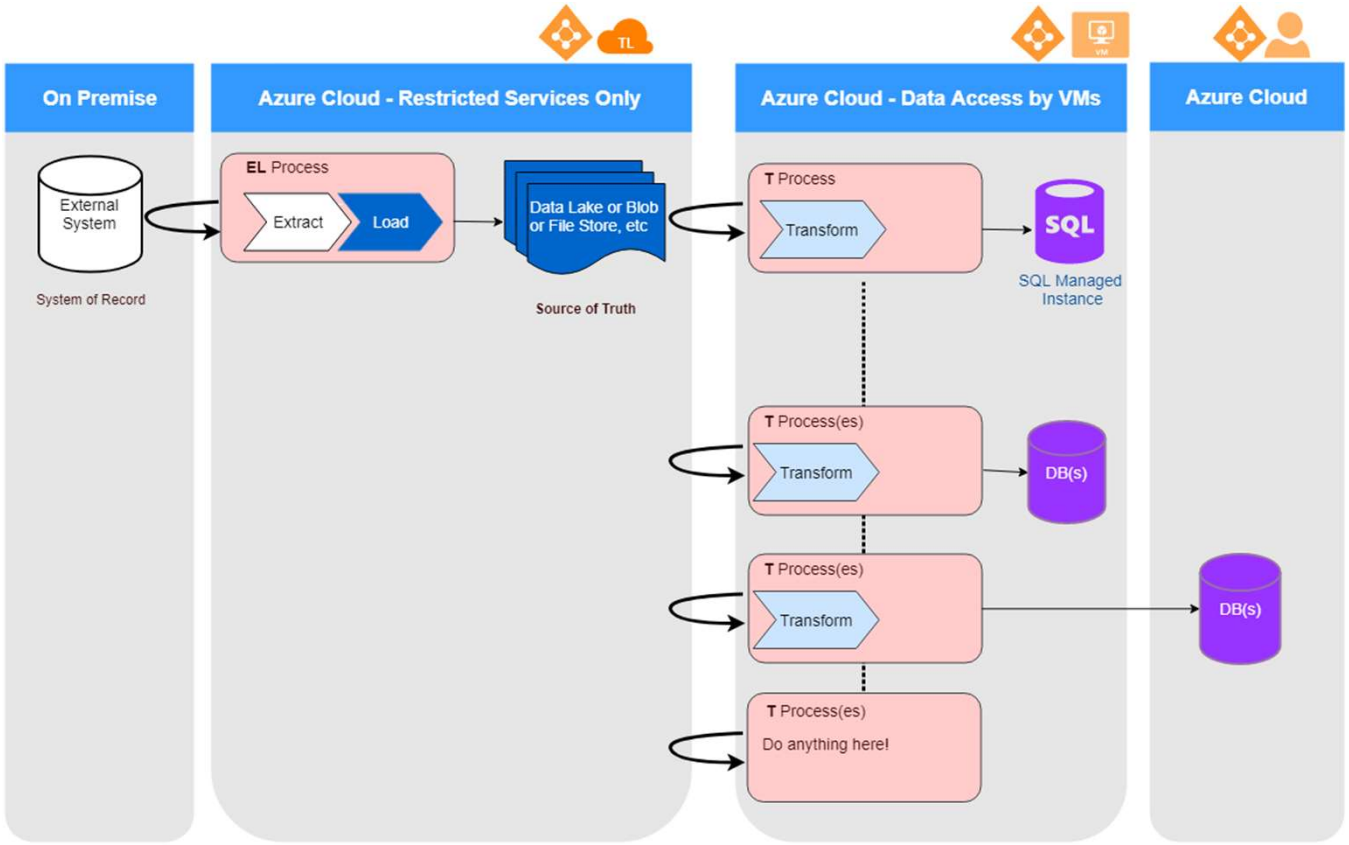
Pros:

- Low Costs

Cons:

- No decoupling from the source to the target, so a change normally impact everything.
- All requirements have to be gathered, as modelling explicit state. Otherwise data will be lost, as no way to go back.
- No audit, traceability or even supportability of the system of record
- Restrictive in Data Privacy, without creating multiple ETL. Which is heavy process
- Storing state and not behaviour, which is more useful to the team
- Testability is fairly impossible or at least difficult (ie regression testing impossible as external source can change) .
- Can become a ball of Mud like the current on premise version (SSIS packages)

ELT



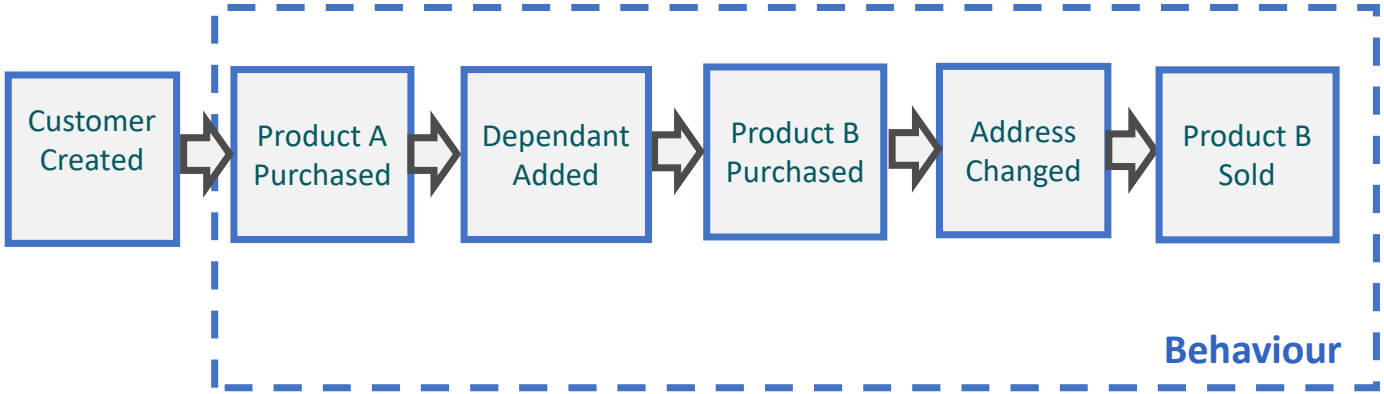
Pros:

- decoupling the source from the target or each interpretation. Also not just having one true schema to load into.
- Source of Truth is not exposed and can be secured and locked down

Cons:

- Not Low Costs
- Running an ELT could still overwrite history
- Tend toward lowest common dominator (so future analytic data is lost) or have multiple superset of all external model features if not
- No traceability or auditability (without extra work)
- **Behaviour data is not captured and could be lost**

Storing Events – Rather than current state



- Events are immutable and can be stored using an append-only operation
- Events are simple objects that describe some action that occurred
- Events typically have meaning for a domain expert
- Current state can be determine by events
- The storage of events provides an audit trail that can be used to monitor actions, regenerate the current state as materialized views or projections by replaying the events at any time, and then kept up to date in real time.

Projection
Is a left fold of the sequence events its interested in and therefore has the current state. It will read the whole stream of events on start up and then will update when any subsequent interest event is received.

Customer Products

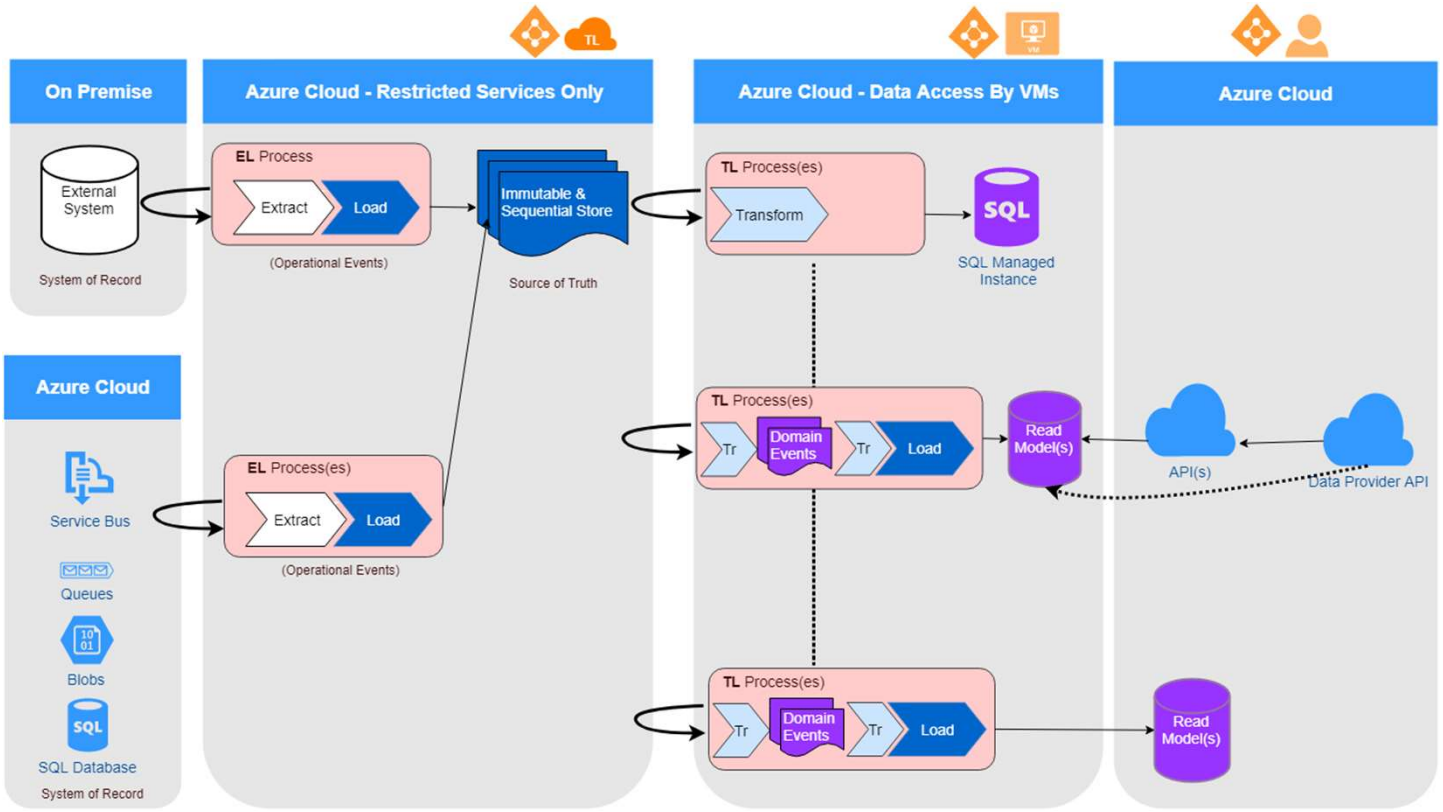
- Product A

Customer

- Name
- Address

Product Sold and Reason

Event Sourcing



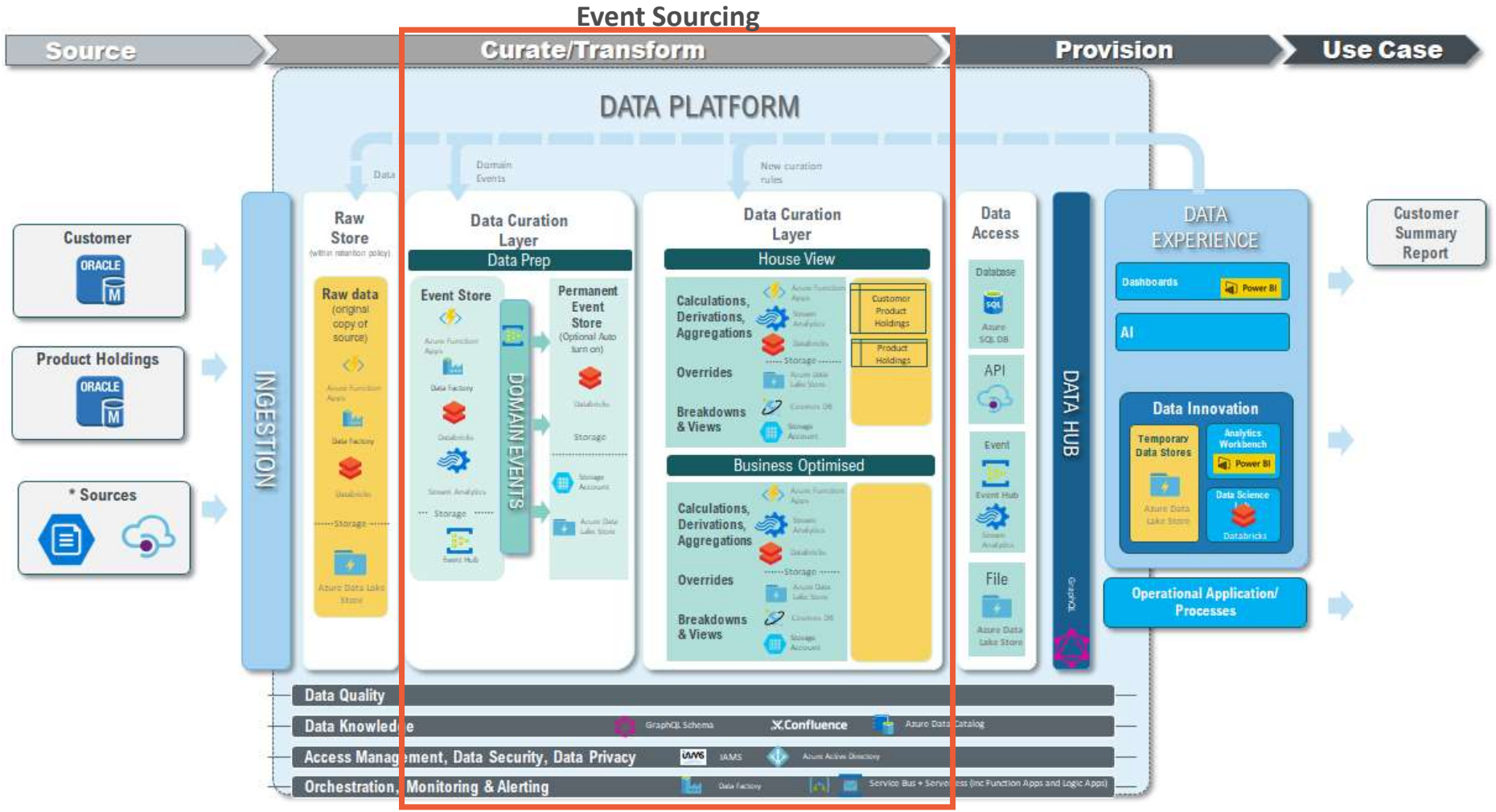
Pros:

- Same Pros as an ELT
- **Capture behaviour, which is probably better for data insight and data mining.**
- Traceability and auditability (without extra work), which is required for GDPR
- It makes it possible to implement temporal queries that determine the state of an entity at any point in time.
- Can create multiple projection/read models – performance, different data grouping or data policies . (CQRS)
- **Projection/read models can be updated in real time.**

Cons

- Not Low Costs
- events are immutable and undeletable. So the right to be forgotten and also data retention for GDPR becomes a challenge

Reference Architecture – with Technology over laid



GDPR compliance challenge in Event Sourcing

- Right to be Forgotten (Article 17) and Data Retention (Article 5)

- **Crypto-shredding**: This means we initially encrypt the data using a specific key. Then, if we want to remove that data, what we do is that we delete the key — thereby removing the personal information throughout the system.
  - Encryption key per: Cluster, Stream/Topic, **Customer**. Customer will be the most flexible.
- **Double encryption**:
  1. create a specific key for each customer
  2. create another key—a temporal key—for each retention period.
- **Identity PII (and data grouping) and build into the Event Schema:** Allowing only partially encryption on data that is required, not effecting other data. Could also have more attribute types (ie data grouping) and define properties/object accordingly to Data Privacy agreement. And will only decrypt to user/systems privileges
  - ProtocolBuffer allows message schemas to be extended in its Custom option.
  - HashiCorp Vault adds it to the json payload
- **Key Management System (KMS):** Managed all in one place. Could also control access to data with different granularity and protect against unauthorized access to data with a Access Control List (ACL) over top of the KMS.

Crypto-shredding

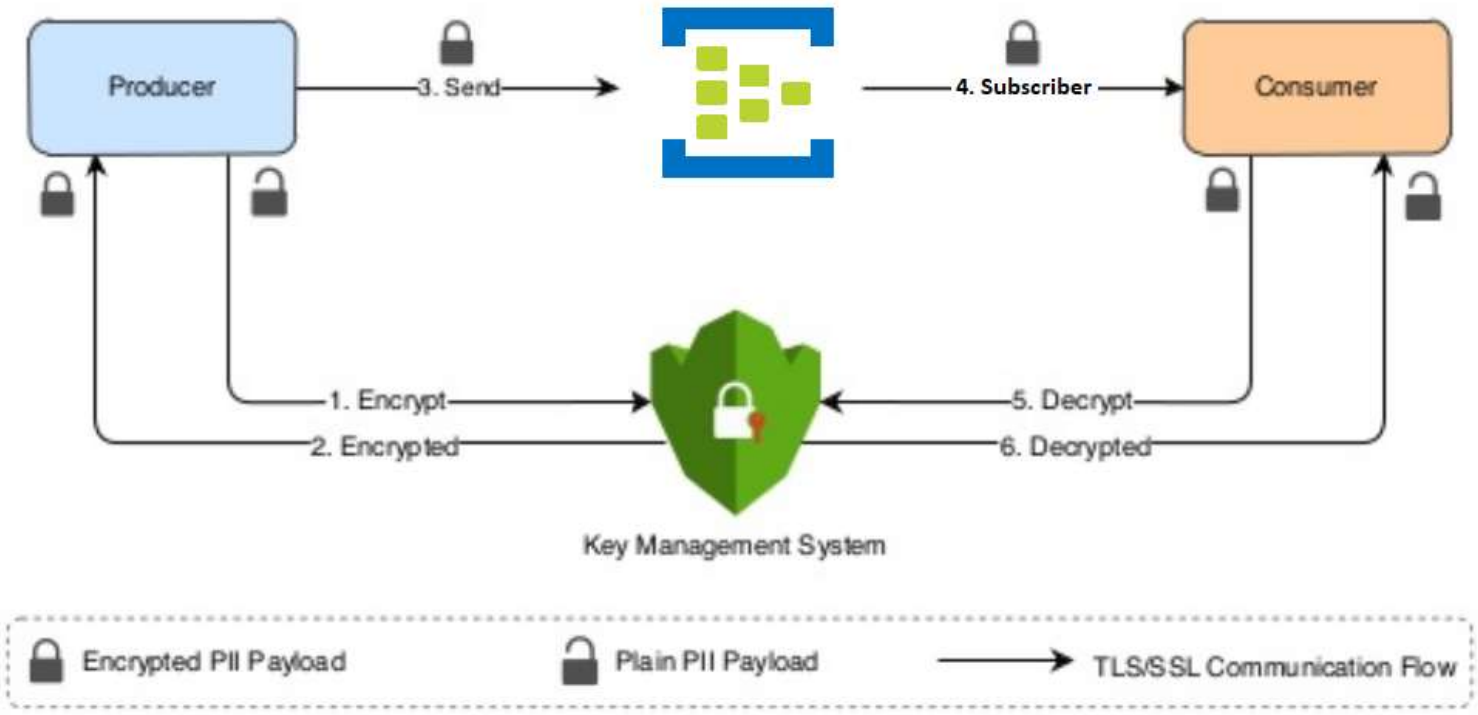
"… the practice of deleting data by deliberately deleting or overwriting the encryption keys."

- Simple to "forget" data
- On demand data removal
  - *don't have to wait for anything out of our control just delete the key*
- Fine-grained access control over data
  - *Rich ACL engineered over the top of the Key Management System*
- Lower risk of leftover data
  - *Backups*

But ….

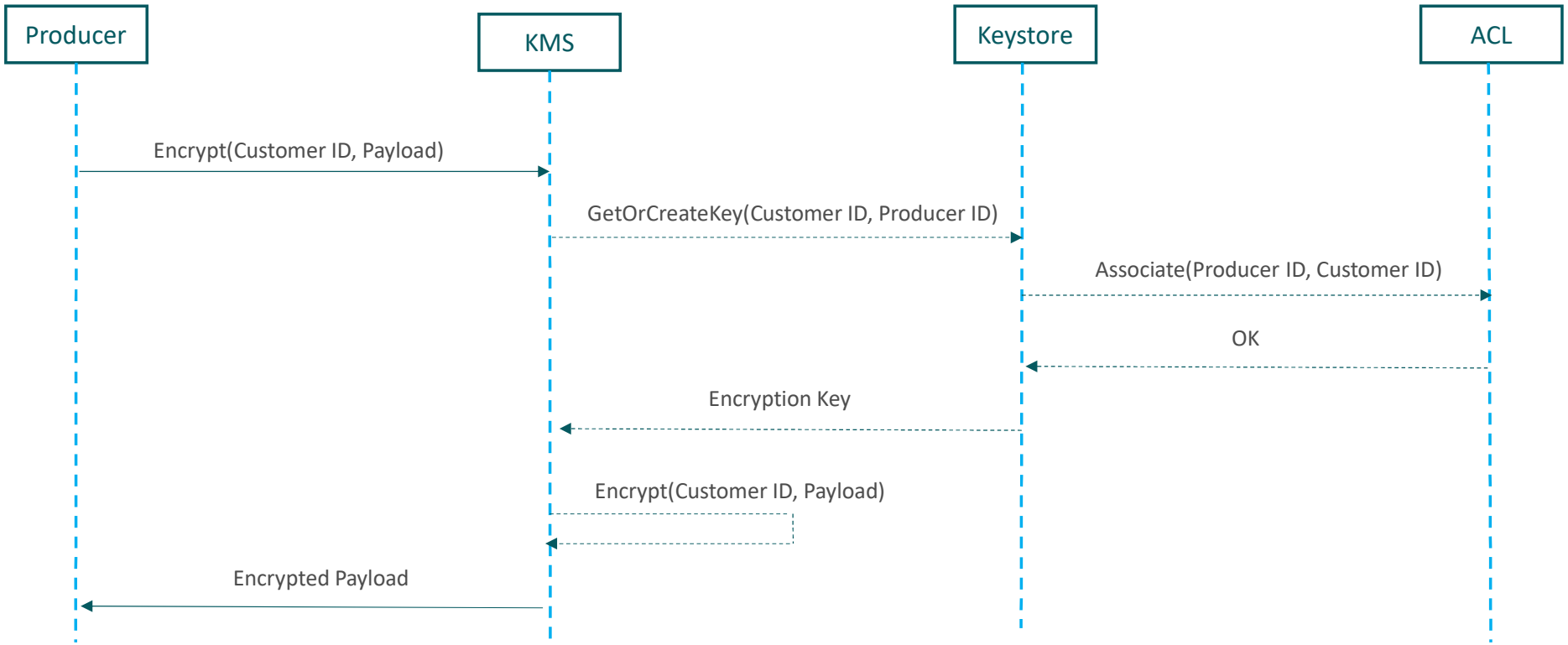
- Medium to high engineering effort.
  - *Will need some custom glue between producer and consumer via the Key Management System*
- Extra COGS.
  - *Because of the encryption and decryption involved*
- Requires careful key management.
- Will impact throughput (ie extra cost)

Per **Customer** Encryption Key



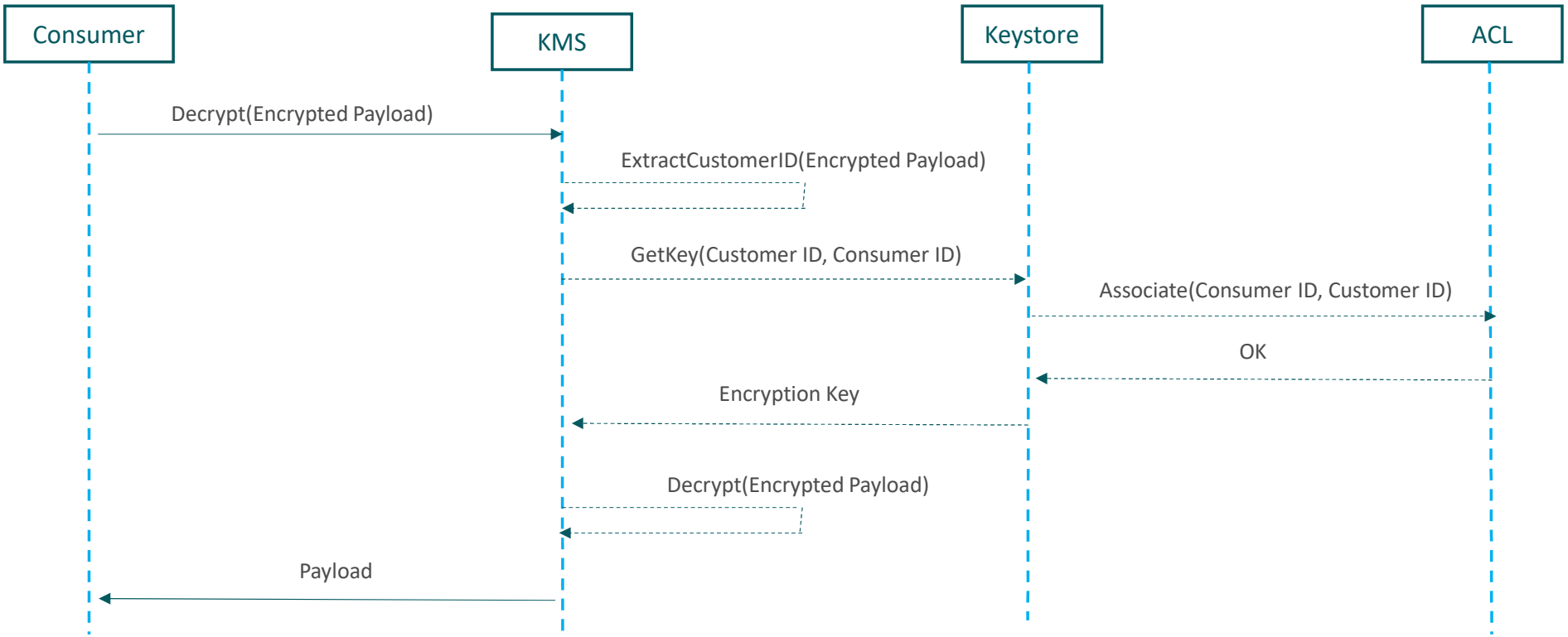
Key Management System

Producer Encrypt

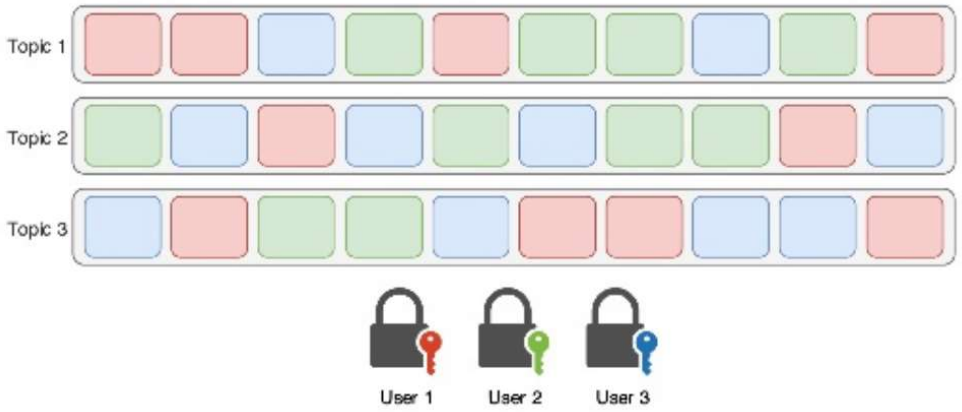


Key Management System

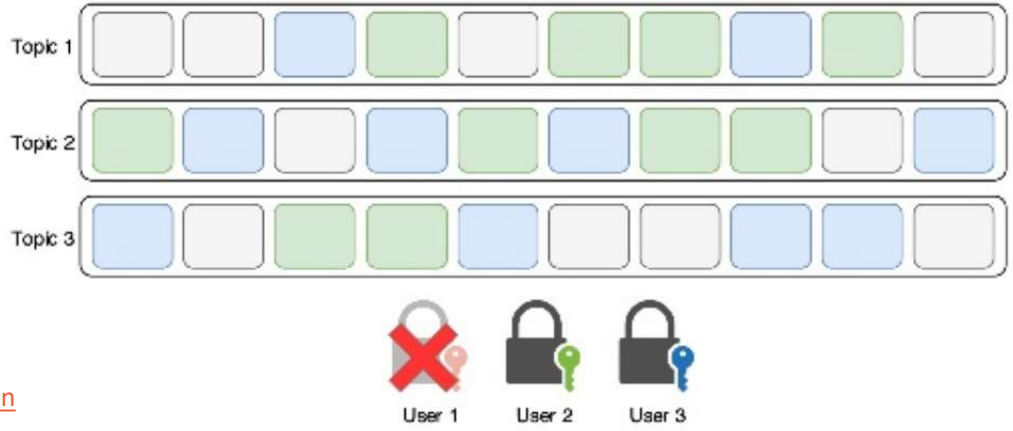
Consumer Decrypt



Crypto-shredding - Per Customer Encryption Key



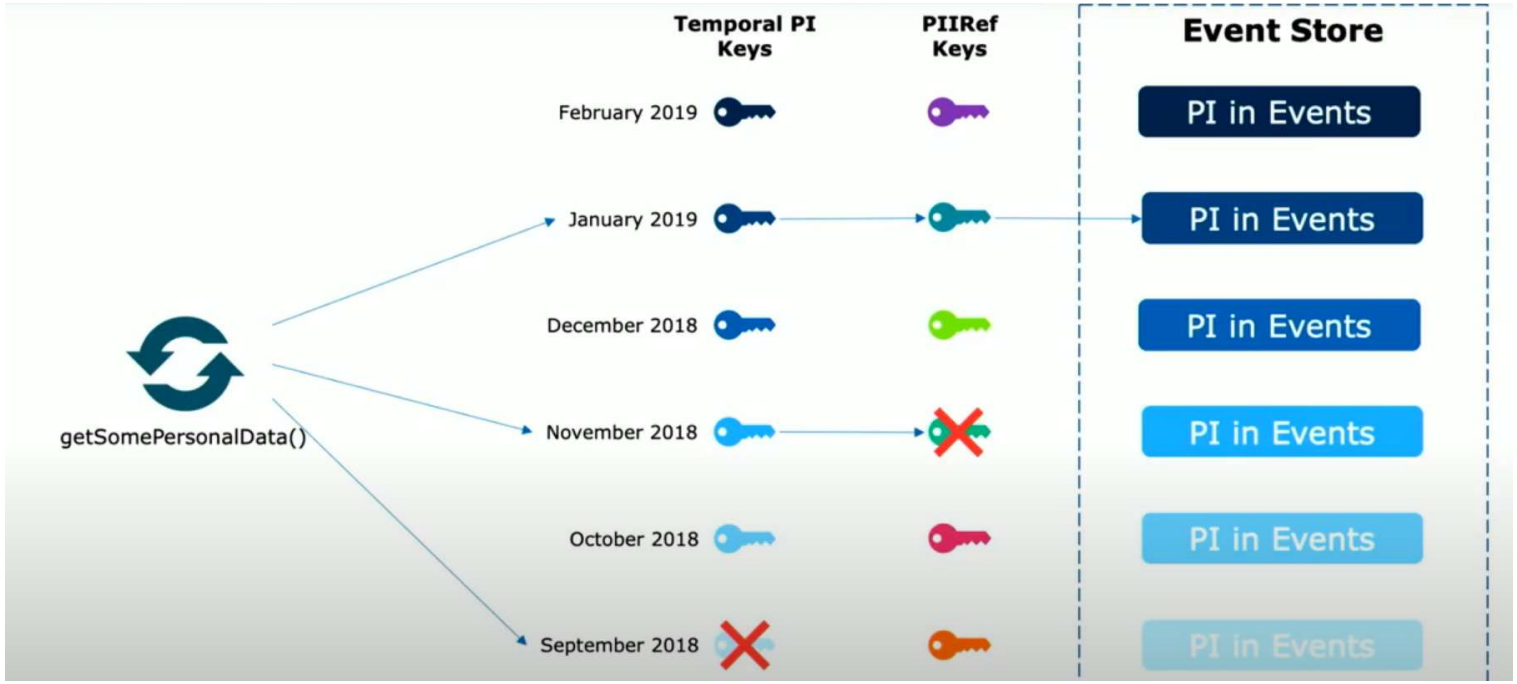
- Topic (Kafka) is the same as Stream, Aggregate, etc
- User = Customer is this example



Reference: GDPR Compliance: Transparent Handing of Personally Identifiable Information in Event-Driven Systems

Data Retention – Example Double Encryption

- “PIIRef Keys” is Per Customer Key



Reference: [GDPR Compliant Event Sourcing With HashiCorp Vault]

Partial Encryption - Per **Customer** Encryption Key

- **Event or Messages** typically have a **schema**
- Not **all fields** are **PII**
- Not **all consumers** care about **PII**

Idea ....

- Identity **PII** fields in an Event
- Also could Identity **Data Grouping** fields in an Event
- Build it into the Event **Schema**.
- Partially **encrypt** Event

Partial Encryption - Per **Customer** Encryption Key

Kafka Summit Example – Using ProtocolBuffer

Original Example Event

```
syntax "proto 3"
import "google/protobuf/desciptor.proto"
message MyMessage {
    string email = 1
    string name =2
    string location =3
    double purchase_value =4
}
```

Marked Up Event (Custom Options in Protocol Buffer)

```
syntax "proto 3"
import "google/protobuf/desciptor.proto"
extend google.protobuf.FieldOptions { GDPRCompliance gdpr = 51234; }
message GDPRCompliance { bool key = 1;  bool pii = 2;  string dataGrouping = 3;}
message MyMessage {
    string email = 1 [(gdpr) = {pii: true, key:true}]
    string name =2  [(gdpr) = {pii: true, dataGrouping: "clientaggregate" }]
    string location =3  [(gdpr) = {pii: false, dataGrouping: "clientaggregate" }]
    double purchase_value =4
}
```

- Can still retrieve non PII data all the time
- Could have more identitfy field types (ie data grouping) and define properties/object accordingly to Data Privacy agreement. Thus allowing decryption based on certain users authorisations.

Reference: GDPR Compliance: Transparent Handing of Personally Identifiable Information in Event-Driven Systems

Partial Encryption - Per **Customer** Encryption Key

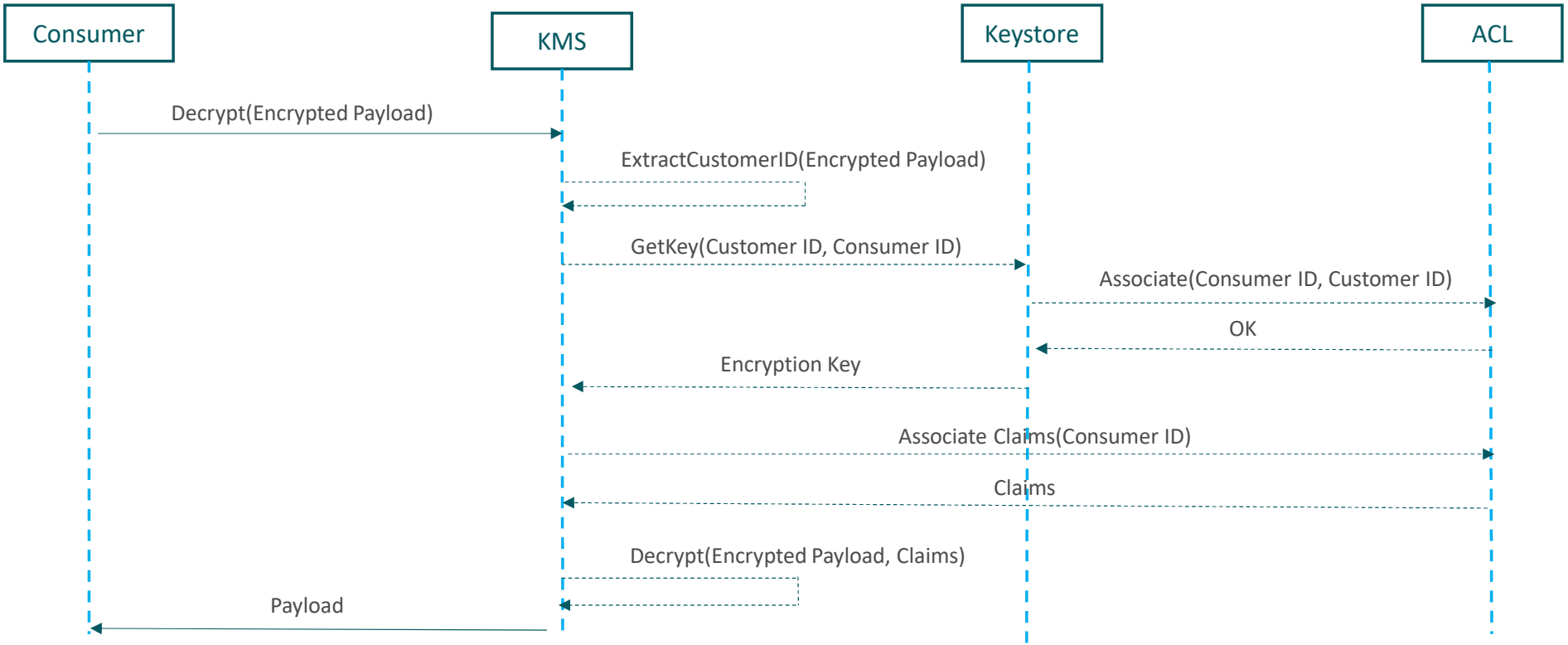
HashiCorp Vault Example – Using Json Schema

```
{
    $schema: "http://json-schema.org/draft-07/schema#",
    $id: "http://example.com/root.json",
    type: "object",
    title: "HealthTreatment encryption schema",
    description: "Example for a JSON Schema that is used to encrypt and object or a String.",
  - required: [
        "piiref",
        "name",
        "drug"
    ],
  - properties: {
    - piiref: {
          type: "string",
          contentMediaType: "text/html"
      },
    - name: {
          type: "string",
          contentEncoding: "application/octet-stream",
          contentMediaType: "aes256gcm"
      },
    - drug: {
          type: "string",
          contentMediaType: "text/html"
      }
    }
}
```

Reference: GDPR Compliant Event Sourcing With HashiCorp Vault

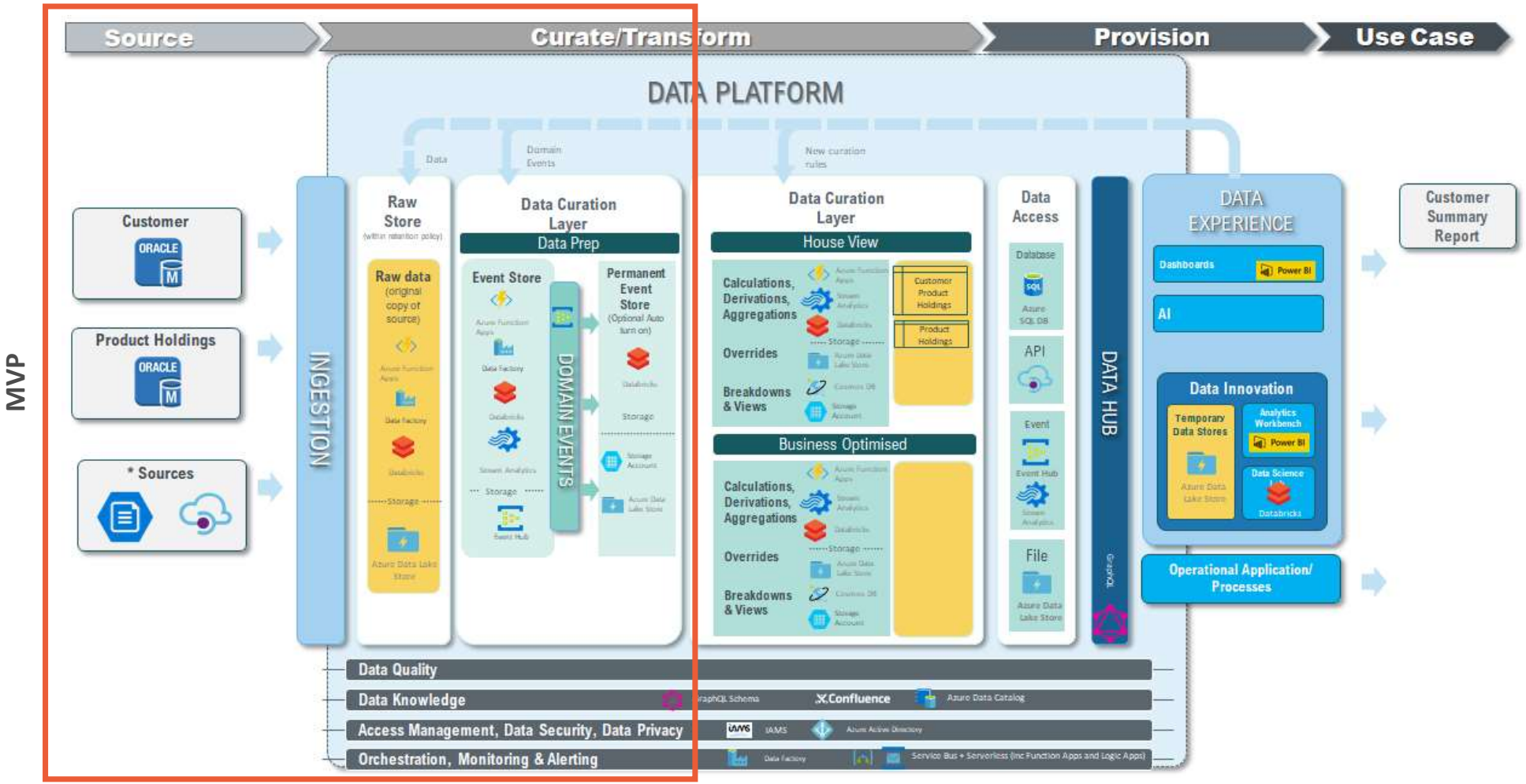
Key Management System

Consumer Decrypt with built Event Schema



MVP

Reference Architecture – with Technology over laid



MVP – raw store, event sourcing and ELT on the cloud

Use Case Summary

The main product goal on the first MVP is to ingest data to raw, to capture some domain events from on premise sources and store in an Cloud event store. So that these business events can be transformed to other multiple business challenges in the future by using any real-time analytics provider or batching/storage adapters.

MVP Capabilities and learning

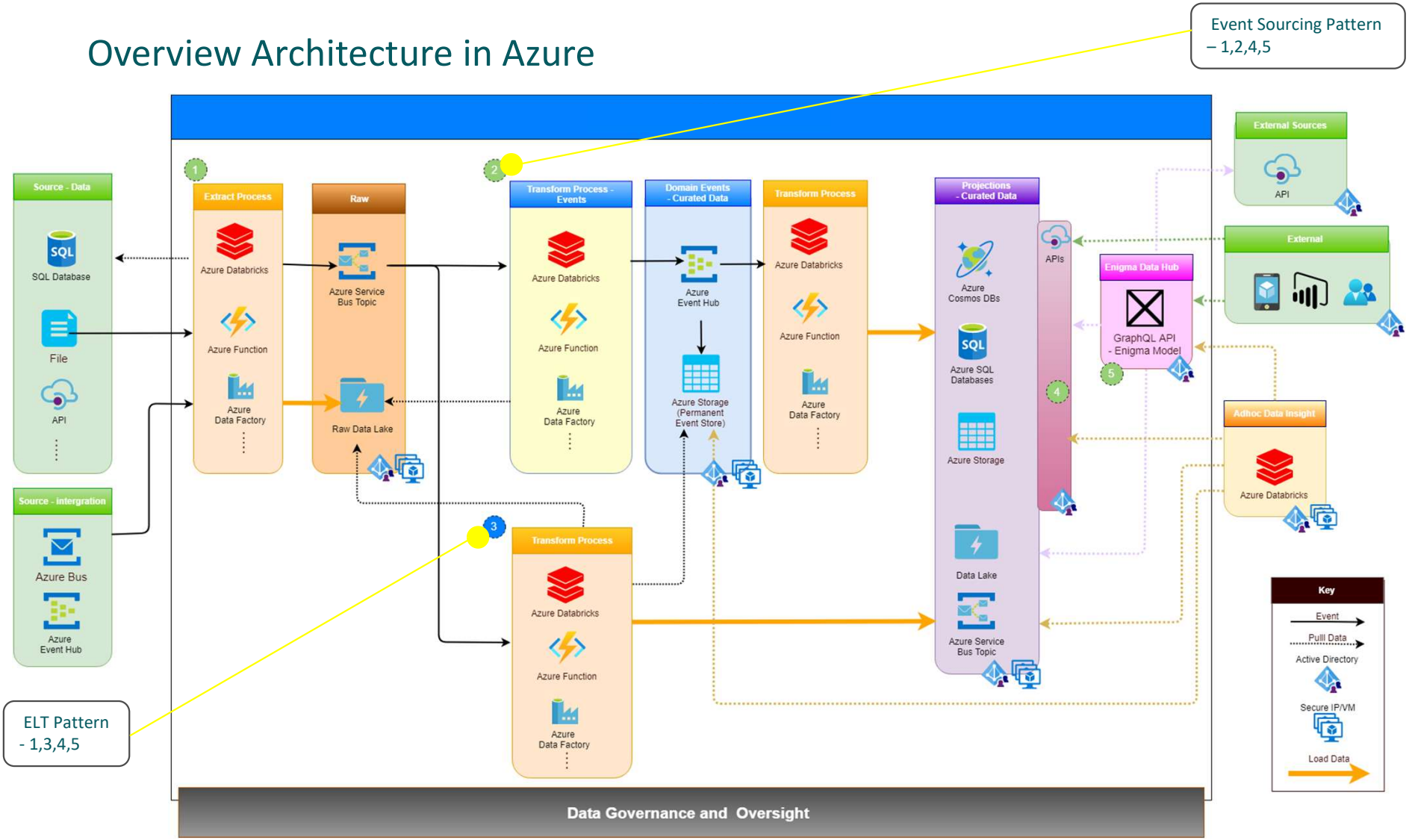
- Ingesting on premise database into a Secure Cloud Raw Data Lake
- Having a secure Event Sourcing Architecture and more of a Cloud-native architecture by using Event Hub or something similar - ES-CQRS.
    - Allowing Capturing Behaviour and Real time processing
- An Event Store to store these immutable events and complaint with GDPR and PII security.
- Crypto-shredding on per customer and per duration. To comply with GDPR both while event is in transit and at rest.
- Partially encryption on an event. Allowing pseudo-anonymous data or anonymous data (if crypto-shredding).
- Encryption and decryption for events all managed in central place - Key Management System (KMS)
- Data Policy Governance Process, especially on events and streams
- Continuous Integration and Continuous Deployment (for Data Bricks).
- Adapting and refining the new data journey process with CDO, enabling the organisation to move faster with on boarding data to the cloud.

Data Privacy and Security Considerations

The MVP will also work with security and privacy teams to define an appropriate treatment method to allow for the safe storage and use of former customer data (e.g. in a depersonalised way) in order to extract analytical value from these data while complying with our requirements under GDPR.

Appendix

Overview Architecture in Azure

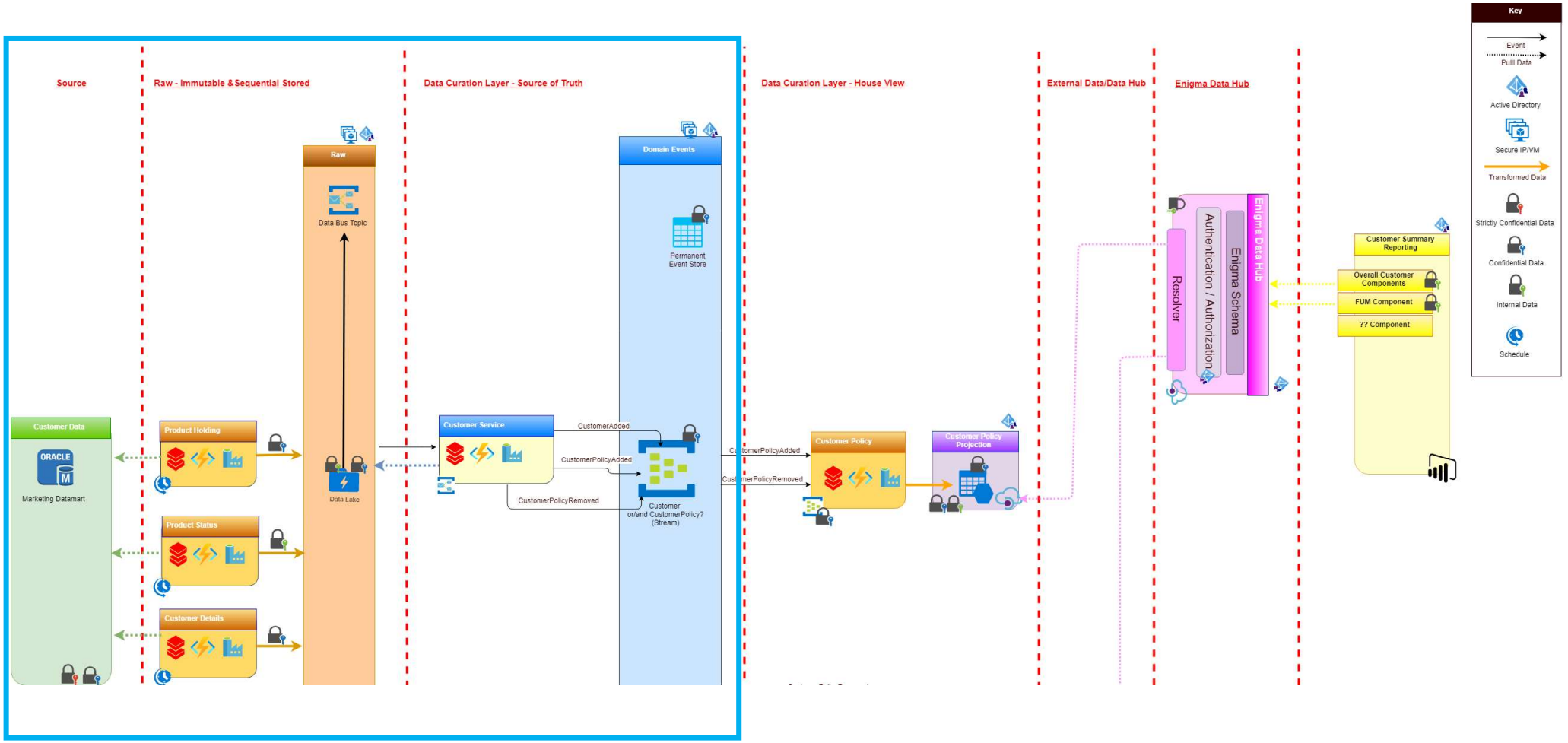


Overview Architecture

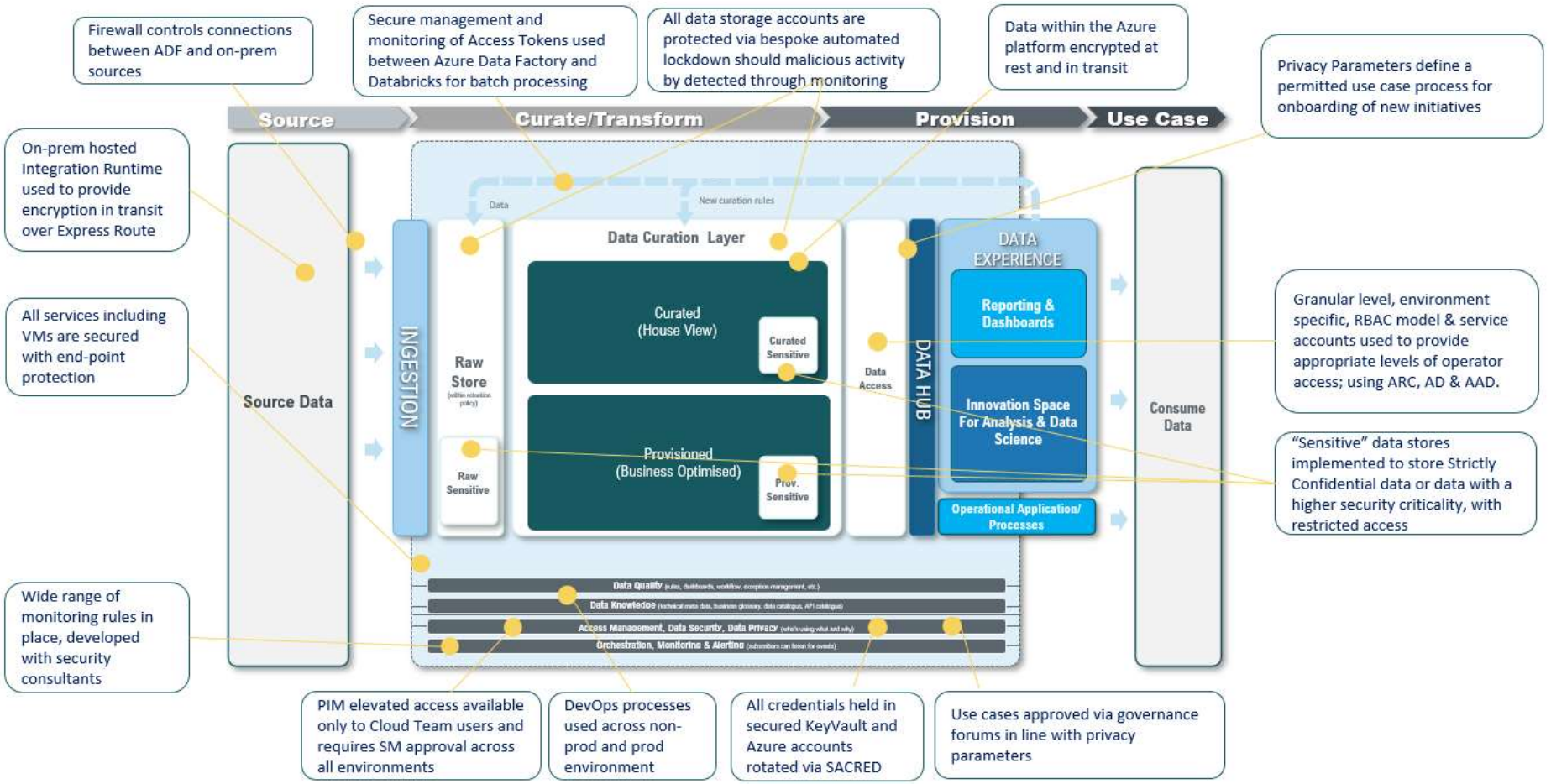
- Description

1. There are several great tools in Azure for ingesting raw data from external sources into the cloud.
    - Azure Data Factory provides the standard for importing data on a schedule or trigger from almost any data source and landing it in its raw format into Azure Data Lake Storage/Blob Storage. Azure Bus/Queue is an useful tool to inform others when the files has been landed.
2. A better option than just transforming the data, after landing the raw data into Azure, is to capture operational/domain events into a immutable domain stream store- **Event Sourcing**
    - Doing this will not only capture behaviour that will be useful for data insight, but will be useful for auditability, supportability, adaptable to source changes without impacting up stream , etc. This will also allow multiple projections to be updated based on this events. Azure EventHub is a useful tool in achieving this. This event store could also become a domain source of truth and our projections can be rebuilt from this, as and when they are required.
3. After landing the raw data into Azure, then the traditional transform process of an **ELT** could happen.
    - Again there are a number of great tools in Azure to transform data. Though this just simply taking the data in its raw, source format, and converting in to curated data source (ie Databricks Delta Lake) where it can be more efficiently and reliably queried and processed.
4. Data Access, especially outside of Data Platform B should be done via an API (ie REST API).
    - Therefore in principle each projection should have its own API over the top it.
5. Should also have a Central Data Hub API that sits over the top for querying, not just Data Platform B ingested data but non ingested data as well. As not all data will need to be ingested into Data Platform B for client application to use.
    - A pattern for this is GraphQL, as it lets you ask for what you want in a single query, saving bandwidth and reducing waterfall requests, which is vey useful where client application's network bandwidth and latency are critical . It also enables clients to request their own unique data specification. This can also help with transition phase approach, security on data and resources, along with helping in an avoiding versions.

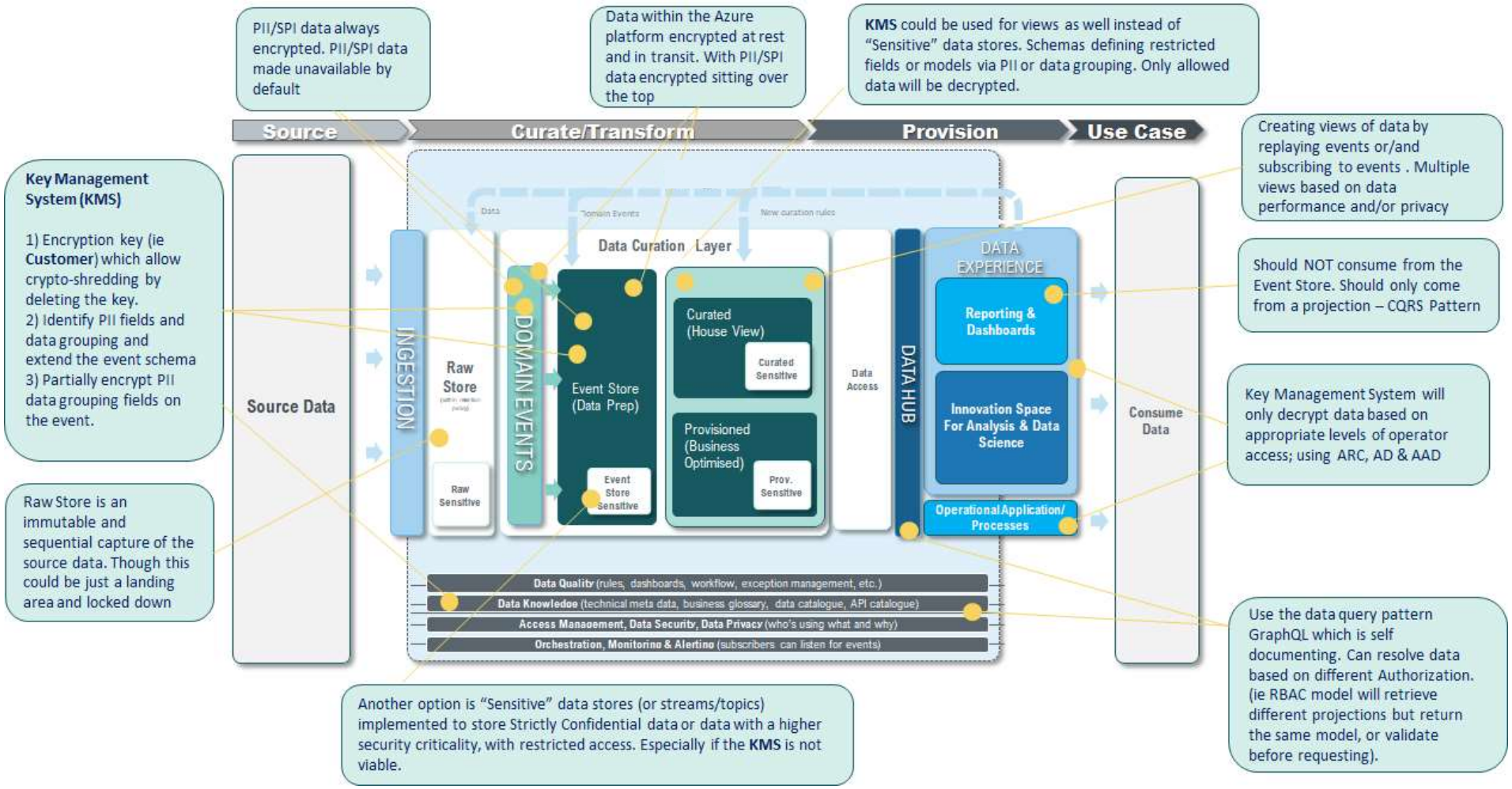
Over Customer Component Example



Reference Architecture – Controls for Standard ELT

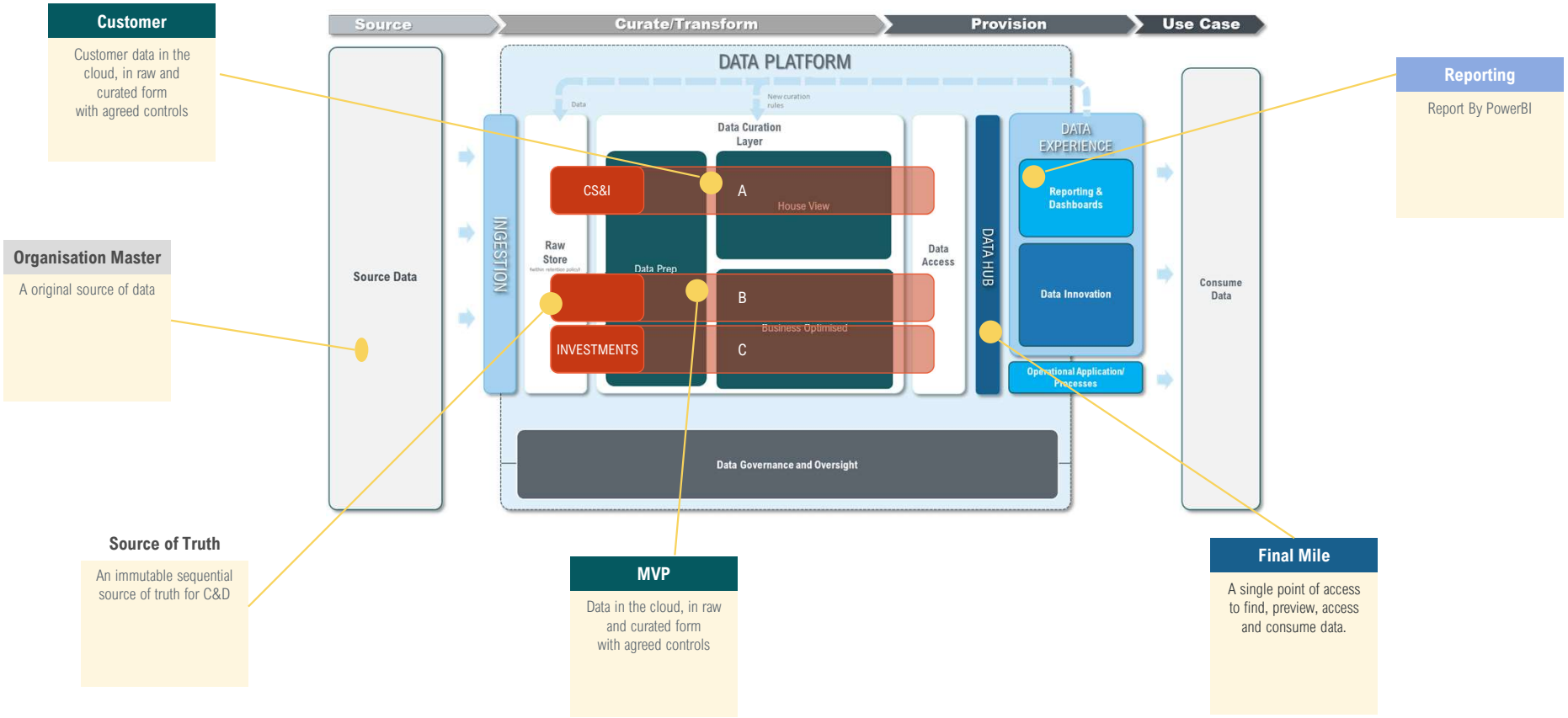


Reference Architecture – Controls for Event Sourcing Pattern

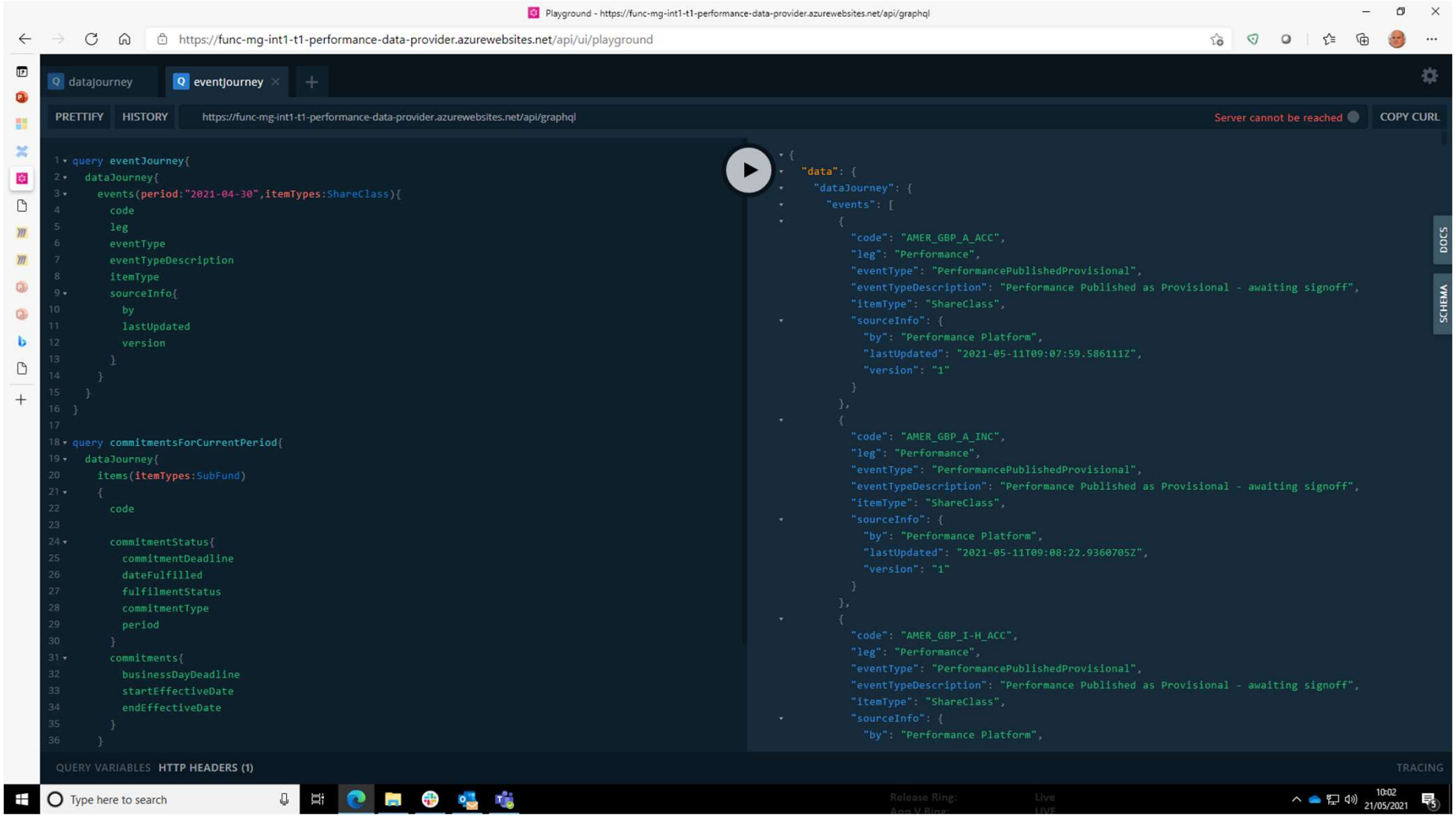


Enterprise Data Platform – Current Initiatives

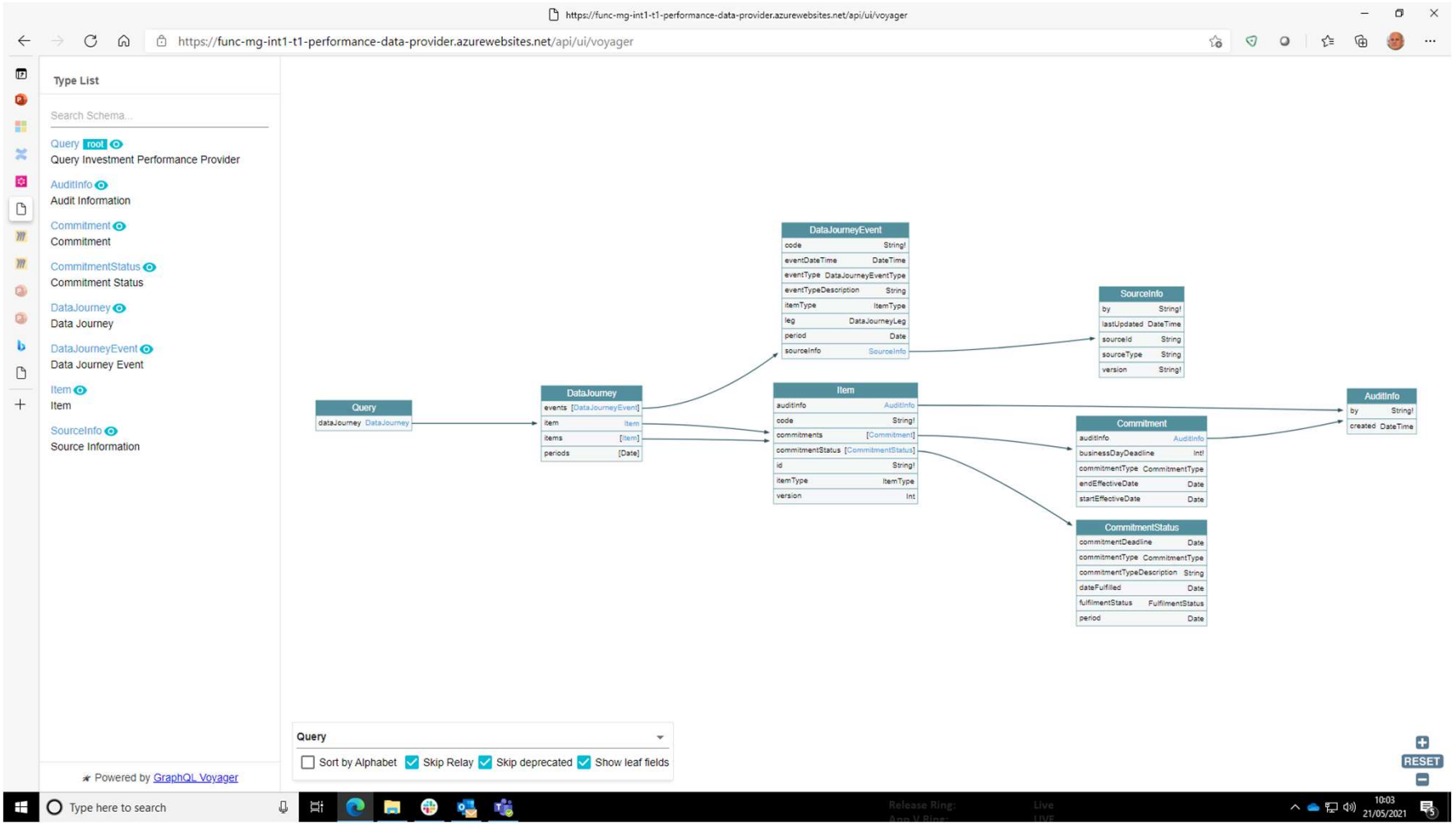
Converging the different platform in progress to enable controlled sharing and reuse of data



GraphQL Example of Query – Playground



GraphQL Example of Self Documenting – Voyager



ETL, ELT, Event Sourcing

	ETL	ELT	Event Sourcing
Decoupling			
Determinism			
Low Cost			
Modelling State Explicitly			
Past as First Class			

Decoupling: decoupling the source from the target or each interpretation. Also not just having one true schema to load into.

Determinism: Running an ETL can overwrite history

Modelling State Explicitly: Tend toward lowest common dominator (so future analytic data is lost) or have multiple superset of all external model features

Past as First Class: the act of focusing on a particular object while ignoring irrelevant information -> can't re-interpret past extracts.

Low Costs: Training, framing, explaining

- Training: Low cost to train new engineers in ETL concept
- Framing: No requirement for up front modelling
- Explaining: intuitive and simple to understand