Niemand macht gerne Data Governance - lassen wir es doch die AI machen

INNOQ

JOCHEN CHRIST

/IN/JOCHENCHRIST

Hi,
I am Jochen

Jochen Christ

Data Mesh Consultant
Co-Founder Data Mesh Manager

💪 *Software Engineering*
👍 *Data Mesh & Data Contracts*
🤩 *Data-driven Decisions*

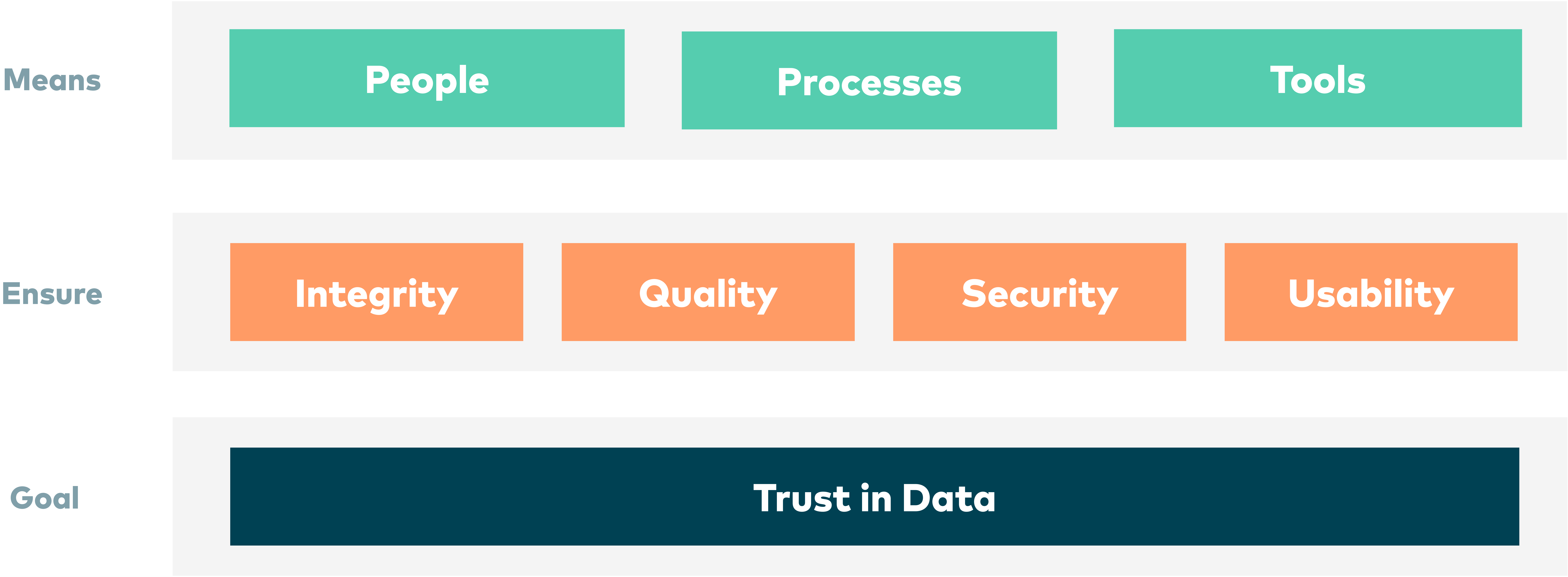
Agenda

1. Classic Data Governance
2. Morden Data Governance
3. Demo: AI-based Data Governance
4. Technical Details: Spring AI

Disclaimer:
For demonstrations, will work with Data Mesh Manager

Data Governance

What is Data Governance



Means

People

Processes

Tools

Ensure

Integrity

Quality

Security

Usability

Goal

Trust in Data

Source: Eryurek et al.: Data Governance The Definitive Guide. O’Reilly

The Reality

Canonical Data Modeling

- Wish to have one enterprise-wide valid data model
- Business knowledge & details gets lost
- Disconnected Ownership
- Hard to change / slow

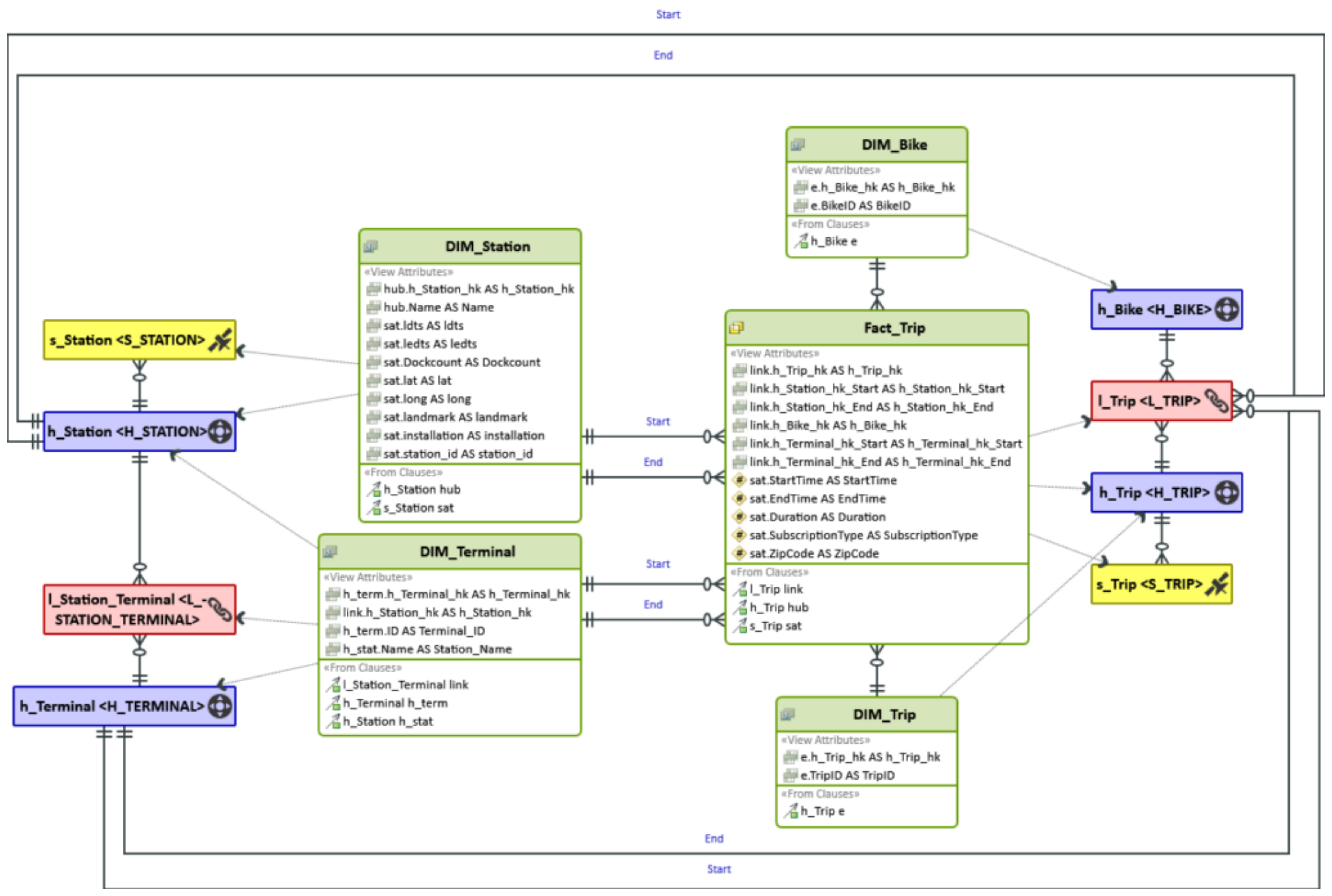
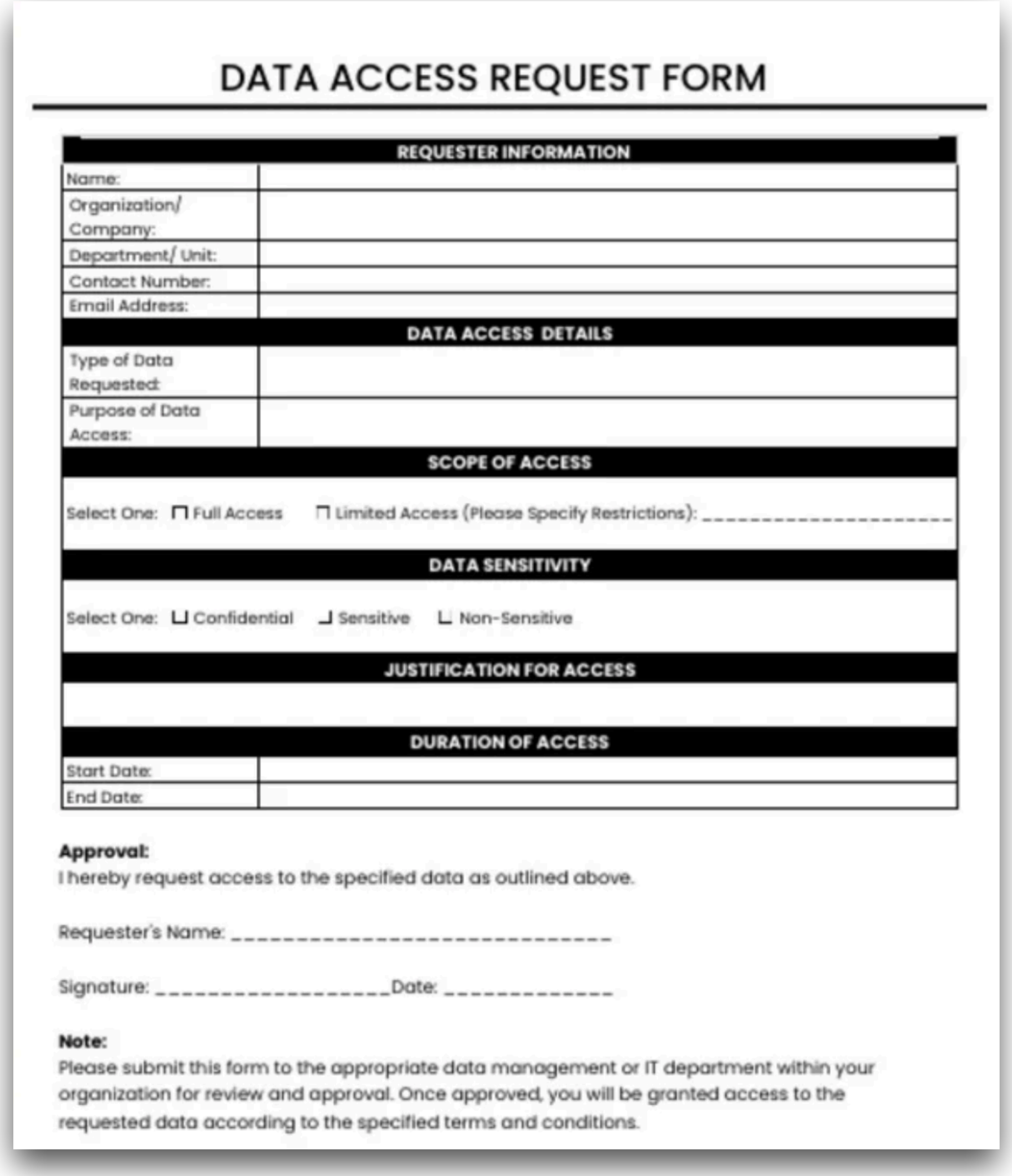


Image Source: innovator.de

**Data Governance got lost in data modeling,
but got disconnected from business and IT teams**

Bureaucracy

- Lots of policies in Word documents, not using the language of business, nor developers
- Forms and JIRA Tickets
- Manual Processes
- Manual Approvals



DATA ACCESS REQUEST FORM

REQUESTER INFORMATION	
Name:	
Organization/ Company:	
Department/ Unit:	
Contact Number:	
Email Address:	
DATA ACCESS DETAILS	
Type of Data Requested:	
Purpose of Data Access:	
SCOPE OF ACCESS	
Select One: □ Full Access □ Limited Access (Please Specify Restrictions): ______________________	
DATA SENSITIVITY	
Select One: □ Confidential □ Sensitive □ Non-Sensitive	
JUSTIFICATION FOR ACCESS	
DURATION OF ACCESS	
Start Date:	
End Date:	

Approval:
I hereby request access to the specified data as outlined above.

Requester's Name: ______________________________

Signature: __________________Date: ______________

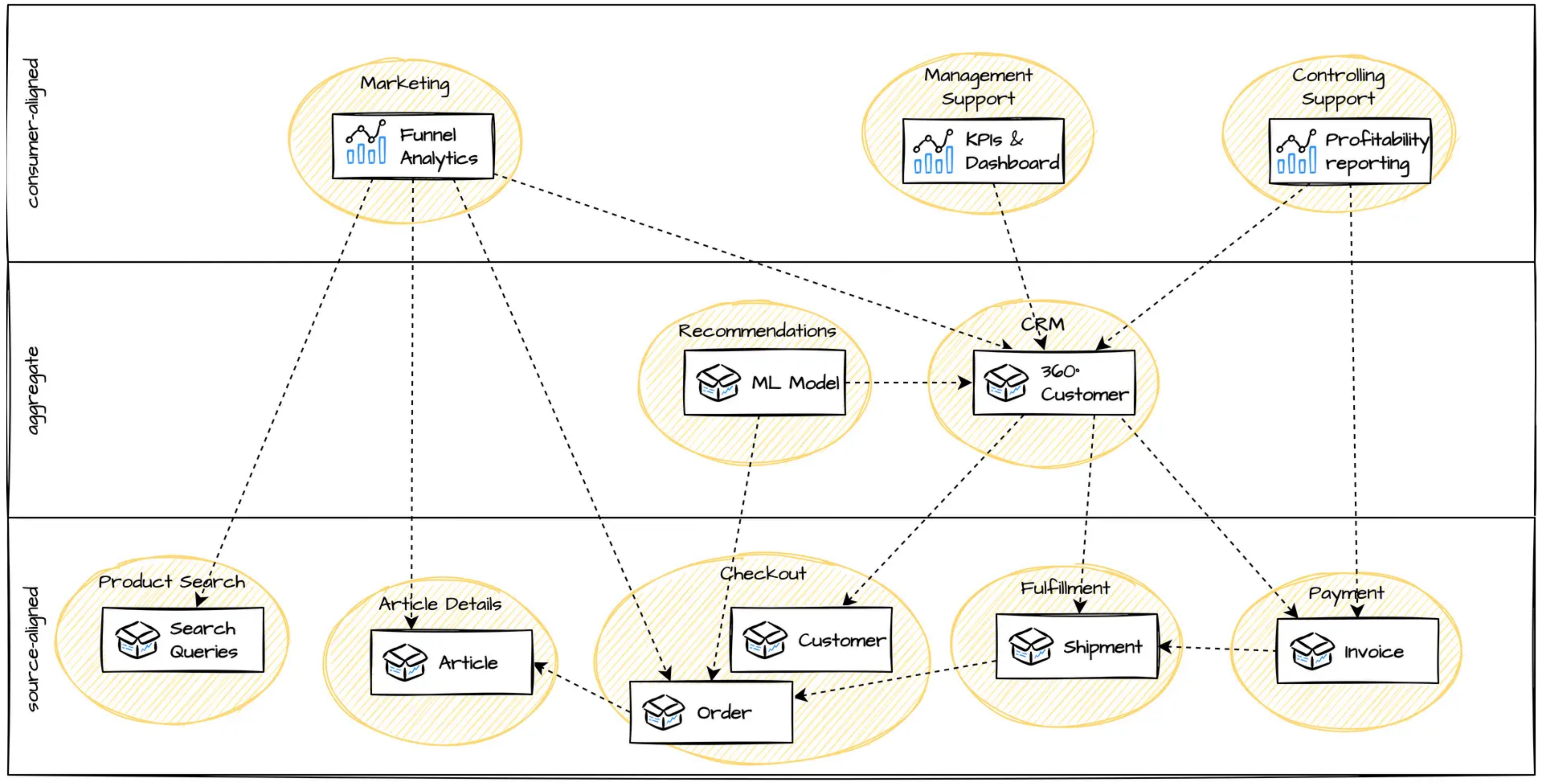
Note:
Please submit this form to the appropriate data management or IT department within your organization for review and approval. Once approved, you will be granted access to the requested data according to the specified terms and conditions.

Data Governance became a bottleneck (not enabler), with rather negative reputation

New Challenges

New Challenges

- **Decentralization (Product Teams, Data Mesh, …)**
  - Ownership & teams
  - Systems & technologies
  - Data models in bounded contexts
- **The Raise of AI**
  - Engine for innovations



datamesh-architecture.com

Data is exchanged across business & IT teams

Modern Data Governance

Federated Computational Data Governance

Responsibility

Data Product Owner

Data is owned decentralized by business & IT experts where data is generated
Product owners are responsible for what happens with their data

Concepts

Data Contracts

Define the syntax, semantics, quality, and terms of use as YAML

Data Marketplace

Data discovery with a self-service access request workflow

Global Policies

The conventions and rules of play for data on the data platform

Automation

Contract Enforcement

Test that data products correctly implement the data contract

Automated Permission Granting

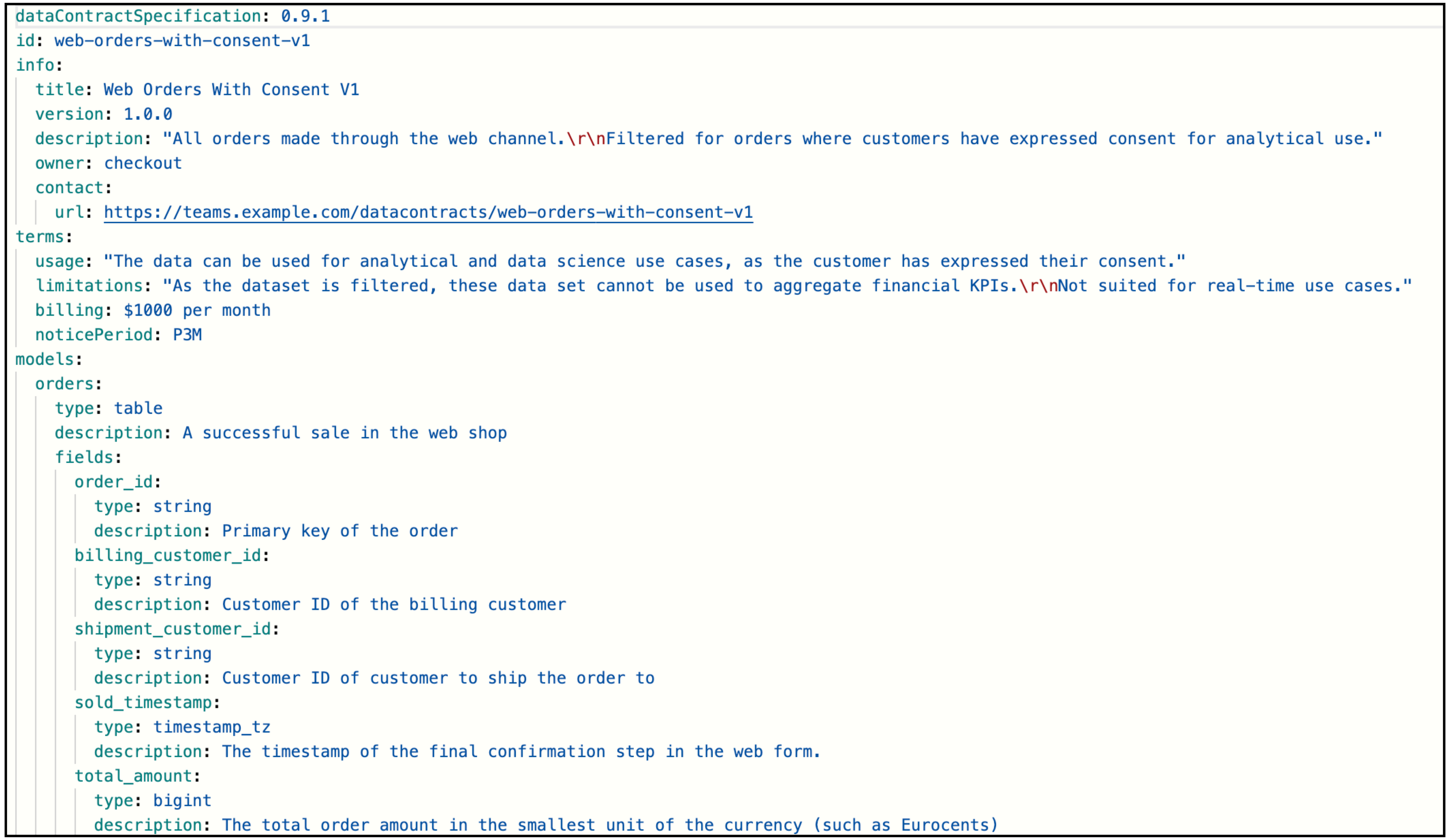
Give table access based on access request approvals

AI-based Policy Checking

Check that policies are correctly adopted by data product owners

Data Contracts

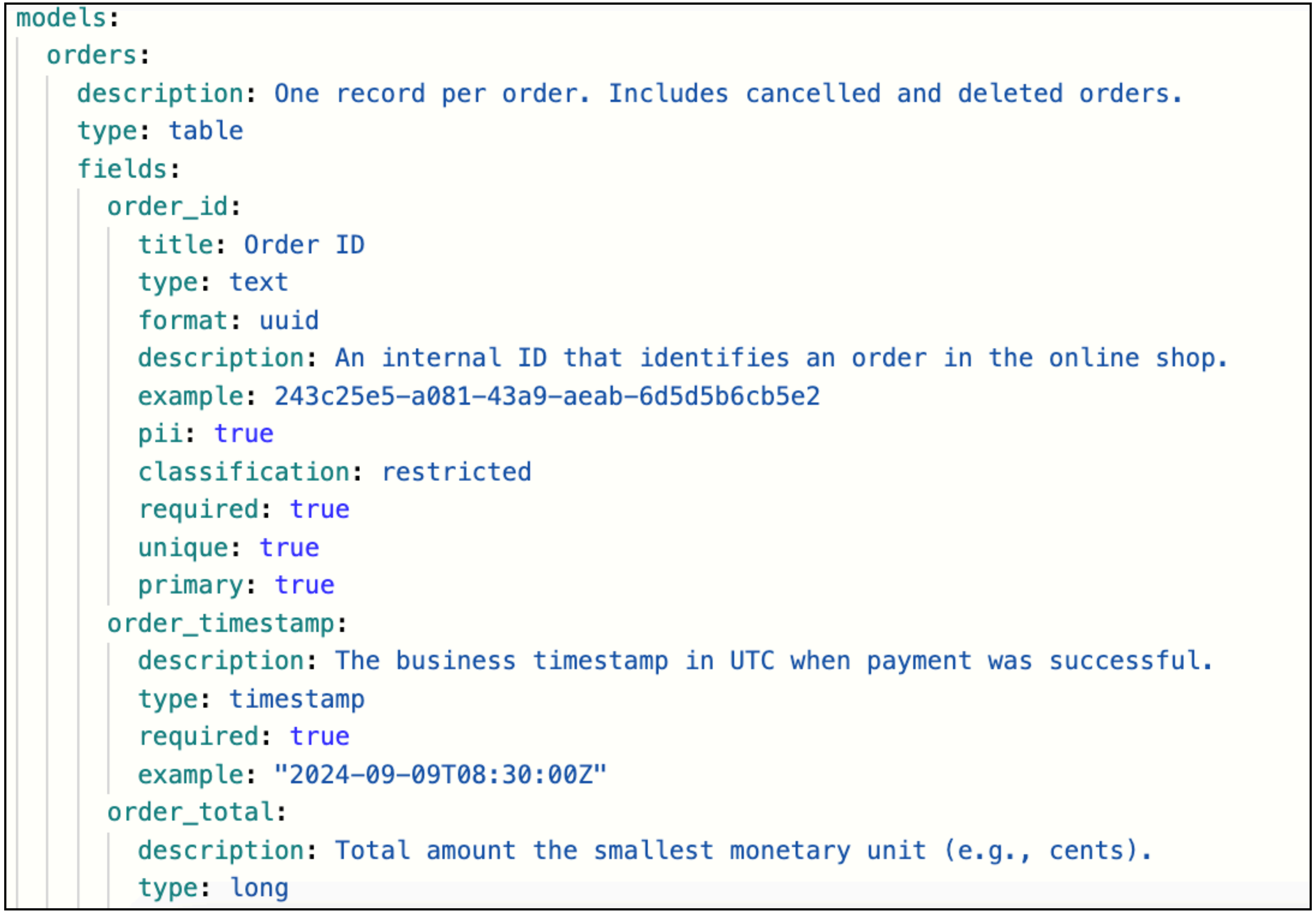
Data Contract:
Schema + Semantics + Quality



```yaml
dataContractSpecification: 0.9.1
id: web-orders-with-consent-v1
info:
  title: Web Orders With Consent V1
  version: 1.0.0
  description: "All orders made through the web channel.\r\nFiltered for orders where customers have expressed consent for analytical use."
  owner: checkout
  contact:
    url: https://teams.example.com/datacontracts/web-orders-with-consent-v1
terms:
  usage: "The data can be used for analytical and data science use cases, as the customer has expressed their consent."
  limitations: "As the dataset is filtered, these data set cannot be used to aggregate financial KPIs.\r\nNot suited for real-time use cases."
  billing: $1000 per month
  noticePeriod: P3M
models:
  orders:
    type: table
    description: A successful sale in the web shop
    fields:
      order_id:
        type: string
        description: Primary key of the order
      billing_customer_id:
        type: string
        description: Customer ID of the billing customer
      shipment_customer_id:
        type: string
        description: Customer ID of customer to ship the order to
      sold_timestamp:
        type: timestamp_tz
        description: The timestamp of the final confirmation step in the web form.
      total_amount:
        type: bigint
        description: The total order amount in the smallest unit of the currency (such as Eurocents)
```

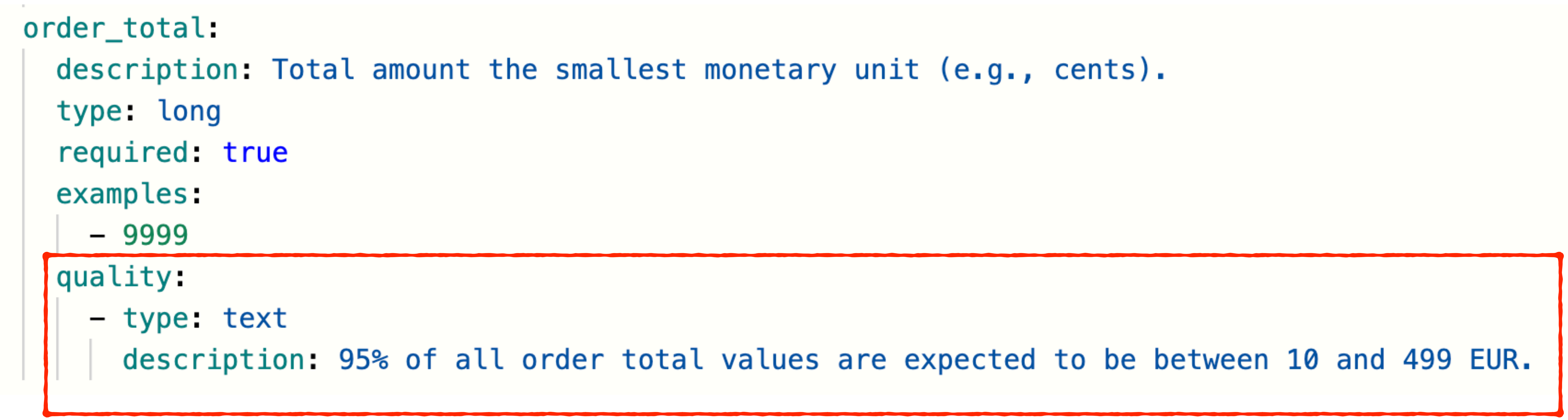
- API for Datasets
- Written by domain experts
- Made implicit knowledge explicit
- Machine-readable

Data Model



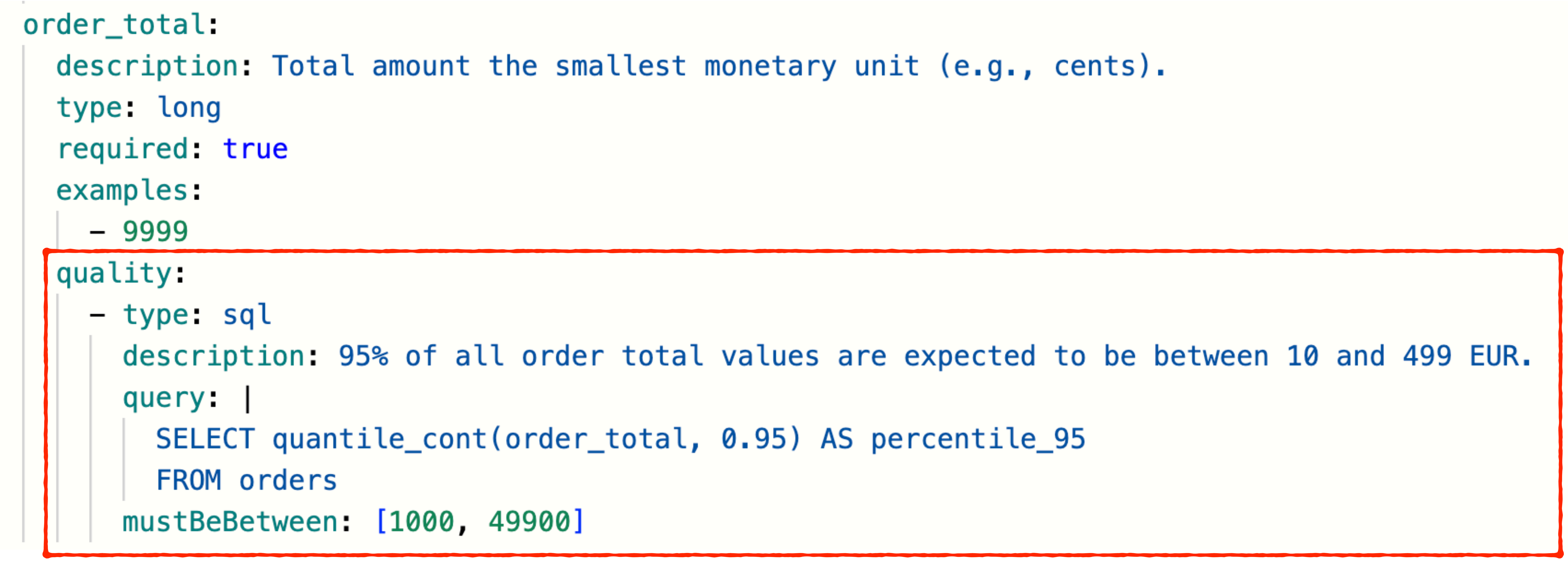
```
models:
  orders:
    description: One record per order. Includes cancelled and deleted orders.
    type: table
    fields:
      order_id:
        title: Order ID
        type: text
        format: uuid
        description: An internal ID that identifies an order in the online shop.
        example: 243c25e5-a081-43a9-aeab-6d5d5b6cb5e2
        pii: true
        classification: restricted
        required: true
        unique: true
        primary: true
      order_timestamp:
        description: The business timestamp in UTC when payment was successful.
        type: timestamp
        required: true
        example: "2024-09-09T08:30:00Z"
      order_total:
        description: Total amount the smallest monetary unit (e.g., cents).
        type: long
```

Quality



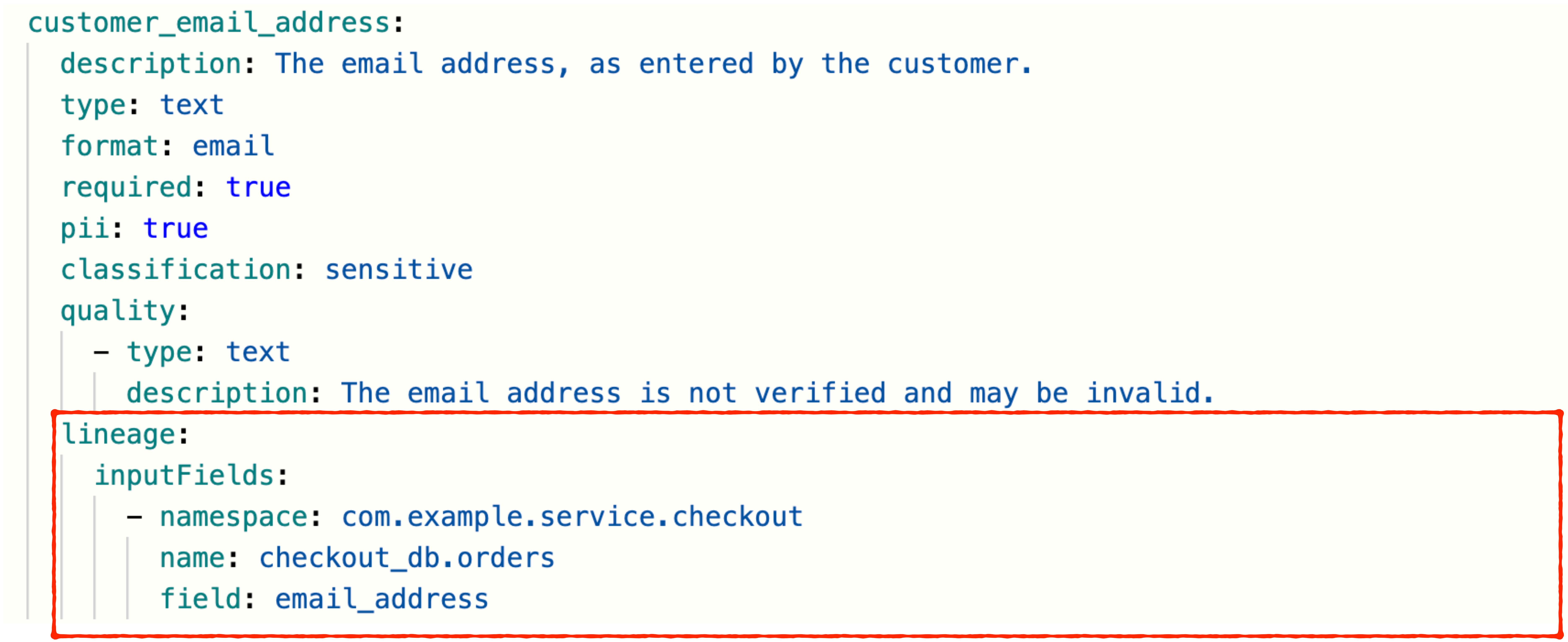
```
order_total:
  description: Total amount the smallest monetary unit (e.g., cents).
  type: long
  required: true
  examples:
    - 9999
  quality:
    - type: text
      description: 95% of all order total values are expected to be between 10 and 499 EUR.
```

Quality



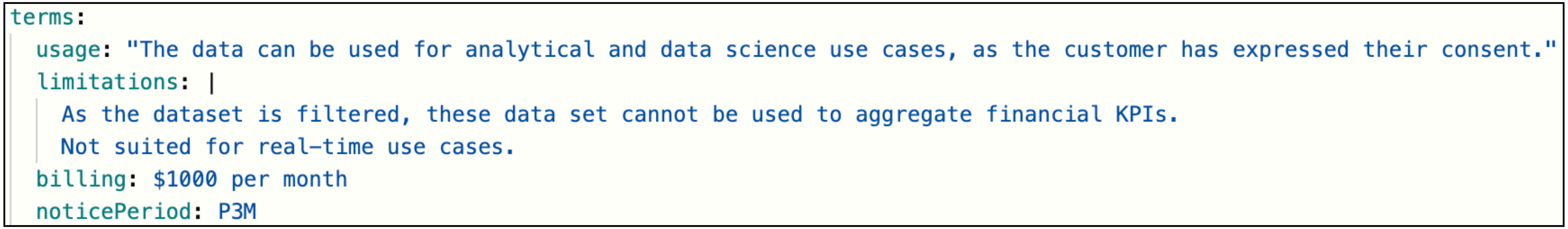
```yaml
order_total:
  description: Total amount the smallest monetary unit (e.g., cents).
  type: long
  required: true
  examples:
    - 9999
  quality:
    - type: sql
      description: 95% of all order total values are expected to be between 10 and 499 EUR.
      query: |
        SELECT quantile_cont(order_total, 0.95) AS percentile_95
        FROM orders
      mustBeBetween: [1000, 49900]
```

Lineage



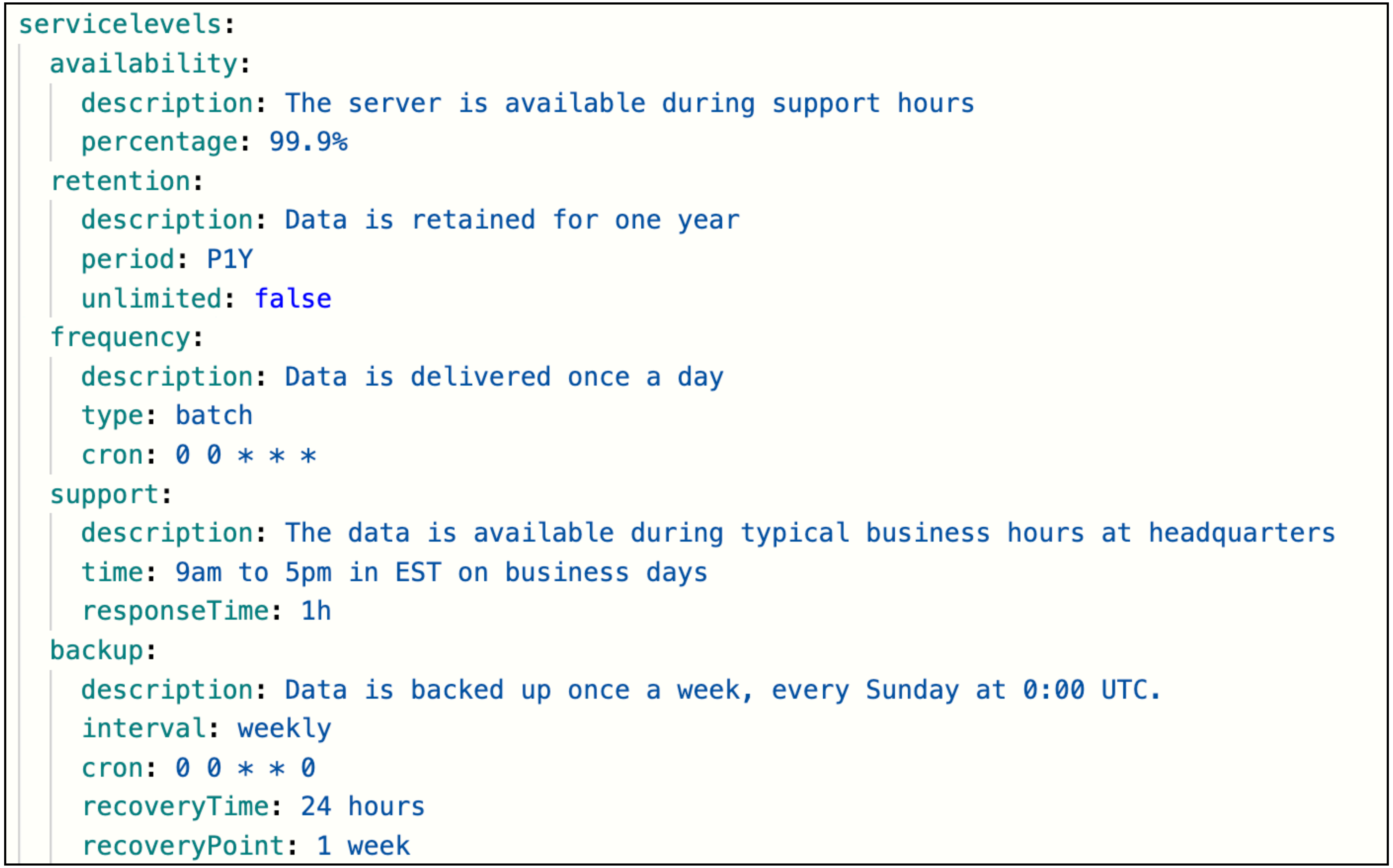
```yaml
customer_email_address:
  description: The email address, as entered by the customer.
  type: text
  format: email
  required: true
  pii: true
  classification: sensitive
  quality:
    - type: text
      description: The email address is not verified and may be invalid.
  lineage:
    inputFields:
      - namespace: com.example.service.checkout
        name: checkout_db.orders
        field: email_address
```

Terms & Conditions



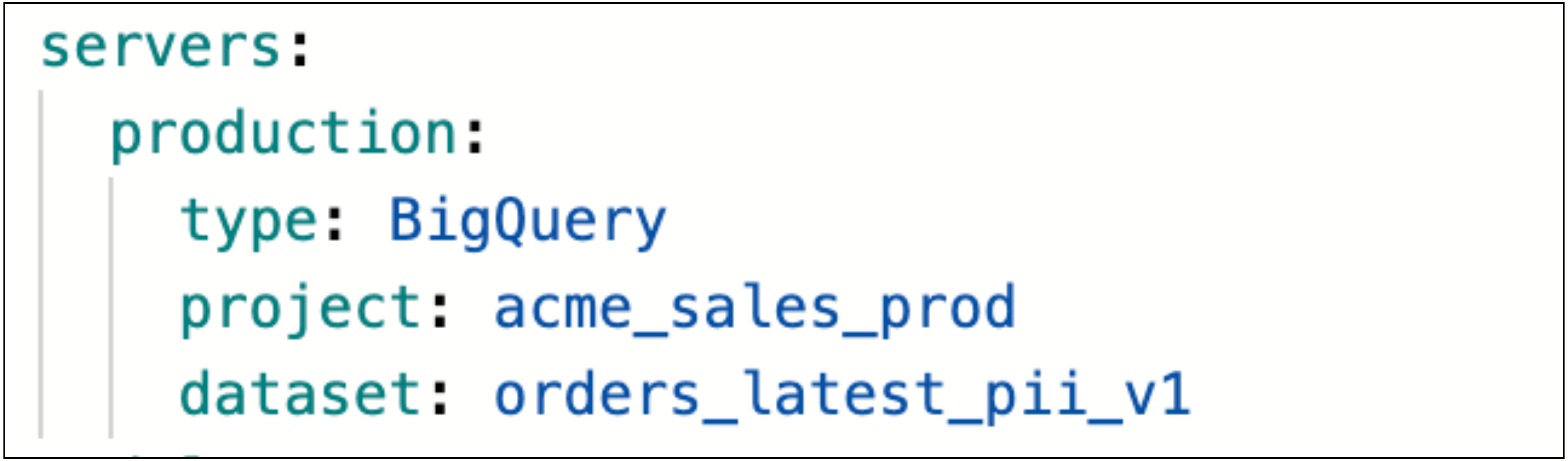
```yaml
terms:
  usage: "The data can be used for analytical and data science use cases, as the customer has expressed their consent."
  limitations: |
    As the dataset is filtered, these data set cannot be used to aggregate financial KPIs.
    Not suited for real-time use cases.
  billing: $1000 per month
  noticePeriod: P3M
```

Service Levels



```
servicelevels:
  availability:
    description: The server is available during support hours
    percentage: 99.9%
  retention:
    description: Data is retained for one year
    period: P1Y
    unlimited: false
  frequency:
    description: Data is delivered once a day
    type: batch
    cron: 0 0 * * *
  support:
    description: The data is available during typical business hours at headquarters
    time: 9am to 5pm in EST on business days
    responseTime: 1h
  backup:
    description: Data is backed up once a week, every Sunday at 0:00 UTC.
    interval: weekly
    cron: 0 0 * * 0
    recoveryTime: 24 hours
    recoveryPoint: 1 week
```

Servers (Physical Endpoints)

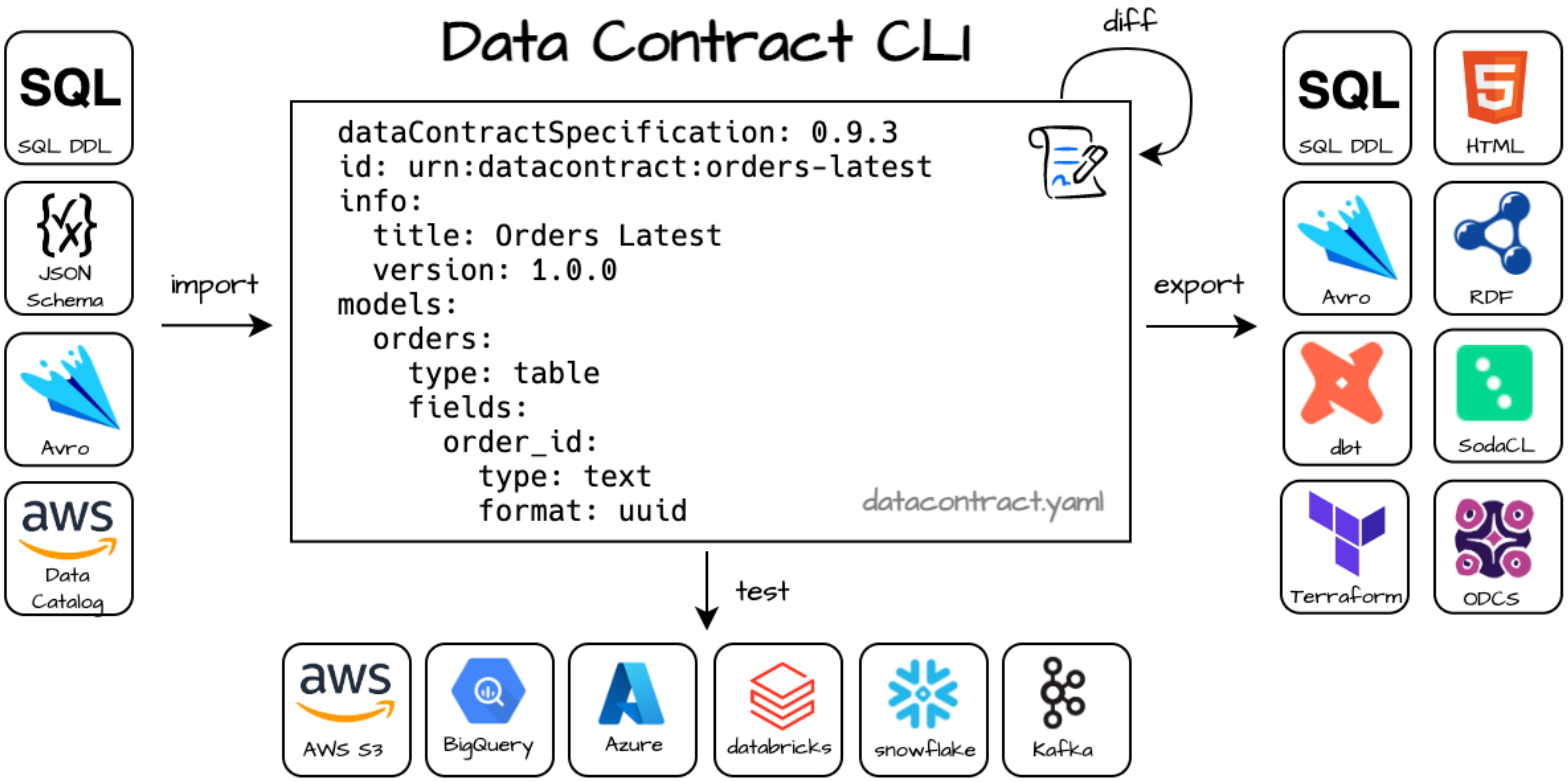


```yaml
servers:
  production:
    type: BigQuery
    project: acme_sales_prod
    dataset: orders_latest_pii_v1
```

Contract Testing

```
$ datacontract test datacontract.yaml
```

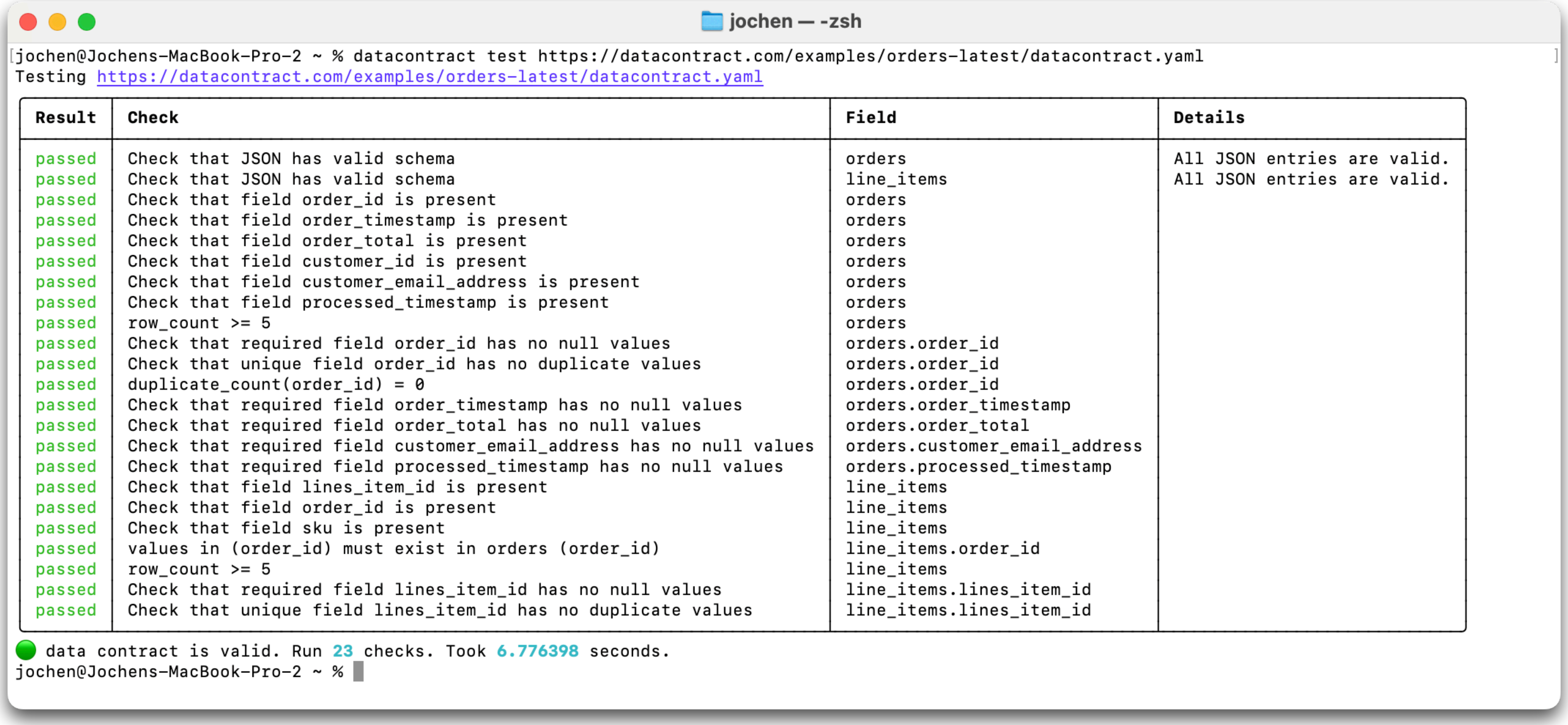
Data Contract CLI



```yaml
dataContractSpecification: 0.9.3
id: urn:datacontract:orders-latest
info:
  title: Orders Latest
  version: 1.0.0
models:
  orders:
    type: table
    fields:
      order_id:
        type: text
        format: uuid
```

github.com/datacontract/cli

Data Contract Testing



```
jochen@Jochens-MacBook-Pro-2 ~ % datacontract test https://datacontract.com/examples/orders-latest/datacontract.yaml
Testing https://datacontract.com/examples/orders-latest/datacontract.yaml
┌────────┬────────────────────────────────────────────────────────────────────┬────────────────────────────────┬─────────────────────────────┐
│ Result │ Check                                                              │ Field                          │ Details                     │
├────────┼────────────────────────────────────────────────────────────────────┼────────────────────────────────┼─────────────────────────────┤
│ passed │ Check that JSON has valid schema                                   │ orders                         │ All JSON entries are valid. │
│ passed │ Check that JSON has valid schema                                   │ line_items                     │ All JSON entries are valid. │
│ passed │ Check that field order_id is present                               │ orders                         │                             │
│ passed │ Check that field order_timestamp is present                        │ orders                         │                             │
│ passed │ Check that field order_total is present                            │ orders                         │                             │
│ passed │ Check that field customer_id is present                            │ orders                         │                             │
│ passed │ Check that field customer_email_address is present                 │ orders                         │                             │
│ passed │ Check that field processed_timestamp is present                    │ orders                         │                             │
│ passed │ row_count >= 5                                                     │ orders                         │                             │
│ passed │ Check that required field order_id has no null values              │ orders.order_id                │                             │
│ passed │ Check that unique field order_id has no duplicate values           │ orders.order_id                │                             │
│ passed │ duplicate_count(order_id) = 0                                      │ orders.order_id                │                             │
│ passed │ Check that required field order_timestamp has no null values       │ orders.order_timestamp         │                             │
│ passed │ Check that required field order_total has no null values           │ orders.order_total             │                             │
│ passed │ Check that required field customer_email_address has no null values│ orders.customer_email_address  │                             │
│ passed │ Check that required field processed_timestamp has no null values   │ orders.processed_timestamp     │                             │
│ passed │ Check that field lines_item_id is present                          │ line_items                     │                             │
│ passed │ Check that field order_id is present                               │ line_items                     │                             │
│ passed │ Check that field sku is present                                    │ line_items                     │                             │
│ passed │ values in (order_id) must exist in orders (order_id)               │ line_items.order_id            │                             │
│ passed │ row_count >= 5                                                     │ line_items                     │                             │
│ passed │ Check that required field lines_item_id has no null values         │ line_items.lines_item_id       │                             │
│ passed │ Check that unique field lines_item_id has no duplicate values      │ line_items.lines_item_id       │                             │
└────────┴────────────────────────────────────────────────────────────────────┴────────────────────────────────┴─────────────────────────────┘
🟢 data contract is valid. Run 23 checks. Took 6.776398 seconds.
jochen@Jochens-MacBook-Pro-2 ~ %
```

← Back to pull request #2
Re-run jobs
Summary
Jobs
Run details
Usage
Workflow file
Search logs

Change column name #11

checkBreakingChanges

checkBreakingChanges

failed 5 days ago in 7s

Set up job 1s

Run actions/checkout@v4 1s

Get CLI 0s

Check backwards compatibility 0s

```
▶ Run ./datacontract breaking --with https://raw.githubusercontent.com/datacontract/cli-examples/main/datacontract.yaml
Found 1 differences between the data contracts!

🔴 Difference 1:
Description:  field 'my_table.my_column' was removed
Type:         field-removed
Severity:     breaking
Level:        field
Model:        my_table
Field:        my_column
Exiting application with error: found breaking differences between the data contracts
Error: Process completed with exit code 1.
```

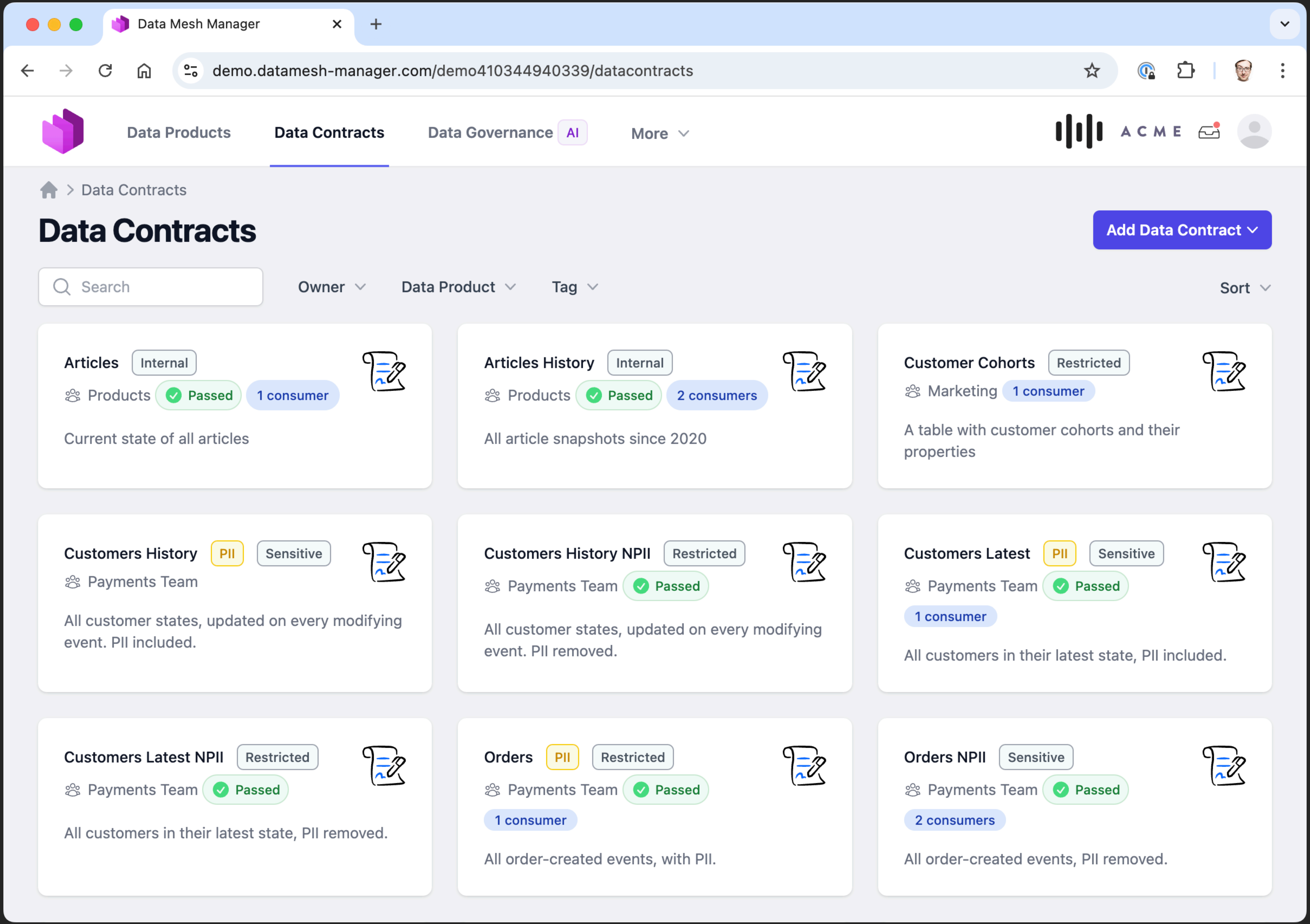
Data Contracts Cover Governance Aspects

- **Ownership** Who is responsible for providing data? **Ownership (Info)**
- **Quality** What quality do we provide? **Quality Attributes & Contract Testing**
- **Compliance** What are we allowed to do with data? **Terms and Conditions**
- **Legal** In which region may data be stored? **Terms and Conditions**
- **Privacy** What is personal (PII) data? **Data Model Attribute**
- **Classification** What sensitivity level has my data? **Data Model Attributes**
- **Security** Who has access to which data and why? --> **Access Requests**

Data Marketplace

Decentralized Approval Process

- Data Marketplace: Central registry for data products
- Self-service Data Access Requests
- Data Product Owners approve Access Requests
- Permissions to actual data are automated by the data platform



Data Contracts

Articles Internal
Products ✓ Passed 1 consumer
Current state of all articles

Articles History Internal
Products ✓ Passed 2 consumers
All article snapshots since 2020

Customer Cohorts Restricted
Marketing 1 consumer
A table with customer cohorts and their properties

Customers History PII Sensitive
Payments Team
All customer states, updated on every modifying event. PII included.

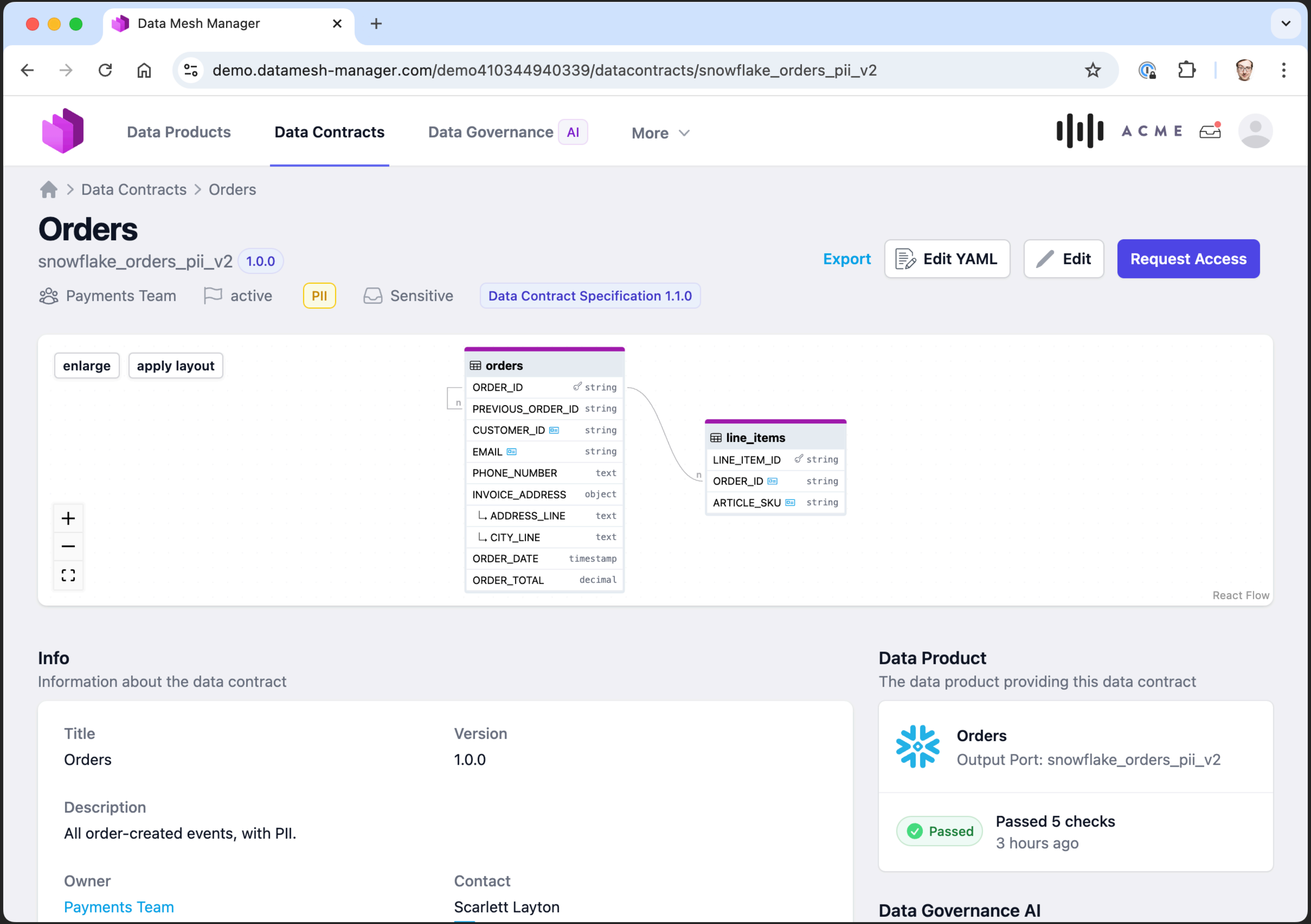
Customers History NPII Restricted
Payments Team ✓ Passed
All customer states, updated on every modifying event. PII removed.

Customers Latest PII Sensitive
Payments Team ✓ Passed 1 consumer
All customers in their latest state, PII included.

Customers Latest NPII Restricted
Payments Team ✓ Passed
All customers in their latest state, PII removed.

Orders PII Restricted
Payments Team ✓ Passed 1 consumer
All order-created events, with PII.

Orders NPII Sensitive
Payments Team ✓ Passed 2 consumers
All order-created events, PII removed.



Data Mesh Manager
demo.datamesh-manager.com/demo410344940339/datacontracts/snowflake_orders_pii_v2
Data Products
Data Contracts
Data Governance AI
More
ACME

Data Contracts > Orders

Orders

snowflake_orders_pii_v2 1.0.0

Payments Team · active · PII · Sensitive · Data Contract Specification 1.1.0

orders

ORDER_ID	string
PREVIOUS_ORDER_ID	string
CUSTOMER_ID	string
EMAIL	string
PHONE_NUMBER	text
INVOICE_ADDRESS	object
└ ADDRESS_LINE	text
└ CITY_LINE	text
ORDER_DATE	timestamp
ORDER_TOTAL	decimal

line_items

LINE_ITEM_ID	string
ORDER_ID	string
ARTICLE_SKU	string

Info

Information about the data contract

Title Orders	Version 1.0.0
Description All order-created events, with PII.	
Owner Payments Team	Contact Scarlett Layton

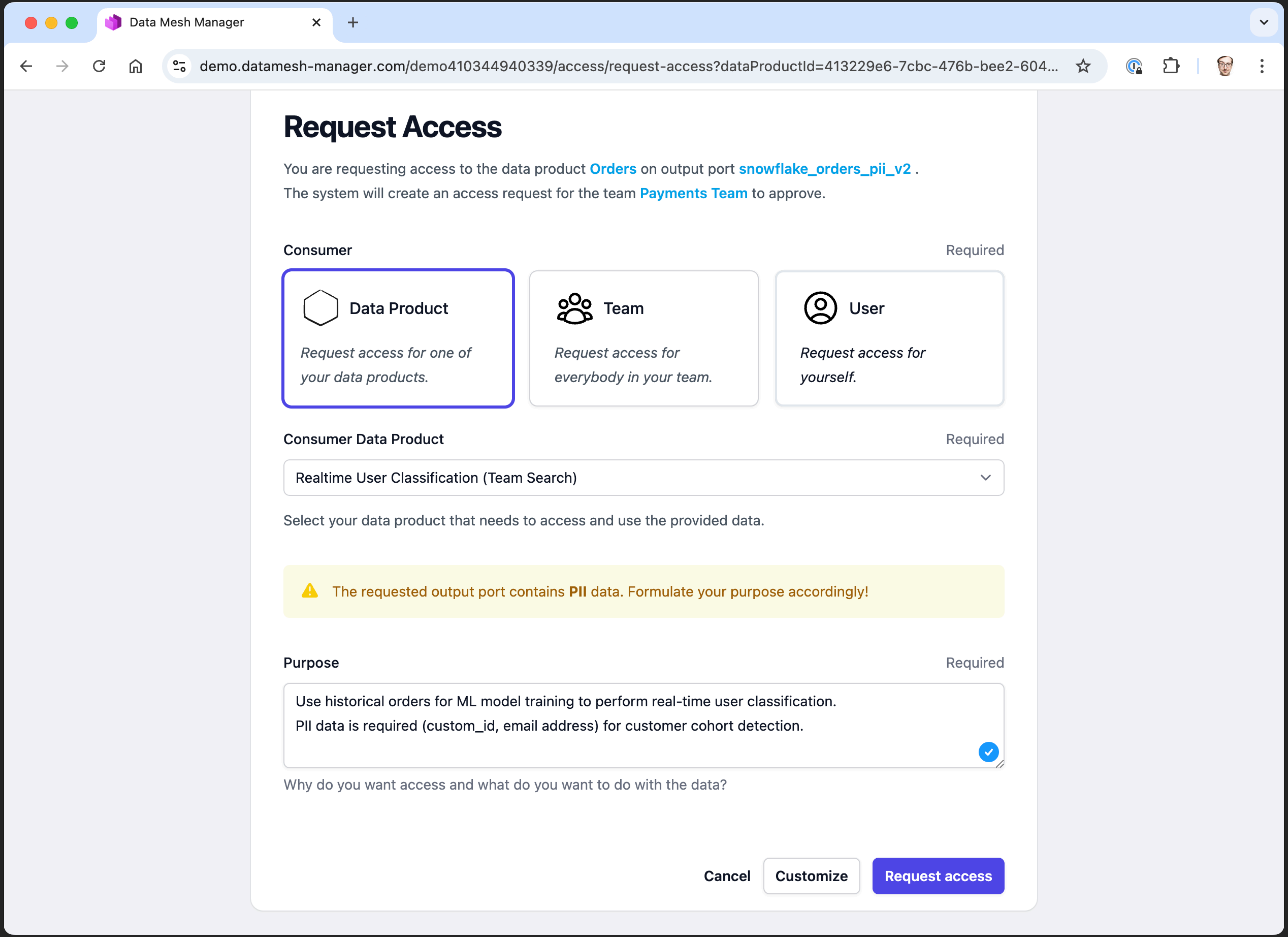
Data Product

The data product providing this data contract

Orders
Output Port: snowflake_orders_pii_v2

Passed — Passed 5 checks
3 hours ago

Data Governance AI


Data Mesh Manager
demo.datamesh-manager.com/demo410344940339/access/request-access?dataProductId=413229e6-7cbc-476b-bee2-604...

Request Access

You are requesting access to the data product **Orders** on output port **snowflake_orders_pii_v2** .
The system will create an access request for the team **Payments Team** to approve.

Consumer Required

● **Data Product** *Request access for one of your data products.*

○ **Team** *Request access for everybody in your team.*

○ **User** *Request access for yourself.*

Consumer Data Product Required

Realtime User Classification (Team Search)

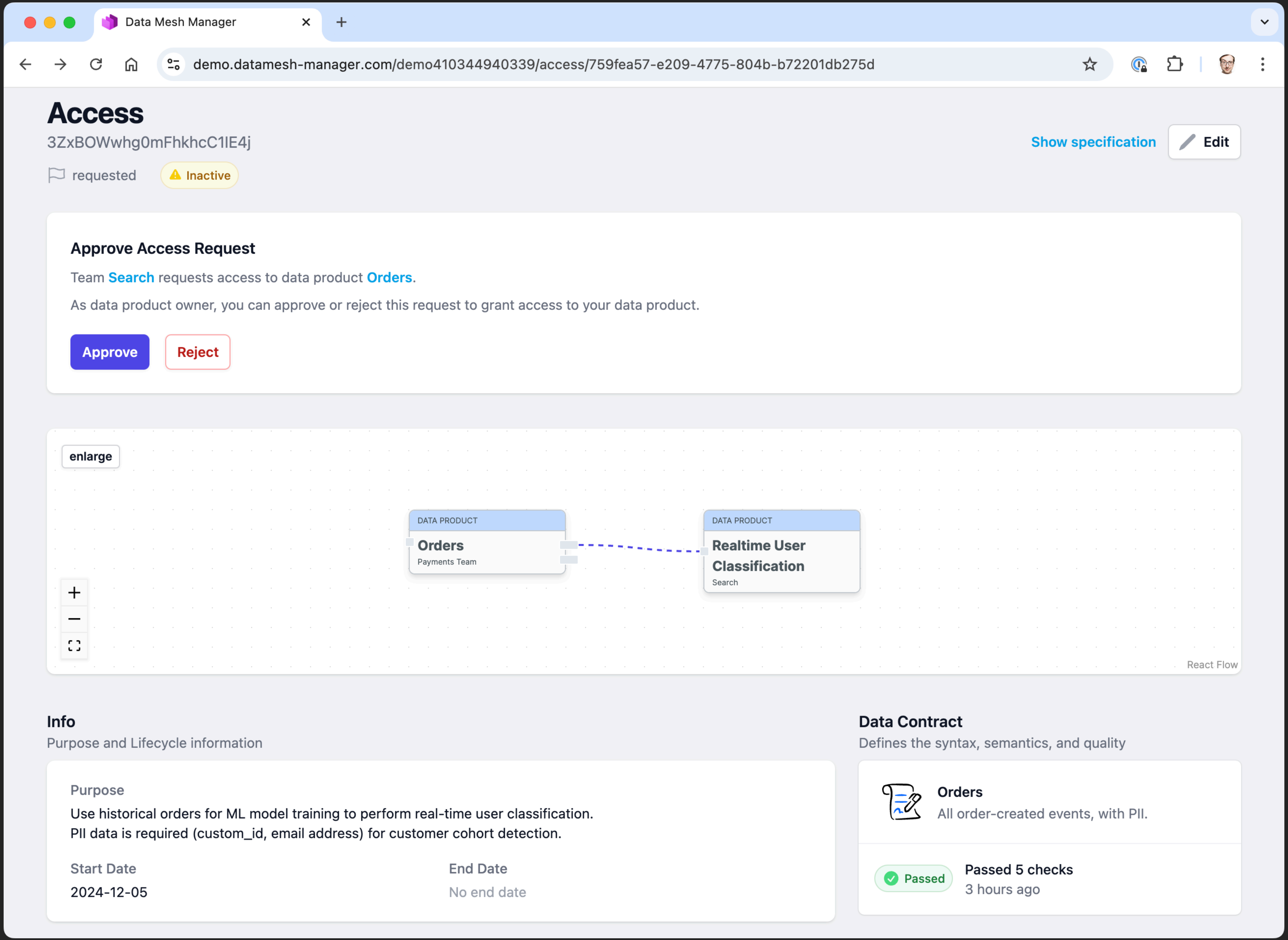
Select your data product that needs to access and use the provided data.

⚠ The requested output port contains **PII** data. Formulate your purpose accordingly!

Purpose Required

Use historical orders for ML model training to perform real-time user classification.
PII data is required (custom_id, email address) for customer cohort detection.

Why do you want access and what do you want to do with the data?


Data Mesh Manager
demo.datamesh-manager.com/demo410344940339/access/759fea57-e209-4775-804b-b72201db275d
Show specification
Edit
enlarge
DATA PRODUCT
Orders
Payments Team
DATA PRODUCT
Realtime User Classification
Search
React Flow

Access

3ZxBOWwhg0mFhkhcC1IE4j

requested ⚠ Inactive

Approve Access Request

Team **Search** requests access to data product **Orders**.

As data product owner, you can approve or reject this request to grant access to your data product.

Approve Reject

Info

Purpose and Lifecycle information

Purpose

Use historical orders for ML model training to perform real-time user classification.
PII data is required (custom_id, email address) for customer cohort detection.

Start Date	End Date
2024-12-05	No end date

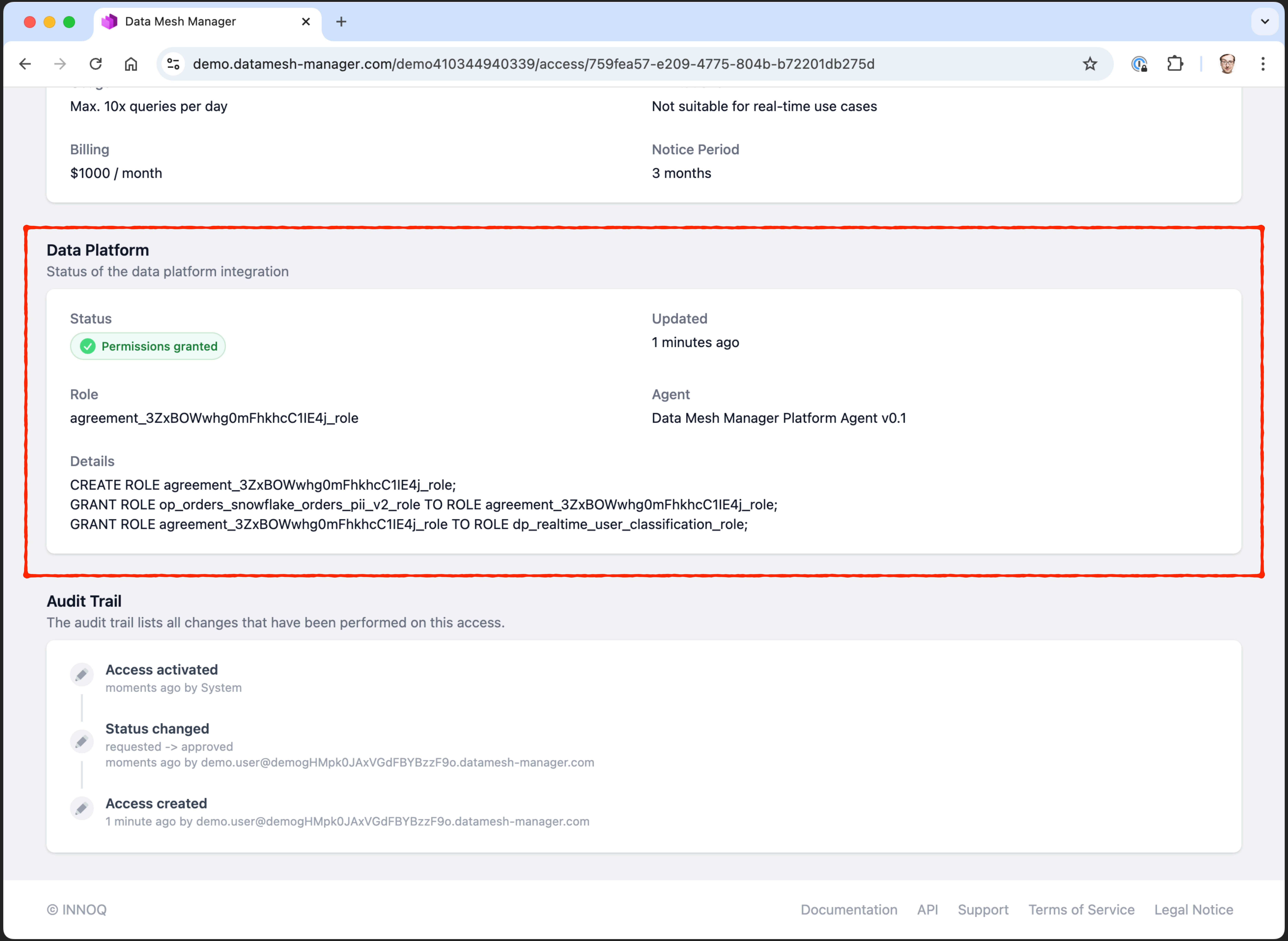
Data Contract

Defines the syntax, semantics, and quality

Orders

All order-created events, with PII.

✓ Passed **Passed 5 checks**
3 hours ago


Data Mesh Manager
demo.datamesh-manager.com/demo410344940339/access/759fea57-e209-4775-804b-b72201db275d

Max. 10x queries per day

Not suitable for real-time use cases

Billing

$1000 / month

Notice Period

3 months

Data Platform

Status of the data platform integration

Status

✓ Permissions granted

Updated

1 minutes ago

Role

agreement_3ZxBOWwhg0mFhkhcC1IE4j_role

Agent

Data Mesh Manager Platform Agent v0.1

Details

```
CREATE ROLE agreement_3ZxBOWwhg0mFhkhcC1IE4j_role;
GRANT ROLE op_orders_snowflake_orders_pii_v2_role TO ROLE agreement_3ZxBOWwhg0mFhkhcC1IE4j_role;
GRANT ROLE agreement_3ZxBOWwhg0mFhkhcC1IE4j_role TO ROLE dp_realtime_user_classification_role;
```

Audit Trail

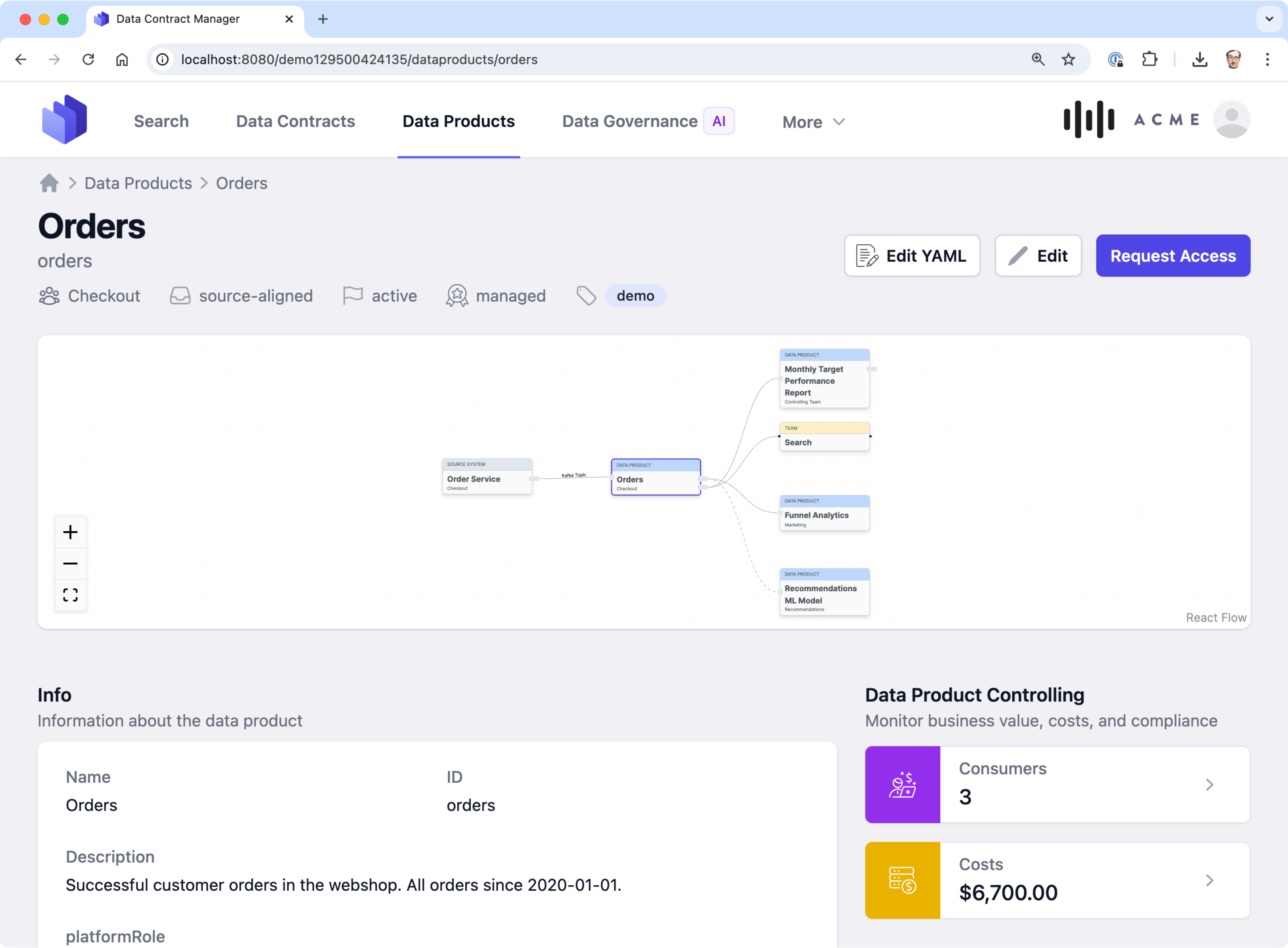
The audit trail lists all changes that have been performed on this access.

Access activated
moments ago by System

Status changed
requested -> approved
moments ago by demo.user@demogHMpk0JAxVGdFBYBzzF9o.datamesh-manager.com

Access created
1 minute ago by demo.user@demogHMpk0JAxVGdFBYBzzF9o.datamesh-manager.com

Know Your Consumers (KYC)



Orders

orders

Info

Information about the data product

Name
Orders

ID
orders

Description
Successful customer orders in the webshop. All orders since 2020-01-01.

platformRole

Data Product Controlling

Monitor business value, costs, and compliance

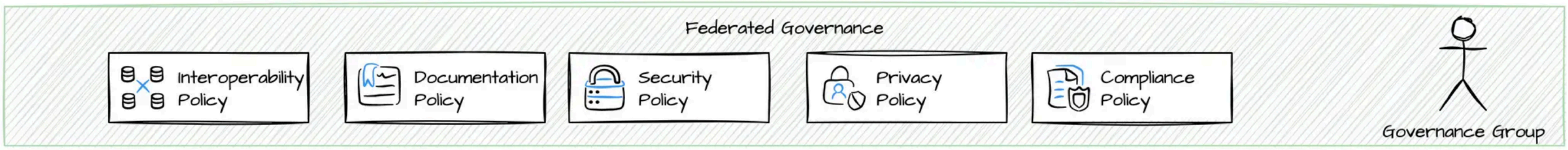
Consumers
3

Costs
$6,700.00

- Data product owners **know their consumers**
- **Access** is fully automated
- **Life-Cycle Management:** Access can be cancelled by consumer or providers
- Important for data product **evolution** (e.g. breaking changes)

Global Policies

Global Policies



Federated Governance

Interoperability Policy

Documentation Policy

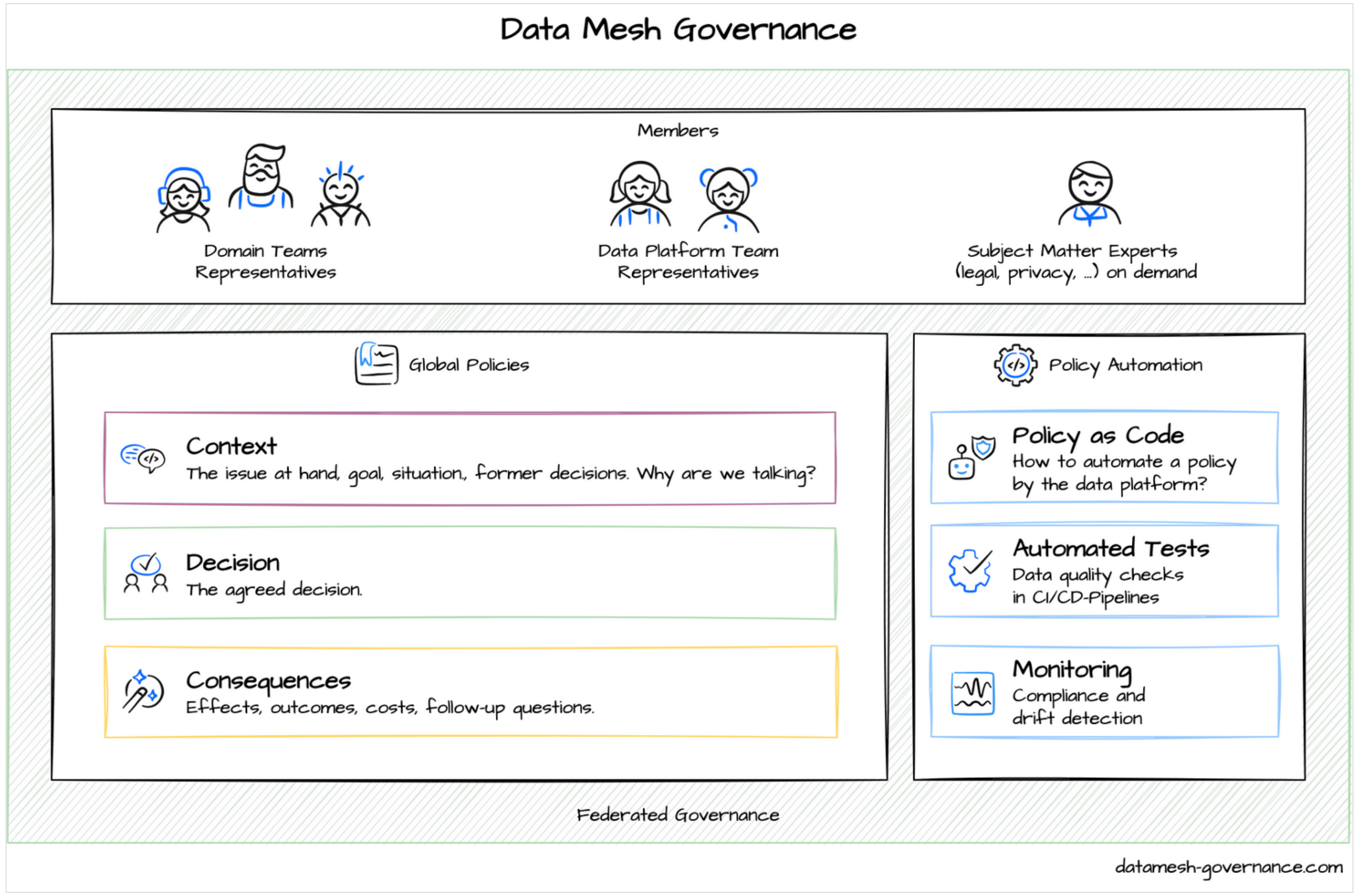
Security Policy

Privacy Policy

Compliance Policy

Governance Group

- Agree on common standards for data products and data contracts
- Deciders: Data Product Owners (supported by SME)
- Responsibility: Data Product Owners
- Automated by: Data Platform



Data Mesh Governance

Members

Domain Teams Representatives

Data Platform Team Representatives

Subject Matter Experts (legal, privacy, ...) on demand

Global Policies

Context
The issue at hand, goal, situation, former decisions. Why are we talking?

Decision
The agreed decision.

Consequences
Effects, outcomes, costs, follow-up questions.

Policy Automation

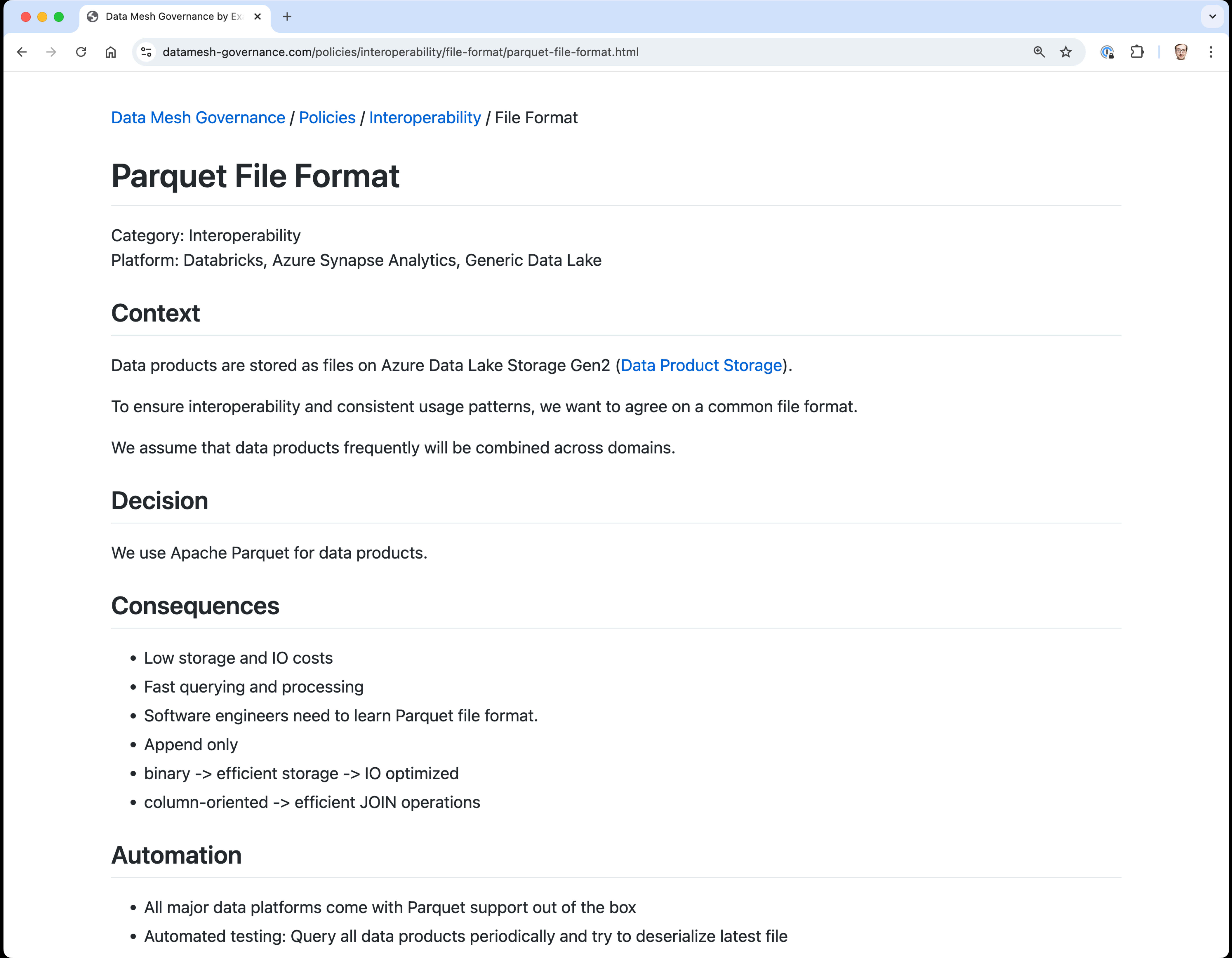
Policy as Code
How to automate a policy by the data platform?

Automated Tests
Data quality checks in CI/CD-Pipelines

Monitoring
Compliance and drift detection

Federated Governance

datamesh-governance.com


Data Mesh Governance by Ex
datamesh-governance.com/policies/interoperability/file-format/parquet-file-format.html

Data Mesh Governance / Policies / Interoperability / File Format

Parquet File Format

Category: Interoperability
Platform: Databricks, Azure Synapse Analytics, Generic Data Lake

Context

Data products are stored as files on Azure Data Lake Storage Gen2 (Data Product Storage).

To ensure interoperability and consistent usage patterns, we want to agree on a common file format.

We assume that data products frequently will be combined across domains.

Decision

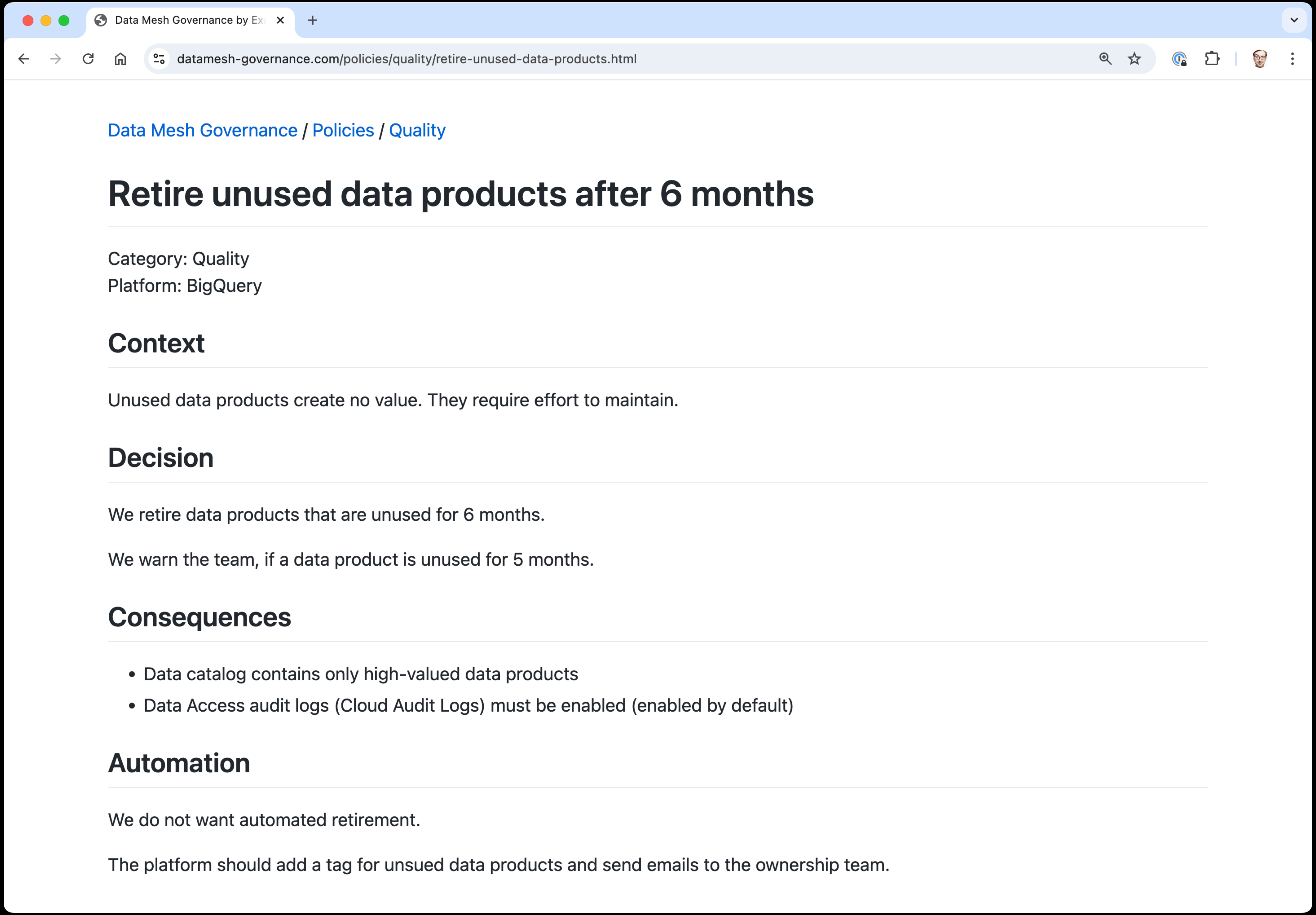
We use Apache Parquet for data products.

Consequences

- Low storage and IO costs
- Fast querying and processing
- Software engineers need to learn Parquet file format.
- Append only
- binary -> efficient storage -> IO optimized
- column-oriented -> efficient JOIN operations

Automation

- All major data platforms come with Parquet support out of the box
- Automated testing: Query all data products periodically and try to deserialize latest file


Data Mesh Governance by Ex
datamesh-governance.com/policies/quality/retire-unused-data-products.html

Data Mesh Governance / Policies / Quality

Retire unused data products after 6 months

Category: Quality
Platform: BigQuery

Context

Unused data products create no value. They require effort to maintain.

Decision

We retire data products that are unused for 6 months.

We warn the team, if a data product is unused for 5 months.

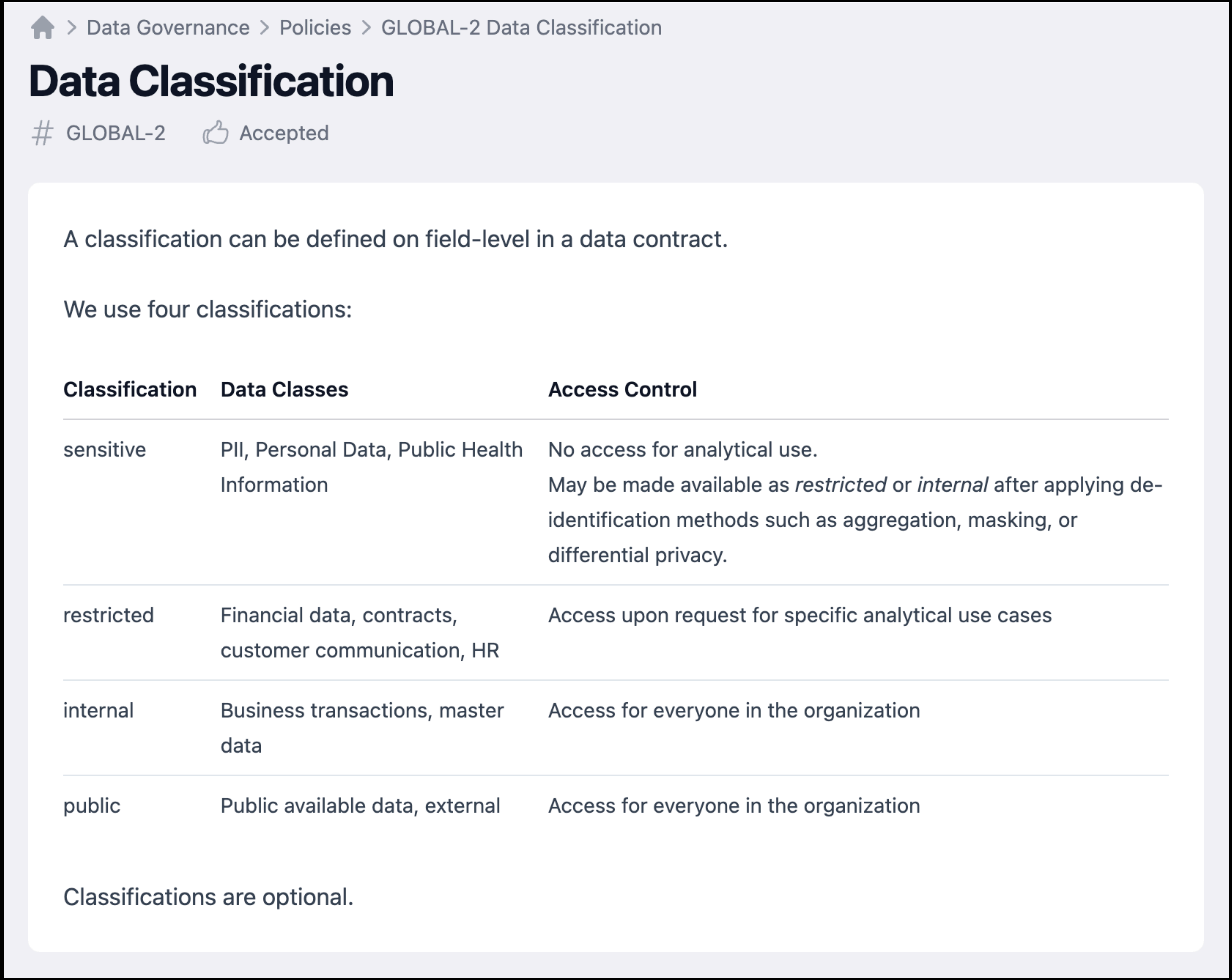
Consequences

- Data catalog contains only high-valued data products
- Data Access audit logs (Cloud Audit Logs) must be enabled (enabled by default)

Automation

We do not want automated retirement.

The platform should add a tag for unsued data products and send emails to the ownership team.



⌂ > Data Governance > Policies > GLOBAL-2 Data Classification

Data Classification

GLOBAL-2

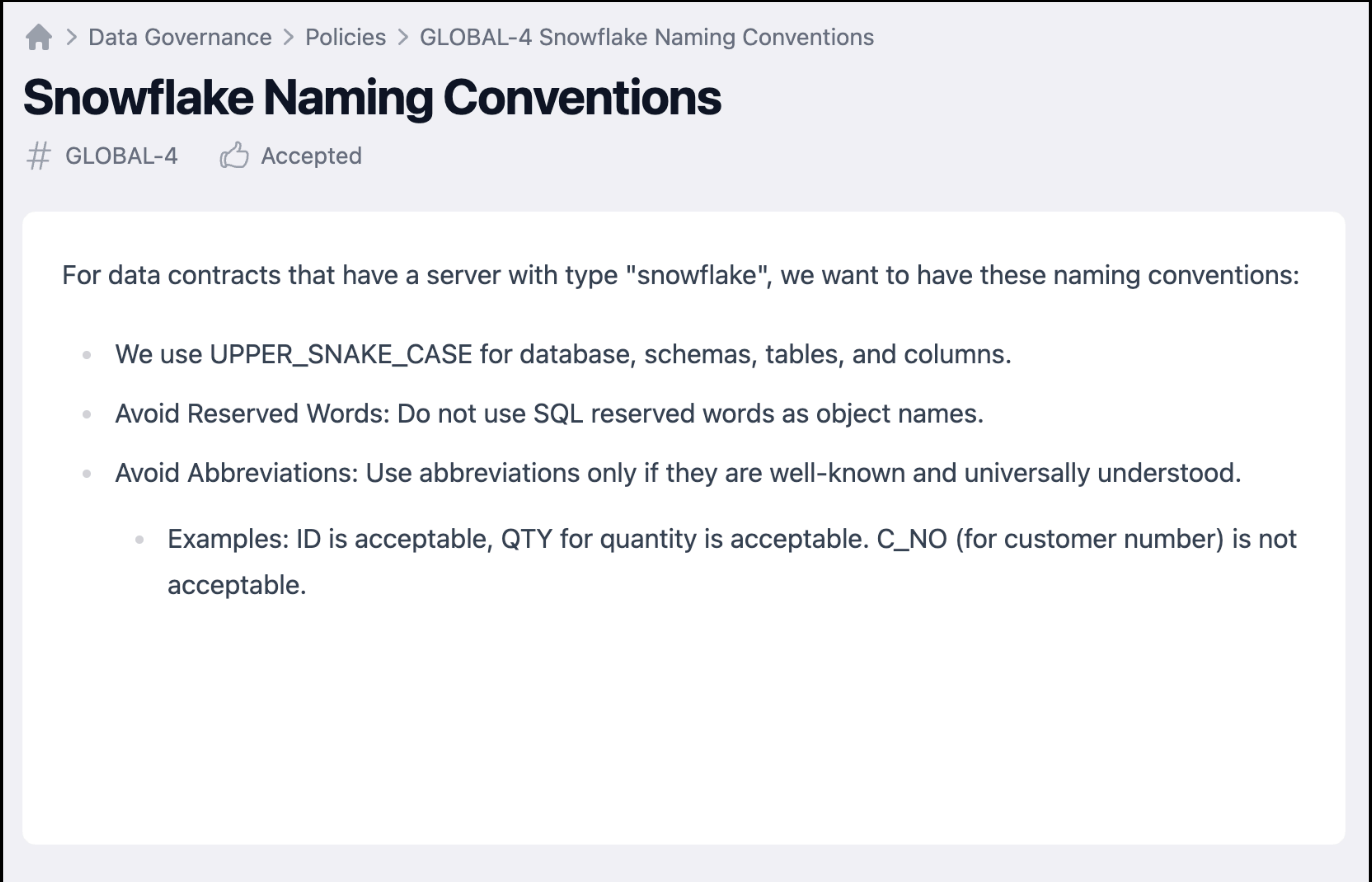
Accepted

A classification can be defined on field-level in a data contract.

We use four classifications:

Classification	Data Classes	Access Control
sensitive	PII, Personal Data, Public Health Information	No access for analytical use. May be made available as *restricted* or *internal* after applying de-identification methods such as aggregation, masking, or differential privacy.
restricted	Financial data, contracts, customer communication, HR	Access upon request for specific analytical use cases
internal	Business transactions, master data	Access for everyone in the organization
public	Public available data, external	Access for everyone in the organization

Classifications are optional.



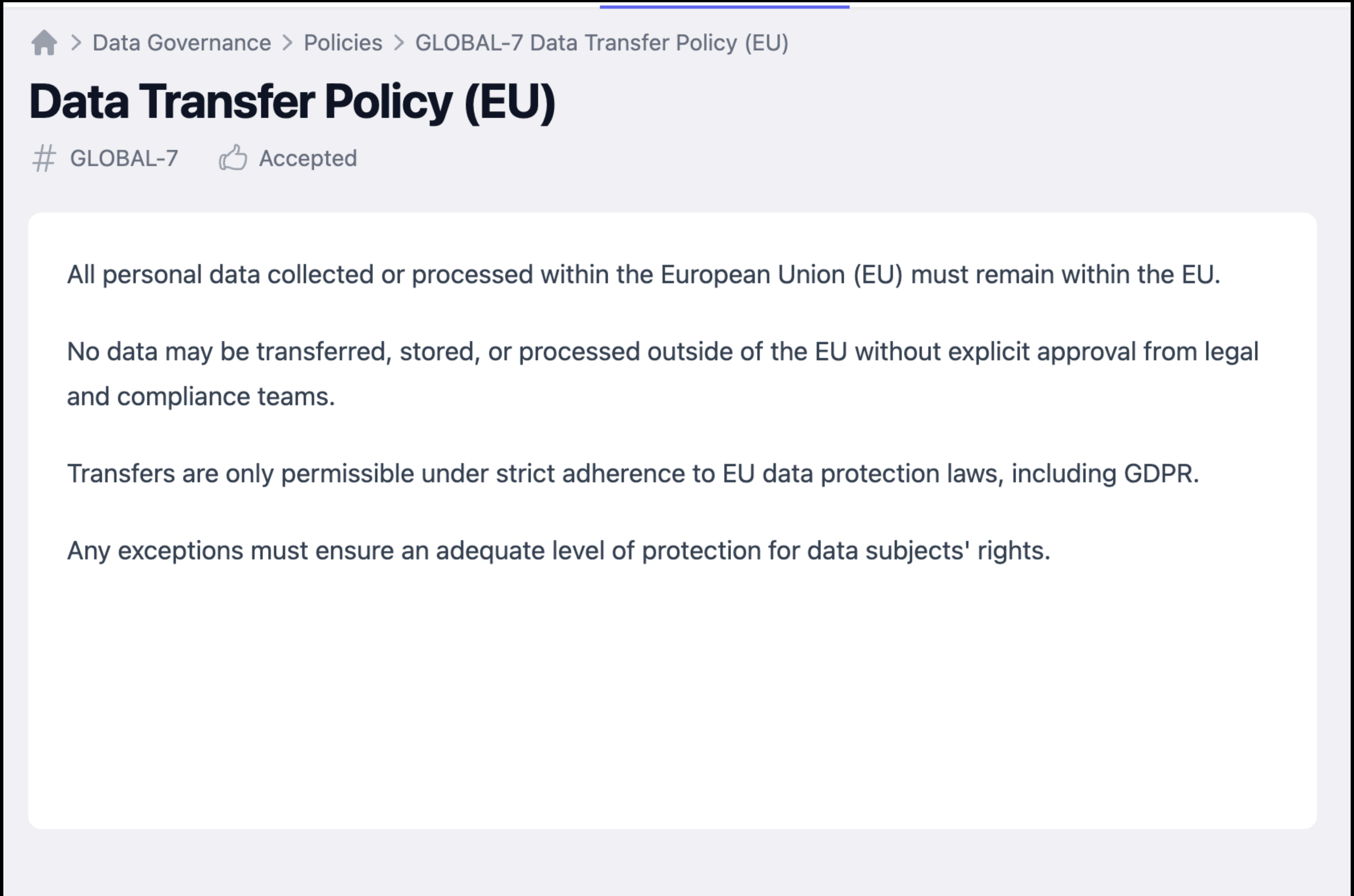
⌂ > Data Governance > Policies > GLOBAL-4 Snowflake Naming Conventions

Snowflake Naming Conventions

GLOBAL-4 👍 Accepted

For data contracts that have a server with type "snowflake", we want to have these naming conventions:

- We use UPPER_SNAKE_CASE for database, schemas, tables, and columns.
- Avoid Reserved Words: Do not use SQL reserved words as object names.
- Avoid Abbreviations: Use abbreviations only if they are well-known and universally understood.
  - Examples: ID is acceptable, QTY for quantity is acceptable. C_NO (for customer number) is not acceptable.



⌂ > Data Governance > Policies > GLOBAL-7 Data Transfer Policy (EU)

Data Transfer Policy (EU)

GLOBAL-7 Accepted

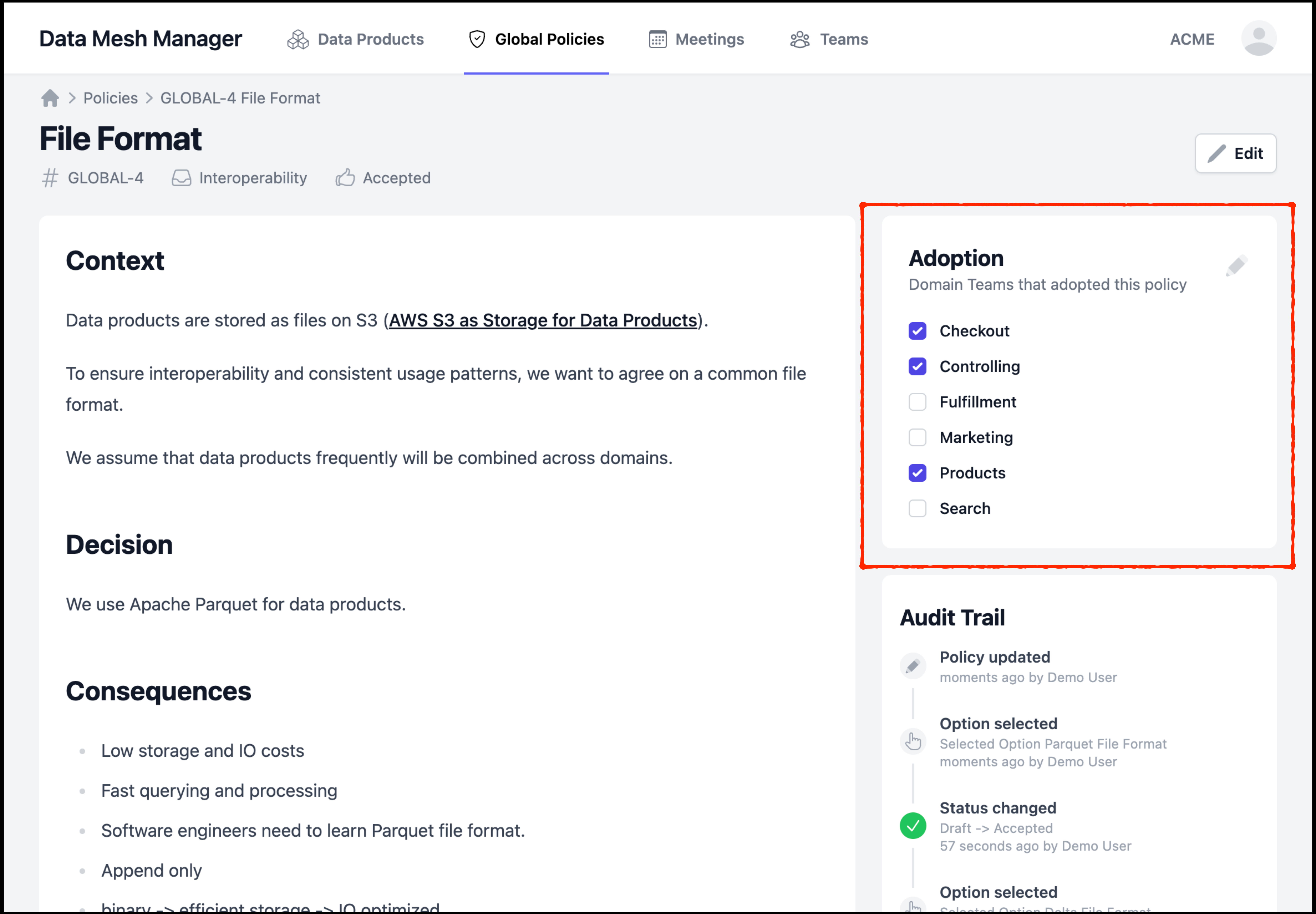
All personal data collected or processed within the European Union (EU) must remain within the EU.

No data may be transferred, stored, or processed outside of the EU without explicit approval from legal and compliance teams.

Transfers are only permissible under strict adherence to EU data protection laws, including GDPR.

Any exceptions must ensure an adequate level of protection for data subjects' rights.

Data Governance



File Format

GLOBAL-4   Interoperability   Accepted

Context

Data products are stored as files on S3 (AWS S3 as Storage for Data Products).

To ensure interoperability and consistent usage patterns, we want to agree on a common file format.

We assume that data products frequently will be combined across domains.

Decision

We use Apache Parquet for data products.

Consequences

- Low storage and IO costs
- Fast querying and processing
- Software engineers need to learn Parquet file format.
- Append only
- binary -> efficient storage -> IO optimized

Adoption

Domain Teams that adopted this policy

☑ Checkout
☑ Controlling
□ Fulfillment
□ Marketing
☑ Products
□ Search

Audit Trail

Policy updated
moments ago by Demo User

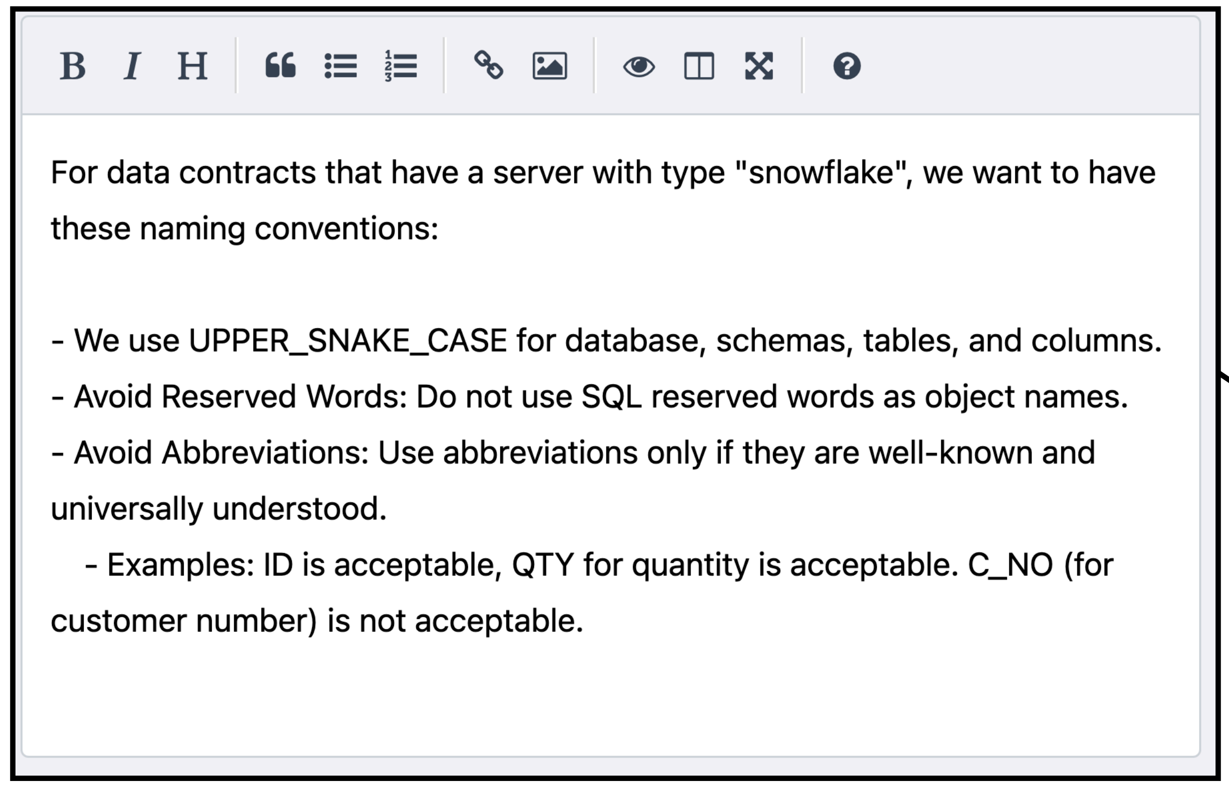
Option selected
Selected Option Parquet File Format
moments ago by Demo User

Status changed
Draft -> Accepted
57 seconds ago by Demo User

Option selected
Selected Option Delta File Format

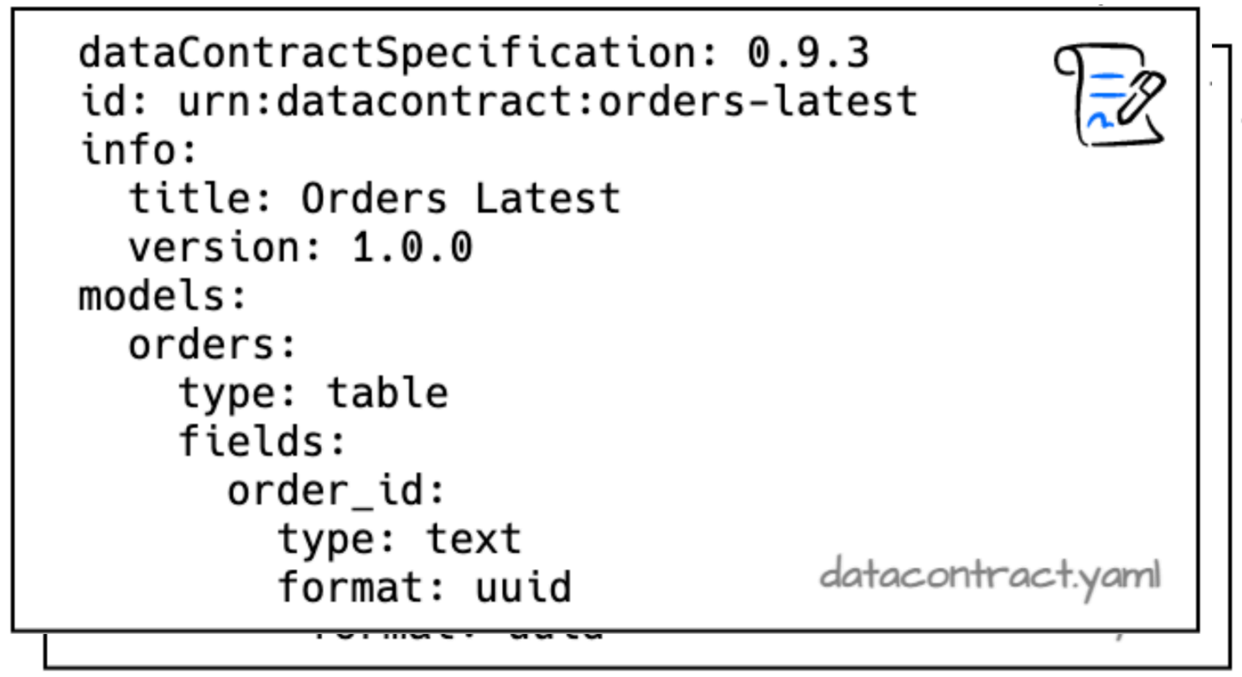
Data Governance AI

Policy (Markdown)


For data contracts that have a server with type "snowflake", we want to have these naming conventions:

- We use UPPER_SNAKE_CASE for database, schemas, tables, and columns.
- Avoid Reserved Words: Do not use SQL reserved words as object names.
- Avoid Abbreviations: Use abbreviations only if they are well-known and universally understood.
   - Examples: ID is acceptable, QTY for quantity is acceptable. C_NO (for customer number) is not acceptable.

Metadata (YAML)

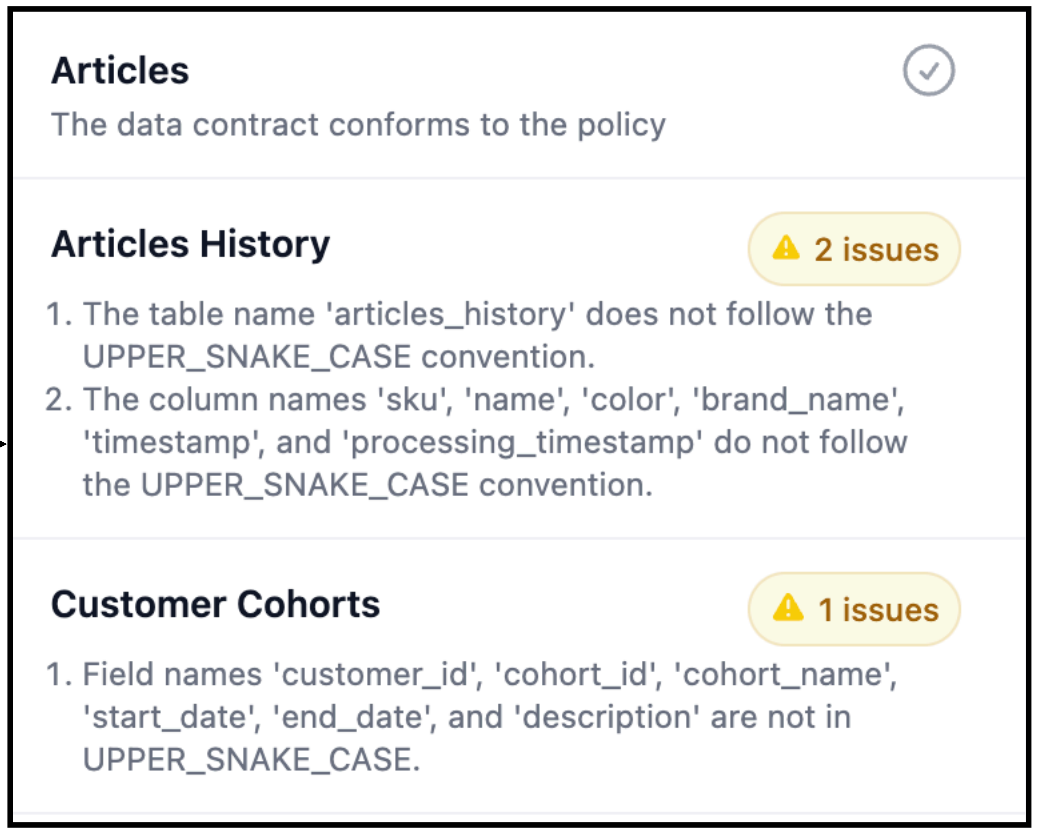


```
dataContractSpecification: 0.9.3
id: urn:datacontract:orders-latest
info:
  title: Orders Latest
  version: 1.0.0
models:
  orders:
    type: table
    fields:
      order_id:
        type: text
        format: uuid
```

datacontract.yaml

LLM

Result (JSON / HTML)


Articles

The data contract conforms to the policy

Articles History ⚠ 2 issues

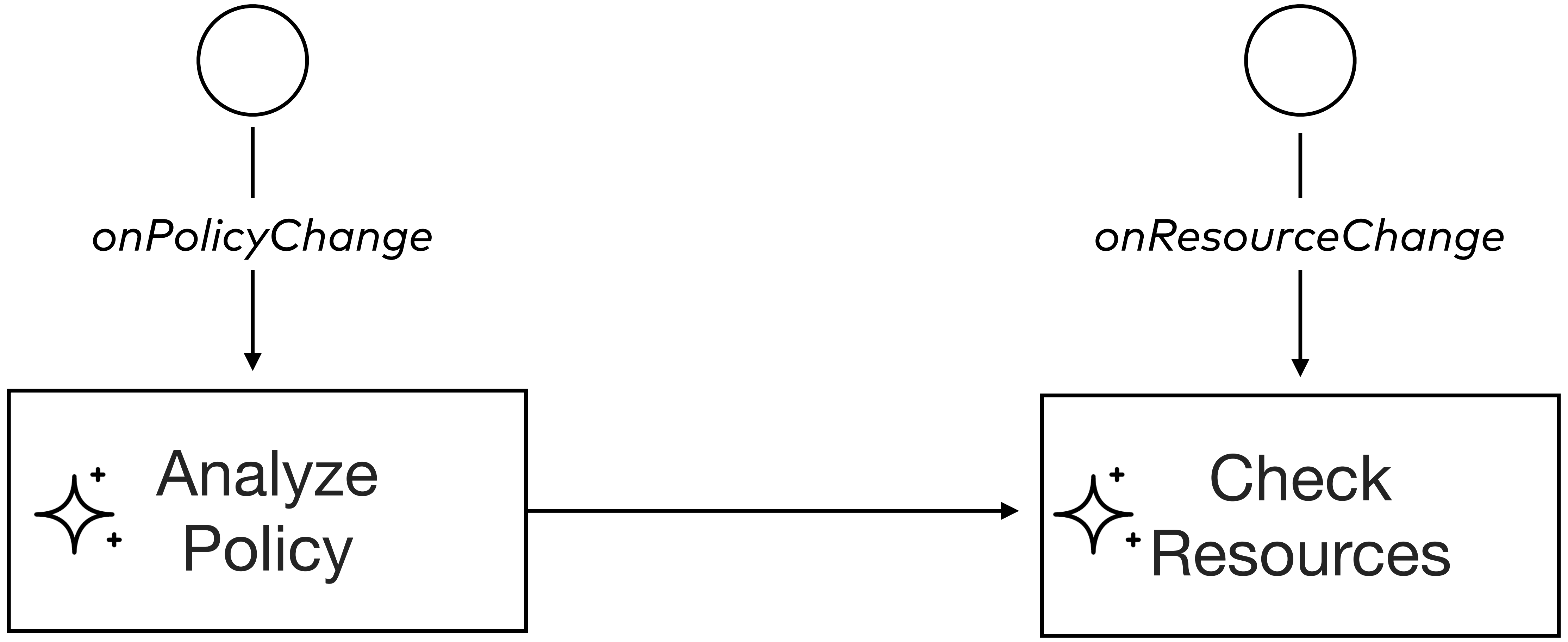
1. The table name 'articles_history' does not follow the UPPER_SNAKE_CASE convention.
2. The column names 'sku', 'name', 'color', 'brand_name', 'timestamp', and 'processing_timestamp' do not follow the UPPER_SNAKE_CASE convention.

Customer Cohorts ⚠ 1 issues

1. Field names 'customer_id', 'cohort_id', 'cohort_name', 'start_date', 'end_date', and 'description' are not in UPPER_SNAKE_CASE.

Demo

Data Governance AI

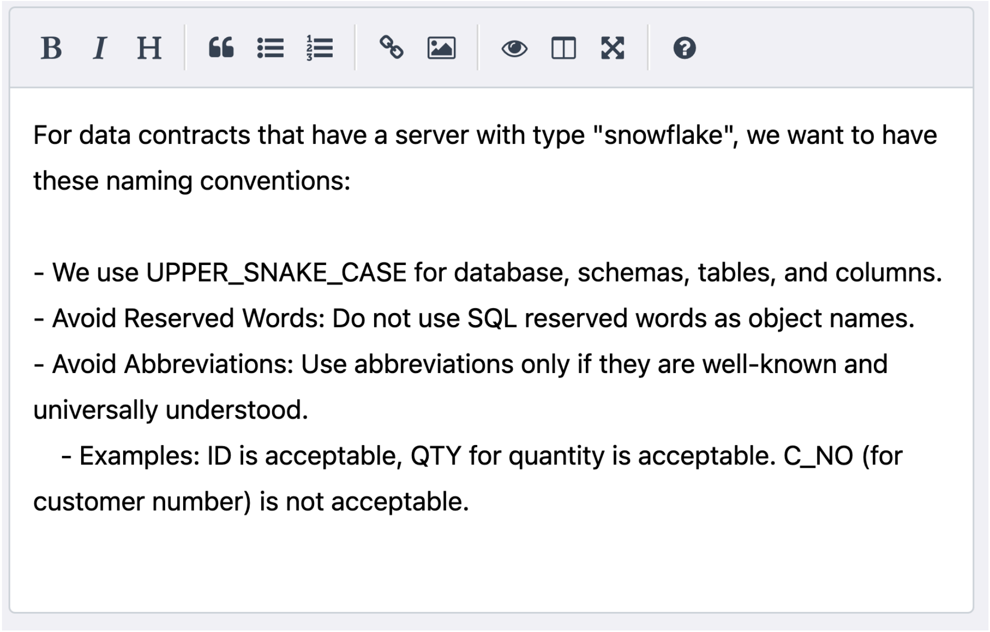


onPolicyChange

onResourceChange

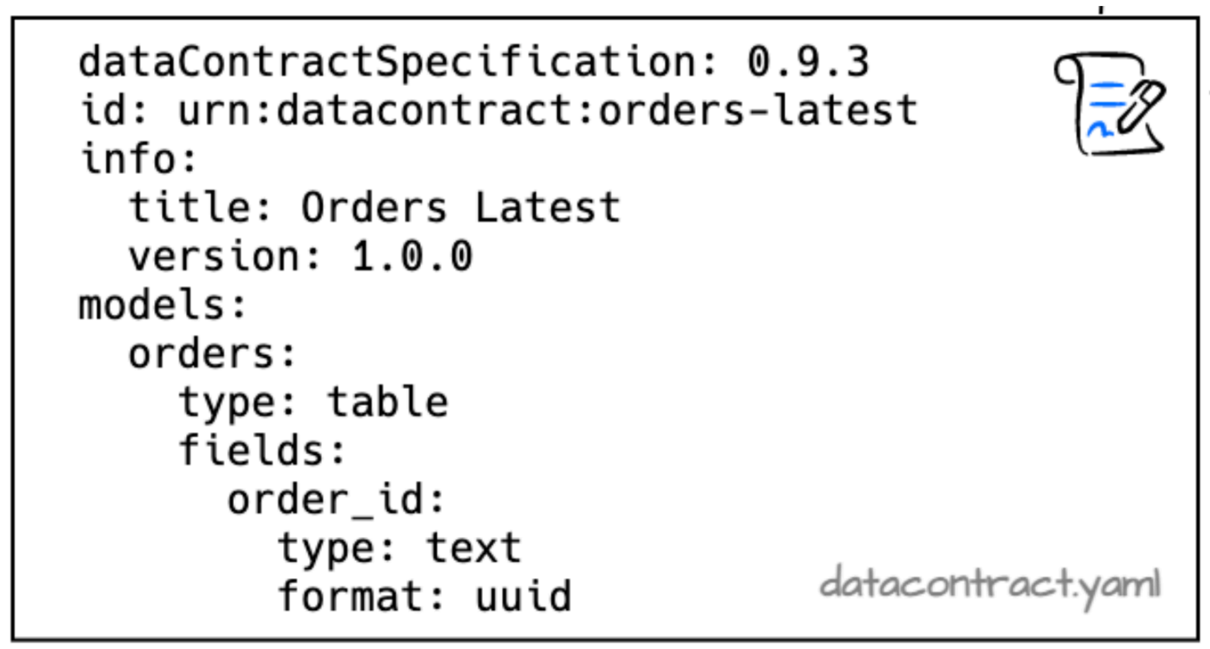
Analyze Policy

Check Resources



For data contracts that have a server with type "snowflake", we want to have these naming conventions:

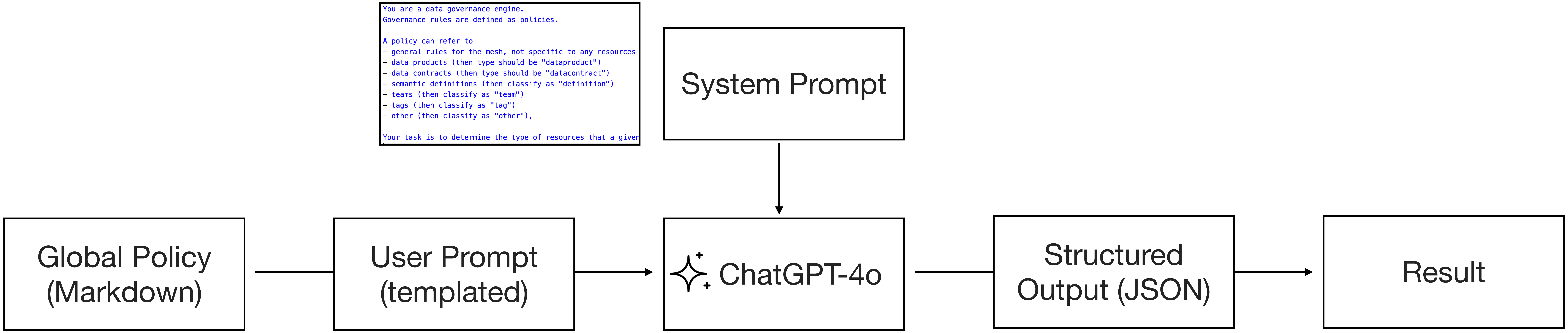
- We use UPPER_SNAKE_CASE for database, schemas, tables, and columns.
- Avoid Reserved Words: Do not use SQL reserved words as object names.
- Avoid Abbreviations: Use abbreviations only if they are well-known and universally understood.
    - Examples: ID is acceptable, QTY for quantity is acceptable. C_NO (for customer number) is not acceptable.



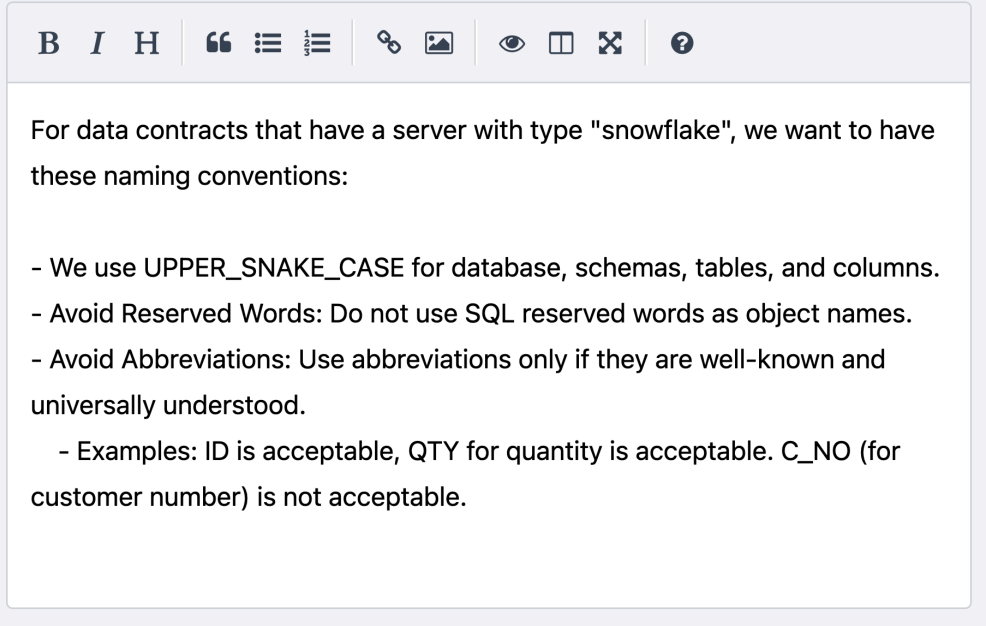
```
dataContractSpecification: 0.9.3
id: urn:datacontract:orders-latest
info:
  title: Orders Latest
  version: 1.0.0
models:
  orders:
    type: table
    fields:
      order_id:
        type: text
        format: uuid
```

datacontract.yaml

Step 1: Analyze Policy

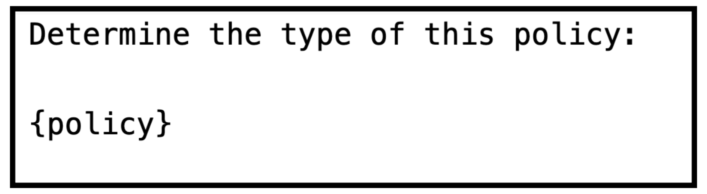


```
You are a data governance engine.
Governance rules are defined as policies.

A policy can refer to
- general rules for the mesh, not specific to any resources
- data products (then type should be "dataproduct")
- data contracts (then type should be "datacontract")
- semantic definitions (then classify as "definition")
- teams (then classify as "team")
- tags (then classify as "tag")
- other (then classify as "other"),

Your task is to determine the type of resources that a given
```

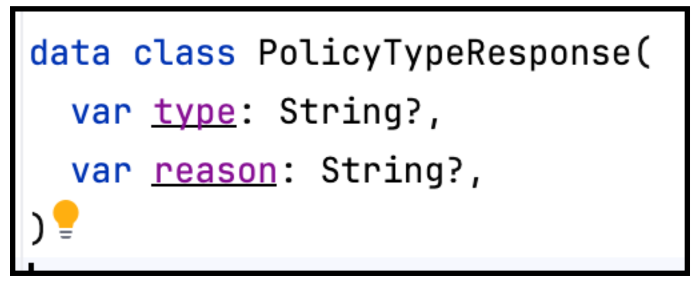


For data contracts that have a server with type "snowflake", we want to have these naming conventions:

- We use UPPER_SNAKE_CASE for database, schemas, tables, and columns.
- Avoid Reserved Words: Do not use SQL reserved words as object names.
- Avoid Abbreviations: Use abbreviations only if they are well-known and universally understood.
    - Examples: ID is acceptable, QTY for quantity is acceptable. C_NO (for customer number) is not acceptable.



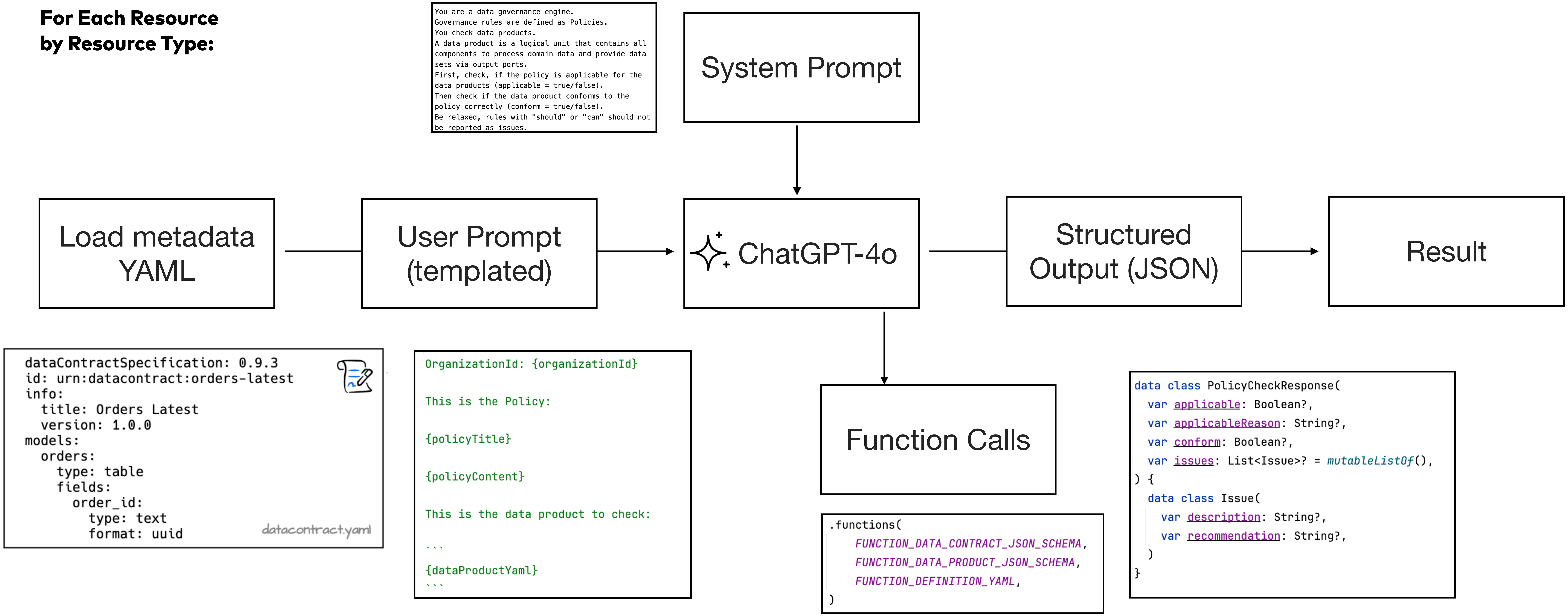
```
Determine the type of this policy:

{policy}
```



```kotlin
data class PolicyTypeResponse(
  var type: String?,
  var reason: String?,
)
```

Step 2: Check Metadata

For Each Resource by Resource Type:



```
You are a data governance engine.
Governance rules are defined as Policies.
You check data products.
A data product is a logical unit that contains all
components to process domain data and provide data
sets via output ports.
First, check, if the policy is applicable for the
data products (applicable = true/false).
Then check if the data product conforms to the
policy correctly (conform = true/false).
Be relaxed, rules with "should" or "can" should not
be reported as issues.
```

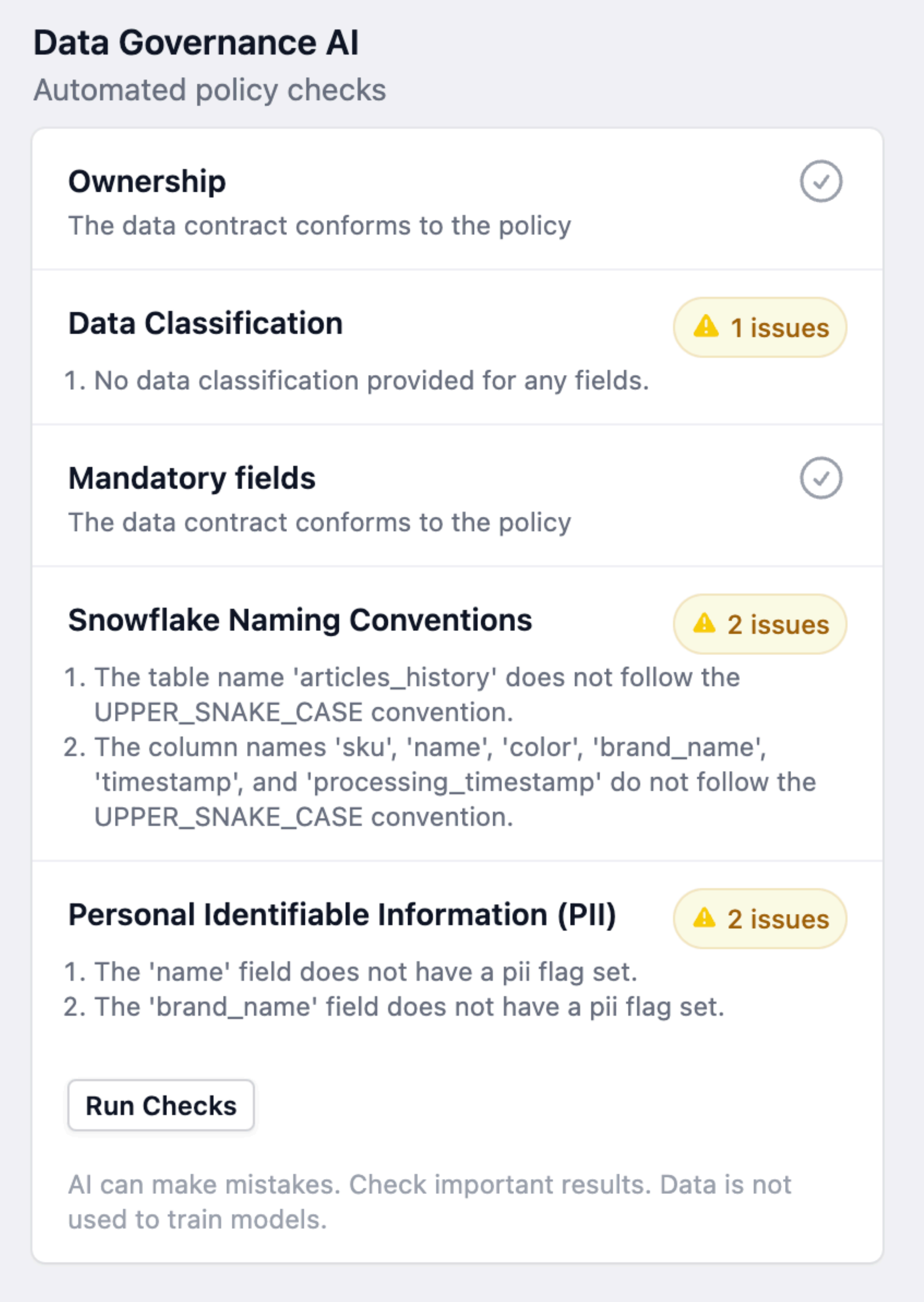
```
dataContractSpecification: 0.9.3
id: urn:datacontract:orders-latest
info:
  title: Orders Latest
  version: 1.0.0
models:
  orders:
    type: table
    fields:
      order_id:
        type: text
        format: uuid
```

```
OrganizationId: {organizationId}

This is the Policy:

{policyTitle}

{policyContent}

This is the data product to check:

```
{dataProductYaml}
```
```

```
.functions(
    FUNCTION_DATA_CONTRACT_JSON_SCHEMA,
    FUNCTION_DATA_PRODUCT_JSON_SCHEMA,
    FUNCTION_DEFINITION_YAML,
)
```

```
data class PolicyCheckResponse(
  var applicable: Boolean?,
  var applicableReason: String?,
  var conform: Boolean?,
  var issues: List<Issue>? = mutableListOf(),
) {
  data class Issue(
    var description: String?,
    var recommendation: String?,
  )
}
```

UI



Data Governance AI

Automated policy checks

Ownership

The data contract conforms to the policy

Data Classification ⚠ 1 issues

1. No data classification provided for any fields.

Mandatory fields

The data contract conforms to the policy

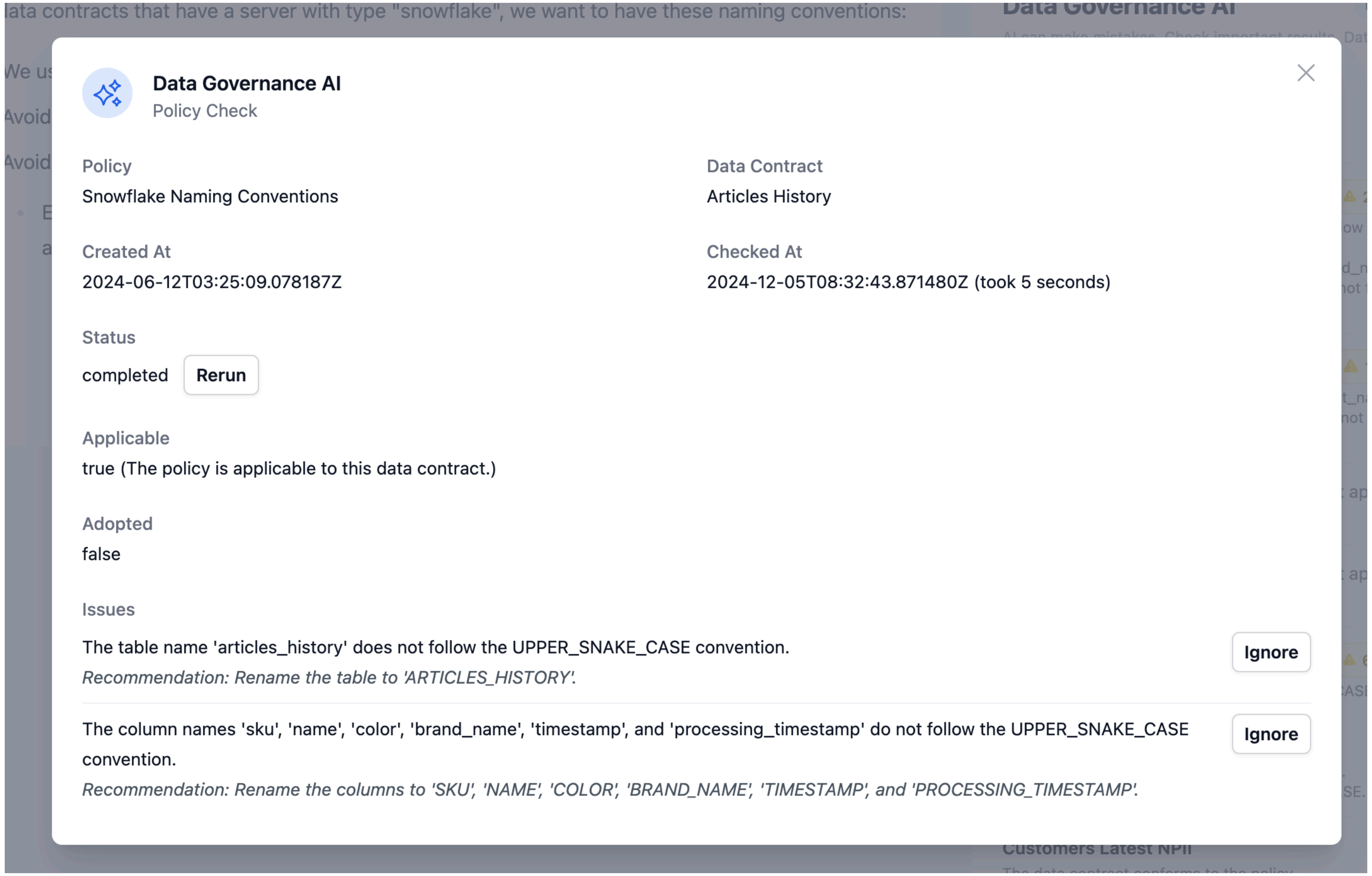
Snowflake Naming Conventions ⚠ 2 issues

1. The table name 'articles_history' does not follow the UPPER_SNAKE_CASE convention.
2. The column names 'sku', 'name', 'color', 'brand_name', 'timestamp', and 'processing_timestamp' do not follow the UPPER_SNAKE_CASE convention.

Personal Identifiable Information (PII) ⚠ 2 issues

1. The 'name' field does not have a pii flag set.
2. The 'brand_name' field does not have a pii flag set.

AI can make mistakes. Check important results. Data is not used to train models.



ata contracts that have a server with type "snowflake", we want to have these naming conventions:

Data Governance AI

Policy Check

Policy	Data Contract
Snowflake Naming Conventions	Articles History

Created At	Checked At
2024-06-12T03:25:09.078187Z	2024-12-05T08:32:43.871480Z (took 5 seconds)

Status

completed

Applicable

true (The policy is applicable to this data contract.)

Adopted

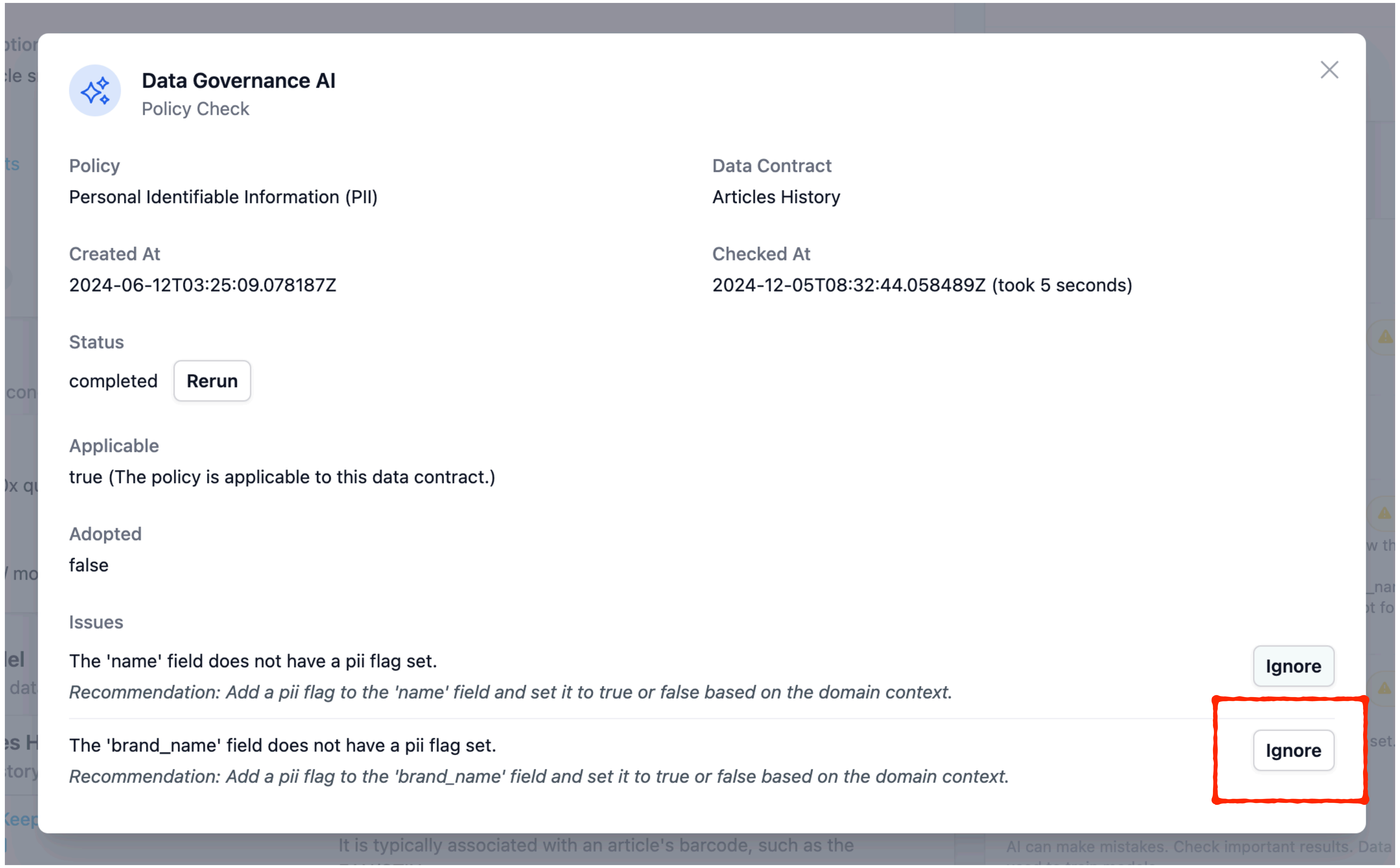
false

Issues

The table name 'articles_history' does not follow the UPPER_SNAKE_CASE convention.
Recommendation: Rename the table to 'ARTICLES_HISTORY'.

The column names 'sku', 'name', 'color', 'brand_name', 'timestamp', and 'processing_timestamp' do not follow the UPPER_SNAKE_CASE convention.
Recommendation: Rename the columns to 'SKU', 'NAME', 'COLOR', 'BRAND_NAME', 'TIMESTAMP', and 'PROCESSING_TIMESTAMP'.

User Feedback!



Data Governance AI

Policy Check

Policy

Personal Identifiable Information (PII)

Data Contract

Articles History

Created At

2024-06-12T03:25:09.078187Z

Checked At

2024-12-05T08:32:44.058489Z (took 5 seconds)

Status

completed

Applicable

true (The policy is applicable to this data contract.)

Adopted

false

Issues

The 'name' field does not have a pii flag set.

Recommendation: Add a pii flag to the 'name' field and set it to true or false based on the domain context.

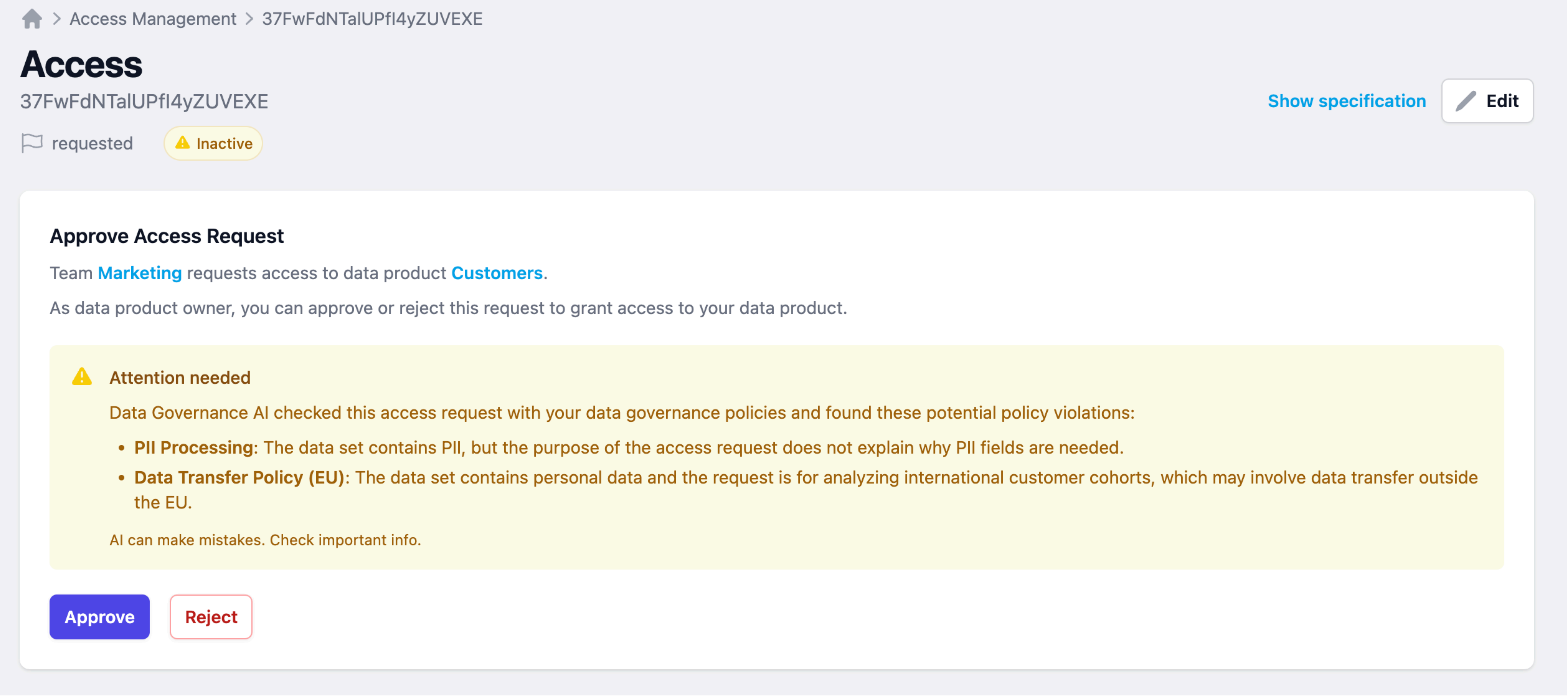
The 'brand_name' field does not have a pii flag set.

Recommendation: Add a pii flag to the 'brand_name' field and set it to true or false based on the domain context.

It is typically associated with an article's barcode, such as the

AI can make mistakes. Check important results. Data

Support Owners in Access Approval Process



Access

37FwFdNTalUPfl4yZUVEXE

requested ⚠ Inactive

Approve Access Request

Team **Marketing** requests access to data product **Customers**.

As data product owner, you can approve or reject this request to grant access to your data product.

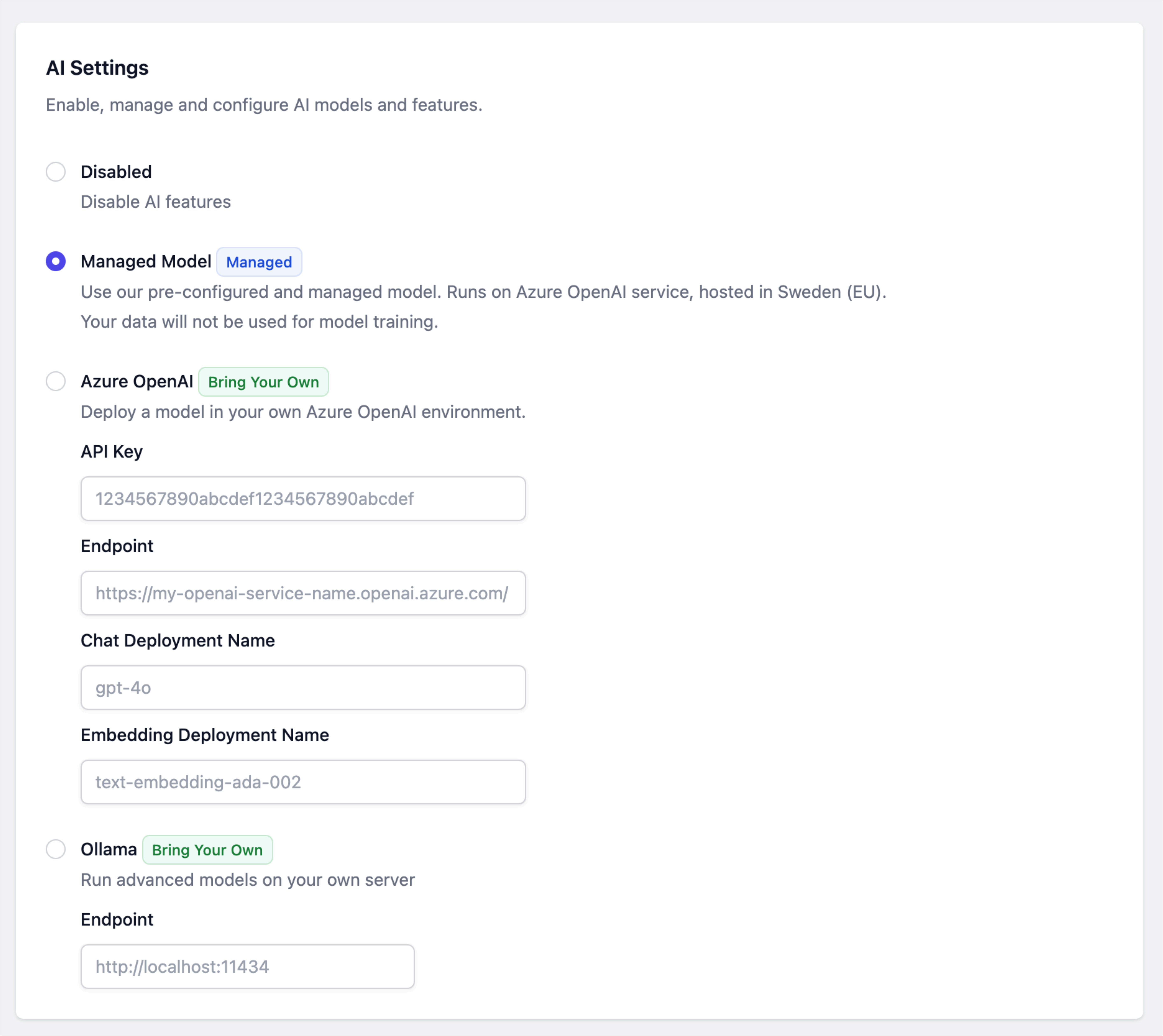
⚠ **Attention needed**

Data Governance AI checked this access request with your data governance policies and found these potential policy violations:

- **PII Processing**: The data set contains PII, but the purpose of the access request does not explain why PII fields are needed.
- **Data Transfer Policy (EU)**: The data set contains personal data and the request is for analyzing international customer cohorts, which may involve data transfer outside the EU.

AI can make mistakes. Check important info.

BYI AI Model



AI Settings

Enable, manage and configure AI models and features.

○ Disabled

Disable AI features

● Managed Model Managed

Use our pre-configured and managed model. Runs on Azure OpenAI service, hosted in Sweden (EU). Your data will not be used for model training.

○ Azure OpenAI Bring Your Own

Deploy a model in your own Azure OpenAI environment.

API Key

1234567890abcdef1234567890abcdef

Endpoint

https://my-openai-service-name.openai.azure.com/

Chat Deployment Name

gpt-4o

Embedding Deployment Name

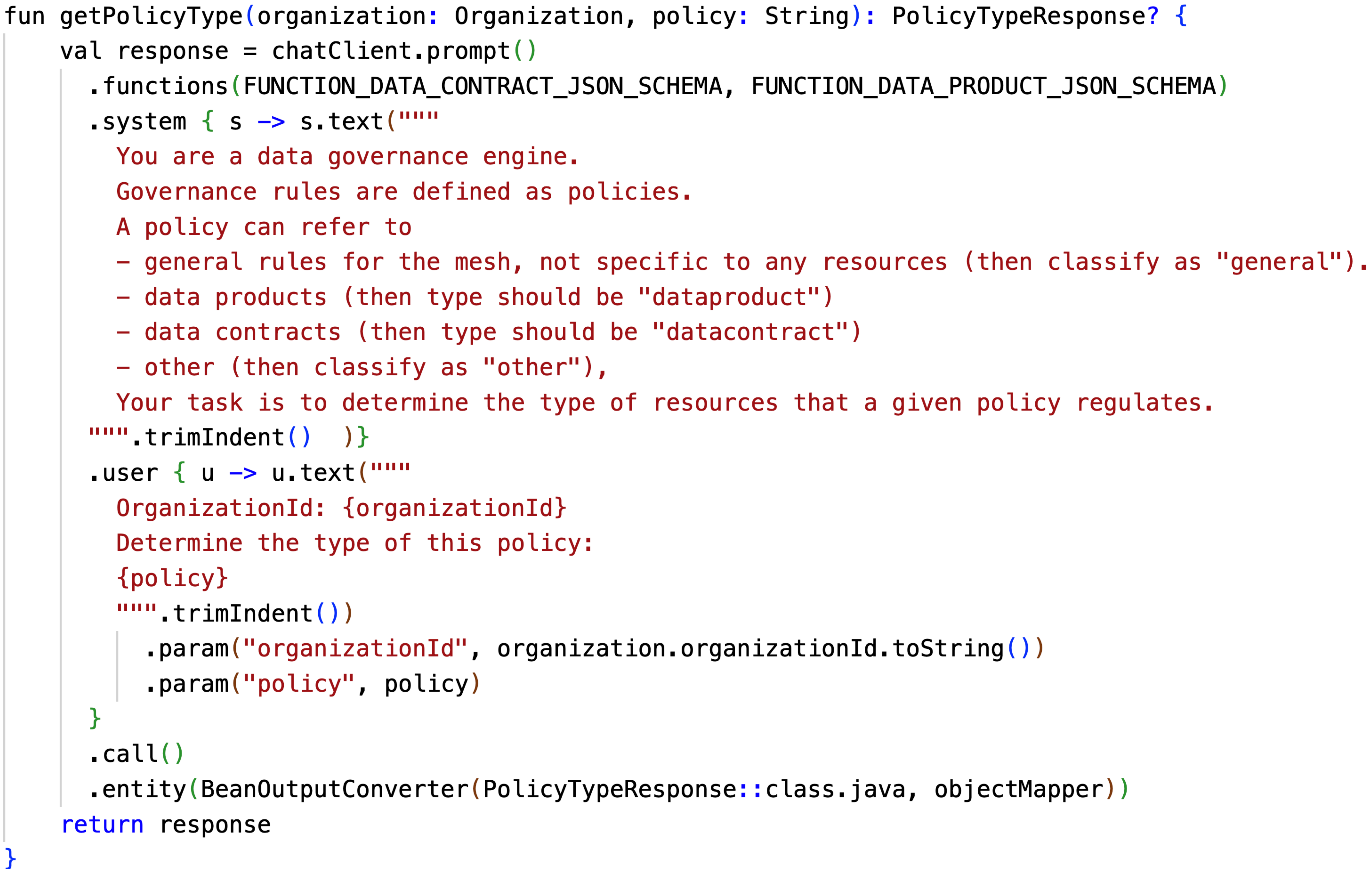
text-embedding-ada-002

○ Ollama Bring Your Own

Run advanced models on your own server

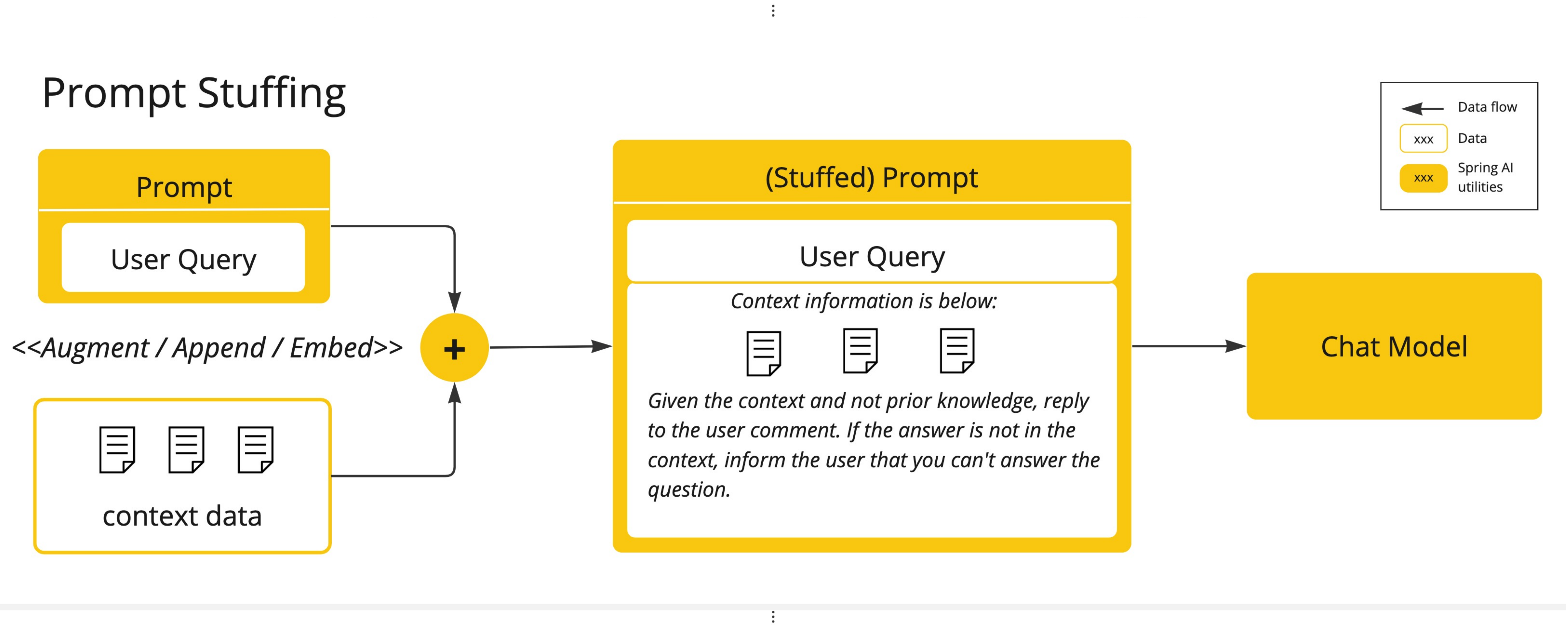
Endpoint

http://localhost:11434



```kotlin
fun getPolicyType(organization: Organization, policy: String): PolicyTypeResponse? {
    val response = chatClient.prompt()
      .functions(FUNCTION_DATA_CONTRACT_JSON_SCHEMA, FUNCTION_DATA_PRODUCT_JSON_SCHEMA)
      .system { s -> s.text("""
        You are a data governance engine.
        Governance rules are defined as policies.
        A policy can refer to
        - general rules for the mesh, not specific to any resources (then classify as "general").
        - data products (then type should be "dataproduct")
        - data contracts (then type should be "datacontract")
        - other (then classify as "other"),
        Your task is to determine the type of resources that a given policy regulates.
      """.trimIndent()  )}
      .user { u -> u.text("""
        OrganizationId: {organizationId}
        Determine the type of this policy:
        {policy}
        """.trimIndent())
          .param("organizationId", organization.organizationId.toString())
          .param("policy", policy)
      }
      .call()
      .entity(BeanOutputConverter(PolicyTypeResponse::class.java, objectMapper))
    return response
}
```

Spring AI: Prompt Stuffing (RAG)



Prompt Stuffing

Prompt

User Query

<<Augment / Append / Embed>>

+

context data

(Stuffed) Prompt

User Query

Context information is below:

Given the context and not prior knowledge, reply to the user comment. If the answer is not in the context, inform the user that you can't answer the question.

Chat Model

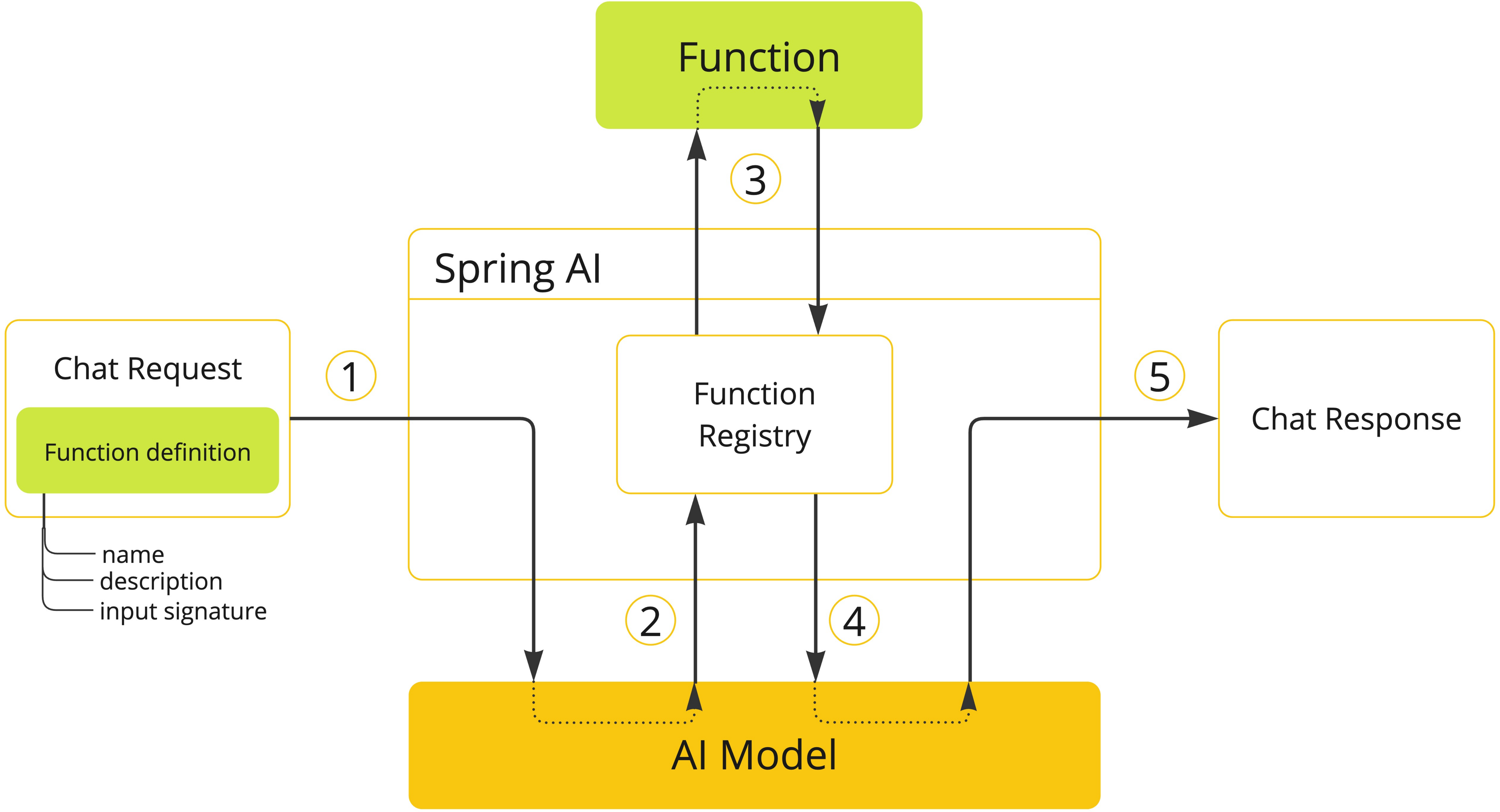
Data flow

xxx Data

xxx Spring AI utilities

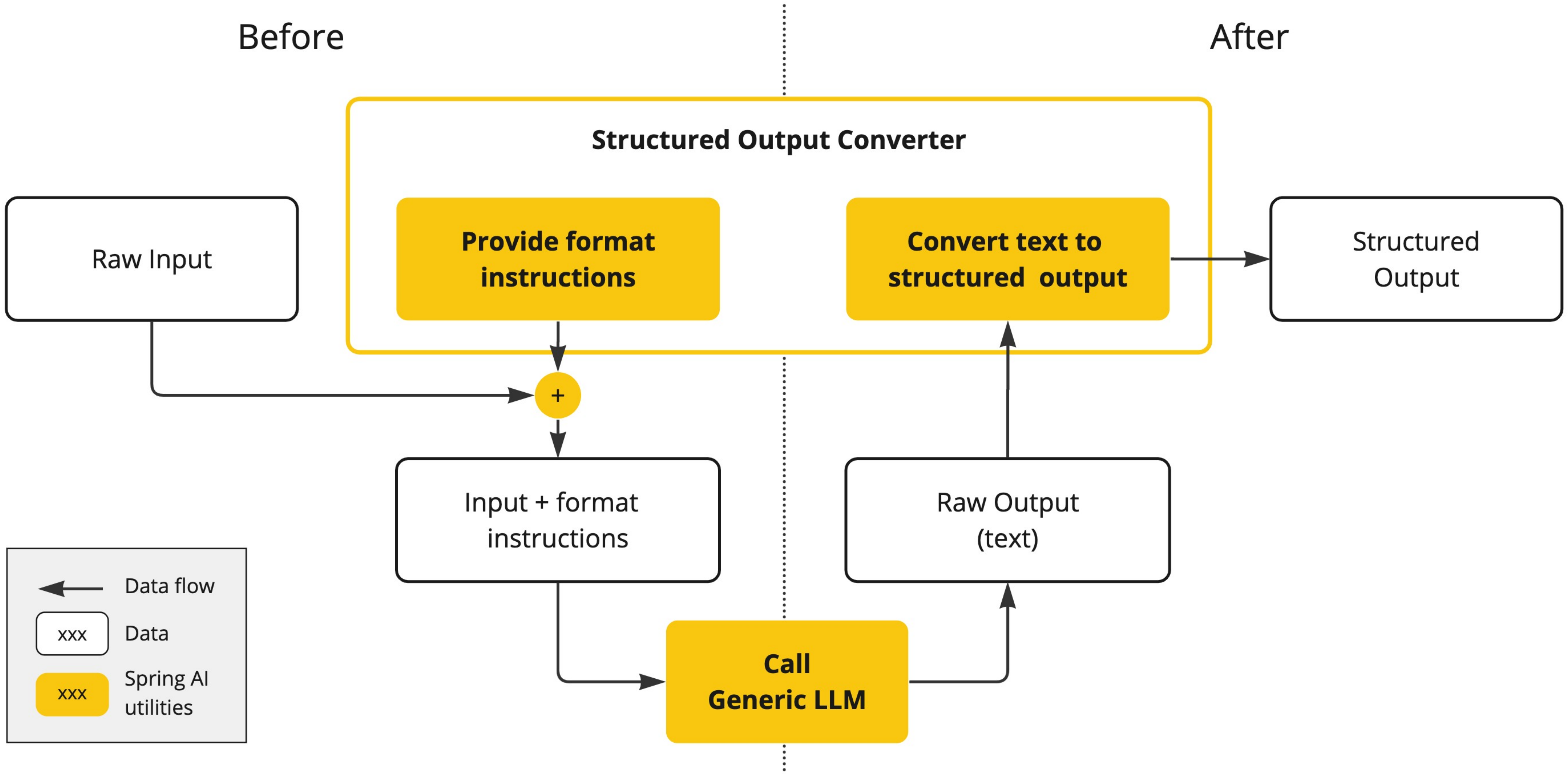
https://docs.spring.io/spring-ai/reference/concepts.html

Spring AI: Function Calling



https://docs.spring.io/spring-ai/reference/concepts.html

Spring AI: Structured Output


Before
After
Structured Output Converter
Raw Input
Provide format instructions
Convert text to structured output
Structured Output
+
Input + format instructions
Raw Output (text)
Call Generic LLM
Data flow
xxx
Data
xxx
Spring AI utilities

https://docs.spring.io/spring-ai/reference/concepts.html

Learnings

Learnings: Modern Data Governance

- Data Contracts are the elementary for modern data governance
- Ownership for governance shifts left to product owners
- Decentralized architectures need a central repository
- Automate as much as possible

Learnings: AI Engineering

- LLMs are another (powerful) tool in software architecture
- Invest in prompt engineering
- Testing is more complex (and costly)
- AI makes mistakes:
  - Use AI to detect issues
  - Ultimately, humans are responsible
  - Incorporate user feedback
- AI can't solve everything

Links

Open Source

- datacontract.com
- cli.datacontract.com
- editor.datacontract.com

Commercial

- datamesh-manager.com

Niemand macht gerne Data Governance - lassen wir es doch die AI machen

INNOQ

JOCHEN CHRIST
/IN/JOCHENCHRIST