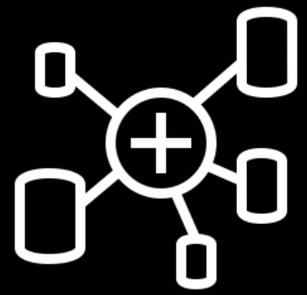

AWS
re:Invent

EMB019

Deep Dive on AWS Glue Elastic Views



Akshat Vig
Principal Engineer
AWS

Almann Goo
Principal Engineer
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Glue Elastic Views

Why AWS Glue Elastic Views?

What are AWS Glue Elastic Views?

Key features

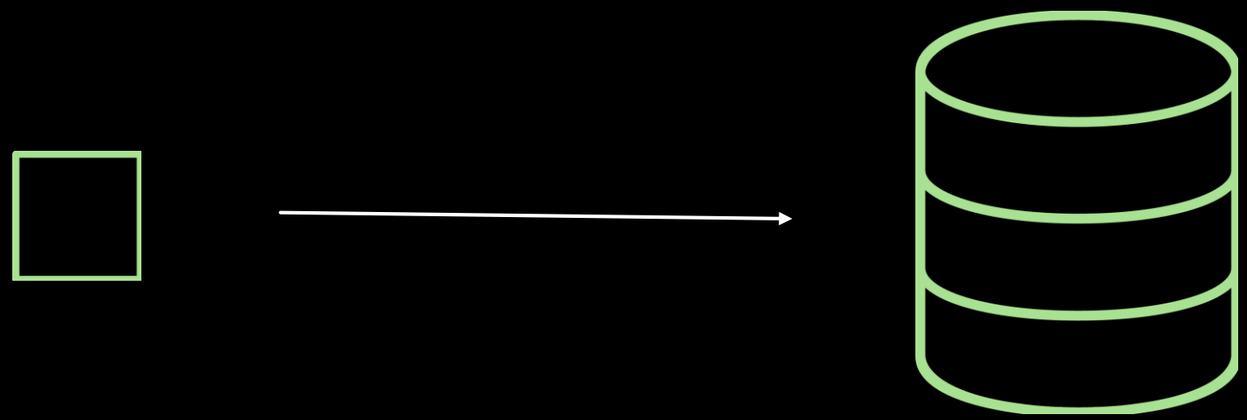
Scalability, correctness, and control



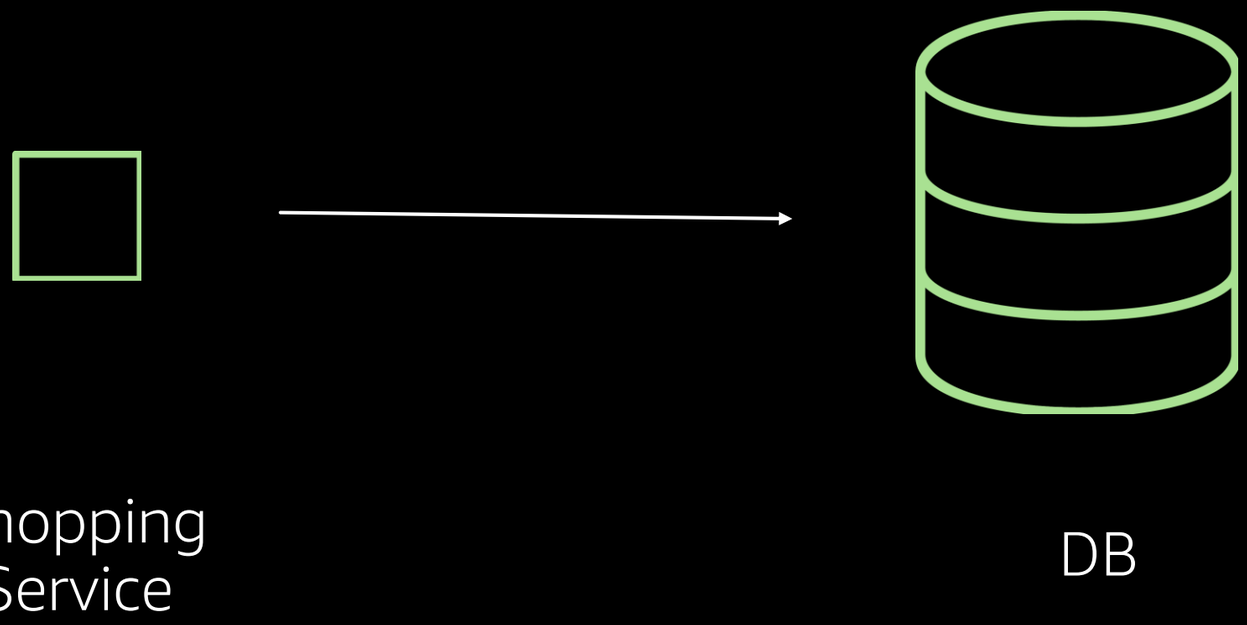
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Application development

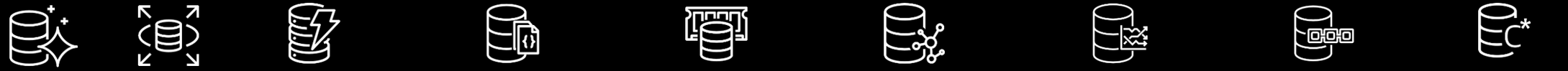


Online shopping portal



One size fits nothing at all

Purpose-built databases



Building for the cloud

Why AWS Glue Elastic Views?



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Shopping portal: Product catalog

Users: 1 million

Performance: Single-digit millisecond

Request rate: 1 million RPS

Economics: Pay as you go

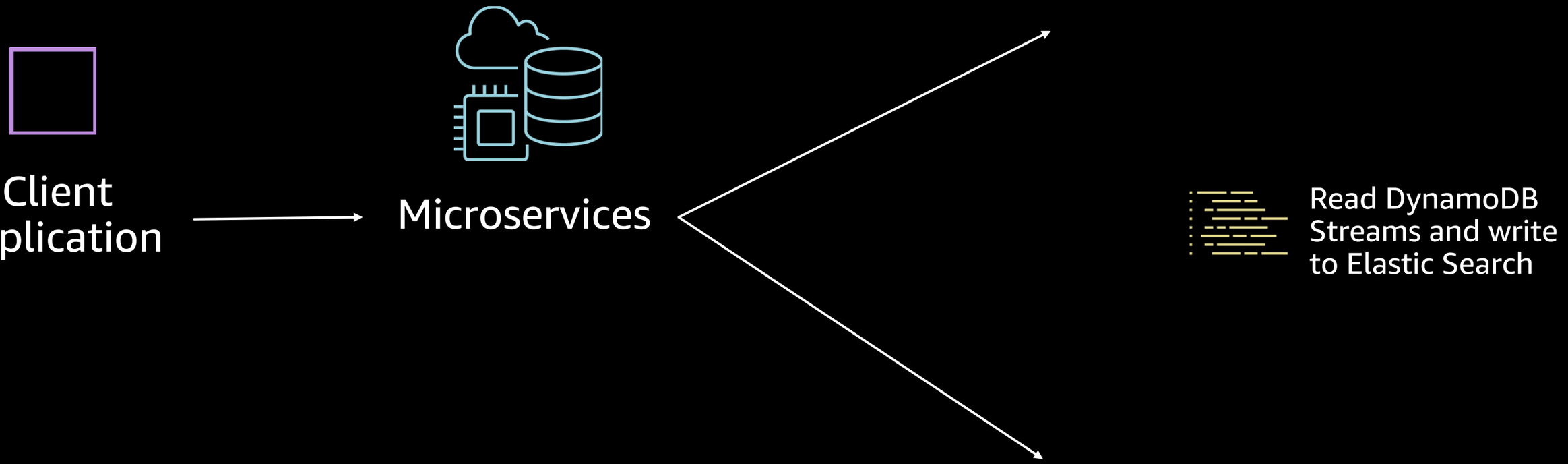
Developer access: Managed API

Imagine if the user wanted to do free text search on the portal…



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

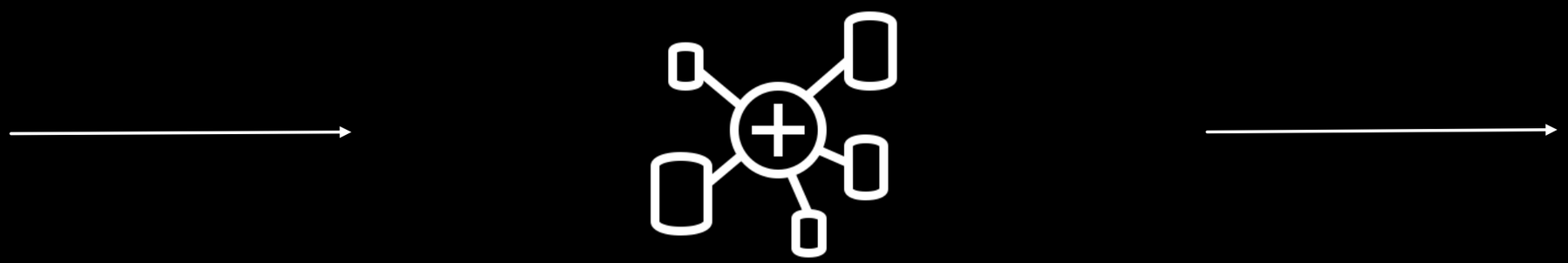
Microservices and applications



Move data between two different systems



AWS Glue Elastic Views



AWS Glue Elastic Views

What are Glue Elastic Views?



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Glue Elastic Views

Combine and **replicate data across multiple datastores** to create materialized views **without** having to write **custom code**

AWS Glue Elastic Views



Combine and transform data

AWS Glue Elastic Views



Easy to use

AWS Glue Elastic Views



Control

PartiQL

AWS Glue Elastic Views



Correct

Transactionally correct

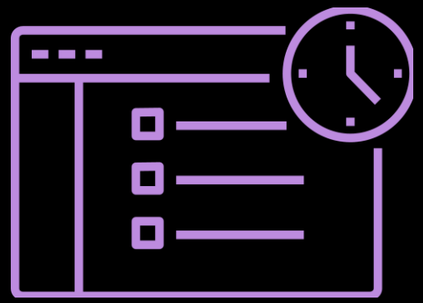
Transactional boundaries

Ordering

Correct transformation

Diverse data types

AWS Glue Elastic Views



Near-real time

AWS Glue Elastic Views

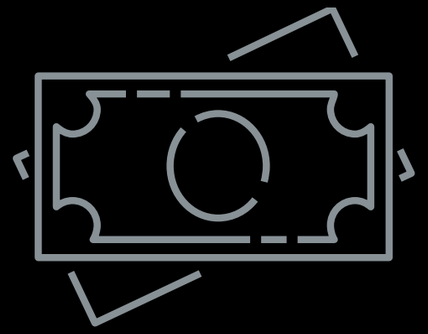


Secure

Encrypted in transit and at rest

AWS Identity and Access Management (IAM)

AWS Glue Elastic Views



Pay only for what you use

AWS Glue Elastic Views



Serverless

Highly available

Scalable

Ease of use



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Product Catalog: DynamoDB to Elasticsearch

Getting started

This is a tutorial for getting started with AWS Glue EV, in which you create and materialize your first view. For more details see Getting started with the AWS Glue EV console



1. Create and activate table

A table represents the data and schema (if any) of an external data store. First create your table, and then activate it.
Learn more

Create table



2. Create view

To create your view, specify a view definition query using PartiQL. Your query should reference the table created in the previous step.
Create view
AWS Glue EV PartiQL Reference Guide

Create view

AWS Glue EV APIs

AWS Glue EV APIs to configure tables, views and targets.
Getting started with AWS Glue EV APIs



3. Materialize view

Materialize your view in a target data store of your choice. Then, you can access the data of the materialized view at the target data store.
Prepare view for materialization from target
Materialize a view

Materialize view

Create table

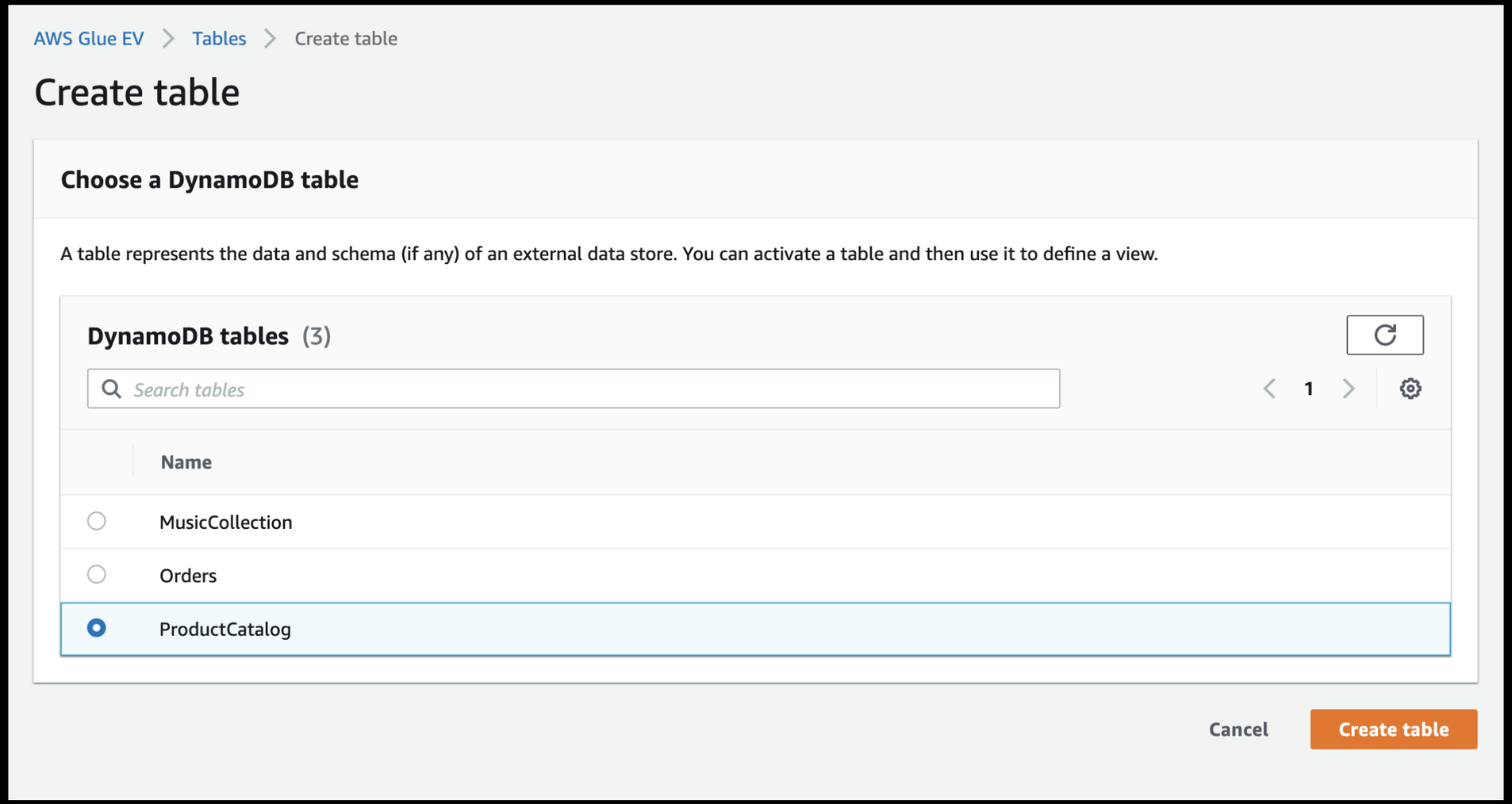
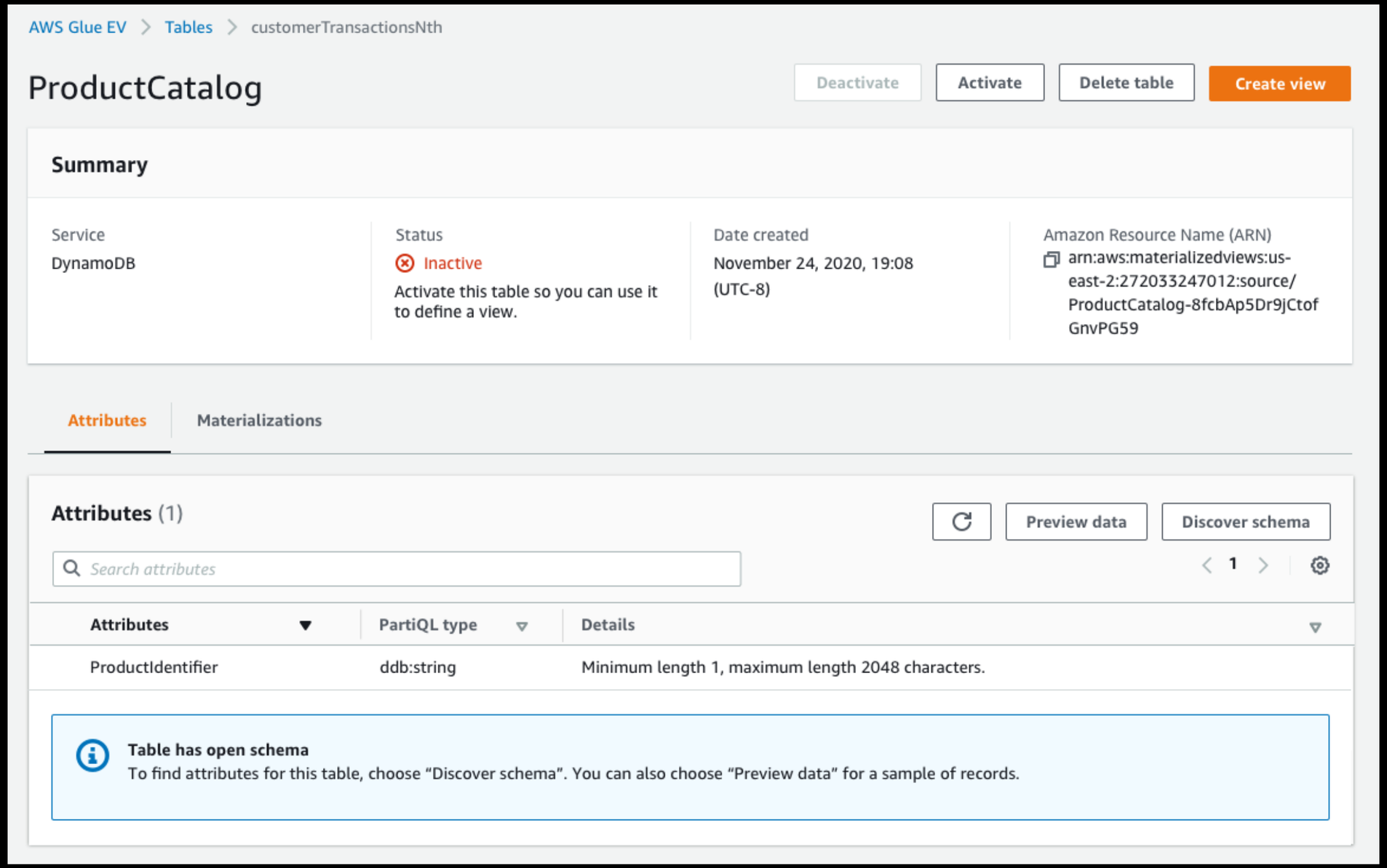
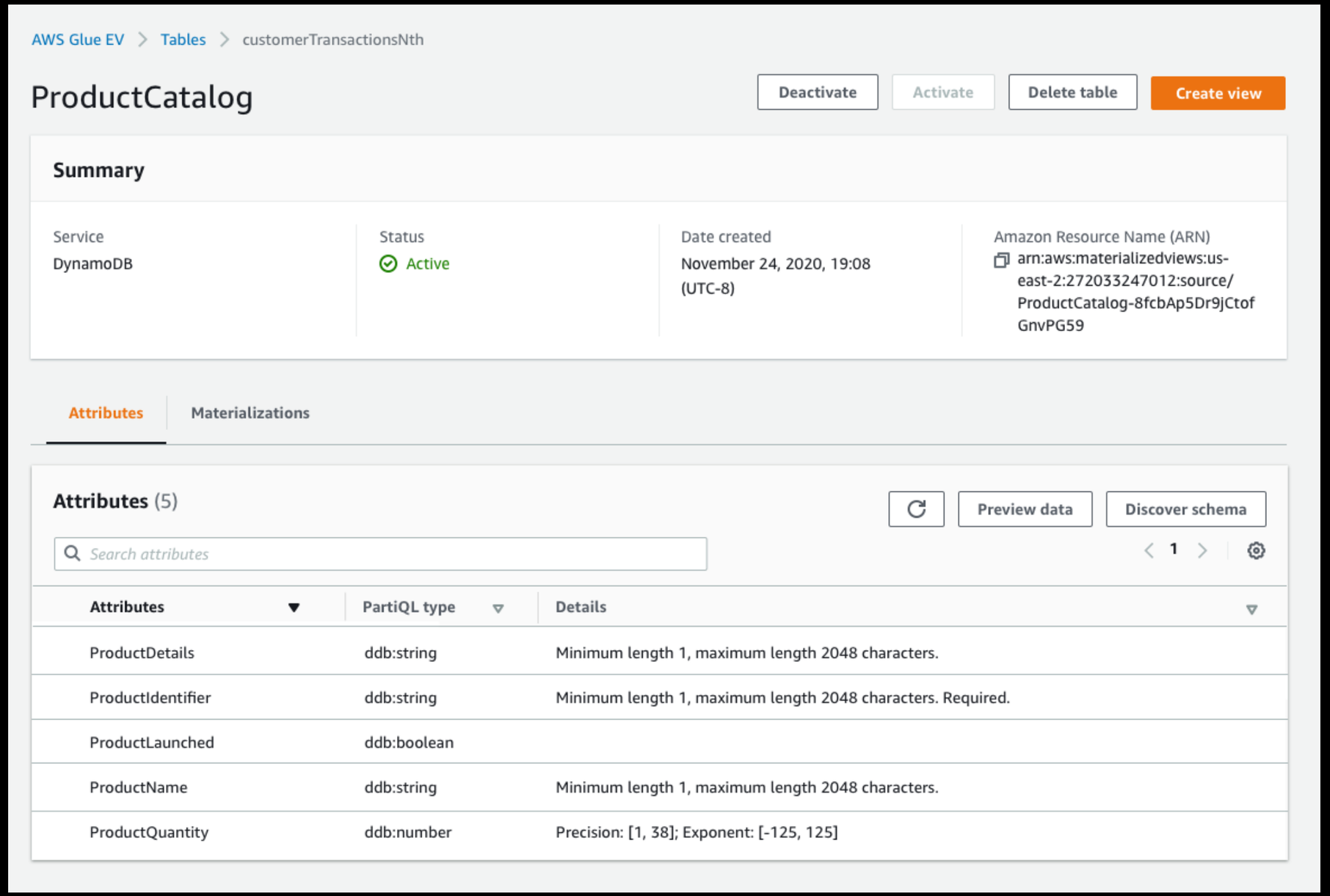


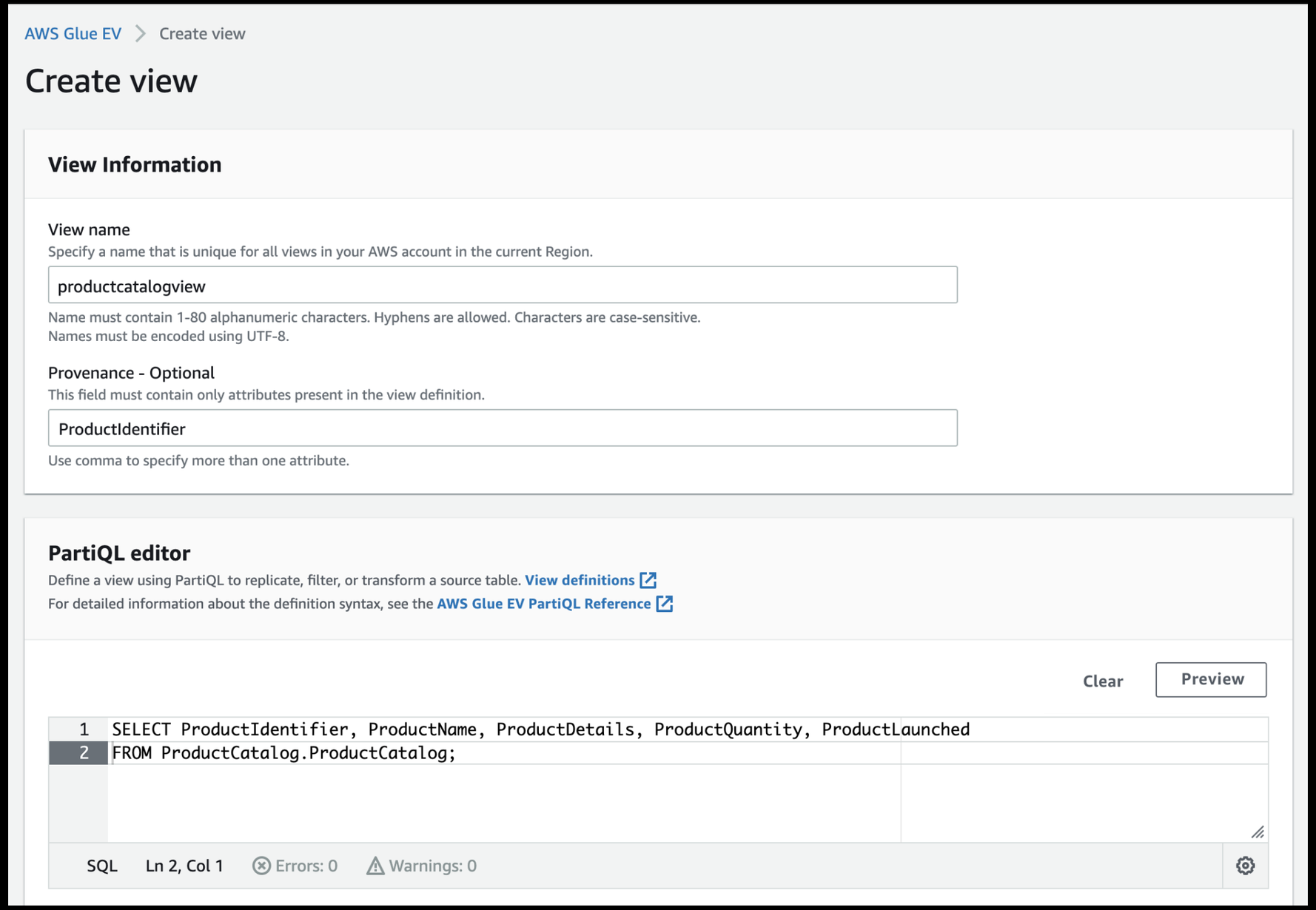
Table details



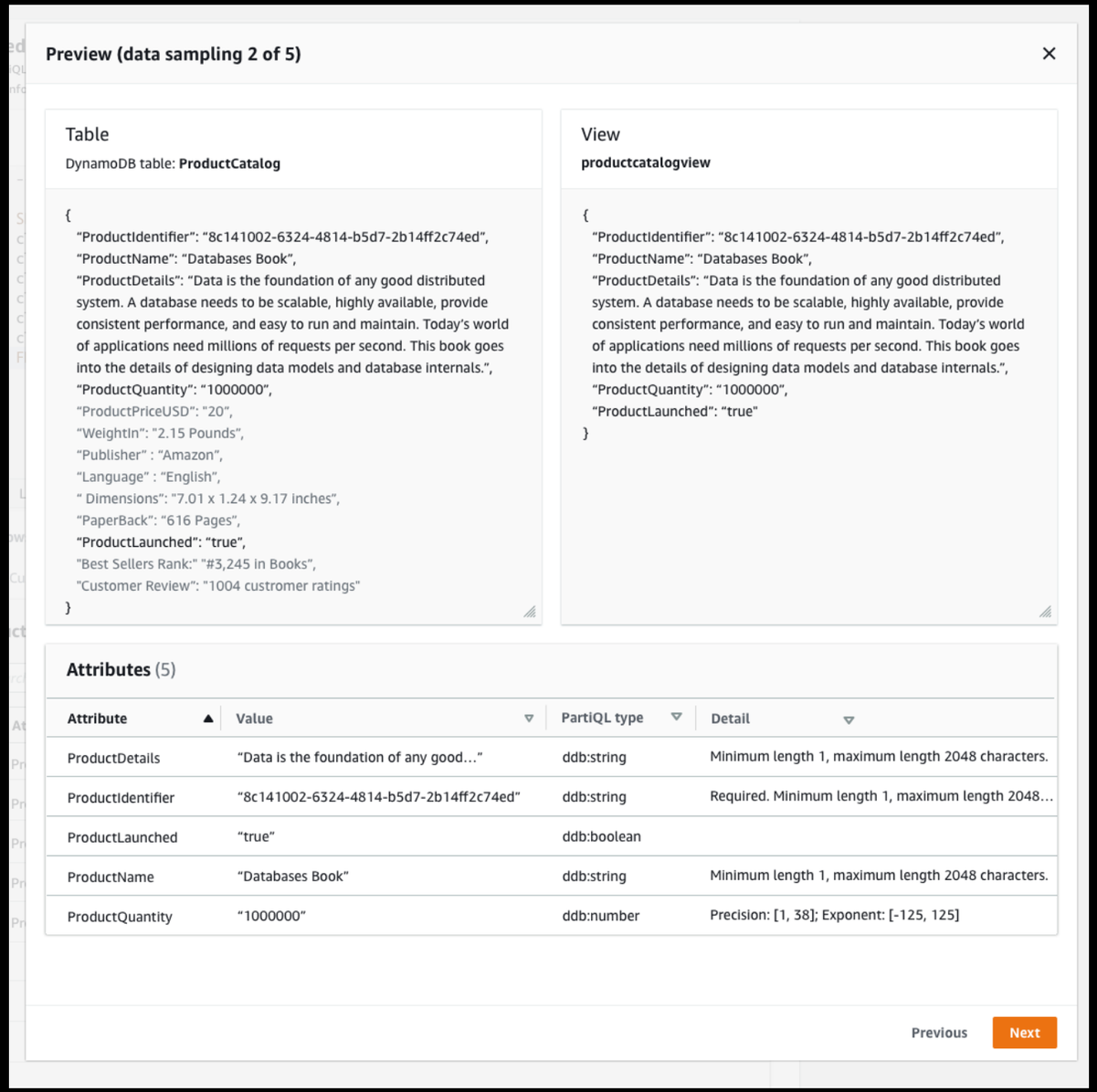
Discover schema



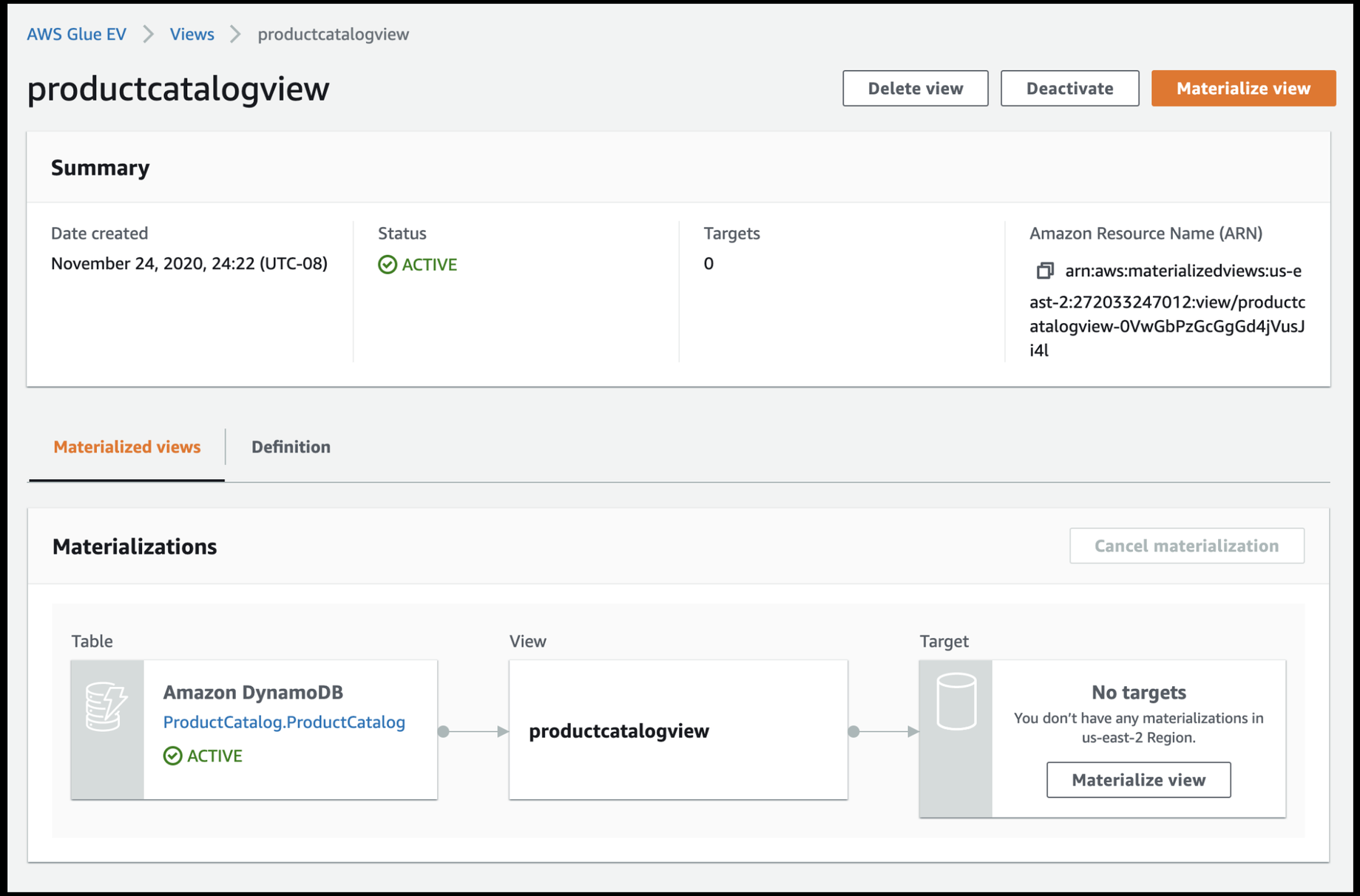
Create view


AWS Glue EV
Create view
Create view
View Information
View name
Specify a name that is unique for all views in your AWS account in the current Region.
productcatalogview
Name must contain 1-80 alphanumeric characters. Hyphens are allowed. Characters are case-sensitive.
Names must be encoded using UTF-8.
Provenance - Optional
This field must contain only attributes present in the view definition.
ProductIdentifier
Use comma to specify more than one attribute.
PartiQL editor
Define a view using PartiQL to replicate, filter, or transform a source table. View definitions
For detailed information about the definition syntax, see the AWS Glue EV PartiQL Reference
Clear
Preview
1 SELECT ProductIdentifier, ProductName, ProductDetails, ProductQuantity, ProductLaunched
2 FROM ProductCatalog.ProductCatalog;
SQL
Ln 2, Col 1
Errors: 0
Warnings: 0

Create View: Preview



View created



Materialize view

Add target

Choose a location to materialize the view. For example, choose a schema in Redshift or a domain in Elasticsearch.
Learn more

Target service

- (●) Amazon Elasticsearch Service 
- ( ) Amazon Redshift 
- ( ) Amazon S3 

Target location

Choose an Elasticsearch domain that is compatible for an AWS Glue EV view.
To create a domain for your view, go to the Elasticsearch console

arn:aws:es:us-east-2:272033247012:domain/productcatalog

Target name

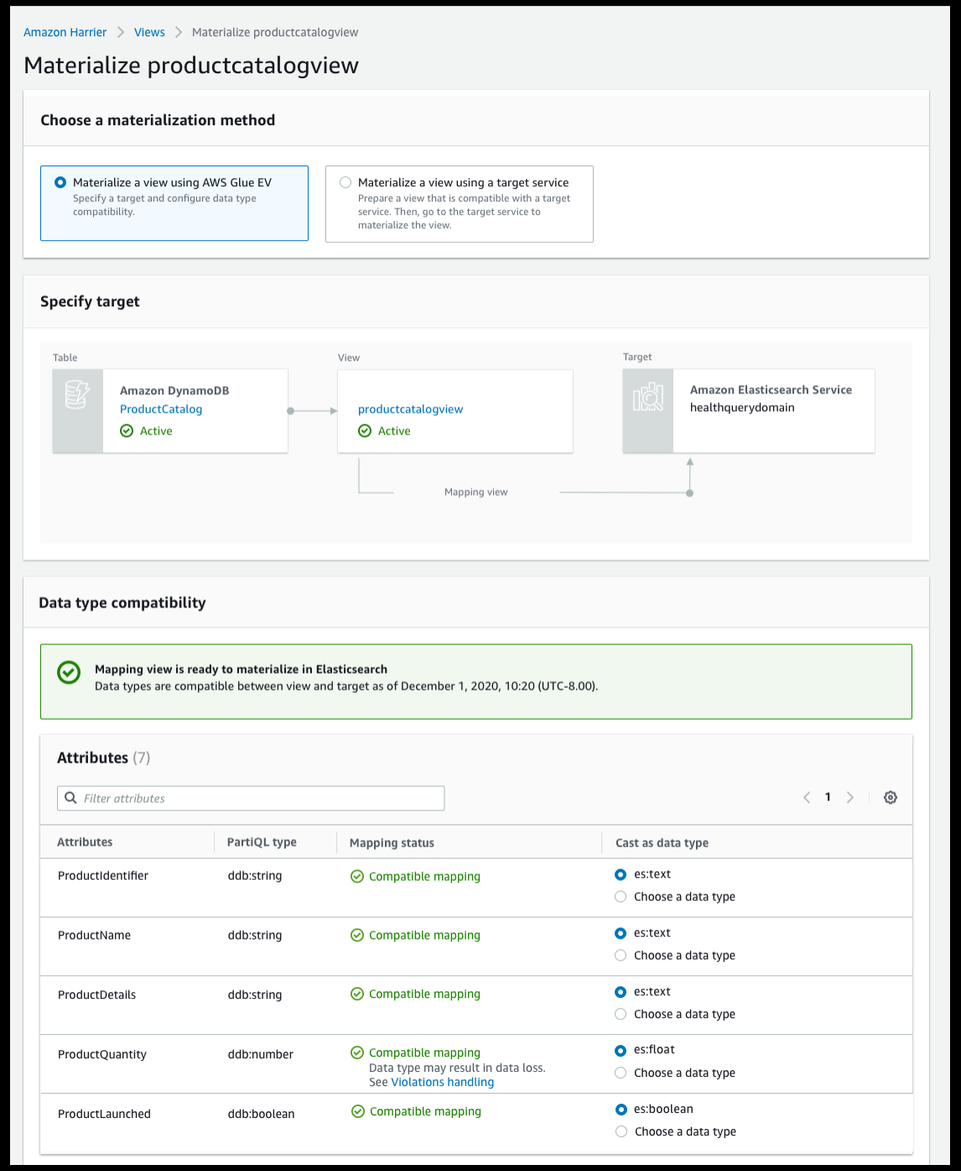
Target name
Specify a name for a target. The name must be unique for all targets per service that are in your AWS account in the current Region.

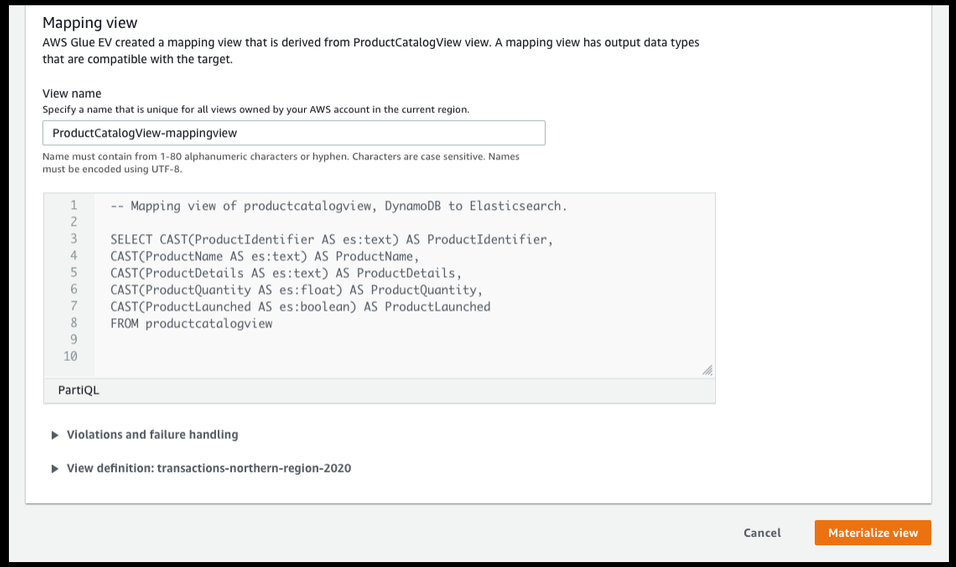
healthquerydomain

Name must contain 1-80 alphanumeric characters (lowercase only), underscore, hyphen.
First character must be alphabetic character.

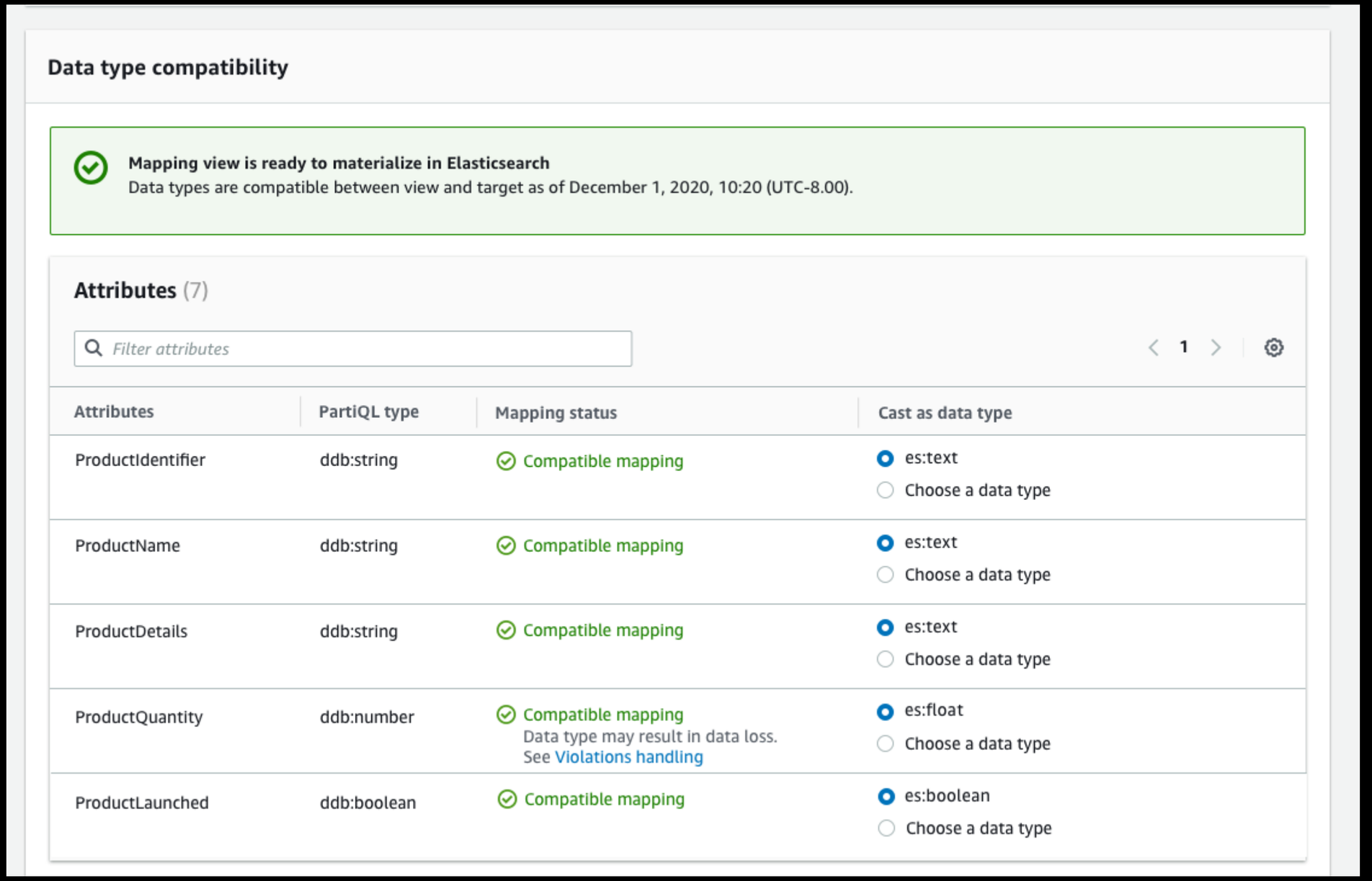
Cancel | Add target

Data type compatibility and recommendations





Materialize view



Data type compatibility

Mapping view is ready to materialize in Elasticsearch
Data types are compatible between view and target as of December 1, 2020, 10:20 (UTC-8.00).

Attributes (7)

Filter attributes

1

Attributes	PartiQL type	Mapping status	Cast as data type
ProductIdentifier	ddb:string	Compatible mapping	es:text Choose a data type
ProductName	ddb:string	Compatible mapping	es:text Choose a data type
ProductDetails	ddb:string	Compatible mapping	es:text Choose a data type
ProductQuantity	ddb:number	Compatible mapping Data type may result in data loss. See Violations handling	es:float Choose a data type
ProductLaunched	ddb:boolean	Compatible mapping	es:boolean Choose a data type

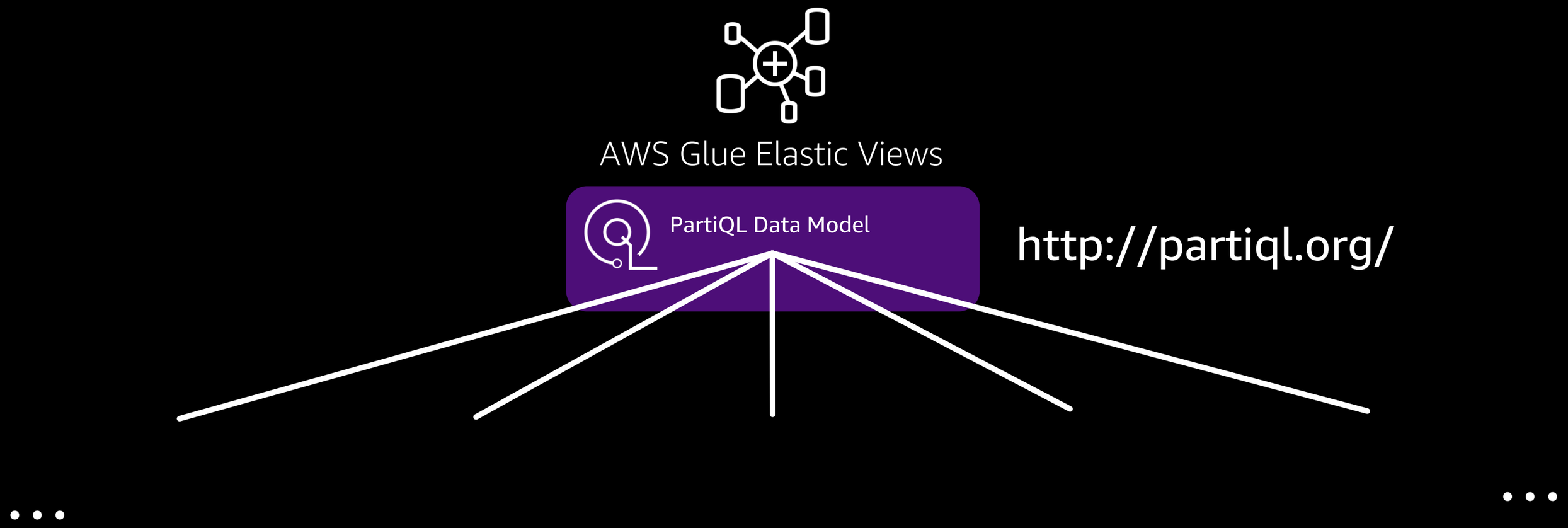
Programmability and correctness



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Diverse data types

WITH DIFFERENT CONSTRAINTS AND LIMITATIONS



Diverse Data Types

SUPPORT FOR STRUCTURED, NESTED, AND SEMI-STRUCTURED

orders *(table)*

order_id	sku	qty	price
1	abc-123	3	0.99
2	def-456	1	22.12

line_items *(table)*

order_id	sku	qty	price
1	abc-123	3	0.99
1	def-456	1	22.12
2	ghi-789	5	16.30

orders *(document)*

```
{
  order_id: 1,
  line_items: [
    {sku: "abc-123", qty: 3, price: 0.99},
    {sku: "def-456", qty: 1, price: 22.12}
  ]
}
{
  order_id: 2,
  line_items: [
    {sku: "ghi-789", qty: 5, price: 16.30}
  ]
}
```

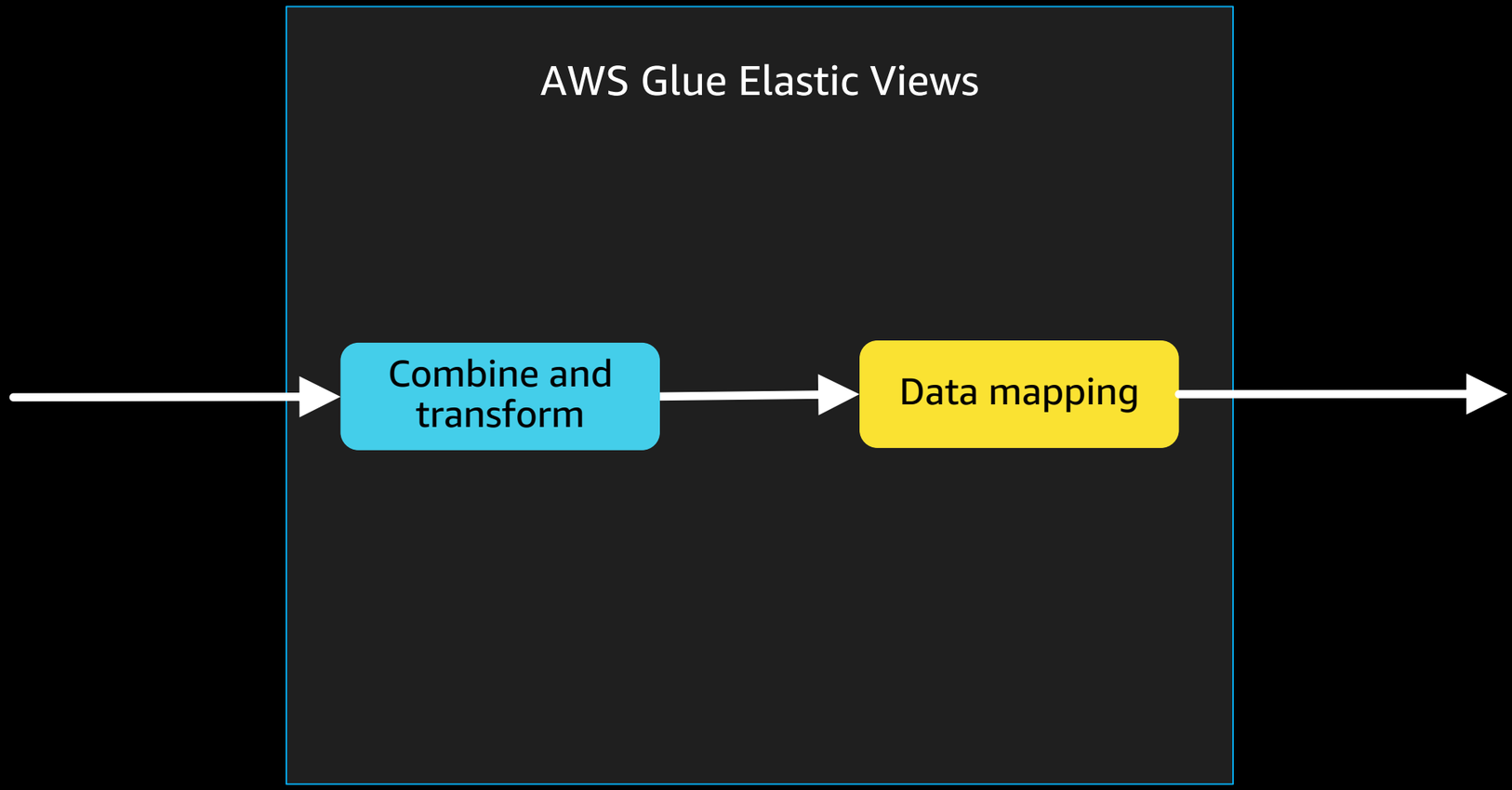
Imagine if the user wanted to perform some tax analytic reporting on the orders…



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

DynamoDB to Amazon Redshift

MATERIALIZING ANALYTIC VIEWS FROM OUR OPERATIONAL STORE



Transformation View in PartiQL

ONE CONSISTENT WAY TO OPERATE ON YOUR DATA

```
CREATE VIEW hi_line_items AS
SELECT
  o.order_id, l.sku, l.qty, l.price,
  l.price * 0.045 AS tax
FROM
  orders AS o,
  o.line_items AS l
WHERE o.city = 'Honolulu'
```

Transformation View in PartiQL

LET'S TAKE A LOOK AT THE DYNAMODB EXAMPLE

orders

```
{
  order_id: 1,
  city: "Honolulu",
  line_items: [
    {sku: "abc-123", qty: 3, price: 0.99},
    {sku: "def-456", qty: 1, price: 22.12}
  ]
}
```

Transformation View in PartiQL

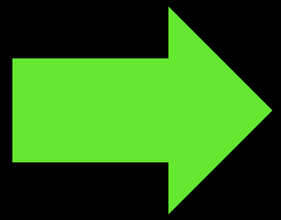
ONE CONSISTENT WAY TO OPERATE ON YOUR DATA

```
CREATE VIEW hi_line_items AS
SELECT
  o.order_id, l.sku, l.qty, l.price,
  l.price * 0.045 AS tax
FROM
  orders AS o,
  o.line_items AS l
WHERE o.city = 'Honolulu'
```

Transformation View in PartiQL

ONE CONSISTENT WAY TO OPERATE ON YOUR DATA

```
{
  order_id: 1.,
  city: "Honolulu",
  line_items: [
    {
      sku: "abc-123",
      qty: 3.,
      price: 0.99
    },
    {
      sku: "def-456",
      qty: 1.,
      price: 22.12
    }
  ]
}
```



```
{
  order_id: 1.,
  sku: "abc-123",
  qty: 3.,
  price: 0.99,
  tax: 0.04455,
}
{
  order_id: 1.,
  sku: "def-456",
  qty: 1.,
  price: 22.12,
  tax: 0.9954,
}
```

Data compatibility

DEALING WITH THE TARGET CONSTRAINTS

Attribute	DynamoDB type	PartiQl type	Redshift type	Target PartiQl type
order_id	NUMBER	**DECIMAL** *precision: [1, 38]* *exponent: [-130, 125]* *no trailing zeroes*	INT8	**INTEGER** *range:[-2^{63}, 2^{63}-1]*
sku	STRING	**STRING** *byte length <= 400KB*	VARCHAR(65535)	**STRING** *byte length <= 65KB*
qty	NUMBER	**DECIMAL** *(see above)*	INT4	**INTEGER** *range:[-2^{31}, 2^{31}-1]*
price	NUMBER	**DECIMAL** *(see above)*	DECIMAL(7,2)	**DECIMAL(7,2)**
tax	*N/A (computed)*	**DECIMAL**	DECIMAL(7,2)	**DECIMAL(7,2)**

Mapping View

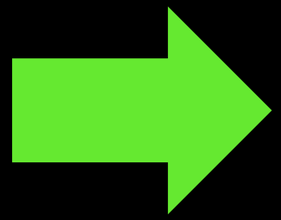
CONVERSIONS ARE EXPLICIT

```
CREATE VIEW hi_line_items_rs AS
SELECT
  CAST(order_id AS INT8),
  CAST(sku      AS RS_VARCHAR_MAX),
  CAST(qty      AS INT4),
  CAST(price    AS DECIMAL(7,2)),
  CAST(tax      AS DECIMAL(7,2))
FROM
  hi_line_items
```

Mapping View

CONVERSIONS ARE EXPLICIT

```
{
  order_id: 1.,
  sku: "abc-123",
  qty: 3.,
  price: 0.99,
  tax: 0.04455,
}
{
  order_id: 1.,
  sku: "def-456",
  qty: 1.,
  price: 22.12,
  tax: 0.9954,
}
```

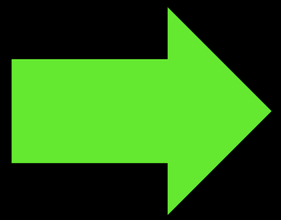


```
{
  order_id: 1,
  sku: "abc-123",
  qty: 3,
  price: 0.99,
  tax: 0.04,
}
{
  order_id: 1,
  sku: "def-456",
  qty: 3,
  price: 22.12,
  tax: 1.00
}
```

Unexpected Data

WHAT HAPPENS IF DATA ISN'T QUITE WHAT WE EXPECT

```
{
  order_id: 1.,
  sku: {{YWJjLTEyMw==}},
  qty: 3.,
  price: 0.99,
  tax: 0.04455,
}
{
  order_id: 1.,
  sku: "def-456",
  qty: 1.,
  price: 22.12,
  tax: 0.9954,
}
```



```
{
  order_id: 1.,
  bad_sku: {{YWJjLTEyMw==}},
}
```

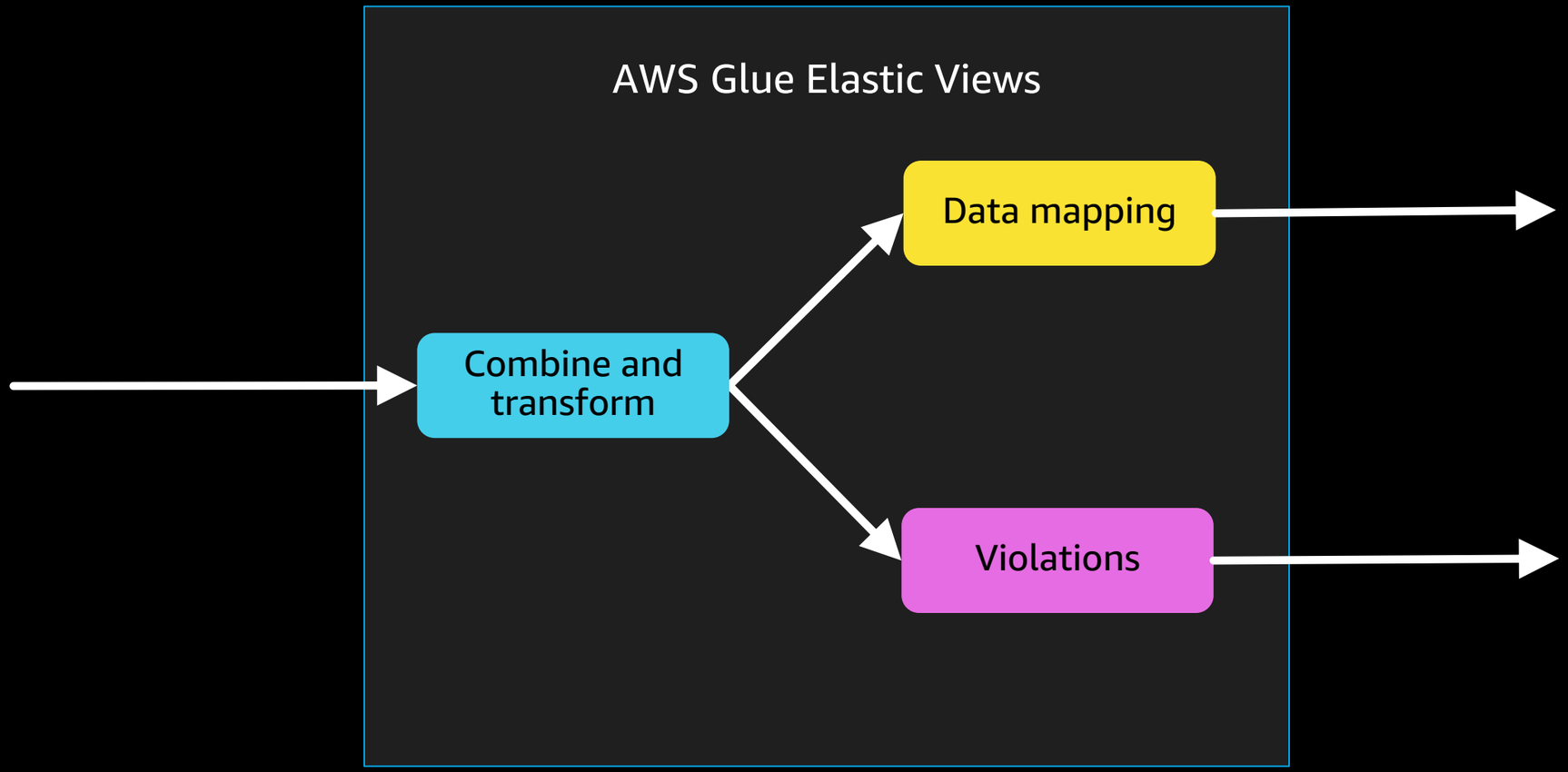
Violation Views

FULL CONTROL TO DEAL WITH CONVERSION ERRORS

```
CREATE VIEW hi_line_items_v AS
SELECT order_id,
   CASE WHEN NOT CAN_CAST(sku AS RS_VARCHAR_MAX)
     THEN sku ELSE MISSING END AS bad_sku,
…
FROM
   hi_line_items
WHEN
   NOT CAN_CAST(sku AS RS_VARCHAR_MAX) OR …
```

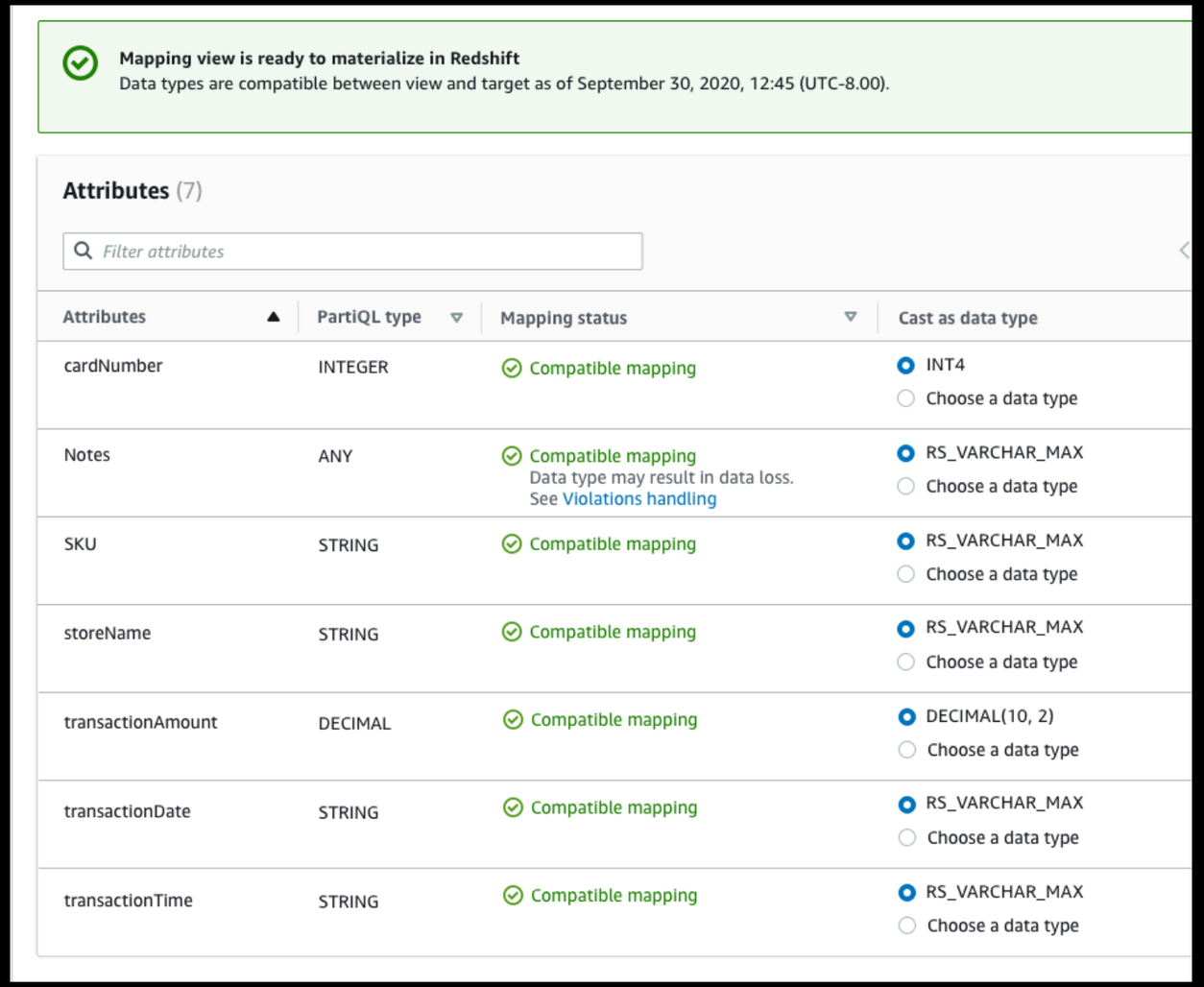
DynamoDB to Redshift and S3

MATERIALIZING ANALYTIC VIEWS FROM OUR OPERATIONAL STORE



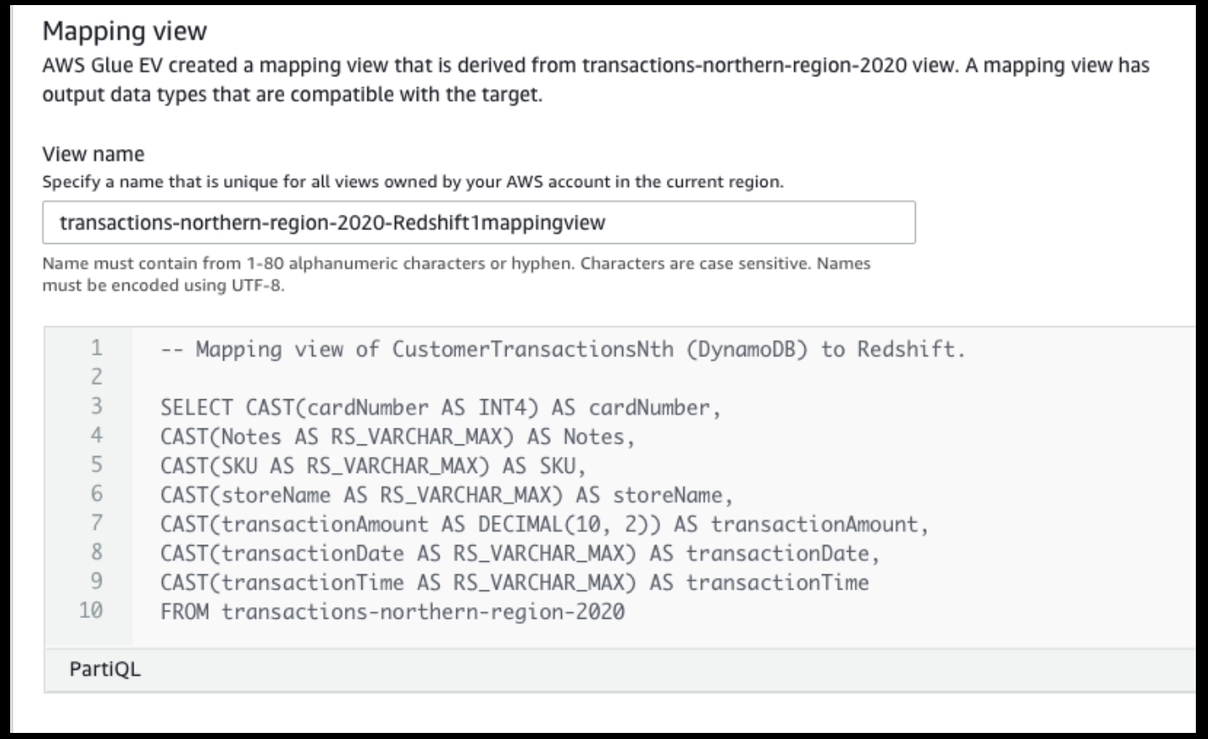
Easy is not either/or

AWS GLUE ELASTIC VIEWS CONSOLE COMBINES POWER WITH EASE

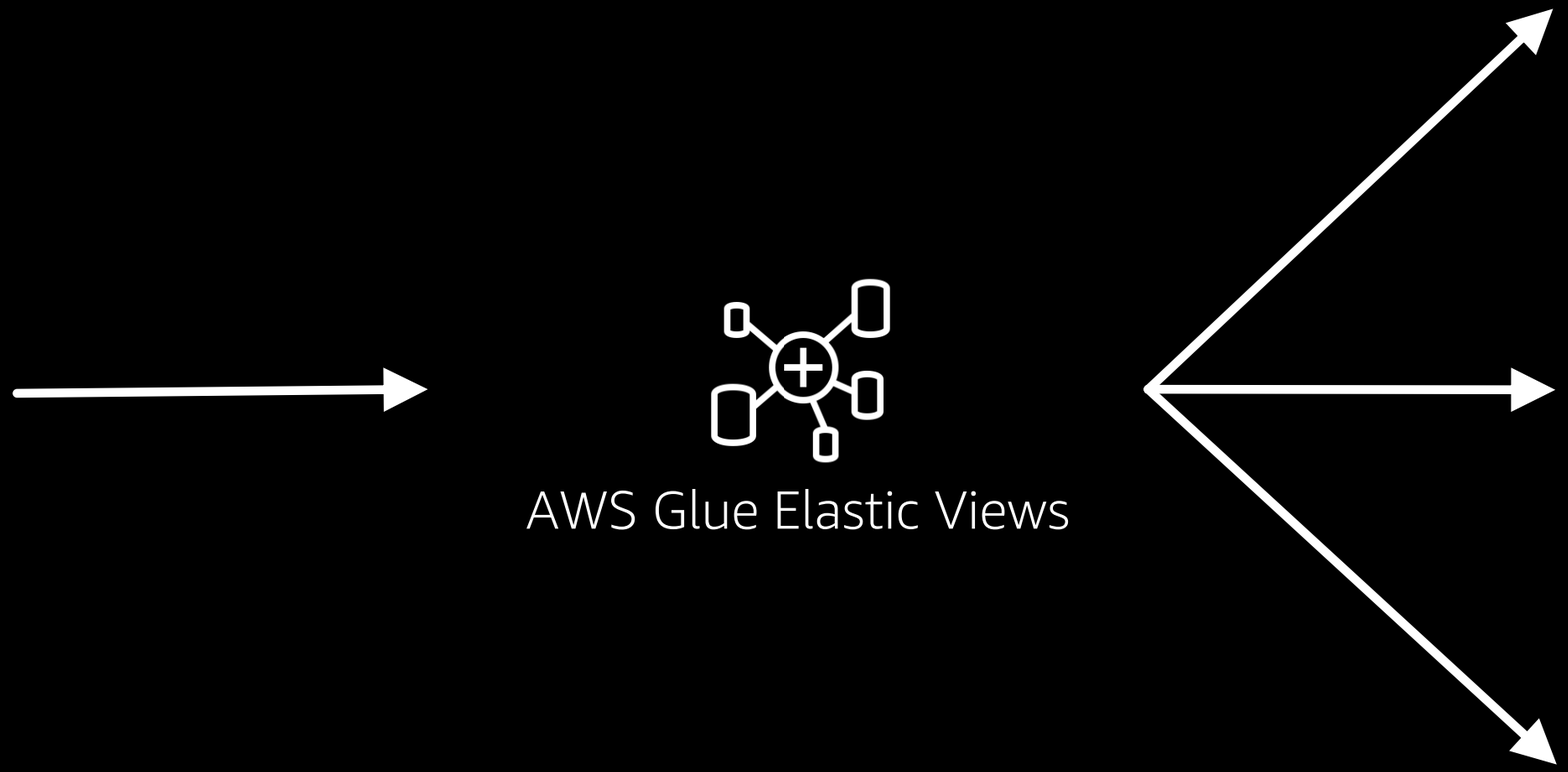


Easy is not either/or

AWS GLUE ELASTIC VIEWS CONSOLE COMBINES POWER WITH EASE



Glue Elastic Views: Recap



Glue Elastic Views: What's coming in 2021

Combination of sources and targets
Enhanced PartiQL support

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.