Provisional Upload Procedure to Celonis OCDM

Due to recent changes breaking compatibility, the uploader from OCEL to Celonis OCDM is not working anymore. This document specifies a provisional upload procedure, waiting for automated support by the new APIs.

The first step is about downloading the file **splitter.py**:

https://github.com/Javert899/ocel20-celonis-connector/blob/main/splitter.py

And modifying the bottom part (just after __main__) to ingest your OCEL 1.0 / 2.0 object-centric event log using pm4py. Executing the file, a collection of CSVs is stored in the folder "target", which are the base for the next steps of the upload procedure.

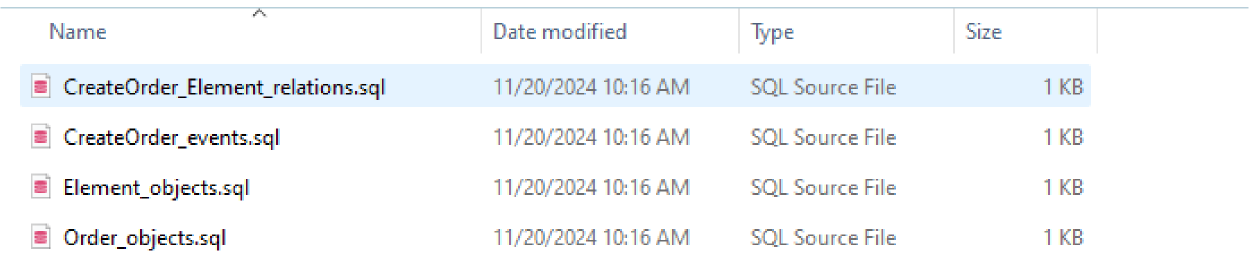
For a minimal example, we loaded the OCEL 1.0 event log available at:

https://github.com/pm4py/pm4py-core/blob/release/tests/input_data/ocel/example_log.jsonocel

and filtered that on a minimal configuration of object and event types:

- Object types: order, element
- Event types: Create Order
- Relationships:
  - (Create Order, order) 1:1
  - (Create Order, element) 1:N

Executing the script, we get the following files inside the "target" folder:



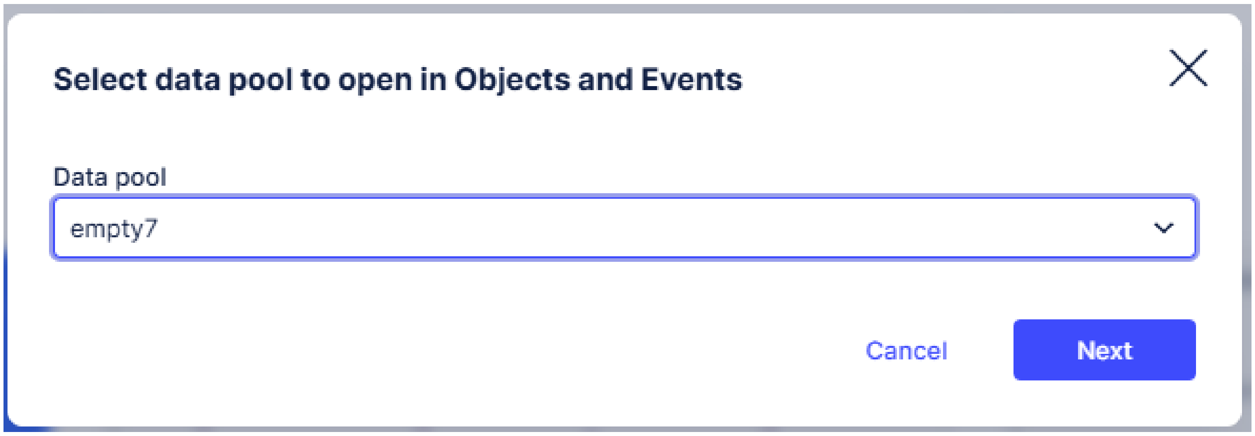
Name	Date modified	Type	Size
CreateOrder_Element_relations.sql	11/20/2024 10:16 AM	SQL Source File	1 KB
CreateOrder_events.sql	11/20/2024 10:16 AM	SQL Source File	1 KB
Element_objects.sql	11/20/2024 10:16 AM	SQL Source File	1 KB
Order_objects.sql	11/20/2024 10:16 AM	SQL Source File	1 KB

We see the two object types stored as SQL files, the activity stored as SQL file, and the (potentially) many-to-many relationships stored in a separate “relations” SQL file.

The next steps are done inside Celonis.

First, create an empty data pool (in our case, “empty7”).

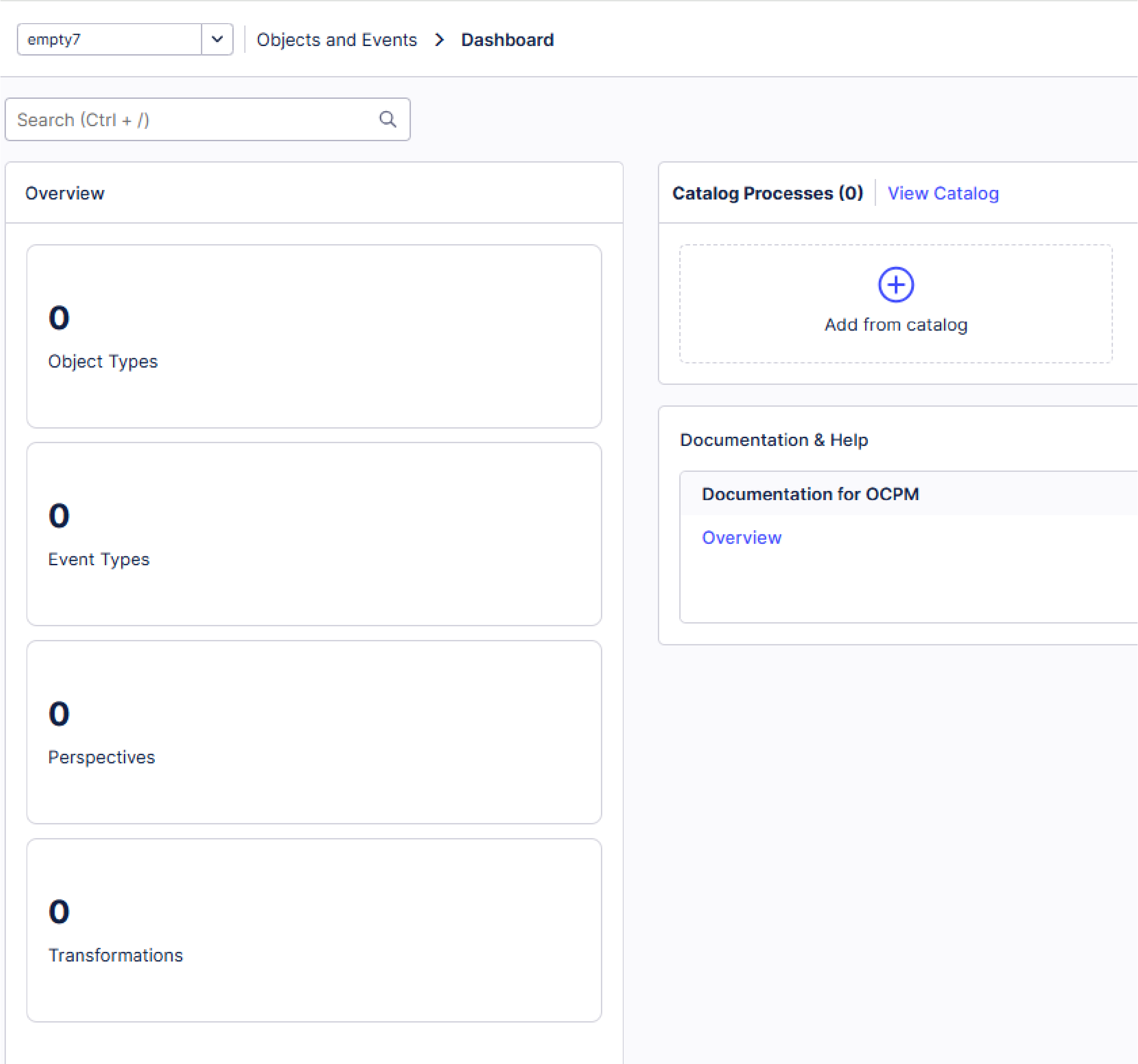
Then, reach the “Objects & Events” feature



Select data pool to open in Objects and Events

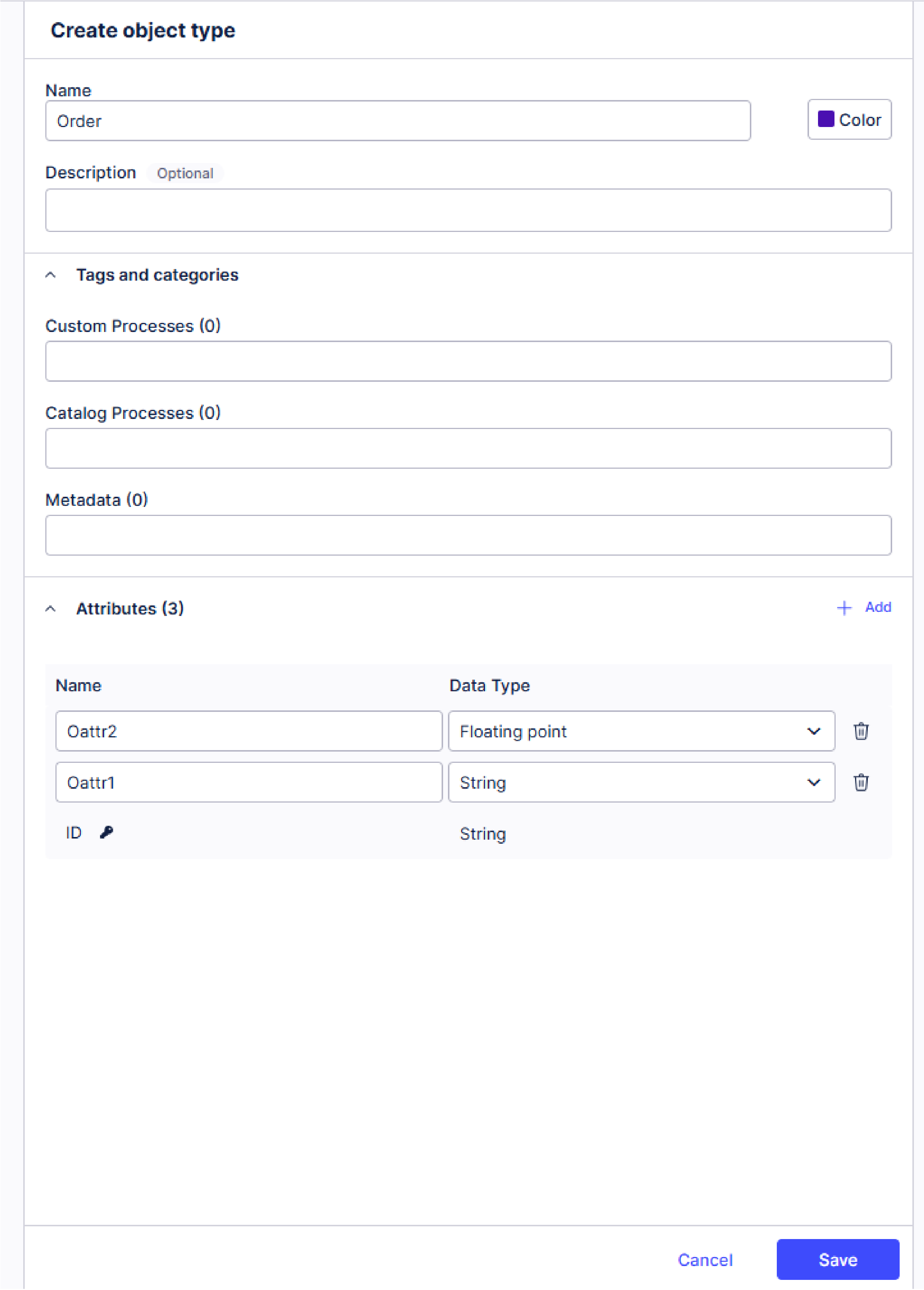
Data pool

empty7



click on “Object Types” and start the creation of an object type.

Let’s start by “Order”, adding the additional attributes that are associated to an “Order” in the given OCEL.



Create object type

Name

Order

Description Optional

Tags and categories

Custom Processes (0)

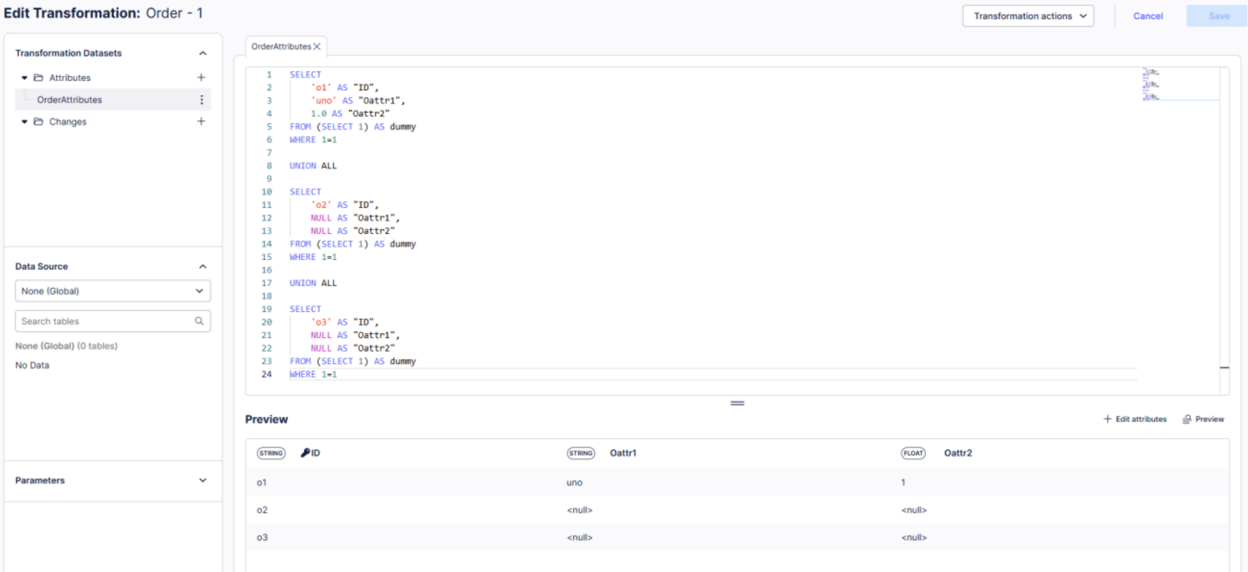
Catalog Processes (0)

Metadata (0)

Attributes (3)

Name	Data Type
Oattr2	Floating point
Oattr1	String
ID	String

To “inject” the actual objects, reach the “Transformations” page by clicking the corresponding button on the right of the page. Proceed to “Add transformation” and copy-paste inside the SQL instructions. Click “Save” and then “Preview”.

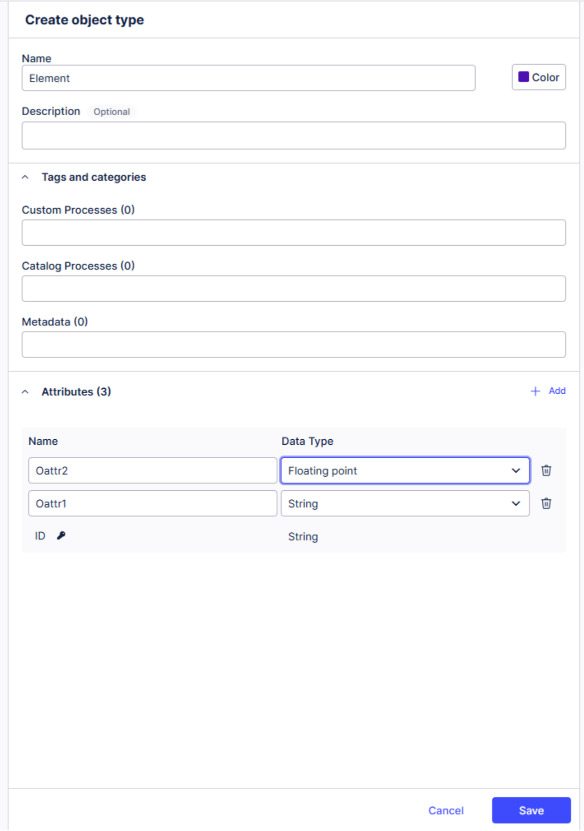


Edit Transformation: Order - 1

```
SELECT
    'o1' AS "ID",
    'uno' AS "Oattr1",
    1.0 AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'o2' AS "ID",
    NULL AS "Oattr1",
    NULL AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'o3' AS "ID",
    NULL AS "Oattr1",
    NULL AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1
```

STRING ID	STRING Oattr1	FLOAT Oattr2
o1	uno	1
o2	<null>	<null>
o3	<null>	<null>

Then, come back to the “Object Types” page of the “Objects and Events” feature. Let’s proceed to add the other object type, “Element”.

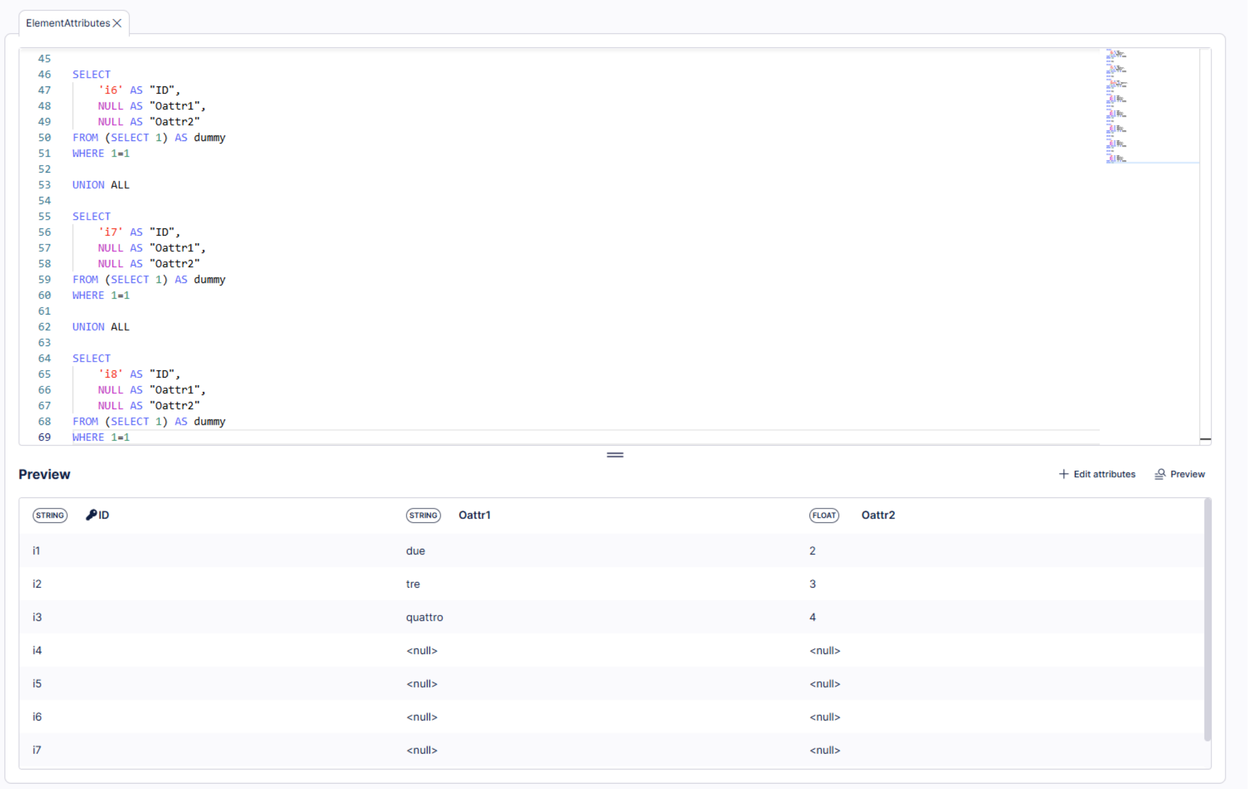


Create object type

Name: Element

Name	Data Type
Oattr2	Floating point
Oattr1	String
ID	String

After creating the attributes, let’s add the transformation in a similar way to before.

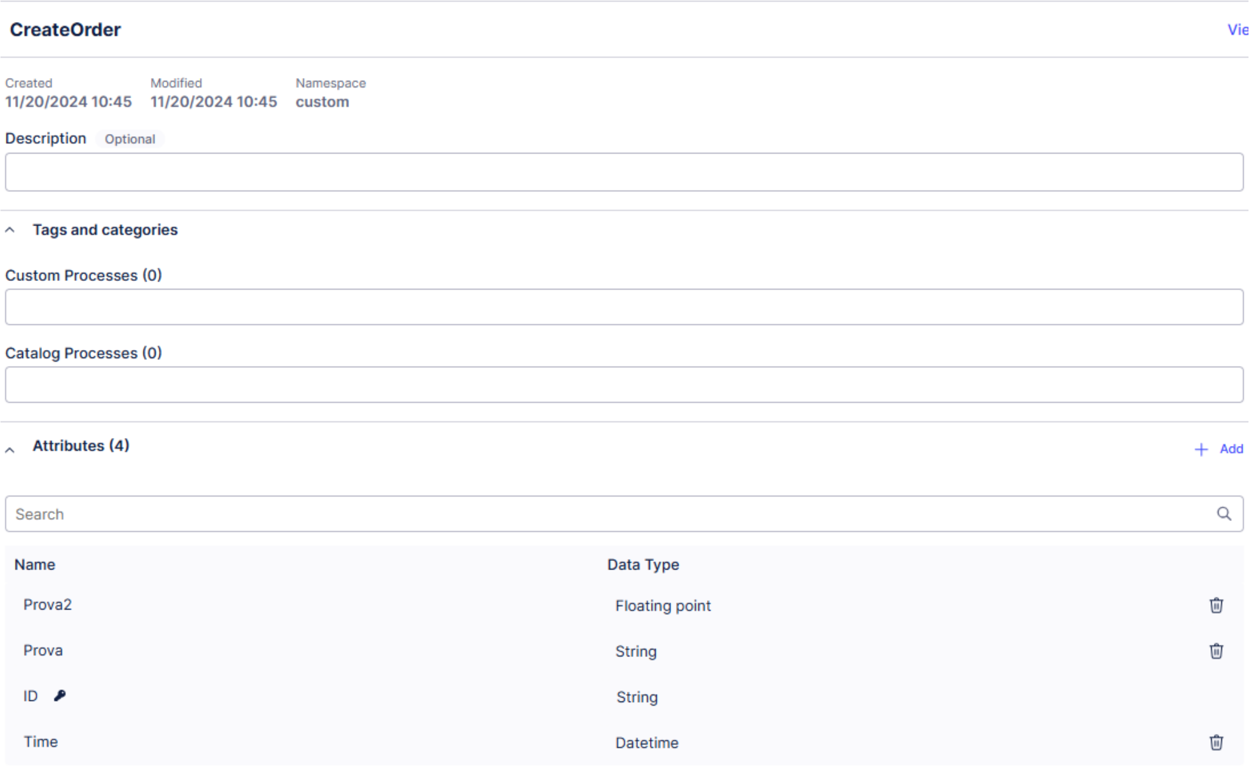


```

SELECT
    'i6' AS "ID",
    NULL AS "Oattr1",
    NULL AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'i7' AS "ID",
    NULL AS "Oattr1",
    NULL AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'i8' AS "ID",
    NULL AS "Oattr1",
    NULL AS "Oattr2"
FROM (SELECT 1) AS dummy
WHERE 1=1
```

Preview

STRING ID	STRING Oattr1	FLOAT Oattr2
i1	due	2
i2	tre	3
i3	quattro	4
i4	<null>	<null>
i5	<null>	<null>
i6	<null>	<null>
i7	<null>	<null>

Now, reach the “Event Types” part. Let’s add the event named “CreateOrder”.



CreateOrder

Created 11/20/2024 10:45

Modified 11/20/2024 10:45

Namespace custom

Description Optional

Tags and categories

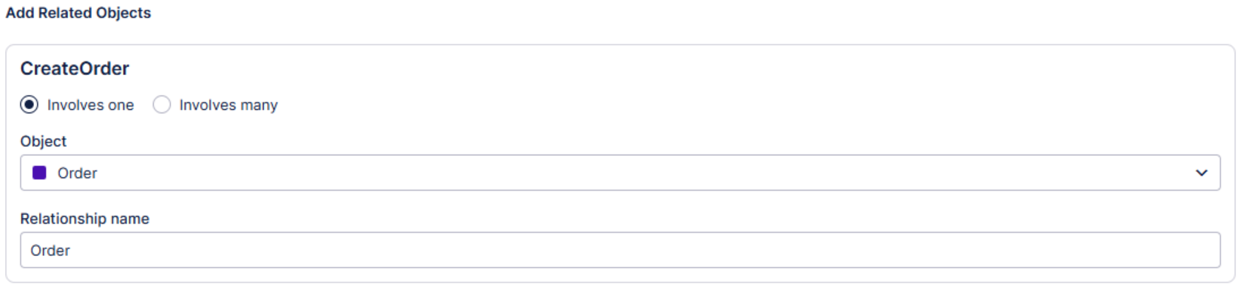
Custom Processes (0)

Catalog Processes (0)

Attributes (4)

Name	Data Type
Prova2	Floating point
Prova	String
ID	String
Time	Datetime

The next step is about adding the relationships to objects. Remind that this event type has 1:1 relation with Order, and 1:N relationship with Element. Use the "Relationships to objects" to add the two relationships.



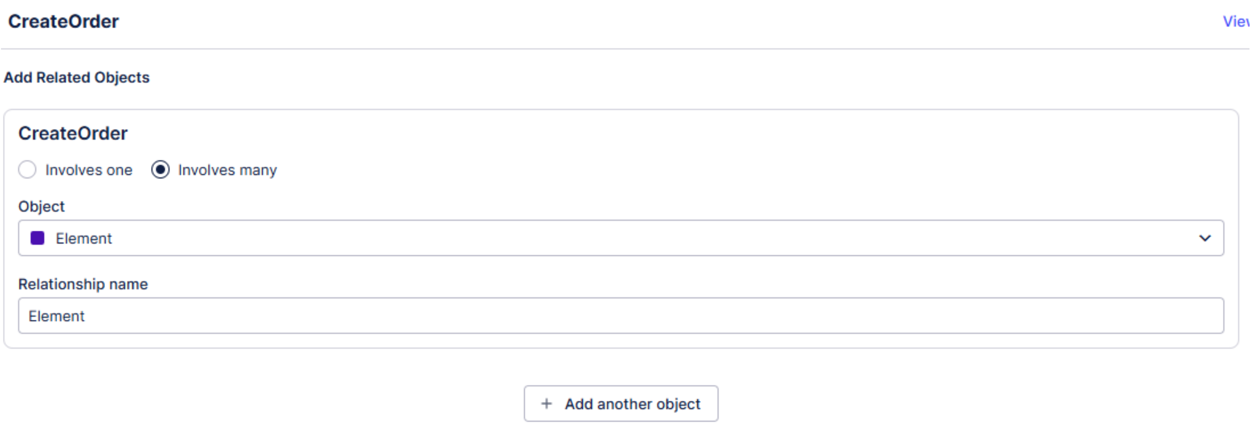
Add Related Objects

CreateOrder

● Involves one ○ Involves many

Object: Order

Relationship name: Order



CreateOrder

Add Related Objects

CreateOrder

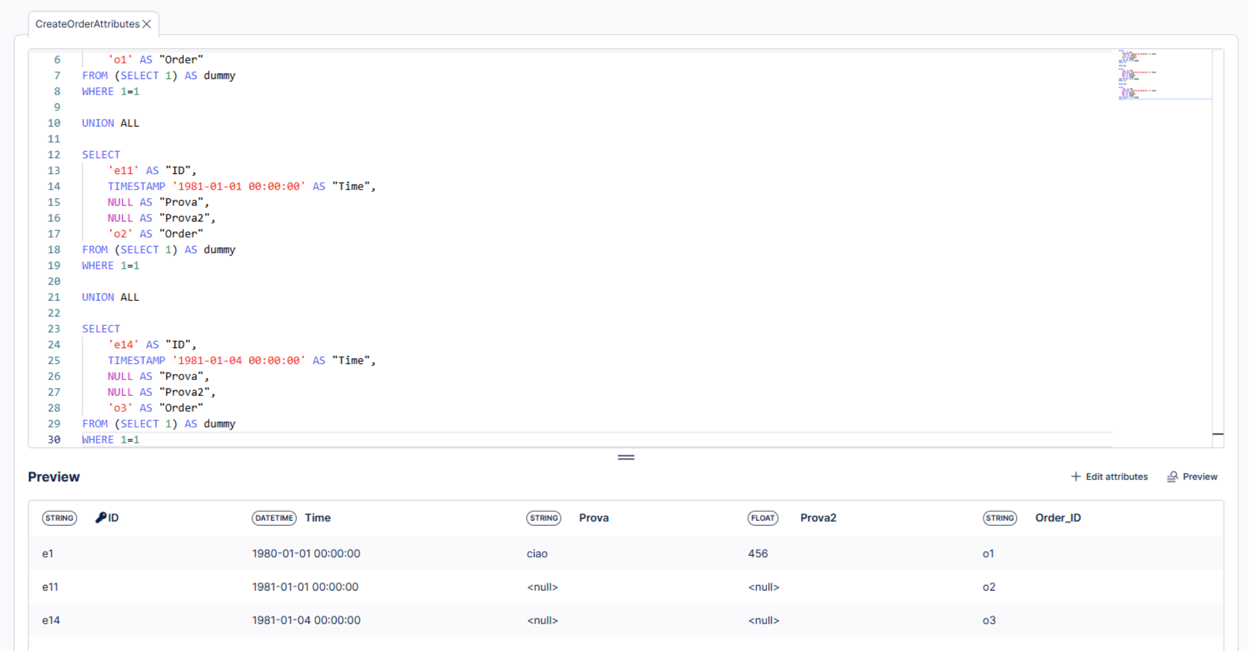
○ Involves one ● Involves many

Object: Element

Relationship name: Element

Then, reach the “Transformation” page and click “Add transformations”.

The first transformation that you need to add is the content of the event table, which also contains the “Order” as column (so, the 1:1 relationship is defined). Copy-paste the corresponding content of the SQL file, save and preview.

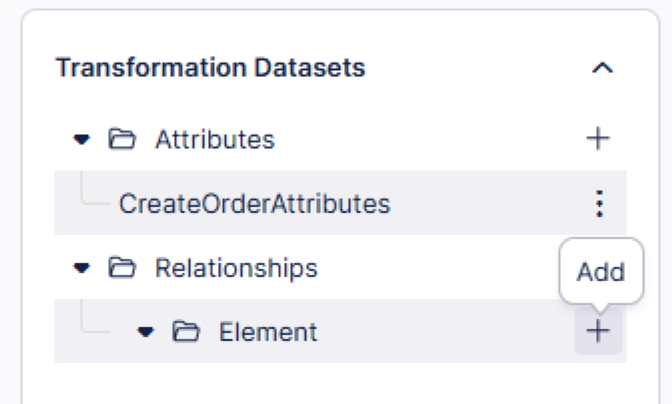


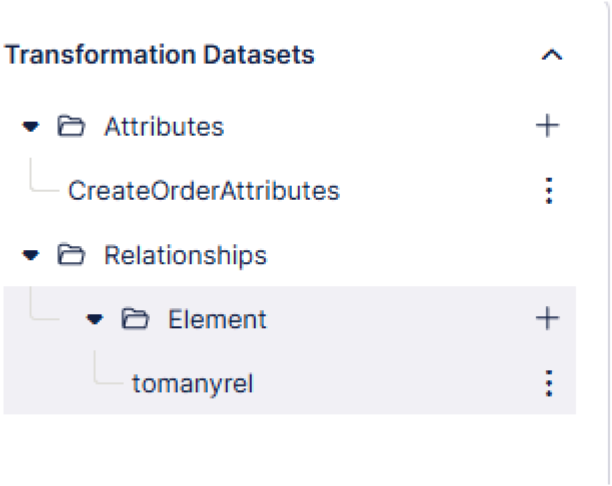
```sql
    'o1' AS "Order"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'e11' AS "ID",
    TIMESTAMP '1981-01-01 00:00:00' AS "Time",
    NULL AS "Prova",
    NULL AS "Prova2",
    'o2' AS "Order"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'e14' AS "ID",
    TIMESTAMP '1981-01-04 00:00:00' AS "Time",
    NULL AS "Prova",
    NULL AS "Prova2",
    'o3' AS "Order"
FROM (SELECT 1) AS dummy
WHERE 1=1
```

Preview

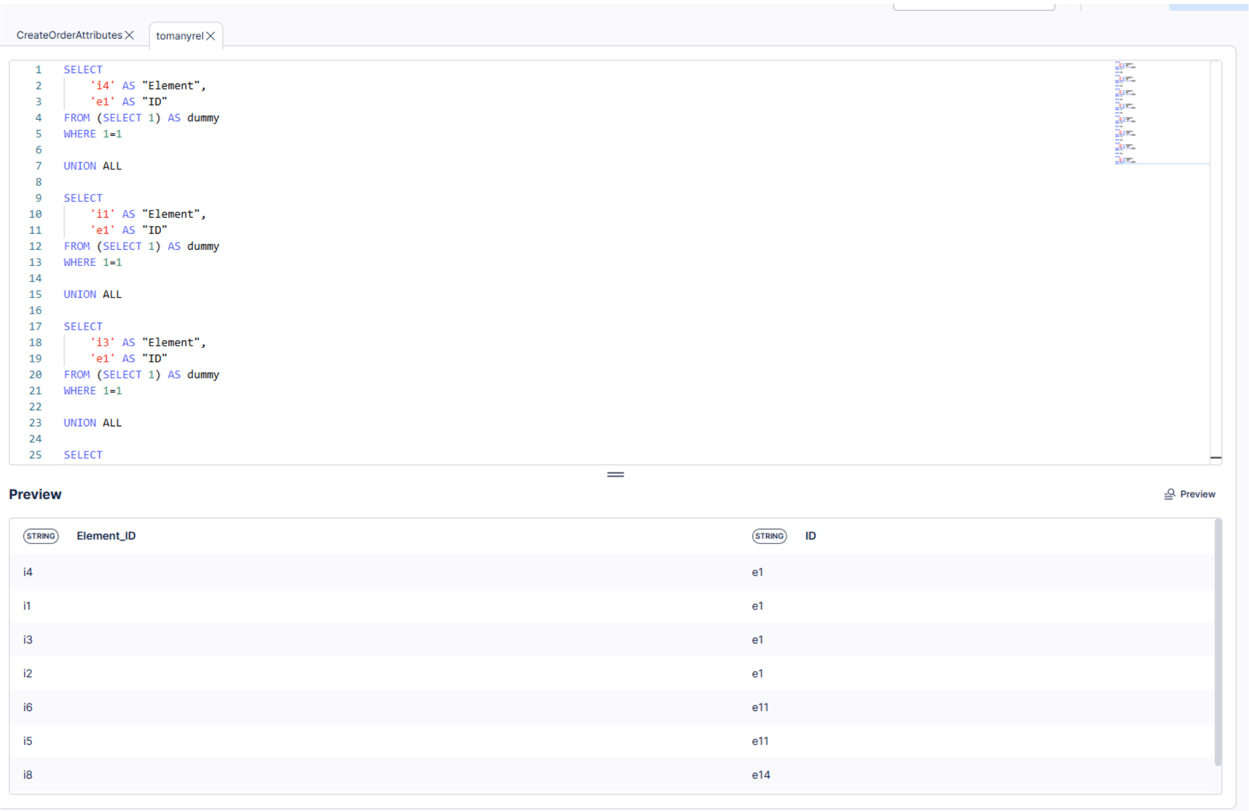
STRING ID	DATETIME Time	STRING Prova	FLOAT Prova2	STRING Order_ID
e1	1980-01-01 00:00:00	ciao	456	o1
e11	1981-01-01 00:00:00	<null>	<null>	o2
e14	1981-01-04 00:00:00	<null>	<null>	o3

The second transformation that you need to add is the 1:N relationship with Element. For this click the add button near “Relationships – Element” on the top left part, and give a name to the relationship:





Then, copypaste the corresponding content of the “relationship” SQL file, save, and preview.

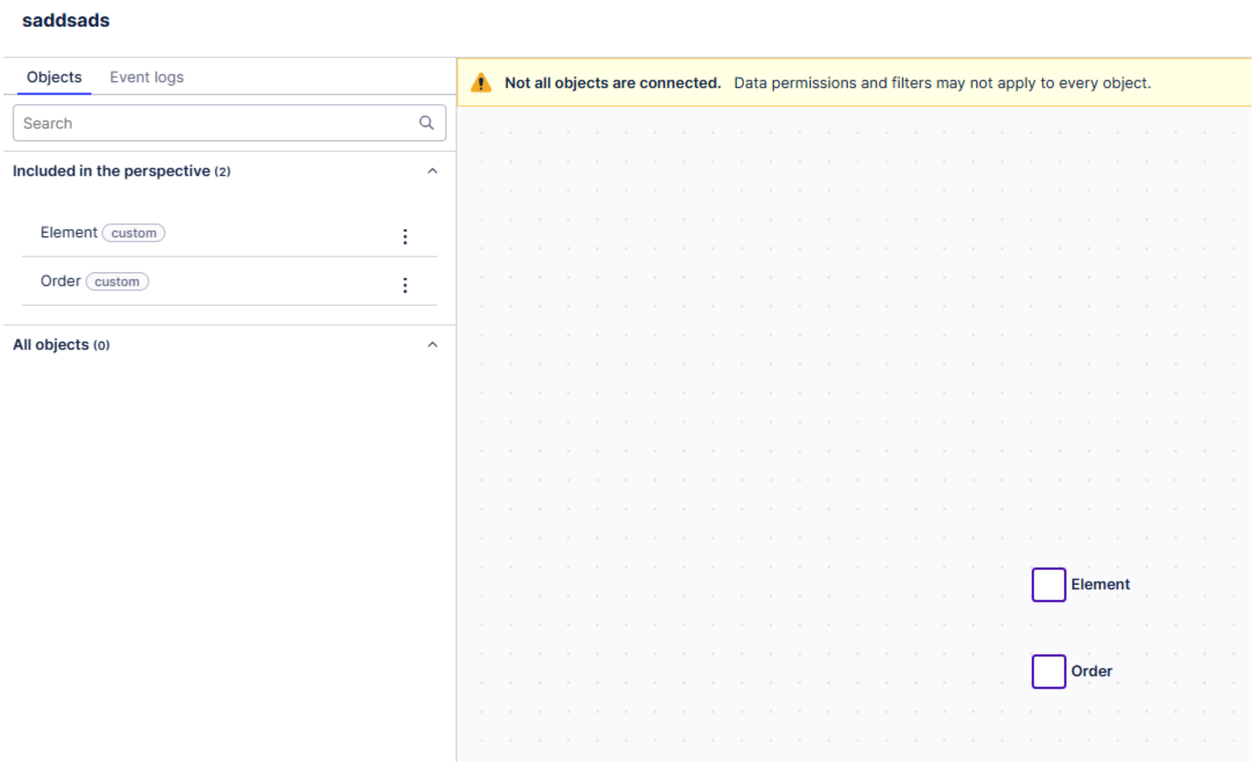


```
SELECT
    'i4' AS "Element",
    'e1' AS "ID"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'i1' AS "Element",
    'e1' AS "ID"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
    'i3' AS "Element",
    'e1' AS "ID"
FROM (SELECT 1) AS dummy
WHERE 1=1

UNION ALL

SELECT
```

Preview

STRING Element_ID	STRING ID
i4	e1
i1	e1
i3	e1
i2	e1
i6	e11
i5	e11
i8	e14

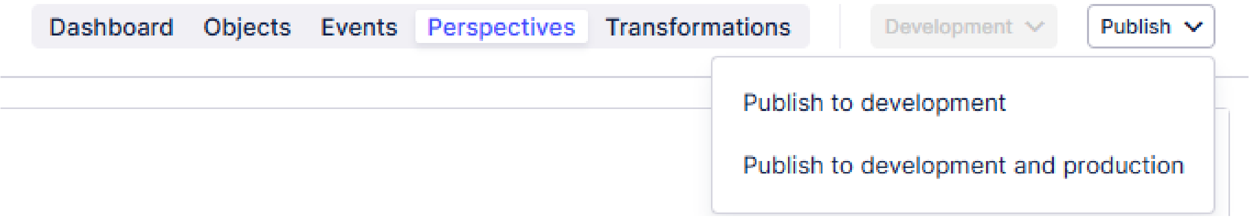
The data modeling part concludes with the creation of an analysis perspective. Reach the “Perspectives” section of the “Objects and Events” feature, click “Create Perspective”, and add the two object types.



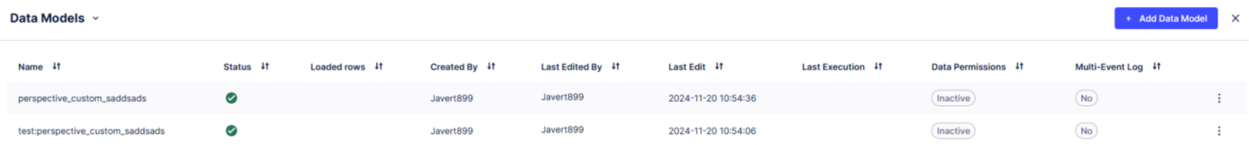
Not all objects are connected. Data permissions and filters may not apply to every object.

And then, click on “Save”.

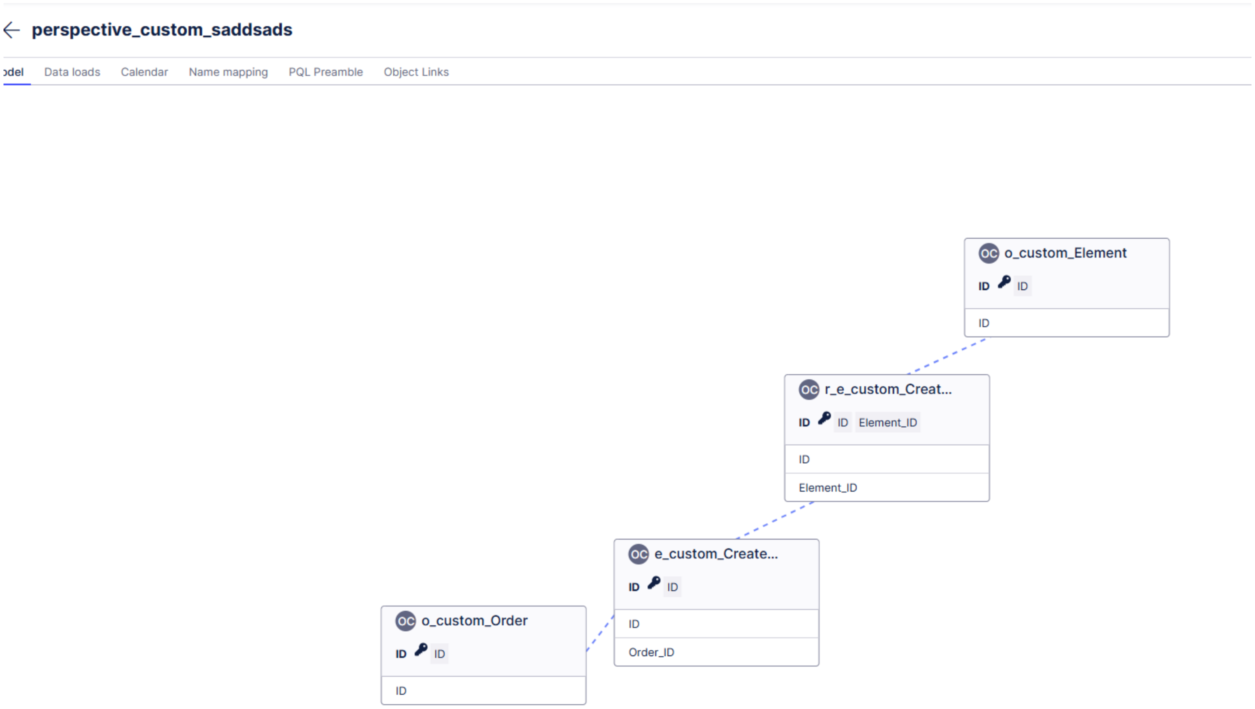
Then, click “Publish” on the top right corner, and select “Publish to development and production”.



When the publishing is done, reach the “Data – Data Integration” component of Celonis, click on the data pool, reach the data models, and click on the data name starting with “perspective_”:



Name	Status	Loaded rows	Created By	Last Edited By	Last Edit	Last Execution	Data Permissions	Multi-Event Log
perspective_custom_saddsads	✓		Javert899	Javert899	2024-11-20 10:54:36		Inactive	No
test:perspective_custom_saddsads	✓		Javert899	Javert899	2024-11-20 10:54:06		Inactive	No



At this point, trigger the load of the data model. Then, the data model is ready for usage in the “Studio” component of Celonis.