

Как навести порядок в двух эксабайтах данных?

Максим Гудзикевич, Кирилл Осинцев

Содержание

- 00 | Предпосылки
- 01 | Почему не взяли готовый каталог?
- 02 | Как написали ingest слой
- 03 | Как становимся точкой истины
- 04 | Выводы

00

Предпосылки





≈ 2 EiB

Особенности системы

1

Каждый реализует
партиции по-своему

Особенности системы

1

Каждый реализует
партиции по-своему

2

Каждый сам
определяет
гарантии
по своим данным

Особенности системы

1

Каждый реализует
партиции по-своему

2

Каждый сам
определяет
гарантии
по своим данным

3

Каждая команда
по-своему хранит
мета-информацию
и регуляризует
подсчёты

Xaoc

Хаос*

*на примере реальных проблем

Поставщик и потребитель



Поставщик

получил факап

и не знает, как собрать

всех пострадавших

Поставщик и потребитель



Поставщик

получил факап
и не знает, как собрать
всех пострадавших



Потребитель

хочет быстро найти
данные, помня лишь
часть названия колонки

Дата-инженер и аналитик



Дата-инженеру

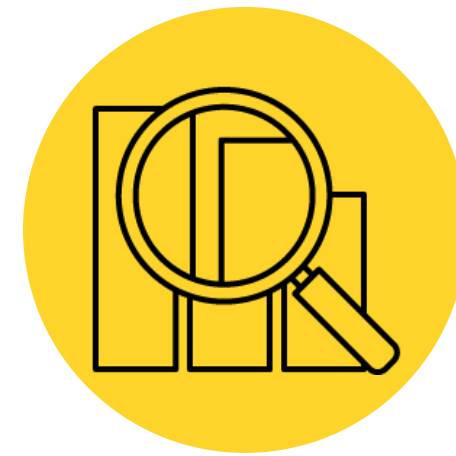
сдали на поддержку
не задокументированный
граф бывшего сотрудника

Дата-инженер и аналитик



Дата-инженеру

сдали на поддержку
не задокументированный
граф бывшего сотрудника



Аналитик

не понимает, какой
смысл у половины
колонок таблицы

Давайте делать

продукт

Давайте делать продукт

Нам важны:

1

Полнота
данных

Давайте делать продукт

Нам важны:

1

Полнота
данных

2

Полнота
связей между
данными

Давайте делать продукт

Нам важны:

1

Полнота
данных

2

Полнота
связей между
данными

3

Удобное
использование

Давайте делать продукт

Нам важны:

1

Полнота
данных

2

Полнота
связей между
данными

3

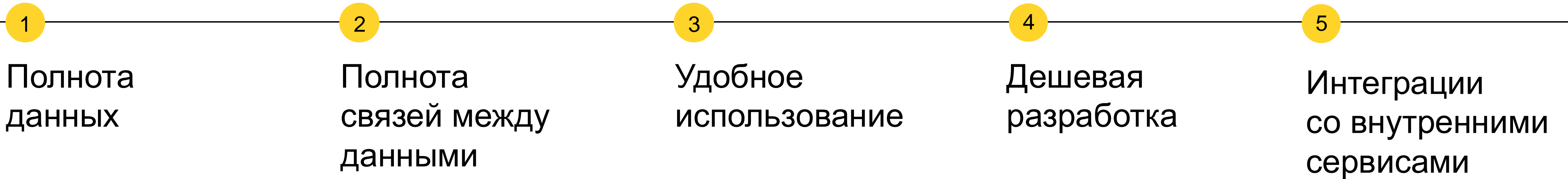
Удобное
использование

4

Дешевая
разработка

Давайте делать продукт

Нам важны:



01

Почему не взяли ГОТОВЫЙ каталог?

Что такое каталог данных?

Платформа, которая позволяет узнавать о данных компании.

› В каталоге данных хранится метainформация — описание данных, их структура, владельцы, атрибуты, связи с другими данными и другое.

The screenshot displays the Data Catalog interface for a Data Entity named 'Regular graph edges'. The page includes a search bar, navigation tabs (Description, Lineage, Schema, Users, Quality, Events, History), and a 'Description' section with text explaining the relationship between Data Entity and Process. A diagram illustrates the transformation from a 'concat_cooked' graph to a 'Run Graph'. Below the diagram is a 'Schema' table with columns for Name, Type, Title, and Description. The right sidebar shows 'Catalog meta' information such as Domain, System, Cluster, Path, and Owners, along with 'Catalog' tags.

Data Entity
Regular graph edges
slug: yt://hahn?path=//datacatalog/regular_graph/v1/edges/1d/*

Description Lineage Schema Users Quality Events History

Description

Рёбра регулярного графа после сжатия run graph до regular graph — описывают связи Data Entity с Process (регулярными). Есть примеры сырых процессов и данных из более детализированных графов (concat cooked и run graph)

Эти рёбра — то что непосредственно заливается в usage и по чему приходит понимание lineage данных в Каталоге .

Schema (5 of 24)

Name	Type	Title	Description
u_node_type	string	Data Entity type (system)	-
u_node_cluster	string	Data Entity cluster	-
u_node_id	string	Data Entity id (path)	-

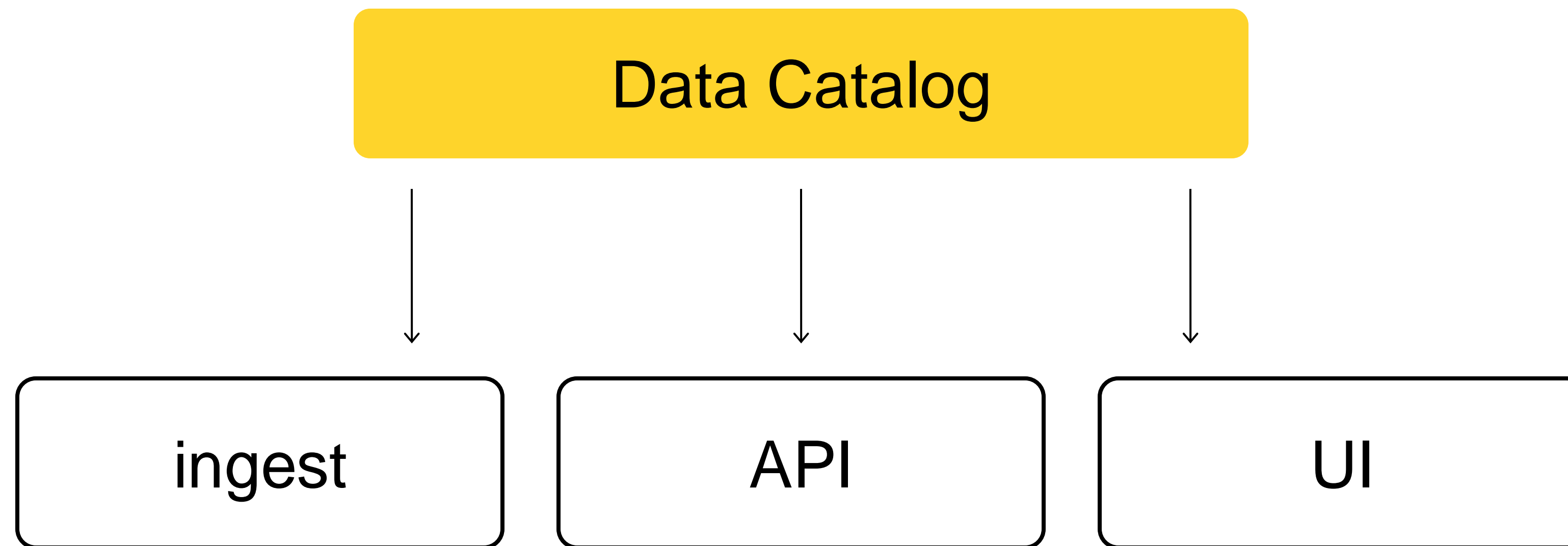
Catalog meta

Domain: datacatalog
System: yt
Cluster: hahn
Path: //datacatalog/regular_graph/v1/edges/1d/*
Owners: Максим Гудзикович

Catalog

Tags: regular_graph, datacatalog

Принцип работы



Известные решения



Что не так?



+



=



Что не так?



+



=



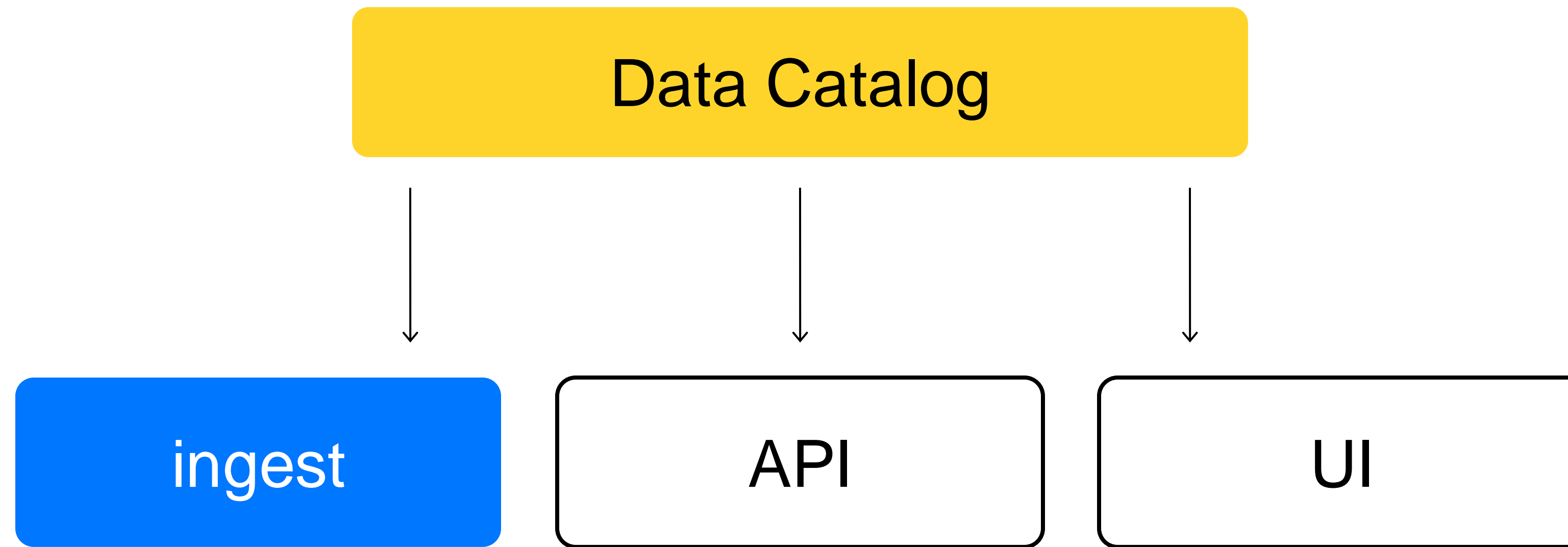
+



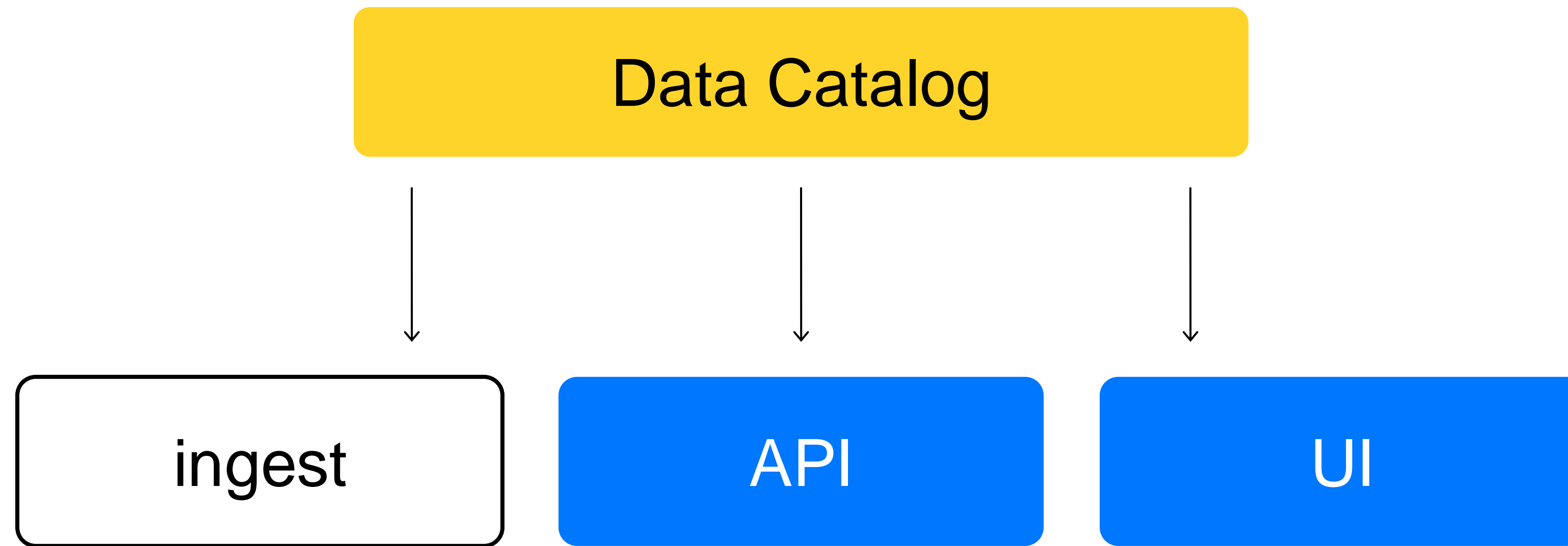
=



Принцип работы



Принцип работы



Почему свое?

Дорогая стоимость
подписок

Почему свое?

Дорогая стоимость
подписок

Сложные интеграции
со внутренними сервисами

Почему свое?

Дорогая стоимость
подписок

Сложные интеграции
со внутренними сервисами

Необходимость
команды разработчиков
для поддержки

02

Как написали ingest слой

Давайте сделаем по-простому

Напишем API, и попросим все команды пушить
информацию о расчете в конце операции



≈ **80m** nodes



≈ **80m** nodes

≈ **4k** accounts



≈ **80m** nodes

≈ **4k** accounts

≈ **15k+** owners

Все плохо?

1

Заставить всю
компанию пушить
в нас данные
не получится

Все плохо?

1

Заставить всю
компанию пушить
в нас данные
не получится

2

Но YTsaurus пишет
логи, которые мы
можем батчово
парсить

Какие логи у нас есть?

`//logs/yt-structured-scheduler-log` — планировщик MapReduce-операций

#	"operation_id"	"cluster"	"spec"	"timestamp"	"user"
0	"1200318-140402cd-3des3e8-9f096233"	"hahn"	<i>null</i>	1725872400	"some-user"
1	"1201219-040a0gce-abfs2e0-8f091204"	"hahn"	{}	1725872401	"some-user"
2	"1210418-140402cd-3des3e8-190a1gcr"	"hahn"	<pre>{ "ban_nodes_with_failed_jobs": true, "combine_chunks": true, "enable_legacy_live_preview": "false", "enable_prefetching_job_throttler": true, "ignore_job_failures_at_banned_nodes": true, "input_table_paths": ["//tmp/datacatalog/1201219-040a0gce-abfs2e0-8f091204"], "job_io": {}, "job_proxy_memory_digest": { "default_value": 0.5, "lower_bound": 0.05, "upper_bound": 2 }, "max_speculative_job_count_per_task": 3, "mode": "ordered", "output_table_path": "//tmp/datacatalog/1200318-140402cd-3des3e8-78sj1bd2", "pool": "bi-other", "started_by": { "command": [</pre>	1725872402	"some-user"

Какие логи у нас есть?

`//logs/yt-access-master-log` — записи о копировании, перемещении данных и особых видах чтения

#	"cluster"↓	"instant"↓	"category"	"destination_path"	"path"	"user"
0	"hahn"	1723194001	"Copy"	"//tmp/some-user/5eb63bbbe01eed093cb22bb8f5acdc3"	"//datacatalog/crawler/v2/exported"	"some-user"
1	"hahn"	1725872400	"Access"	<i>null</i>	"//datacatalog/crawler/v2/exported"	"some-user"
2	"hahn"	1725872400	"Access"	<i>null</i>	"//datacatalog/crawler/v2/2023-01-01"	"some-user"
3	"hahn"	1725872401	"Move"	"//datacatalog/crawler/v2/2023-01-01"	"//datacatalog/crawler/v2/2024-01-01"	"some-user"

Какие логи у нас есть?

`//snapshot-exports` — записи обо всех узлах YTsaurus

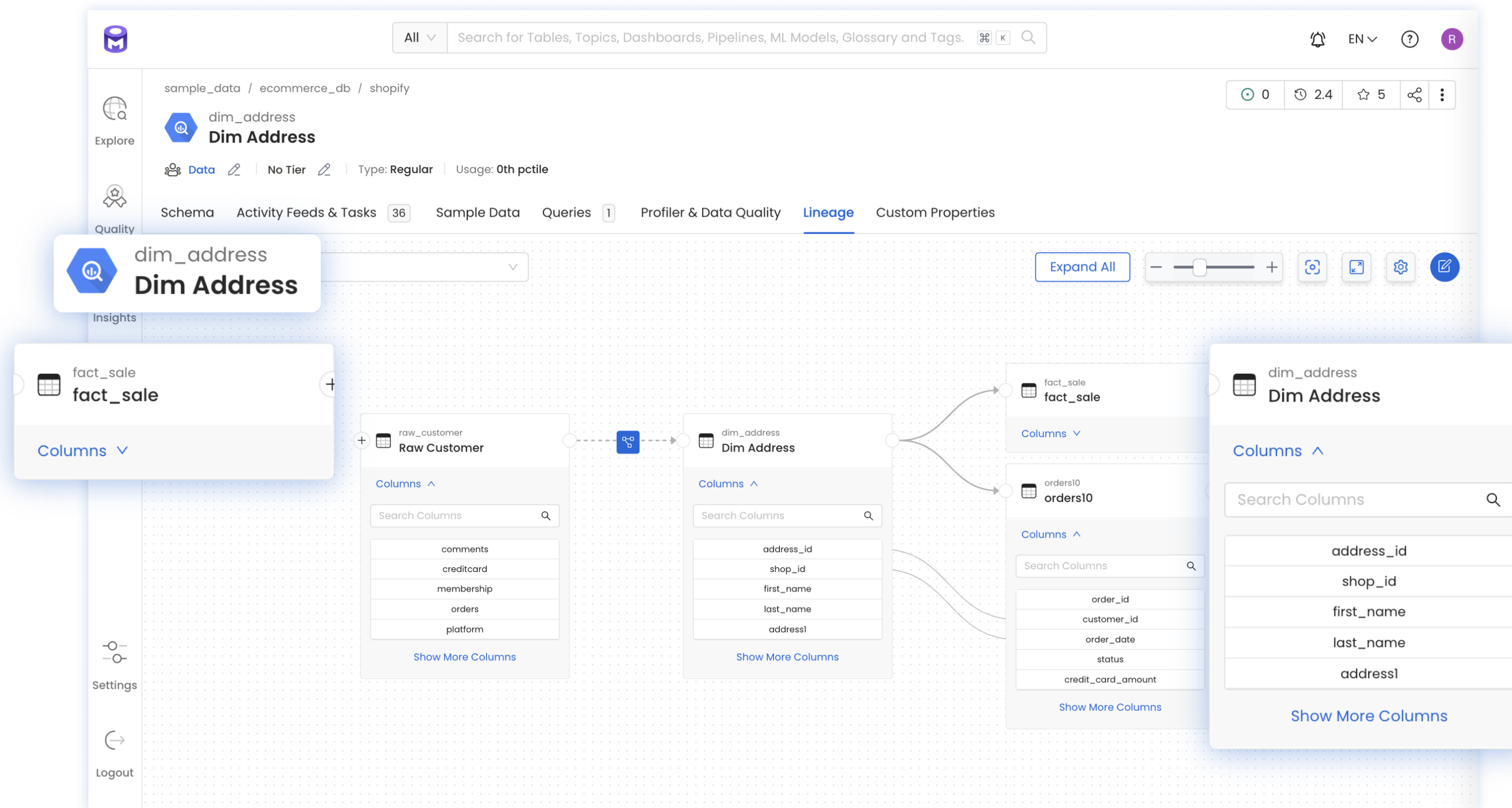
#	"path" ↓	"account"	"dynamic"	"row_count"	"type"
0	"//datacatalog"	"datacatalog"	<i>null</i>	<i>null</i>	"map_node"
1	"//datacatalog/crawler"	"datacatalog"	<i>null</i>	<i>null</i>	"map_node"
2	"//datacatalog/crawler/metadata"	"datacatalog"	<i>null</i>	<i>null</i>	"map_node"
3	"//datacatalog/crawler/metadata/2024-09-08"	"datacatalog"	false	23709	"table"
4	"//datacatalog/crawler/metadata/2024-09-09"	"datacatalog"	false	25112	"table"
5	"//datacatalog/crawler/metadata/cached"	"datacatalog"	true	7213	"table"
6	"//datacatalog/crawler/metadata/last"	"datacatalog"	<i>null</i>	<i>null</i>	"link"
7	"//snapshot-exports"	"sys"	false	60235178	"table"
8	"//tmp/some-user/5eb63bbbe01eed093cb22bb8f5acdc3"	"tmp"	false	568100	"table"

99% операций и
узлов YTsaurus

Покрывается при работе с низкоуровневыми логами

А ЧТО ХОТИМ В ИТОГЕ?

Граф пользовательских данных
и процессов со связями между ними



Закрепим различия

Операция (YTsaurus) — единица обработки данных
— Map/Reduce/MapReduce/Sort/Merge и т. п.

Пользовательский процесс — логика
по преобразованию входных данных в выходные

Узел (YTsaurus) — директория, таблица, файл и др.

Пользовательские данные — полезные для юзеров
таблицы

Логи Ytsaurus → Cooked логи

- › Аккуратно парсим внешние логи (raw-слой типичного DWH)
- › Сохраняем удобный формат лога для датаинженера
- › Если внешний лог имеет TTL — у себя можно держать историю

Cooked логи → Concat Graph

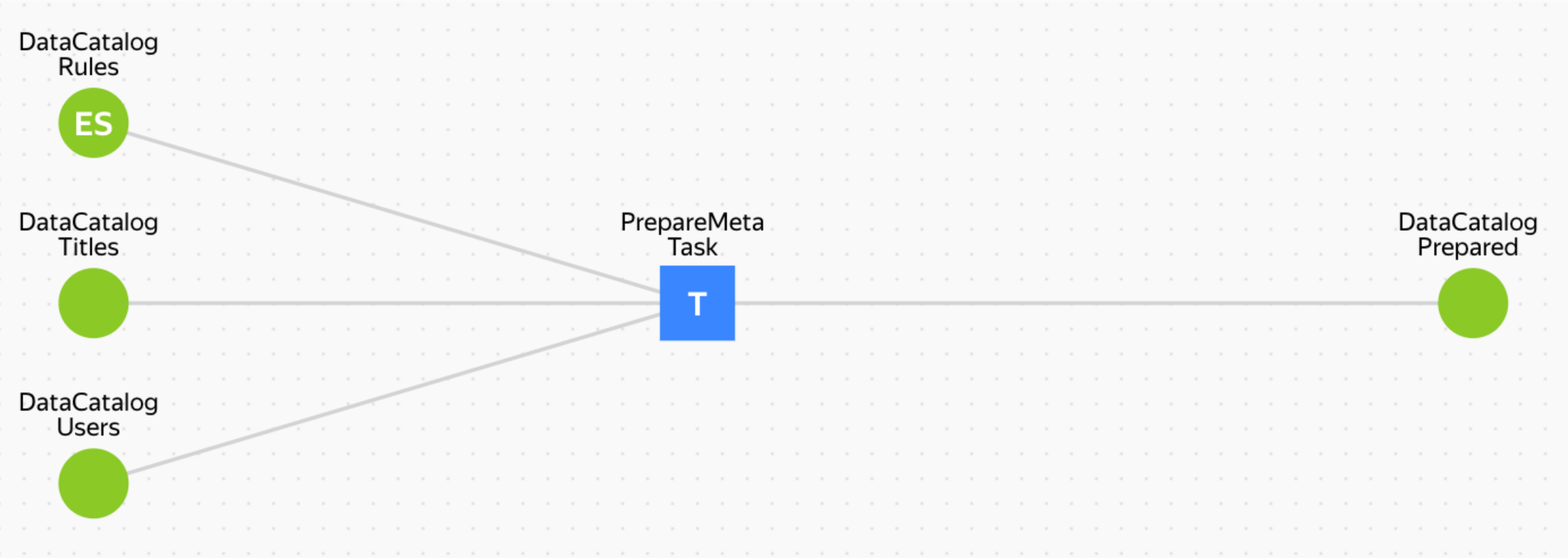
Преобразуем данные в бизнесовое представление датакаталога — данные и процессы:

- › Вершины данных и вершины процессов составляют ориентированный чередующийся граф
- › Все вершины характеризуются ключом из трех значений: тип системы, кластер и id
- › Сам граф храним в виде таблицы вершин и таблицы рёбер

Получаем детализированный граф того, как данные на YTsaurus связаны друг с другом через различные процессы. Следующая задача — **упростить этот граф.**

Concat Graph: ожидание

После предыдущего этапа получаем граф пользовательских процессов

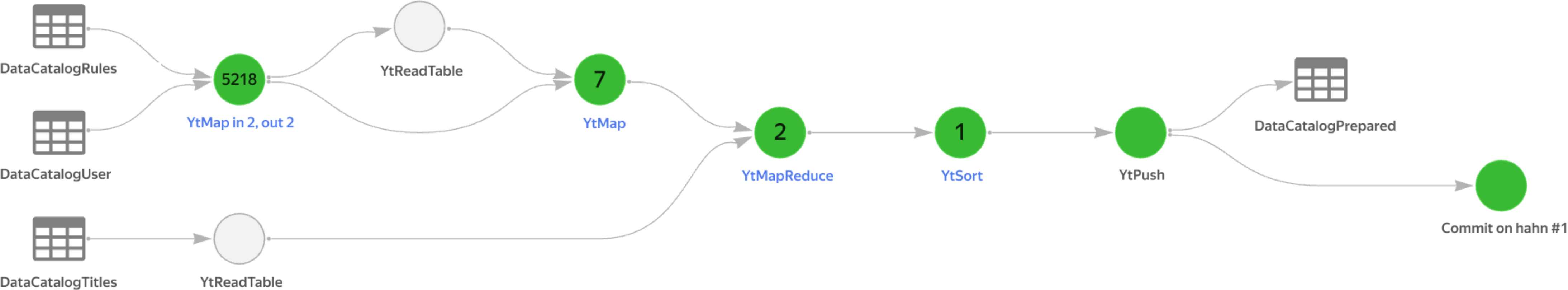


Concat Graph: реальность

Исходные логи содержат информацию о промежуточных операциях YTsaurus, а не о пользовательском процессе

#	"authenticated_user"	"cluster"	"is_source"	"operation_id"	"path"
0	"some-user"	"hahn"	false	"2b1a5-4edd66-3fe0191-1a242e50"	"//tmp/some-user/5eb63bbbe01eed093cb22bb8f5acdc3"
1	"some-user"	"hahn"	false	"2b2a5-4edd66-3fe0191-1a242e51"	"//tmp/some-user/5e44a9ce239320776166d40c5579607e"
2	"some-user"	"hahn"	false	"173a5-4edd66-3f22191-1a2aae53"	"//tmp/some-user/346bfd800179b9a085d5488d1fc85043"
3	"some-user"	"hahn"	false	"2b2a5-4edd66-3fe0191-1a242e52"	"//tmp/some-user/c34ea3d34e693b62b9b1b545d221b900"
4	"some-user"	"hahn"	false	"2b2a5-4edd66-3fe0191-1a242e52"	"//tmp/some-user/89c2bd30f28a1eccb3fa365739f39ceb"
5	"some-user"	"hahn"	false	"172a5-4edd66-3f22191-1a2ane53"	"//tmp/some-user/c21c00715af9dc9c75cd27fac5b442ce"
6	"some-user"	"hahn"	false	"174a5-4edd66-3f22191-1a2ane53"	"//tmp/some-user/097cedd3e568ad3ccf73a03e543f5add"
7	"some-user"	"hahn"	false	"172a5-4edd66-3f22191-1a2a2e53"	"//tmp/some-user/547fab8c790f5aad40e7e23d8da4ca15"
8	"some-user"	"hahn"	false	"2b2a5-4edd66-3fe0191-1a242e55"	"//tmp/some-user/120ff38c444d575367a656593dad76a5"
9	"some-user"	"hahn"	false	"2b5a5-4edd66-3fe0191-1a242e55"	"//tmp/some-user/1944dd28a7392d4aff685420e6e04c8b"

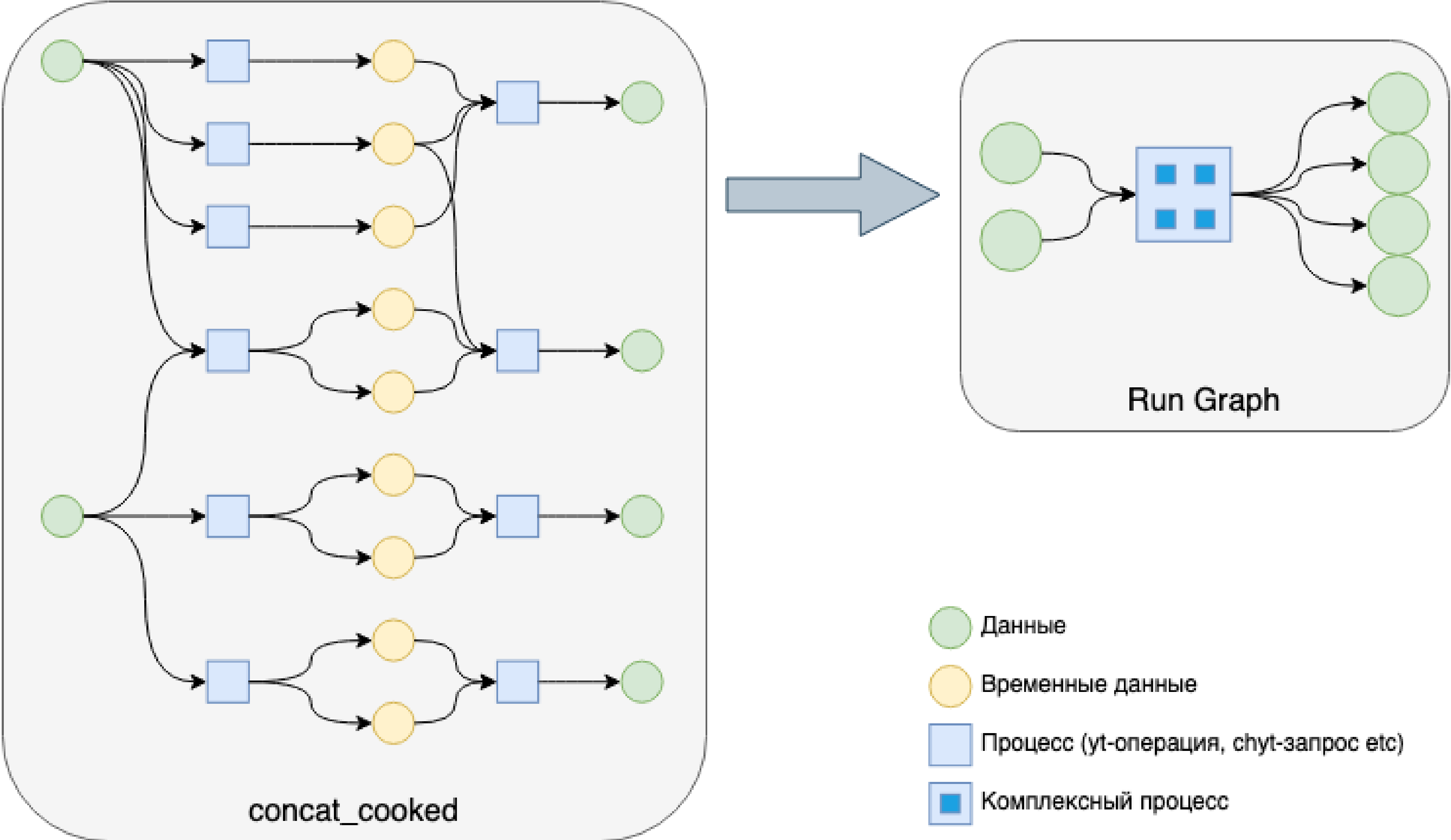
Concat Graph: реальность



Concat Graph → Run Graph

- › Решение — пишем алгоритм кластеризации, который восстанавливает прообраз процесса пользователя
- › “Временность таблицы” (например все в //tmp) определяет набор эвристик
- › Алгоритм для каждого набора (user, cluster) превращает компоненты связности в единый процесс

Concat Graph → Run Graph



Run Graph → Regular Graph

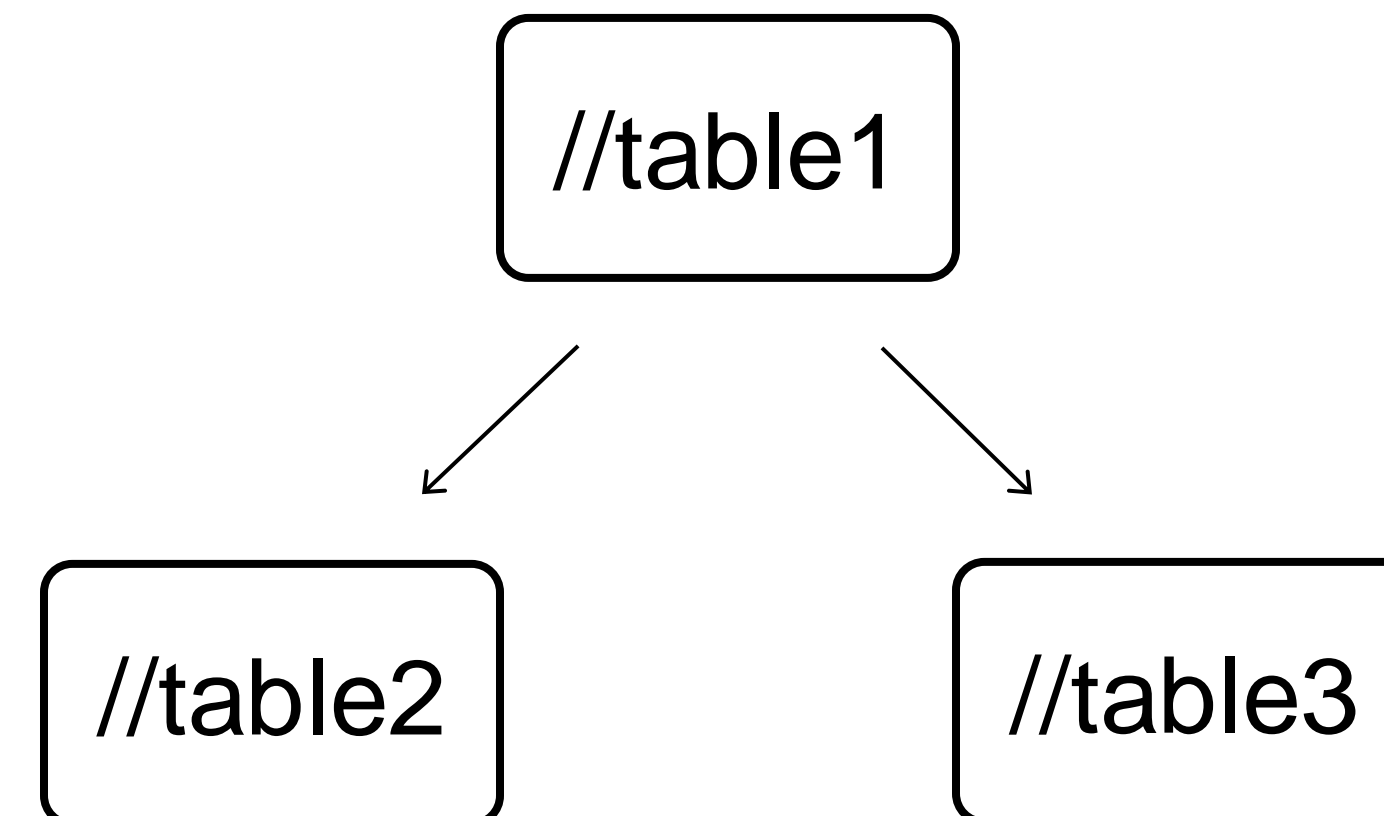
- › Подклеиваем информацию из других логов, не требующих кластеризации (например, из //snapshot-exports)
- › Группируем информацию о данных и процессах

#	"node_system"↓	"node_cluster"↓	"node_id"↓	"_knowledge_source"	"yt_type"
0	"yt"	"hahn"	"//datacatalog/crawler/v2/exported"	["crawler_graph"]	"table"
1	"yt"	"hahn"	"//datacatalog/crawler/v2/files"	["crawler_graph"]	"table"
2	"yt"	"hahn"	"//datacatalog/crawler/v2/nodes"	["crawler_graph"]	"table"
3	"yt"	"hahn"	"//datacatalog/crawler/v2/scheduler-cooked"	["crawler_graph", "yt_snapshots"]	"table"
4	"yt"	"hahn"	"//datacatalog/crawler/v2/usage"	["crawler_graph"]	"table"
5	"yt"	"hahn"	"//datacatalog/logos/graph"	["logos"]	"table"
6	"yt"	"hahn"	"//datacatalog/regular/joined"	["crawler_graph", "yt_snapshots"]	"table"

Результаты

Получили
максимально
полный регулярно
обновляемый граф

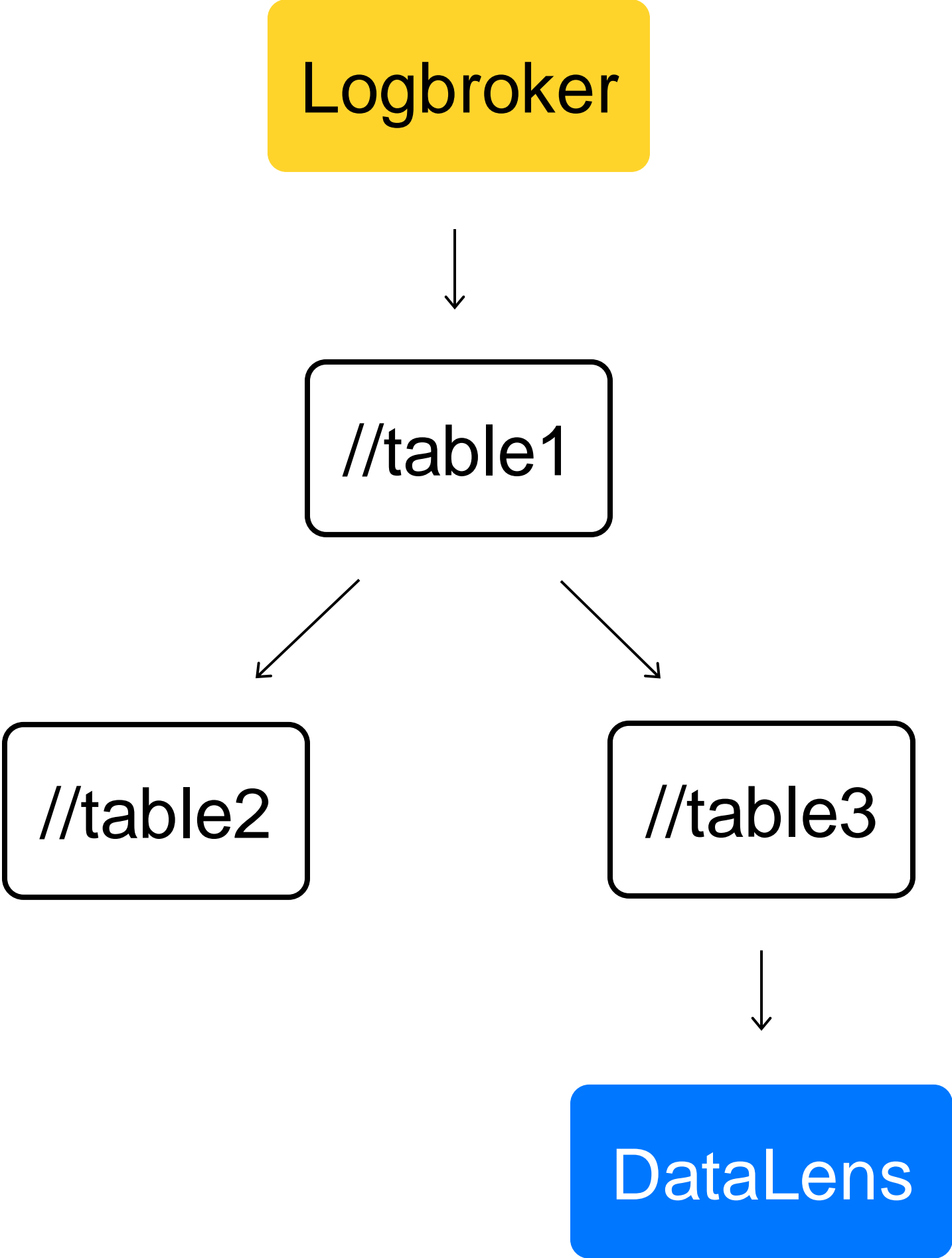
1



Результаты

Отмасштабировали граф на логи DataLens и Logbroker

2

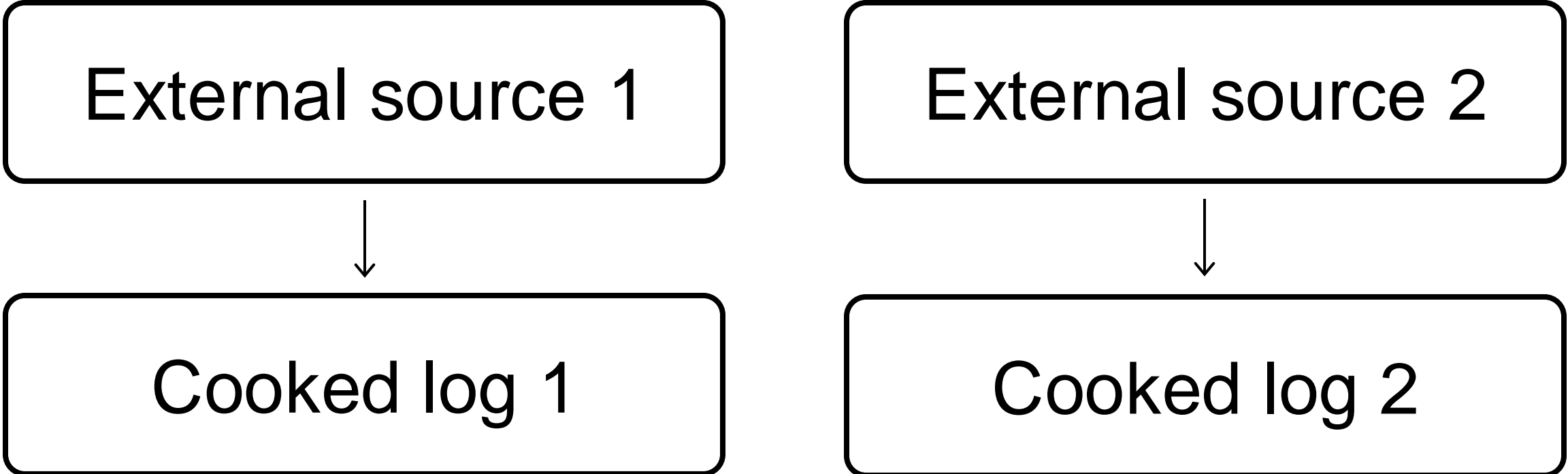


Итоговый пайплайн

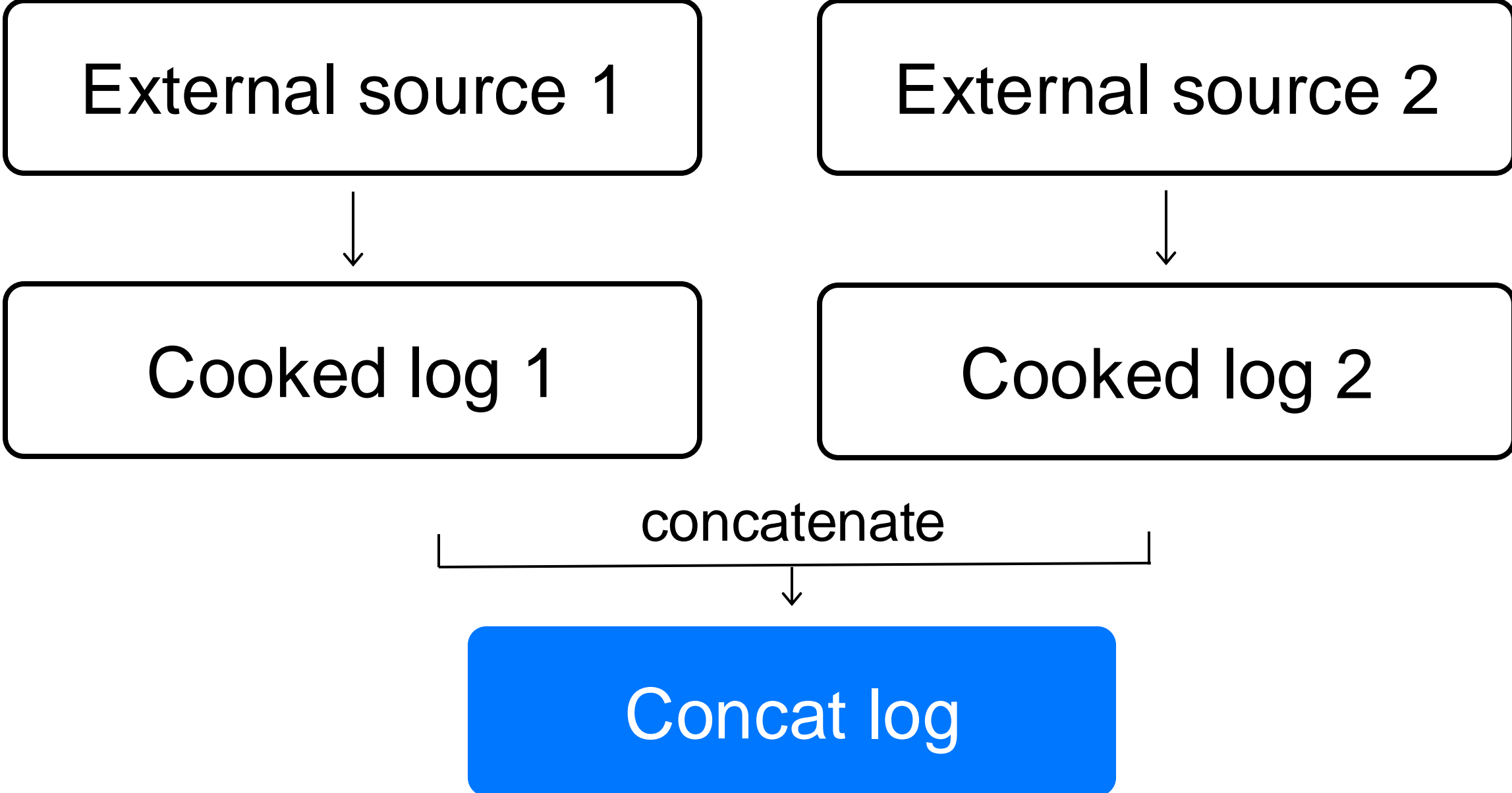
External source 1

External source 2

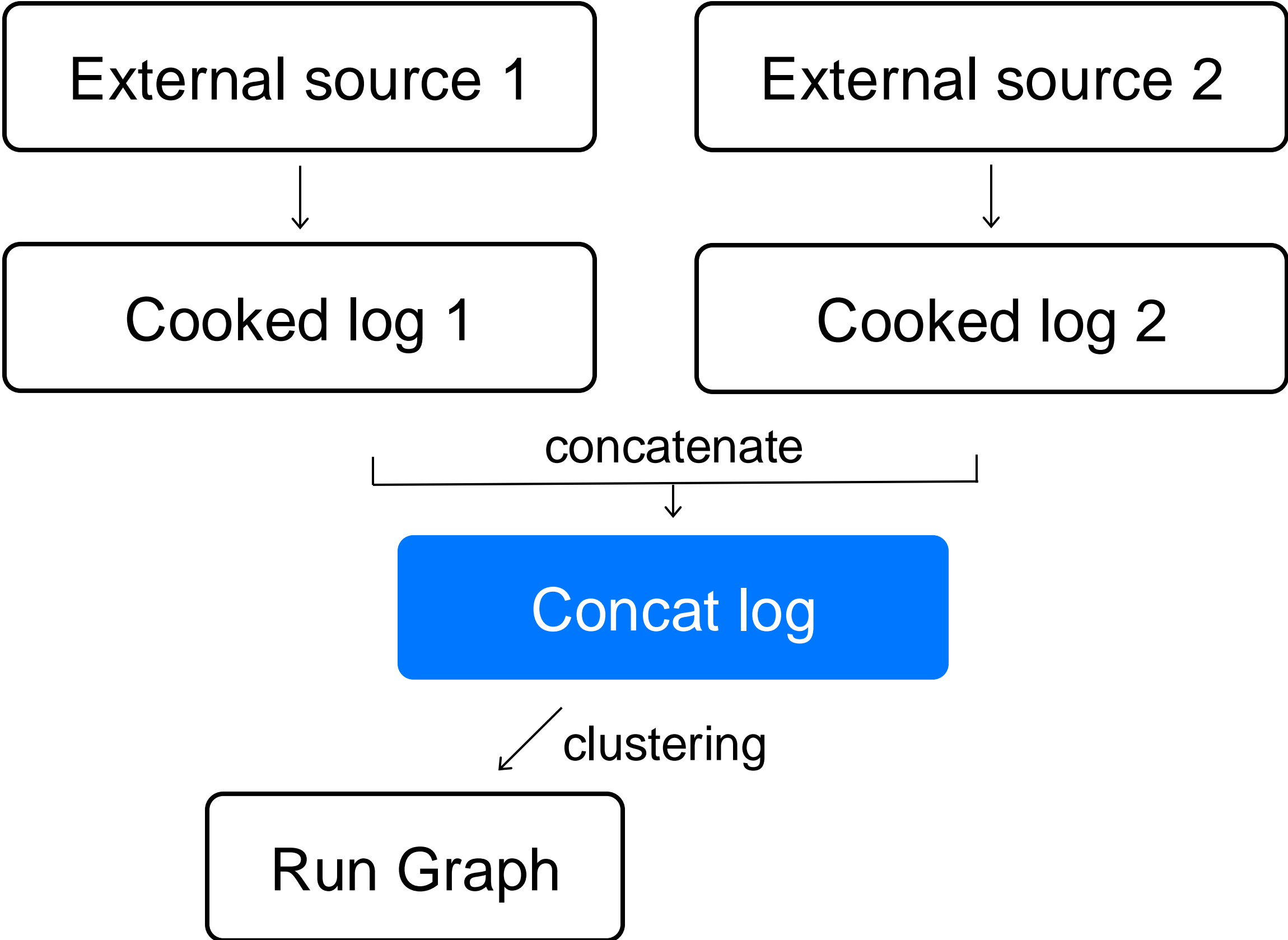
Итоговый пайплайн



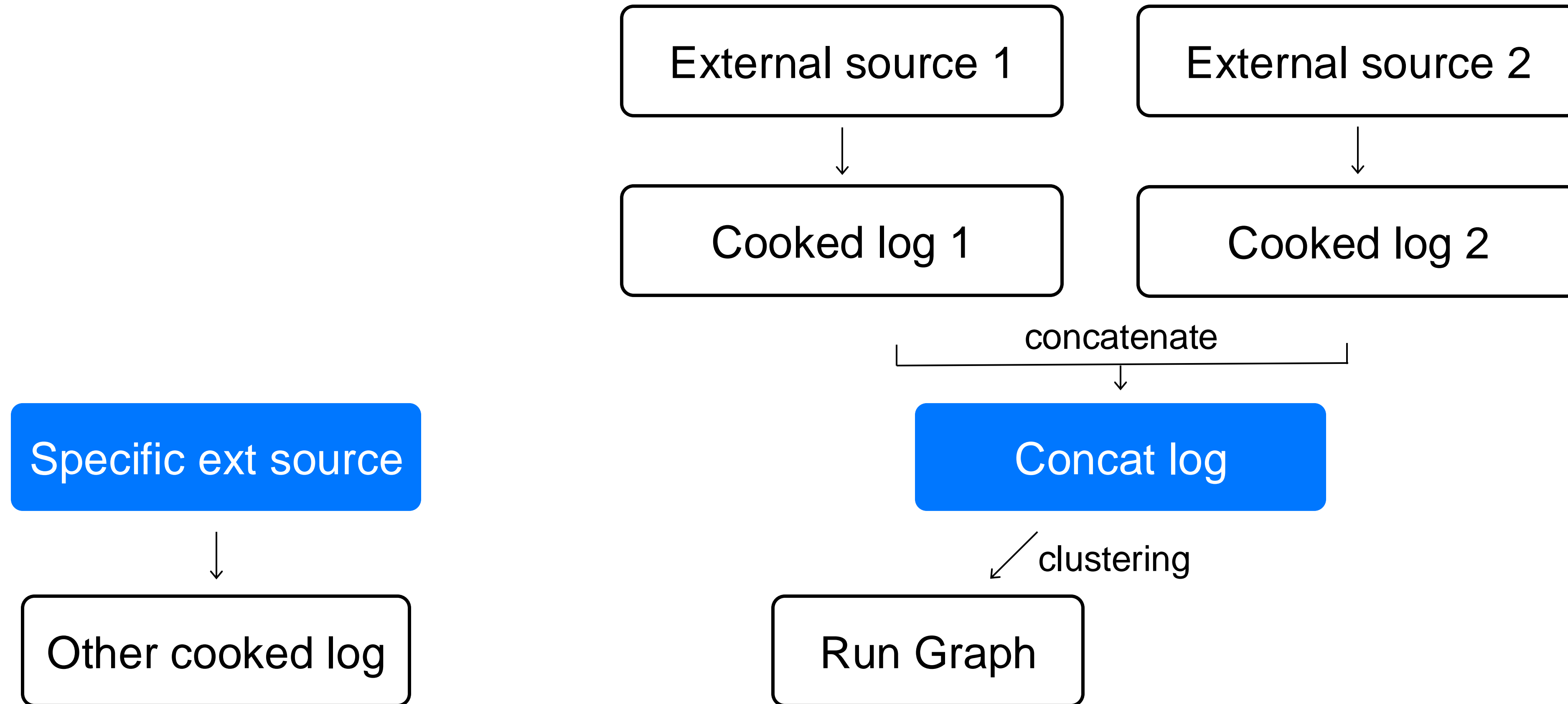
Итоговый пайплайн



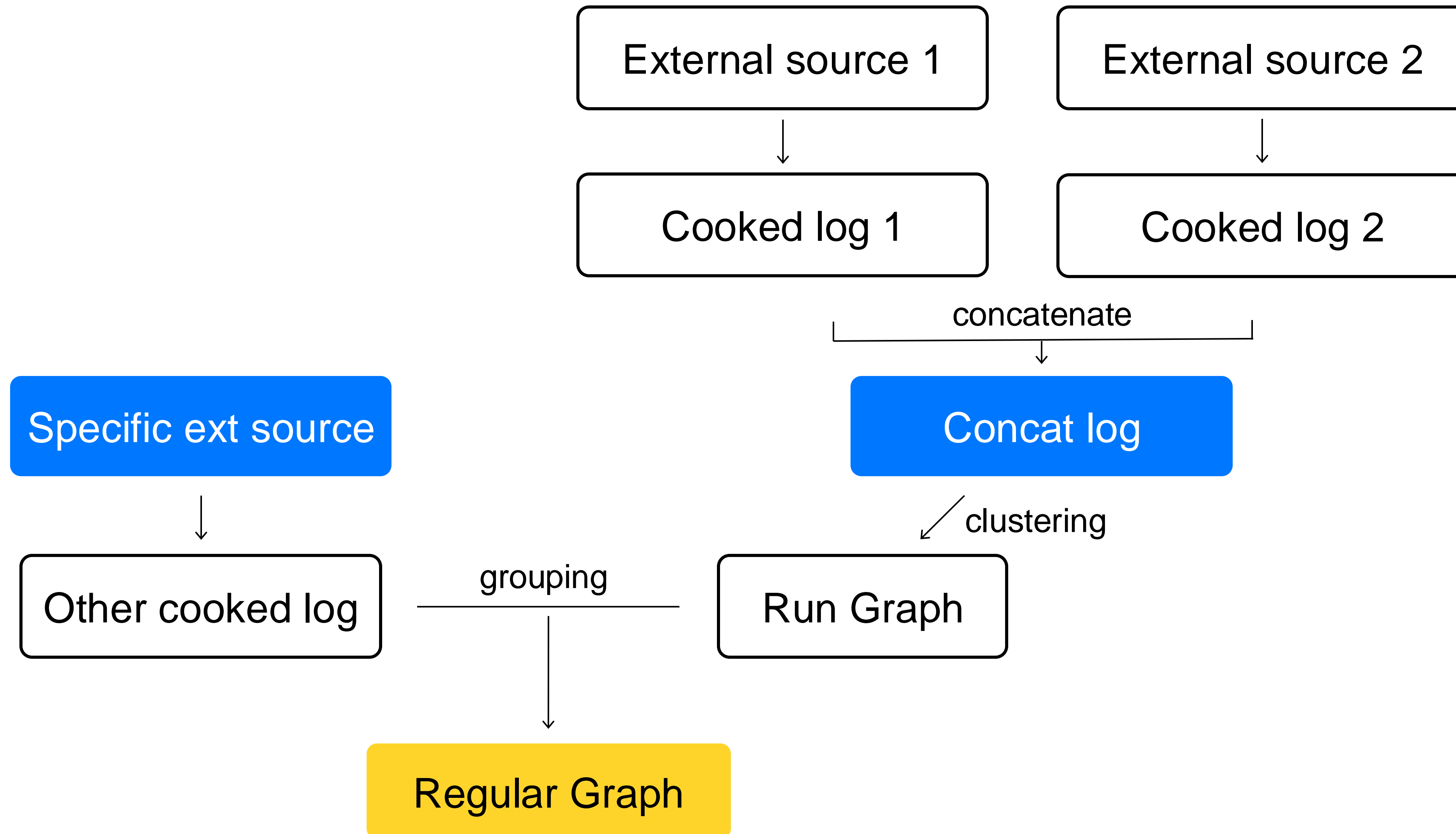
Итоговый пайплайн



Итоговый пайплайн



Итоговый пайплайн



**Забыли
упомянуть про
пару нюансов**

Логи YTsaurus → Cooked логи

- › Аккуратно обрабатываем внешние логи
- › Сохраняем удобный формат лога для датаинженера
- › Если внешний лог имеет TTL — у себя можно держать историю
- › В нашей системе есть символические ссылки, что с ними делать?

<input type="checkbox"/>	🔗 latest view link → 2024-09-08T12:00:00		statbox-cube	-
<input type="checkbox"/>	📅 2024-09-08T12:00:00	TTL	statbox-cube	59.43 GiB
<input type="checkbox"/>	📅 2024-09-08T09:00:00	TTL	statbox-cube	59.52 GiB
<input type="checkbox"/>	📅 2024-09-08T06:00:00	TTL	statbox-cube	59.60 GiB
<input type="checkbox"/>	📅 2024-09-08T03:00:00	TTL	statbox-cube	59.73 GiB
<input type="checkbox"/>	📅 2024-09-08T00:00:00	TTL	statbox-cube	59.64 GiB
<input type="checkbox"/>	📅 2024-09-07T21:00:00	TTL	statbox-cube	59.60 GiB
<input type="checkbox"/>	📅 2024-09-07T18:00:00	TTL	statbox-cube	59.50 GiB
<input type="checkbox"/>	📅 2024-09-07T15:00:00	TTL	statbox-cube	59.75 GiB
<input type="checkbox"/>	📅 2024-09-07T12:00:00	TTL	statbox-cube	59.70 GiB

Символические ссылки

1

У одних
и тех же данных
может быть
несколько путей

Символические ссылки

1

У одних
и тех же данных
может быть
несколько путей

2

Пользователи
YTsaurus знают
и используют
тот, который
удобнее для них

Символические ссылки

1

У одних
и тех же данных
может быть
несколько путей

2

Пользователи
YTsaurus знают
и используют
тот, который
удобнее для них

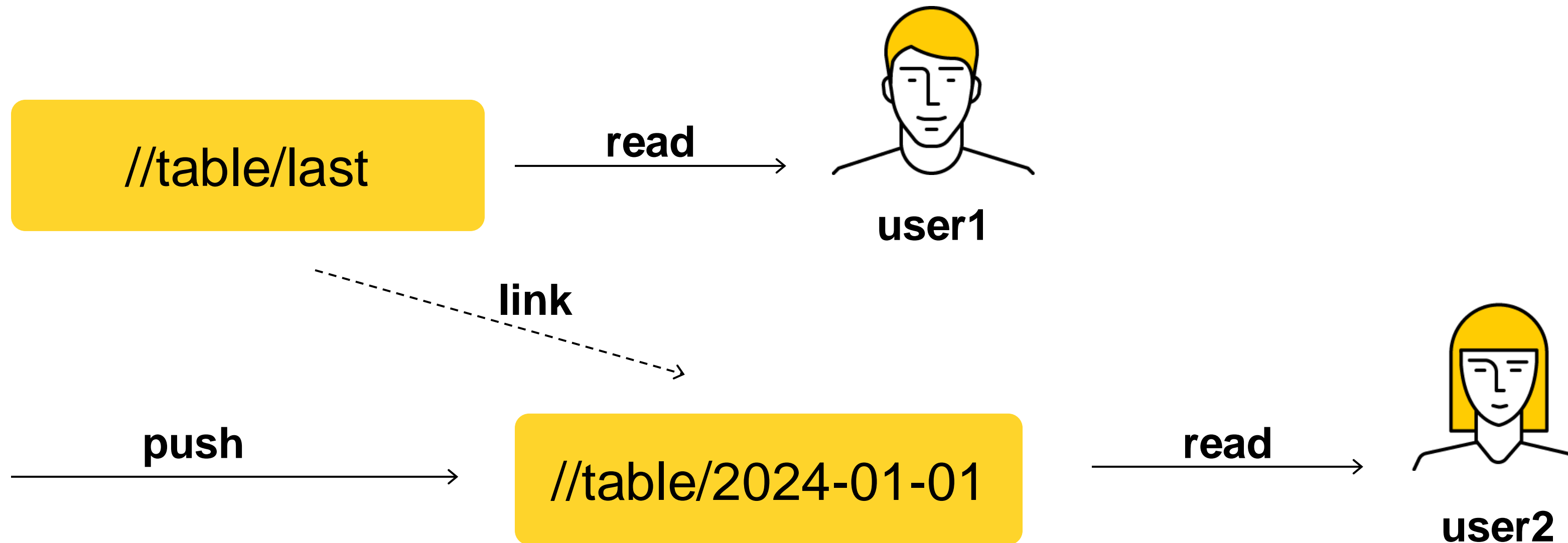
3

Данные в логах могут
быть описаны
по любому из путей

Символические ссылки

› **Бизнесово** — сущность одна

› **Технически** — сущностей несколько



Как избавились от ссылок?

1

Считаем,
что данные всегда
имеют единственный
верный путь

Как избавились от ссылок?

1

Считаем,
что данные всегда
имеют единственный
верный путь

2

Написали библиотеку
для разрешения
символических ссылок
в путях на основе
снимотов YTsaurus

Как избавились от ссылок?

1

Считаем,
что данные всегда
имеют единственный
верный путь

2

Написали библиотеку
для разрешения
символических ссылок
в путях на основе
снимотов YTsaurus

3

Для каждого пути
собираем примеры
путей через
символическую
ссылку для поиска
(но об этом позже)

Concat Graph → Run Graph

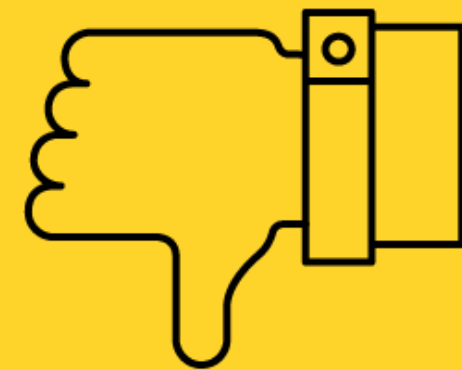
- › Решение — пишем алгоритм кластеризации, который восстанавливает прообраз процесса пользователя
- › “Временность таблицы” (например все в //tmp) определяет набор эвристик
- › Алгоритм для каждого набора (user, cluster) превращает компоненты связности в единый процесс
- › А что делать с процессами на границе дней в случае батчовой обработки?

Учитываем процессы на границе

Кластеризуем данные “внахлест”
— то есть с окном в два дня

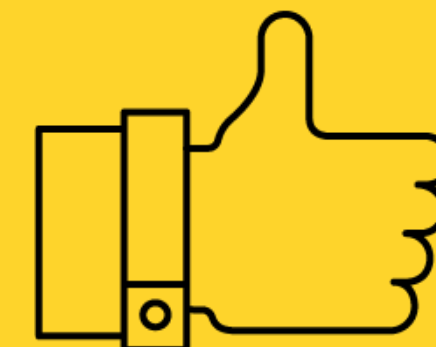
МИНУСЫ

Задержка
в поставке графа



ПЛЮСЫ

Не теряем связность графа
для пользовательских
процессов на границе дней



Run Graph → Regular Graph

К обработанному Run Graph подклеиваем данные из других логов, не требующих обработки (например, из снапшотов YTsaurus, в которых есть информация обо всех объектах в системе)

А как решается проблема партиций?

<input type="checkbox"/>	Name ▾	Locks ⚙	Account ⚙	Disk space ⚙	Rows ⚙	Modification time ⚙	Creation time ⚙			
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-06T00:00:00	<input type="checkbox"/>	statbox-cube	419.25 MiB	5 858 064	08 Sep 2024 15:11:02	08 Sep 2024 14:50:35	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-05T00:00:00	<input type="checkbox"/>	statbox-cube	420.74 MiB	5 859 428	08 Sep 2024 05:22:01	08 Sep 2024 04:48:15	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-04T00:00:00		statbox-cube	418.60 MiB	5 831 862	06 Sep 2024 20:40:42	06 Sep 2024 20:02:48	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-03T00:00:00		statbox-cube	417.27 MiB	5 815 200	06 Sep 2024 20:52:11	06 Sep 2024 20:22:23	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-02T00:00:00		statbox-cube	416.46 MiB	5 791 482	04 Sep 2024 15:18:43	04 Sep 2024 14:44:27	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-09-01T00:00:00		statbox-cube	418.19 MiB	5 804 769	04 Sep 2024 15:20:54	04 Sep 2024 14:51:08	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-08-31T00:00:00		statbox-cube	421.74 MiB	5 849 827	02 Sep 2024 12:35:11	02 Sep 2024 12:14:25	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-08-30T00:00:00		statbox-cube	416.67 MiB	5 791 297	01 Sep 2024 08:58:21	01 Sep 2024 08:26:46	<input type="checkbox"/>	<input type="checkbox"/>	...
<input type="checkbox"/>	<input type="checkbox"/> 2024-08-29T00:00:00		statbox-cube	414.92 MiB	5 761 329	31 Aug 2024 14:39:07	31 Aug 2024 14:07:32	<input type="checkbox"/>	<input type="checkbox"/>	...

Партиционирование

Мы решаем эту проблему путем некоторых эвристик

//table/1d/2024-01-01

//my/2024-01/money

//test/1704067200

//table/1d/2024-01-03

//my/2024-02/money

//test/1704067201

//table/1d/2024-01-03

//my/2024-03/money

//test/1704067202

//table/1d/*

//my/*/money

//test/*

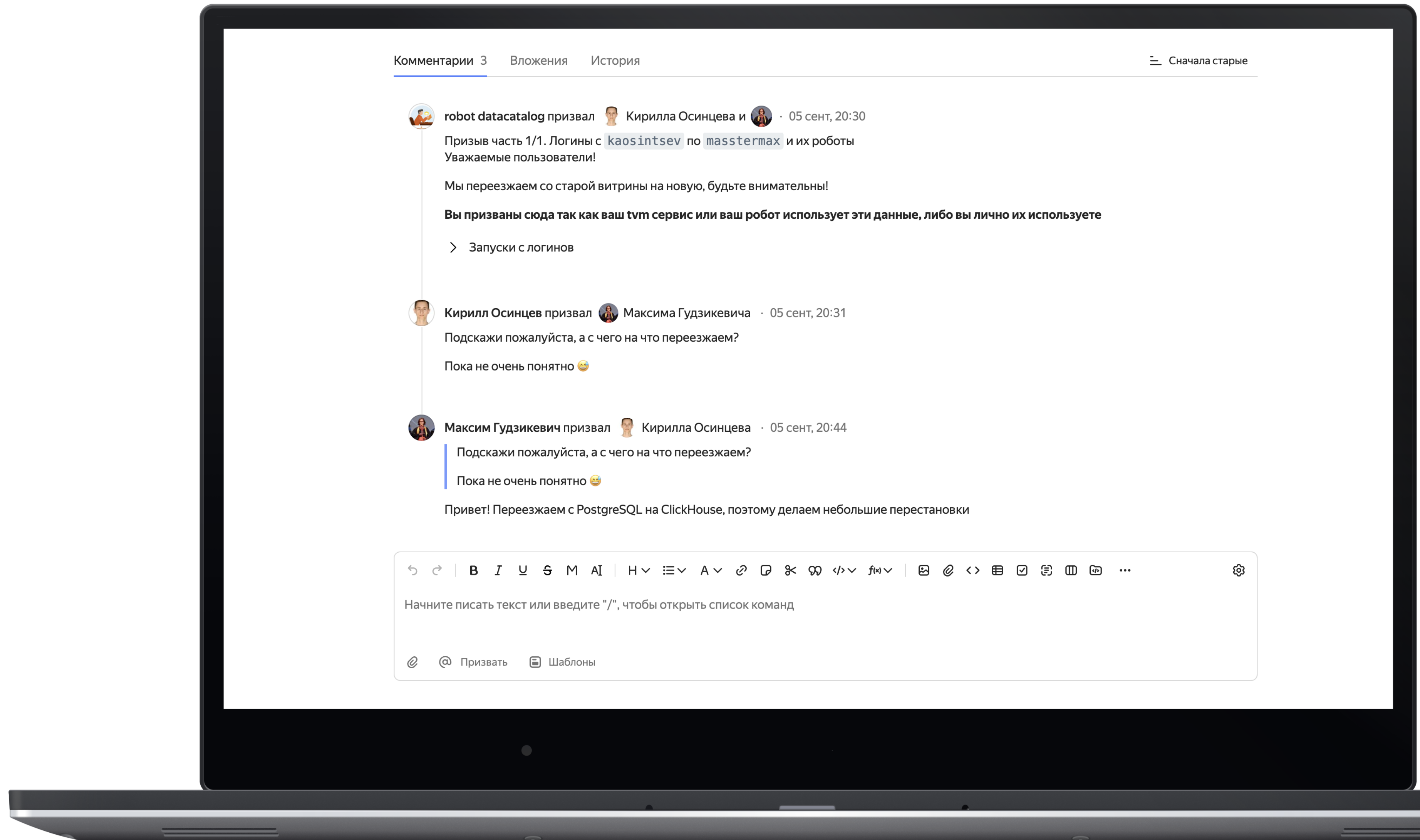


03

Как становимся точкой истины

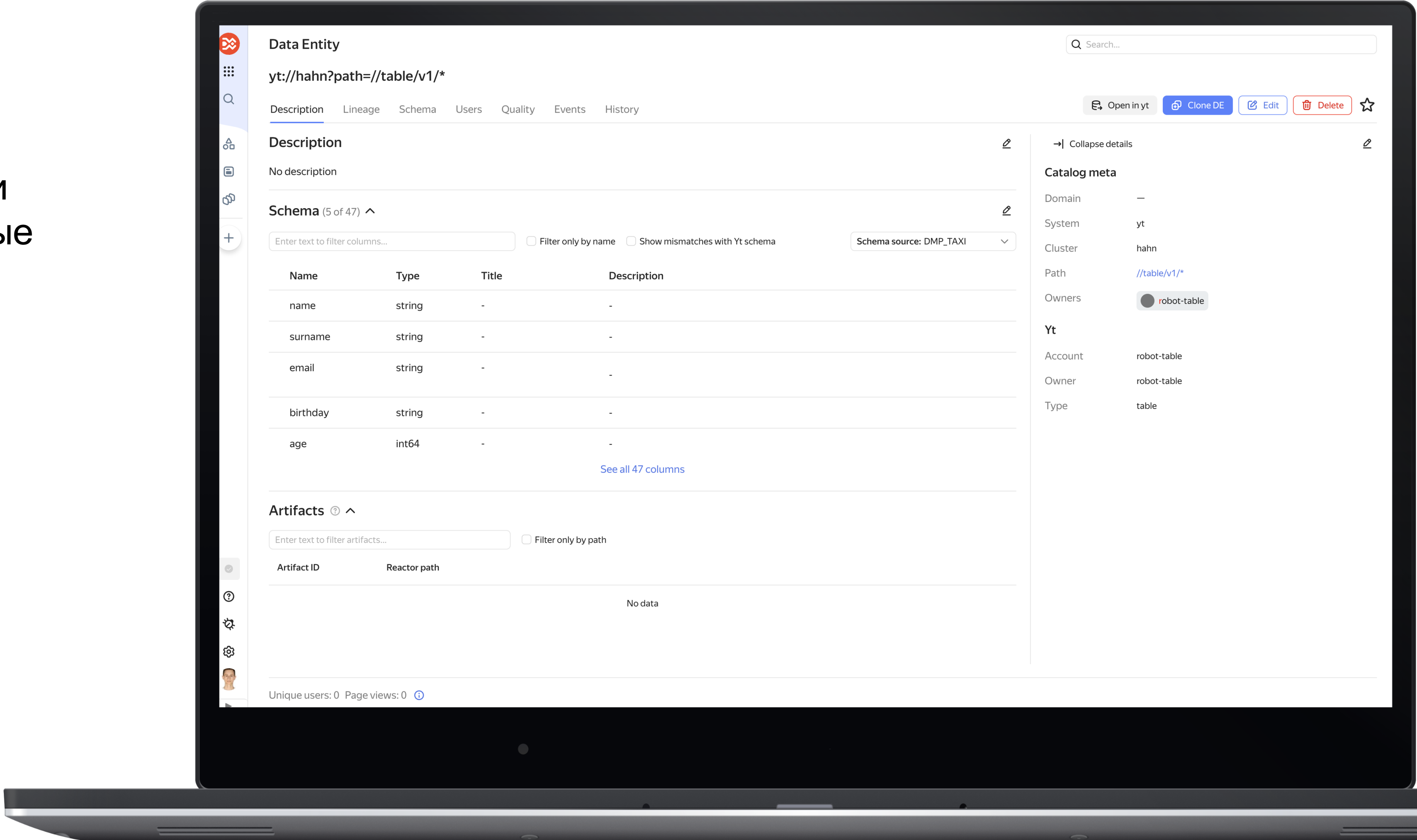
Наши преимущества

1. События



Наши преимущества

- 1. События
- 2. Нет необходимости пушить в нас данные



Наши преимущества

1. События
2. Нет необходимости
пушить в нас данные
3. Полноценный поиск
по данным

Search filters

Verticals	Data entities, Articles, Domains	✕	▼	
Systems	Select system	▼		🗑️
Clusters	Select cluster	▼		🗑️
Domain	Select domain	▼		🗑️
Owners	Select users, staff departments or abc serv...	▼		🗑️
Responsibles	Select users, staff departments or abc serv...	▼		🗑️
Editors	Select users, staff departments or abc serv...	▼		🗑️
Attributes	Add attribute			🗑️
Fast search	Enabled	Disabled		🗑️

Add filter

Search Reset all filters

Наши преимущества

1. События
2. Нет необходимости пушить в нас данные
3. Полноценный поиск по данным
4. Выстроенная интеграция с YT

Schema (5 of 12) ^

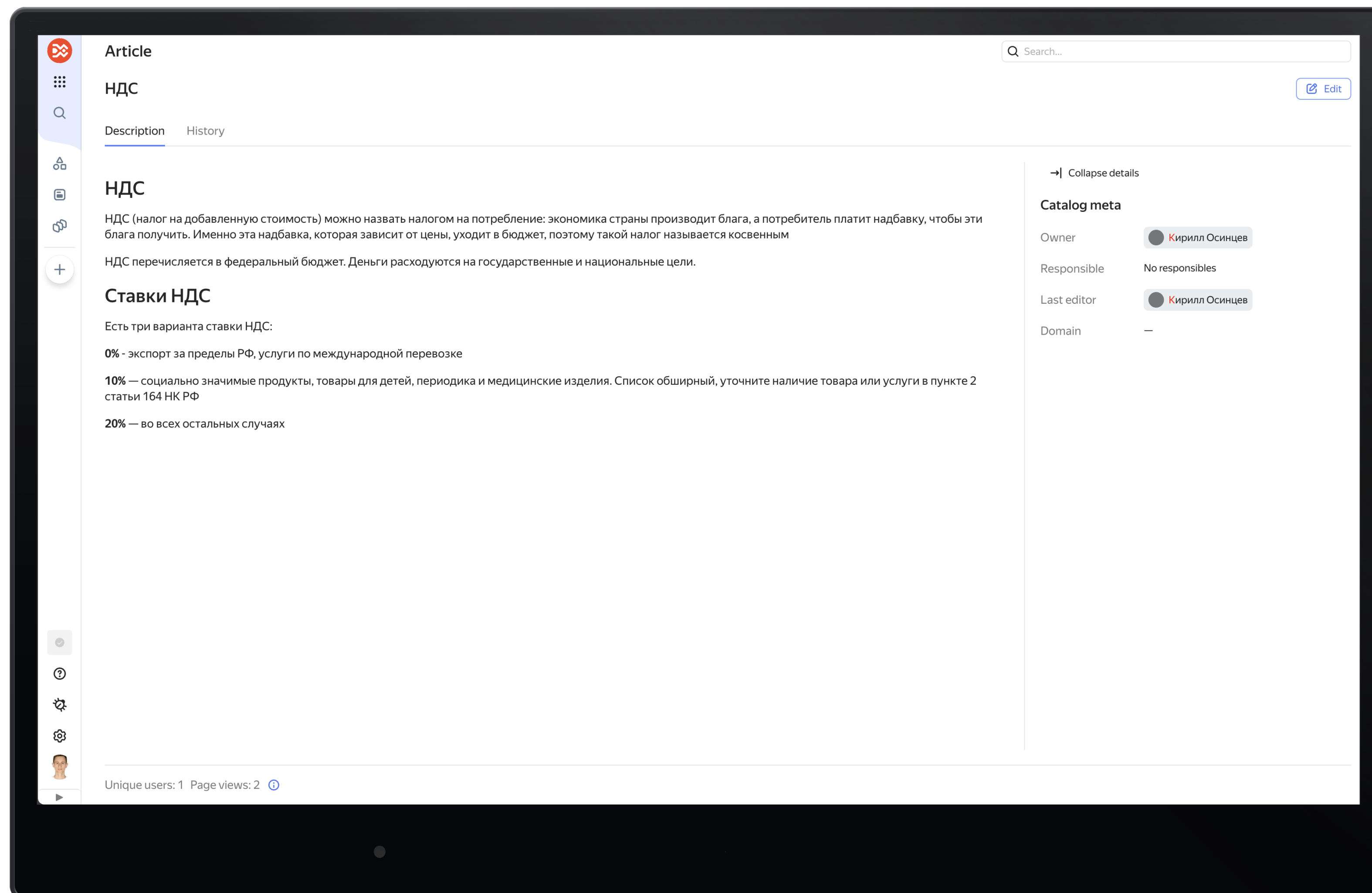
Enter text to filter columns... Filter only by name Show mismatches with Yt schema Schema source: YT

Name	Type	Title	Description
name	string	Имя человека	-
surname	string	Фамилия человека	-
birthday	string	День рождения человека	-
age	int64	Возраст человека	-
job	string	Работа человека	-

[See all 12 columns](#)

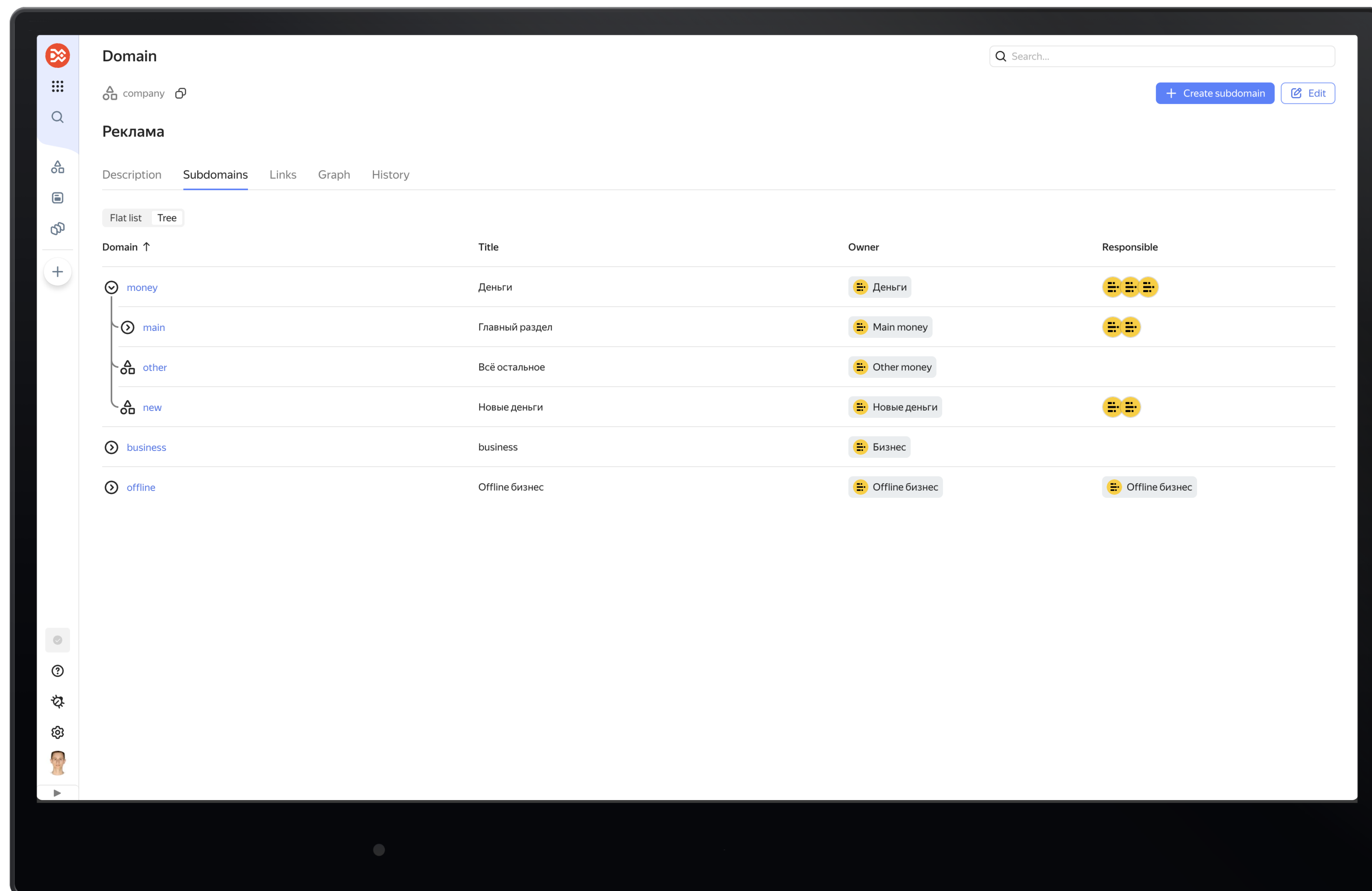
Наши преимущества

1. События
2. Нет необходимости пушить в нас данные
3. Полноценный поиск по данным
4. Выстроенная интеграция с YТ
5. Статьи



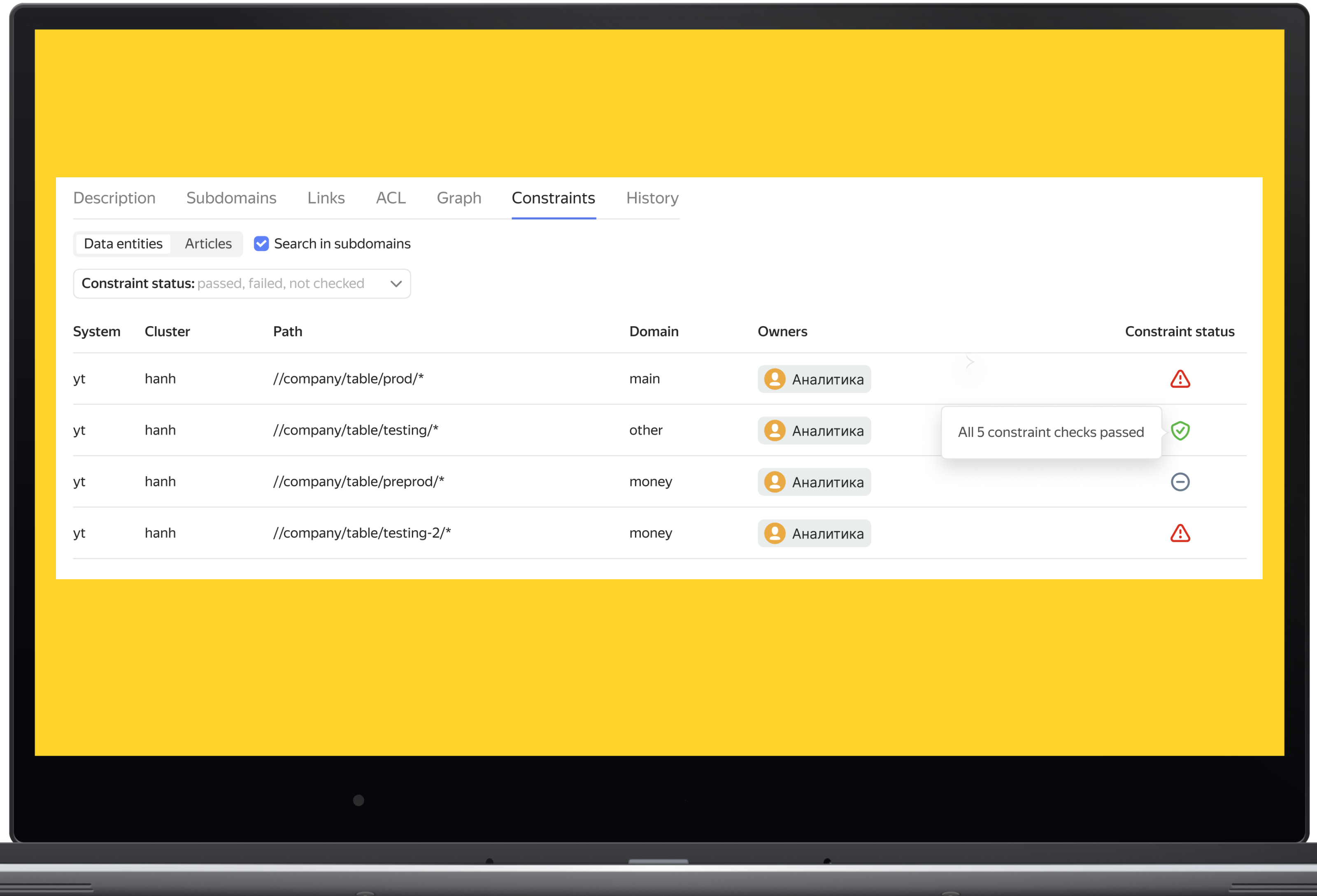
Наши преимущества

1. События
2. Нет необходимости пушить в нас данные
3. Полноценный поиск по данным
4. Выстроенная интеграция с YT
5. Статьи
6. Домены



Наши преимущества

1. События
2. Нет необходимости пушить в нас данные
3. Полноценный поиск по данным
4. Выстроенная интеграция с YT
5. Статьи
6. Домены
7. Отличаем хорошие данные от плохих



04

Выводы

Метрики

250

DAU

573

MAU Q3

1243

MAU Q2

Метрики

250

DAU

573

MAU Q3

1243

MAU Q2

863

events

665

articles

3000

domanis

Метрики

170 млн

Сущностей данных

300 млн

Процессов

1 млрд

Рёбер

Яндекс



SmartData

2024

Ваши вопросы