Т БАНК × SmartData

Инструменты Data Quality: как, зачем, почему. Опыт Т-Банка

Руднев Дмитрий

Руднев Дмитрий

@dorudnev

- 10 лет опыта разработки инструментов для работы с данными: Data Visualization (Dashboard), Data Preparation & ETL, Data Governance (Data Catalogs & DQ)
- Team Lead and Product Owner
- Руковожу направлением разработки платформы управления данными в Т-Банк

О чем поговорим?!

О чем поговорим?!

- Поговорим зачем нужны DQ инструменты

О чем поговорим?!

- Поговорим зачем нужны DQ инструменты
- Пройдем путь по выбору подхода к внедрению DQ инструментов

О чем поговорим?!

- Поговорим зачем нужны DQ инструменты
- Пройдем путь по выбору подхода к внедрению DQ инструментов
- Оценим плюсы и минусы каждого из подходов

О чем поговорим?!

- Поговорим зачем нужны DQ инструменты
- Пройдем путь по выбору подхода к внедрению DQ инструментов
- Оценим плюсы и минусы каждого из подходов
- Расскажу опыт разработки и внедрения DQ инструментов в Т-Банке

Этап 1.
Осознание

Качество данных — важно

Качество данных

6f3643aa	Ив@нов	15	+79991112233
3039c97e	Петров	45	89991112233
3039c97e		-1	+7999111223
0d42f28d	Смирнов	21	+79991112233

Качество данных

- Субъективно и ситуативно

6f3643aa	Ив@нов	15	+79991112233
3039c97e	Петров	45	89991112233
3039c97e		-1	+7999111223
0d42f28d	Смирнов	21	+79991112233

Качество данных

- Субъективно и ситуативно
- Делай сейчас — результат потом

6f3643aa	Ив@нов	15	+79991112233
3039c97e	Петров	45	89991112233
3039c97e		-1	+7999111223
0d42f28d	Смирнов	21	+79991112233

Качество данных

- Субъективно и ситуативно
- Делай сейчас — результат потом
- Сложно измерить эффект

6f3643aa	Ив@нов	15	+79991112233
3039c97e	Петров	45	89991112233
3039c97e		-1	+7999111223
0d42f28d	Смирнов	21	+79991112233


ПЛАН ЭВАКУАЦИИ
А-А-А-А-А!


ПЛАН ЭВАКУАЦИИ
А-А-А-А-А!

Люди

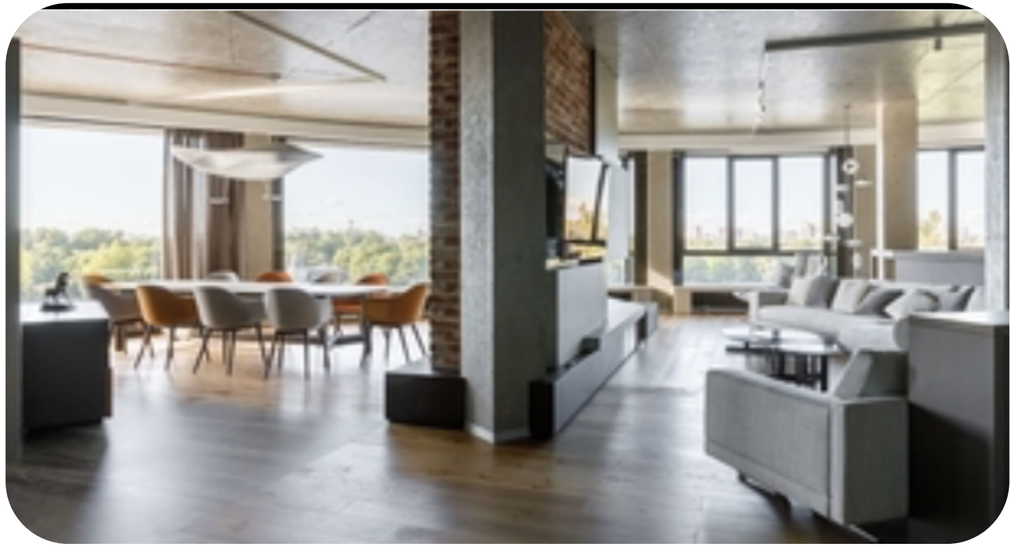
Процессы

Технологии

People, Process, Technology Framework

Осознание особенностей компании

VS



Осознание особенностей компании

Осознание особенностей компании

- Размер и структура компании

Осознание особенностей компании

- Размер и структура компании
- Уровень зрелости процессов

Осознание особенностей компании

- Размер и структура компании
- Уровень зрелости процессов
- Уровень компетенции людей

Осознание особенностей компании

- Размер и структура компании
- Уровень зрелости процессов
- Уровень компетенции людей
- Уровень критичности процессов на данных

Осознание особенностей компании

- Размер и структура компании
- Уровень зрелости процессов
- Уровень компетенции людей
- Уровень критичности процессов на данных
- IT ландшафт

Особенности Data Platform Т- Банк

Особенности Data Platform T- Банк

Единая Data Platform

- MAU 16k+, 170k+ запросов к GP ежедневно, 15+ кластеров GP

Особенности Data Platform Т- Банк

Единая Data Platform

- MAU 16k+, 170k+ запросов к GP ежедневно, 15+ кластеров GP

Много бизнес доменов

- 27 крупных групп пользователей
- Разный уровень зрелости, разные потребности в качестве данных

Особенности Data Platform T- Банк

Единая Data Platform

- MAU 16k+, 170k+ запросов к GP ежедневно, 15+ кластеров GP

Много бизнес доменов

- 27 крупных групп пользователей
- Разный уровень зрелости, разные потребности в качестве данных

“Динамичный” технологический ландшафт

Выводы первого этапа

Выводы первого этапа

- Качество данных – субъективно.

Выводы первого этапа

- Качество данных – субъективно.
- Помимо инструментов, нужны люди и процессы.

Выводы первого этапа

- Качество данных – субъективно.
- Помимо инструментов, нужны люди и процессы.
- Нужно понимать особенности компании.

Выводы первого этапа

- Качество данных – субъективно.
- Помимо инструментов, нужны люди и процессы.
- Нужно понимать особенности компании.
- Нет универсального решения.

Этап 2.
Выбор типа инструмента

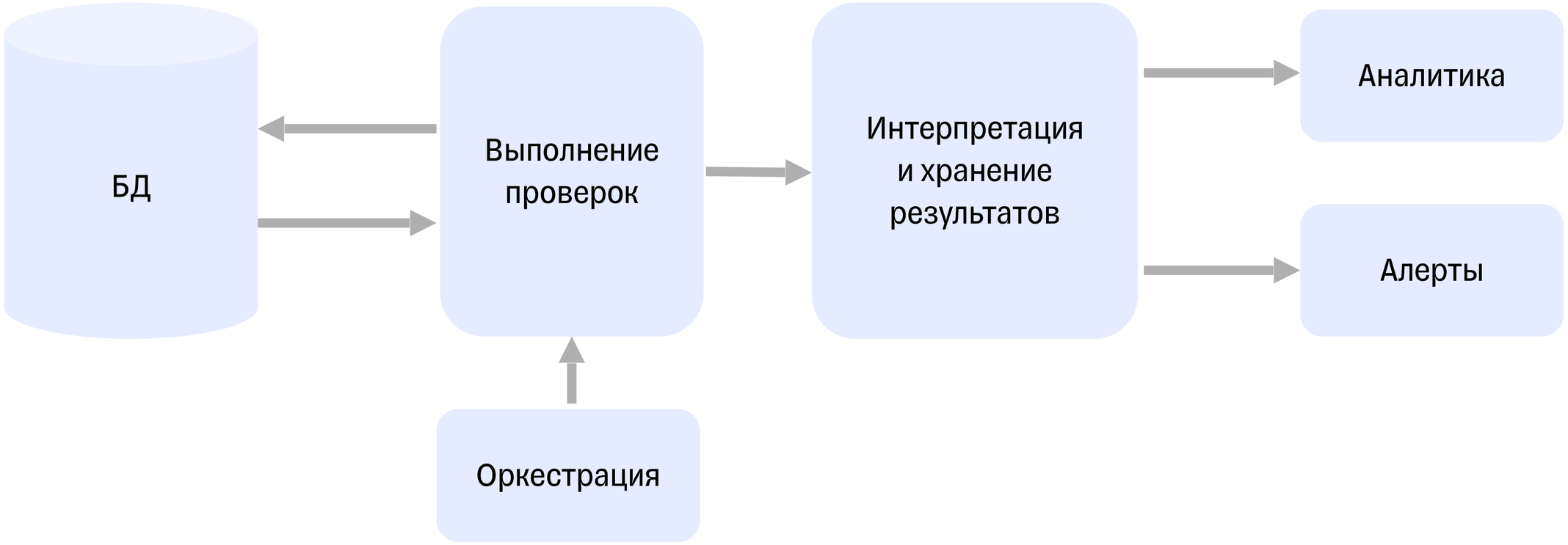
Функциональные возможности DQ инструментов

- Мониторинг (Observability)
- Профилирование данных (Profiling)
- Подготовка данных (Parsing, standardizing and cleansing)
- Инструменты фикса данных (Data Fixing)
- Поддержка DataOps (CI/CD)
- … еще 12 возможностей по версии Gartner

Magic Quadrant for Data Quality Solutions 2022.

Инструменты мониторинга данных

Узнавать о проблемах с данными раньше чем потребители



Этап 3.
Выбор архитектуры

Структура DQ инструмента

Структура DQ инструмента

Выполнение проверок

- оркестрация
- коннект к данным
- среда выполнения запросов

Структура DQ инструмента

Выполнение проверок

- оркестрация
- коннект к данным
- среда выполнения запросов

Логика проверок

- технические проверки
- бизнесовые проверки

Структура DQ инструмента

Выполнение проверок

- оркестрация
- коннект к данным
- среда выполнения запросов

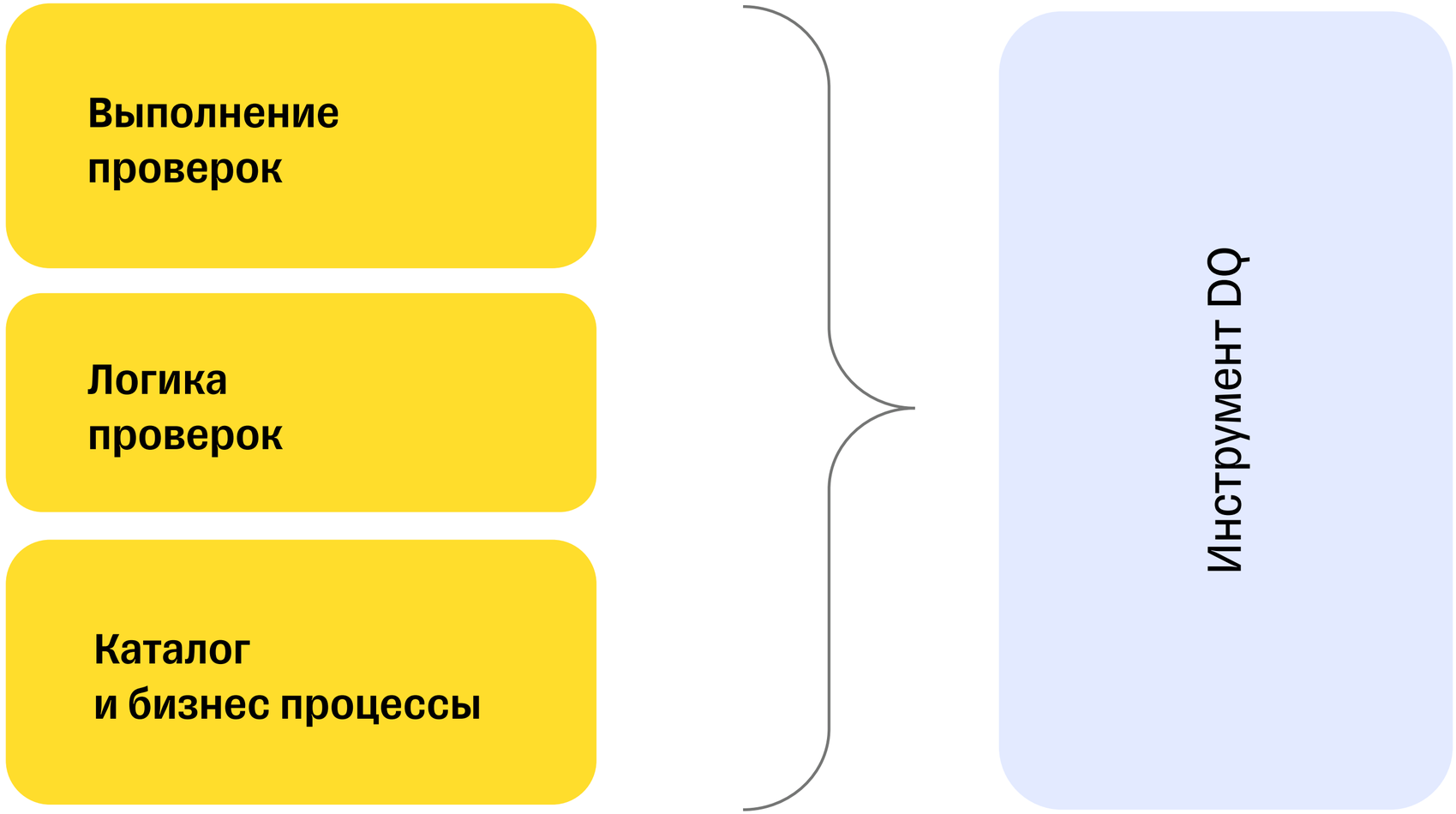
Логика проверок

- технические проверки
- бизнесовые проверки

Каталог и бизнес процессы

- хранение результатов
- аналитика
- нотификации, владение

Архитектура “все-в-одном”



Архитектура «все-в-одном»

- Informatica Data Quality (IDQ)
- SAP Data Services
- Ataccama ONE
- Oracle Enterprise Data Quality
- Microsoft Data Quality Services
- SAS Data Quality
- DQLabs Platform
- … и т.д.

Gartner Data Quality Solutions Reviews

Архитектура «все-в-одном» — наш опыт

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI
- Собственный синтаксис написания проверок КД

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI
- Собственный синтаксис написания проверок КД
- Собственный коннект к данным

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI
- Собственный синтаксис написания проверок КД
- Собственный коннект к данным
- Оркестрация запусков внутри приложения

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI
- Собственный синтаксис написания проверок КД
- Собственный коннект к данным
- Оркестрация запусков внутри приложения
- Спустя время: “внешние проверки”

Архитектура «все-в-одном» — наш опыт

- Web приложение на Scala
- Настройка проверок, запуска и нотификации через UI
- Собственный синтаксис написания проверок КД
- Собственный коннект к данным
- Оркестрация запусков внутри приложения
- Спустя время: “внешние проверки”

Архитектура «все-в-одном» — наш опыт

Архитектура «все-в-одном» — наш опыт

- Противоречивые требования от пользователей

Архитектура «все-в-одном» — наш опыт

- Противоречивые требования от пользователей
- Дорого “догонять” динамичную Data Platform

Архитектура «все-в-одном» — наш опыт

- Противоречивые требования от пользователей
- Дорого “догонять” динамичную Data Platform
- Ограниченный ресурс разработки

Архитектура «все-в-одном» — выводы

Архитектура «все-в-одном» — выводы

Удобна для не технического пользователя

Архитектура «все-в-одном» — выводы

- Удобна для не технического пользователя
- Хороша, когда ты ее покупаешь как коробку вместе с другими инструментами

Архитектура «все-в-одном» — выводы

- Удобна для не технического пользователя
- Хороша, когда ты ее покупаешь как коробку вместе с другими инструментами

Дорого и долго разрабатывать самим

Этап 3. Выбор архитектуры (заход второй)

Структура DQ инструмента

Выполнение проверок	▪ оркестрация ▪ коннект к данным ▪ среда выполнения запросов
Логика проверок	▪ технические проверки ▪ бизнесовые проверки
Каталог и бизнес процессы	▪ хранение результатов ▪ аналитика ▪ нотификации, владение

Выполнение проверок

Логика проверок

Каталог и бизнес процессы

Выполнение проверок

Standalone orchestration

Логика проверок

Каталог и бизнес процессы

Выполнение проверок

Standalone orchestration

Integrate in ETL tool

Логика проверок

Каталог и бизнес процессы

Выполнение проверок

Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

Каталог и бизнес процессы

Выполнение проверок

Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library

Каталог и бизнес процессы

Выполнение проверок

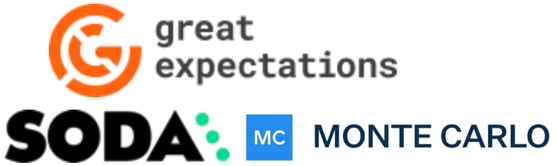
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

Выполнение проверок

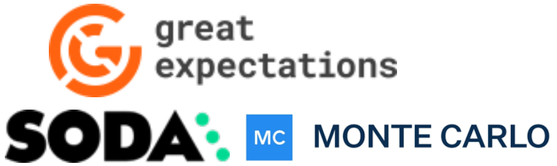
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

DB table

Выполнение проверок

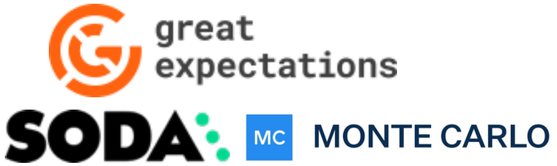
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

DB table

observability tools

Выполнение проверок

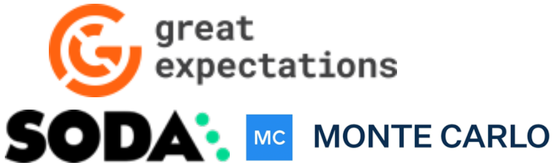
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

DB table

observability tools

data catalogs

Выполнение проверок

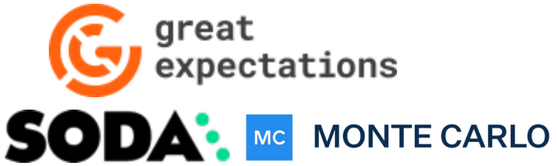
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

DB table

observability tools

data catalogs

cloud

Выполнение проверок

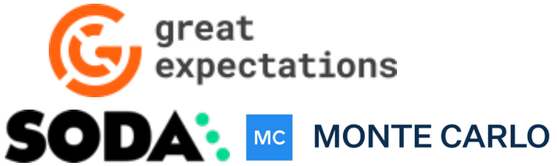
Standalone orchestration

Integrate in ETL tool

Логика проверок

raw/functions

raw/in-house library



open source library

Каталог и бизнес процессы

DB table

observability tools

data catalogs

cloud

in-house

Выполнение проверок

Helicopter
(Notebooks)

Логика проверок

raw/functions

raw/in-house library

Каталог и бизнес процессы

DB table

Выполнение проверок

Helicopter
(Notebooks)

Логика проверок

raw/functions

raw/in-house library

Каталог и бизнес процессы

DB table

Каталог проверок

Каталог проверок

Каталог проверок

- Быстрый поиск информации о качестве данных

Каталог проверок

- Быстрый поиск информации о качестве данных
- Возможность переиспользовать созданные проверки

Каталог проверок

- Быстрый поиск информации о качестве данных
- Возможность переиспользовать созданные проверки
- Быстрое создание проверок качества данных

Каталог проверок

- Быстрый поиск информации о качестве данных
- Возможность переиспользовать созданные проверки
- Быстрое создание проверок качества данных
- Нотификации “из коробки”

Каталог проверок

- Быстрый поиск информации о качестве данных
- Возможность переиспользовать созданные проверки
- Быстрое создание проверок качества данных
- Нотификации “из коробки”
- Аналитика по качеству данных

Каталог проверок

- Быстрый поиск информации о качестве данных
- Возможность переиспользовать созданные проверки
- Быстрое создание проверок качества данных
- Нотификации “из коробки”
- Аналитика по качеству данных
- Построение бизнес процессов по Data Quality и Data Governance

Каталог проверок – пишем свое

Каталог проверок – пишем свое

Облачные решения – недоступны

Каталог проверок – пишем свое

Облачные решения – недоступны

Observability инструменты – не хватило специальных фичей

Каталог проверок – пишем свое

Облачные решения – недоступны

Observability инструменты – не хватило специальных фичей

Open source – только в составе дата каталогов

Каталог проверок – пишем свое

Облачные решения – недоступны

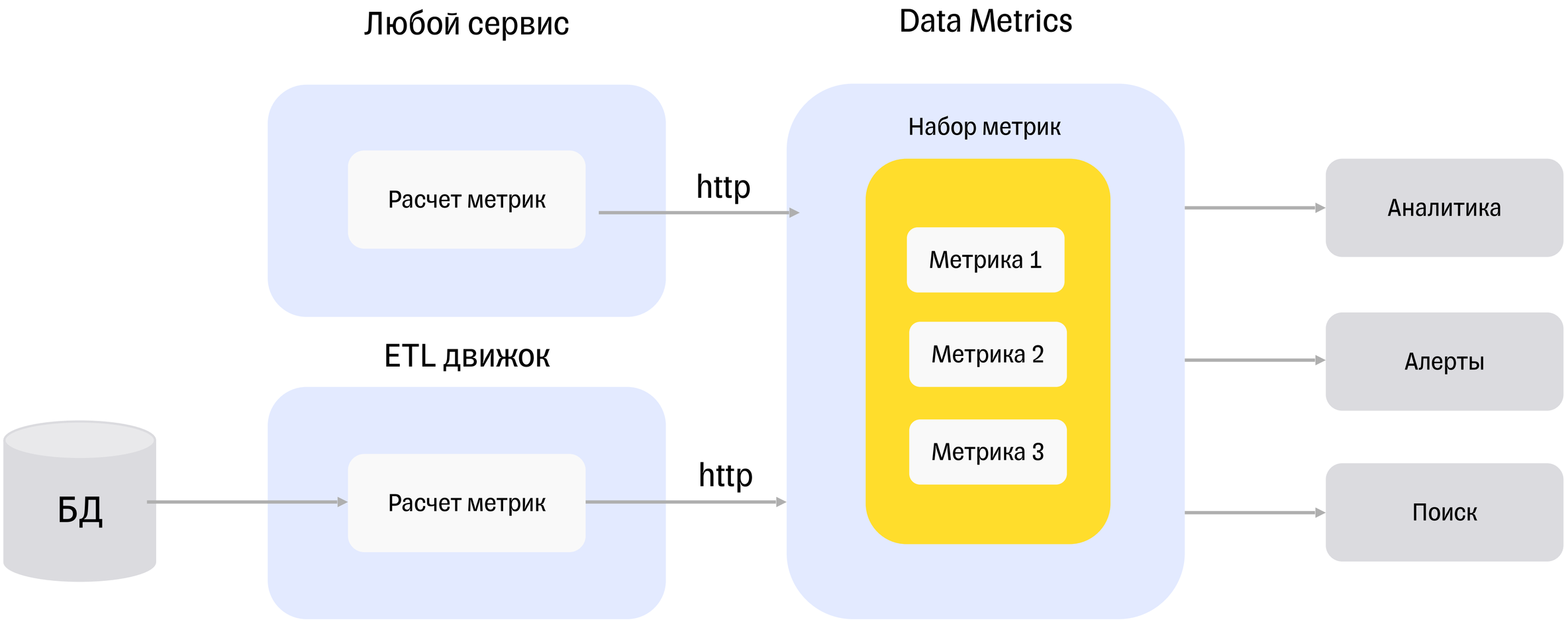
Observability инструменты – не хватило специальных фичей

Open source – только в составе дата каталогов

⚠ DQ Ops – open source каталог (релиз v1 – январь 2024)

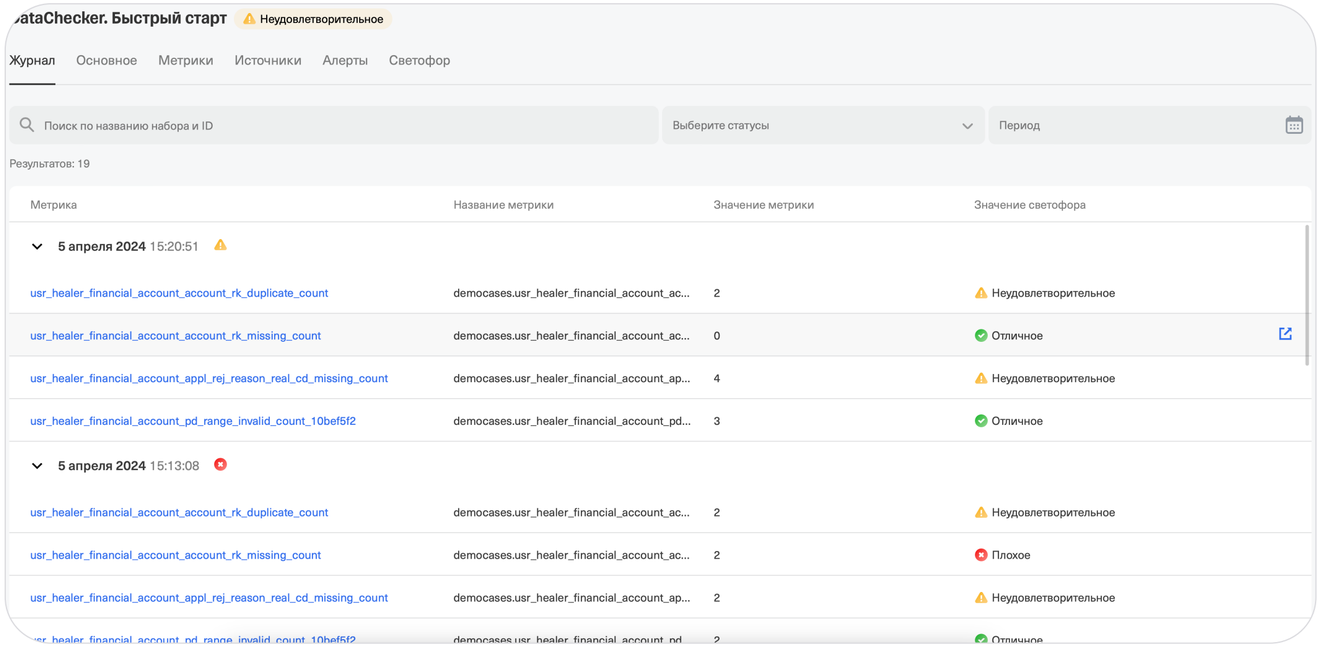
DQOps

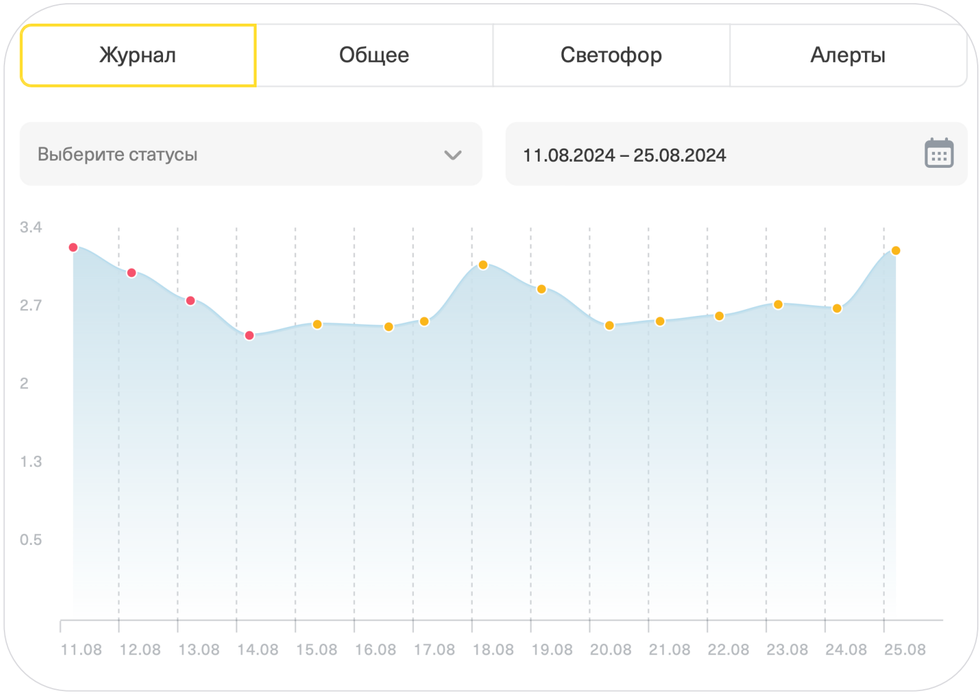
Data Metrics – каталог проверок



Data Metrics

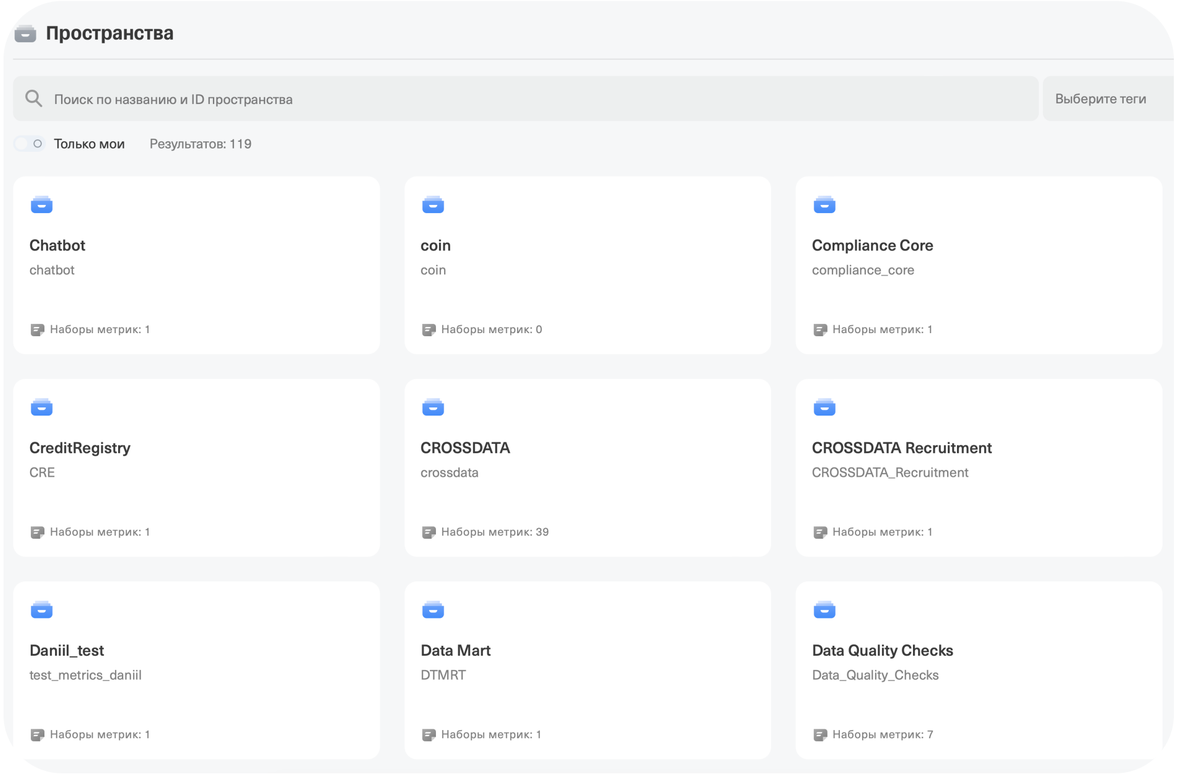
Хранилище метрик качества данных

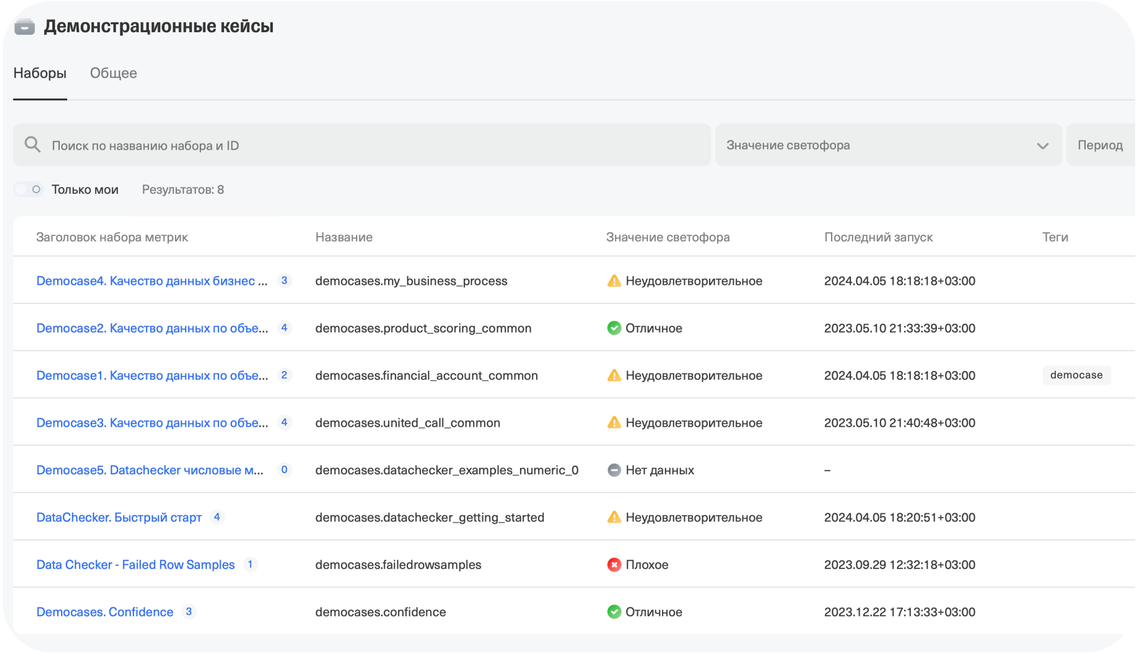




Data Metrics

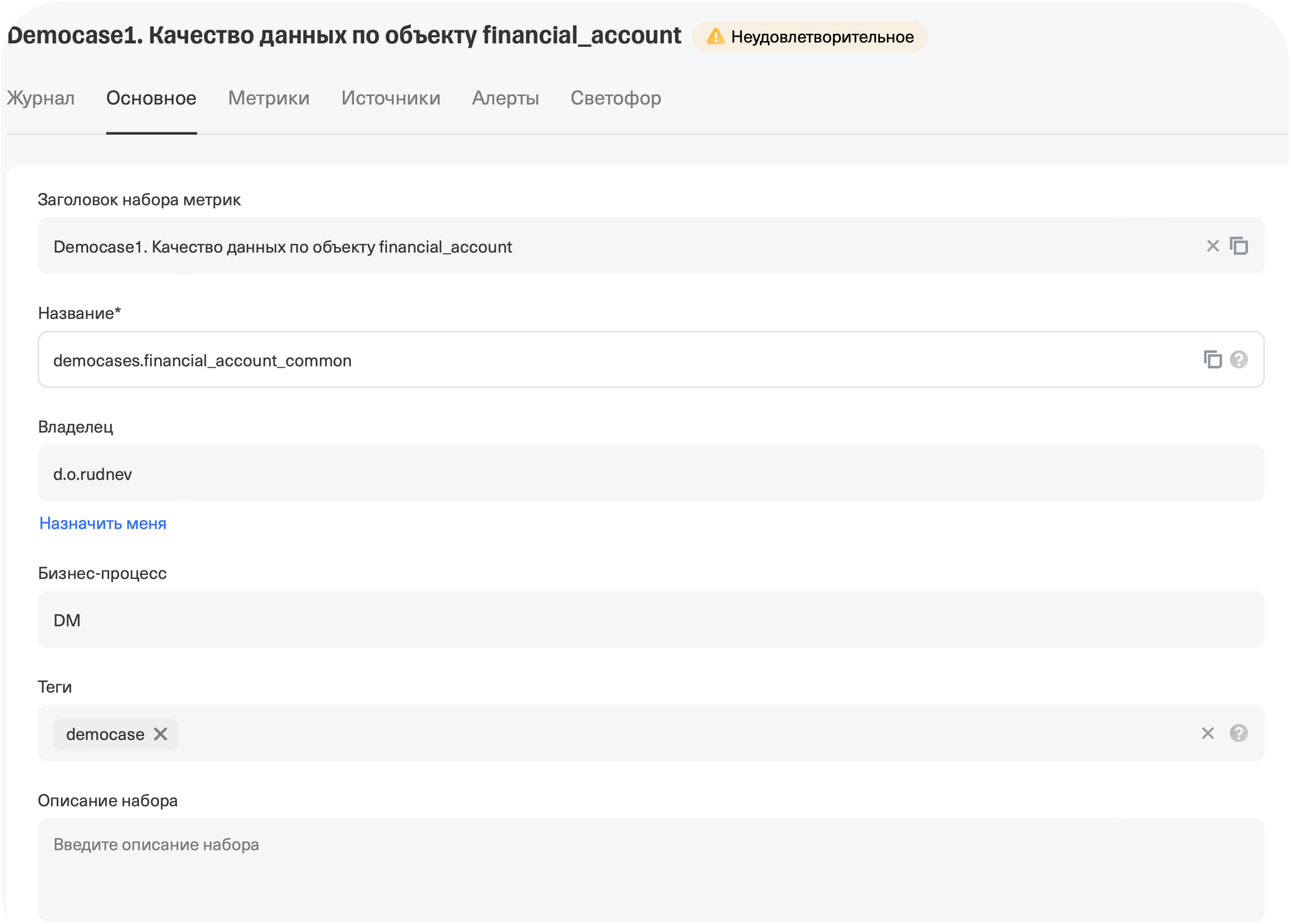
Группировка проверок по пространствам и наборам метрик
Поиск проверок

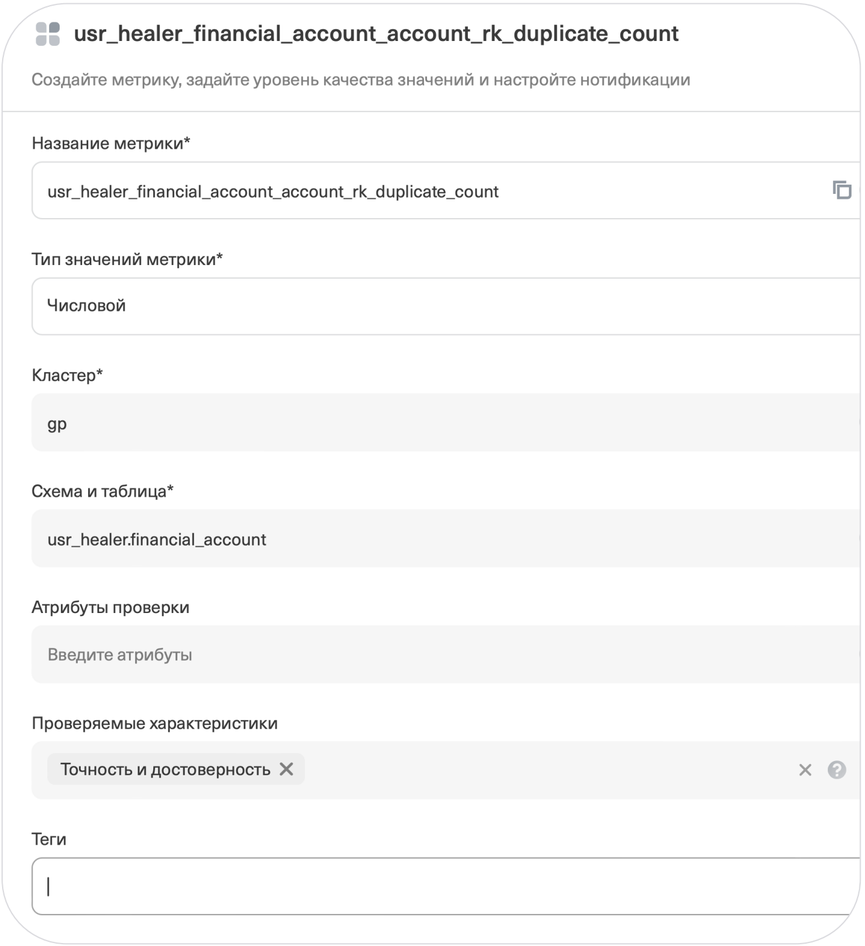




Data Metrics

Метаданные о проверках КД





Data Metrics

Интерпретация значений метрик (светофоры)

Helicopter

Untitled 1

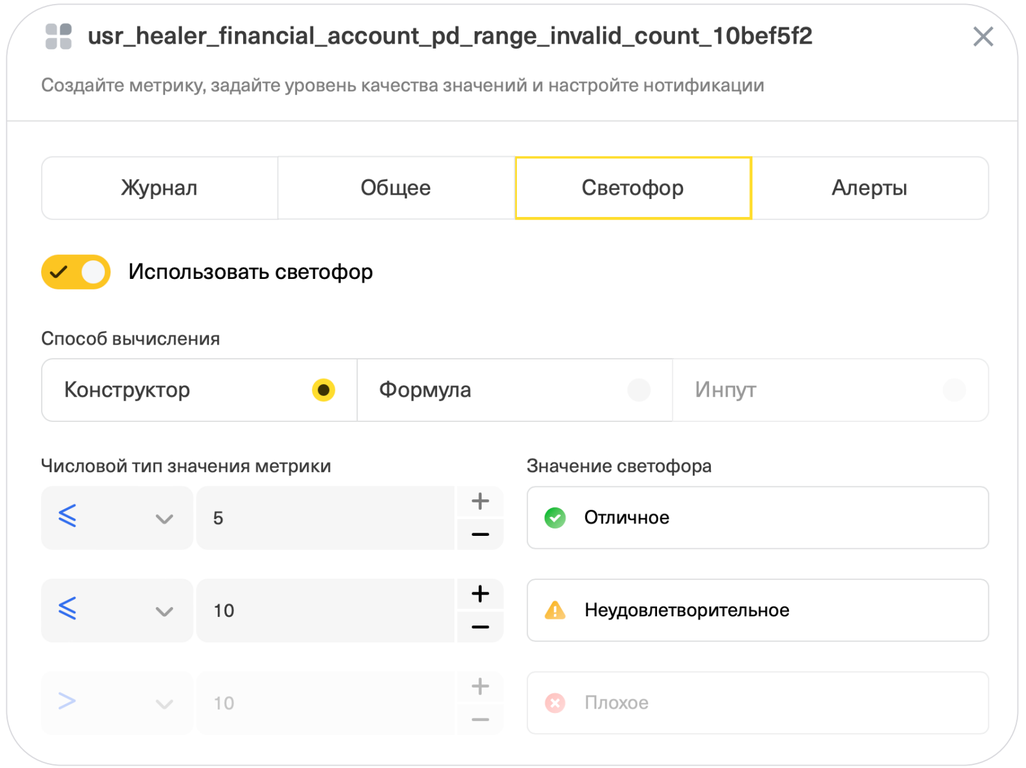
Source: gp

```
1  select
2         -- Неотрицательность полей
3         count(*) filter(where end_dttm <= start_dttm) as end_dttm_corrent,
4         -- null-ы начало\окончание звонка
5         count(*) filter(where united_call_end_dttm is null) as call_end_null_count
6  from usr_healer.united_call;
```

Text | Table-1

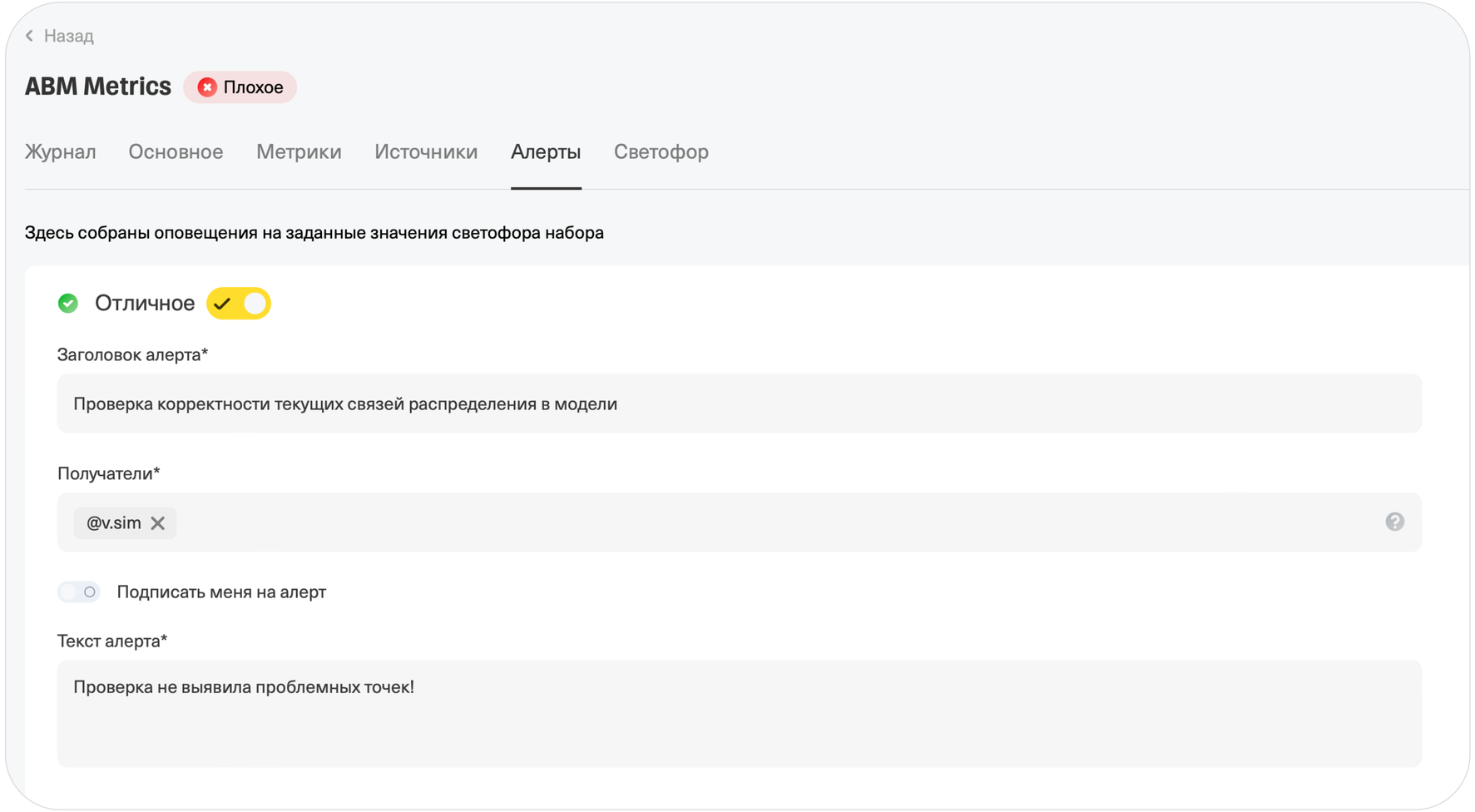
#	end_dttm_corrent	call_end_null_count
1	2	0

Data Metrics



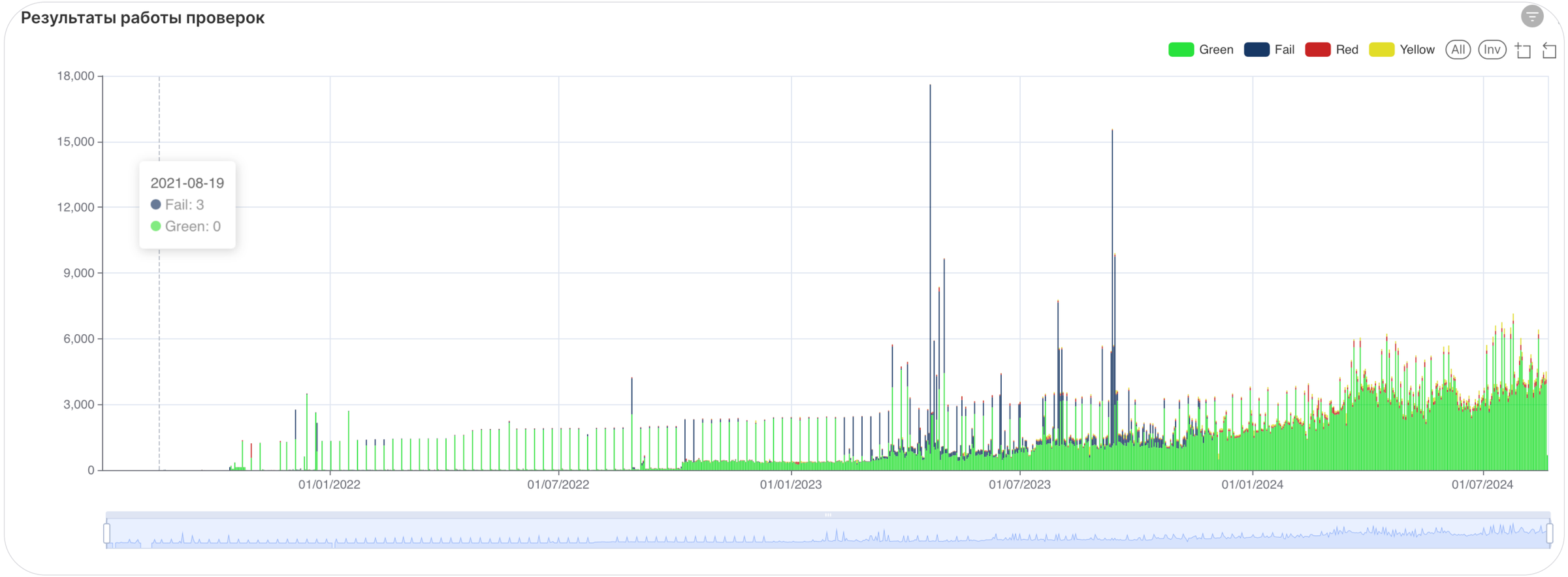
Data Metrics

Оповещения



Data Metrics

Аналитика по проверкам качества данных



Выполнение проверок

Helicopter
(Notebooks)

Tedi
(ETL)

Логика проверок

raw/functions

raw/in-house library

Каталог и бизнес процессы

(Data Metrics)
Каталог проверок

Выполнение проверок

Helicopter (Notebooks)

Tedi (ETL)

Логика проверок

raw/functions

raw/in-house library

Библиотека DQ проверок

Каталог и бизнес процессы

(Data Metrics) Каталог проверок

Библиотека DQ проверок

Библиотека DQ проверок

- Снижение порога входа в DQ

Библиотека DQ проверок

- Снижение порога входа в DQ
- Уменьшение времени на разработку проверок

Библиотека DQ проверок

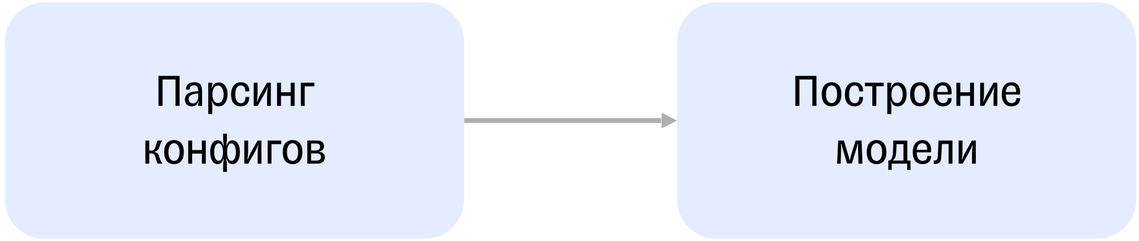
- Снижение порога входа в DQ
- Уменьшение времени на разработку проверок
- Оптимальные запросы

Библиотека DQ проверок

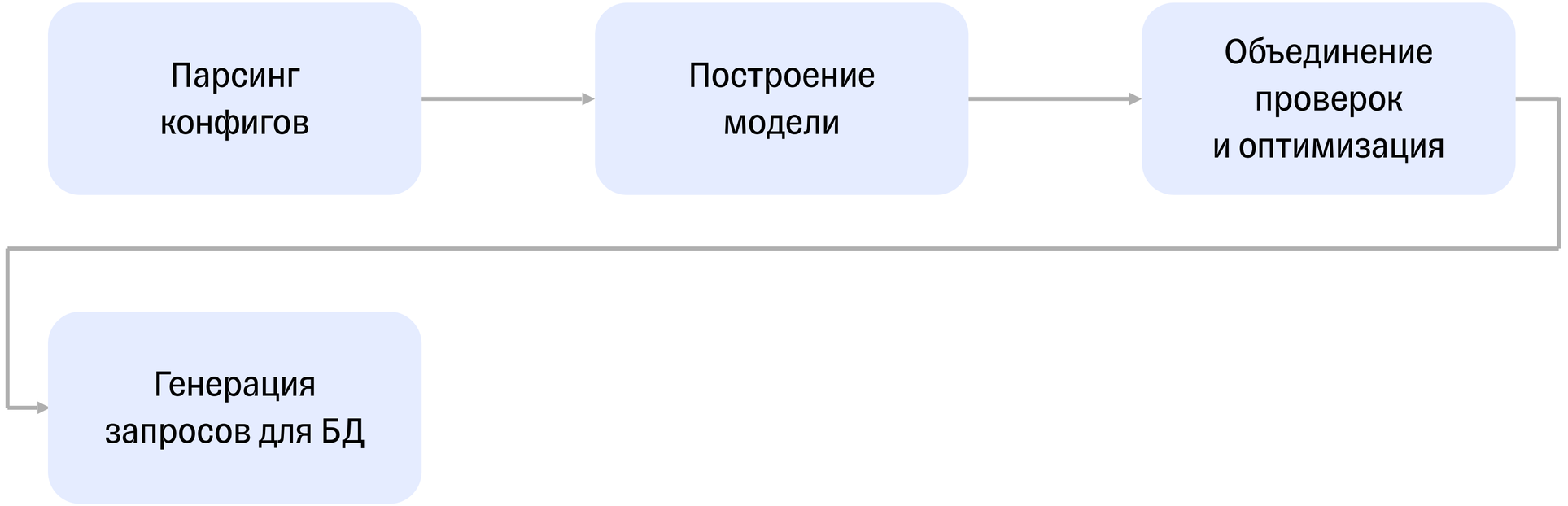
- Снижение порога входа в DQ
- Уменьшение времени на разработку проверок
- Оптимальные запросы
- Единый синтаксис под разные БД

Архитектура библиотека DQ проверок

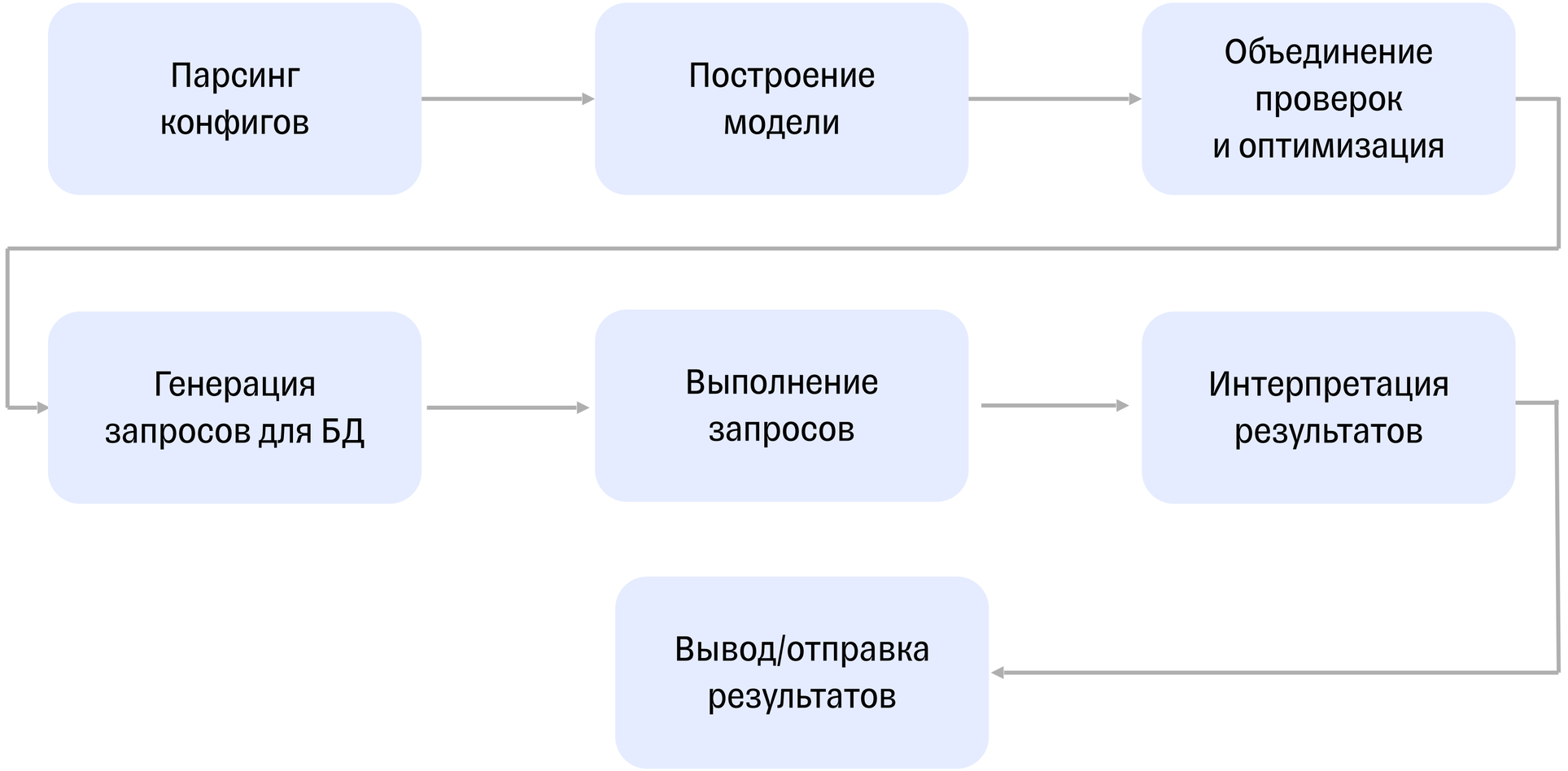
Архитектура библиотека DQ проверок



Архитектура библиотека DQ проверок



Архитектура библиотека DQ проверок




SODA


VS.


great
expectations

Soda Core SODA

- Github Start: 1846
- API: yaml-based DSL
- Генерация SQL: строки
- Мало зависимостей
- Низкая модульность

```
# Checks for basic validations
checks for dim_customer:
  - row_count between 10 and 1000
  - missing_count(birth_date) = 0
  - invalid_percent(phone) < 1 %:
      valid format: phone number
  - invalid_count(number_cars_owned) = 0:
      valid min: 1
      valid max: 6
  - duplicate_count(phone) = 0
```

soda.io

Great Expectation

- Github Start: 9700
- API: python function или JSON
- Генерация SQL: SqlAlchemy
- Много зависимостей
- Высокая модульность

Python

```python
validator.expect_column_values_to_not_be_null("pickup_datetime")
validator.expect_column_values_to_be_between(
    "passenger_count", min_value=1, max_value=6
)
validator.save_expectation_suite(discard_failed_expectations=False)
```

GreatExp
ectation

Soda Core vs Great Expectation

Great Expectation

Type: python

```python
batch.add_source("select * from prod_advtmart.social_stats where business_dt > cur_date")
batch.expect_column_values_to_not_be_null("advt_cabinet_nm")
batch.expect_compound_columns_to_be_unique(["business_dt", "banner_rk", "source_code"])
batch.expect_column_values_to_not_be_null(column="advt_group_nm", row_condition='col("source_code")=="mytarget"')
```

Soda Core

Type: python

```
'''
filter prod_advtmart.social_stats [daily]:
    where: business_dt > current_date'

checks for prod_advtmart.social_stats [daily]:
    - missing_count(banner_nm)
    - duplicate_count(business_dt, banner_rk, source_code)
    - missing_count(advt_group_nm):
        filter: sales_territory_key = 11
'''
```

Soda Core vs Great Expectation

По функционалу проверок — паритет

Тип метирки	Подтип	GreatExpectation	Soda Core
По колонке	Числовые агрегаты	✅	✅
	Строковые агрегаты	✅	✅
	Дубликаты/уникальность	✅	✅
	Null	✅	✅
	countif, количество или процент по условию	✅	✅
	Непрерывность	✅	⛔
По паре колонок	Сравнение значения	✅	✅
	Консистенстность/ссылочная целостность	✅	✅
На основе агрегатов		✅	⛔
Статистика	PSI и тд	✅	✅
ML	Аномалии	⛔	✅
	Прогнозирование	⛔	✅
Проверка на схему		✅	✅

Soda Core vs Great Expectation

- yaml-based DSL
- Низкая модульность
- Мало зависимостей
- Все проверки транслируются в SQL

- Python или JSON
- Высокая модульность
- Много зависимостей
- Часть проверок реализованы только в pandas

Data Checker – DQ библиотека

Data Checker – DQ библиотека

- Python библиотека основанная на Soda Core

Data Checker – DQ библиотека

- Python библиотека основанная на Soda Core
- Декларативное описание проверок с помощью yaml

Data Checker – DQ библиотека

- Python библиотека основанная на Soda Core
- Декларативное описание проверок с помощью yaml
- Большой набор встроенных функций проверки данных из коробки

Data Checker – DQ библиотека

- Python библиотека основанная на Soda Core
- Декларативное описание проверок с помощью yaml
- Большой набор встроенных функций проверки данных из коробки
- Расчет примеров ошибок

Data Checker – доработки Soda Core

Data Checker – доработки Soda Core

- Интеграция с Helicopter (Notebooks)

Data Checker – доработки Soda Core

- Интеграция с Helicopter (Notebooks)
- Интеграция с Data Metrics (Каталог проверок)

Data Checker – доработки Soda Core

- Интеграция с Helicopter (Notebooks)
- Интеграция с Data Metrics (Каталог проверок)
- Переделали сборку пакетов

Data Checker – доработки Soda Core

- Интеграция с Helicopter (Notebooks)
- Интеграция с Data Metrics (Каталог проверок)
- Переделали сборку пакетов
- Добавлены проверки на аномалии на ETNA (T-bank time series library)

Выполнение проверок

Helicopter
(Notebooks)

Tedi
(ETL)

Логика проверок

raw/functions

raw/in-house library

Библиотека DQ
проверок

Каталог и бизнес процессы

(Data Metrics)
Каталог проверок

Модульная архитектура – выводы

Модульная архитектура – выводы

- Дешевле за счет переиспользования готовых компонентов

Модульная архитектура – выводы

- Дешевле за счет переиспользования готовых компонентов
- Легче встраивать в другие инструменты

Модульная архитектура – выводы

- Дешевле за счет переиспользования готовых компонентов
- Легче встраивать в другие инструменты
- Легче подстраивать под пользовательские процессы

Модульная архитектура – выводы

- Дешевле за счет переиспользования готовых компонентов
- Легче встраивать в другие инструменты
- Легче подстраивать под пользовательские процессы

Дорого и долго разрабатывать самим

Что не хватает

Что не хватает

- «Простого» заведения проверок в простых кейсах
-

Что не хватает

- «Простого» заведения проверок в простых кейсах
- Фрагментированный пользовательский опыт — работа в нескольких инструментах

Этап 4.
Внедрение

Кейс 1. Генерация кода для «массовых проверок»

Type: python

```python
tables = get_all_tables_form_schema("prod_v_schema")
check_str = ""

for (table, keys) in tables:
    check_str += f'''
            checks for {table}:
                - missing_count({keys}):
                    name: {table_name}_missing_count
                - duplicate_count({keys}):
                    name: {table_name}_nonunique
                - confidence level for rowcount with daily granularity of dttm > 0.95:
                    name: {table_name}_rowcount_anomaly
                - row_count:
                    name: {table_name}_count
            '''

dc.run(check_str)
```

Заголовок набора метрик	
prod_ods_siebel	408
prod_ods_tcrm_proc_development	33
prod_ods_credb	261
prod_ods_fwb	350
prod_ods_xxi	691
prod_ods_tcrmp_backend	96
prod_ods_tcrm_tasks	440
prod_ods_acq	534

Кейс 2. Шаблоны

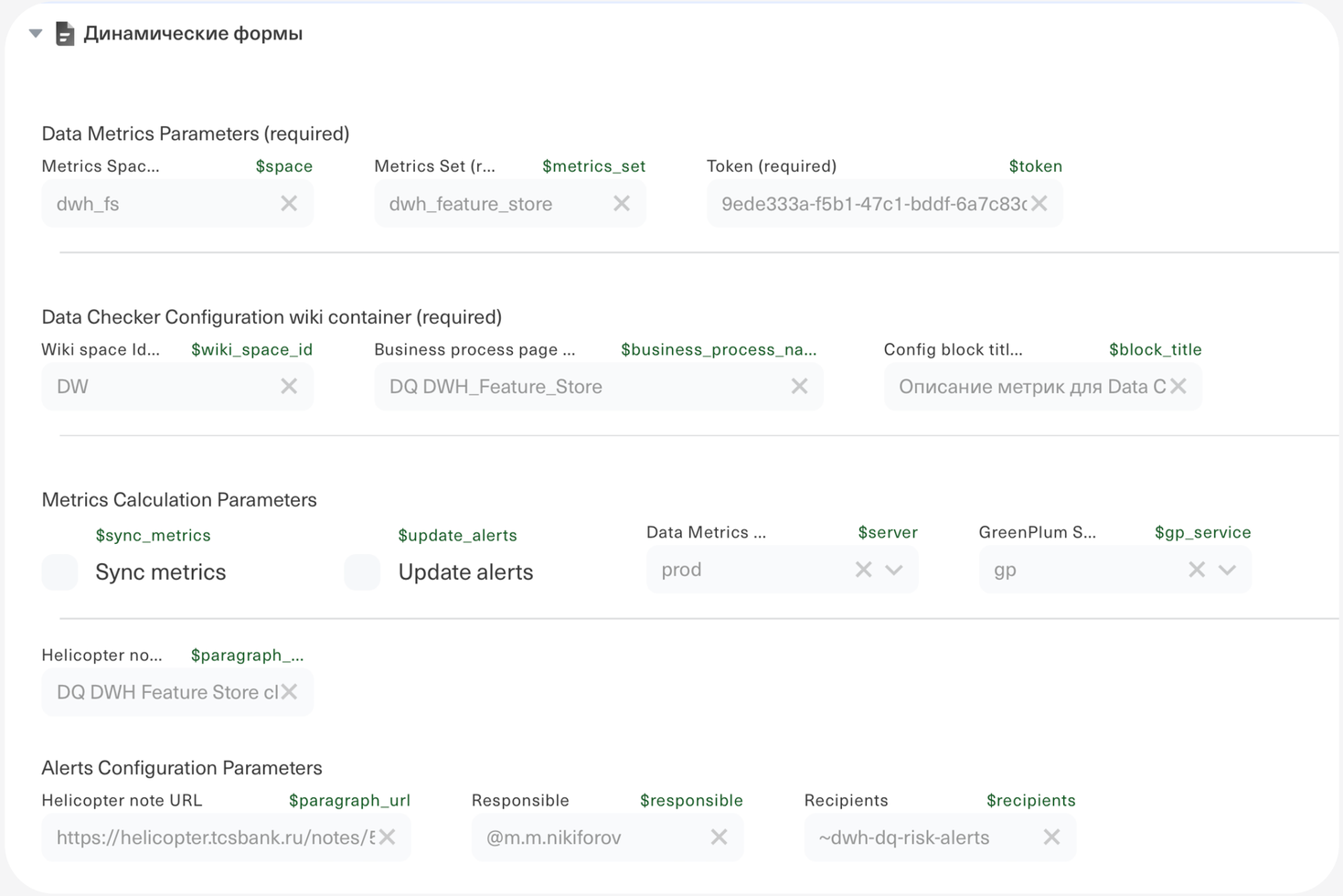
Проверки DDS.CHAT_SEGMENT за все время 15	dwh_dko.DDS_CHAT_SEGMENT	Неудовлетворительное	2024.08.25 03:37:29+03:00
Проверки DDS.CHAT_SEGMENT за вчера 16	dwh_dko.DDS_CHAT_SEGMENT_yesterday	Отличное	2024.08.25 03:33:24+03:00
Проверки DDS.CHAT_SEGMENT за месяц 15	dwh_dko.DDS_CHAT_SEGMENT_month	Плохое	2024.08.25 03:33:00+03:00

Type: python

```
conditions = {
    "yesteday": "processed_dttm >= CURRENT_DATE - INTERVAL '1 day' AND processed_dttm < CURRENT_DATE",
    "this_month: ""processed_dttm >= DATE_TRUNC('month', CURRENT_DATE) AND processed_dttm < DATE_TRUNC('month', CURRENT_DATE) + INTERVAL '1 month'"
}

check_str = f'''
    filter prod_v_dds.crm_user [filtered]:
        where: {conditions[yesteday]}
    checks for prod_v_dds.crm_user [filtered]:
        - duplicate_percent(crm_user_rk, valid_from_dttm):
            name: DDS_CRM_USER_crm_user_rk_dupl
        - missing_percent(crm_user_rk):
            name: DDS_CRM_USER_crm_user_rk_notnull
        - missing_percent(crm_user_id):
            name: DDS_CRM_USER_crm_user_id_notnull
  '''
```

Кейс 3. Notebook based UI



Заключение

Заключение

Заключение

Повышение качества данных – ежедневная рутина

Заключение

Повышение качества данных – ежедневная рутина

Никакие инструменты не решат ваших проблем, но упростят путь к качественным данным

Заключение

Повышение качества данных – ежедневная рутина

Никакие инструменты не решат ваших проблем, но упростят путь к качественным данным

DQ инструменты могут адаптировать под запросы и возможности, начните с простого

Т БАНК

Спасибо!