

# dbt в деле: реальные кейсы и лайфхаки

Александров Антон



# Содержание

1. О докладчике и Группе компаний Детский мир
2. Начало
  1. BigData
  2. Задачи
3. Первые подходы к dbt
4. Организация CI/CD
5. Подготовка Аналитиков
6. Деплой в Airflow
7. Новые вызовы
8. Итоги





# 01

## О докладчике



## 1. Обучение

МИИТ Прикладная математика и Информатика  
МФТИ Прикладная математика и физика

## 2. Предыдущие проекты

Beeline - Performance Management платформа  
200 GB raw data /day  
Работа с MapReduce API

Luxoft - Озеро данных Почты России/ Отчеты для команды Логистики  
3 PB Hadoop Cluster size

SeverGroup - собственная платформа ClickStream аналитики  
Использование ClickHouse как основного хранилища

Ростелеком – Создание клиентского профиля  
1.5 TB/day  
Uid matching  
Внедрение ML моделей в Production  
Решение проблем производительности

## 3. Суммарный опыт работы с Big Data tools 9 лет

4. В 2022 команда Data Office была 6 человек  
Текущий размер команды 25 человек

# Группа компаний «Детский мир» сегодня

- 3 страны присутствия
- более 1300 магазинов
- более 440 городов
- более 20 тысяч сотрудников
- 6 распределительных центров
- более 900 тысяч SKU онлайн-ассортимента
- более 600 млн онлайн-посещений ежегодно
- более 250 млн посещений магазинов в год
- 12+ млн пользователей мобильного приложения
- более 42 тысяч пунктов выдачи
- более 7000 поставщиков маркетплейса

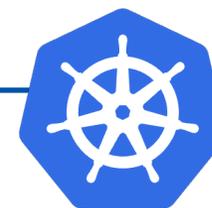
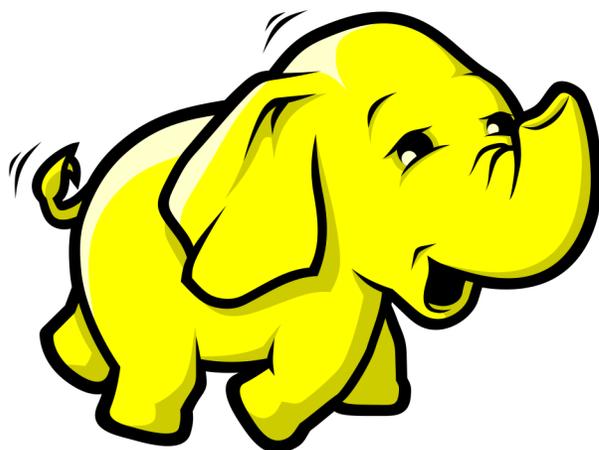




# 02 Начало BigData в DM

# Детский мир BigData 2022

Источники  
данных



# Omni Dashboard / CEO

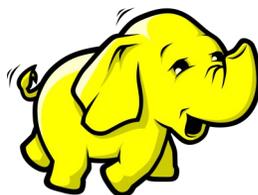
## Источники данных



## Загрузка



## Хранение



## Манипуляции с данными



## Оркестрация



# ОМНИ-ДАШБОРД

08 февраля 2023

## GMV

RUB, млн



Бюдж.

98 %  
105 %  
102 %

YoY

6 %  
12 %  
11 %



Д  
М  
Г

## Фронт-маржа

Ф.-маржа

18,5 %  
19,7 %  
19,1 %

Бюдж.

(1,2 пп)  
(0,2 пп)  
(0,6 пп)

Бюдж., RUB

92 %  
103 %  
99 %

YoY

8,8 пп  
8,3 пп  
6,1 пп



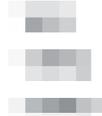
Д  
М  
Г

## GMV онлайн

Доля от общ. GMV

34,3 %  
31,8 %  
28,7 %

RUB, млн



Бюдж.

88 %  
93 %  
101 %

YoY

(10 %)  
(8 %)  
(4 %)



Д  
М  
Г

## Фронт-маржа онлайн

Ф.-маржа

7,5 %  
7,4 %  
7,7 %

Бюдж.

(3,7 пп)  
(3,8 пп)  
(2,8 пп)

Бюдж., RUB

67 %  
70 %  
83 %

YoY

7,9 пп  
8,5 пп  
6,5 пп



Д  
М  
Г

# Omni Dashboard / C Level Reporting / Version 1

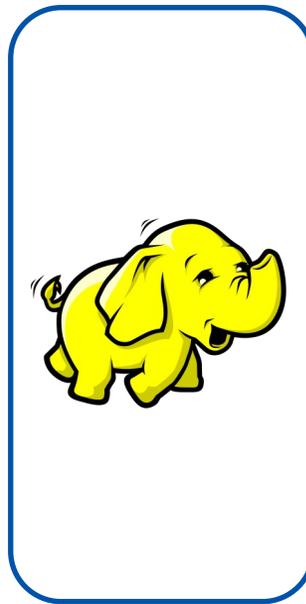
Источники  
данных



Загрузка



Хранение



Манипуляции  
с данными



Визуализация  
данных



Оркестрация

# Omni Dashboard / C Level Reporting / Version 1

## Переработка SQL-запроса для загрузки в Qlik Sense.



### Описание

Редактировать описание

### Загрузка новых SQL-запросов по замене показателя



### Вложения 7

15 нояб. 2022, 12:11 PM

10 окт. 2022, 07:26

### Описание

DOD

1. Обновить справочник
2. Обновить sql
  - a.
  - b.

### Вложения 6

### Связанные задачи

blocks

## Создать витрину по продажам онлайн



### Описание

Прошу создать новую витрину по продажам онлайн в схеме dashboard. Код формирования витрины во вложении.

Таблицу можно назвать

### Вложения 1

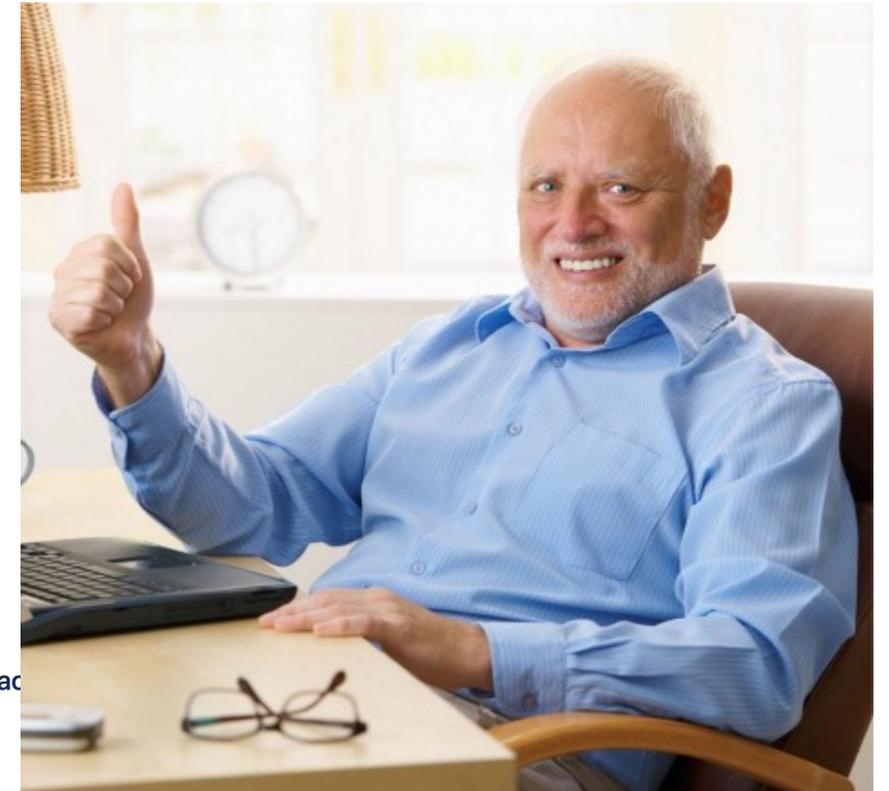
## Создание витрины с данными по товарному запасу



### Описание

Создайте, пожалуйста, новую витрину в Hadoop (схема Dashboard). Код во вложении. Назвать таблицу можно

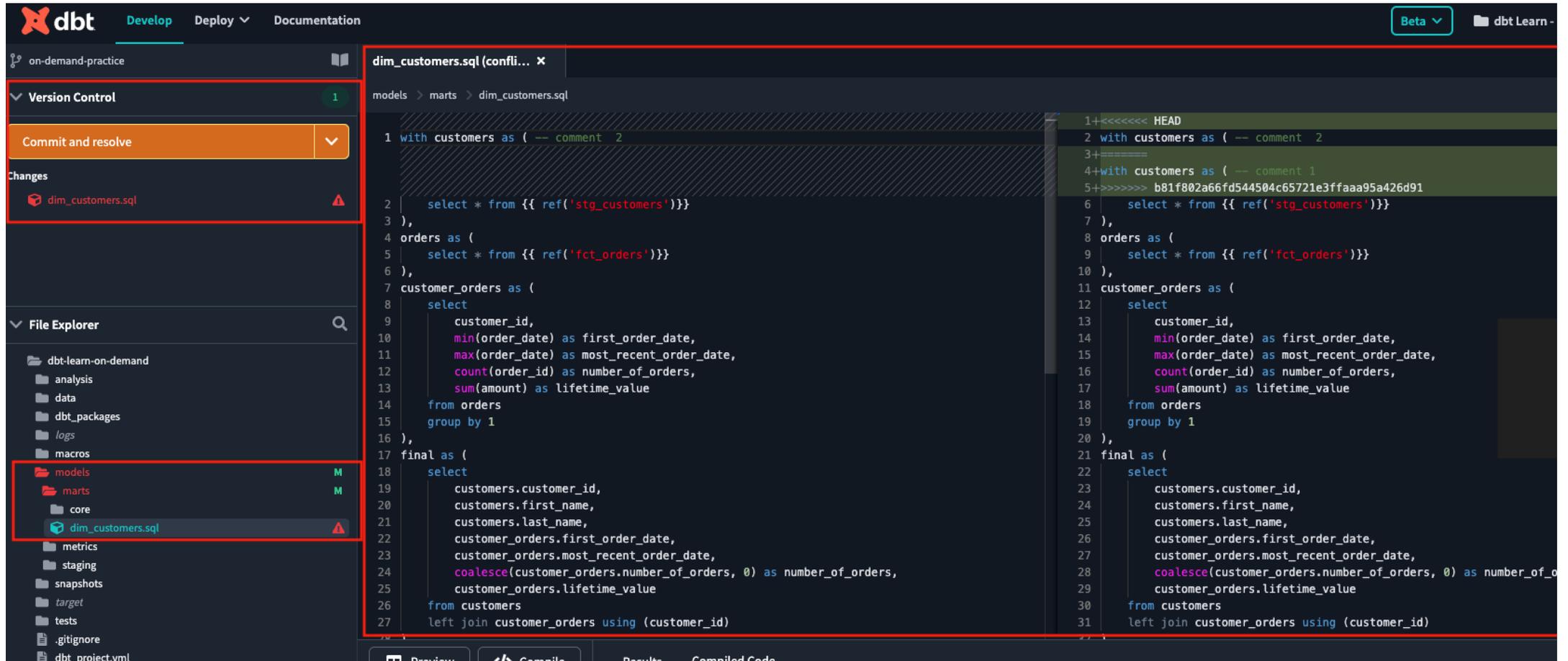
### Вложения 1





# 03 Первые подходы к dbt

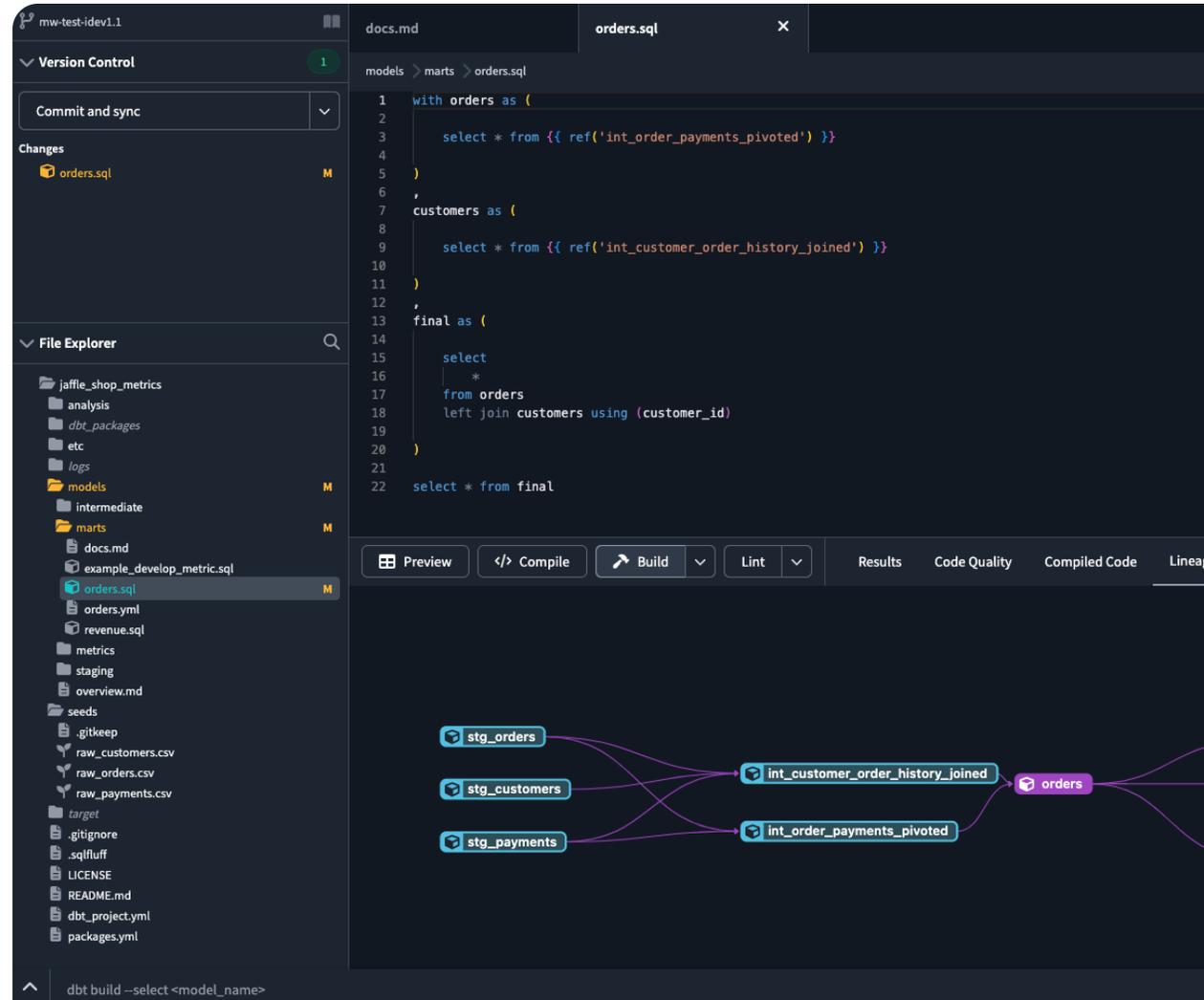
# Коротко о DBT / Git ops хранение



The screenshot displays the dbt CLI interface with the following components:

- Top Bar:** dbt logo, tabs for 'Develop', 'Deploy', and 'Documentation', a 'Beta' dropdown, and a folder icon labeled 'dbt Learn -'.
- Left Sidebar:**
  - Version Control:** Shows 'Commit and resolve' and a list of changes including 'dim\_customers.sql'.
  - File Explorer:** Shows a directory tree with folders like 'analysis', 'data', 'dbt\_packages', 'logs', 'macros', 'models', 'marts', 'core', 'metrics', 'staging', 'snapshots', 'target', 'tests', and files like '.gitignore' and 'dbt\_project.yml'. The 'models' and 'marts' folders are highlighted.
- Central Editor:** Displays the SQL code for 'dim\_customers.sql' with line numbers 1 through 27. The code includes CTEs for 'customers', 'orders', and 'customer\_orders', and a final 'select' statement.
- Right Panel:** Shows a diff view comparing the current code with the HEAD version. It highlights changes in the 'with customers as' and 'customer\_orders as' sections.
- Bottom Bar:** Contains buttons for 'Preview', 'Compile', 'Results', and 'Compiled Code'.

# Коротко о DBT / Построение зависимостей



Скриншот IDE, демонстрирующий код SQL для модели `orders` и диаграмму зависимостей. В центре экрана отображен код SQL:

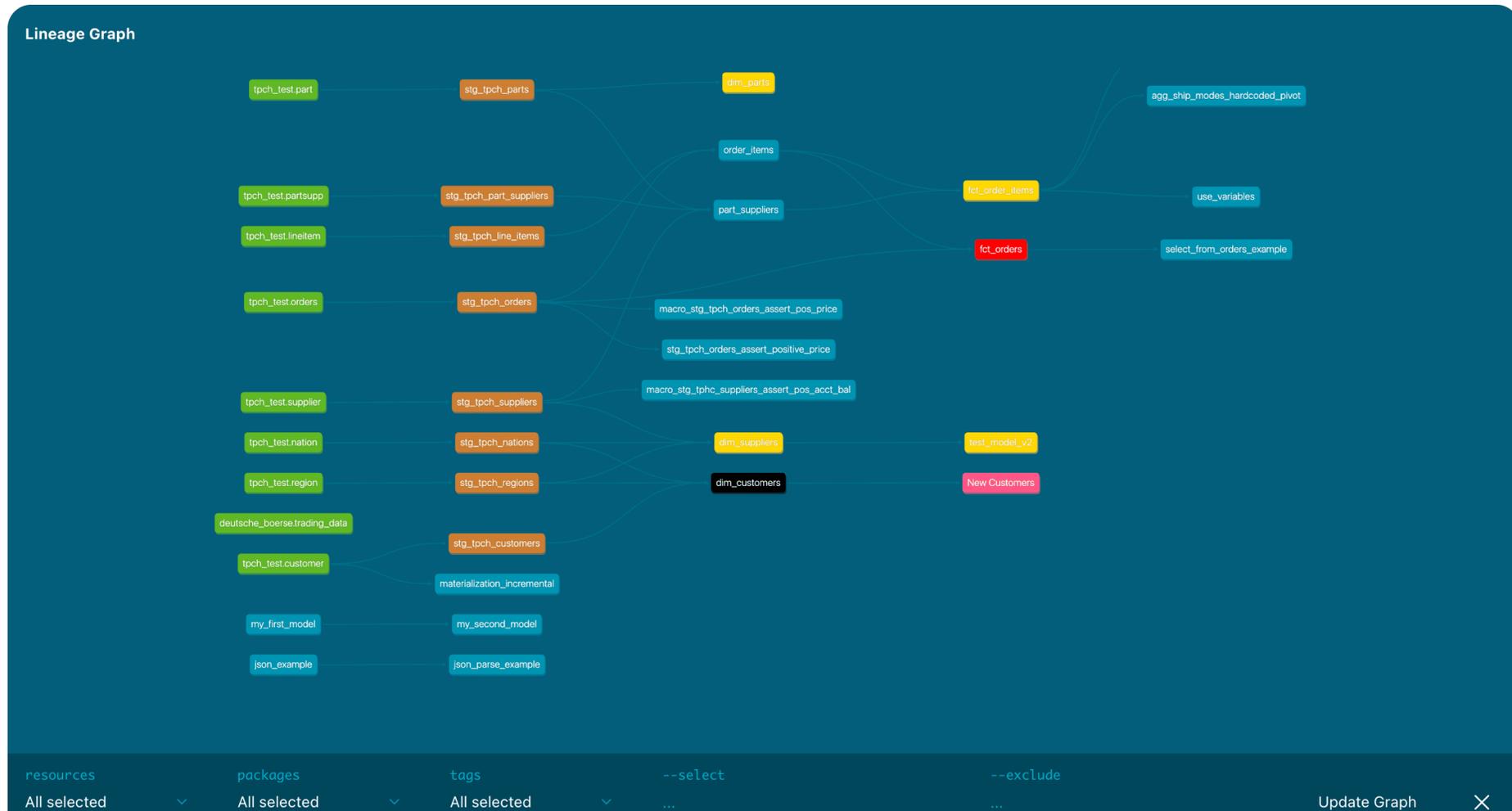
```
1 with orders as (  
2  
3     select * from {{ ref('int_order_payments_pivoted') }}  
4  
5 )  
6  
7 ,  
8 customers as (  
9     select * from {{ ref('int_customer_order_history_joined') }}  
10  
11 )  
12  
13 final as (  
14  
15     select  
16         *  
17     from orders  
18     left join customers using (customer_id)  
19  
20 )  
21  
22 select * from final
```

В нижней части экрана отображена диаграмма зависимостей (Dependency Graph), показывающая зависимости модели `orders` от других моделей:

- `stg_orders`, `stg_customers` и `stg_payments` зависят от `int_customer_order_history_joined`.
- `stg_payments` и `stg_payments` зависят от `int_order_payments_pivoted`.
- `int_customer_order_history_joined` и `int_order_payments_pivoted` являются промежуточными моделями, которые зависят от исходных данных.
- Модель `orders` зависит от `int_customer_order_history_joined` и `int_order_payments_pivoted`.

В левом меню видна структура проекта, включая папки `models`, `staging` и `seeds`. В нижней панели терминала отображена команда: `dbt build --select <model_name>`

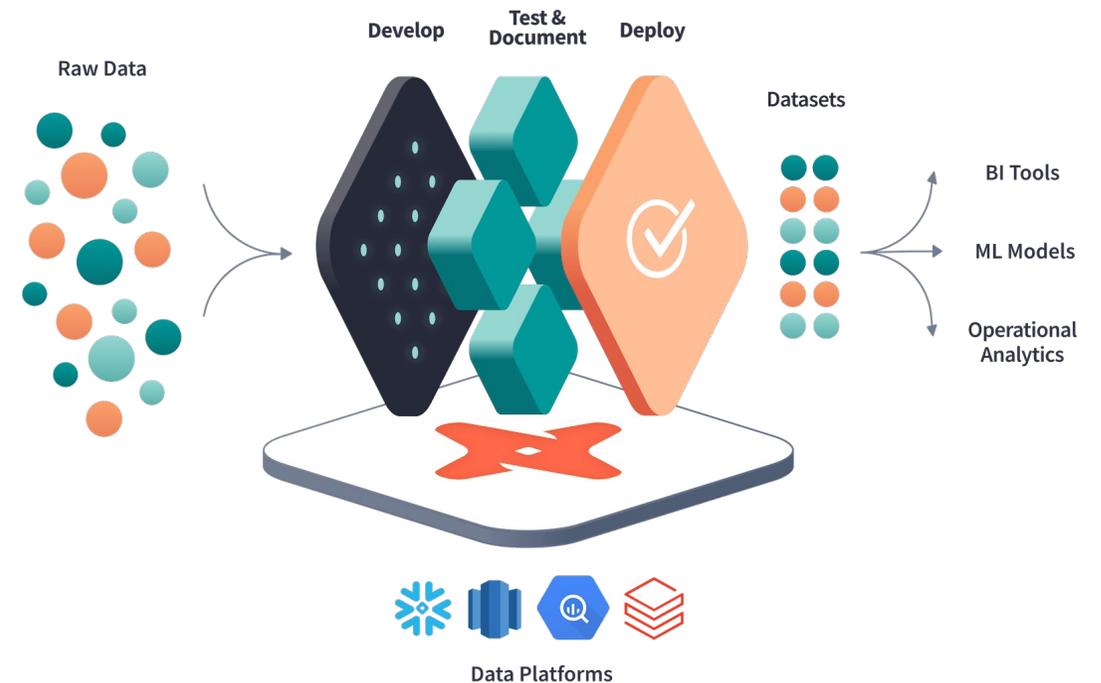
# Коротко о DBT / Data lineage





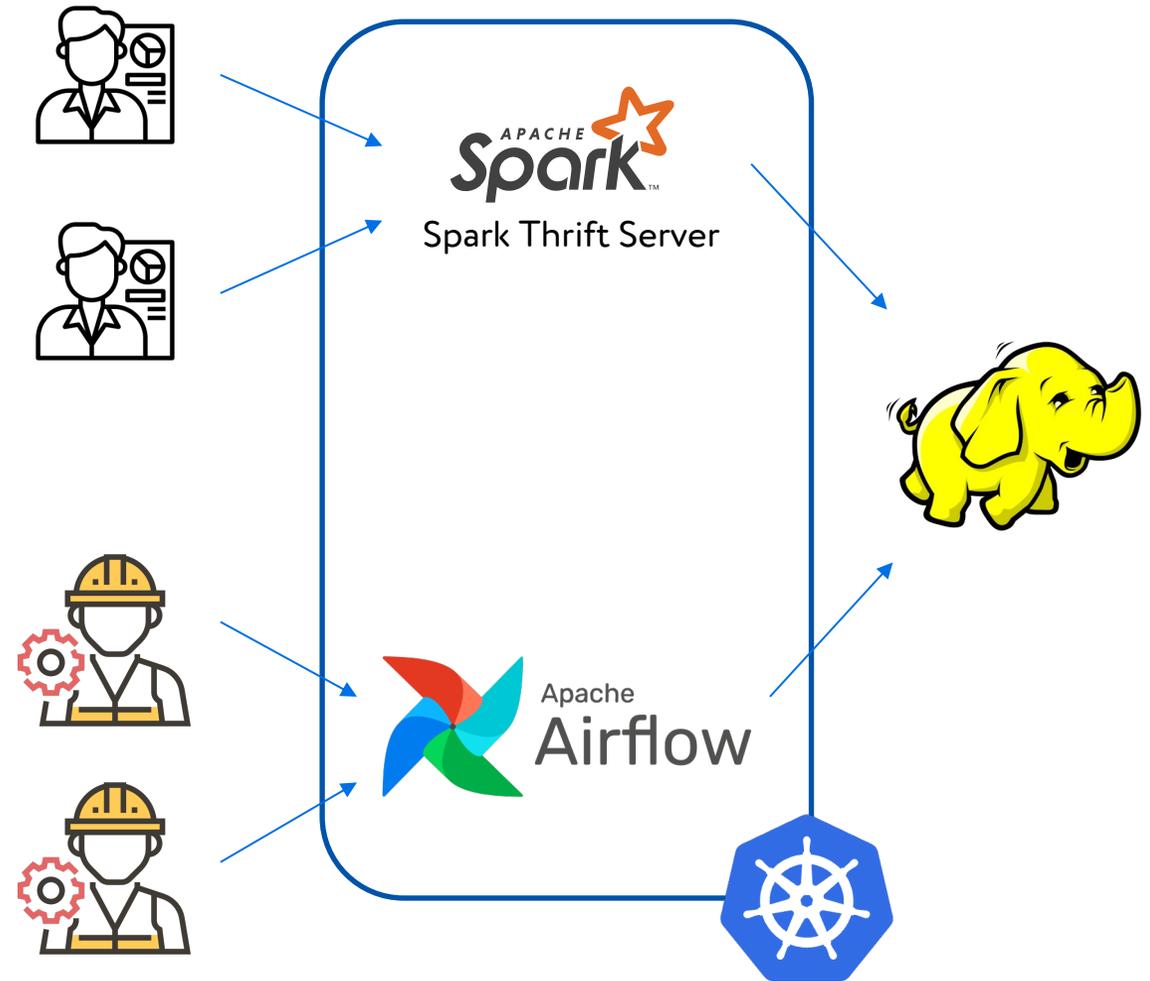
# DBT / Все вместе

- SQL без порядка это больно
- Добиться порядка в SQL сложно
- Витрины данных по методологии с GitOps
- Порядок из коробки
- Значительное ускорение Time To Market
- Поиск зависимостей в витринах данных
- Довольные Аналитики и Инженеры



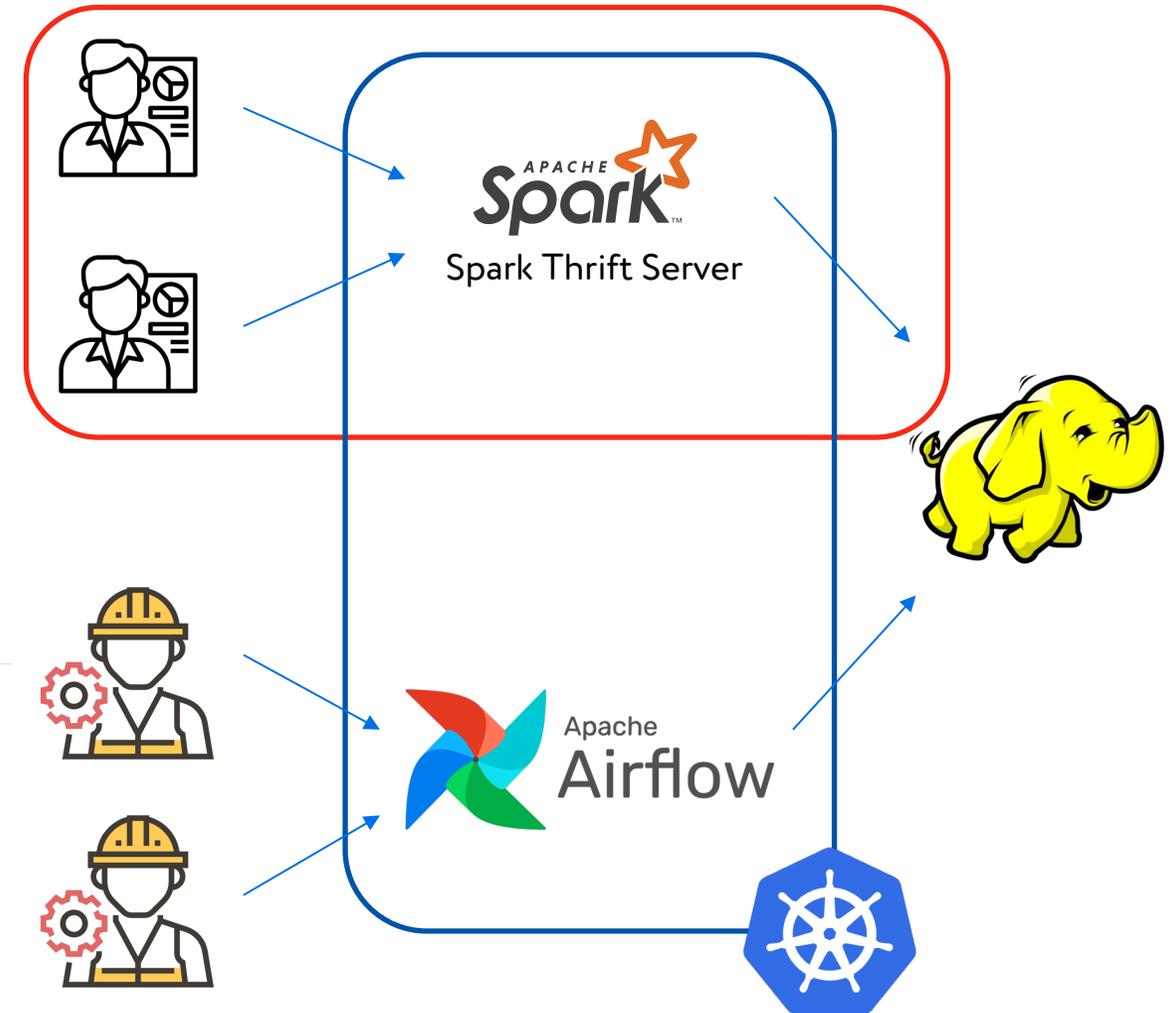
# DBT / Про стек

- Наш стек
  - Hadoop
  - SparkThriftServer
  - Airflow in K8S



# DBT / работа с Hadoop для аналитиков

- Есть адаптер dbt-spark
- Все хорошо заработал сразу
- Из винды не запускаются построения
- Для этого сделали в GitLab пайплайн который запускает построение определенной модели



dbt\_analyst / Pipelines / Run pipeline

## Run pipeline

Run for branch name or tag

master

Variables

Variable DBT\_SELECT exchangerates.sql

Что будем строить

Variable DBT\_TARGET dev

Определение куда деплоить результат

Variable Input variable key Input variable value

Specify variable values to be used in this run. The variables specified in the configuration file as well as [CI/CD settings](#) are used by default.

Variables specified here are **expanded** and not **masked**.

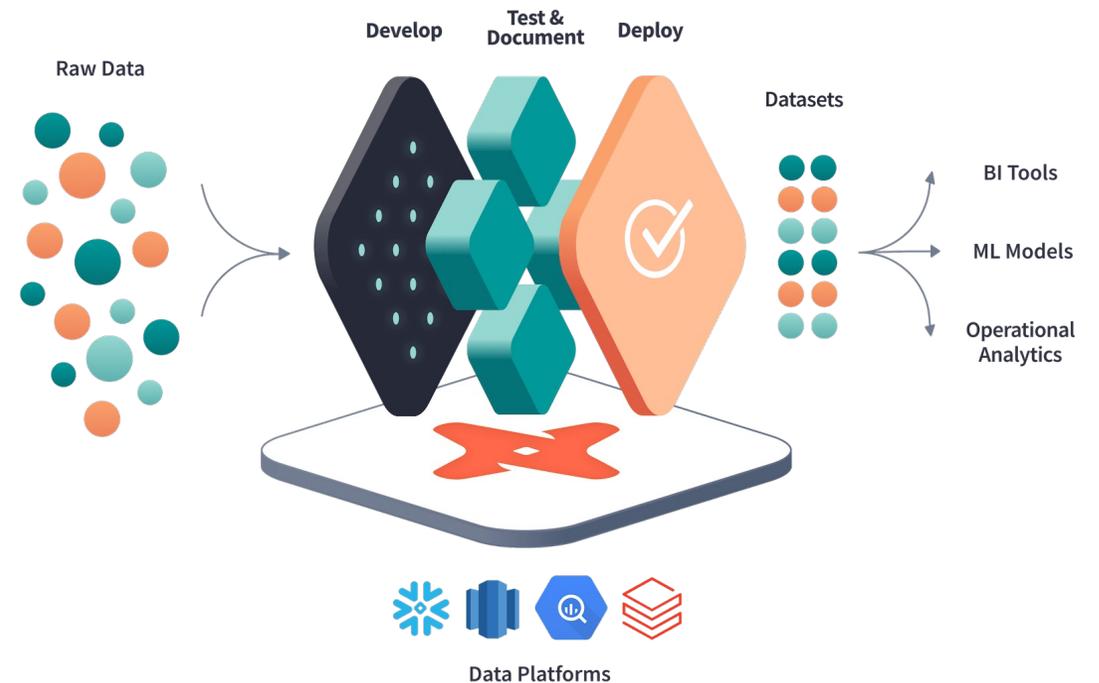
Run pipeline Cancel

# DBT / Генерация документации

Генерируем доку  
Складываем в GitLab Pages



```
pages:  
  stage: deploy  
  cache:  
    - key: target  
      paths:  
        - public  
  variables:  
    DBT_PROFILES_DIR: "${CI_PROJECT_DIR}/omni_1"  
  script:  
    - cd omni_1  
    - dbt debug -t prod  
    - dbt deps -t prod  
    - dbt docs generate -t prod  
    - mkdir -p ${CI_PROJECT_DIR}/public/$CURRENT_CONTENT_PATH  
    - cp -r ${CI_PROJECT_DIR}/omni_1/target/*  
    ${CI_PROJECT_DIR}/public/$CURRENT_CONTENT_PATH  
  artifacts:  
    paths:  
      - public  
    expire_in: 4 weeks
```



# DBT / Интеграция в airflow



```
def load_manifest():  
    local_filepath = f"{DBT_PROJECT_DIR}/target/manifest.json"  
    with open(local_filepath) as f:  
        data = json.load(f)  
    return data
```

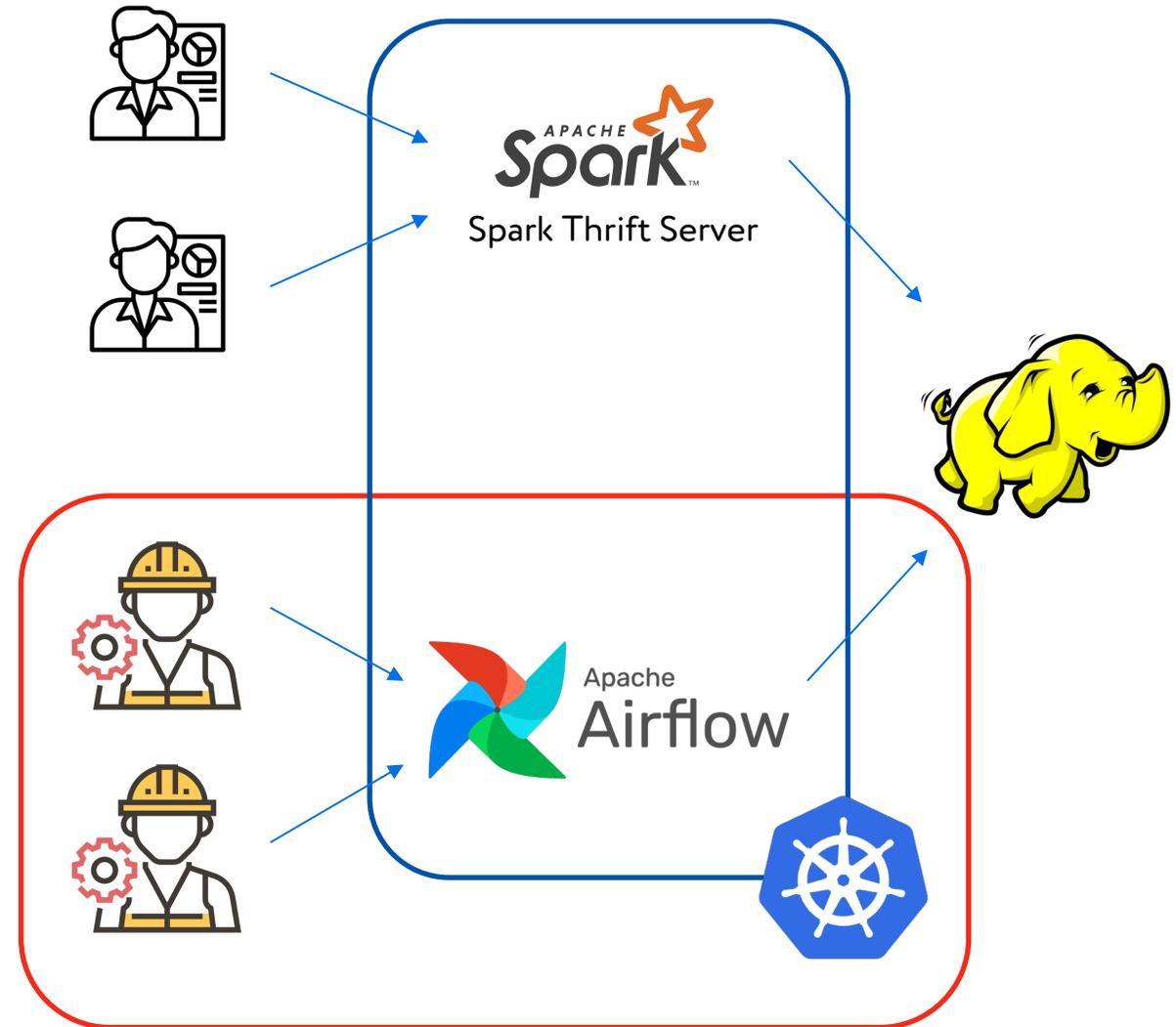
- На 2022 был только cosmos в зачаточном состоянии
- На 2024 есть
  - Astronomer cosmos
  - Toloka / dbt-af

# DBT / работа с Hadoop для инженеров

- не подходит sparkSqlOperator
  - Из за запуска в кубере сеть виртуализирована, еxecutor не могут общаться с драйвером
- Dbt создает артефакты
  - manifest.json
    - построение графа
    - содержит мету о тегах
  - Скопированный sql
    - Запуск с помощью спарксабмит

```
val source = Source.fromFile(queryPath.mkString.split("/").last, "UTF-8")  
val query = try source.mkString finally source.close()
```

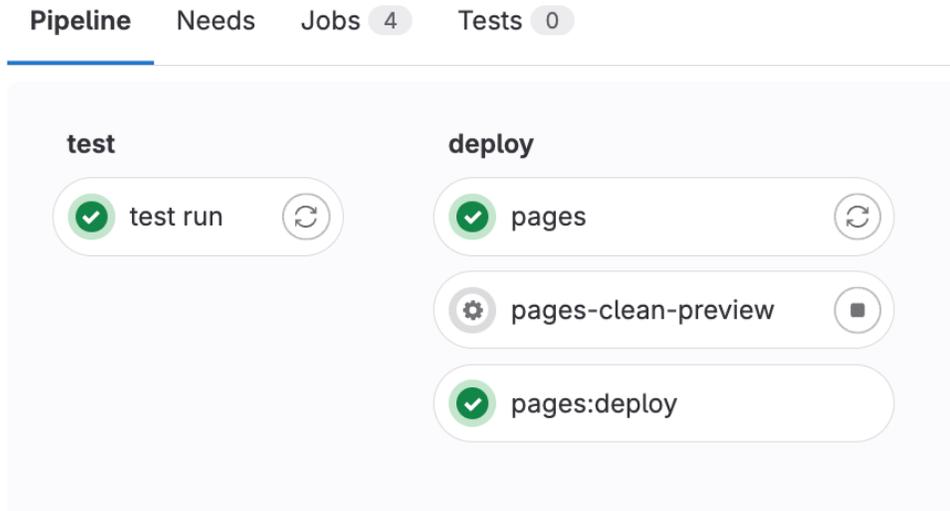
```
spark  
  .sql(query)  
  .repartition(numPartitions.toInt)  
  .write  
  .mode(SaveMode.Overwrite)  
  .format("parquet")  
  .saveAsTable(resultTable)
```





# 04 Организация CI/CD для dbt

# CI/CD для dbt репо



```
test run:
stage: test
image: dbt:rel_1.8.4_u1
variables:
  DBT_PROFILES_DIR: "${CI_PROJECT_DIR}/omni_1"
script:
  - python validation/validation.py --models_directory
    ${CI_PROJECT_DIR}/omni_1/models
  - cd omni_1
  - dbt debug -t dev_view --project-dir ../ci_cd
  - dbt deps -t dev_view --project-dir ../ci_cd
  - dbt compile -t dev_view --project-dir ../ci_cd
  - dbt -x ls -t dev_view --project-dir ../ci_cd
  - dbt run -t dev_view --project-dir ../ci_cd
  - dbt test -t dev_view --project-dir ../ci_cd
```

- Нужно что аналитики сразу знали фидбек на свои изменения
- Организовано построение вьюх
  - понимаем, что витрины валидны технически и мы можем их ставить на исполнение
- В шаг CI/CD добавляем:
  - Защиту от прямого использования таблиц
  - Если не используется `{{ref}}` `{{source}}`, падаем



# 05 Подготовка Аналитиков

# Рассказываем о новом процессе

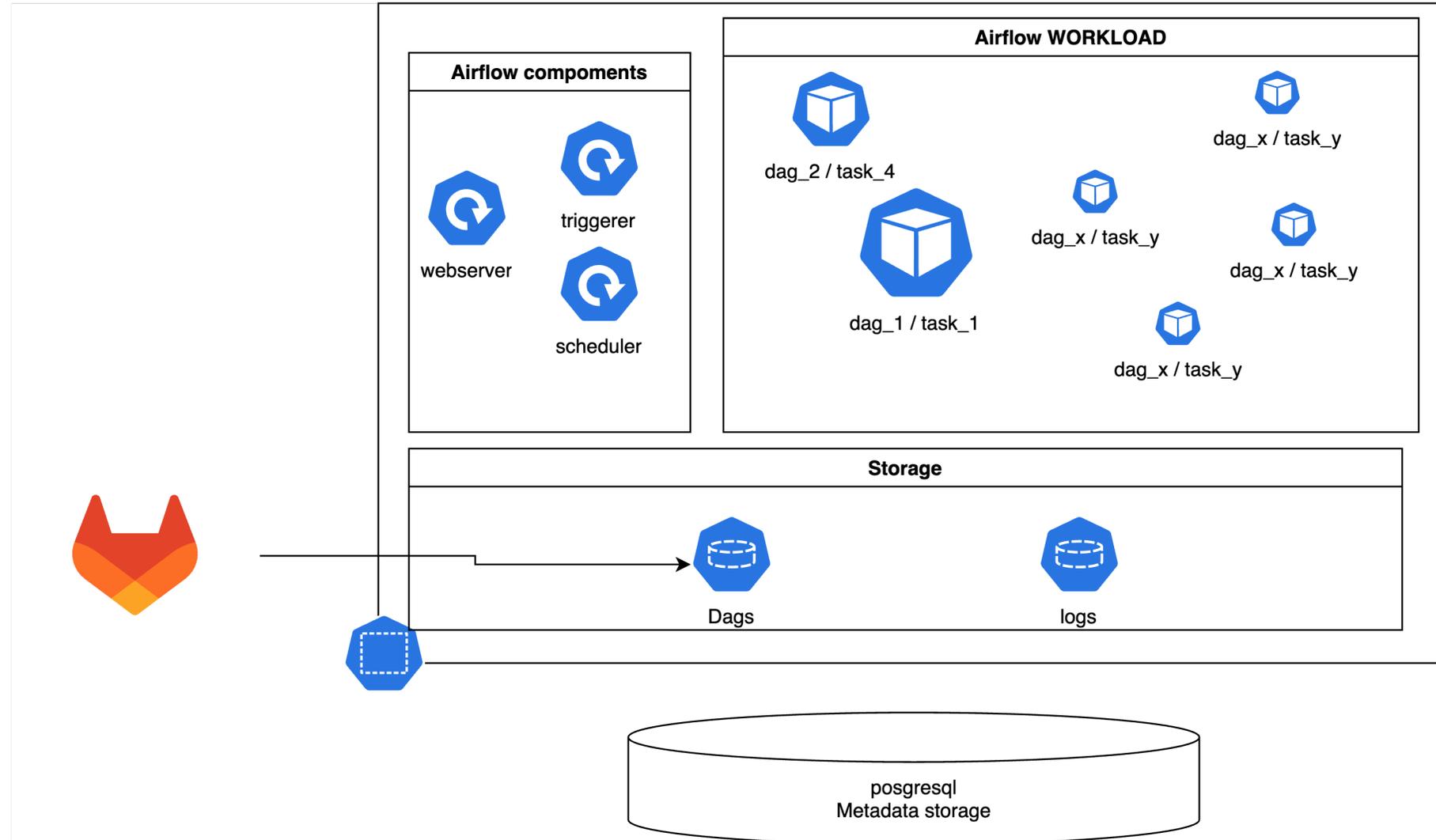


- Вспоминаем, для кого мы все это делали
- Заранее проверяем хотят ли аналитики этих оптимизаций
- Презентуем изменения до готовности
- Обучение git в формате workshop
- Запирались в переговорке и учились делать коммиты в ветки
- Настройки локальных окружений
- Настраиваем одному, чтобы он рассказал остальным



# 06 Деплой в Airflow

# Airflow k8s architecture



# Airflow repo structure

- > .gitlab
- > \_archive
- ✓ airflow
  - > dashboard\_notifier
  - > dashboard\_notifier\_monthly\_statistic
  - > dashboard\_notifier\_sla\_report
  - > dbt\_cvm
  - dbt\_omni**
  - > export\_gogift
  - > export\_kafka\_2
  - > hadoop\_tech\_operations
  - ✓ import\_catalog\_service
    - conf
      - dev.json
      - prod.json
    - import\_catalog\_service.py
  - > util
- ✓ airflow\_test
  - external\_sensor\_test.py
- > dags\_conf
- ✓ image\_req
  - af\_262.txt
  - af\_284.txt
- .gitignore
- .gitlab-ci.yml
- .gitmodules
- .pre-commit-config.yaml
- deploy\_dag.sh
- Makefile
- pyproject.toml
- README.md

- dbt\_omni
  - conf
    - dev.json
    - prod.json
  - dbt\_omni
    - ci\_cd
    - docker
    - omni\_1
    - pip\_deps
    - preprocessing
    - validation
  - .gitignore
  - .gitlab-ci.yml
  - .pre-commit-config.yaml
  - README.md
  - dbt\_1.py
  - dbt\_full.py

Overview 0 Commits 1 Pipelines 1 **Changes 1**

Compare master and latest version

Search (e.g. \*.vue) (%P)

- airflow/dbt\_omni
  - dbt\_omni** +1 -1

master dbt\_analyst Author Search by message

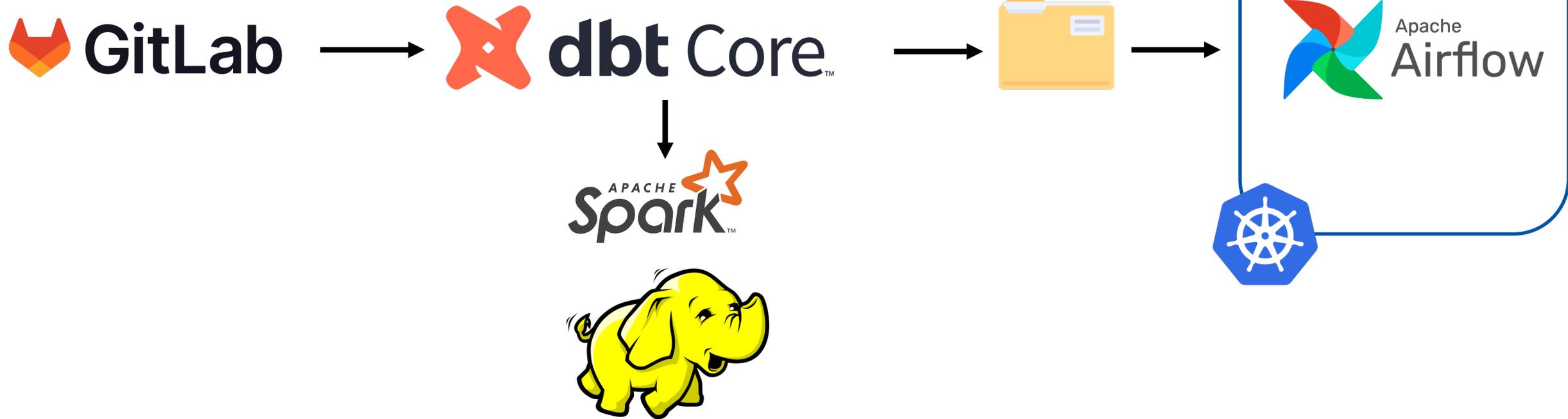
airflow/dbt\_omni/dbt\_omni @ 4f4e5d63

1	- Subproject commit	1	+ Subproject commit
	a8ab429f80be25460637a6e702c855865a904a93		4f4e5d63026e87466289779e73054b187c7bcb15

Commit list on the right:

- 4f4e5d63
- eaab47380
- d13bf962
- a8ab429f
- c9e01dc0
- a8a84c91
- 42c70754
- 89bfe10f
- dc1451a4
- bdc03caa

# Subrepository model



# Omni Dashboard / C Level Reporting / Version 2

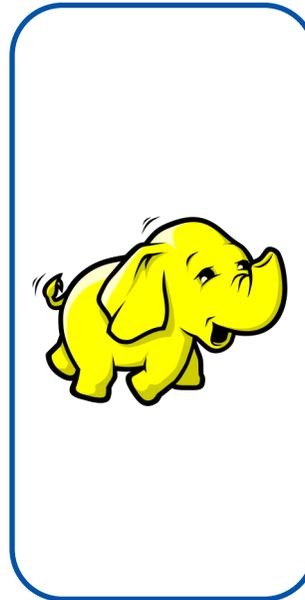
Источники  
данных



Загрузка



Хранение



Манипуляции  
с данными



Визуализация  
данных



Оркестрация

# Сравнение процессов разработки витрин

Поток доставки задачи как изменился

## Было

DA пишет 1-3 sql витрины

Отправляет их DE в задаче в jira

DE начинает проверять sql

DE разбирается, что за чем идет

DE строит view на базе витрин

DE пишет большой даг файл с нашими sql



## Стало

DA пишет 1-3 sql витрины

Все витрины строятся автоматически в ci/cd,

сразу понимает что витрины валидны

DA создает Merge Requets

DE начинает смотреть, что изменилось в SQL

DE ставит апрув и мержит

DE обновляет dbt в airflow



# 07 Новые вызовы

# Разделение на легкие/тяжелые витрины

- Сделали отдельные пулы
- Разные конфиги ресурсов
- Зашедуленные задачи убивались по непонятным причинам и причем тут AIRFLOW\_SCHEDULER\_TASK\_QUEUED\_TIMEOUT

List Pool

Search ▾

+ Actions ▾ ← Record Count: 16

<input type="checkbox"/>	Pool ↓	Description ↓	Slots ↓	Running Slots	Queued Slots	Scheduled Slots	Deferred Slots
<input type="checkbox"/>	dbt_omni	dbt_omni exec limit	2	2	0	9	0
<input type="checkbox"/>	dbt_omni_light	dbt_omni easy task exec limit	4	0	0	0	0



# Витрины все нужны, но не одинаково важны

Когда готовы все сенсоры что делать в первую очередь  
Сделали приоритезацию по тегам в моделях  
Airflow имеет уже готовый функционал `priority_weight`

```
tag_weights = {  
    "aaa": 8,  
    "b": 7,  
    "c": 6,  
    "d": 5,  
    "e": 4,  
    "other": 3,  
    "project_100": 2  
}
```

```
dbt_task = SparkSubmitOperator(  
    task_id=node,  
    name=f"{DAG_ID} / {target_table} / " + "{{ ds }}",  
    pool=pool,  
    priority_weight=task_weight,  
    weight_rule="absolute",  
)
```

		Impact		
		High	Medium	Low
Urgency	High (Prevents primary work functions)	High System-wide	Medium Multiple users, not an entire department	Low Single user
	Medium (Some work functions impaired, workaround in place)	Critical	High	Moderate
	Low (Minor inconvenience)	Moderate	Moderate	Low

# Инкрементальные витрины

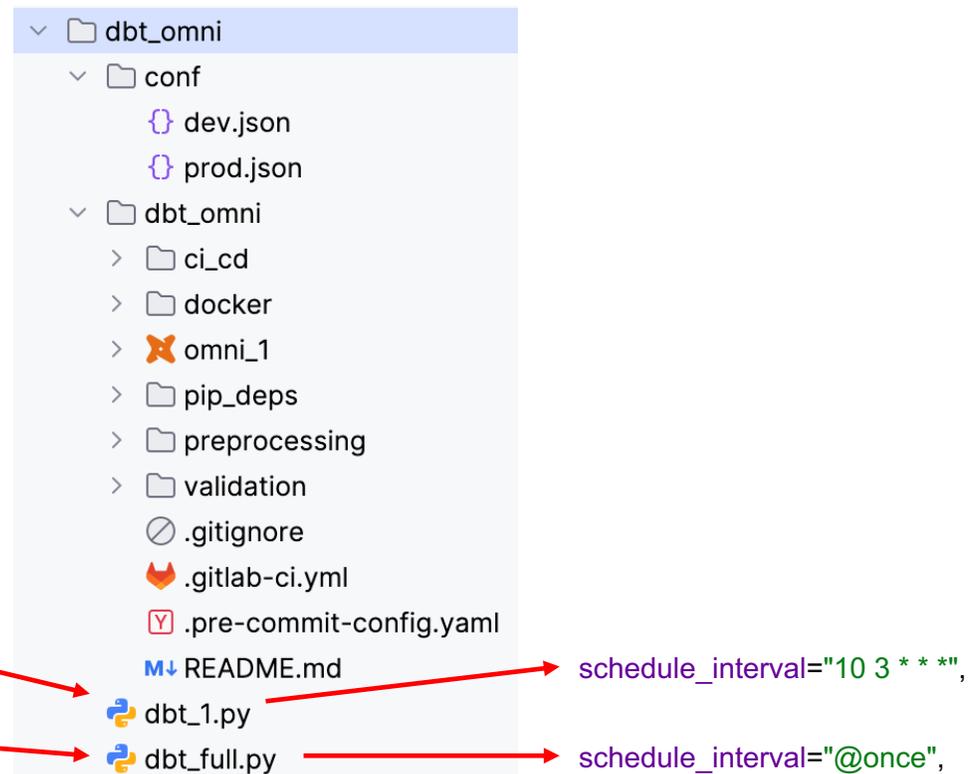
```
{{ config(tags = ['aaaa'],  
  spark_materialized='incremental',  
  spark_partition_fields= ["calday"]) }}
```

```
SELECT a.*,
```

```
{% if var("is_incremental") | as_bool %}  
explode((sequence(current_date() - 7, current_date(), interval 1 day))) calday  
{% else %}  
explode((sequence(to_date('2023-01-01'), current_date(), interval 1 day))) calday  
{% endif %}
```

```
FROM (SELECT DISTINCT a,  
  b,  
  c  
  from {{ source('scheme', 'table') }}) a
```

- Dbt is\_incremental() не подходит, живет своей жизнью
- Сделали отдельный sparkSubmit
  - принимает колонку партиционирования
  - А так же режим выполнения full или incremental
- Создаем 2 дага
  - Основной для инкрементальных дневных расчетов
  - Дополнительный даг надо руками триггерить для полного пересчета изменённых моделей



A screenshot of a file explorer showing the directory structure for 'dbt\_omni'. The directory contains several subfolders and files:

- conf
  - dev.json
  - prod.json
- dbt\_omni
  - ci\_cd
  - docker
  - omni\_1
  - pip\_deps
  - preprocessing
  - validation
- .gitignore
- .gitlab-ci.yml
- .pre-commit-config.yaml
- README.md
- dbt\_1.py
- dbt\_full.py

Red arrows point from the text in the list to the corresponding files in the screenshot:

- From 'Основной для инкрементальных дневных расчетов' to 'dbt\_1.py'
- From 'Дополнительный даг надо руками триггерить для полного пересчета изменённых моделей' to 'dbt\_full.py'
- From 'schedule\_interval="10 3 \* \* \*"' to 'README.md'
- From 'schedule\_interval="@once"' to 'dbt\_full.py'

# Долгое CI/CD

Каждый коммит в airflow это проверка моделей в dbt  
Проверка идет несколько минут



Написали скрипт для кеширования артефактов  
Если когда данных коммит был собран то его надо собирать  
Функционал уже есть в Gitlab

Во втором проекте DBT созданий view не отрабатывает за час  
В коде витрин много CTE



схема копия прода но с небольшим куском данных  
Создаем не view а настоящую таблицу

Но в первом проекте оказалось что быстрее работает создание view

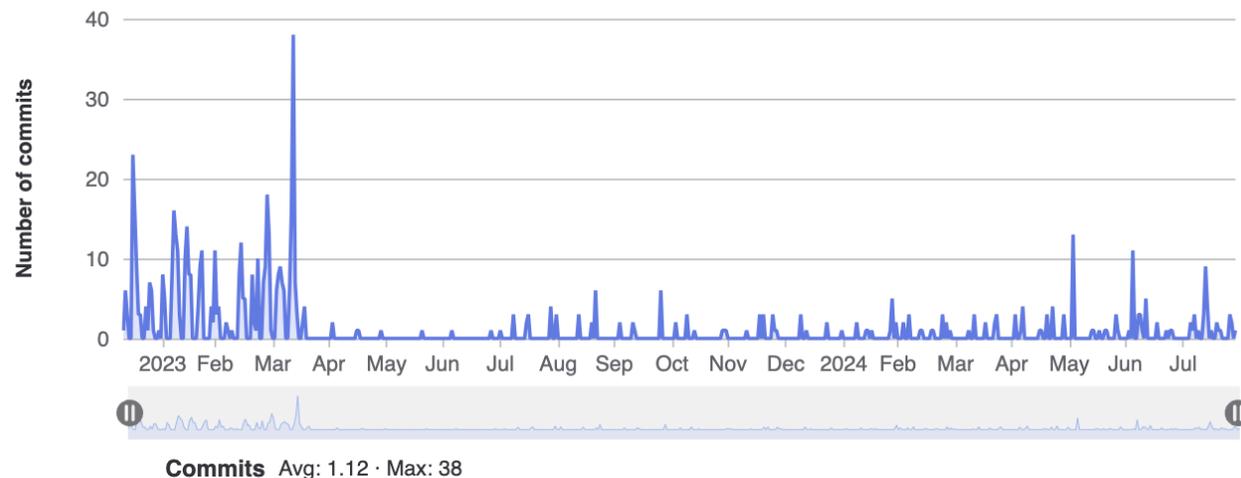


# 08 Подводим итоги

# Витрины спустя 2 года

## Commits to master

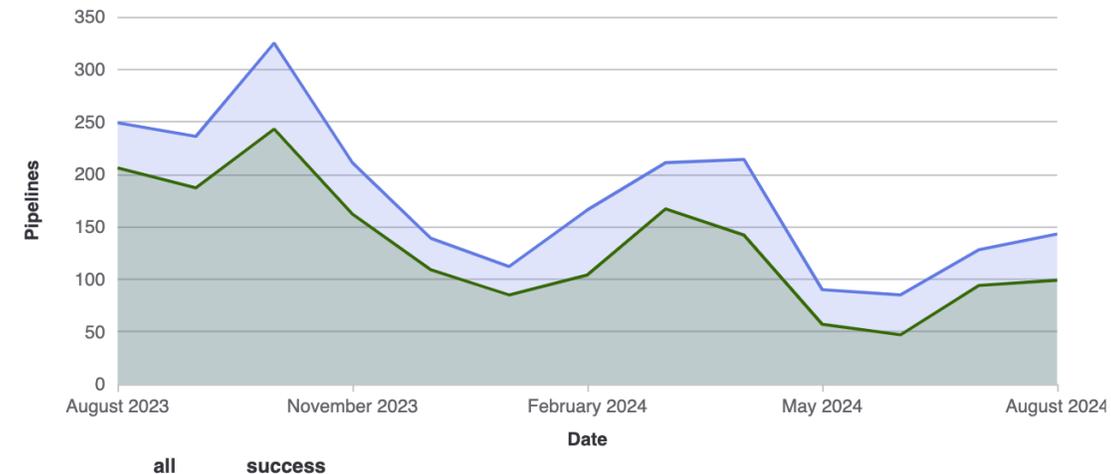
Excluding merge commits. Limited to 6,000 commits.



## Pipelines charts

Last week Last month Last year

Date range: Sep 01, 2023 - Aug 31, 2024



## Высокий темп изменений

- project\_1
  - Finished running 85 view models in 0 hours 11 minutes and 31.05 seconds (691.05s).
- project\_2
  - Finished running 33 table models, 8 view models in 0 hours 14 minutes and 16.70 seconds (856.70s).
- project\_3
  - Found 106 models, 6 sources

# О команде 2022

6 человек

OMNI

Snowplow (Уход  
от Google  
Analytics)

Team Lead / 1

DA / 1

DA / 1

Head DE / 1

DE / 1

DE / 1

# О команде 2024

25 человек

