Itera Data Platform

Функциональные характеристики

Оглавление

1 Общие положения

1.1 Наименование программы

Полное наименование программы: Itera Data Platform.

Краткое наименование программы: IDP.

1.2 Назначение документа

В настоящем документе представлено описание функциональных возможностей Itera Data Platform, а также подробное руководство по работе с системой.

Документ входит в комплект эксплуатационной документации по Itera Data Platform и предназначен для пользователей системы.

1.3 Разработчик системы

Полное наименование: Общество с ограниченной ответственностью «ИТЕРА».

Сокращенное наименование: ООО «ИТЕРА».

1.4 Контакты технической поддержки

тел.: +7 913 741-73-88
e-mail: info@4dwh.ru

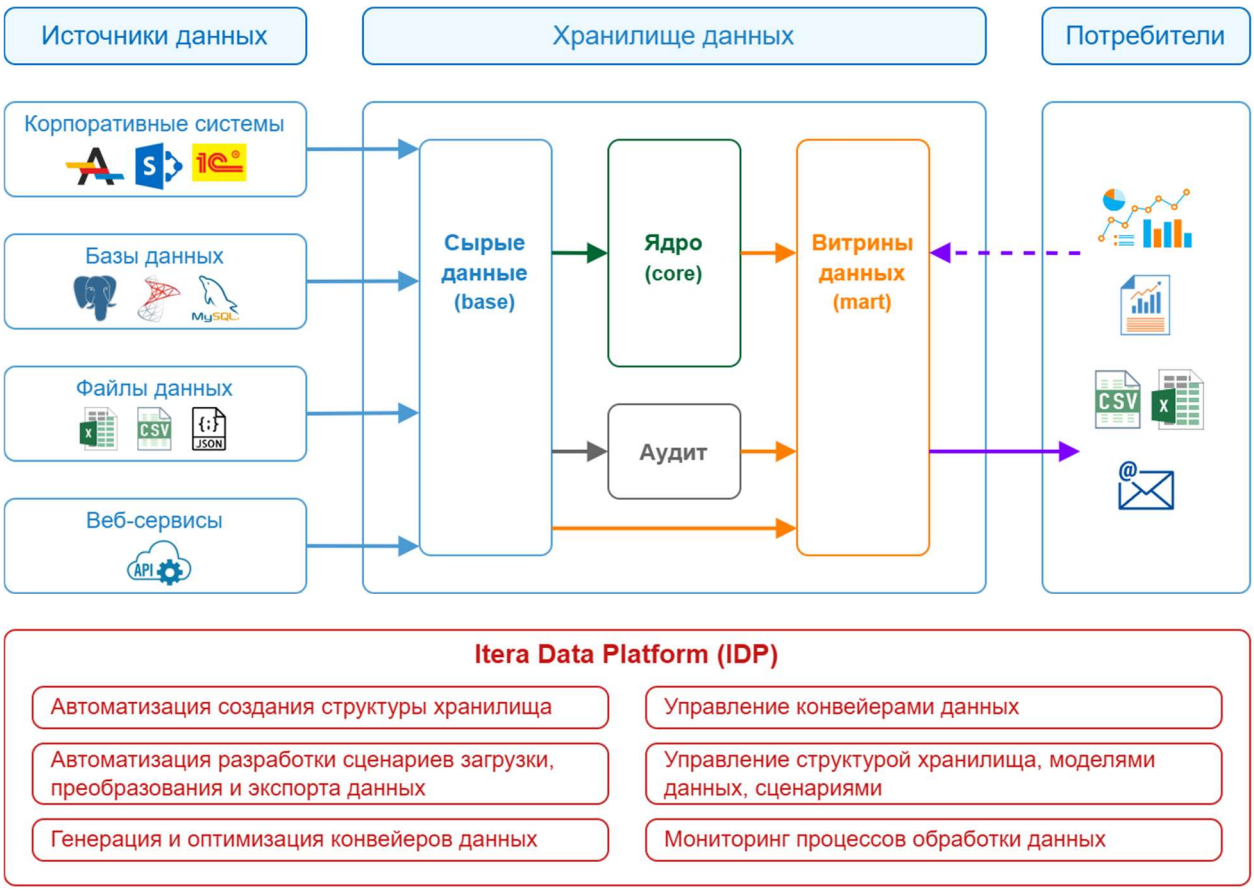
2 Функциональное описание

Itera Data Platform (IDP) – ETL-решение, включающее набор инструментов для решения задач построения корпоративного хранилища данных, загрузки данных в хранилище из источников, преобразования данных и построения витрин для отчетов, администрирования хранилища и мониторинга процессов обработки данных, экспорта и отправки отчетов в различные каналы.

IDP поддерживает работу с разными видами источников данных: корпоративные информационные системы (1С, Адванта), базы данных, файлы данных, веб-сервисы.

Основная задача, которую позволяет решить реализованный в IDP подход, – максимально снизить нагрузку на системы-источники данных и при этом обеспечить скорость доставки данных в отчеты пользователей, сопоставимую с прямым запросом BI-инструментов к системам-источникам данных.

Используемый подход предполагает организацию в хранилище данных многоуровневой архитектуры LSA (Layered Scalable Architecture), предусматривающей отдельные области для сырых данных, ядра хранилища и витрин для анализа данных. При этом последовательно выполняются конвейеры данных, включающие операции извлечения данных из источников, загрузки в слой сырых данных, очистки, трансформации и консолидации для целей анализа, формирования витрин для анализа данных средствами визуализации, а также для экспорта и отправки в различные каналы (электронная почта, корпоративный портал, общий диск, внешний ресурс).



Itera Data Platform (IDP)

- Автоматизация создания структуры хранилища
- Управление конвейерами данных
- Автоматизация разработки сценариев загрузки, преобразования и экспорта данных
- Управление структурой хранилища, моделями данных, сценариями
- Генерация и оптимизация конвейеров данных
- Мониторинг процессов обработки данных

Itera Data Platform предоставляет возможность автоматизации задач на всех этапах:

В рамках первоначального создания системы:

- Создание структуры хранилища: областей хранилища, моделей данных;
- Разработка сценариев обработки данных (скриптов, SQL-запросов), на основе которых в системе выполняются операции извлечения, загрузки, очистки, преобразования и экспорта;

- Разработка конвейеров данных – последовательностей задач извлечения, загрузки, очистки, преобразования и экспорта, согласованных с точки зрения зависимостей между сущностями.

В рамках автоматического обновления в системе при изменениях в источниках данных:

- Обновление структуры хранилища;
- Обновление сценариев обработки данных;
- Обновление конвейеров данных.

В рамках функционирования системы:

- Управление конвейерами данных и отдельными задачами: запуск/остановка по расписанию, вручную;
- Анализ и изменение процессов обработки данных с целью их улучшения;
- Анализ и изменение моделей и сценариев преобразования данных.

2.1 Функциональные возможности

Itera Data Platform предоставляет следующие функциональные возможности:

- **Инициализация структуры хранилища данных**
  - Инициализация областей хранилища данных:
    - Сырые данные,
    - Ядро,
    - Витрины для анализа данных,
    - Витрины для аудита данных,
    - Витрины для экспорта данных.
  - Автоматическое создание структуры хранения загруженных и преобразованных данных.
  - Автоматическое обновление структуры хранения загруженных и преобразованных данных при изменении структуры данных в источниках.
- **Подключение к источникам данных и загрузка данных в хранилище**
  - Поддержка разных видов источников данных: базы данных, файлы данных, веб-сервисы.
  - Интеграция с коннекторами для высокоскоростной загрузки данных из корпоративных систем:
    - 1С,
    - Адванта.
  - Возможности при работе с файловыми источниками данных:
    - Выбор загружаемых диапазонов данных (для загрузки данных из Excel),
    - Отслеживание изменений файлов данных (загрузка только измененных файлов данных).
  - Инкрементальная загрузка данных из реляционных БД (загрузка только измененных данных).
- **Обработка данных в хранилище, подготовка витрин данных**
  - Поддержка шаблонов сценариев обработки данных.

  - Оптимизированный процесс обработки данных: автоматическое построение плана обработки данных с максимальным распараллеливанием выполняемых операций.

- **Управление качеством данных (корректировка, анализ ошибок)**
  - Доступные операции корректировки данных:
    - Удаление дубликатов,
    - Проверка пустых значений.
  - Сохранение сведений об ошибках данных.
  - Подготовка витрин для анализа ошибок данных.
- **Экспорт данных**
  - Регулярная выгрузка данных из хранилища в формате csv либо xlsx.
  - Сохранение файлов выгрузки на корпоративном портале, в локальном или сетевом каталоге либо рассылка по электронной почте.
- **Интеграция с инструментами визуализации**
  - Подготовленные с помощью IDP витрины могут напрямую использоваться любым профессиональным BI-инструментом визуализации данных, включая отечественные или зарубежные аналитические системы (Modus BI, Polymatica, Visiology, Power BI и т.д.), open-source решения (Superset) или собственные разработки.
- **Отображение структуры хранилища и этапов обработки данных**
  - Через веб-интерфейс доступны следующие сведения:
    - Детальная информация по всем источникам данных: параметры каждого источника, названия и типы полей;
    - Детальная информация по всем объектам (таблицы и представления) в разных слоях хранилища данных: параметры каждого объекта, названия и типы полей;
    - Исходные и скомпилированные скрипты обработки данных для каждого объекта хранилища.
- **Управление и мониторинг процессов обработки данных**
  - Через веб-интерфейс доступны операции запуска /остановки конвейеров данных, а также следующие представления:
    - Последовательность операций обработки данных с возможностью просмотра деталей и текущего статуса каждой операции;
    - Подробный журнал выполнения операций с информацией об ошибках выполнения;
    - График динамики скорости выполнения операций.

2.2 Компоненты системы

Itera Data Platform включает 2 модуля: модуль управления хранилищем данных и модуль управления процессами обработки данных.

Модуль управления хранилищем данных

Модуль включает следующие компоненты:

- Библиотека макросов IDP для автоматизации разработки сценариев обработки данных, в том числе:

  - Макросы загрузки и обновления данных из внешних источников (PostgreSQL, MsSQL, файлы, веб-сервисы).

    Макросы загрузки и обновления данных из внешних источников предназначены для автоматизации разработки сценариев сбора данных из источников в область сырых данных и предоставляют общую комплексную логику обработки данных, необходимую при подключении любого нового источника:

    - создание / обновление структуры данных в хранилище,
    - очистка данных (проверка уникальности, типа данных и др.),
    - логика инкрементального обновления (получение и загрузка только измененных данных),
    - и т.д.

  - Макросы преобразования данных в хранилище.

    Данные макросы предназначены для автоматизации разработки сценариев преобразования данных, и содержат общую логику обработки данных, которая необходима при создании или обновлении моделей в хранилище.

  - Вспомогательные макросы – специфические функции, используемые в операциях загрузки, преобразования и экспорта (например, генерация списка полей, транслитерация названий полей, получение параметров из файла конфигурации, преобразования массивов данных и др.).

  Использование заранее подготовленной логики (макросов), а также преднастроенных файлов конфигурации и образцов позволяет существенно сократить время и трудозатраты по настройке хранилища и разработке сценариев обработки данных, а также снижает требования к квалификации специалиста по разработке.

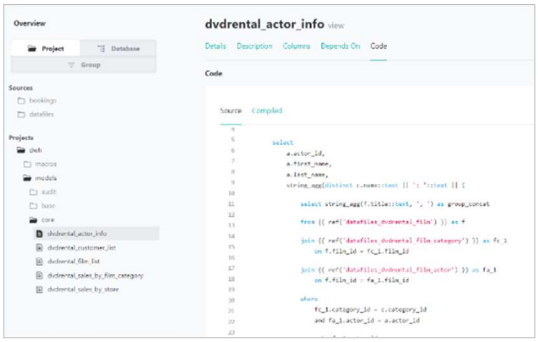
  Например, для добавления таблиц реляционного источника данных потребуется только перечислить таблицы этого источника в конфигурационном файле, а соответствующая структура таблиц в области сырых данных хранилища будет создана автоматически, а также автоматически обновлена при изменении структуры в источнике.

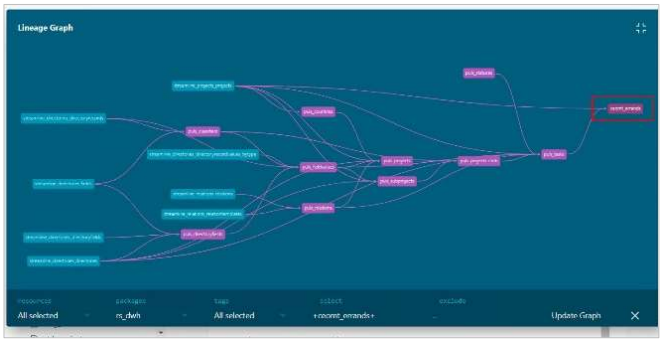
  Для решения основной целевой задачи IDP, минимизации времени доставки данных из источников в отчеты пользователей, нецелесообразно использование визуальных конструкторов для настройки преобразования данных в хранилище, так как участие разработчика в подготовке оптимальных сценариев преобразования данных является необходимым условием. Поэтому при создании решения IDP не стояло задачи предоставить no-code возможности для настройки преобразований. При этом Itera Data Platform позволяет автоматически во все сценарии преобразования добавить общую логику (различные проверки, очистку таблиц перед обновлением, и т. п.), а также использовать шаблоны преобразований. Решение включает пакет шаблонов, который может быть дополнен любым разработчиком, использующим IDP.

- IDP dbt – компонент Itera Data Platform, реализующий следующие функции:

  - компиляция сценариев обработки данных на основе макросов IDP;
  - Отображение структуры хранилища и этапов обработки данных.

  Компиляция сценариев посредством компонента IDP dbt позволяет автоматически задокументировать структуру хранилища, сценарии, модели и зависимости между ними, и отобразить все это в удобном для анализа веб-интерфейсе.





Модуль управления процессами обработки данных

Модуль включает следующие компоненты:

- Модуль генерации конвейеров данных IDP позволяет автоматически создавать конвейеры данных (DAG) для Airflow.

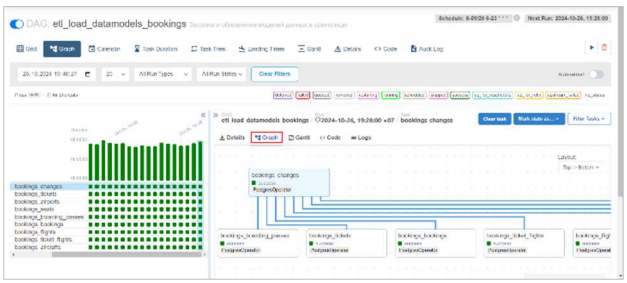
  На основании сведений об источниках, моделях данных и зависимостях между ними модуль генерации конвейеров данных IDP полностью автоматически строит последовательность задач для Airflow, рассчитывая при этом оптимальный план обработки данных с максимальным распараллеливанием задач при поддержке согласованности между ними. При изменении моделей и зависимостей между ними конвейеры автоматически обновляются, без участия разработчика, при этом выстраиваются новые оптимальные маршруты обработки данных.

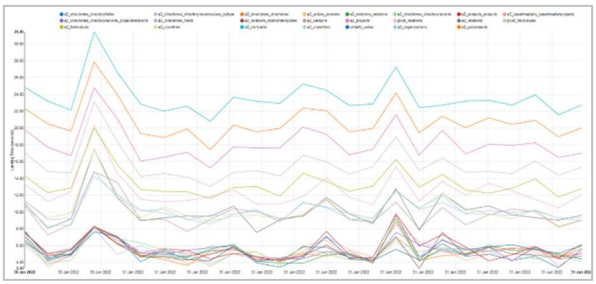
  Модуль генерации поддерживает возможность создания конвейеров следующих видов:

  - init_sources – инициализация источников данных;
  - load_datafiles – получение из локальных или внешних расположений исходных файлов данных и подготовку их к загрузке в хранилище данных;
  - load_datamodels – загрузка или обновление данных из различных источников в хранилище, а также преобразование данных внутри хранилища для подготовки конечных наборов данных для аналитических отчетов;
  - reset_sources – инициализация перезагрузки данных для всех моделей, относящихся к выбранному источнику;
  - reset_datafiles – инициализация перезагрузки файлов данных;
  - reset_datamodels – инициализации перезагрузки данных для отдельных моделей;
  - export_data – регулярная выгрузка данных из хранилища в различных форматах (xlsx, csv, json), и отправка в различные каналы (по электронной почте, в библиотеку корпоративного портала, на общий диск, на внешний ресурс);
  - другие служебные конвейеры.

- Модуль управления конвейерами данных IDP Airflow – компонент IDP, позволяющий управлять конвейерами данных, сгенерированными модулем генерации конвейеров данных IDP.

  Модуль позволяет в веб-интерфейсе осуществлять мониторинг выполнения задач, находить узкие места, быстро локализовать и устранять инциденты.





Таким образом, решение IDP позволяет путем автоматизации задач разработчика, администратора и аналитика хранилища данных максимально эффективно выполнять задачи по построению, поддержке и развитию ETL-системы.

Подробное описание модуля управления хранилищем и модуля управления процессами обработки данных, а также инструкции по их настройке и эксплуатации приведены в дальнейших разделах.

3 Модуль управления хранилищем данных

3.1 Конфигурирование хранилища данных

Конфигурирование хранилища данных осуществляется с помощью проекта dwh.

3.1.1 Описание проекта dwh

Проект dwh – проект хранилища данных, содержащий информацию об источниках данных, сценарии обработки данных, структуры промежуточных и результирующих моделей данных, а также параметры аудита данных.

Файловая структура и состав проекта dwh:

Каталог/файл	Описание
profiles.yml	Файл конфигурации профилей подключения к хранилищу данных
dwh	Корневой каталог проекта
dbt_project.yml	Файл конфигурации проекта
models	Корневой каталог размещения файлов моделей
schema.yml	Файл конфигурации моделей
audit	Каталог размещения файлов моделей схемы audit
base	Каталог размещения файлов моделей схемы base
core	Каталог размещения файлов моделей схемы core
export	Каталог размещения файлов моделей схемы export
target	Каталог размещения результатов компиляции проекта
manifest.json	Файл манифеста проекта

Модель dwh (модель) – единичная сущность хранилища данных (таблица, представление). Для каждой модели *проекта dwh* определяется скрипт, используемый для подготовки и загрузки данных в соответствующую таблицу хранилища, либо экспорта данных из хранилища. Файл скрипта модели размещается в каталоге *схемы dwh*, к которой относится данная модель.

При необходимости для модели могут быть указаны дополнительные параметры в *файле конфигурации моделей dwh*.

Файл модели – файл, содержащий скрипт обработки данных для соответствующей *модели dwh*.

Источник данных – набор параметров подключения, используемых для получения данных для одной или нескольких моделей. Могут быть определены источники данных следующих видов:

- база данных – подключение к базе данных на локальном или удаленном сервере баз данных;
- набор файлов данных.

Параметры каждого источника данных, а также набор относящихся к нему *таблиц-источников* определяются в *файле конфигурации моделей dwh*.

Таблица-источник – определение структуры отдельной таблицы *источника данных*, загружаемой в отдельную *модель dwh*. Физически каждой таблице-источнику соответствует пустая таблица в схеме sources базы данных хранилища. Для каждого *источника данных* могут быть определены одна или несколько таблиц-источников. Для источников данных вида «база данных» название таблиц-источников задаются в следующем формате: <название источника>_<название таблицы,

загружаемой из источника>, например, «bookings_flights», где «bookings» – название источника данных, а «flights» – название загружаемой таблицы.

Схема dwh – область хранилища данных, используемая для хранения моделей определенного типа или назначения. Система предусматривает следующий начальный набор схем хранилища данных:

- sources – схема для хранения *таблиц-источников* (структур отдельных таблиц *источников данных*);
- base – модели для загрузки и хранения сырых данных, полученных из внешних источников;
- core – модели для загрузки и хранения очищенных и обработанных данных, приведенных к удобной для применения в отчетах логической структуре;
- audit – модели для хранения результатов аудита загруженных данных;
- export – модели, используемые для задания выборок данных для экспорта.

Каждой схеме, за исключением схемы sources, соответствует отдельный каталог в каталоге models *проекта dwh*, содержащий *файлы моделей* данной схемы. В базе данных хранилища каждой схеме dwh соответствует отдельная схема базы данных.

Файл конфигурации проекта dwh (dbt_project.yml) – конфигурационный файл, в котором задаются следующие настраиваемые параметры проекта:

- name – название проекта.
- version – версия проекта.
- profile – профиль подключения к хранилищу данных.
- model-paths – путь размещения файлов сценариев моделей.
- macro-paths – путь размещения файлов макросов.
- target-path – путь размещения результатов компиляции проекта.
- datafiles_path – путь размещения файлов данных, загружаемых в хранилище.
- models – параметры обработки и отображения моделей, относящихся к разным схемам данных.

Файл конфигурации проекта dwh размещается в корневом каталоге проекта.

Файл конфигурации моделей dwh (schema.yml) – конфигурационный файл, содержащий параметры *источников данных* и *моделей* хранилища данных. Формат файла конфигурации моделей dwh описывается в п. 3.1.2. Файл конфигурации моделей dwh размещается в каталоге models *проекта dwh*.

Файл манифеста проекта dwh (manifest.json) – файл, содержащий полное описание всех ресурсов *проекта dwh*: параметры подключений и таблицы *источников данных*, параметры и скомпилированные скрипты обработки *моделей dwh*. Файл манифеста создается автоматически модулем dbt на основе и *файлов моделей dwh* и *файла конфигурации моделей dwh*. Файл манифеста размещается в каталоге target *проекта dwh*.

3.1.2 Файл конфигурации моделей dwh

Файл конфигурации моделей dwh служит для определения источников данных и моделей хранилища данных и содержит разделы:

- sources,
- models.

3.1.2.1 Раздел sources

Данный раздел содержит массив конфигураций источников данных. Конфигурация источника данных включает следующие настраиваемые параметры:

name – название источника данных.

database – база данных, в которой хранятся структуры таблиц источника данных. По умолчанию – база данных хранилища.

schema – схема базы данных, в которой хранятся структуры таблиц источника данных. По умолчанию – схема sources базы данных хранилища.

meta – набор дополнительных параметров подключения к локальной или удаленной базе данных для импорта в хранилище исходных данных и/или структуры данных. Для источника, представляющего базу данных Microsoft SQL или PostgreSQL, как структура данных (состав и типы полей таблиц загружаемых данных), так и сами данные импортируются из указанной базы данных. Для файлового источника указанные параметры позволяют подключиться к базе данных, в которой хранятся только структуры таблиц загружаемых данных, при этом сами данные загружаются из файлов.

Параметры подключения к источнику данных и/или источнику структуры данных:

- **source_type** – тип источника: «mssql» или «postgres».
- **source_host** – сервер базы данных (IP-адрес, либо полное доменное имя).
- **source_port** – порт подключения к серверу базы данных.
- **source_database** – название базы данных.
- **source_schema** – схема базы данных.
- **fdw_server** – название внешнего сервера, создаваемого для подключения к базе данных.
- **source_user** – имя пользователя базы данных.
- **source_password** – пароль пользователя базы данных.
- **mapping_user** – имя пользователя базы данных хранилища, сопоставляемого с пользователем базы данных источника.

tables – массив таблиц-источников, определяемых для источника данных. Для каждой таблицы-источника указывается значение параметров:

- **name** – название таблицы-источника. Значение определяет название пустой физической таблицы, создаваемой в схеме sources базы данных хранилища и определяющей структуру данных, загружаемых в модель. Данная таблица создается автоматически, если в подразделе meta указаны параметры подключения к источнику структуры данных. Если, в случае файлового источника, параметры подключения к источнику структуры данных отсутствуют, таблица-источник должна быть создана в схеме sources вручную.

3.1.2.2 Раздел models

Данный раздел содержит массив конфигураций моделей dwh. Конфигурация модели dwh включает следующие настраиваемые параметры:

name – название модели.

config:meta – набор дополнительных свойств, задаваемых для модели dwh.

Свойства модели, связанной с файловым источником данных («datafiles»):

- **source** – название источника, определенного в разделе sources. Указывается значение «datafiles».
- **source_table** – название таблицы-источника для модели dwh (название таблицы в схеме sources базы данных хранилища, которая определяет структуру данных, загружаемых из файлового источника в данную модель). Указанное значение должно содержаться в массиве tables раздела sources (п. 3.1.2.1).
- **storage_type** – тип ресурса, на котором размещен файл-источник загружаемых данных. Возможные значения: «web», «sharepoint», «yandex_disk». Необязательный параметр, значения по умолчанию: если значение параметра source_file начинается с «http» или «https», значение storage_type принимается равным «web», иначе тип ресурса рассматривается как файловый каталог.
- **storage_auth** – тип аутентификации. Возможные значения: «ntlm», «oauth». Необязательный параметр.
- **storage_conn** – идентификатор параметров подключения к удаленному ресурсу, сохраненных на сервере Airflow.
- **source_file** – путь размещения файла-источника. Значение параметра может представлять:
    1) полный путь к файлу, размещенному в каталоге контейнера ETL,
    2) относительный путь к файлу, размещенному на Яндекс Диске (для типа ресурса «yandex_disk» и типа аутентификации «oauth»),
    3) публичная ссылка на файл, размещенный на Яндекс Диске (для типа ресурса «yandex_disk» без указания типа аутентификации),
    4) URL-адрес файла, размещенного на портале MS SharePoint (для типа ресурса «sharepoint»),
    5) URL-адрес файла, размещенного на другом веб-ресурсе (для типа ресурса «web»).
- **source_format** – формат файла-источника. Возможные значения: «xlsx», «xls», «csv», «txt». Необязательный параметр, значение по умолчанию – «csv».
- **source_sheet** – название листа Excel в файле-источнике, на котором размещаются загружаемые данные. Обязательный параметр для файлов формата xls/xlsx.
- **delimiter** – разделитель, используемый при загрузке из файла формата csv/txt. Необязательный параметр, значение по умолчанию – «;».
- **rows_start** – начальная строка загружаемых данных, нумерация с 1. Необязательный параметр, значение по умолчанию – 1.
- **rows_count** – количество загружаемых строк. При значении 0 – без ограничений. Необязательный параметр, значение по умолчанию – 0.
- **columns_start** – начальный столбец загружаемых данных, нумерация с 1. Необязательный параметр, значение по умолчанию – 1.
- **columns_count** – количество загружаемых столбцов данных. Обязательный параметр.
- **columns_count_abs** – полное количество столбцов при загрузке из файла формата csv/txt. Необязательный параметр, если не указан, рассчитывается как columns_start + columns_count – 1.
- **columns_count_check** – указывает, следует ли выполнять проверку файла csv/txt и пропускать строки с количеством полей меньше columns_count_abs. Необязательный параметр.

Общие свойства модели (для любых типов источника данных):

- **filter** – фильтр на основе значений полей таблицы. Синтаксис SQL. Необязательный параметр.

3.1.3 Операции для построения хранилища данных

3.1.3.1 Добавление источника данных

3.1.3.1.1 Добавление реляционного источника данных

Для добавления реляционного источника данных (база данных PostgreSQL или MS SQL Server) необходимо выполнить следующие действия:

1. Добавить конфигурацию источника в раздел sources файла конфигурации моделей dwh (schema.yml): указать параметры подключения к источнику и список таблиц-источников в соответствии с п. 3.1.2.1.
2. Для источника типа «postgres» в каталоге «base» проекта dwh создать файл модели для загрузки изменений данных.

   Название файла: «<название_источника>_changes.sql».

   Содержимое файла:

   ```
   {{ generate_import_pg_changes('<название_источника>') }}
   ```

3. Создать файлы моделей для каждой из загружаемых таблиц-источников.

   Название файла: «<название_источника>_<название_таблицы_источника>.sql».

   Содержимое файла:

   - для источника типа «postgres»:

     ```
     {{ generate_import_pg(<название_источника>, <название_таблицы_источника>, [<список_полей_первичного_ключа>]) }}
     ```

   - для источника типа «mssql»:

     ```
     {{ generate_import_mssql(<название_источника>, <название_таблицы_источника>, [<список_полей_первичного_ключа>]) }}
     ```

4. Скомпилировать проект dwh в соответствии с п. 3.1.3.3.
5. Запустить задачу инициализации нового или измененного источника данных в соответствии с п. 4.2.2.2.2.
6. Добавить, при необходимости, в файл проекта ETL (etl_project.yml) конфигурацию конвейера загрузки и обновления данных для новых моделей dwh.

3.1.3.1.2 Добавление файлового источника данных

В качестве файлового источника может выступать файл формата «xlsx», «xls», «csv» или «txt», размещенный на ресурсе одного из следующих типов:

- файловый каталог,
- Яндекс Диск (с авторизацией OAuth, либо публичная ссылка),
- портал SharePoint,
- другой веб-ресурс.

Для добавления файлового источника данных необходимо выполнить следующие действия:

1. Разместить файл-источник на выбранном ресурсе и обеспечить к нему доступ для служб сервера ETL.

   1.1. В качестве файлового каталога может быть использован:

   - локальный каталог, размещенный в /opt/etl/airflow/data сервера ETL;
   - символическая ссылка на локальный каталог, размещенная в /opt/etl/airflow/data сервера ETL;
   - удаленный каталог, смонтированный в подкаталог /opt/etl/airflow/data сервера ETL.

   1.2. Если используется Яндекс Диск с авторизацией OAuth, необходимо получить OAuth-токен для доступа к данным в соответствии с инструкцией https://yandex.ru/dev/disk-api/doc/ru/concepts/quickstart и выполнить скрипт инициализации подключения на сервере ETL:

```
sudo docker exec -it airflow-webserver bash -c 'airflow connections add <ид_подключения> \
    --conn-type generic \
    --conn-password <oauth_токен>'
```

   1.3. Если используется портал SharePoint NTLM авторизацией, необходимо выполнить скрипт инициализации подключения на сервере ETL:

```
sudo docker exec -it airflow-webserver bash -c 'airflow connections add <ид_подключения> \
    --conn-type http \
    --conn-host <ip_или_fdqn_сервера_sharepoint> \
    --conn-schema https \
    --conn-login <доменное_имя_пользователя> \
    --conn-password <пароль_пользователя>
```

2. Определить структуру таблицы-источника: создать пустую таблицу в схеме sources хранилища, соответствующую загружаемым данным по составу и типам полей.
3. Добавить определенную в п. 2 таблицу-источник в список таблиц источника datafiles в файле конфигурации моделей dwh (schema.yml).
4. Создать файл модели dwh для загрузки сырых данных из файла-источника.

   Название файла: «datafiles_<название_таблицы>.sql».

   Содержимое файла:

```
{{ generate_import_csv() }}
```

5. Скомпилировать проект dwh в соответствии с п. 3.1.3.3.
6. Добавить конфигурацию модели в раздел models файла конфигурации моделей dwh (schema.yml): указать параметры загрузки данных в соответствии с п. 3.1.2.2.
7. Скомпилировать проект dwh в соответствии с п. 3.1.3.3.
8. Добавить, при необходимости, в файл проекта ETL (etl_project.yml) конфигурацию конвейера загрузки и обновления данных для новых моделей dwh.
9. Добавить, при необходимости, в файл проекта ETL (etl_project.yml) конфигурацию конвейера загрузки и обновления файлов данных (load_datafiles).

3.1.3.2 Добавление модели dwh

3.1.3.2.1 Общий порядок добавления моделей dwh

1. Создание файла модели в соответствующем каталоге проекта dwh (см. описание каталогов проекта dwh в п. 3.1.1). Например, файлы моделей для загрузки сырых данных, размещаются в каталоге models/base, файлы для загрузки очищенных и преобразованных данных размещаются в каталоге models/core.
2. Компиляция проекта dwh (см. п. 3.1.3.3).

3.1.3.2.2 Файл модели для загрузки и обновления сырых данных

Содержимое файла модели для загрузки и обновления сырых данных зависит от типа источника данных.

- Файл модели для источника типа «postgres»:

```
{{ generate_import_pg(<название_источника>, <название_таблицы_источника>,
[<список_полей_первичного_ключа>]) }}
```

- Файл модели для источника типа «mssql»:

```
{{ generate_import_mssql(<название_источника>, <название_таблицы_источника>,
[<список_полей_первичного_ключа>]) }}
```

- Файл модели для файлового источника:

```
{{ generate_import_csv() }}
```

3.1.3.2.3 Файл модели для преобразования данных

Файл модели dwh содержит SQL-скрипт получения и преобразования данных. В исходном скрипте модели могут использоваться Python-подобные выражения и операторы, которые в результате компиляции преобразуются в код SQL. Для обращения к данным других моделей используются ссылочные выражения вида «{{ ref('название_модели_источника') }}», что позволяет модулю ETL выстраивать оптимальный граф загрузки моделей в хранилище данных. Пример исходного скрипта модели dwh приведен на Рис. 1. Исходный и скомпилированный скрипты любой модели dwh можно посмотреть в панели мониторинга структуры хранилища данных (см. п. 3.2).

Рис. 1. Пример исходного скрипта модели dwh

```
{{ config(tags='dvdrental') }}
{{ generate_update_begin() }}

        ------------------------------------------------------------------------------

        select
            c.name as category,
            sum(p.amount) as total_sales

        from {{ ref('datafiles_dvdrental_payment') }} as p

        join {{ ref('datafiles_dvdrental_rental') }} as r
            on p.rental_id = r.rental_id

        join {{ ref('datafiles_dvdrental_inventory') }} as i
            on r.inventory_id = i.inventory_id

        join {{ ref('datafiles_dvdrental_film') }} as f
            on i.film_id = f.film_id

        join {{ ref('datafiles_dvdrental_film_category') }} as fc
            on f.film_id = fc.film_id

        join {{ ref('datafiles_dvdrental_category') }} as c
            on fc.category_id = c.category_id

        group by c.name

        order by (sum(p.amount)) desc;

        ------------------------------------------------------------------------------

{{- generate_update_end() }}
```

3.1.3.3 Компиляция проекта dwh

После внесения изменений в конфигурацию хранилища данных, таких как добавление или изменение источников данных либо моделей dwh, необходимо выполнить компиляцию проекта dwh.

Компиляция проекта производится посредством выполнения на сервере ETL следующей команды:

```
sudo docker exec -it dbt bash -c 'cd /usr/app/dbt/dbt_projects/dwh/ && dbt docs generate'
```

Результат выполнения компиляции можно проверить в панели мониторинга структуры хранилища данных (см. п. 3.2).

3.1.3.4 Настройка отслеживания измененных данных в источнике

Для обеспечения возможности инкрементального обновления данных из удаленной базы данных PostgreSQL или MS SQL Server необходимо настроить на стороне источника отслеживание измененных данных (CDC, Change Data Capture).

3.1.3.4.1 Настройка CDC для PostgreSQL

На сервере базы данных источника необходимо выполнить следующие действия:

1. Установить уровень поддержки логического декодирования для журнала (WAL).

   SQL-запрос (после выполнения требуется перезапуск службы PostgreSQL):

   ```
   alter system set wal_level = logical;
   ```

2. Установить расширение wal2json:

```
git clone https://github.com/eulerto/wal2json.git
cd wal2json
export PATH=/usr/lib/postgresql/15/bin:$PATH
USE_PGXS=1 make
USE_PGXS=1 make install
```

3.1.3.4.2 Настройка CDC для MS SQL

На сервере базы данных источника необходимо выполнить следующие действия:

1. Включить отслеживание изменений для базы данных источника:

```
alter database <название_базы_данных> set change_tracking = on (change_retention = 10 days, auto_cleanup = on);
```

2. Включить отслеживание изменений для каждой из загружаемых таблиц:

```
alter table <название_таблицы> enable change_tracking with (track_columns_updated = on);
```

3. Предоставить пользователю разрешения на отслеживание изменений на уровне схемы:

```
grant view change tracking on schema::[<название_схемы>] to [<имя_пользователя>];
```

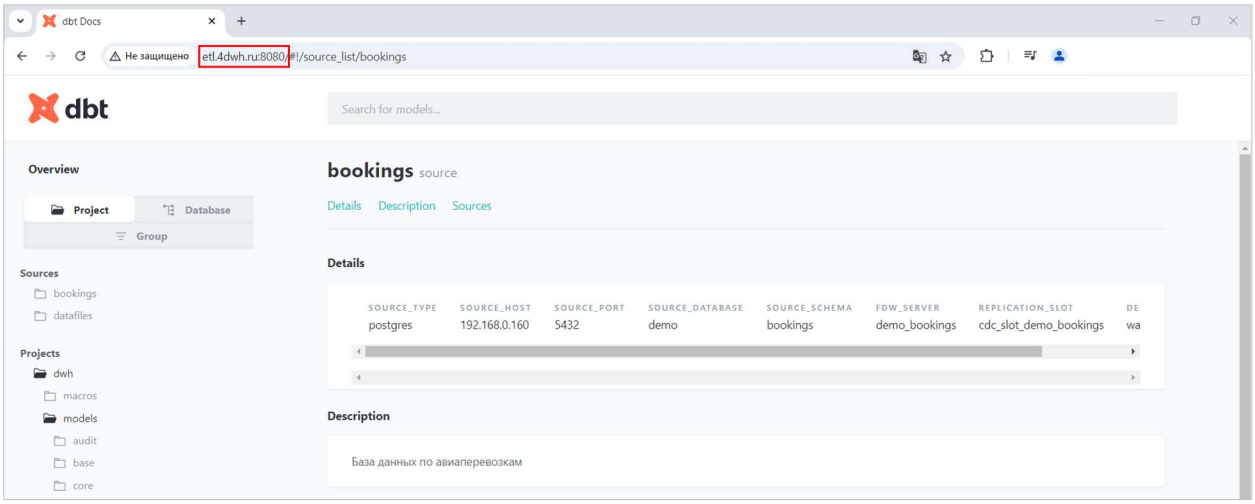
4. Предоставить пользователю разрешения на просмотр плана исполнения на уровне базы данных:

```
grant showplan to [<имя_пользователя>];
```

3.2 Панель мониторинга структуры хранилища данных

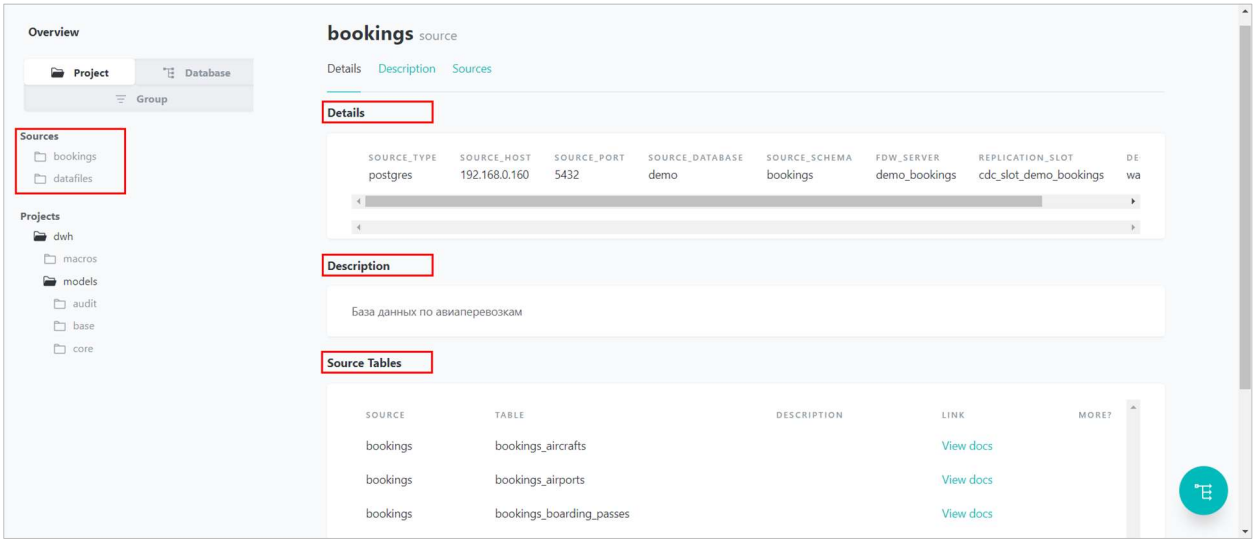
. Панель мониторинга предоставляет детальную структурированную информацию обо всех источниках данных и моделях dwh, а также зависимостях между.

Рис. 2. Панель управления структурой хранилища данных



Информация об источниках данных хранилища доступна в разделе «Sources». При выборе нужного источника на вкладках «Details», «Description» и «Source Tables» отображаются параметры подключения к источнику, описание и список таблиц источника данных (см. Рис. 3).

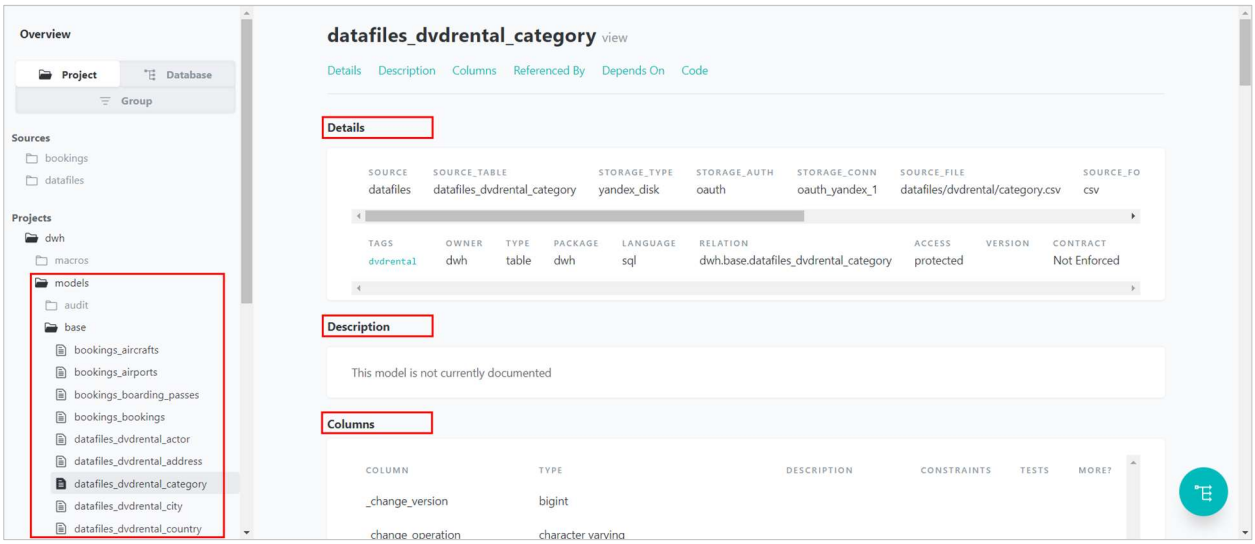
Рис. 3. Раздел «Источники данных»



Информация о моделях dwh доступна в разделе «models». Модель dwh – единичная сущность хранилища данных (таблица, представление). Для каждой модели определяется сценарий, используемый для подготовки и загрузки данных в соответствующую таблицу хранилища, либо экспорта данных из хранилища.

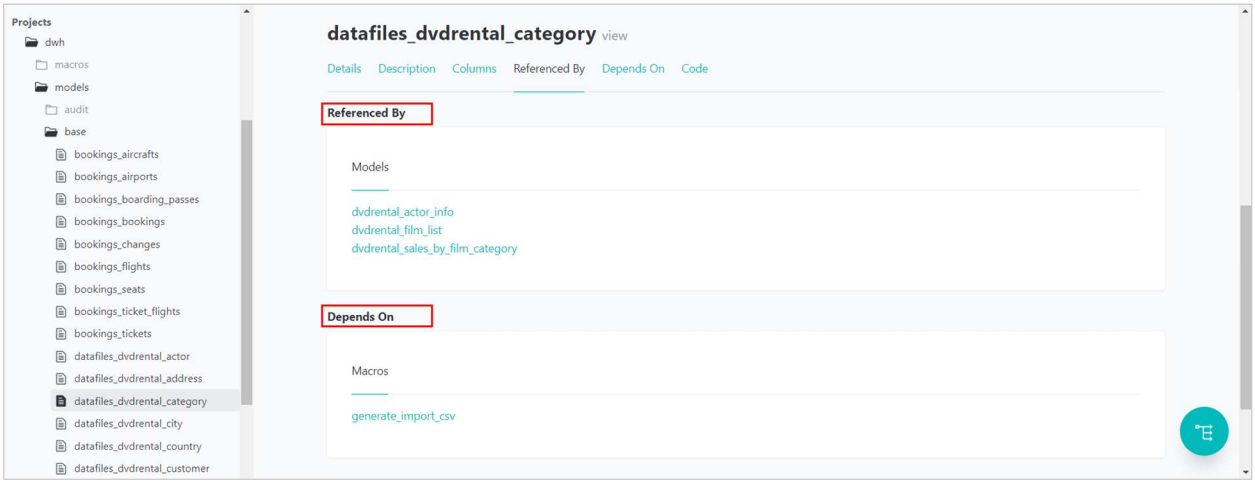
При выборе нужной модели dwh на вкладках «Details», «Description» и «Columns» отображаются свойства модели, описание, а также перечень и типы полей (см. Рис. 4).

Рис. 4. Свойства модели dwh



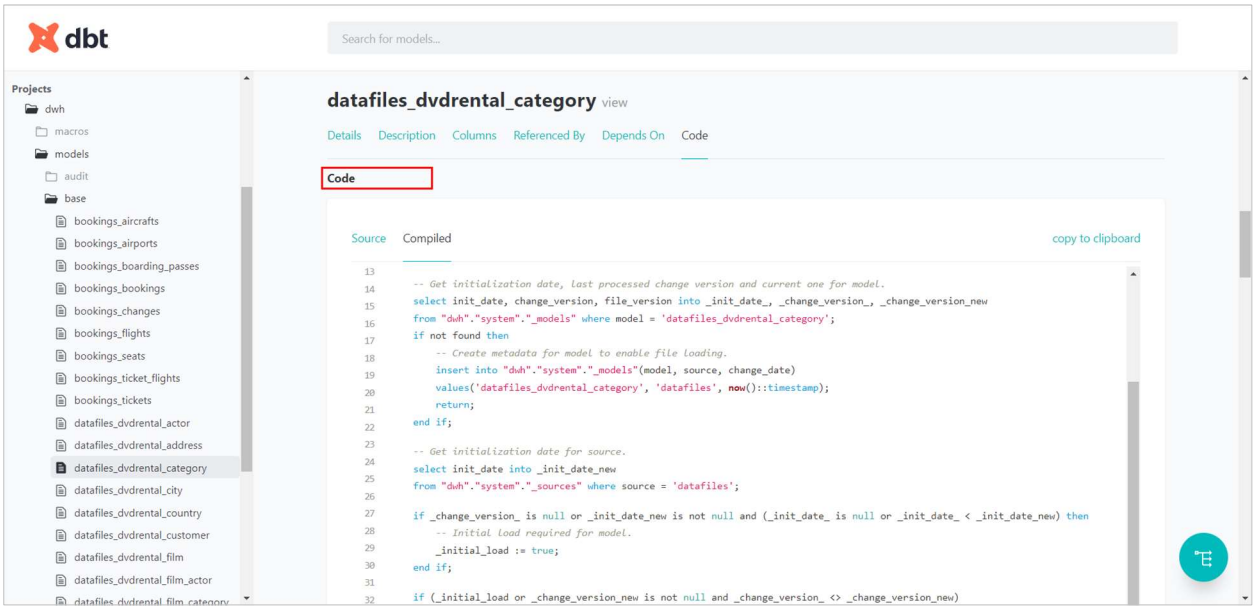
На вкладках «Referenced By» и «Depends On» отображаются зависимости других моделей от данной модели, а также модели и макросы, от которых зависит данная модель (см. Рис. 5).

Рис. 5. Зависимости модели dwh



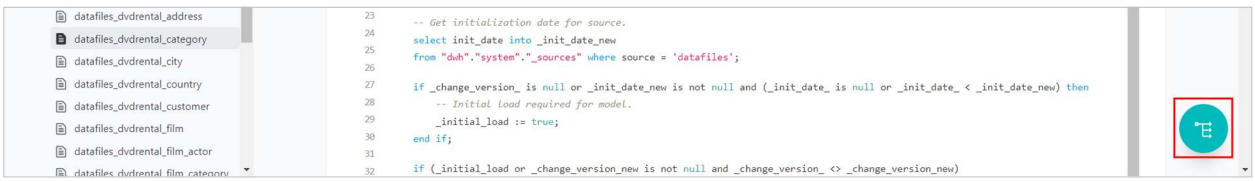
На вкладке «Code» доступен к просмотру и копированию генерируемый скрипт обработки выбранной модели (см. Рис. 6).

Рис. 6. Скрипт обработки модели dwh



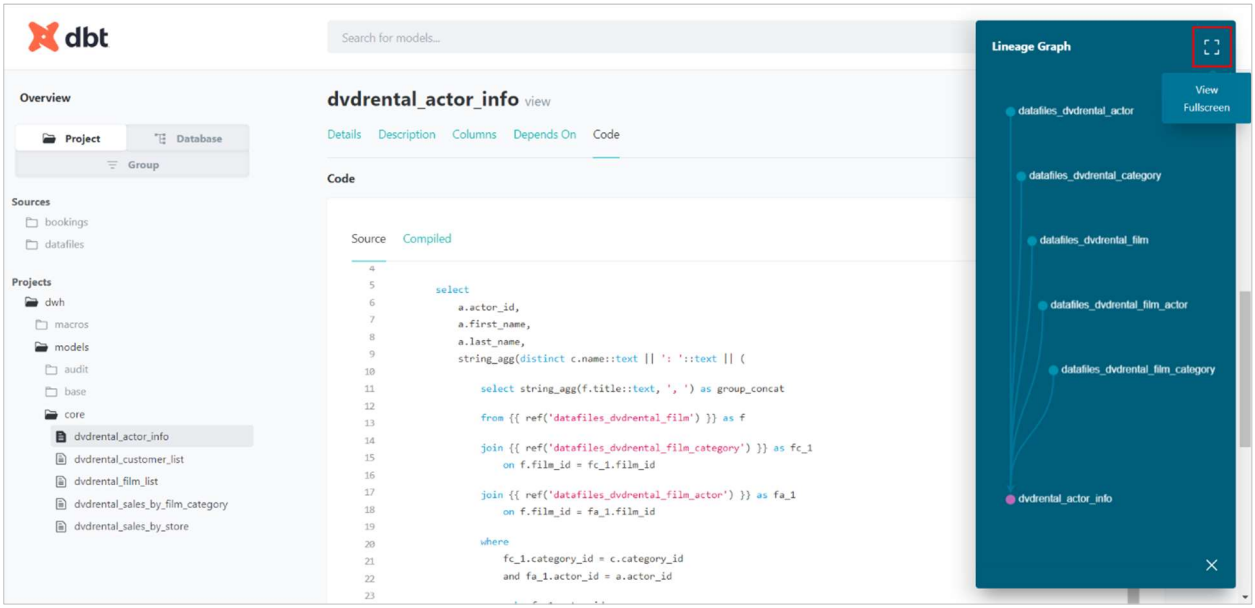
Панель мониторинга также предоставляет графическое отображение связей моделей dwh друг с другом и с источниками данных. Для просмотра необходимо кликнуть на значок схемы в правом нижнем углу экрана (см. Рис. 7).

Рис. 7. Переход к просмотру графа зависимостей



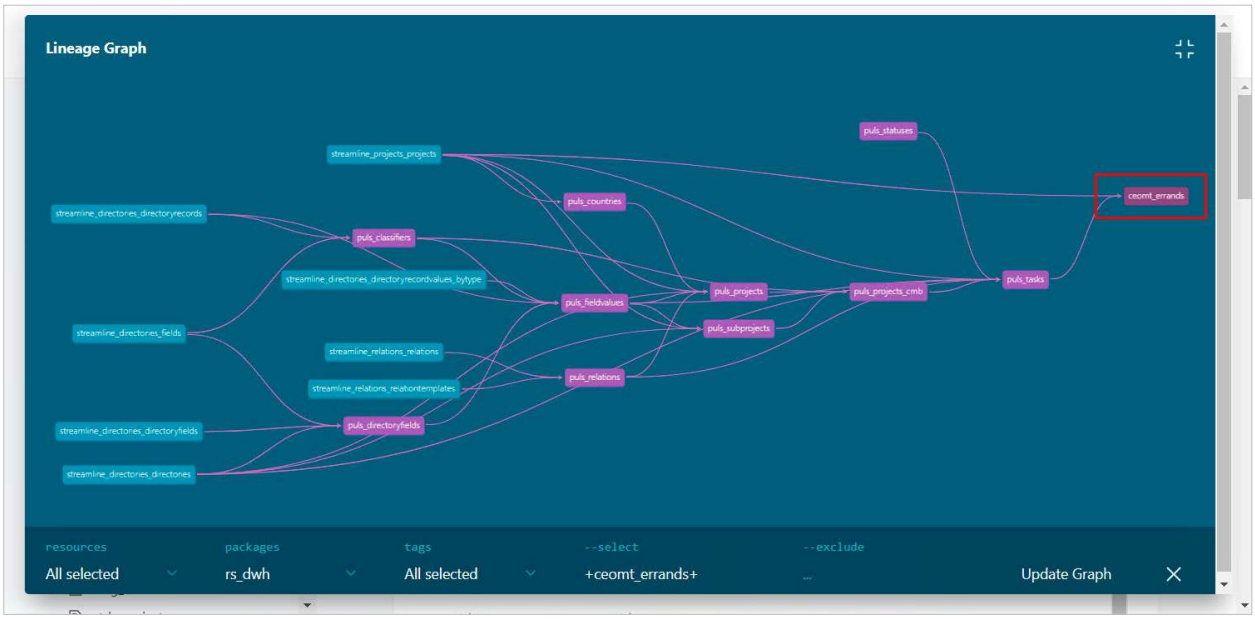
Для перехода к полноэкранному отображению необходимо кликнуть по иконке в правом верхнем углу графа зависимостей (см. Рис. 8).

Рис. 8. Переход к полноэкранному отображению графа зависимостей



Здесь можно выбрать нужную модель и посмотреть её предшественников и последователей (см. Рис. 9).

Рис. 9. Полноэкранное отображение графа зависимостей



4 Модуль управления процессами обработки данных

4.1 Конфигурирование конвейеров данных

Конфигурирование конвейеров данных осуществляется с помощью проекта ETL.

4.1.1 Описание проекта ETL

DAG – конвейер данных – серия задач, выполняемых в определенном порядке, последовательно или параллельно, для реализации процесса загрузки/обработки данных.

Проект ETL – серия конвейеров данных (DAG), реализующих процессы подготовки и загрузки данных из различных источников (файловых, реляционных) в хранилище данных, экспорта данных из хранилища, а также другие служебные задачи.

Файл проекта ETL (etl_project.yml) – конфигурационный файл, содержащий значения общих параметров загрузки, обработки и экспорта данных, а также конфигурации отдельных конвейеров данных. Формат файла проекта ETL описывается в п. 4.1.2.

4.1.2 Файл проекта ETL

Файл проекта ETL служит для задания системных параметров и настроек конвейеров данных.

4.1.2.1 Раздел general_settings

В данном разделе устанавливаются общие свойства проекта ETL.

dbt_project – название проекта хранилища данных (проекта dwh).

dbt_schema_path – путь к файлу конфигурации моделей dwh.

dbt_manifest_path – путь к файлу манифеста проекта dwh.

datafiles_path – путь сохранения файлов данных, подготовленных к загрузке в хранилище данных.

upload_path – путь размещения исходных файлов данных.

export_path – путь сохранения файлов данных при экспорте из хранилища.

dwh_conn_id – идентификатор параметров подключения к базе данных хранилища, сохраненных на сервере Airflow.

dwh_conn_id – идентификатор параметров SSH-подключения к серверу ETL, сохраненных на сервере Airflow.

sp_conn_id – идентификатор параметров подключения к порталу SharePoint для сохранения файлов выгрузки данных, сохраненных на сервере Airflow.

4.1.2.2 Раздел general_dag_args

В данном разделе устанавливаются общие параметры конвейеров данных.

owner – пользователь, владелец запускаемых конвейеров данных.

timezone – часовой пояс, относительно которого настраиваются расписания запуска конвейеров.

email – один или несколько адресов электронной почты для отправки сообщений об ошибках.

tags – теги, присваиваемые конвейерам данных для возможности фильтрации списка конвейеров в пользовательском интерфейсе.

4.1.2.3 Раздел dags

Данный раздел содержит массив конфигураций отдельных конвейеров данных. Конфигурация конвейера включает следующие настраиваемые параметры:

dag_id – идентификатор конвейера.

description – описание конвейера, отображаемое в пользовательском интерфейсе.

schedule – интервал запуска конвейера в минутах (если числовое значение), либо строка расписания запуска в cron-формате.

dagrun_timeout – максимально допустимое время выполнения конвейера в минутах, в случае если конвейер не успел завершиться до времени следующего запуска.

retries – количество повторных попыток выполнения каждой задачи конвейера в случае ошибки.

retry_delay – интервал времени между повторными попытками выполнения задачи в секундах.

email_on_retry – указывает, следует ли отправлять оповещения по электронной почте при повторной попытке выполнения задачи.

email_on_failure – указывает, следует ли отправлять оповещения по электронной почте в случае сбоя задачи (если задача не выполнилась после всех повторных попыток).

dag_settings – набор значений дополнительных параметров конвейера данных, состав которых определяется типом конвейера. Тип конвейера определяется параметром **dag_type**. Проект ETL может включать конвейеры следующих типов:

- **init_sources** – конвейер, содержащий задачи инициализации источников данных.
- **load_datafiles** – конвейер, обеспечивающий получение из локальных или внешних расположений исходных файлов данных и подготовку их к загрузке в хранилище данных (см. параметры п. 4.1.2.3.1).
- **load_datamodels** – конвейер, обеспечивающий загрузку или обновление данных из различных источников в хранилище, а также преобразование данных внутри хранилища для подготовки конечных наборов данных для аналитических отчетов (см. параметры п. 4.1.2.3.2).

  В целях оптимизации могут быть созданы несколько конвейеров загрузки данных с различными расписаниями запуска. Например, для моделей, связанных с обработкой и хранением больших объемов редко обновляемых данных (архивные данные), может быть создан отдельный конвейер с низкой частотой запуска.
- **reset_sources** – конвейер, содержащий задачи инициализации перезагрузки данных для всех моделей, относящихся к выбранному источнику.
- **reset_datafiles** – конвейер, содержащий задачи инициализации перезагрузки файлов данных.
- **reset_datamodels** – конвейер, содержащий задачи инициализации перезагрузки данных для отдельных моделей.
- **export_data** – конвейер, обеспечивающий регулярную выгрузку данных из хранилища в формате csv либо xlsx с сохранением файлов выгрузки на корпоративном портале, в локальном или сетевом каталоге либо отправкой по электронной почте (см. параметры п. 4.1.2.3.3).
  Может быть определено множество конвейеров экспорта данных с разными расписаниями, каждый из которых содержит одну или несколько задач выгрузки требуемых данных с отправкой в соответствующий канал экспорта.

- **clean_logs** – конвейер, обеспечивающий регулярную очистку журналов событий на сервере ETL (см. параметры п. 4.1.2.3.4).

4.1.2.3.1 Параметры конвейеров типа load_datafiles

model_tags – перечень тегов, при указании которого в конвейер будут включены только модели с перечисленными тегами (например, «log» для обработки файлов журналов). Если в обработку необходимо включить модели, у которых не задано ни одного тега, параметр model_tags не указывается, либо должен содержать значение «none».

4.1.2.3.2 Параметры конвейеров типа load_datamodels

model_schemas – перечень схем хранилища данных, модели которых должны быть включены в конвейер (например, «base», «core»).

model_tags – перечень тегов, при указании которого в конвейер будут включены только модели с перечисленными тегами (например, «highload» для моделей, связанных с обработкой больших объемов данных). Если в обработку необходимо включить модели, у которых не задано ни одного тега, параметр model_tags не указывается, либо должен содержать значение «none».

4.1.2.3.3 Параметры конвейеров типа export_data

tasks – массив конфигураций задач экспорта данных.

Конфигурация задачи экспорта включает следующие настраиваемые параметры:

task_id – уникальный идентификатор задачи.

source_model – название модели dbt, в которой определяется выборка данных. Для экспорта данных могут быть использованы только модели, относящиеся к схеме «export».

export_file – название файла выгрузки данных. Поддерживаются только файлы формата csv, tsv и xlsx.

upload_path – локальный или сетевой каталог, в котором должен быть сохранен файл выгрузки.

sharepoint_site – адрес узла SharePoint, на котором должен быть сохранен файл выгрузки.

sharepoint_folder – относительный путь каталога SharePoint, в котором должен быть сохранен файл выгрузки.

mail_list_to – перечень адресов электронной почты, на которые должен быть отправлен файл выгрузки.

mail_list_cc – перечень адресов электронной почты, указываемых в поле «Копия» при отправке файла выгрузки.

mail_subject – Текст, размещаемый в теме письма при отправке файла выгрузки.

4.1.2.3.4 Параметры конвейеров типа clean_logs

log_paths – список каталогов размещения журналов событий на сервере ETL.

containers – список контейнеров на сервере ETL, для которых должна выполняться очистка журналов.

4.2 Панель управления конвейерами данных

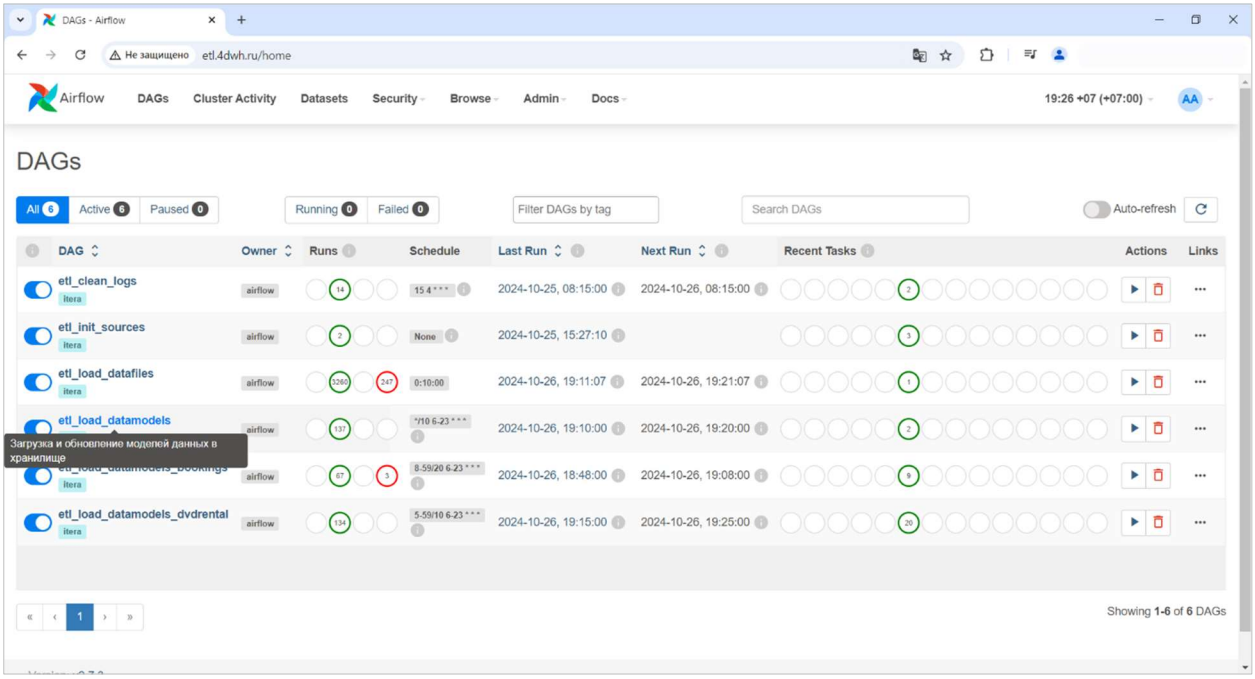
Панель управления конвейерами данных предоставляет возможности мониторинга выполнения конвейеров обработки данных, а также выполнения ряда ручных операций, связанных с инициализацией источников данных либо перезагрузкой данных в хранилище.

Создание и настройка конвейеров данных производится через конфигурационный файл проекта ETL. Спецификация файла проекта ETL, а также перечень и описание доступных типов конвейеров приведены в п. 4.1.2.

4.2.1 Мониторинг выполнения конвейеров данных

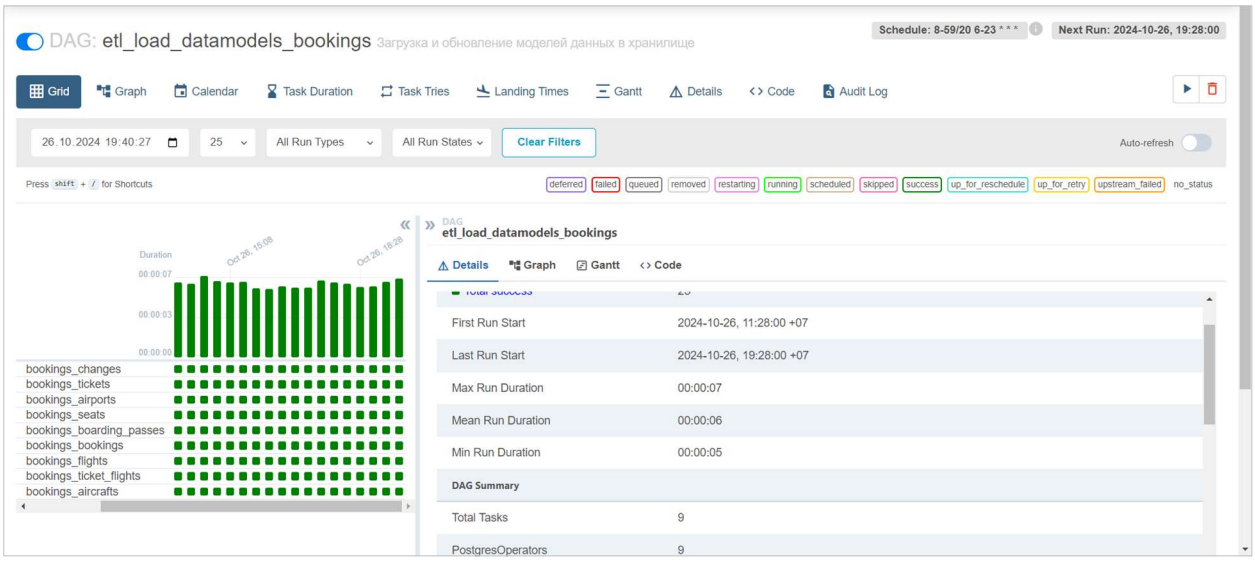
Панель управления конвейерами данных доступна по адресу http://idp.4dwh.ru:1447 и предоставляет доступ к списку конвейеров данных. При наведении мышки на названии конвейера всплывает подсказка с описанием назначения данного конвейера (см. Рис. 10).

Рис. 10. Панель управления конвейерами данных



При клике на названии конвейера данных выполняется переход к представлению задач конвейера (см. Рис. 11).

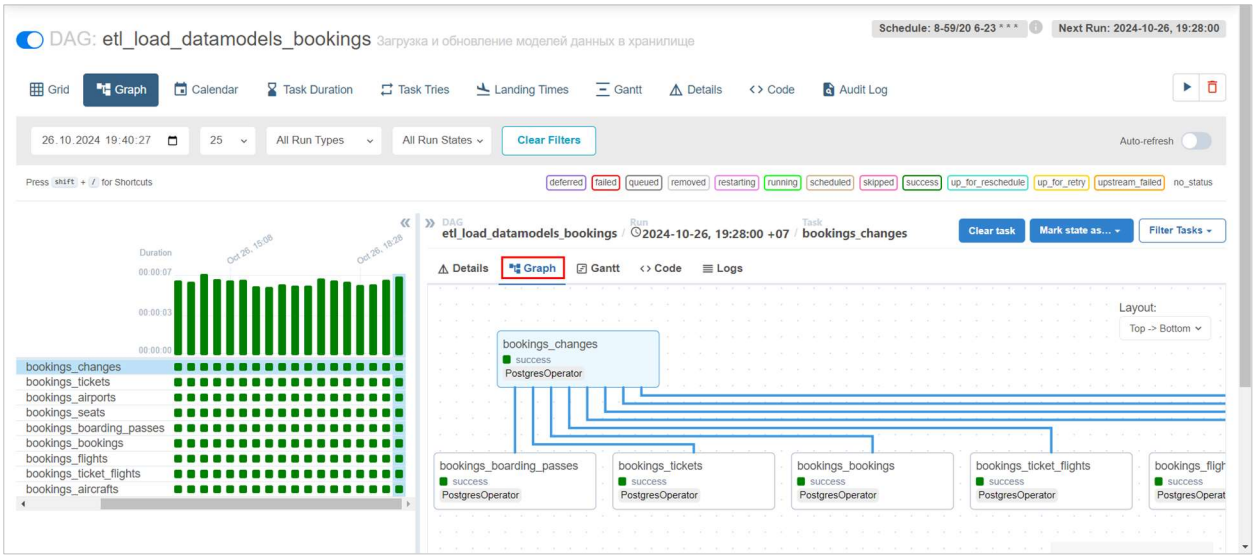
Рис. 11. Представление задач конвейера данных



Для списка задач конвейера данных доступны представления:

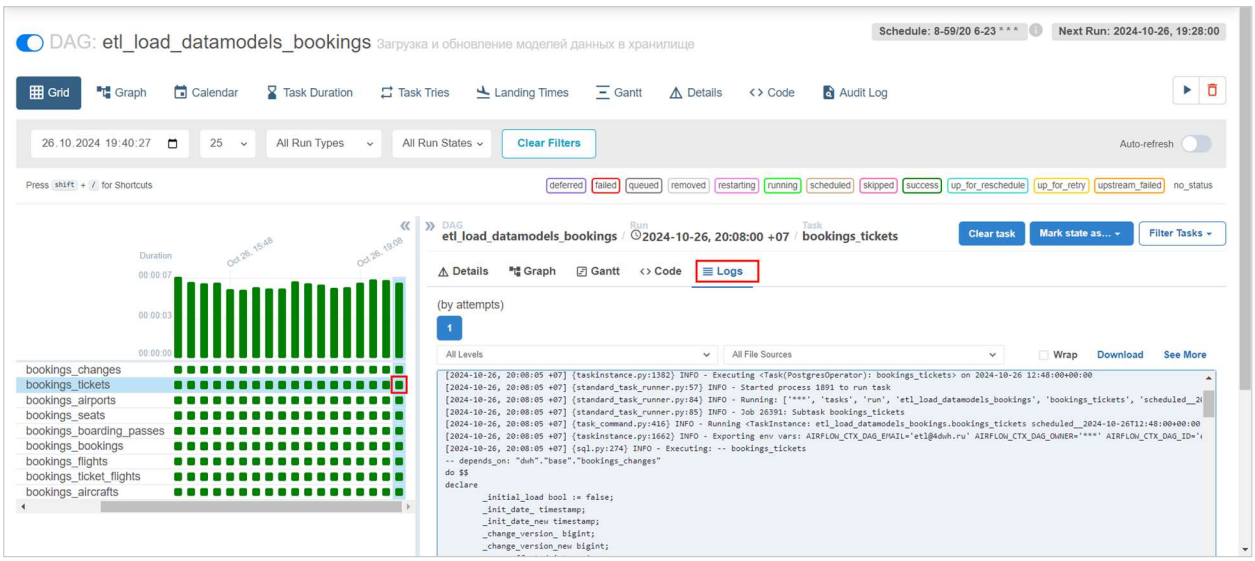
- в виде графа,
- в виде диаграммы Гантта,
- лог выполнения для каждой задачи конвейера.

Рис. 12. Представление задач конвейера данных в виде графа



Для просмотра лога выполнения задачи необходимо выбрать на графике нужный запуск задачи и перейти на вкладку «Logs» (см. Рис. 13).

Рис. 13. Лог выполнения задачи конвейера данных



4.2.2 Управление конвейерами данных

Запуск конвейеров осуществляется автоматически, в соответствии с настроенным для каждого конвейера расписанием.

Панель управления конвейерами данных также позволяет вручную осуществлять следующие операции:

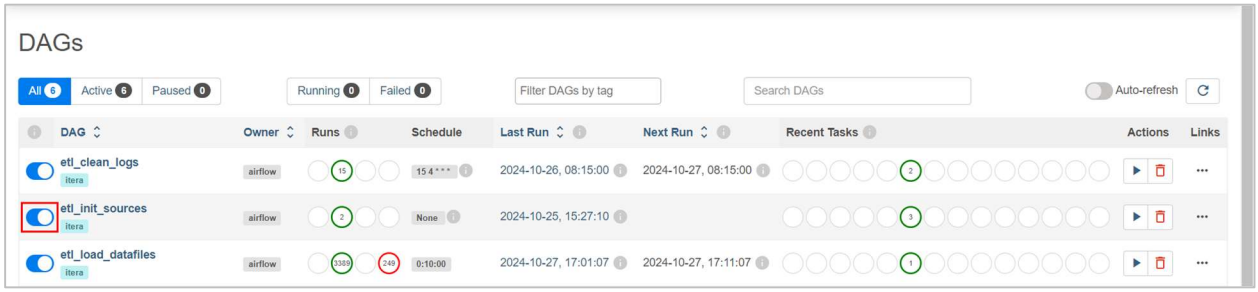
- Запуск/остановка отдельного конвейера данных.

- Инициализация источников данных.
- Запуск обновления файлов данных.
- Запуск загрузки и обработки данных в хранилище.
- Запуск перезагрузки файлов данных.
- Запуск перезагрузки данных в хранилище.

4.2.2.1 Запуск/остановка конвейера данных

Запуск/остановка отдельного конвейера данных выполняется в представлении списка конвейеров с использованием переключателя «Запуск/остановка» для соответствующего элемента списка (см. Рис. 14).

Рис. 14. Запуск/остановка конвейера данных



4.2.2.2 Инициализация источников данных

Для инициализации (или переинициализации) источника данных dwh необходимо:

- сконфигурировать источник данных в проекте dwh,
- выполнить ручной запуск задачи инициализации нужного источника данных.

4.2.2.2.1 Конфигурирование источника данных в проекте dwh

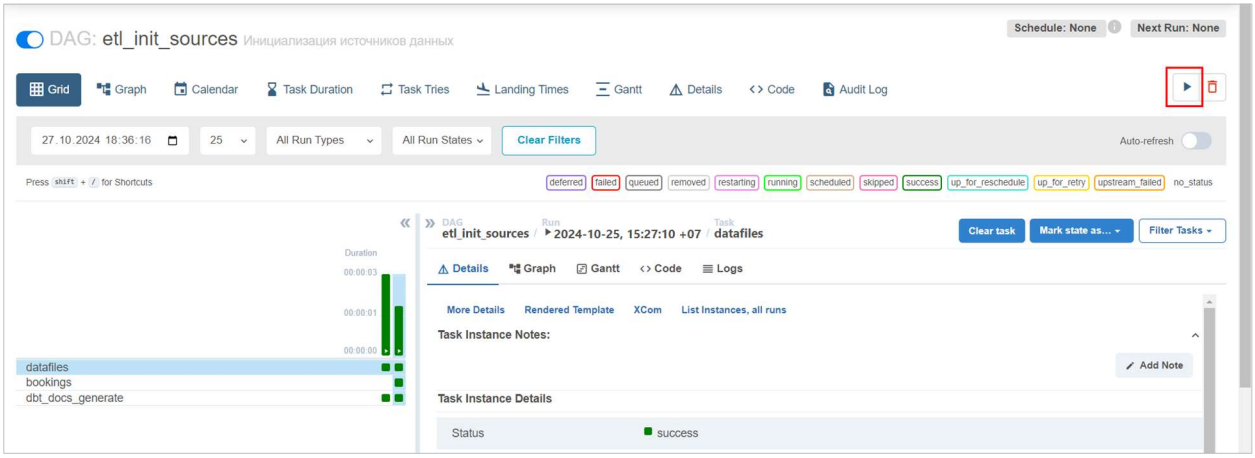
Конфигурирование источника данных в проекте dwh описано в п. 3.1.3.1.

4.2.2.2.2 Запуск задачи инициализации источника данных

Если запуск задачи инициализации нужного источника данных еще не производился, необходимо выполнить обновление очереди задач инициализации источников данных.

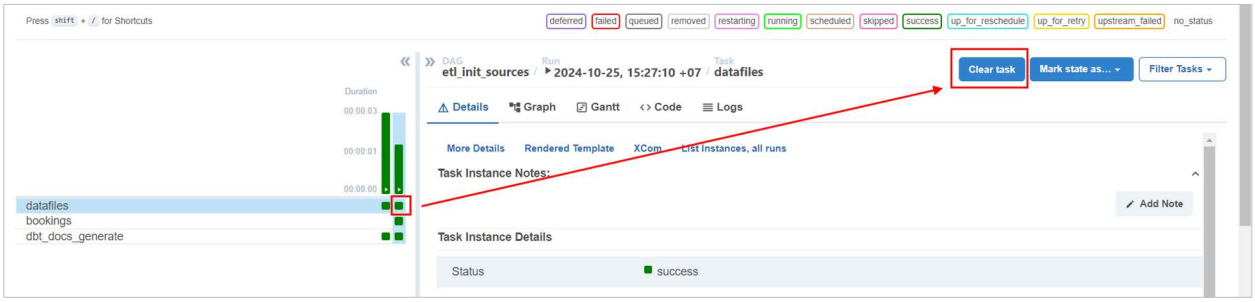
Для обновления очереди задач инициализации источников данных необходимо перейти к представлению задач конвейера etl_init_sources и выполнить ручной запуск очереди задач нажатием на кнопку «Trigger DAG» (см. Рис. 15).

Рис. 15. Обновление очереди задач инициализации источников данных



Для запуска задачи инициализации источника данных необходимо в списке задач конвейера etl_init_sources выбрать задачу, соответствующую нужному источнику данных, и нажать кнопку «Clear task» (см. Рис. 16).

Рис. 16. Выбор задачи инициализации источника данных



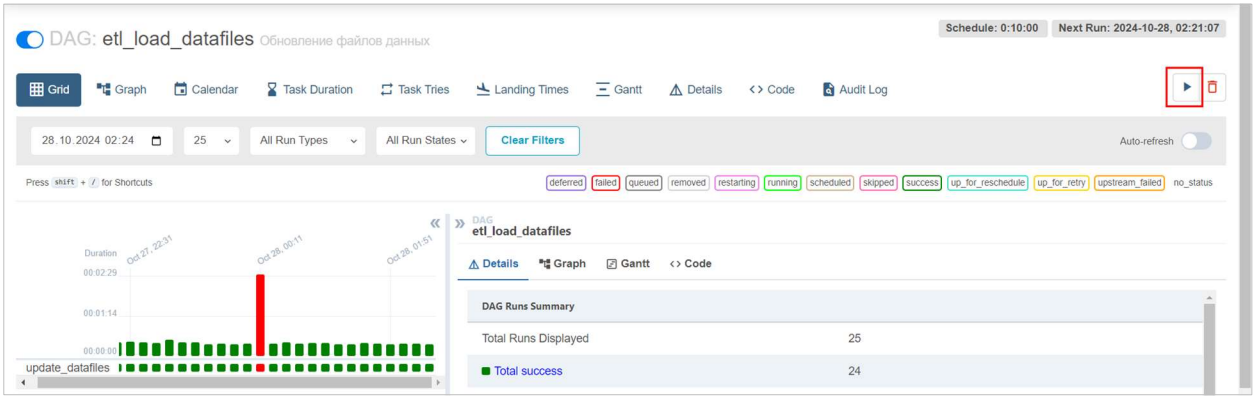
В появившемся окне необходимо снять выделение со всех кнопок включения дополнительных задач и нажать кнопку «Clear» (см. Рис. 17).

Рис. 17. Запуск задачи инициализации источника данных

4.2.2.3 Запуск обновления файлов данных

Для запуска обновления файлов данных необходимо перейти к представлению задач конвейера etl_load_datafiles и выполнить ручной запуск задач нажатием на кнопку «Trigger DAG» (см. Рис. 18).

Рис. 18. Запуск обновления файлов данных



4.2.2.4 Запуск задач загрузки и обработки данных

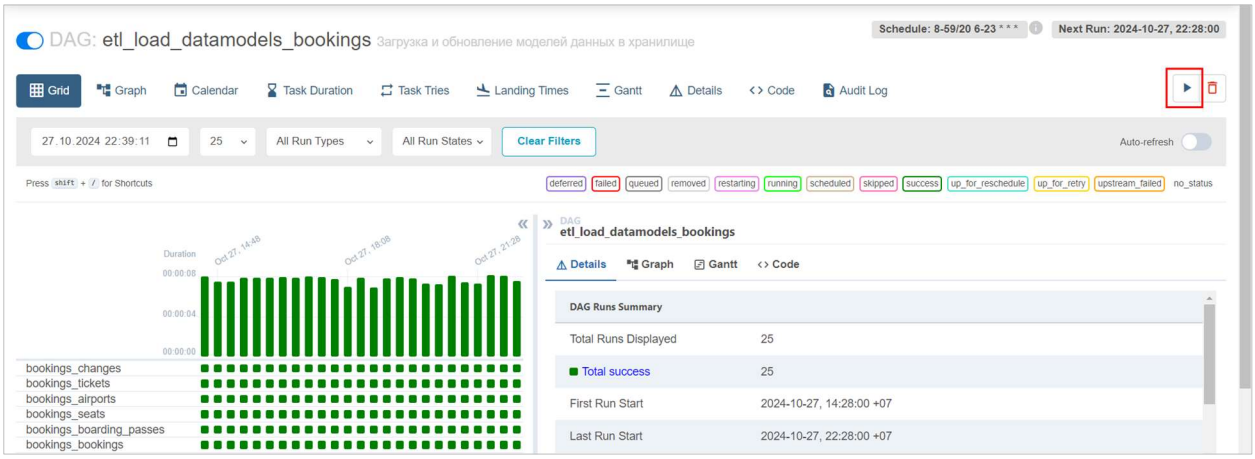
Панель управления конвейерами предоставляет следующие варианты ручного запуска задач загрузки и обработки данных:

- запуск полной очереди задач конвейера загрузки и обработки данных,
- запуск загрузки и обработки данных для отдельной модели dwh.

4.2.2.4.1 Запуск полной очереди задач конвейера загрузки и обработки данных

Для запуска полной очереди задач конвейера загрузки и обработки данных необходимо перейти к представлению задач нужного конвейера (типа load_datamodels) и выполнить ручной запуск очереди задач нажатием на кнопку «Trigger DAG» (см. Рис. 19).

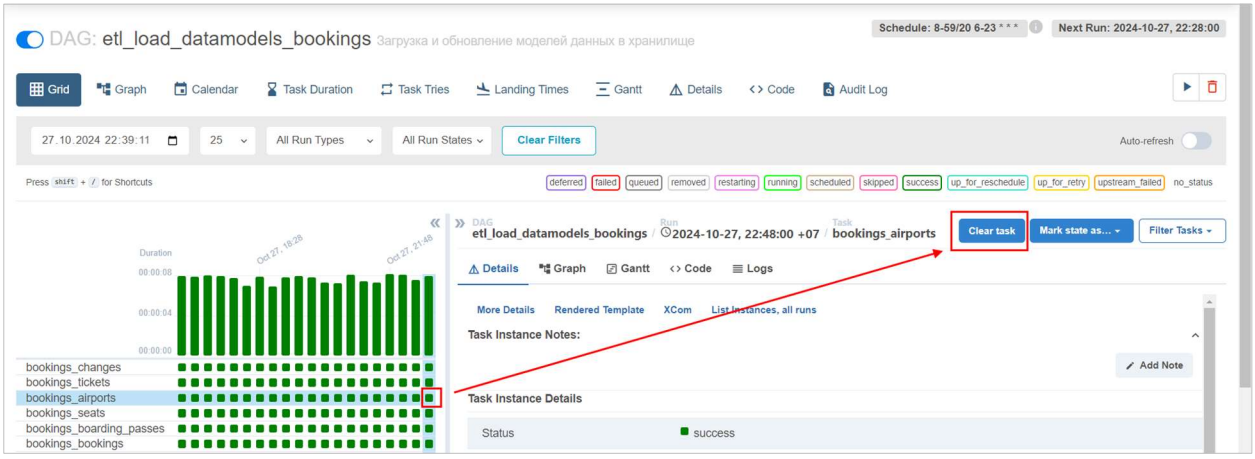
Рис. 19. Запуск полной очереди задач конвейера загрузки и обработки данных



4.2.2.4.2 Запуск загрузки и обработки данных для отдельной модели dwh

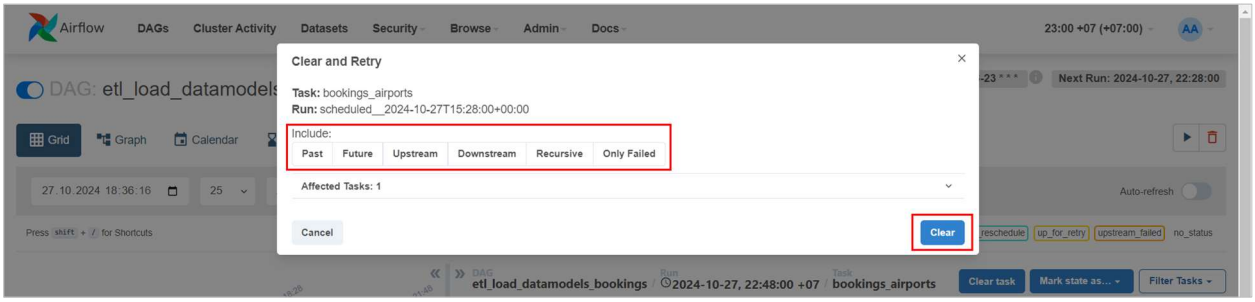
Для запуска загрузки и обработки данных для отдельной модели dwh необходимо в списке задач конвейера выбрать задачу, соответствующую нужной модели dwh, и нажать кнопку «Clear task» (см. Рис. 20).

Рис. 20. Выбор задачи загрузки и обработки данных



В появившемся окне необходимо снять выделение со всех кнопок включения дополнительных задач и нажать кнопку «Clear» (см. Рис. 21).

Рис. 21. Запуск задачи загрузки и обработки данных

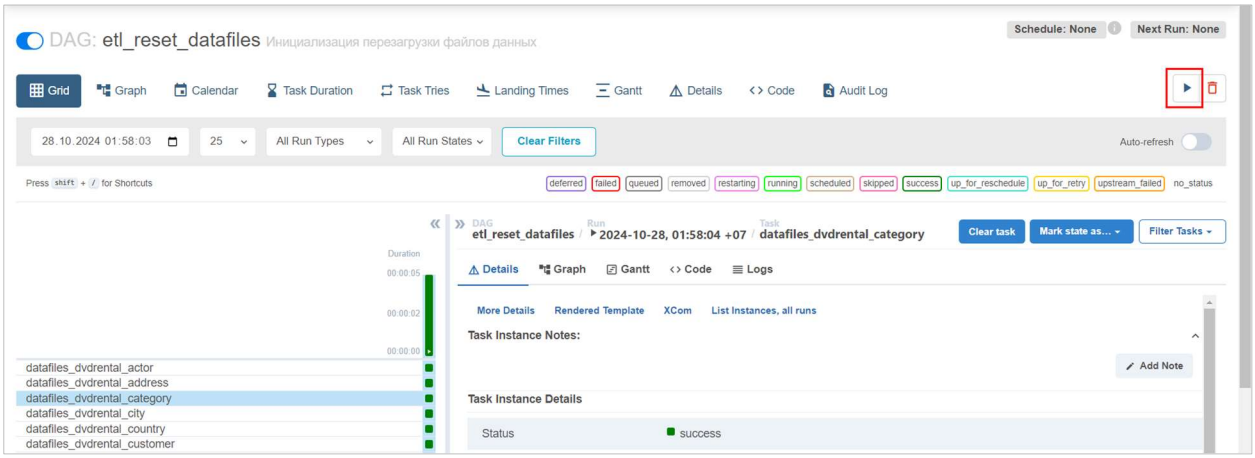


4.2.2.5 Запуск перезагрузки файлов данных

Если запуск перезагрузки нужного файла данных еще не производился, необходимо выполнить обновление очереди задач перезагрузки файлов данных.

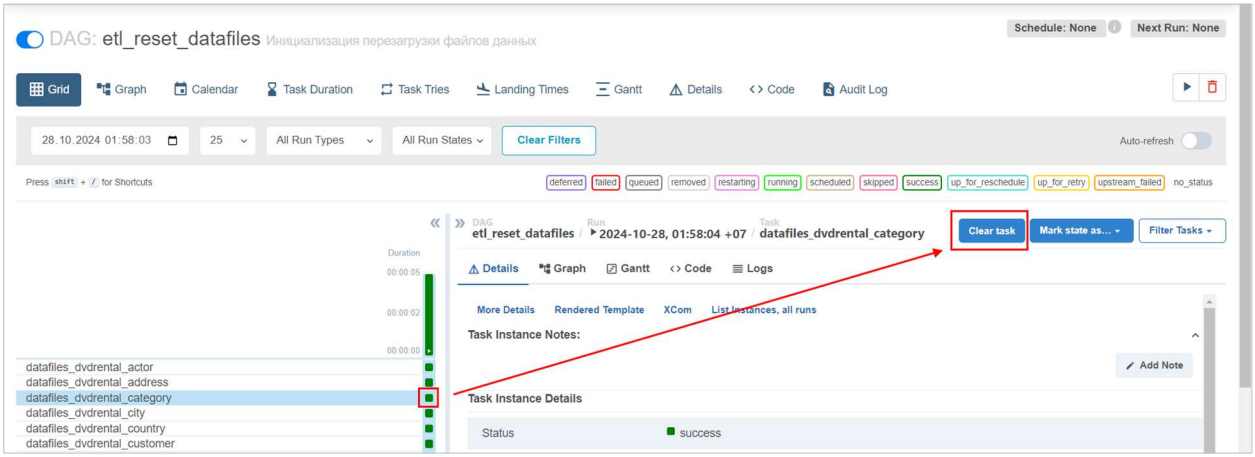
Для обновления очереди задач перезагрузки файлов данных необходимо перейти к представлению задач конвейера etl_reset_datafiles и выполнить ручной запуск очереди задач нажатием на кнопку «Trigger DAG» (см. Рис. 22).

Рис. 22. Обновление очереди задач перезагрузки файлов данных



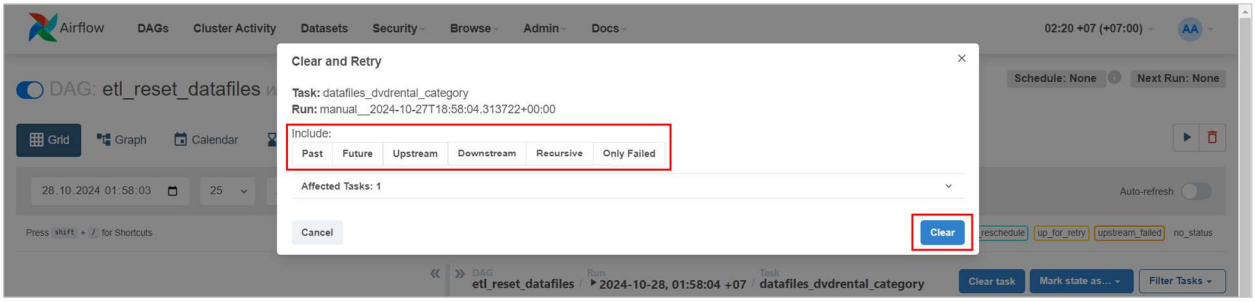
Для запуска перезагрузки отдельного файла данных необходимо в списке задач конвейера etl_reset_datafiles выбрать задачу, соответствующую нужному файлу данных, и нажать кнопку «Clear task» (см. Рис. 23).

Рис. 23. Выбор задачи перезагрузки отдельного файла данных



В появившемся окне необходимо снять выделение со всех кнопок включения дополнительных задач и нажать кнопку «Clear» (см. Рис. 24).

Рис. 24. Запуск задачи перезагрузки отдельного файла данных



4.2.2.6 Запуск перезагрузки данных в хранилище

Панель управления конвейерами предоставляет следующие варианты ручного запуска перезагрузки данных в хранилище:

- запуск перезагрузки отдельной модели dwh,
- запуск перезагрузки всех моделей dwh, связанных с отдельным источником данных.

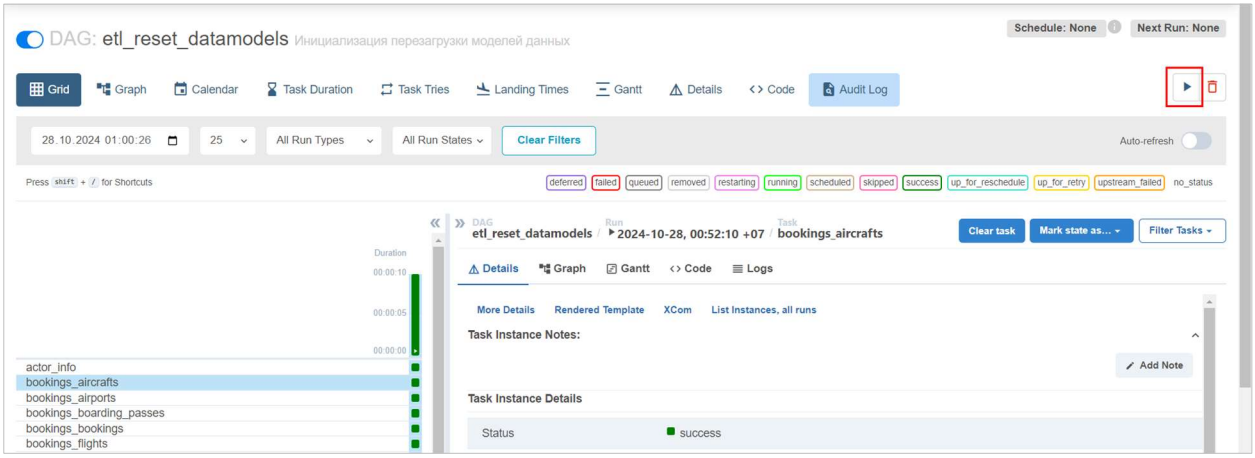
Перезагрузка модели dwh включает пересоздание структуры и перезагрузку данных модели, а также пересоздание структур и перезагрузку данных для всех моделей-последователей (моделей dwh, зависимых от данной модели).

4.2.2.6.1 Запуск перезагрузки отдельной модели dwh

Если запуск перезагрузки для нужной модели dwh еще не производился, необходимо выполнить обновление очереди задач перезагрузки моделей dwh.

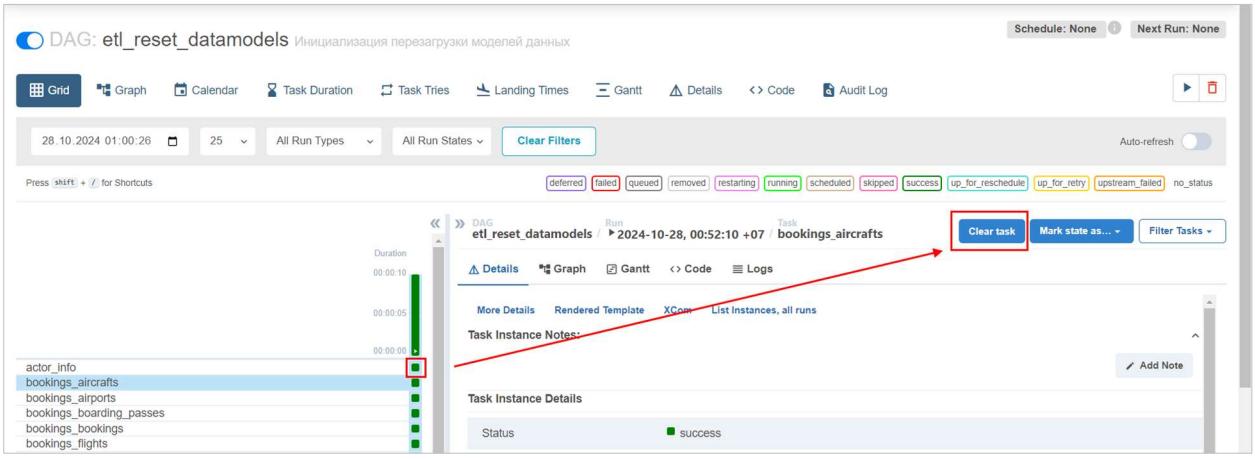
Для обновления очереди задач перезагрузки моделей dwh необходимо перейти к представлению задач конвейера etl_reset_datamodels и выполнить ручной запуск очереди задач нажатием на кнопку «Trigger DAG» (см. Рис. 25).

Рис. 25. Обновление очереди задач перезагрузки моделей dwh



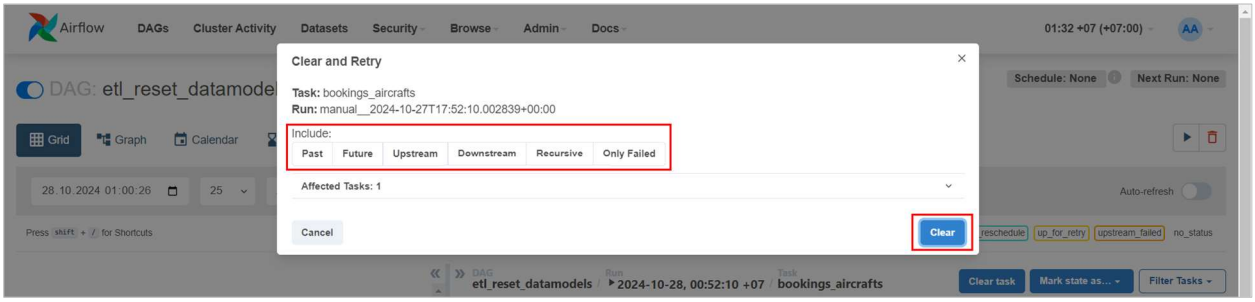
Для запуска перезагрузки отдельной модели dwh необходимо в списке задач конвейера etl_reset_datamodels выбрать задачу, соответствующую нужной модели dwh, и нажать кнопку «Clear task» (см. Рис. 26).

Рис. 26. Выбор задачи перезагрузки отдельной модели dwh



В появившемся окне необходимо снять выделение со всех кнопок включения дополнительных задач и нажать кнопку «Clear» (см. Рис. 27).

Рис. 27. Запуск задачи перезагрузки отдельной модели dwh

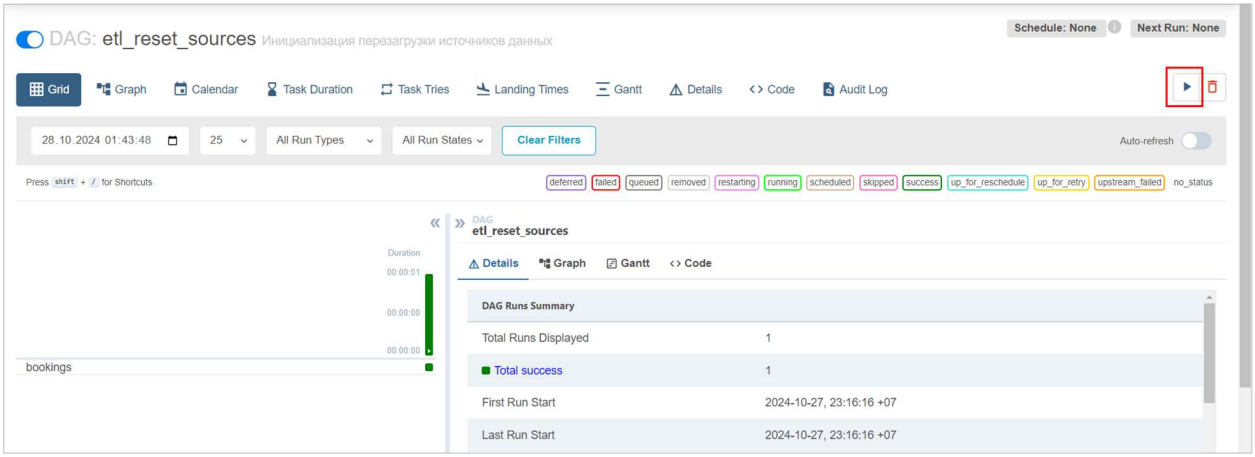


4.2.2.6.2 Запуск перезагрузки для моделей dwh, связанных с отдельным источником данных

Если запуск перезагрузки данных для нужного источника еще не производился, необходимо выполнить обновление очереди задач перезагрузки данных для всех источников.

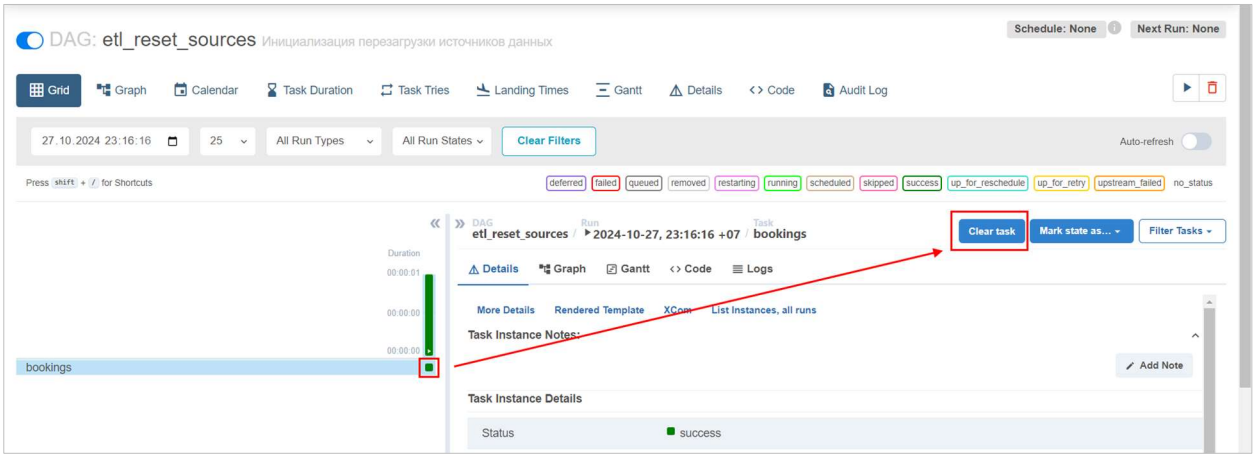
Для обновления очереди задач перезагрузки данных для всех источников необходимо перейти к представлению задач конвейера etl_reset_sources и выполнить ручной запуск очереди задач нажатием на кнопку «Trigger DAG» (см. Рис. 28).

Рис. 28. Обновление очереди задач перезагрузки данных для всех источников



Для запуска перезагрузки всех моделей dwh, связанных с отдельным источником данных, необходимо в списке задач конвейера etl_reset_sources выбрать задачу, соответствующую нужному источнику данных, и нажать кнопку «Clear task» (см. Рис. 29).

Рис. 29. Выбор задачи перезагрузки всех моделей dwh, связанных с отдельным источником данных



В появившемся окне необходимо снять выделение со всех кнопок включения дополнительных задач и нажать кнопку «Clear» (см. Рис. 30).

Рис. 30. Запуск задачи перезагрузки всех моделей dwh, связанных с отдельным источником данных