

# Введение в ETL

## Оглавление

- > [Оглавление](#)
- > [Что такое ETL](#)
- > [Принципы построения ETL](#)
  - [Будьте готовы](#)
- > [Обзор планировщиков](#)
  - [Базовые варианты](#)
  - [Платные шедулеры](#)
  - [Open Source](#)
- > [Почему Airflow](#)
- > [Глоссарий](#)

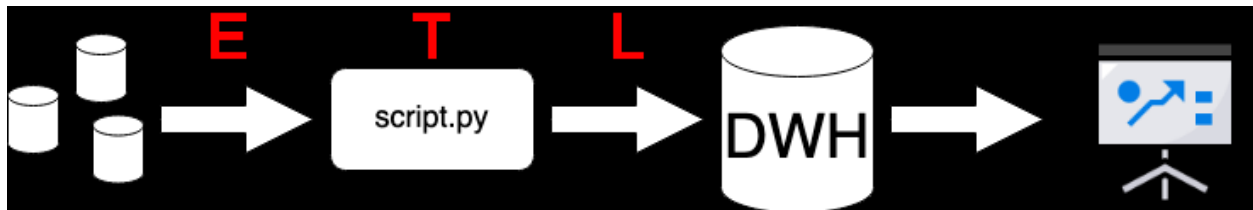
## Что такое ETL

**ETL** - это процесс миграции данных из одного хранилища данных в другое. ETL требуется не всегда - приведем пример. Поставлена задача посмотреть воронку продаж, посчитать конверсию и вывести динамику использования сервисом, логи

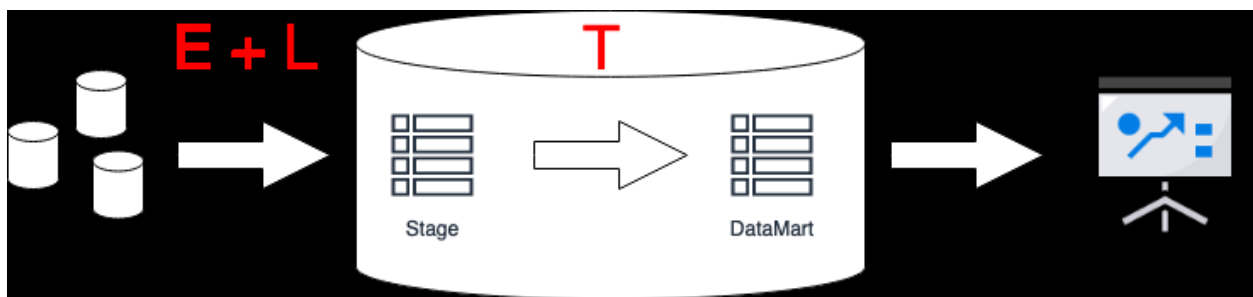
которого лежат в одной базе данных. Для такой задачи ETL не требуется, мы можем выполнить ее вручную, получив данные из БД различными способами.

Однако, в случае нескольких источников и требовании анализировать данные с какой-либо периодичностью, то без ETL не обойтись.

Пример простого ETL процесса



Пример простого ELT процесса



Основное отличие ELT от ETL процесса заключается в том, что в ELT процессе данные сразу загружаются в хранилище данных, где производится этап transform и последующие вспомогательные (например создание витрин данных).

## Принципы построения ETL

- **Чистый код** - хорошим тоном считается следовать правилам `per8`
- **Простота** - старайтесь писать читаемый и лаконичный код
- **Единообразие** - необходимо стремиться к единым принципам построения ваших дата-пайплайнов
- **Время выполнения пайплайна** - необходимо следить за SLA, например ежедневный процесс не должен выполняться более суток

- **Меньше сетевого трафика** - используйте только необходимые данные, не перегружая систему. В данном случае можем заметить преимущество ELT - данные выгружаются в хранилище один раз по сравнению как минимум с двумя в случае ETL.
- **Работа с репликой** - используйте только реплики источников для предотвращения неожиданных падений систем/баз данных/др.
- **Оптимизация забора данных** - пользуйтесь индексами, пишите оптимальные запросы, старайтесь оптимизировать ваш дата-пайплайн.
- **Способы загрузки данных (SCD)** - ориентируйтесь на количество данных, чтобы принять решение о применении инкрементальной загрузки.
- **Партиционирование** - при возможности используйте партиции.
- **Инкрементальный пересчет витрин** - стоит пересчитывать только новые данные витрины.
- **Загрузка всего без ограничений** - получать все данные в большинстве случаев - нормально.
- **Избавляться от неактуального** - периодически проводите аудит вашего хранилища, чтобы не перегрузить его.
- **Идемподентность** (свойство при повторном запуске выдавать точно такой же результат как при первом запуске) - отдавайте предпочтение merge перед insert для избежания возникновения дубликатов.
- **Аудиторский след** - стоит хранить данные, такие как в источниках

## Будьте готовы



- Отсутствие целостности
- Сетевые проблемы
- Незапланированные изменения
- Пайплайны задерживаются
- Данные в разных системах противоречивы

# Обзор планировщиков

## Базовые варианты

### > CRON

Самый дешевый и простой способ загрузки данных без дополнительных возможностей и функционала.

 Плюсы	 Минусы
Максимально простой	Максимально простой
Untitled	
Untitled	

### > JENKINS / GITLAB CI

Инструмент для CI / CD - формально можно использовать для ETL, однако не является стандартом и используется скорее для непрерывной интеграции.

### > Написать свой

Подойдет больше всего вам, однако разработка трудозатратная, займет много времени и потребует квалифицированных программистов.

## Платные шедулеры

- Дорогие
- Нет доступа к коду
- Но есть поддержка
- Визуальный редактор

### > MS SQL SERVER INTEGRATION SYSTEM

- Интегрирован в MS SQL Server
- Транзакционность из коробки
- Визуальный редактор

---

#### > ORACLE DATA INTEGRATOR

- Входит в систему Oracle
- Визуальный редактор

---

#### > INFORMATICA POWER CENTER

- Визуальный редактор.
- Встроенные утилиты контроля качества данных.

---

#### > SAS DATA INTEGRATION STUDIO

- Бесплатное обучение
- Визуальное построение пайплайнов
- Язык SAS Base
- SAS Access – плагины для источников

## Open Source

- Бесплатные
- Можно посмотреть в код
- Можно контрибьютить

#### > Apache OOOIE

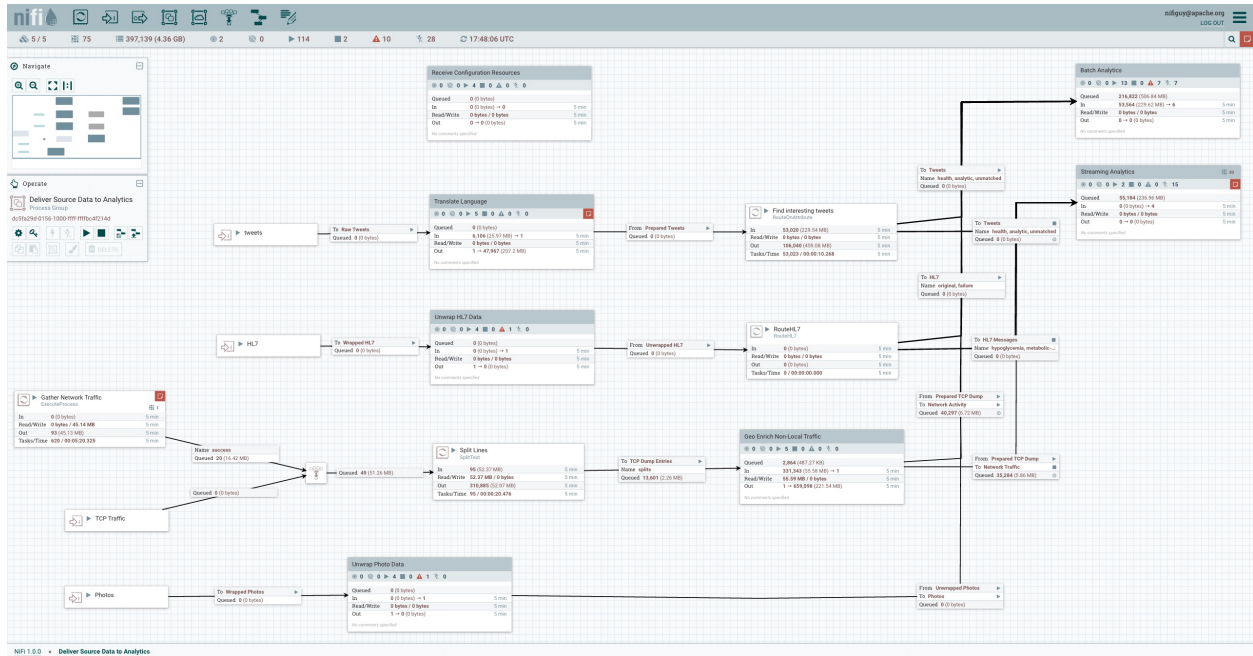
- Планировщик для Hadoop
- Пайплайны на xml

---

#### > Apache NIFI

- Работает внутри Hadoop
- В том числе с внешними источниками

- Потокковая и батчевая обработка



Пример дата пайплайна в Apache NIFI

## > Talend

- Open Source
- Визуальный редактор
- Потоки между задачами
- Требует знания Java



Пример дата пайплайна в Talend

## > Luigi

- Open Source
- Код на Питоне
- Работает поверх артефактов
- Нет встроенного планировщика
- Почти не развивается





6. Интеграция с основными источниками
7. Кастомизация
8. Масштабирование
9. Большое комьюнити

## Глоссарий

**ETL** - это процесс миграции данных из одного хранилища данных в другое.

**Идемпоидентность** - это свойство при повторном запуске выдавать точно такой же результат как при первом запуске.

**Дата пайплайн** - процесс выгрузки, обработки, трансформации и загрузки данных.