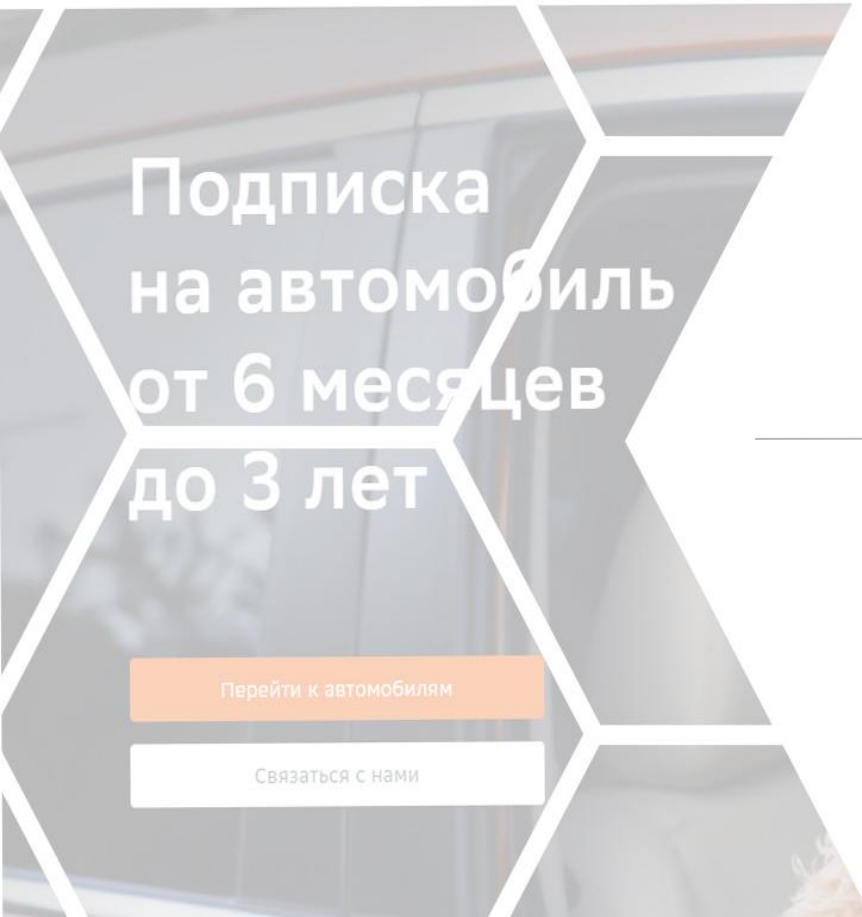
Финальная работа

СБЕР АВТОПОДПИСКА



Анализ сайта “СберАвтоподписка”

исполнитель: Чмир Михаил Петрович

«Введение в Data Science»,
специализация: Data Engineer

Skillbox

СБЕР АВТОПОДПИСКА

Коротко о продукте

СберАвтоподписка - это сервис долгосрочной аренды автомобилей для физических лиц.

Клиент платит фиксированный ежемесячный платеж и получает в пользование машину на срок от 6 месяцев до 3-х лет.

Детали можно получить на сайте: `https://podpiska.sberauto.com`

Это новый для российского рынка способ владения автомобилем и выступает в качестве альтернативы кредиту.

СБЕР АВТОПОДПИСКА

Коротко об анализе на сервисе

На сайте пользователь совершает некоторые действия, которые подразделяют на **целевые действия** и **нецелевые действия**. Например, целевое действие - нажимаем кнопки типа “*Оставить заявку*”, “*Заказать заявку*” и т.д. Нецелевые действия - просмотр карточек авто или просто “блуждание” по страницам сайта.

Все данные обрабатываются и аккумулируются посредством веб-аналитического инструмента от компании Google - Google Analytics (GA) **.

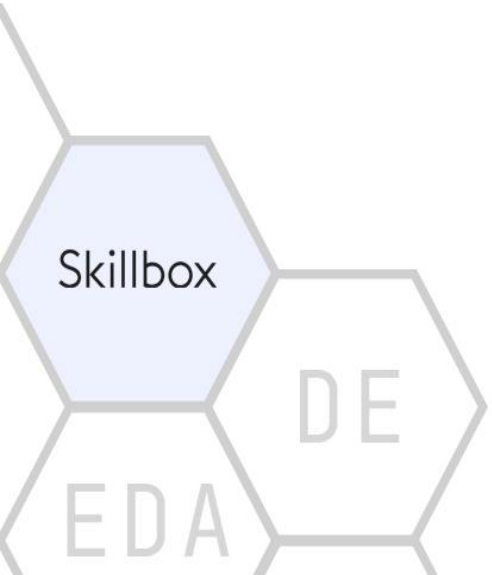
*** Google Analytics (GA) - это веб-аналитический инструмент, разработанный Google, который позволяет владельцам веб-сайтов и мобильных приложений анализировать трафик и поведение пользователей на их ресурсах. С его помощью можно получить подробную информацию о том, как пользователи взаимодействуют с сайтом, на каких страницах проводят больше времени, какие действия совершают, откуда пришли и т.д.*

Skillbox

DE

EDA

План работ



Провести подготовительную работу:

- ❏ Прочитать предоставленные датасеты;
- ❏ Ознакомится с описаниями представленных атрибутов;
- ❏ Оценить полноту и чистоту данных. Привести данные в удобный / нормальный вид для дальнейшей работы.

Провести разведочный анализ данных:

- ❏ Базовая очистка данных: дубликаты, пустые значения, типизация данных, ненужные атрибуты и т.д.;
- ❏ Посмотреть на распределение ключевых атрибутов и их отношения.

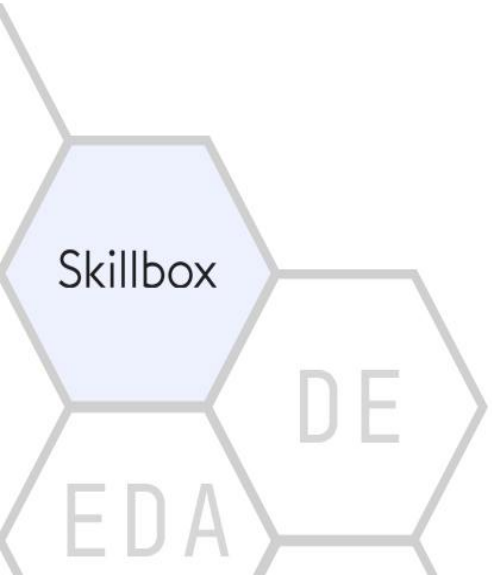
Выполнить задание согласно специализации - Data Engineer:

- ❏ Определить план, механизмы и инструменты для разработки и поддержки инфраструктуры хранения и обработки данных.

Подготовительная работа

Провел подготовительную работу:

- Получил датасеты для работы:
  - `ga_hits-002.pkl  - 4.09 Gb;`
  - `ga_sessions.pkl  - 377 MB;`
- Ознакомился с инструкциями, документацией к финальной работе;
- Ознакомился с описаниями представленных атрибутов:
  - *Данного блокнота нет в проекте, так как это черновик, предназначенный только для подготовительных работ.*
- После первичного изучения предоставленных данных, был составлен план проекта и структура проекта, для дальнейшей работы.

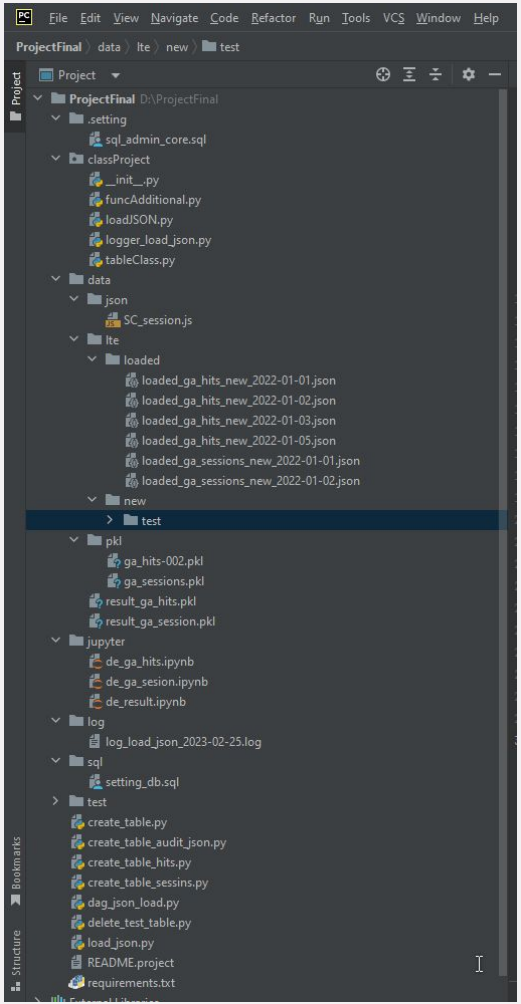


Подготовительная работа

Skillbox

DE

EDA



Структура проекта:

1. Папка “.**setting**” - файлы для настройки базы данных, проекта и т.д;
2. “**classProject**” - классы, модули проекта.
3. “**data**” - папка содержит все типы данных, входящие, промежуточные, исходящие, результат;
4. “**jupyter**” - файлы Jupyter Notebook;
5. “**log**” - файлы системы аудита, логирование;
6. “**sql**” - файлы для работы с SQL сервером;
7. “**test**” - промежуточные данные, необходимые для тестирования.

exploratory data analysis

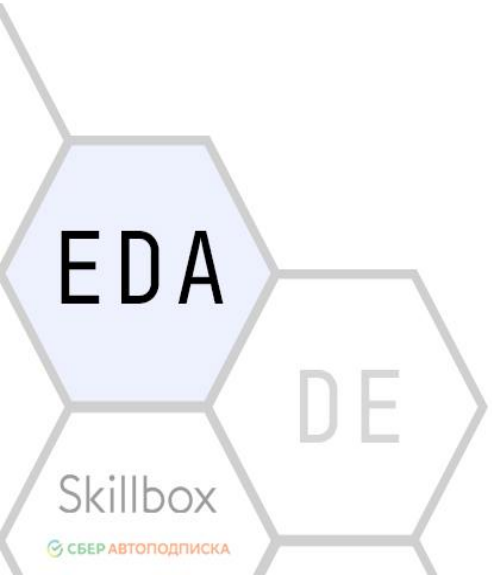
EDA - *процесс исследования и анализа данных, направленный на получение полезных знаний о наборе данных.*

**** Данные работы этапа "EDA" расположены в папке* **"jupyter"**

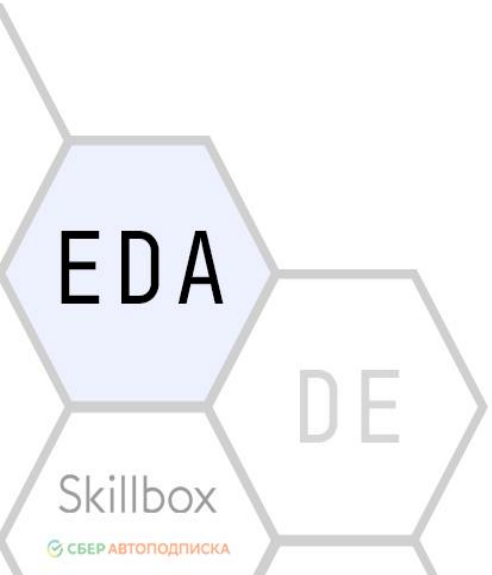
1. `de_ga_hits.ipynb` файл с обработкой данных датасета "ga_hits_002.pkl". Промежуточный результат сохранен в проект в "`data/result_ga_hits.pkl`";

2. `de_ga_session.ipynb` файл с обработкой данных датасета "ga_sessions.pkl". Промежуточный результат сохранен в проект в "`data/result_ga_session.pkl`";

3. `de_result.ipynb` файл с обработкой данных датасета промежуточных результатов: объединение двух датасетов, анализ данных, нормализация, визуализация



exploratory data analysis



EDA - *процесс исследования и анализа данных, направленный на получение полезных знаний о наборе данных.*

**** Данные работы этапа "EDA" расположены в папке* **"jupyter"**

В результате проведения мероприятий на данном этапе:

1. Получено понимание данных, выбраны наиболее подходящие методы для анализа.
2. Проведена полная очистка данных: работа с NaN записями, обработка дубликатов, создание новых признаков на основе имеющихся признаков, например признак "Целевого действия", который принимает значения 1 или 0 и однозначно идентифицирует событие, марка авто, модель авто и т.д.
3. Проведена нормализация данных: удалены ненужные признаки, в некоторых признаках значения приведены к рабочему распределению (менее 1% уникальных признаков были заменены категорией "other"), рассмотрена типизация данных.
4. Два датасета были объединены по ключевому признаку. Построено распределение посещений и визитов во времени.

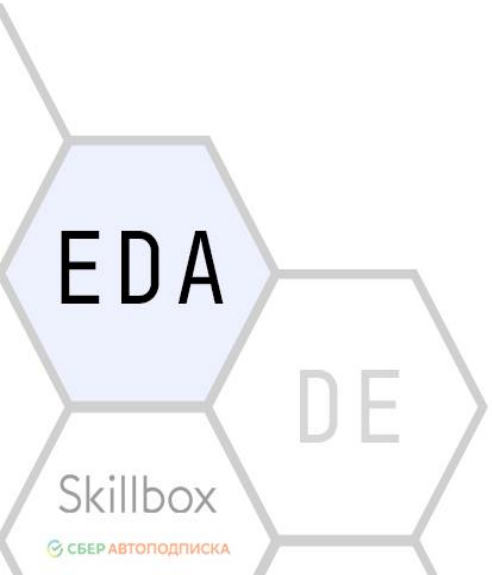
exploratory data analysis

EDA - *процесс исследования и анализа данных, направленный на получение полезных знаний о наборе данных.*

**** Данные работы этапа "EDA" расположены в папке* **"jupyter"**

5. Проведен дополнительный анализ данных после объединения датасетов.
6. Проведена дополнительная очистка и нормализация объединенных датасетов.
7. Для подсчета корреляции между качественными и числовыми переменными использовали ***коэффициент корреляции Крамера***.
8. Построены графики распределения ключевых атрибутов и их отношения.

Для более конкретного и предметного обсуждения пунктов выполнения задач рассматриваемого этапа можно обратиться к содержанию файлов проекта Jupyter Notebook.



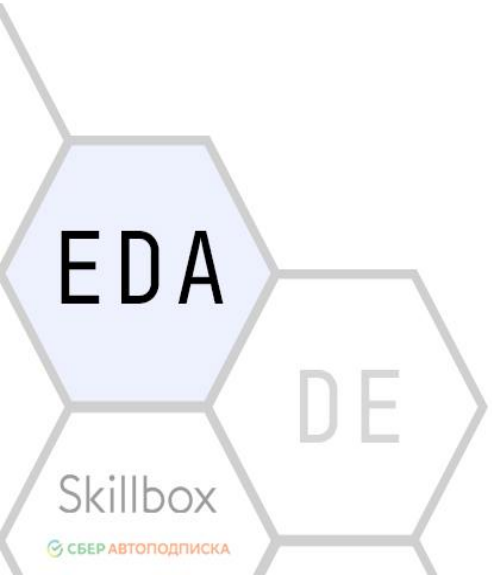
exploratory data analysis

EDA - *процесс исследования и анализа данных, направленный на получение полезных знаний о наборе данных.*

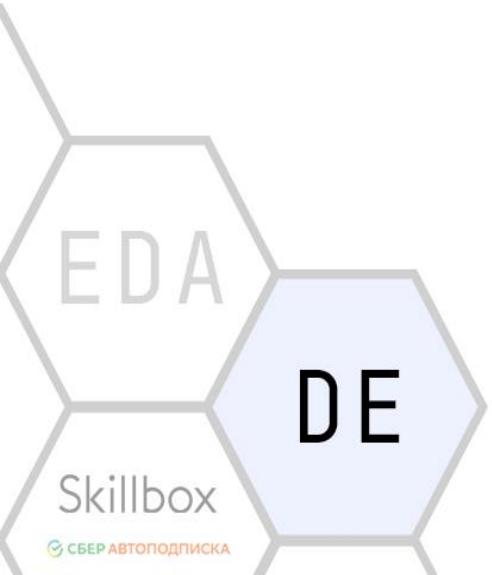
**** Данные работы этапа “EDA” расположены в папке* **“jupyter”**

Итоги:

В рамках выбранной специализации, на первом этапе мы лишь обрабатывали и подготавливали данные для первичного технического анализа. Построили рабочий, объединенный датасет. Никаких аналитических выводов не строилось.



Data Engineer



В рамках задачи по специализации мы должны:

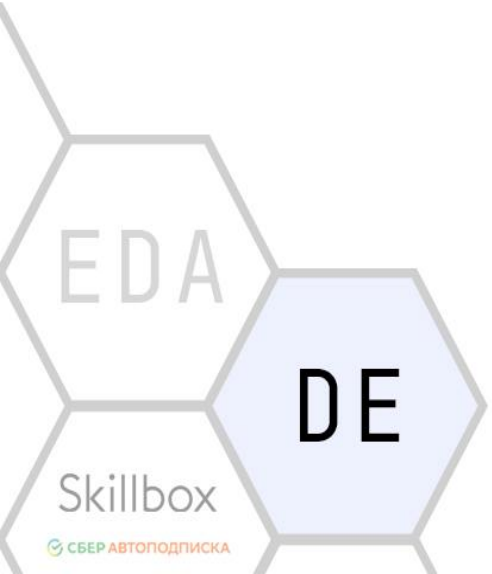
- ❏ Разработать технологическую структуру хранения данных *.
- ❏ Разработать механизмы загрузки данных в созданную структуру хранения **.
- ❏ Развертывание созданной системы на сервисе Airflow.

* будем определять всю ***технологию*** по созданию и быстрому развертыванию на продакшене структуры хранения данных. Технология должна быть универсальной, по мере возможности, и как максимум идемпотентной *** ;

** будут определены, созданы, описаны(*документирование функций*) механизмы и инструменты по загрузке данных. Данная система должна быть безопасной, отказоустойчивой, с адаптивной подсистемой аудита действий и аудита данных.

*** ***Идемпоте́нтность*** *— свойство объекта или операции при повторном применении операции к объекту давать тот же результат, что и при первом.*

Data Engineer



Технологическая структура хранения данных.

Инструменты:

- в качестве базы данных выбрана PostgreSQL версии 15;
- в качестве инструмента создания базы данных и первичных административных настроек, вплоть до тестирования - создан SQL скрипт, который создаст базу данных, создаст пользователей, определит роли, схемы, права. В самом скрипте имеются готовые заготовки для тестирования операций DDL, DML во вновь созданной базе данных;
- в качестве инструмента для создания таблиц в базе данных и их ограничений - будем использовать Python скрипт, с использованием программной библиотеки SQLAlchemy* и технологии ORM.

* **SQLAlchemy** — *это программная библиотека на языке Python для работы с реляционными СУБД с применением технологии ORM. Служит для синхронизации объектов Python и записей реляционной базы данных. SQLAlchemy позволяет описывать структуры баз данных и способы взаимодействия с ними на языке Python* ***без использования SQL****. Библиотека была выпущена в феврале 2006 под лицензией открытого ПО MIT.*
Работает back-end для баз данных: MySQL, PostgreSQL, SQLite, Oracle и других, ***между которыми можно переключаться изменением конфигурации****.*

Data Engineer

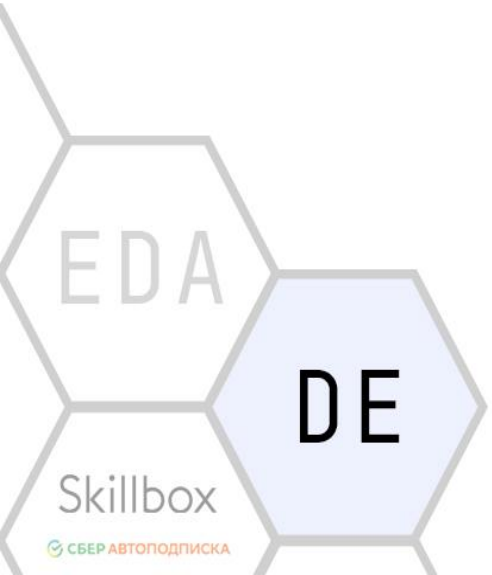
Технологическая структура хранения данных.

Итоги:

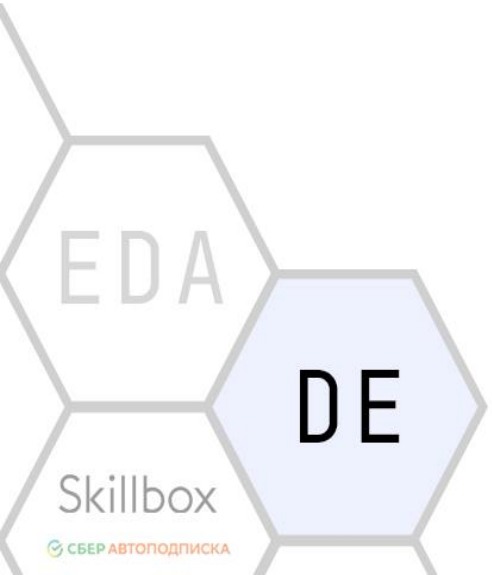
- SQL-скрипт в данной версии создан для определенной СУБД. Для СУБД другого типа необходимо по аналогии, скорректировать скрипт в виду особенностей применяемой СУБД. *Например, у меня, при развертывании на MS SQL, это заняло не более получаса.*
- Использование Python скрипта, с применением программной библиотеки SQLAlchemy и технологии ORM, при создании таблиц (*для каждой таблицы, определен свой скрипт*) позволяет применять данный инструмент в не зависимости от вида СУБД. Для изменения вида СУБД в скрипте достаточно изменить только строку подключения к серверу базы данных, которая определена как глобальная переменная в начале каждого скрипта.

Шаги по развертыванию системы хранения:

1. Создаем базу данных `sql/setting_db.sql`
2. Добавляем в базу Адм. функции `.setting/sql_admin_core.sql`
3. Проверяем создание в базе тестовой таблицы `create_table.py`
4. Создаем таблицу аудита `json_load_audit` из `create_table_audit_json.py`
5. Создаем таблицу событий `ga_sessions` из `create_table_sessins.py` (*она имеет первичный ключ*)
6. Создаем таблицу визитов `ga_hits` из `create_table_hits.py` (*связывается с sessions по вторичному ключу*)
7. Удалить тестовые таблицы `delete_test_table.py`



Data Engineer



Загрузка данных.

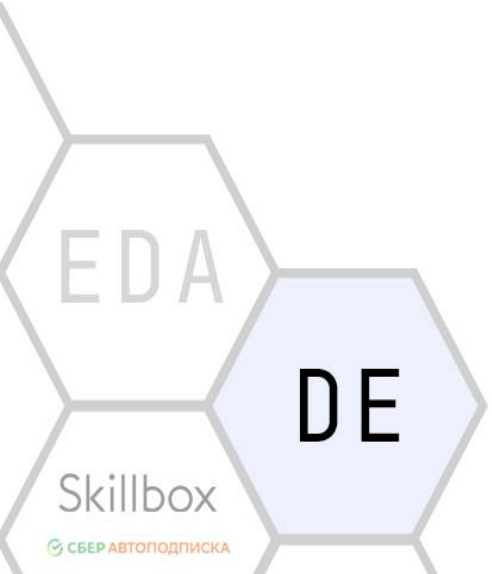
В качестве загружаемых данных нам предложены JSON файлы. Файлов JSON два вида - для загрузки в таблицу “`sessions`” и в таблицу “`hits`”.

1. При запуске главной функции, на первом этапе проводится проверка на тип файла JSON. В обработке участвуют только JSON файлы.

2. Далее проводится поиск только файлов в имени которых встречается слово “`session`”. Если такой файл найден, вызывается функция загрузки “`load_json_session`” из пакета.модуля “`classProject.loadJSON`”. Загрузка файлов начинает именно с таблицы “`sessions`”, так как это первичная таблица и содержит первичный ключ для отношения с таблицей “`hits`”

3. Далее проводится поиск файлов в имени которых встречается слово “`hits`”. Функция загрузки для таких файлов “`load_json_hits`” из того же пакета.модуля.

Data
Engineer



Загрузка данных.

Первые три пункта, реализуют идею безопасности загрузки данных на уровне операционной системы (Далее - ОС).

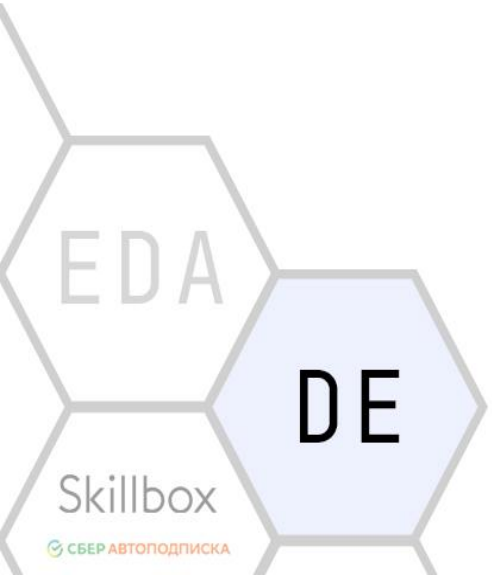
Далее…

4. При начале загрузки файла, информация об этом фиксируется в лог-файле (аудит на уровне ОС).

5. Перед началом обработки JSON-файла происходит проверка через функцию “`load_file_if_not_loaded_yet`” на предмет того, загружался ли уже данный файл. Данная система аудита на уровне базы данных, предполагает, при успешной загрузки файла, информация об имени файла, дата его создания и дата его загрузки, заносятся в таблицу аудита в базе данных. Если имя файла уже присутствует в таблице аудита, файл не загружается, а информация об этом заносится в лог-файл.

6. Начинается обработка (*парсинг*) JSON файла. Если JSON не соответствует структуре по первичному ключу, или данные для первого ключа отсутствуют, т.е. JSON файл не содержит данных, такой файл далее не обрабатывается и об этом фиксируется информация в лог-файле.

Data Engineer



Загрузка данных.

7. Здесь же, происходит обработка NaN значений, которые содержатся в JSON файле.

8. Если JSON файл имеет данные, то его содержимое передается в функцию “`load_data`”, в которой осуществляется загрузка данных в соответствующую таблицу. Загрузка данных осуществляется по технологии `SQLAlchemy.ORM`

9. Любые stop-события как и ошибки внутри каждой функции обрабатываются, если необходимо перехватываются и обрабатываются с помощью отдельного модуля “`classProject.logger_load_json`”. Данный модуль настроен на сопровождение логирования событий в очень подробном виде. Для каждого события фиксируется, время, тип события, модуль и класс события, имя функции и строка в файле где данной событие было вызвано. Таким образом, в лог-файле можно получить точечное расположение в коде причины возникновения записи.

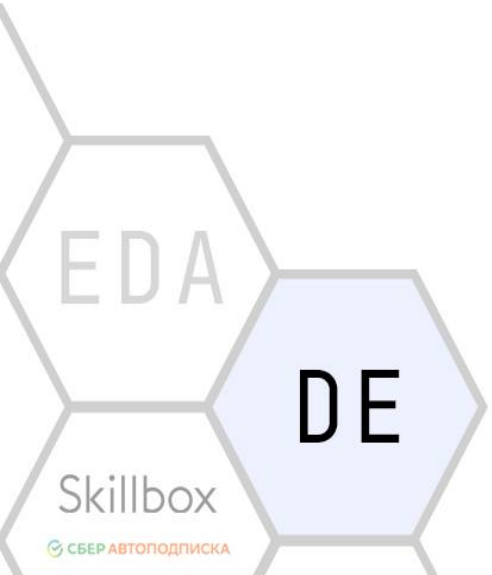
Data Engineer

Пример файла аудита.

```
2023-02-25 11:49:20,798 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_sessions_new_2022-01-01.json' - load_json_session:110
2023-02-25 11:49:20,893 - classProject.logger_load_json - INFO - Файл ga_sessions_new_2022-01-01.json не содержит данных.' - load_json_session:127
2023-02-25 11:49:20,893 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_sessions_new_2022-01-02.json' - load_json_session:110
2023-02-25 11:50:22,472 - classProject.logger_load_json - INFO - Загружено записей: '7277' - load_data:303
2023-02-25 11:50:22,477 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_SESSION: загружен файл 'ga_sessions_new_2022-01-02.json' - load_json_session:134
2023-02-25 11:50:22,478 - classProject.logger_load_json - INFO - Файл ga_sessions_new_2022-01-02.json был перемещен в data/lte/loaded с новым именем loaded_ga_sessions_new_2022-01-02.json - load_json_session:139
2023-02-25 11:50:22,531 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_session:144
2023-02-25 11:50:22,534 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_sessions_new_2022-01-03.json' - load_json_session:110
2023-02-25 11:50:37,509 - classProject.logger_load_json - INFO - Загружено записей: '8569' - load_data:303
2023-02-25 11:50:37,509 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_SESSION: загружен файл 'ga_sessions_new_2022-01-03.json' - load_json_session:134
2023-02-25 11:50:37,509 - classProject.logger_load_json - INFO - Файл ga_sessions_new_2022-01-03.json был перемещен в data/lte/loaded с новым именем loaded_ga_sessions_new_2022-01-03.json - load_json_session:139
2023-02-25 11:50:37,561 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_session:144
2023-02-25 11:50:37,564 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_sessions_new_2022-01-04.json' - load_json_session:110
2023-02-25 11:50:52,525 - classProject.logger_load_json - INFO - Загружено записей: '9383' - load_data:303
2023-02-25 11:50:52,525 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_SESSION: загружен файл 'ga_sessions_new_2022-01-04.json' - load_json_session:134
2023-02-25 11:50:52,526 - classProject.logger_load_json - INFO - Файл ga_sessions_new_2022-01-04.json был перемещен в data/lte/loaded с новым именем loaded_ga_sessions_new_2022-01-04.json - load_json_session:139
2023-02-25 11:50:52,587 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_session:144
2023-02-25 11:50:52,592 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_sessions_new_2022-01-05.json' - load_json_session:110
2023-02-25 11:51:09,321 - classProject.logger_load_json - INFO - Загружено записей: '9493' - load_data:303
2023-02-25 11:51:09,321 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_SESSION: загружен файл 'ga_sessions_new_2022-01-05.json' - load_json_session:134
2023-02-25 11:51:09,321 - classProject.logger_load_json - INFO - Файл ga_sessions_new_2022-01-05.json был перемещен в data/lte/loaded с новым именем loaded_ga_sessions_new_2022-01-05.json - load_json_session:139
2023-02-25 11:51:09,373 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_session:144
2023-02-25 11:51:09,377 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_hits_new_2022-01-01.json' - load_json_hits:59
2023-02-25 11:51:09,429 - classProject.logger_load_json - INFO - Файл ga_hits_new_2022-01-01.json не содержит данных.' - load_json_hits:76
2023-02-25 11:51:09,429 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_hits_new_2022-01-02.json' - load_json_hits:59
2023-02-25 11:52:22,857 - classProject.logger_load_json - INFO - Загружено записей: '73499' - load_data:234
2023-02-25 11:52:22,857 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_HITS: загружен файл'ga_hits_new_2022-01-02.json' - load_json_hits:83
2023-02-25 11:52:22,857 - classProject.logger_load_json - INFO - Файл ga_hits_new_2022-01-02.json был перемещен в data/lte/loaded с новым именем loaded_ga_hits_new_2022-01-02.json - load_json_hits:88
2023-02-25 11:52:22,933 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_hits:93
2023-02-25 11:52:22,958 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_hits_new_2022-01-03.json' - load_json_hits:59
[SQL: INSERT INTO admin_app.ga_hits (session_id, hit_date, hit_time, hit_number, hit_type, hit_referer, hit_page_path, event_category, event_action, event_label, event_value) VALUES (%(session_id)s, %(hit_date)s, %(
[parameters: {'session_id': '8418660712445111146.1641278319.1641278319', 'hit_date': datetime.date(2022, 1, 4), 'hit_time': '110239', 'hit_number': 18, 'hit_type': 'event', 'hit_referer': None, 'hit_page_path': 'sber
(Background on this error at: https://sqlalche.me/e/14/gkpj)' - load_data:226
2023-02-25 11:55:07,086 - classProject.logger_load_json - WARNING - Загрузка данных в таблицу GA_HITS для session_id=8418660712445111146.1641278319.1641278319 завершилась ошибкой: '(psycopg2.errors.ForeignKeyViolati
DETAIL: Ключ (session_id)=(8418660712445111146.1641278319.1641278319) отсутствует в таблице "ga_sessions".

[SQL: INSERT INTO admin_app.ga_hits (session_id, hit_date, hit_time, hit_number, hit_type, hit_referer, hit_page_path, event_category, event_action, event_label, event_value) VALUES (%(session_id)s, %(hit_date)s, %(
[parameters: {'session_id': '8418660712445111146.1641278319.1641278319', 'hit_date': datetime.date(2022, 1, 4), 'hit_time': '109660', 'hit_number': 15, 'hit_type': 'event', 'hit_referer': None, 'hit_page_path': 'sber
(Background on this error at: https://sqlalche.me/e/14/gkpj)' - load_data:226
2023-02-25 11:55:14,083 - classProject.logger_load_json - INFO - Загружено записей: '91562' - load_data:234
2023-02-25 11:55:14,083 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу GA_HITS: загружен файл'ga_hits_new_2022-01-04.json' - load_json_hits:83
2023-02-25 11:55:14,083 - classProject.logger_load_json - INFO - Файл ga_hits_new_2022-01-04.json был перемещен в data/lte/loaded с новым именем loaded_ga_hits_new_2022-01-04.json - load_json_hits:88
2023-02-25 11:55:14,135 - classProject.logger_load_json - INFO - Успешная загрузка данных в таблицу аудита - load_json_hits:93
2023-02-25 11:55:14,135 - classProject.logger_load_json - INFO - Начата загрузка файла 'ga_hits_new_2022-01-05.json' - load_json_hits:59
2023-02-25 11:56:44,549 - classProject.logger_load_json - INFO - Загружено записей: '92170' - load_data:234
```

Data Engineer



Загрузка данных.

10. Загрузка данных из файла производится по принципу “одна запись - одна фиксация”, таким образом, в случае ошибки, в базу не попадут только ошибочные записи, по которым будет необходимо провести дополнительное расследование.

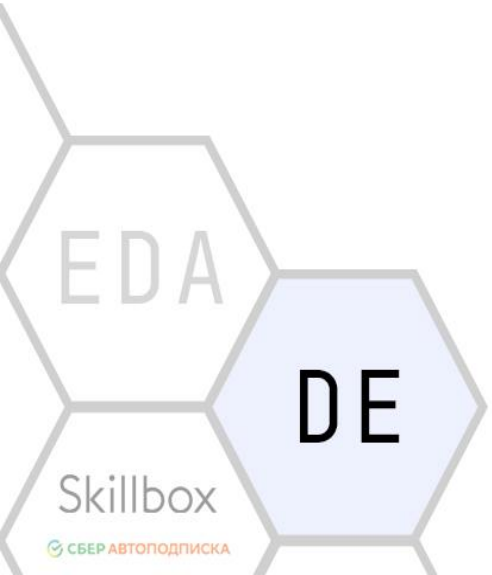
11. После успешной загрузки данных, в лог-файле будет зафиксировано данное событие, с указанием количества обработанных строк.

12. Далее в таблицу аудита заносится информация об обработанном успешно файле.

13. Обработанный файл (даже если он пустой), переименовывается, к имени файла добавляется префикс “`loaded_`” и перемещается в папку загруженных файлов “`data/lte/loaded`”. Папка “`data/lte/new`” в случае успешной загрузки данных, должна быть пуста.

14. Для файлов с данными “`hits`” обработка происходит по аналогичной схеме.

Data Engineer



Загрузка данных. Автоматизация процесса.

15. Для автоматизации процесса загрузки данных, можно воспользоваться любым способом уместным для настройки автоматизации задач ETL. Например: либо настроить запуск скрипт-файла “`load_json.py`” через утилиту “cron” в Unix-подобных ОС или через систему назначенных заданий по расписанию в ОС семейства Windows.

16. Для данного проекта был разработан Python-скрипт “`dag_json_load.py`” с применением библиотеки “`Airflow DAG`”* для размещения на платформе Airflow**.

17. В корне проекта, находится текстовый файл “`README.project`” в котором изложена пошаговая инструкция для развертывания проекта, с описанием всех модулей и файлов.

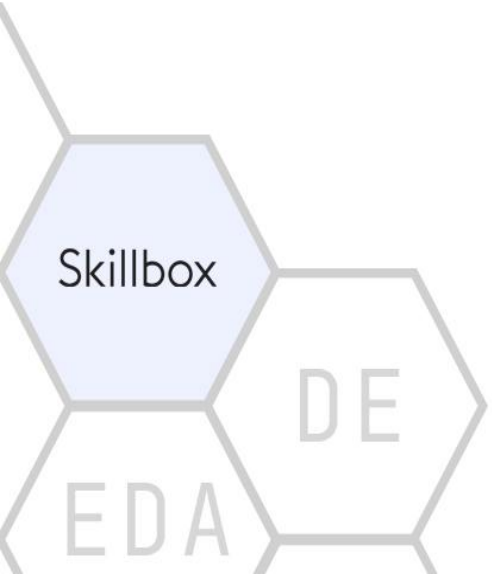
* Библиотека **Airflow DAG** - *это часть Apache Airflow, открытого и расширяемого инструмента управления рабочими процессами и автоматизации задач. DAG - это Directed Acyclic Graph, который представляет графическое представление задач, которые необходимо выполнить в процессе рабочего процесса.*

** **Airflow -** *это открытая платформа для создания, планирования и управления рабочих процессов (workflows) в компьютерных системах. Она позволяет разработчикам определять, планировать и мониторить рабочие процессы как код, используя Python.*

Финальная работа - Итоги

В результате работы было выполнено:

- ❏ Изучены предлагаемые данные, получено понимание данных, собраны необходимые методы для анализа;
- ❏ Данные очищены, дополнены, нормализованы. Построены необходимые распределения, найдены зависимости ключевых атрибутов и их отношения.
- ❏ Создана полноценная система ETL по обработки и загрузки данных из файлов JSON в реляционную базу данных для СУБД PostgreSQL. Соблюдены все требования, указанные при постановки задач.
- ❏ Процесс ETL размещен на платформе Airflow.



Благодарю Вас за внимание.

Skillbox.
«Введение в Data Science», специализация: Data Engineer
2023 год.