

## Домашнее задание 2

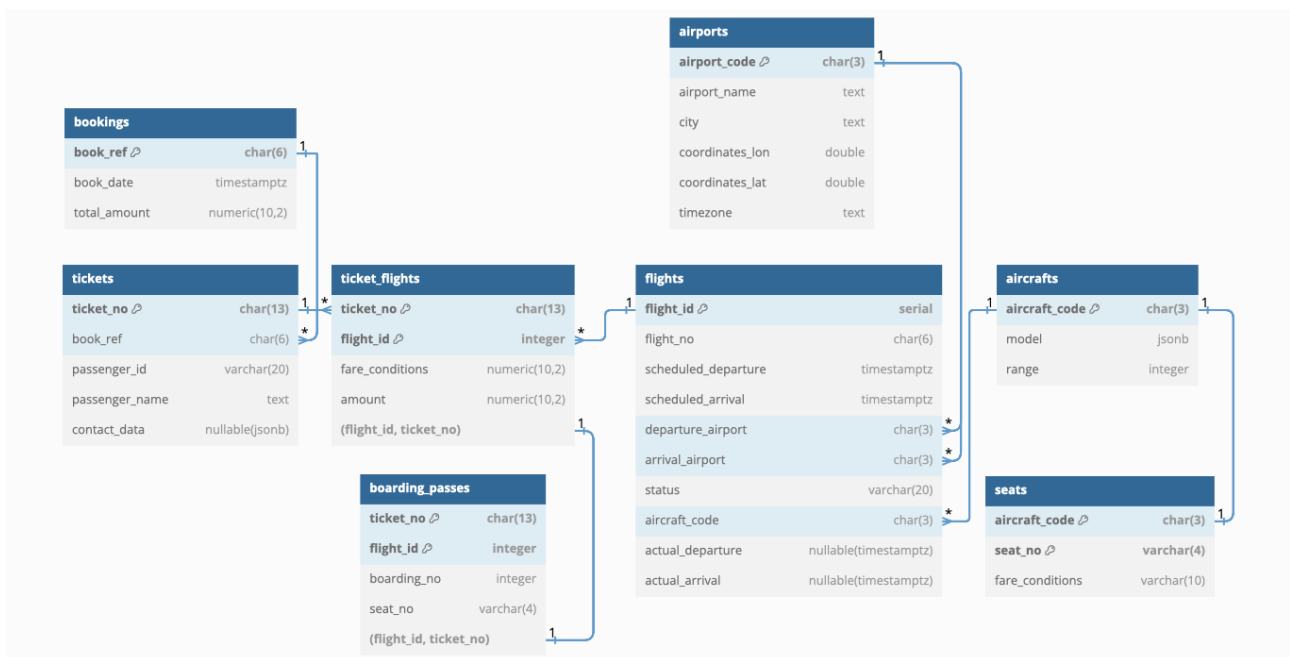
### Important notice

- 1) **Вы не сможете приступить к выполнению ДЗ №2, пока не выполните хотя бы минимум из ДЗ №1.** Поэтому если вы в ДЗ №1 не получили работающий master-хост с инициализацией структуры БД - делать задание №2 нет никакого смысла. Лучше посмотрите первые занятия, сделайте ДЗ №1 и обратитесь при необходимости за помощью/консультацией.
- 2) **Это задание выполняется и оценивается индивидуально!** Если вы используете код из открытых источников (репозитории ваших одногруппников таковыми не считаются) - пожалуйста, указывайте ссылки на них (можно в readme вести лог всех источников, откуда вы берете код). Находить готовые рецепты в интернете - хорошо, списывать - плохо.
- 3) **При обнаружении списывания (одинаковый код в двух репозиториях без указания внешнего источника) оценка будет выставляться студенту (группе), чей коммит с решением был первый.** Остальным - 0 баллов и докладная в УО.
- 4) Чтобы исключить возможность списывания, **рекомендуется сделать ваш репозиторий с домашним заданием приватным.**

### Формулировка

У нас есть поисковик дешевых авиабилетов.

БД системы, которая содержит информацию о забронированных с его помощью билетах, выглядит следующим образом:



Это мы уже с вами видели в ДЗ №1.

Сейчас мы имеем:

- \* master-хост с БД
- \* async-replica, на которую копируются данные для аналитической нагрузки

### Задача:

1. **Выбрать одну из предложенных архитектур** для реализации детального слоя DWH: **Data Vault, Data Vault 2.0 или якорная модель**. Обосновать выбор архитектуры в README
2. **Написать DDL** для вашего детального слоя DWH (если ваша реализация этого требует - об этом см. далее)  
**Составить ER-диаграмму** для детального слоя DWH (а вот этот пункт уже

обязательный; можно воспользоваться специальными сервисами для построения диаграмм, когда ваше DWH уже будет готово, или построить диаграмму по DDL из п. 1) **Поднять еще один инстанс PostgreSQL** для DWH и инициализировать его полученной выше структурой

Данные детального слоя необходимо **сложить в схему dwh\_detailed**.

**Важно:**

- помните, что системы-источники также являются сущностями в терминах DWH
- помните, что DWH по определению insert-only, у нас не должны использоваться операции update и delete
- помните про версионирование данных
- помните, что в dwh существуют конвенции нейминга для таблиц и полей, а также - технические поля (sourceSystemId, createdAt и другие)
- DataGrip и draw.io из коробки умеют строить ER-диаграммы из структуры в sql-файле

3. **Поднять и подключить debezium** к master-хосту вашего сервиса

4. **Реализовать на python (или другом ЯП) DMP-сервис**. Его задача:

1. Читать данные об изменениях в таблицах сервиса из kafka
2. Формировать данные для добавления в DWH
3. Осуществлять вставку данных в DWH

5. **Посадить DMP-сервис в контейнер** в Docker-compose

**Бонусные задания:**

1. **Использование генератора кода для создания DDL** детального слоя (не важно, opensource или самописного, главное - чтобы он принимал config-файл на вход, а на выходе выдавал sql-файл для создания таблицы). Принцип работы необходимо описать и продемонстрировать в README

2. **Использовать для DWH не PostgreSQL, а масштабируемое решение - MPP-базу или стек Iceberg+S3**. Обосновать выбор в README

**Важно:**

- помните, что не все MPP могут эффективно работать с большим кол-вом JOIN (например, ClickHouse), и это надо учитывать при выборе СУБД и/или архитектуры DWH
- помните, что для эффективной обработки данных в MPP могут потребоваться сортировка/сегментация по ключам
- помните, что S3 предоставляет вам только хранение, и для обработки данных вам также потребуется compute engine (Spark, Trino, DuckDB или другие)

3. **Использовать в DMP один универсальный класс и yaml-конфиги** вместо отдельных классов для каждой таблицы. Принцип работы необходимо описать и продемонстрировать в README

**Важно:**

- в пунктах 1 и 3 бонусного задания можно использовать dbt (подробнее о dbt - на занятии 7)
- при использовании dbt писать ddl для детального слоя не нужно - можно просто привести финальную схему

### Сроки

**Важно:** домашнее задание очень объемное! За один день его сделать вряд ли получится. Времени дается на выполнение много, приступайте к выполнению ДЗ заранее.

1. Дедлайн на 100% - 4 недели - **12.01.2025 23:59:59 включительно**
2. Дедлайн на 75% - до дедлайна следующего ДЗ
3. Дедлайн на 50% - до конца курса

### Как сдавать ДЗ?

- Готовое ДЗ загружается на GitHub (в приватный репо + выдать мне доступ - @mgcrp)
- Домашнее задание №2 можно продолжать делать в том же репозитории, что и домашнее задание №1 (можно для связанности отвести отдельную ветку)
- К репо должен быть приложен README.md с описанием того, что вы сделали и как это запустить
- Задание сдается в форму: <https://forms.gle/fKgBNPEy95JY2fLLA>

### Критерии оценки

Балл	Критерий
4	DDL для детального слоя DWH ER-диаграмма Поднят новый инстанс PostgreSQL для DWH, он инициализируются со структурой детального слоя
6	Поднят, подключен и работает debezium
10	Реализован и работает DMP, происходит наполнение детального слоя DWH
+2 балла	Использование генератора кода для создания DDL детального слоя
+4 балла	Использование для DWH не PostgreSQL, а MPP/S3 по вашему выбору
+4 балла	Реализовать DMP как один универсальный класс и уaml-конфиги вместо отдельных классов для каждой таблицы (или dbt)

Максимальный балл за ДЗ - 20

### Как это будет проверяться?

- 1) **Запуск системы по инструкции** из вашего README
- 2) **Проверка работы реплики**  
Автотест: python-скрипт, который создает синтетические данные и пишет их в master, после проверяет, что все сгенеренные данные попали в реплику
- 3) **Проверка работы debezium**  
Автотест: проверяет, что для таблиц создались топики в kafka, в них пишутся данные
- 4) **Проверка работы DMP**  
Автотест: проверяет, что данные успешно пишутся в таблицы детального слоя DWH
- 5) Ручная проверка бонусов и VIEW