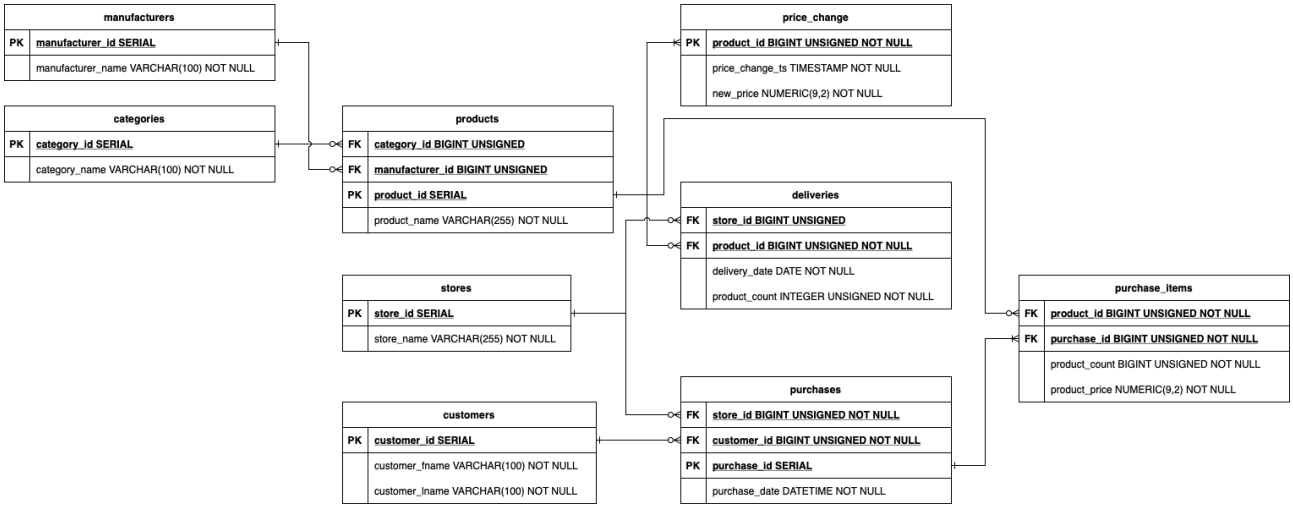
# Домашнее задание 2

**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 инициализацей структуры БД - делать задание №2 нет никакого смысла.
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, 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**.

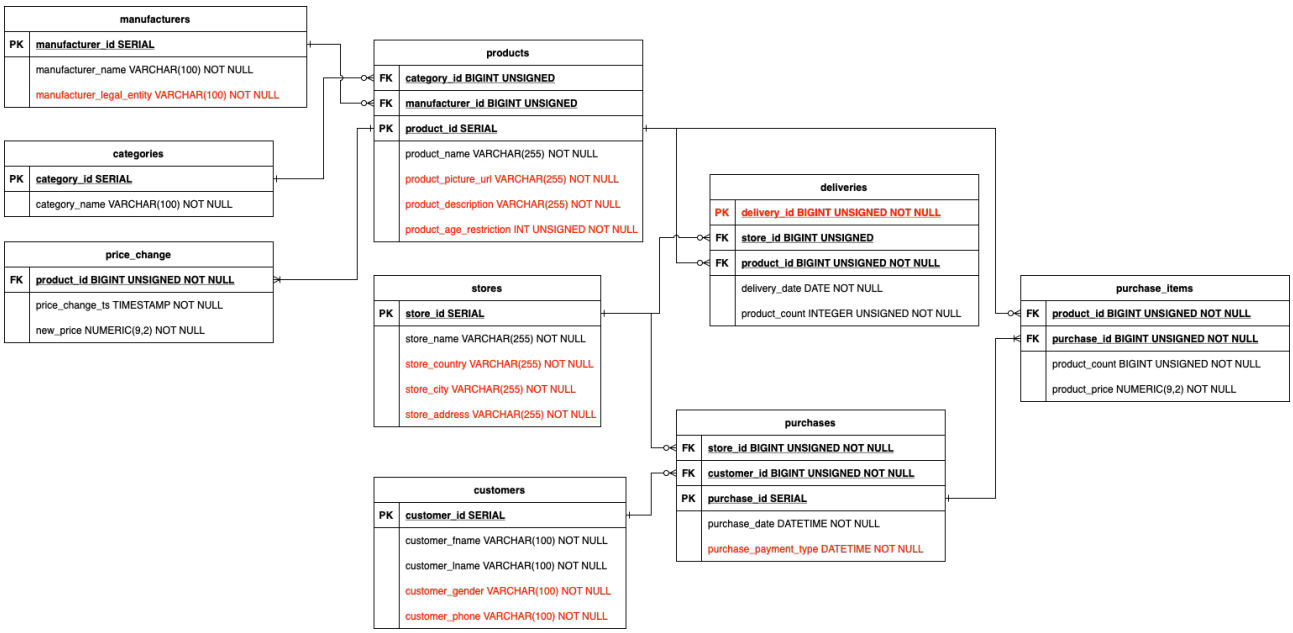
**Формулировка**

У нас есть сеть магазинов.
БД системы, которая обеспечивает её работу, 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

Для того, чтобы сделать проект интереснее, сюда были добавлены новые атрибуты.
Обновленная схема выглядит следующим образом (новые поля выделены красным):




**Задача**:

1. **Добавить новые поля** в существующий DDL базы сервиса
2. **Выбрать одну из предложенных архитектур** для реализации детального слоя DWH: **Data Vault, Data Vault 2.0 или якорная модель**. Обосновать выбор архитектуры в README
3. **Написать DDL** для вашего детального слоя DWH
   **Составить ER-диаграмму** для детального слоя DWH
   **Поднять еще один инстанс PostgreSQL** для DWH и инициализировать его полученной выше структурой
   Данные детального слоя необходимо **сложить в схему dwh_detailed.**
   **Важно**:
   - не забывайте, что системы-источники также являются сущностями
   - не забывайте про то, что DWH по определению insert-only, у нас не должны использоваться операции update и delete
   - не забывайте про версионирование данных
   - не забывайте, что в dwh существуют конвенции нейминга для таблиц и полей
   - DataGrip и draw.io из коробки умеют строить ER-диаграммы из структуры в sql-файле
4. **Поднять и подключить debezium** к master-хосту вашего сервиса
5. **Написать на python 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
6. **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 MPP базу по вашему выбору (Clickhouse, Greenplum, Vertica). Обосновать выбор базы в README
3. Использовать в DMP один универсальный класс и yaml-конфиги вместо отдельных классов для каждой таблицы. Принцип работы необходимо описать и продемонстрировать в README

## 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 - **19.11.2023 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 (приватный репозиторий, для проверки предоставить доступ @mgcrp)
- Домашнее задание №2 можно продолжать делать в том же репозитории, что и домашнее задание №1
- К репозиторию должен быть приложен README с описанием того, что вы сделали и как это запустить
- Задание сдается в форму: https://forms.gle/eMoxrHp3Ujy8muY87

**Критерии оценки**

| **Балл** | **Критерий** |
|---|---|
| 2 | В структуру данных из ДЗ №1 добавлены новые поля |
| 4 | DDL для детального слоя DWH<br>ER-диаграмма<br>Поднят новый инстанс PostgreSQL для DWH |
| 6 | Поднят и подключен 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 базы по выбору |
| +4 балла | Реализовать DMP как один универсальный класс и yaml-конфиги вместо отдельных классов для каждой таблицы |

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