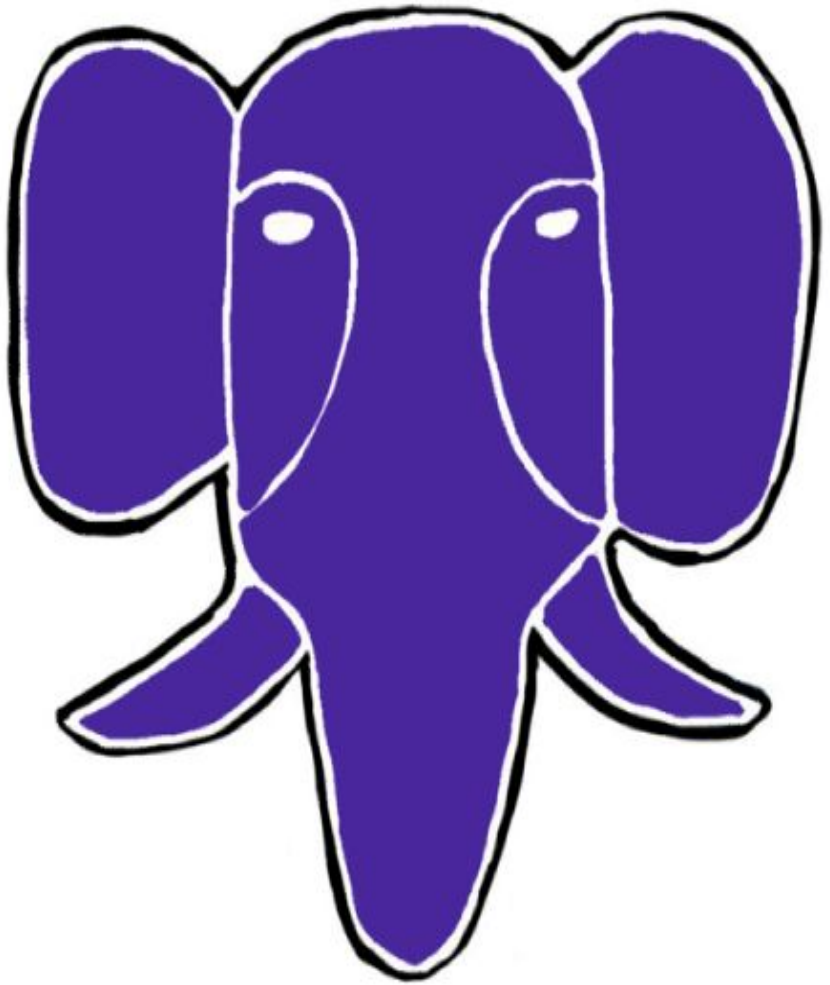
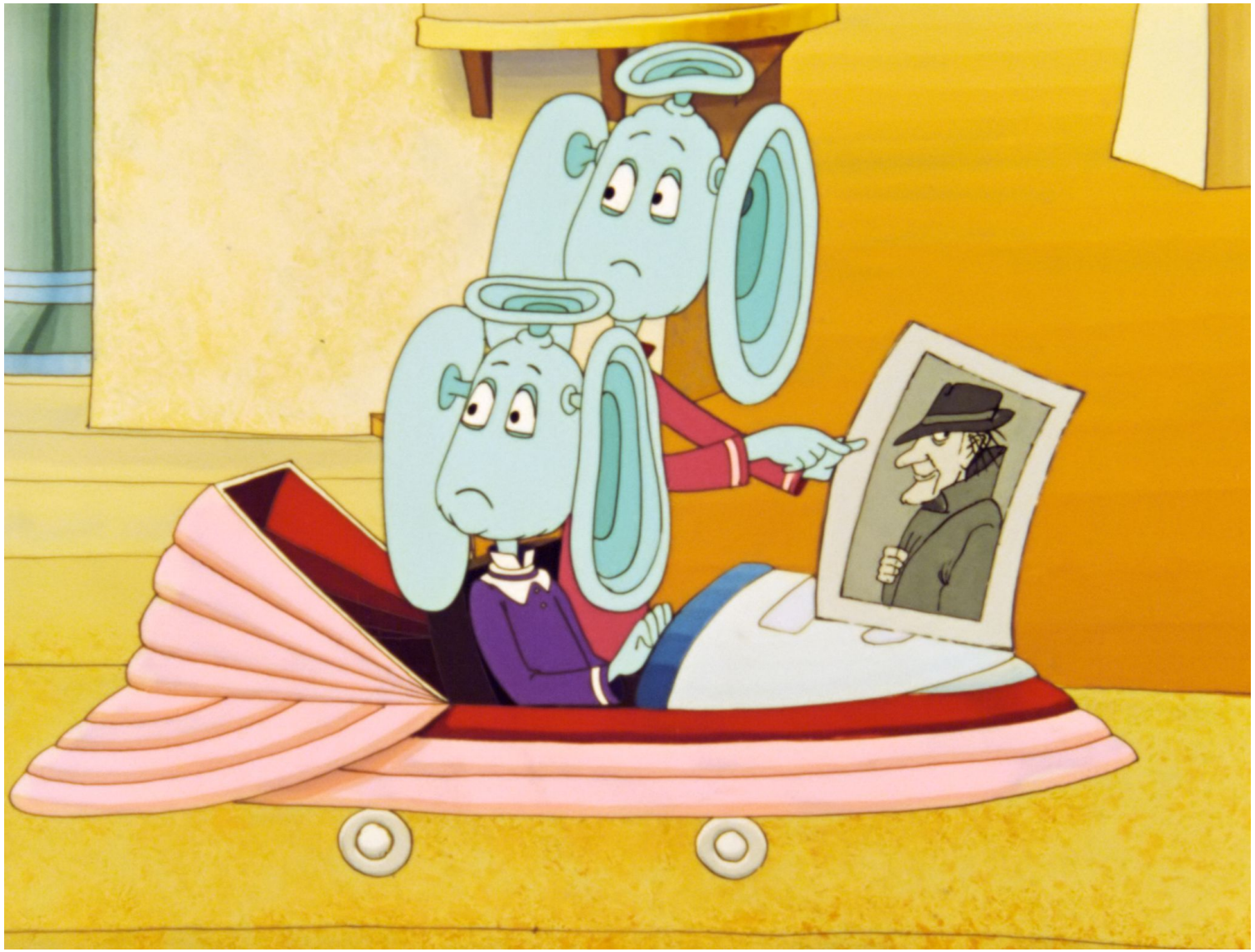
Основы проектирования - 3 нормальная форма

Евгений Аристов



https://aristov.tech



Правила вебинара

Задаем вопрос в чат

Вопросы вижу, отвечу в момент логической паузы

Если есть вопрос голосом - поставьте знак ? в чат

Если остались вопросы, можно их задать на следующем занятии

Маршрут вебинара

1. Нормальные формы
2. Уровни проектирования
3. Инструменты проектирования

https://aristov.tech

Реляционная модель

Вспоминаем слайд:

- минимизация логической избыточности
- сокращает объем хранимых данных
- увеличивает производительность - **спорный аргумент**
- **нужно джойнить** эти таблицы - это **не бесплатно**
- **в разумных пределах** полезна для транзакционных БД
- **вредна** для аналитических БД

Нормализация отношений. Шесть нормальных форм / Хабр

Нормальная форма — Википедия

Уровни проектирования

- Концептуальная модель
- Логическая модель
- Физическая модель

Концептуальная модель:

- Определяемся с заказчиком по области хранения данных - что храним, какие запросы и тд
- Определение сущностей и их **документирование (по полю comment)**
- Создание ER-модели - Определение связей между сущностями
- Определение атрибутов у каждой сущности
- Определение потенциальных ключей (набор записей, делающих запись уникальной), внешних ключей (ссылки на первичный ключ родительской таблицы) - *рассмотрим на следующей лекции*

ER модель:

Сущности:

Сущность (entity) – это реальный или представляемый тип объекта, информация о котором должна сохраняться и быть доступна.

Связь (relationship) – это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Связь может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь). Возможны связи на основе отношений:

- один-к-одному;
- один-ко-многим;
- многие-ко-многим.

Связь может быть задана ограничением в виде **FOREIGN KEY**

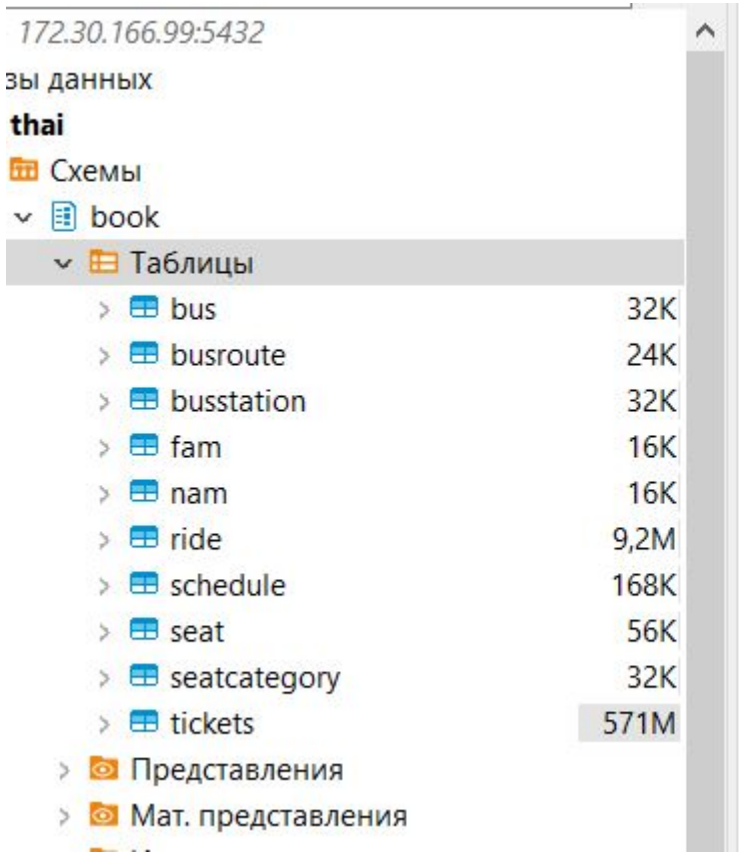
Логическая модель:

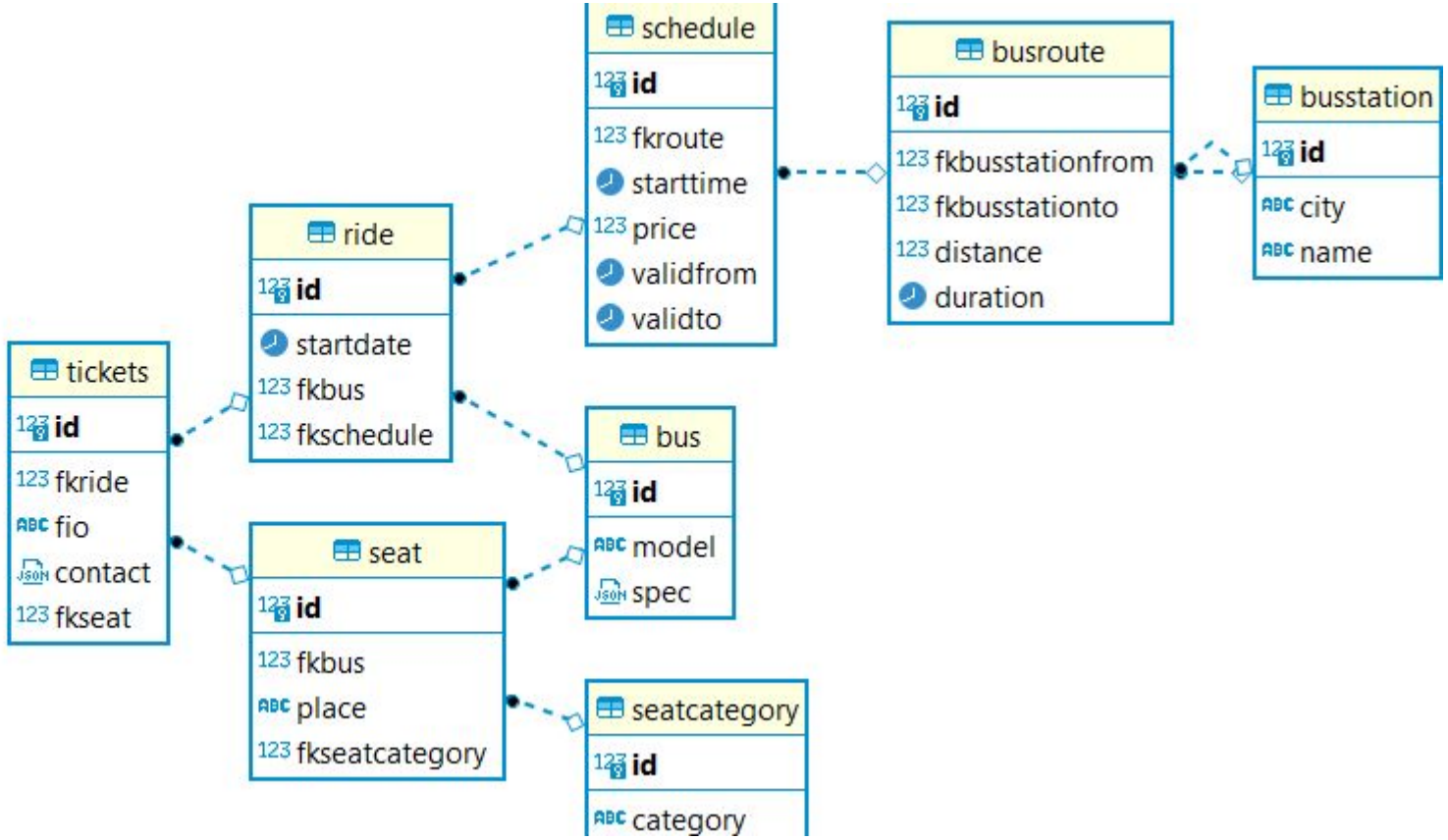
- Выбор модели данных (SQL, NoSQL, newSQL)
- Определение набор таблиц
- Декомпозиция
- Нормализация данных
- Денормализация данных (избыточность, например **materialized view**)
- Определения набора транзакций и соответствие структуры данных
- Определение требований поддержки целостности
- Создание окончательной логической модели и обсуждение ее с командой и заказчиками

Физическая модель

- Проектирование таблиц данных средствами **выбранной** СУБД
- Даже в рамках одной СУБД есть разница в движках (не во всех субд), видах индексов и т.д.
- Реализация бизнес правил в выбранной СУБД
- Хранимые процедуры как на SQL, так и на других языках в конкретной СУБД
- Определение транзакционной модели
- Проектирование физической организации данных
- Настраиваем высокую доступность и схему бэкапирвания
- Планирование ресурсов (pool connect, proxy)
- Определение правил безопасности и защиты информации
- Организация сопровождения, мониторинга и алертинга!

Пример модели

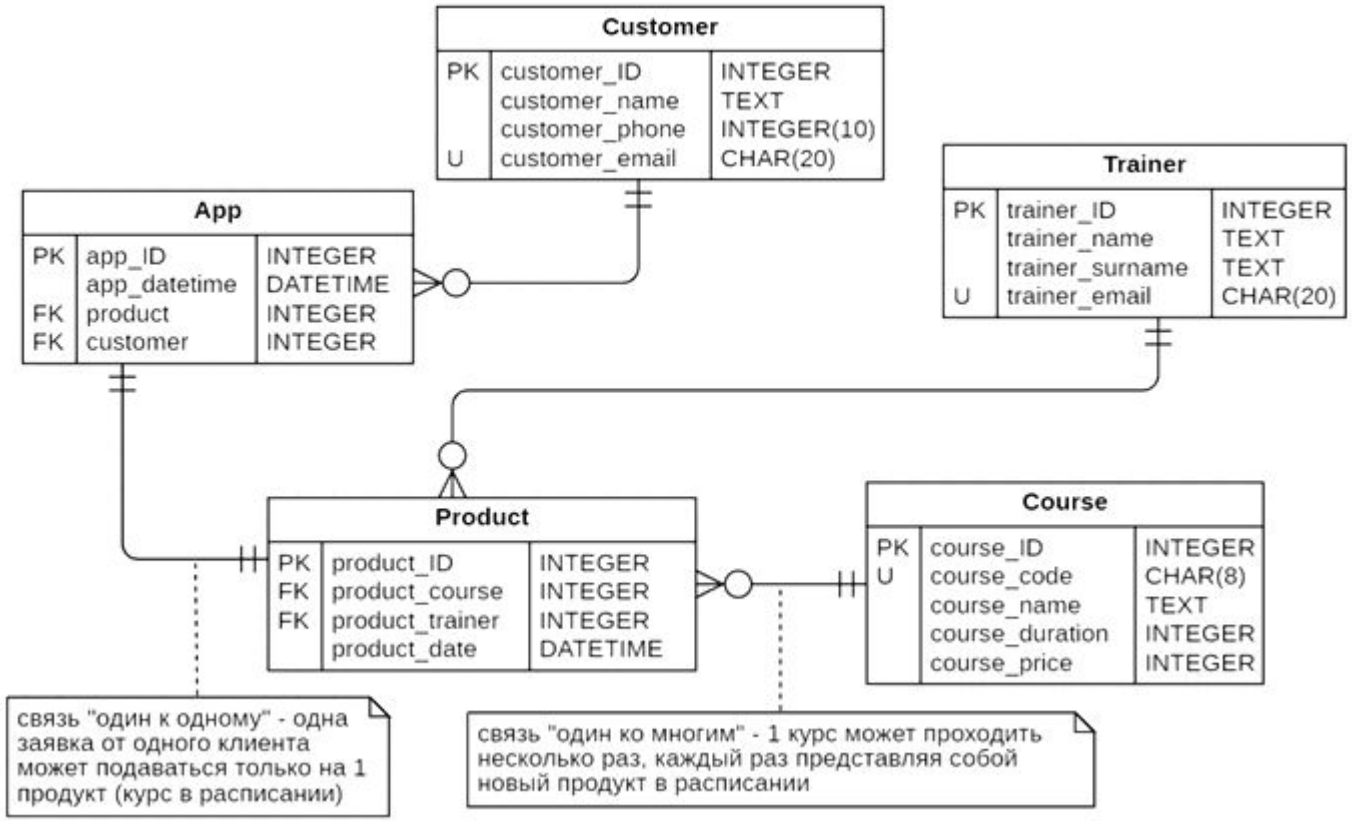




https://github.com/aeuge/postgres16book/tree/main/database

Пример модели

Пример модели



https://babok-school.ru/blog/erd-and-sql-for-junior-analyst-practical-example/

https://aristov.tech

Где можно проектировать:

- непосредственно в psql
- Dbeaver
- https://app.diagrams.net/
- https://www.lucidchart.com/pages/ru/examples/er-diagram-tool
- Microsoft Access
- https://www.db-fiddle.com/
- куча бесплатных онлайн инструментов

Основные правила

у каждой компании свои стандарты, но в общем:

- имена объектов только на английском без пробелов
- 2 классических вида
  - snake_case
  - camelCase
  - все равно к строчному виду автоматически приводятся
  - можно в кавычках использовать разный регистр и русский
- пишите документацию

Итоги

https://aristov.tech

Итоги

Остались ли вопросы?

Увидимся на следующем занятии

https://aristov.tech

Спасибо за внимание!

Когда дальше и куда?
скину в чате
материалы для бесплатного доступа будут появляться на ютубе

Аристов Евгений