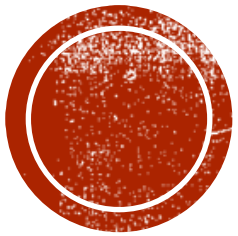
ЯЗЫКИ ПРИКЛАДНОГО ПРОГРАММИРОВАНИЯ

Лекция 7



РЕЛЯЦИОННЫЕ БД

Чуть подробнее…

РЕЛЯЦИОННАЯ МОДЕЛЬ

- Данные организованы в виде набора таблиц, называемых отношениями, состоящих из столбцов и строк.
- В таблицах хранится информация об объектах, представленных в БД.
- Каждая строка таблицы представляет собой набор связанных значений, относящихся к одному объекту, или сущности.
- Каждая строка в таблице может быть помечена уникальным идентификатором, называемым первичным ключом,
- Строки из нескольких таблиц могут быть связаны с помощью внешних ключей.

- **ACID** - набор требований к системе, обеспечивающий наиболее надёжную и предсказуемую её работу:
  - атомарность,
  - согласованность,
  - изоляция,
  - надежность;
- Требования сформулированы в конце 1970-х годов.
- Реляционные СУБД соответствуют требованиям **ACID**, не реляционные, зачастую – нет.
- **https://habr.com/ru/post/555920/**

АТОМАРНОСТЬ

- Транзакция – это одно или несколько действий, выполненных в виде последовательности операций, представляющих собой единую логическую задачу.
- **Атомарность** – это условие, при котором либо транзакция успешно выполняется целиком, либо, если какая-либо из ее частей не выполняется, вся транзакция отменяется.
- Т.е. транзакция – неделимое (атомарное) действие.

СОГЛАСОВАННОСТЬ

- **Согласованность** – это условие, при котором данные, записываемые в базу данных в рамках транзакции, должны соответствовать всем правилам и ограничениям, включая ограничения целостности, каскады и триггеры.
- Поддерживается за счет:
  - Первичных ключей,
  - Внешних ключей,
  - Ограничений на значения столбцов
- Ограничения позволяют применять правила предметной области к данным в таблицах и гарантировать точность и надежность данных.
- Большинство ядер БД также поддерживает выполнение SQL-скрипта, который выполняется в ответ на определенные операции в БД (триггеры).

ПЕРВИЧНЫЙ И ВНЕШНИЙ КЛЮЧИ

- Каждая сущность (строка) имеет уникальный идентификатор, называемый **первичным ключом.**
- Строка одной таблицы может быть связана с данными из другой при помощи **внешнего ключа.**

ИЗОЛЯЦИЯ

- **Изоляция** необходима для контроля над согласованностью и гарантирует независимость транзакций друг от друга.

НАДЕЖНОСТЬ

- **Надежность** подразумевает, что все внесенные в базу данных изменения на момент успешного завершения транзакции считаются постоянными.

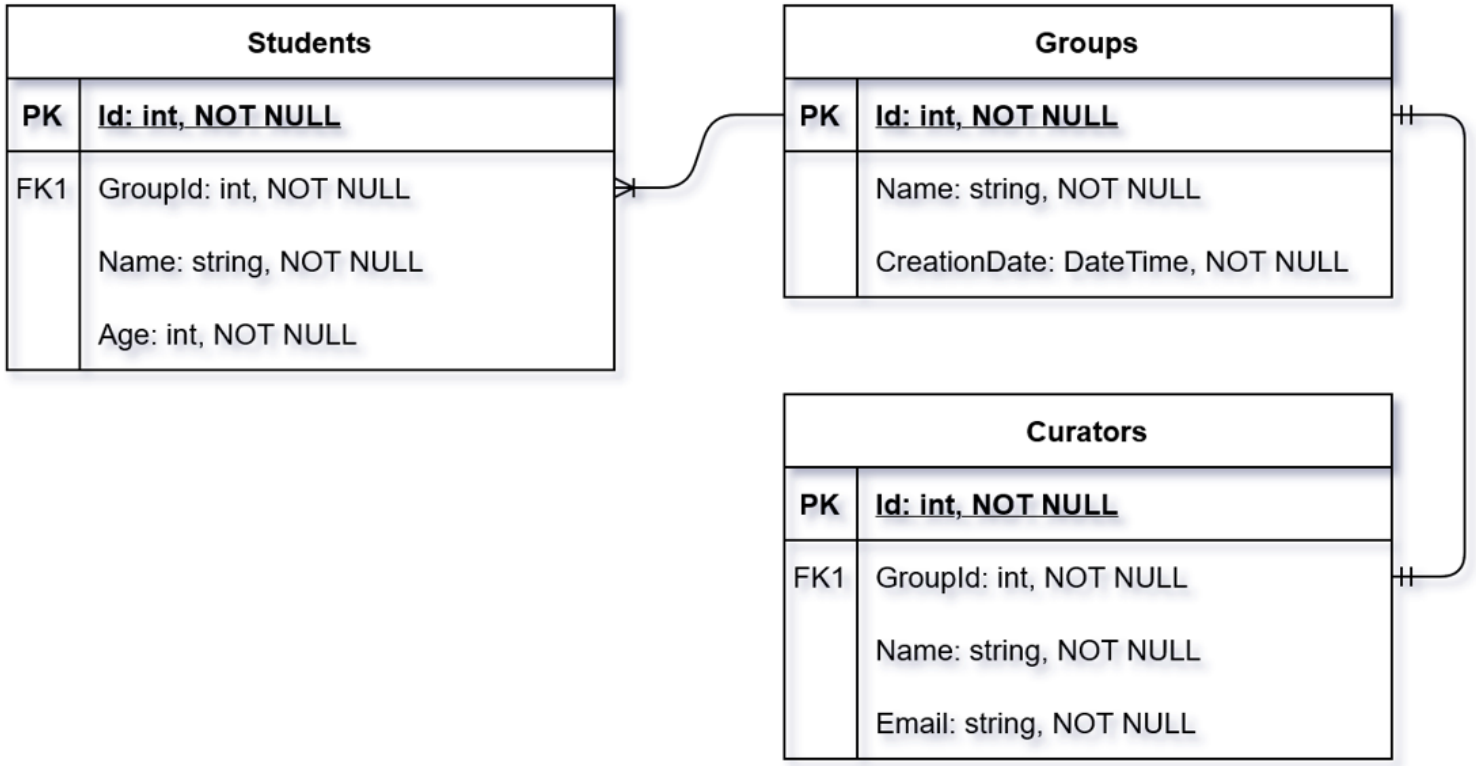
ТИПЫ ДАННЫХ

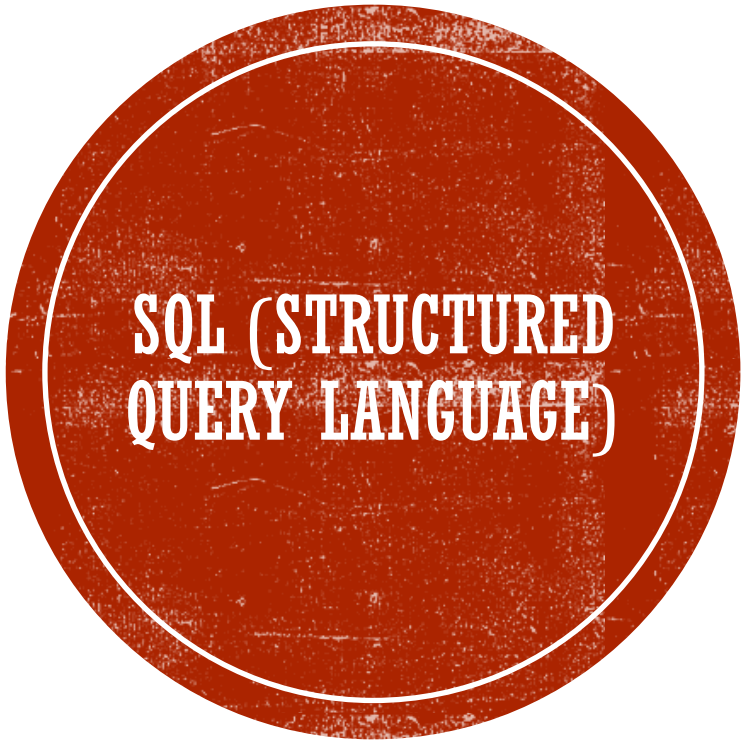
Совместимость: В стандарте SQL описаны следующие типы (или их имена): `bigint`, `bit`, `bit varying`, `boolean`, `char`, `character varying`, `character`, `varchar`, `date`, `double precision`, `integer`, `interval`, `numeric`, `decimal`, `real`, `smallint`, `time` (с часовым поясом и без), `timestamp` (с часовым поясом и без), `xml`.

ОПИСАНИЕ БАЗЫ ДАННЫХ

- Схема базы данных (Database schema) — её структура, описанная на формальном языке, поддерживаемом СУБД. В реляционных базах данных схема определяет таблицы, поля в каждой таблице (обычно с указанием их названия, типа, обязательности), и ограничения целостности (первичный, потенциальные и внешние ключи и другие ограничения).
- ER-диаграмма (Entity-Relationship. Сущность-Связь) — диаграмма для отображения связи между сущностями в предметной области. Чаще всего записывается в нотации Crow’s foot.

ОПИСАНИЕ БАЗЫ ДАННЫХ





- SQL – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

SQL ЗАПРОСЫ

- Есть четыре основных типа запросов данных в SQL, которые относятся к так называемому языку манипулирования данными (Data Manipulation Language или DML):
- **SELECT – выбрать строки из таблиц;**
- **INSERT – добавить строки в таблицу;**
- **UPDATE – изменить строки в таблице;**
- **DELETE – удалить строки в таблице;**

SELECT

```
SELECT ('столбцы или * для выбора всех столбцов; обязательно')
FROM ('таблица; обязательно')
WHERE ('условие/фильтрация, например, city = 'Moscow'; необязательно')
GROUP BY ('столбец, по которому хотим сгруппировать данные; необязательно')
HAVING ('условие/фильтрация на уровне сгруппированных данных; необязательно')
ORDER BY ('столбец, по которому хотим отсортировать вывод; необязательно')
```

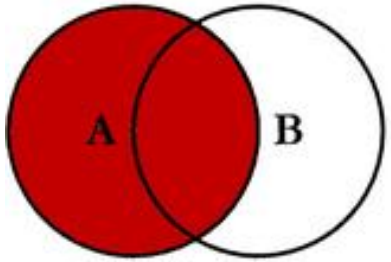
```
select City, count(CustomerID) from Customers
WHERE Country = 'Germany'
GROUP BY City
```

JOIN

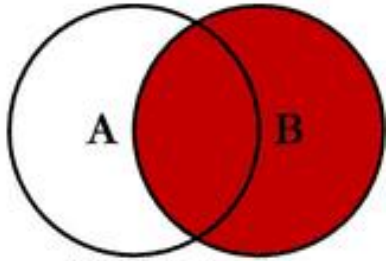
- JOIN — необязательный элемент, используется для объединения таблиц по ключу, который присутствует в обеих таблицах. Перед ключом ставится оператор ON.

```
select * from Orders
join Customers on Orders.CustomerID = Customers.CustomerID
where Customers.CustomerID >10
```

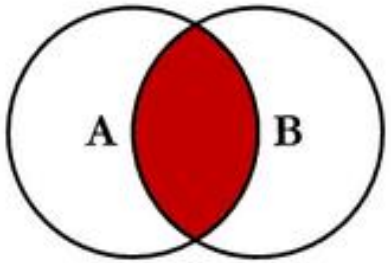
SQL JOINS



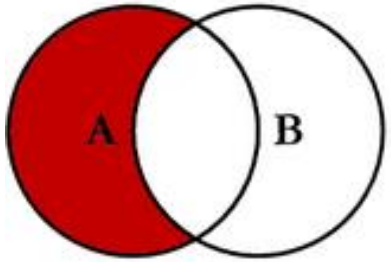

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



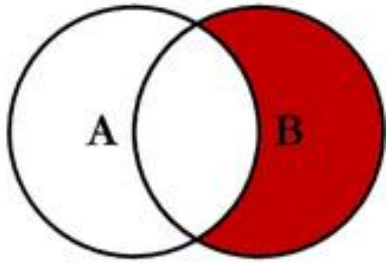

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



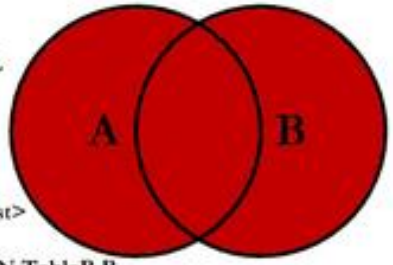

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



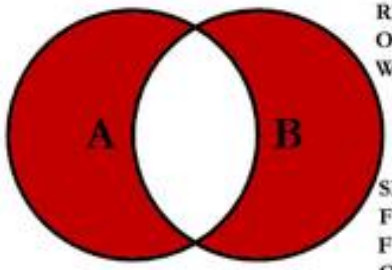

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

- Это план перехода базы данных от старой схемы к новой.

ORM (OBJECT-RELATIONAL MAPPING)

- технология, суть которой заключается в создании «виртуальной объектной базы данных».
- ORM берет на себя взаимодействие с БД выступая в качестве слоя абстракции, выполняя оптимизации, упрощая программисту работу с БД.

ПОДХОДЫ К РАБОТЕ ORM

- **Database first** – подключаемся к существующей БД, по базе данных создаем классы
- **Model first** – описывается модель данных, на основании которой создается БД и классы (EDMX)
- **Code first** – создание БД на основании описанной в коде модели данных.

ПРЕИМУЩЕСТВА

- Модель описана в одном месте. Это упрощает поддержку и повторное использование кода.
- Разработчику не нужно писать **SQL**, отсутствует
- Дополнительный слой абстракции (зачастую только в рамках реляционных баз) – позволяющий с легкостью изменять СУБД не изменяя код

НЕДОСТАТКИ

- Медленнее чем хорошо написанный **SQL**
- В крупных системах необходимо изучить нюансы **ORM** фреймворка который используете

- Entity Framework (EF) Core — кроссплатформенная и расширяемая технология доступа к данным с открытым исходным кодом для **.NET**
- Более новая версия **Entity Framework**, разработанного для **.NET Framework**

МОДЕЛЬ В EF CORE

- Модель состоит из классов сущностей и объекта контекста, который представляет сеанс взаимодействия с базой данных. Объект контекста позволяет выполнять запросы и сохранять данные.

- Способы создания моделей:
  - **Data annotation** – настройка сопоставления моделей и таблиц с помощью атрибутов.
  - **Fluent API** – настойка сопоставления моделей и таблиц с помощью набора методов специализированного **API**.

- Почитать можно:
  - https://learn.microsoft.com/ru-ru/ef/core/
  - https://metanit.com/sharp/entityframeworkcore/