4. SQL

Structured Query Language

к.т.н., доцент кафедры ИиСП

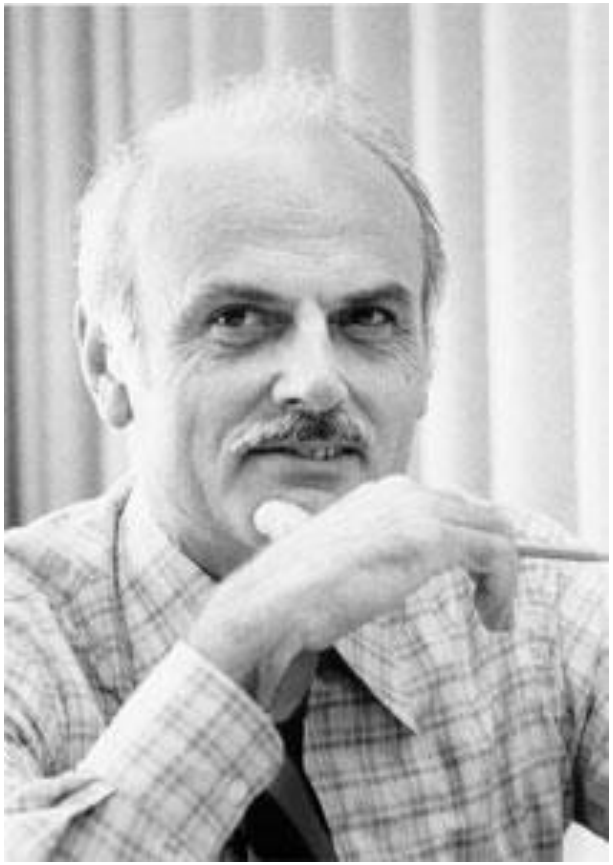
Лучинин
Захар Сергеевич

Операторы SQL

- DDL (Data Definition Language)
- **DML (Data Manipulation Language)**
- DCL (Data Control Language)
- TCL (Transaction Control Language)

Реляционная алгебра. Операции над множествами

- Выборка
- Проекция
- Объединение
- Пересечение
- Разность
- Произведение
- Деление
- Соединение



Выборка

Персоны

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80

$\sigma_{\text{Возраст} \geq 34}(\text{Персоны})$

Имя	Возраст	Вес
Harry	34	80
Helena	54	54
Peter	34	80

Проекция

Персоны

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80

$\pi_{\text{Возраст,Вес}}(\text{Персоны})$

Возраст	Вес
29	70
54	54
34	80

Объединение

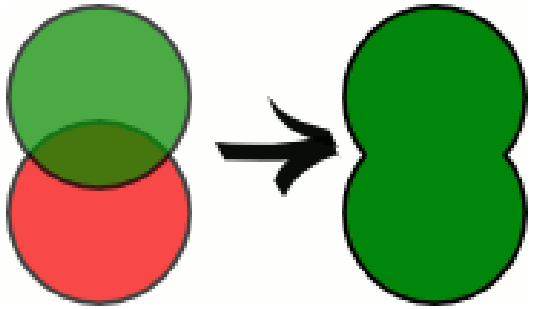
Персоны

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80

Персонажи

Имя	Возраст	Вес
Daffy	24	19
Donald	29	70
Scrooge	81	27

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80
Daffy	24	19
Scrooge	81	27



Пересечение

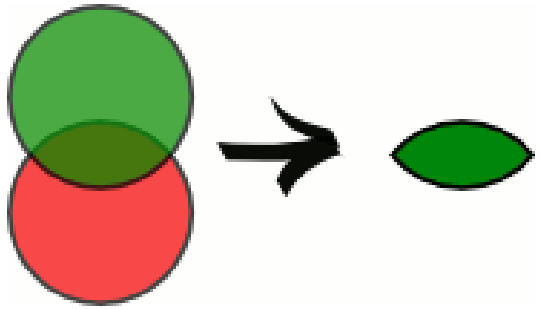
Персоны

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80

Персонажи

Имя	Возраст	Вес
Daffy	24	19
Donald	29	70
Scrooge	81	27

Имя	Возраст	Вес
Donald	29	70



Разность

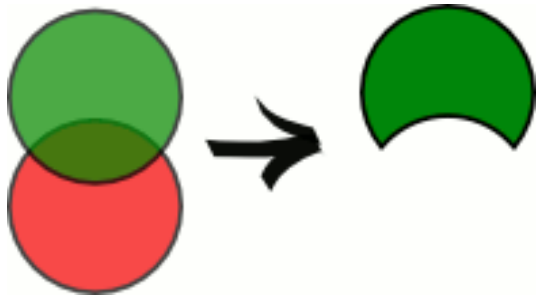
Персоны

Имя	Возраст	Вес
Harry	34	80
Donald	29	70
Helena	54	54
Peter	34	80

Персонажи

Имя	Возраст	Вес
Daffy	24	19
Donald	29	70
Scrooge	81	27

Имя	Возраст	Вес
Harry	34	80
Helena	54	54
Peter	34	80



Произведение

Мультфильмы

Код_мульта	Название_мульта
0	The Simpsons
1	Family Guy
2	Duck Tales

Каналы

Код_канала	Название_канала
0	СТС
1	2x2

Код_мульта	Название_мульта	Код_канала	Название_канала
0	The Simpsons	0	СТС
0	The Simpsons	1	2x2
1	Family Guy	0	СТС
1	Family Guy	1	2x2
2	Duck Tales	0	СТС
2	Duck Tales	1	2x2

Соединение

Мультфильмы

Код_мульта	Название_мульта	Название_канала
0	The Simpsons	2x2
1	Family Guy	2x2
2	Duck Tales	RenTV

Каналы

Код_канала	Частота
RenTV	3,1415
2x2	783,25

Код_мульта	Название_мульта	Название_канала	Код_канала	Частота
0	The Simpsons	2x2	2x2	783,25
1	Family Guy	2x2	2x2	783,25
2	Duck Tales	RenTV	RenTV	3,1415

DML - data manipulation language (язык манипулирования данными)

Включает команды INSERT, DELETE, UPDATE, SELECT...

INSERT - команда добавления записей в таблицу;

UPDATE - команда обновления записей таблицы;

DELETE - команда удаления записей таблицы;

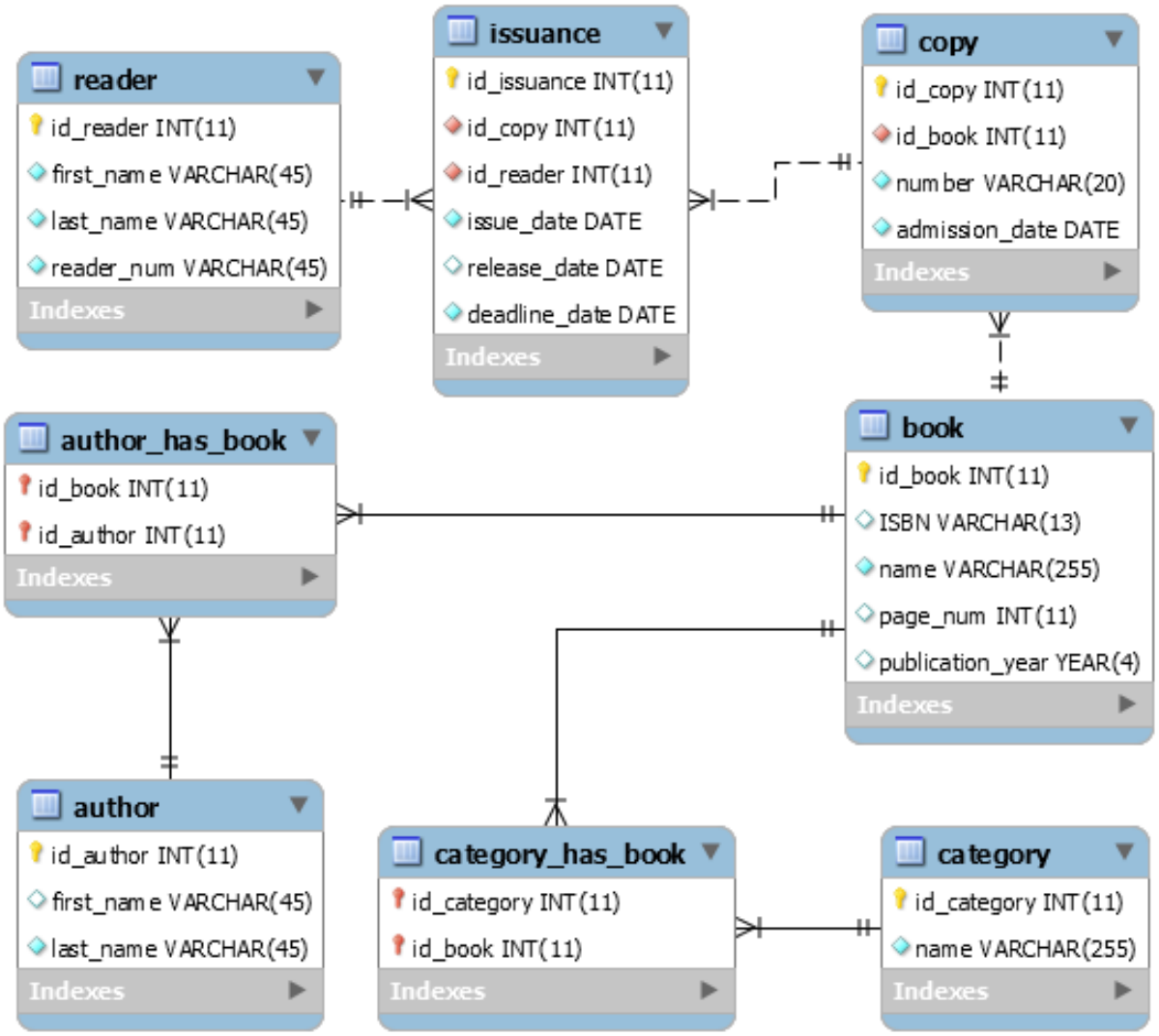
SELECT - команда выборки записей таблицы;

CRUD

CRUD (сокр. от англ. create, read, update, delete — «создать, прочесть, обновить, удалить») — акроним, обозначающий четыре базовые функции, используемые при работе с персистентными хранилищами данных.

Operation	SQL	HTTP
Create	INSERT	PUT / POST
Read (Retrieve)	SELECT	GET
Update (Modify)	UPDATE	PUT / POST / PATCH
Delete (Destroy)	DELETE	DELETE

Предметная область “Библиотека”



Оператор INSERT

Оператор **INSERT** позволяет вносить изменения в структуру таблиц: добавлять записи (строки) и заполнять их значениями.

Оператор INSERT имеет следующий синтаксис:

INSERT INTO table_name **([**column_name**, ...]) VALUES (**expressions**, ...)**

Так же, значения могут быть записаны и без указания столбцов:

INSERT INTO table_name **VALUES (**expressions**, ...)**

Запись значений может производиться и с помощью оператора SELECT:

INSERT INTO table_name **SELECT** column_name**,... FROM** table_name

Используя оператор SQL SELECT можно вставить более одной записи. В случае, если для каких-либо полей таблицы не указаны значения, они будут заменены значением по умолчанию null.

Оператор SELECT

- Оператор **SELECT** является одним из основных операторов языка SQL. Именно с его помощью происходит выборка значений, хранящихся в базе данных.
- В структуру запроса оператора **SELECT** могут быть включены многие дополнительные операторы: уточняющие условие выборки, производящие группировку, сортировку выходных значений и т.д.

- Оператор SQL SELECT имеет следующий синтаксис:

SELECT column_list
FROM table_name
[WHERE condition **GROUP BY** expression **HAVING** condition **ORDER BY** expression**]**

Необязательные операторы обрамлены квадратными скобками []

Предикат WHERE

```
SELECT что_выбрать
        FROM из_какой_таблицы
        WHERE какое_условие_должно_выполняться;

SELECT name, salary FROM employee
        WHERE name = 'Alex';

SELECT name, salary FROM employee
        WHERE salary > 100000;
```

Фильтрация

Условные операторы, поддерживаемые в **SQL**

1. **=**
2. **!=, <>**
3. **>, <, >=, <=**
4. **BETWEEN**
5. **IN**
6. **LIKE (_, %)**

Логические операторы, поддерживаемые в **SQL**

1. **OR**
2. **AND**
3. **IS**
4. **NOT**
5. **NULL**

Примеры фильтрация.
Выражения BETWEEN, LIKE, IN, IS NULL

```
SELECT * FROM tbl_name
        WHERE auto_col IS NULL;

SELECT * FROM table WHERE id IN (1,2,3,4);

SELECT * FROM Products
        WHERE Price BETWEEN 10 AND 20;

SELECT * FROM Products
        WHERE Price NOT BETWEEN 10 AND 20;

SELECT aut_name, country FROM author
        WHERE aut_name LIKE 'W%';
```

Предикат LIMIT / TOP

Оператор **LIMIT** позволяет вывести указанное число строк из таблицы. Оператор **LIMIT** записывается всегда в конце запроса.

Используется в СУБД MySQL. Аналогом в MS SQL SERVER является оператор SQL TOP.

```
SELECT * FROM book;
```

MySQL

```
SELECT * FROM book LIMIT 5;   -- Вернет строки 1-5

SELECT * FROM book LIMIT 5,10;   -- Вернет строки 6-15
```

MSSQL

```
SELECT TOP 5 * FROM book;   -- Вернет строки 1-5
```

Предикат ORDER

Оператор **ORDER BY** выполняет сортировку выходных значений. Оператор **ORDER BY** можно применять как к числовым столбцам, так и к строковым. В последнем случае, сортировка будет происходить по алфавиту.

ASC - сортирует по возрастанию

DESC - сортирует по убыванию

```
SELECT CONCAT(last_name,', ',first_name) AS full_name
        FROM mytable ORDER BY full_name;
SELECT college, region, seed FROM tournament
        ORDER BY region, seed;
SELECT college, region AS r, seed AS s FROM tournament
        ORDER BY r, s;
SELECT college, region, seed FROM tournament
        ORDER BY 2, 3;
```

Пример сортировки ORDER BY

```
mysql> SELECT name, birth
FROM pet ORDER BY birth;
+----------+------------+
| name     | birth      |
+----------+------------+
| Buffy    | 1989-05-13 |
| Bowser   | 1989-08-31 |
| Fang     | 1990-08-27 |
| Fluffy   | 1993-02-04 |
| Claws    | 1994-03-17 |
| Slim     | 1996-04-29 |
| Whistler | 1997-12-09 |
| Chirpy   | 1998-09-11 |
| Puffball | 1999-03-30 |
+----------+------------+
```

```
mysql> SELECT name, species, birth
FROM pet ORDER BY species, birth DESC;
+----------+---------+------------+
| name     | species | birth      |
+----------+---------+------------+
| Chirpy   | bird    | 1998-09-11 |
| Whistler | bird    | 1997-12-09 |
| Claws    | cat     | 1994-03-17 |
| Fluffy   | cat     | 1993-02-04 |
| Fang     | dog     | 1990-08-27 |
| Bowser   | dog     | 1989-08-31 |
| Buffy    | dog     | 1989-05-13 |
| Puffball | hamster | 1999-03-30 |
| Slim     | snake   | 1996-04-29 |
+----------+---------+------------+
```

Оператор UPDATE

Оператор **UPDATE** используется для изменения значений в записях таблицы.

Оператор **UPDATE** имеет следующий синтаксис:

UPDATE table_name **SET** expression **[WHERE** condition**]**

Оператор DELETE

Оператор **DELETE** используется для удаления записей таблиц.

Если предложение **WHERE** отсутствует, удаляются все строки из таблицы

DELETE FROM <имя таблицы > **[WHERE** <предикат>**];**

Оператор **TRUNCATE** используется для очистки таблицы

TRUNCATE TABLE <имя таблицы>

Оператор GROUP BY

Оператор **GROUP BY** используется для объединения результатов выборки по одному или нескольким столбцам.

Оператор **GROUP BY** имеет следующий синтаксис:

```
GROUP BY column_name, [column2_name]
```

С использованием оператора **GROUP BY** тесно связано использование агрегатных функций и оператор **HAVING**

```
SELECT singer, SUM(sale) AS all_sales
   FROM artist GROUP BY singer
```

Агрегатные функции

- Агрегатная функция выполняет вычисление на наборе значений и возвращает одиночное значение.
- Агрегатные функции, за исключением COUNT, не учитывают значения NULL.

Name	Description
AVG()	Возвращает среднее арифметическое группы значений
COUNT(DISTINCT)	Возвращает количество уникальных элементов, найденных в группе
COUNT()	Возвращает количество элементов, найденных в группе
MAX()	Возвращает максимальное значение группы
MIN()	Возвращает минимальное значение группы
SUM()	Возвращает сумму группы значений

https://docs.microsoft.com/ru-ru/sql/t-sql/functions/aggregate-functions-transact-sql?view=sql-server-ver15

Оператор GROUP BY

singer	album	year	sale
The Prodigy	Invaders Must Die	2008	1200000
Drowning Pool	Sinner	2001	400000
Massive Attack	Mezzanine	1998	2300000
The Prodigy	Fat of the Land	1997	600000
The Prodigy	Music For The Jilted Generation	1994	1500000
Massive Attack	100th Window	2003	1200000
Drowning Pool	Full Circle	2007	800000
Massive Attack	Danny The Dog	2004	1900000
Drowning Pool	Resilience	2013	500000

Используя оператор GROUP BY найти сумму продаж альбомов (sale) всех исполнителей (singer)

```
SELECT singer, SUM(sale) AS all_sales
  FROM artist GROUP BY singer
```

singer	all_sales
Drowning Pool	1700000
Massive Attack	5400000
The Prodigy	3300000

Оператор GROUP BY

singer	album	year	sale
The Prodigy	Invaders Must Die	2008	1200000
Drowning Pool	Sinner	2001	400000
Massive Attack	Mezzanine	1998	2300000
The Prodigy	Fat of the Land	1997	600000
The Prodigy	Music For The Jilted Generation	1994	1500000
Massive Attack	100th Window	2003	1200000
Drowning Pool	Full Circle	2007	800000
Massive Attack	Danny The Dog	2004	1900000
Drowning Pool	Resilience	2013	500000

Узнать в каком году был выпущен последний альбом каждой из групп используя оператор **GROUP BY**

```
SELECT singer, MAX(year) AS last_album_year
  FROM artist GROUP BY singer
```

singer	last_album_year
Drowning Pool	2013
Massive Attack	2004
The Prodigy	2008

Оператор HAVING

Оператор **HAVING** аналогичен оператору **WHERE** за тем исключением, что применяется не для всего набора столбцов таблицы, а для набора созданного оператором **GROUP BY** и применяется всегда строго после него.