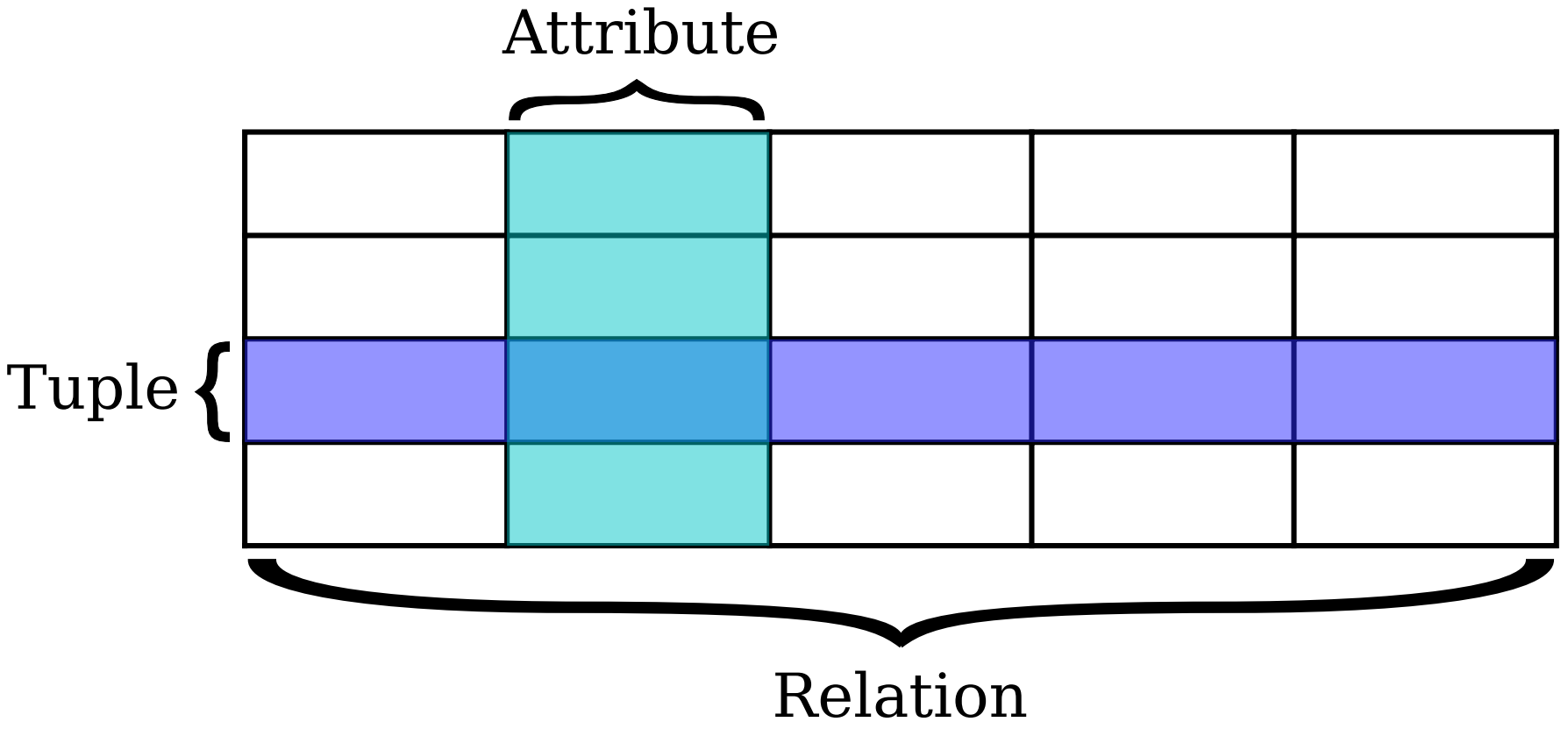
RDBMS

Ба́за да́нных — представленная в объективной форме совокупность самостоятельных материалов (статей, расчётов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ).

Систе́ма управле́ния ба́зами да́нных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Реляционная база данных — БД, основанная на реляционной модели представления данных. Реляционная модель определает объекты (таблицы, колонки, строки) и отношения между ними (relations).

Такие БД используют язык запросов SQL (structural query language), а программные продукты, управляющие ими — РСУБД (RDBMS).



Основные понятия:

- Строка (row, tuple, record) — набор данных, представляющий один элемент.
- Колонка (column, attribute, field) — именованный элемент строки.
- Таблица (table, relation) — набор строк, имеющих одинаковый набор атрибутов.
- Представление (view, result set) — любой набор строк или результат запроса.

Пример представления данных в таблицах.

Таблица course:

	id [PK] integer	title character varying(255)	duration smallint	price money
1	1	Python	150	20 000.00 гр
2	2	PHP	174	25 000.00 гр
3	3	Ruby	135	17 000.00 гр
4	4	Java	145	30 000.00 гр
*				

Таблица teacher:

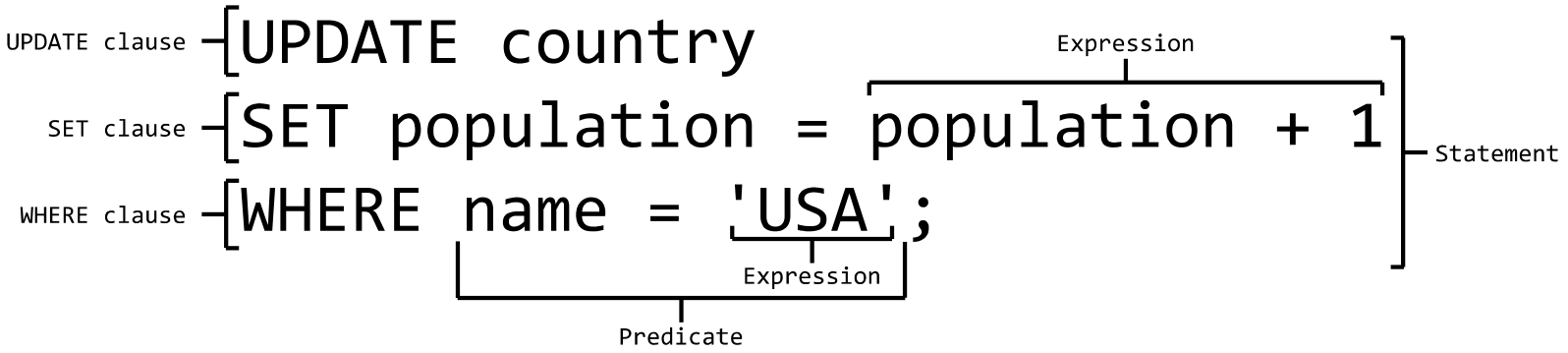
	id [PK] integer	name character varying(255)	salary money	course_id integer
1	1	Ivan Drago	15 000.00 гр	1
2	2	Chuck Norris	18 000.00 гр	2
3	3	Takeshi Kitano	7 000.00 гр	3
4	4	James Gosling	357 000.00 гр	4
*				

SQL (англ. structured query language — «язык структурированных запросов») — формальный непроцедурный язык программирования, применяемый для создания, модификации и управления данными в произвольной реляционной базе данных, управляемой соответствующей системой управления базами данных (СУБД).

SQL делится на следующие подтипы:

- DDL (data definition language) — язык описания данных;
- DML (data manipulation language) — язык управления данными;
- DCL (data control language) — язык управления доступом.

Структура SQL запроса:



- Clause — структурный копонент запроса.
- Expression — создает скалярную величину или таблицу.
- Predicate — могут возвращать true/false/unknown при некоторых условиях.
- Query — извлекает данные по определенным критериям.
- Statement — выражение по обработке данных, оканчивающееся `;`.

Стандартный SQL имеет набор типов данных, что дает максимальное удобство использования и высокую скорость работы БД.

Частоиспользуемые типы:

- CHARACTER(n) — строка фиксированной длины n;
- VARCHAR(n) — строка переменной длины, макс. n;
- BOOLEAN — булевое значение;
- SMALLINT, INTEGER, BIGINT — целые числа;
- DECIMAL(p,s) — дробное число, p — количество знаков перед точкой, s — после точки;
- FLOAT, REAL — дробные числа;
- DATE, TIME, TIMESTAMP, INTERVAL — форматы хранения даты/времени

SQL поддерживает и операторы:

- = - равно;
- !=, <> - не равно;
- <, >, <=, >= - строгие и нестрогие неравенства;
- BETWEEN — вхождение в дипазон;
- LIKE — равенство по шаблону;
- IN — равенство хотябы одному значению из перечисленных;
- IS или IS NOT — сравнение с null.

Таблицы создаются при помощи команды CREATE TABLE:

```
CREATE TABLE course
(
  id INTEGER PRIMARY KEY,
  title VARCHAR(255),
  duration smallint,
  price money
);
```

Создастся таблица:

```
  Column  |           Type           | Modifiers
----------+--------------------------+-----------
 id       | integer                  | not null
 title    | character varying(255)   |
 duration | smallint                 |
 price    | money                    |
```

```
CREATE TABLE teacher
(
  id INTEGER PRIMARY KEY,
  name VARCHAR(255),
  salary MONEY,
  course_id INTEGER NOT NULL
);
```

Создаст таблицу:

```
  Column   |           Type           | Modifiers
-----------+--------------------------+-----------
 id        | integer                  | not null
 name      | character varying(255)   |
 salary    | money                    |
 course_id | integer                  | not null
```

NOT NULL означает, что значение не может быть null.

Но таблица пустая, чтобы наполнить ее данными используется конструкция INSERT INTO:

```
INSERT INTO course VALUES (1, 'Python', 150, 20000);
INSERT INTO course VALUES (2, 'PHP', 174, 25000);
INSERT INTO course VALUES (3, 'Ruby', 135, 17000);
INSERT INTO course VALUES (4, 'Java', 145, 30000);

INSERT INTO teacher VALUES (1, 'Ivan Drago', 15000, 1);
INSERT INTO teacher VALUES (2, 'Chuck Norris', 18000, 2);
INSERT INTO teacher VALUES (3, 'Takeshi Kitano', 7000, 3);
INSERT INTO teacher VALUES (4, 'Games Gosling', 357000, 4);
```

Мы можем выбрать все значения из таблиц при помощи SELECT *:

SELECT * FROM course;

```
 id | title  | duration |    price
----+--------+----------+--------------
  1 | Python |      150 | 20 000.00 rp
  2 | PHP    |      174 | 25 000.00 rp
  3 | Ruby   |      135 | 17 000.00 rp
  4 | Java   |      145 | 30 000.00 rp
(4 rows)
```

SELECT * FROM teacher;

```
 id |       name       |     salary      | course_id
----+------------------+-----------------+-----------
  1 | Ivan Drago       |  15 000.00 rp |          1
  2 | Chuck Norris     |  18 000.00 rp |          2
  3 | Takeshi Kitano   |   7 000.00 rp |          3
  4 | James Gosling    | 357 000.00 rp |          4
(4 rows)
```

Следует обратить внимание, что course_id в таблице teacher соттветсвует id из таблицы course, что позволяет узнать какой курс ведет преподаватель.

Можно выбирать не все столбцы, а только некоторые, например:

SELECT title FROM course;

```
 title
--------
 Python
 PHP
 Ruby
 Java
(4 rows)
```

Можно отфильтровать нужные нам строки с помощью WHERE:

SELECT * FROM course WHERE title=`Ruby`;

```
 id | title | duration |    price
----+-------+----------+--------------
  3 | Ruby  |      135 | 17 000.00 rp
(1 row)
```

Можем сортировать при помощи ORDER BY:

SELECT * FROM course ORDER BY duration;

```
 id | title  | duration |    price
----+--------+----------+--------------
  3 | Ruby   |      135 | 17 000.00 rp
  4 | Java   |      145 | 30 000.00 rp
  1 | Python |      150 | 20 000.00 rp
  2 | PHP    |      174 | 25 000.00 rp
(4 rows)
```

Или в обратном порядке:

SELECT * FROM course ORDER BY duration DESC;

```
 id | title  | duration |    price
----+--------+----------+--------------
  2 | PHP    |      174 | 25 000.00 rp
  1 | Python |      150 | 20 000.00 rp
  4 | Java   |      145 | 30 000.00 rp
  3 | Ruby   |      135 | 17 000.00 rp
(4 rows)
```

Или применить агрегирующие функции:

SELECT count(*) from course;

```
 count
-------
     4
(1 row)
```

С помощью таких функций можно подсчитать, например, общую стоимость всех курсов:

SELECT sum(price) FROM course;

```
      sum
--------------
 92 000.00 гр
(1 row)
```

Или среднюю:

SELECT avg(price) FROM course;

```
  avg
-------
 23000
(1 row)
```

Оператор AS позволяет назначать другие имена столбцам:

SELECT title AS language FROM course;

```
 language
----------
 Python
 PHP
 Ruby
 Java
(4 rows)
```

A LIMIT ограничить результат выдачи (найдем два самых дешевых курса):

SELECT * FROM course ORDER BY price LIMIT 2;

```
 id | title  | duration |    price
----+--------+----------+--------------
  3 | Ruby   |      135 | 17 000.00 rp
  1 | Python |      150 | 20 000.00 rp
(2 rows)
```

Удаление строк осуществляется с помощью команды DELETE:

DELETE * FROM course WHERE title=`PHP`;

Если сделать DELETE без фильтра, то он удалит все значения из таблицы.

Чтобы удалить таблицу используется DROP TABLE:

DROP TABLE course;