Основы базы данных
SQL
Урок 4

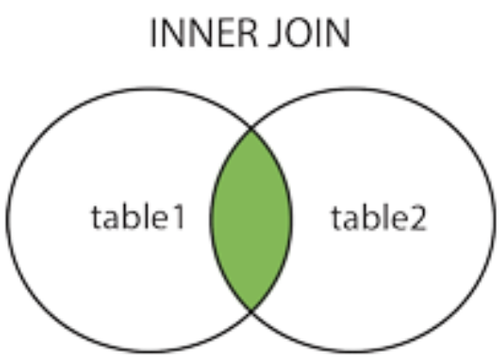
2020 г.

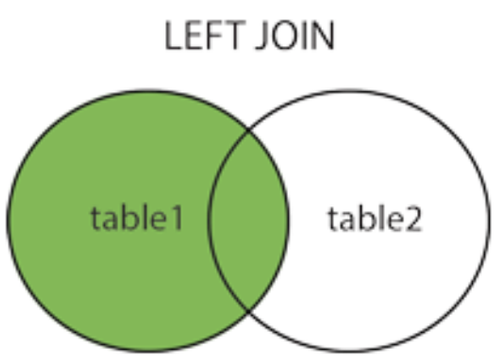
Summary of third week

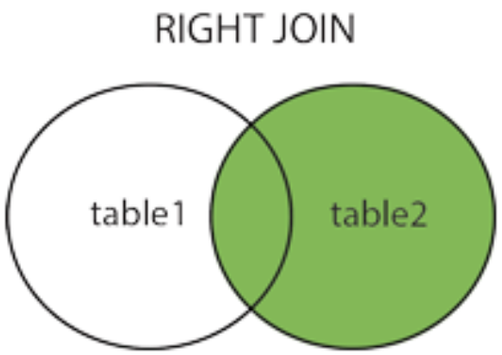
1. **HAVING**
2. **INNER JOIN**
3. **LEFT JOIN**
4. **RIGHT JOIN**
5. **FULL JOIN**
6. **SELF JOIN**
7. **UNION / UNION ALL**
8. **SUBQUERY**

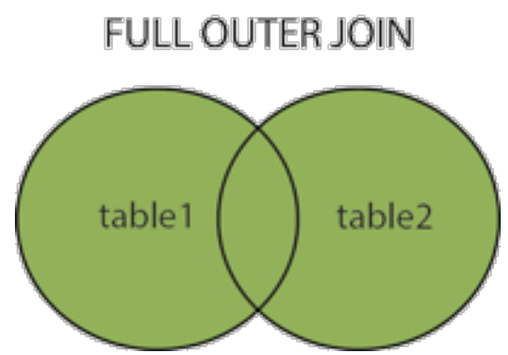
```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

```
SELECT    select_list
FROM      table
WHERE     expr operator
          (SELECT    select_list
           FROM      table);
```









Операторы сравнения

Операторы сравнения используются в операторе WHERE, чтобы определить, какие записи выбрать.

Оператор ставнения	Описание
=	Равно
<>	Не равно
!=	Не равно
>	Больше, чем
>=	Больше или равно
<	Меньше, чем
<=	Меньше или равно
!>	Не больше, чем
!<	Не меньше чем
IN ( )	Соответствует значению в списке
NOT	Отрицает условие
BETWEEN	В пределах диапазона (включительно)
IS NULL	Значение NULL
IS NOT NULL	Значение, отличное от NULL

SUBSTRING() Function

SUBSTRING() - позволяет извлечь из выражения его часть заданной длины, начиная от заданной начальной позиции.

```
SELECT SUBSTRING(input_string, start, length);
```

input_string - строка, из которой нужно извлечь
start - позиция начала извлечения. Первая позиция в строке всегда 1.
length - количество символов для извлечения.
Должно быть положительное число

> *Если длина(length) — отрицательное число, то функция SUBSTRING вернет ошибку.*

CASE Statement

Оператор CASE перебирает условия и возвращает значение, когда выполняется первое условие (например, оператор IF-THEN-ELSE). Итак, как только условие выполнено, оно перестанет читать и вернет результат. Если ни одно из условий не выполняется, возвращается значение из предложения ELSE.

Если части **ELSE** нет и не выполняются никакие условия, возвращается NULL.

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

CONCAT() Function

Функция CONCAT () складывает две или более строк вместе.

```
SELECT CONCAT(string1, string2, ...., string_n)
```

string1, string2, string_n - строки для сложения

UPPER / LOWER Functions

Функция **UPPER** преобразует все буквы указанной строки в верхний регистр.

```
SELECT UPPER(String);
```

Функция **LOWER** преобразует все буквы в указанной строке в нижний регистр.

```
SELECT LOWER(String);
```

String *— строка для преобразования в нижний регистр.*

> *Если в строке есть символы, которые не являются буквами, они не зависят от этой функции.*

DATEDIFF() & DATEPART() Function

Функция **DATEDIFF** возвращает разность между двумя значениями даты в зависимости от указанного интервала.

```
SELECT DATEDIFF(interval, date1, date2);
```

Функция **DATEPART()** возвращает указанную часть даты. Эта функция возвращает результат в виде целого числа.

```
SELECT DATEPART(interval, date);
```

interval — интервал времени для вычисления разницы между *date1* и *date2*.

- year, yyyy, yy = Year
- quarter, qq, q = Quarter
- month, mm, m = month
- dayofyear, dy, y = Day of the year
- day, dd, d = Day of the month
- week, ww, wk = Week
- weekday, dw, w = Weekday
- hour, hh = hour
- minute, mi, n = Minute
- second, ss, s = Second
- millisecond, ms = Millisecond

date1 и *date2* - две даты для расчета разницы между ними.

CONVERT() Function

Функция **CONVERT** преобразует выражение из одного типа данных в другой тип данных.

```
SELECT CONVERT(data_type(length), expression, style)
```

data_type* — тип данных, в который вы хотите преобразовать выражение. Это может быть одно из следующих: bigint, int, smallint, tinyint, bit, decimal, numeric, money, smallmoney, float, real, datetime, smalldatetime, char, varchar, text, nchar, nvarchar, ntext, binary, varbinary, или image.

length — длина результирующего типа данных для char, varchar, nchar, nvarchar, binary и varbinary.

expression* — значение для преобразования в другой тип данных.

style — формат, используемый для преобразования между типами данных, такими как формат даты или строковый формат. Это может быть одно из следующих значений:

LEN() Function

Функция **LEN() length** используется для подсчета количества символов в строках.

```
SELECT LEN(String);
```

> *Примечание. Конечные пробелы в конце строки не учитываются при вычислении длины. Однако при расчете длины учитываются ведущие пробелы в начале строки.*

TRIM() Functions

Функция **TRIM()** удаляет пробел или другие указанные символы из начала или конца строки.

```
SELECT TRIM([characters FROM ]string);
```

characters FROM - определенные символы для удаления
string * - строка, из которой удаляются пробелы или символы

> *По умолчанию функция TRIM () удаляет начальные и конечные пробелы из строки.*

VIEW Statement

Оператор **VIEW** — объект базы данных, представляющий собой представление. Представление — это виртуальная таблица, внутреннее содержимое которой определяется исходя из параметров запроса.

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Представления широко используются когда необходимо представить структуру базы данных в удобном для восприятия человеком виде, а так же в соображениях безопасности, предоставляя пользователям возможность обращаться к данным, но не разрешая им доступ к исходным таблицам.

> *Примечание: представление всегда показывает актуальные данные! Ядро базы данных воссоздает данные, используя оператор SQL представления, каждый раз, когда пользователь запрашивает представление.*

ORDER BY никогда не используется в определении представлений. Вывод запроса формирует содержание представления, которое напоминает базовую таблицу и является - по определению - неупорядоченным.

Questions?