SQL
Занятие 6
Функции
DML операции

Бояр Владислав

Центр непрерывного образования ФКН ВШЭ 2024

Функции

Функции

Математические

- **abs**(a) – модуль числа a
- **power**(a,b) - возведение a в степень b
- **sqrt**(a) – квадратный корень a
- **round**(a,b) – округление a до b десятичных знаков
- **sign**(a) – возвращает знак аргумента
- **least**(a, b, ...) – возвращает наименьшее значение из списка
- **greatest**(a, b, ...) – возвращает наибольшее значение из списка

Функции

Строковые

- `'a' || 'b'` - конкатенация строк (NULL учитываются)
- `concat('a', 'b')` – конкатенация строк (NULL игнорируются)
- `substring('string' from 2 for 3)` - извлекает из string подстроку, начиная с позиции `from`, длиной до кол-ва символов `for`
- `replace('string', from, to)` - заменяет все вхождения в string подстроки from подстрокой to
- `left('string', n)` – возвращает первые n символов строки
- `right('string', n)` – возвращает последние n символов строки
- `lower('string')` - приведение к нижнему регистру
- `upper('string')` – приведение к верхнему регистру
- `length` – возвращает длину строки
- `ltrim(str1, str2)` – удаляет из str1 наибольшую подстроку, содержащую только символы str2 (по умолчанию пробелы) с начала строки str1
- `rtrim(str1, str2)` – аналогично предыдущему пункту, только с конца строки str1

Поиск по шаблону LIKE

- Выражение **LIKE** возвращает **`true`**, если строка соответствует заданному шаблону.
- Подчёркивание (_) в шаблоне подменяет любой символ.
- Знак процента (%) подменяет любую (в том числе и пустую) последовательность символов.
- Вместо LIKE можно использовать ключевое слово ILIKE, чтобы поиск был регистронезависимым.

Примеры:

```
'abc' LIKE 'abc' true
'abc' LIKE 'a%' true
'abc' LIKE '_b_' true
'abc' LIKE 'c' false
```

COALESCE

- Функция COALESCE возвращает первый попавшийся аргумент, отличный от NULL.
- Если же все аргументы равны NULL, результатом тоже будет NULL.
- Часто используется для подстановки некоторого значения по умолчанию вместо значений NULL.

```
COALESCE(значение1, значение2, [, ...])
```

Условное выражение CASE

Является аналогом конструкции IF / ELSE в других языках программирования

```
CASE  -- общий синтаксис
    WHEN условие THEN результат
    [WHEN ...]
    [ELSE результат]
END
```

- Возвращается первый результат при совпадении условия. Остальная часть выражения при этом не вычисляется.
- Если не выполняется ни одно из условий WHEN, значением CASE становится результат, записанный в предложении ELSE.
- Если отсутствует ELSE – вернётся NULL.

```
CASE выражение  -- упрощённый синтаксис
    WHEN значение THEN результат
    [WHEN ...]
    [ELSE результат]
END
```

Crosstab

Функция Crosstab позволяет преобразовывать исходные данные следующим образом:

model	feature	value
m1	f1	v1
m1	f2	v2
m2	f1	v3
m2	f2	v4
m3	f1	v5
m3	f2	v6

model	f1	f2
m1	v1	v2
m2	v3	v4
m3	v5	v6

Crosstab

Пример запроса

```
SELECT *
  FROM crosstab(
       'select model
             ,feature
             ,value
         from models
        order by 1,2')
       AS models(model text, f1 text, f2 text)
;
```

WITH

- Представляет собой способ записывать дополнительные операторы для применения в больших запросах.
- Эти операторы, которые также называют общими табличными выражениями (Common Table Expressions, CTE), можно представить как определения временных таблиц, существующих только для одного запроса.
- Основное предназначение SELECT в предложении WITH заключается в разбиении сложных запросов на простые части.
- Также конструкция WITH позволяет избежать повторения кода.

```
WITH cte_name AS (
    SELECT ...
      FROM ...
), cte_name_2 (
    ...
)
SELECT *
  FROM cte_name
    JOIN cte_name_2
       ON ...
```

DML операции

Группы операторов SQL

- **DML** (**D**ata **M**anipulation **L**anguage) – манипуляции с данными
  - SELECT – выборка;
  - INSERT – вставка;
  - UPDATE – обновление;
  - DELETE – удаление.

- **DDL** (**D**ata **D**efinition **L**anguage) – работа с объектами БД
  - CREATE – создание;
  - ALTER – изменение;
  - DROP – удаление.

- **DCL** (**D**ata **C**ontrol **L**anguage) – определение доступа к данным
  - GRANT – предоставление прав;
  - REVOKE – отзыв прав;
  - DENY – запрет действий.

- **TCL** (**T**ransaction **C**ontrol **L**anguage) – управление транзакциями

Группы операторов SQL

- **DML** (**D**ata **M**anipulation **L**anguage) – манипуляции с данными
  - SELECT – выборка;
  - **INSERT – вставка;**
  - **UPDATE – обновление;**
  - **DELETE – удаление.**

- **DDL** (**D**ata **D**efinition **L**anguage) – работа с объектами БД
  - CREATE – создание;
  - ALTER – изменение;
  - DROP – удаление.

- **DCL** (**D**ata **C**ontrol **L**anguage) – определение доступа к данным
  - GRANT – предоставление прав;
  - REVOKE – отзыв прав;
  - DENY – запрет действий.

- **TCL** (**T**ransaction **C**ontrol **L**anguage) – управление транзакциями

INSERT

- **INSERT** добавляет строки в таблицу;
- Имена целевых полей могут быть перечислены в любом порядке.

Основные способы вставки:

```
-- явное указание значений
INSERT INTO table_name (column_name_1, column_name_2, ...)
VALUES
(value_1, value_2, ...),
(value_3, value_4, ...);

-- формирование таблицы значений для вставки в подзапросе
INSERT INTO table_name (column_name_1, column_name_2, ...)
SELECT column_name_1, column_name_2, ...
  FROM ... ;
```

UPDATE

- **UPDATE** изменяет значения полей во всех строках, удовлетворяющих условию;
- В предложении **SET** должны указываться только те столбцы, которые будут изменены; столбцы, не изменяемые явно, сохраняют свои предыдущие значения;
- Изменить строки в таблице, используя информацию из других таблиц в базе данных, можно двумя способами: применяя вложенные запросы или указав дополнительные таблицы в предложении FROM.

Синтаксис:

```
UPDATE table_name -- название таблицы, где будут изменяться значения полей
   SET column_name = value, -- название поля и новое значение
       [column_name_n = value_n]
[FROM ...]  -- опционально, название дополнительной таблицы
[WHERE ...] -- опционально, условие замены значений
```

DELETE

- **DELETE** удаляет из таблицы строки, удовлетворяющие условию WHERE;
- Если предложение WHERE отсутствует, команда удаляет из таблицы все строки. В результате будет получена рабочая, но пустая таблица.

Синтаксис:

```
DELETE FROM table_name
 WHERE ...;
```

TRUNCATE

- **TRUNCATE** – удаляет все строки из таблицы или набора таблиц;
- более быстрый механизм удаления всех строк из таблицы;
- наиболее полезна для очистки больших таблиц.

Синтаксис:

```
TRUNCATE table_name_1, table_name_2, ..., table_name_n;
```

Практика