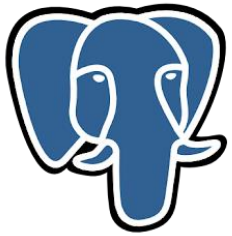
Написание простых запросов

Синтаксис оператора SELECT



PostgreSQL

Элемент	Выражение	Описание
SELECT	Список столбцов через запятую	Определяет, какие столбцы должна содержать результирующая таблица
FROM	Определение таблиц-источников строк	Определяет таблицы-источники для извлечения данных
WHERE	Условие отбора исходных строк	Фильтрует данные из таблиц-источников с помощью предиката
GROUP BY	Группировка по списку столбцов	Упорядочивает строки по группам
HAVING	Условие отбора групп	Фильтрует группы с помощью предиката
ORDER BY	Сортировка по списку столбцов	Сортирует строки результирующей таблицы

Логическая последовательность выполнения оператора SELECT

- Порядок, в котором запрос записывается отличается от порядка в котором запрос выполняется сервером БД

5.	SELECT	<select list>
7.	[INTO	new_table_name]
1.	**FROM**	<table source>
2.	WHERE	<search condition>
3.	GROUP BY	<group by list>
4.	HAVING	<search condition>
6.	ORDER BY	<order by list>[ASC | DESC]

Применение логического порядка операций к написанию SELECT


```
SELECT empid,
       extract('year' from orderdate) AS OrderYear
FROM "Sales"."Orders"
WHERE custid  = 71
GROUP BY empid, extract('year' from orderdate)
HAVING COUNT(*) > 2
ORDER BY empid, OrderYear;
```

	empid	orderyear
1	1	2 008
2	4	2 008
3	5	2 007
4	6	2 007
5	8	2 007

SELECT «безо всего»

- Используется для:
  - Инициализации переменных;
  - Возврата результата выражений и функций;

```
SELECT now(), current_database(), current_user, session_user;
```

	now	current_database	current_user	session_user
1	2022-06-14 11:51:56	dbSQL	postgres	postgres

Заголовки !

```
SELECT 2 * 2 , 5 % 2, 2 * 2 AS Four, 5 % 2 AS Ostatok;
```

Заголовки ?

	?column?	?column?	four	ostatok
1	4	1	4	1

Заголовки !

Извлечение данных из таблицы

- Извлечение из всех столбцов таблицы

```
SELECT *
FROM "Sales"."Customers";
```

	custid	companyname	contactname	contacttitle	address	city	region	postalcod	country	phone	fax	tag
1	1	Customer NRZBB	Allen, Michael	Sales Representative	Obere Str. 0123	Berlin	[NULL]	10092	Germany	030-34567	030-0123	1
2	2	Customer MLTDN	Hassall, Mark	Owner	Avda. de la Cons	México D.F.	[NULL]	10077	Mexico	(5) 789-012	(5) 456-78	1
3	3	Customer KBUDE	Peoples, John	Owner	Mataderos 789(	México D.F.	[NULL]	10097	Mexico	(5) 123-456	[NULL]	1

- Извлечение из отдельных столбцов таблицы

```
SELECT companyname, country
FROM "Sales"."Customers";
```

	companyname	country
1	Customer NRZBB	Germany
2	Customer MLTDN	Mexico
3	Customer KBUDE	Mexico
4	Customer HFBZG	UK

Элементы языка

Элементы языка:	**Предикаты и Операторы:**
Предикаты	BETWEEN, IN, LIKE, IS, ALL, ANY, SOME
Операторы сравнения	=, >, <, >=, <=, <> (!=)
Логические операторы	AND, OR, NOT
Арифметические операторы	*, /, %, +, -, - (унарный)
Конкатенация (*зависит от диалекта языка)	|| *(&, +)

Вычисляемые столбцы и псевдонимы столбцов

- Создание вычисляемых столбцов

```
SELECT unitprice, qty, (qty * unitprice)
FROM "Sales"."OrderDetails";
```

Заголовок?

	unitprice	qty	?column?
1	$14.00	12	$168.00
2	$9.80	10	$98.00
3	$34.80	5	$174.00

- Псевдонимы
  - Заключаются в двойные кавычки, если содержат пробелы, специальные символы или необходимо различать регистры символов

```
SELECT unitprice, qty Quantity , (qty * unitprice) AS Total
FROM "Sales"."OrderDetails";
```

	unitprice	quantity	total
1	$14.00	12	$168.00
2	$9.80	10	$98.00
3	$34.80	5	$174.00
4	$18.60	9	$167.40

Псевдонимы таблиц

- Создаются в предложении FROM
- Полезны при выборке данных из нескольких таблиц

```
SELECT custid, orderdate
FROM "Sales"."Orders" AS SO;
```

- Ссылка на столбцы таблицы с использованием псевдонима таблицы

```
SELECT "Sales"."Orders".custid, "Sales"."Orders".orderdate
FROM "Sales"."Orders" AS SO;
```

```
SELECT SO.custid, SO.orderdate
FROM "Sales"."Orders" AS SO;
```

Влияние логического порядка выполнения запроса на псевдонимы

- Предложения FROM, WHERE и HAVING обрабатываются **до** SELECT
- Псевдонимы столбцов создаются в SELECT и **видны только** в ORDER BY
- Выражения, для которых в предложении SELECT определены псевдонимы, должны быть повторно использованы в остальных предложениях запроса

Использование выражения CASE в предложении SELECT

- Выражение CASE возвращает скалярное значение
- CASE может использоваться:
  - Для создания вычисляемого столбца в **SELECT**
  - Для формирования условия в **WHERE** или **HAVING**
  - Для задания порядка сортировки в **ORDER BY**
- CASE возвращает результат вычисления выражения

Формы выражений CASE

- **Simple CASE**
  - Сравнивает одно выражение со списком возможных значений
  - Возвращает первое совпадение
  - Если совпадений не обнаружено, возвращает значение, основываясь на выражении ELSE
  - Если не найдено совпадений и не определено выражение ELSE, возвращает NULL
- **Searched CASE**
  - Проверяет набор предикатов или логических выражений
  - Возвращает значение указанное в выражении THEN первого выражения, которое возвращает TRUE

Simple CASE


```
CASE input_expr
    WHEN when_expr THEN result_expr
    […]
    [ELSE else_result_expr]
END
```

Если поле
- **Равно** значение1 **тогда** результат1
- **Равно** значение2 **тогда** результат2
- **Равно** значение3 **тогда** результат3
- **…**
- **ИНАЧЕ** Результат_Иначе

```
select orderid, orderdate,shipperid,
case shipperid
    when 1 then 'Shipper GVSUA'
    when 2 then 'Shipper ETYNR'
    when 3 then 'Shipper ZHISN'
    else 'MMM & Ko'
end as "Shippers"
from "Sales"."Orders";
```

	orderid	orderdate	shipperid	Shippers
1	10 250	2006-07-08	2	Shipper ETYNR
2	10 253	2006-07-10	2	Shipper ETYNR
3	10 256	2006-07-15	2	Shipper ETYNR
4	10 257	2006-07-16	3	Shipper ZHISN
5	10 261	2006-07-19	2	Shipper ETYNR
6	10 262	2006-07-22	3	Shipper ZHISN
7	10 268	2006-07-30	3	Shipper ZHISN
8	10 269	2006-07-31	1	Shipper GVSUA
9	10 271	2006-08-01	2	Shipper ETYNR
10	10 272	2006-08-02	2	Shipper ETYNR

Searched CASE


```
SELECT productname, unitprice,
    CASE
        WHEN discontinued = 1::bit THEN 'Снят с продажи'
        WHEN unitprice < 25::money THEN 'Нижняя ценовая категория'
        WHEN unitprice BETWEEN 25::money AND 30::money THEN 'Средняя ценовая категория'
        WHEN unitprice BETWEEN 31::money AND 50::money THEN 'Высокая ценовая категория'
    ELSE 'VIP товар'
END AS "Price category"
FROM "Production"."Products";
```

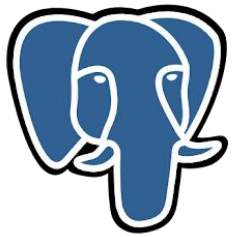
	productname	unitprice	Price category
1	Product HHYDP	18.0000	Нижняя ценовая категория
2	Product RECZE	19.0000	Нижняя ценовая категория
3	Product IMEHJ	10.0000	Нижняя ценовая категория
4	Product KSBRM	22.0000	Нижняя ценовая категория
5	Product EPEIM	21.3500	Снят с продажи
6	Product VAIIV	25.0000	Средняя ценовая категория
7	Product HMLNI	30.0000	Средняя ценовая категория
8	Product WVJFP	40.0000	Высокая ценовая категория

Использование функций

Функции форматирования и преобразования	Поддержка приведения и преобразования типов данных	CAST, TO_CHAR, TO_DATE, TO_NUMBER, TO_TIMESTAMP
Логические функции	Выполнение логических операций	NULLIF, GREATEST, LEAST
Функции даты и времени	Выполняют операции над значениями даты и времени	AGE, NOW, CURRENT_DATE, CURRENT_TIME, LOCALTIME, DATE_PART, DATE_TRUNC, MAKE_DATE, EXTRACT
Строковые функции	Выполняют операции со строковыми (char или varchar) значениями	CONCAT, CONCAT_WS, FORMAT, LEFT, LENGTH, LOWER, LTRIM, REPLACE, REGEXP_REPLACE, REVERSE, RIGHT, RTRIM, SUBSTRING, TRIM, UPPER
Математические функции	Выполняют вычисления, основанные на числовых значениях	ABS, CEILING, FLOOR, POWER, ROUND, SQRT, TRUNC
Функции для перечислений	Используются для работы с типами перечислений (ENUM)	ENUM_FIRST, ENUM_LAST, ENUM_RANGE

https://postgrespro.ru/docs/postgresql/14/functions

Использование функций




```
SELECT  companyname
      , REPLACE(companyname, 'Customer ','') AS NAME
FROM "Sales"."Customers" c ;
```

	companyname	NAME
1	Customer AHPOP	AHPOP
2	Customer AHXHT	AHXHT
3	Customer AZJED	AZJED
4	Customer BSVAR	BSVAR
5	Customer CCFIZ	CCFIZ

```
SELECT  contactname
      , LEFT(contactname
      , POSITION(',' in contactname)-1) AS FName
FROM "Sales"."Customers" c ;
```

	contactname	FName
1	Allen, Michael	Allen
2	Hassall, Mark	Hassall
3	Peoples, John	Peoples
4	Arndt, Torsten	Arndt
5	Higginbotham, Tom	Higginbotham

```
SELECT  orderid, orderdate,
        CONCAT(date_part('year',orderdate)
               , '-'
               , date_part('month',orderdate)) AS PERIOD
FROM "Sales"."Orders" o ;
```

	orderid	orderdate	PERIOD
1	10 248	2006-07-04 00:00:00.000	2006-7
2	10 249	2006-07-05 00:00:00.000	2006-7
3	10 250	2006-07-08 00:00:00.000	2006-7
4	10 251	2006-07-08 00:00:00.000	2006-7
5	10 252	2006-07-09 00:00:00.000	2006-7

СОРТИРОВКА ДАННЫХ

Использование ORDER BY


```
ORDER BY { order_by_expression [ ASC | DESC ] } [ ,...n ]
```

- ORDER BY сортирует записи в результирующем наборе
  - Без ORDER BY порядок записей результирующей выборки не гарантируется
  - Сортирует все NULL значения вместе
- ORDER BY может ссылаться на:
  - Имя столбца, псевдоним или позицию столбца в результирующей выборке (не рекомендуется)
  - Результат выражения
  - Столбцы, не используемые в результирующей выборке
    - Если не используется DISTINCT
- ORDER BY не поддерживается в инструкциях SELECT/INTO

Пример использования ORDER BY


```
SELECT companyname, contactname
FROM "Sales"."Customers" c
ORDER BY country ASC,  city desc;
```

	companyname	contactname
1	Customer PSNMQ	Ray, Mike
2	Customer TDKEG	Tiano, Mike
3	Customer LWGMD	Gaffney, Lawrie
4	Customer LOLJO	Meston, Tosh
5	Customer THHDP	Kane, John

```
SELECT custid, orderdate
FROM "Sales"."Orders" o
ORDER BY DATE_PART('year',orderdate) DESC;
```

	custid	orderdate
1	55	2008-01-01 00:00:00.000
2	88	2008-01-01 00:00:00.000
3	42	2008-01-01 00:00:00.000
4	47	2008-01-02 00:00:00.000
5	66	2008-01-02 00:00:00.000

```
SELECT companyname, contactname
FROM "Sales"."Customers" c
ORDER BY 1;
```

	companyname	contactname
1	Customer AHPOP	Welcker, Brian
2	Customer AHXHT	Fakhouri, Fadi
3	Customer AZJED	Carlson, Jason
4	Customer BSVAR	Rizaldy, Arif
5	Customer CCFIZ	Petrov, Ivan

Фильтрация с помощью LIMIT *


```
SELECT
FROM
[ORDER BY …]
LIMIT {integer_expression | ALL }  [ OFFSET integer_expression ]
```

- Ограничивает число строк, возвращаемых в результирующем наборе
  - *integer_expression* - число или числовое выражение, определяющее количество возвращаемых строк
  - *ALL* - равносильно отсутствию указания LIMIT
  - *OFFSET* - указывает число строк, которые необходимо пропустить, прежде чем начать выдавать строки
  - Для получения предсказуемого и согласованного результата необходимо использовать фильтрацию отсортированного набора - **ORDER BY**

Фильтрация с помощью LIMIT *

```
SELECT productname, unitprice
FROM "Production"."Products"
WHERE  unitprice <= 40::money
ORDER BY unitprice DESC;
```

	productname	unitprice
1	Product WVJFP	40.0000
2	Product BLCAX	39.0000
3	Product OSFNS	38.0000
4	Product VKCMF	38.0000
5	Product COAXA	36.0000
6	Product GEEOO	34.8000
7	Product WHBYK	34.0000
8	Product HCQDE	33.2500

```
SELECT productname, unitprice
FROM "Production"."Products"
WHERE  unitprice <= 40::money
ORDER BY unitprice desc
limit 3;
```

	productname	unitprice
1	Product WVJFP	40.0000
2	Product BLCAX	39.0000
3	Product OSFNS	38.0000

```
SELECT productname, unitprice
FROM "Production"."Products"
WHERE  unitprice <= 40::money
ORDER BY unitprice desc
limit 2 offset 2;
```

	productname	unitprice
1	Product OSFNS	$38.00
2	Product VKCMF	$38.00

Фильтрация в ORDER BY с помощью OFFSET-FETCH


```
OFFSET { integer_expression } { ROW | ROWS }
[ FETCH { FIRST | NEXT } {integer_expression } { ROW | ROWS } ONLY ] }
```

- OFFSET-FETCH – это расширение ORDER BY:
  - Позволяет отфильтровать требуемый диапазон строк
  - Предоставляет механизм для разбиения результирующего набора на страницы
- Определяет количество строк, которые необходимо:
  - Пропустить - OFFSET (может быть ноль, если не нужно пропускать строки)
  - Вернуть - FETCH (должно быть больше или равно единице)
- Если FETCH опущено – возвращаются все записи до конца набора

Фильтрация с помощью OFFSET-FETCH

Извлекает первые 50 строк

```
SELECT orderid, custid, orderdate
FROM "Sales"."Orders" o
ORDER BY orderdate DESC
OFFSET 0 ROWS FETCH FIRST 50 ROWS ONLY;
```

	orderid	custid	orderdate
1	11 077	65	2008-05-06
2	11 075	68	2008-05-06
3	11 076	9	2008-05-06
4	11 074	76	2008-05-06
5	11 073	58	2008-05-05
6	11 070	44	2008-05-05
7	11 072	20	2008-05-05

Извлекает строки 51-100

```
SELECT orderid, custid, orderdate
FROM "Sales"."Orders" o
ORDER BY orderdate DESC
OFFSET 50 ROWS FETCH FIRST 50 ROWS ONLY;
```

	orderid	custid	orderdate
1	11 027	10	2008-04-16
2	11 025	87	2008-04-15
3	11 024	19	2008-04-15
4	11 026	27	2008-04-15
5	11 020	56	2008-04-14
6	11 023	11	2008-04-14
7	11 021	63	2008-04-14