

Язык SQL

SQL

- ◆ *Structured Query Language* — (структурированный язык запросов) — универсальный язык, применяемый для создания, модификации и управления данными в реляционных базах данных
- ◆ SQL основывается на исчислении кортежей, но включает также некоторые элементы исчисления доменов и реляционной алгебры, а также дополнительные расширяющие возможности
- ◆ Целью разработки было создание простого непроцедурного языка, которым мог воспользоваться любой пользователь, даже не имеющий навыков программирования

История создания

- ◆ 70-е годы язык SEQUEL (для СУБД IBM System R)
- ◆ 1979 год SQL (Oracle V2 для VAX)
- ◆ 1986 году первый стандарт языка SQL был принят ANSI
- ◆ Наиболее существенные изменения в стандарте, принятом в 2011 году (SQL:2011), с небольшими модификациями для поддержки JSON (SQL:2016)
- ◆ В июле 2023 официально принята 6 редакция стандарта (SQL:2023). Добавлена поддержка Property Graph Queries (SQL/PGQ)

Достоинства/ недостатки

- ◆ С помощью SQL описывается только то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса
- ◆ Несмотря на наличие диалектов и различий в синтаксисе, в большинстве своём тексты SQL-запросов, могут быть достаточно легко перенесены из одной СУБД в другую
- ◆ Неполное соответствие реляционной модели (повторяющиеся строки, NULL-значения и пр.)
- ◆ Сложность
- ◆ Отступление в реализациях от стандарта
- ◆ Проблемы при работе с иерархическими данными

Язык SQL

Язык SQL представляет собой совокупность

- ◆ операторов
- ◆ инструкций
- ◆ функций

Операторы

- ◆ операторы определения данных (*Data Definition Language, DDL*)
- ◆ операторы манипулирования данными (*Data Manipulation Language, DML*)
- ◆ операторы определения доступа к данным (*Data Control Language, DCL*)
- ◆ операторы управления транзакциями (*Transaction Control Language, TCL*)

Операторы DDL

- ◆ Позволяют работать с метаданными (схемой БД)
 - CREATE создает объект БД
 - ALTER изменяет объект
 - DROP удаляет объект

Операторы DML

- ◆ Позволяют управлять данными в БД
 - SELECT выбор данных из таблиц (создание новой таблицы как результата суперпозиции реляционных операций)
 - INSERT добавление новых данных
 - UPDATE изменение существующих данных
 - DELETE удаление данных

Операторы DCL

- ◆ Определяют права доступа пользователей к данным
 - GRANT предоставляет разрешения на определенные операции с объектом
 - REVOKE отзывает ранее выданные разрешения

Операторы TCL

- ◆ обеспечивают управление транзакциями
 - COMMIT зафиксировать изменения, выполненные транзакцией
 - ROLLBACK откатить все изменения, сделанные в контексте текущей транзакции
 - SAVEPOINT разделить транзакцию на более мелкие участки
 - SET TRANSACTION ISOLATION LEVEL установить необходимый уровень изолированности для транзакции

Инструкции и функции

- ◆ Встроенные функции
- ◆ Используются при реализации процедурных расширений SQL
 - Хранимые процедуры
 - Триггеры
 - Функции, блоки, пакеты и пр.



DML в Firebird

Из документации Firebird

```
[WITH [RECURSIVE] <cte> [, <cte> ...]]
SELECT
  [FIRST m] [SKIP n]
  [DISTINCT | ALL] <columns>
FROM
  source [[AS] alias]
  [<joins>]
[WHERE <condition>]
[GROUP BY <grouping-list>]
[HAVING <aggregate-condition>]]
[PLAN <plan-expr>]
[UNION [DISTINCT | ALL] <other-select>]
[ORDER BY <ordering-list>]
[  {ROWS m [TO n]}
 | {[OFFSET n {ROW | ROWS}}
   {FETCH {FIRST | NEXT} [m] {ROW | ROWS} ONLY}}
]
[FOR UPDATE [OF <columns>]]
[WITH LOCK]
[INTO <variables>]

<variables> ::= [:]varname [, [:]varname ...]
```

SELECT

SELECT

[FIRST m] [SKIP n] [DISTINCT | ALL] <columns>

FROM

source [[AS] alias]

[<joins>]

[WHERE <condition>]

[GROUP BY <grouping-list>

[HAVING <aggregate-condition>]]

[UNION [DISTINCT | ALL] <other-select>]

[ORDER BY <ordering-list>]

[ROWS m [TO n]]

Объем выборки

[FIRST (m)] [SKIP (n)]

имеет смысл только для запросов с упорядочением результата (order by)

не соответствует стандарту, рекомендуется заменять на ROWS

[ROWS m [TO n]]

В некоторых СУБД – limit(n)

Структура таблицы – результата

◆ Определение заголовка таблицы-результата

- Столбцы из таблиц, участвующих в запросе и вычисляемые столбцы

```
select a1, '--', a1+a2*0.5, . . .
```

- Для однозначности имя столбца может уточняться именем таблицы или алиасом (псевдонимом). Недопустимо смешивать алиас и имя таблицы

```
select tbl1.a1 from tbl1;  
select t.a1 from tbl1 t;
```

Структура таблицы – результата

- ◆ DISTINCT - приведение результата к реляционной модели, альтернатива ALL

- ◆ включить в результат все столбцы
select *

допустимо уточнение имени

```
select tbl1.* from tbl1;  
select t.* from tbl1 t;
```

- ◆ Для любого столбца допустима операция переименования

```
select a1 as SALARY, '=',  
a1+a2*0.5 as TOTAL,...
```

Вычисляемые столбцы

- ◆ Допустимо использование арифметических операций
- ◆ Для строковых типов определена операция конкатенации ||
- ◆ Могут использоваться встроенные функции (реализуются сервером) и внешние функции пользователя (подключаются)
- ◆ Функции
 - CAST(<значение> AS <тип данных>)
 - EXTRACT (<часть> FROM <поле>)
 - SUBSTRING (<строка>FROM<позиция> [FOR <длина>])
 - UPPER(<строка>)
 - LOWER(<строка>)
 - CHAR_LENGTH(<строка>)
 - BYTE_LENGTH(<строка>)

Вычисляемые столбцы

◆ Функции

- TRIM(<строка>)
- GEN_ID(<генератор>,<шаг>)
- IIF (<условие>, результат1, результат2)
- NULLIF(<выражение 1>,<выражение 2>)
- COALESCE(<выражение 1>{,<выражение 2>
[,...<выражение n>]})

◆ агрегатные функции (особые правила использования)

- SUM(),AVG()
- MAX(),MIN(),
- COUNT()
- LIST()

Вычисляемые столбцы

- ◆ Возможно вычисление по условию

```
CASE {<выражение 1>|<пустое предложение>}  
  WHEN {{NULL|<значение 1>}|<предикат поиска>}  
    THEN {<результат 1>|NULL}  
  WHEN . . . THEN {<результат 2>|NULL}  
  [WHEN . . . THEN {<результат n>|NULL}]  
  [ELSE {<результат n+1>|NULL}]  
END
```

Вычисляемые столбцы

◆ Системные функции и контекстные значения

- CURRENT_DATE
- CURRENT_TIME
- CURRENT_USER
- ' NOW '
- ' TODAY '
- ' YESTERDAY '
- ' TOMORROW '
- . . .

} требуют преобразования типа

Вычисляемые столбцы

- ◆ При вычислениях учитывается влияние Null-значений :
если один из операндов Null,
то и результат - Null
- ◆ В агрегатных функциях поля с Null –значением не
учитываются

Предложение FROM

◆ Определяет таблицы – операнды

FROM <source>

[<joins>]

[...]

<source> ::= { <table> | <view>
| <selectable-stored-procedure [(args)]>
| <derived-table>
| <common-table-expression>
} [[AS] <alias>]

Соединение таблиц

◆ Соединение по условию
from TABL1 [<алиас>]
[[*inner*] | [*outer*] *left* | *right* | *full*]
JOIN
TABL2 [<алиас>] on <*θ-условие*>

θ-условие определяет условие сопоставления кортежей таблиц при соединении

Чаще всего – это условие равенства значений сопоставимых атрибутов (эквисоединение)

Соединение таблиц

- Эквисоединение

```
select department
  from department D
      join employee E
  on D.mngr_no=E.emp_no
```

Соединение таблиц

- ◆ Эквисоединение для столбцов с одинаковыми именами

```
SELECT *  
FROM employee E  
    JOIN job D  
    ON E.job_code = D.job_code and E.job_grade=D.job_grade  
        and E.job_country = D.job_country
```

```
SELECT *  
FROM employee E  
    JOIN job D USING (job_code, job_grade, job_country )
```

Соединение таблиц

- ◆ Естественное соединение (с версии 2.1)

from TABL1 *natural join* TABL2

Соединение устанавливается по одноименным атрибутам путем сравнения их значений на равенство

Не рекомендуется использовать!

```
SELECT *
```

```
FROM employee E NATURAL JOIN job D
```

- ◆ Декартово произведение

from TABL1 {*cross join* | , }TABL2

Не рекомендуется использовать!

Соединение (inner/outer)

```
select full_name, department
from employee
join department
using (dept_no)
```

T1

| NT1 | CODE1 |
|-----|-------|
| 1 | c1 |
| 1 | c2 |
| 2 | c3 |

T2

| NT2 | CODE2 |
|-----|-------|
| 1 | c4 |
| 3 | c5 |

T1 [inner] join T2 on NT1=NT2

| NT1 | CODE1 | NT2 | CODE2 |
|-----|-------|-----|-------|
| 1 | c1 | 1 | c4 |
| 1 | c2 | 1 | c4 |

Соединение (inner/outer)

```
select country, job_title
from country C
     left join job J
     on C.country=J.job_country
     where J.job_country is null
```

```
select department, full_name
from department D
     left join employee E
     on D.mngr_no=E.emp_no
     where D.mngr_no is null
```

T1

| NT1 | CODE1 |
|-----|-------|
| 1 | c1 |
| 1 | c2 |
| 2 | c3 |

T2

| NT2 | CODE2 |
|-----|-------|
| 1 | c4 |
| 3 | c5 |

T1 [outer] left join T2 on NT1=NT2

| NT1 | CODE1 | NT2 | CODE2 |
|-----|-------|------|-------|
| 1 | c1 | 1 | c4 |
| 1 | c2 | 1 | c4 |
| 2 | c3 | null | null |

T1 right join T2 on NT1=NT2

| NT1 | CODE1 | NT2 | CODE2 |
|------|-------|-----|-------|
| 1 | c1 | 1 | c4 |
| 1 | c2 | 1 | c4 |
| null | null | 3 | c5 |

Соединение (inner/outer)

T1

| NT1 | CODE1 |
|-----|-------|
| 1 | c1 |
| 1 | c2 |
| 2 | c3 |

T2

| NT2 | CODE2 |
|-----|-------|
| 1 | c4 |
| 3 | c5 |

T1 full join T2 on NT1=NT2

| NT1 | CODE1 | NT2 | CODE2 |
|------|-------|------|-------|
| 1 | c1 | 1 | c4 |
| 1 | c2 | 1 | c4 |
| 2 | c3 | null | null |
| null | null | 3 | c5 |

Несколько соединений с одной таблицей

select *

from employee A join

department D on (A.dept_no=D.dept_no)

join employee B on (B.emp_no = D.mngr_no)

Нет сотрудников из отделов, у которых нет руководителя

mngr_no IS NULL

Чтобы увидеть всех сотрудников

select *

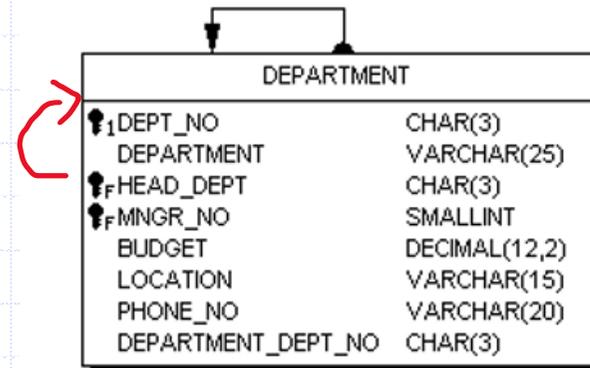
from employee A join

department D on (A.dept_no=D.dept_no)

left join employee B on (B.emp_no = D.mngr_no)

Соединение таблицы с самой собой

```
select A.department as HEAD,  
       B.department as BRANCH  
from department A join  
       department B on (A.dept_no = B.head_dept)
```



Предложение WHERE

WHERE < θ - условие отбора строк >

Условие представляет собой предикат, в который подставляются значения атрибутов из получаемого кортежа
Реализует реляционную операцию селекции

При записи предиката можно использовать :

- операции сравнения (=, <>, >, < . . .)
- логические операции NOT, AND, OR
- специальные предикаты
- функции

Предикаты

- IS NULL
- IS NOT NULL
- BETWEEN ... AND...
- CONTAINING
- IN (...)
- LIKE ' шаблон с символами % и _ '
- STARTING WITH
- Предикаты существования (EXISTS, ALL, ANY, SINGULAR)
-

Группировка

GROUP BY <список столбцов для группировки>
[HAVING < θ - условие отбора групп >]

Используется для получения общей информации по группе кортежей, имеющих одинаковые значения одного или нескольких атрибутов

Столбец или группа столбцов, указанные в предложении GROUP BY, называются элементами группировки

HAVING задает условие выбора для групп

group by location

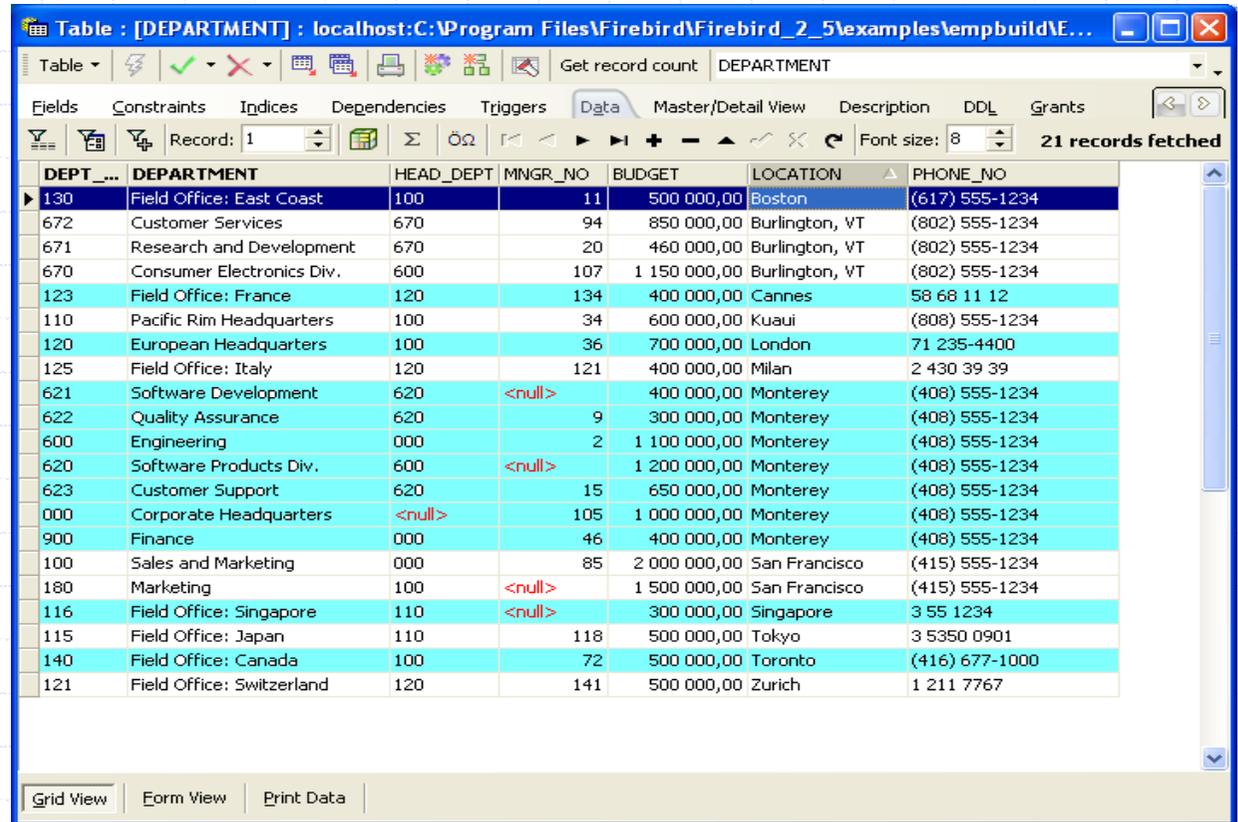


Table : [DEPARTMENT] : localhost:C:\Program Files\Firebird\Firebird_2_5\examples\empbuild\E...

Table | Get record count | DEPARTMENT

Fields | Constraints | Indices | Dependencies | Triggers | Data | Master/Detail View | Description | DDL | Grants

Record: 1 | Font size: 8 | 21 records fetched

| DEPT_... | DEPARTMENT | HEAD_DEPT | MNGR_NO | BUDGET | LOCATION | PHONE_NO |
|----------|---------------------------|-----------|---------|--------------|----------------|----------------|
| 130 | Field Office: East Coast | 100 | 11 | 500 000,00 | Boston | (617) 555-1234 |
| 672 | Customer Services | 670 | 94 | 850 000,00 | Burlington, VT | (802) 555-1234 |
| 671 | Research and Development | 670 | 20 | 460 000,00 | Burlington, VT | (802) 555-1234 |
| 670 | Consumer Electronics Div. | 600 | 107 | 1 150 000,00 | Burlington, VT | (802) 555-1234 |
| 123 | Field Office: France | 120 | 134 | 400 000,00 | Cannes | 58 68 11 12 |
| 110 | Pacific Rim Headquarters | 100 | 34 | 600 000,00 | Kuauai | (808) 555-1234 |
| 120 | European Headquarters | 100 | 36 | 700 000,00 | London | 71 235-4400 |
| 125 | Field Office: Italy | 120 | 121 | 400 000,00 | Milan | 2 430 39 39 |
| 621 | Software Development | 620 | <null> | 400 000,00 | Monterey | (408) 555-1234 |
| 622 | Quality Assurance | 620 | 9 | 300 000,00 | Monterey | (408) 555-1234 |
| 600 | Engineering | 000 | 2 | 1 100 000,00 | Monterey | (408) 555-1234 |
| 620 | Software Products Div. | 600 | <null> | 1 200 000,00 | Monterey | (408) 555-1234 |
| 623 | Customer Support | 620 | 15 | 650 000,00 | Monterey | (408) 555-1234 |
| 000 | Corporate Headquarters | <null> | 105 | 1 000 000,00 | Monterey | (408) 555-1234 |
| 900 | Finance | 000 | 46 | 400 000,00 | Monterey | (408) 555-1234 |
| 100 | Sales and Marketing | 000 | 85 | 2 000 000,00 | San Francisco | (415) 555-1234 |
| 180 | Marketing | 100 | <null> | 1 500 000,00 | San Francisco | (415) 555-1234 |
| 116 | Field Office: Singapore | 110 | <null> | 300 000,00 | Singapore | 3 55 1234 |
| 115 | Field Office: Japan | 110 | 118 | 500 000,00 | Tokyo | 3 5350 0901 |
| 140 | Field Office: Canada | 100 | 72 | 500 000,00 | Toronto | (416) 677-1000 |
| 121 | Field Office: Switzerland | 120 | 141 | 500 000,00 | Zurich | 1 211 7767 |

Grid View | Form View | Print Data

select location, sum(budget)
from department
group by location

| LOCATION | SUM |
|----------------|--------------|
| Boston | 500 000,00 |
| Burlington, VT | 2 460 000,00 |
| Cannes | 400 000,00 |
| Kuauai | 600 000,00 |
| London | 700 000,00 |
| Milan | 400 000,00 |
| Monterey | 5 050 000,00 |
| San Francisco | 3 500 000,00 |
| Singapore | 300 000,00 |
| Tokyo | 500 000,00 |
| Toronto | 500 000,00 |
| Zurich | 500 000,00 |

◆ В списке группировки можно использовать

- столбцы таблицы
- выражения, вычисляемые на основе значений в столбцах таблицы
- алиасы столбцов из списка в предложении select (с версии 2.1)
- их порядковые номера (с версии 2.1) – *противоречит реляционной модели!!!*

◆ ограничение:

- столбцами таблицы, формируемой оператором SELECT, могут быть или *элементы группировки*, или *константы*, или *выражения с использованием агрегатных функций для столбцов*, не являющихся элементами группировки

Агрегатные функции

- SUM(<столбец>)
- AVG(<столбец>)
- MAX (<столбец>)
- MIN(<столбец>)
- COUNT(*) или COUNT(<столбец>)
или COUNT(DISTINCT <столбец>)
- LIST (<столбец> [,<разделитель>])
- ◆ не учитывают NULL, если в столбце есть значения, отличные от NULL
- ◆ можно использовать при описании условий на группу

Count

```
select location,  
       count(*),  
       count(head_dept),  
       count(distinct head_dept)  
from department  
group by location
```

| LOCATION | COUNT | COUNT1 | COUNT2 |
|----------------|-------|--------|--------|
| Boston | 1 | 1 | 1 |
| Burlington, VT | 3 | 3 | 2 |
| Cannes | 1 | 1 | 1 |
| Kuauai | 1 | 1 | 1 |
| London | 1 | 1 | 1 |
| Milan | 1 | 1 | 1 |
| Monterey | 7 | 6 | 3 |
| San Francisco | 2 | 2 | 2 |
| Singapore | 1 | 1 | 1 |
| Tokyo | 1 | 1 | 1 |
| Toronto | 1 | 1 | 1 |
| Zurich | 1 | 1 | 1 |

Агрегатные функции

- ◆ Агрегатные функции используются при описании условия на группу

```
SELECT PROJ_ID,  
       SUM (PROJECTED_BUDGET) AS TOTAL_BUDGET  
FROM   PROJ_DEPT_BUDGET  
WHERE  FISCAL_YEAR=1994  
GROUP BY PROJ_ID  
HAVING SUM (PROJECTED_BUDGET)>100000;
```

Агрегатные функции

```
select location,  
       count(distinct head_dept)  
from department  
group by location  
       having count(distinct head_dept)>1
```

Объединение

< оператор select >

UNION [ALL | **DISTINCT**]

< оператор select >

Оператор UNION используется для объединения результатов двух или более операторов SELECT, результаты которых должны быть *совместимы для объединения* (количество, порядок и типы соответствующих столбцов должны совпадать)

Результат объединения – множество без повторений

Сортировка

```
ORDER BY <список сортировки>  
[ASC[ENDING] | DESC[ENDING]]  
[NULLS {FIRST|LAST}]
```

Сортировка позволяет задать упорядоченность кортежей в таблице-результате

список сортировки может содержать имена столбцов, выражения, алиасы (2.0) и порядковые номера столбцов (2.0)