SQL – язык структурированных запросов

- Первые языки запросов для РБД появились в 1970х гг.
- SEQUEL – предшественник SQL
- Стандарты: SQL-89, SQL-92, SQL-99, SQL-2003
- Основу языка составляют 2 класса команд: DDL и DML

Язык DDL (Data Definition Language)

$$\left.\begin{array}{l} CREATE \\ ALTER \\ DROP \end{array}\right\} \text{объект имя}_{\text{объекта}} \text{ [дополнительные параметры]}$$

- Примеры:
- DROP TABLE t;
- CREATE TABLE t(n NUMBER,x VARCHAR(50));

Основные объекты БД

1. DATABASE
2. SCHEMA
3. TABLE
4. INDEX
5. VIEW
6. STORED PROCEDURE
7. STORED FUNCTION
8. TRIGGER
9. SEQUENCE
10. USER
11. ROLE
12. LINK
13. SNAPSHOT
14. SYNONYM

Команды DDL для работы с таблицами Создание таблицы

- CREATE TABLE имя_таблицы (список_определений_столбцов [,список_ограничений_на_таблицу])

Определение каждого столбца:

- Имя_столбца тип_столбца [DEFAULT значение_по_умолчанию] [ограничения_на_столбец]

Числовые типы данных

- SMALLINT
- INTEGER (INT)
- BIGINT

- NUMERIC (p,s)
- DECIMAL (p,s)

- FLOAT(p)
- REAL
- DOUBLE PRECISION

- ORACLE: NUMBER[(p[,s])]

Символьные типы данных

- CHAR[ACTER][(размер)]
- VARCHAR (размер)
- NCHAR[(размер)]

National Character

- NVARCHAR (размер)

Типы даты и времени

- DATE
- TIME
- TIMESTAMP

Типы для хранения больших объектов

- BLOB – binary large object
- CLOB – character large object
- NCLOB – national character large object

Ограничения (constraints) столбца

- NOT NULL/NULL
- PRIMARY KEY
- UNIQUE
- REFERENCES имя_главной_таблицы (имя_столбца) [правила_поддержки_ссылочной_целостности]
  - ON DELETE RESTRICT (по умолчанию)
  - ON DELETE CASCADE
  - ON DELETE SET NULL
  - ON DELETE SET DEFAULT
  - Аналогично ON UPDATE …
- CHECK (логическое_выражение)

CREATE TABLE t(c1 NUMBER(8) DEFAULT 0 NOT NULL CONSTRAINT un UNIQUE, c2 VARCHAR(100) NOT NULL))

Ограничения таблицы

- PRIMARY KEY (список_столбцов)
- UNIQUE (список_столбцов)
- FOREIGN KEY имя_внешнего_ключа (список_столбцов) REFERENCES имя_главной_таблицы (список_столбцов) [правила_поддержки_ссылочной_целостности]
  - ON DELETE RESTRICT (используется по умолчанию)
  - ON DELETE CASCADE
  - ON DELETE SET NULL
  - ON DELETE SET DEFAULT
  - Аналогично ON UPDATE …
- CHECK (выражение_с_несколькими_столбцами)

Ограничения таблицы

- CREATE TABLE t(c1 NUMBER(3) NOT NULL,

C2 DATE NOT NULL,

C3 NUMBER(3) NOT NULL,

CONSTRAINT pk_t1 PRIMARY KEY(c1,c2),

CONSTRAINT ck_t1 CHECK(c1+c3<=200))

Удаление и изменение структуры таблиц

- DROP имя_таблицы
- ALTER TABLE имя_таблицы указания_по_изменению
  - ADD [COLUMN] определение_столбца
  - > ALTER TABLE t ADD n NUMBER(4) NOT NULL
  - ADD [CONSTRAINT] ограничение
  - > ALTER TABLE t ADD pk_t PRIMARY KEY (n)
  - DROP COLUMN имя_столбца
  - DROP [CONSTRAINT] ограничение

Синтаксис запроса SELECT

Оператор SELECT имеет следующую структуру:

```
SELECT [DISTINCT | DISTINCTROW | ALL]
      select_expression,...
   FROM table_references
    [WHERE where_definition]
    [GROUP BY {unsigned_integer | col_name | formula}]
    [HAVING where_definition]
    [ORDER BY {unsigned_integer | col_name | formula} [ASC |
DESC], ...]
```

select_expression

- {*|столбец|константа|<выражение>}

* означает, что выбрать необходимо из всех столбцов, кроме того можно указать список из столбцов, констант или выражений для выборки в скобках через запятую.

Пример

Таблица «T»	Запрос	Результат
C1 C2 1 a 2 b	`SELECT * FROM T;`	C1 C2 1 a 2 b
C1 C2 1 a 2 b	`SELECT C1 FROM T;`	C1 1 2
C1 C2 1 a 2 b	`SELECT * FROM T WHERE C1 = 1;`	C1 C2 1 a
C1 C2 1 a 2 b	`SELECT * FROM T ORDER BY C1 DESC;`	C1 C2 2 b 1 a

Предложение WHERE

WHERE

- [NOT] <условие_поиска1> [[AND|OR][NOT] <условие_поиска2>]…

Предложение WHERE

Условие поиска применительно к однотабличным запросам определяется следующим образом:

<условие_поиска> ::=

{ <значение> <операция_сравнения> <значение1>

| <значение> [NOT] BETWEEN <значение1> AND <значение2>

| <значение> [NOT] LIKE 'шаблон' [ESCAPE 'символ_пропуска']

| <значение> [NOT] IN (<значение1> [, <значение2> …])

| <значение> IS [NOT] NULL

| <значение> IS [NOT] DISTINCT FROM <значение1>,

где

<значение> ::= { столбец | константа | <выражение> | функция}.

Примеры

1. SELECT AccountCD, IncomingDate FROM Request WHERE Executed = 0
2. SELECT AccountCD, NachislSum FROM NachislSumma WHERE NachislSum BETWEEN 60 AND 250
3. SELECT Fio FROM Abonent WHERE Fio LIKE 'C%‘
4. SELECT Fio FROM Abonent WHERE Fio LIKE '$%%' ESCAPE '$‘
5. SELECT * FROM Disrepair WHERE FailureCD IN (1,5,12)
6. SELECT AccountCD, IncomingDate FROM Request WHERE ExecutionDate IS NULL

Функции SQL

- *Скалярные*
  - *Строковые*
  - *Числовые*
  - *Даты и времени*
  - *Преобразования типа*
- *Агрегатные*
  - *MAX*
  - *MIN*
  - *AVG*
  - *SUM*
  - *COUNT*

Строковые функции

- SUBSTRING (<строковое_выражение> FROM позиция [FOR длина])
  - SELECT A.AccountCD, SUBSTRING (A.Fio FROM 1 for 3) AS Fio3 FROM Abonent A
- REVERSE (<строковое_ выражение>)
- LEFT (<строковое_выражение>, длина)
- RIGHT (<строковое_выражение>, длина)
  - SELECT * FROM Abonent WHERE RIGHT (Fio, 4) = 'E.B.';.
- REPLACE (<строковое_выражение>, <подстрока>, <строка_для_замены>)

Строковые функции

- TRIM ([[LEADING | TRAILING | BOTH] [<удаляемая_подстрока>] FROM] <строковое_выражение>)
- LPAD(<строковое_выражение>, длина [, <строка_заполнитель>])
- RPAD(<строковое_выражение>, длина [, <строка_заполнитель>])
- UPPER (<значение>)
- LOWER (<значение>)

Строковые функции

- CHAR (n1[,n2,n3..])
- ASCII(str)
- POSITION (<подстрока> IN <строковое_выражение>)
  - SELECT AccountCD, Fio FROM Abonent WHERE POSITION ('У' IN Fio) = 2
- { BIT_LENGTH | CHAR[ACTER]_LENGTH | OCTET_LENGTH } (<строковое_выражение>)

Числовые функции

- RAND() Случайное число от 0 до 1
- ABS (число) Абсолютное значение
- SIGN (число) Знаковая функция (возвращает 1 для
- положительного числа, 0 – для нуля, -1 – для отрицательного числа)
- MOD (делимое, делитель) Остаток от деления

Числовые функции

- LOG (основание, число) Логарифм числа по указанному основанию
- LN (число) Натуральный логарифм числа
- LOG10 (число) Десятичный логарифм числа
- EXP (число) Экспоненциальная функция (e в степени аргумента)
- PI() Константа π = 3.1459...
- POWER (число, степень) Возведение числа в степень

Числовые функции

- SQRT (число) Квадратный корень
- FLOOR (число) Округление до целого числа вниз
- CEIL | CEILING (число) Округление до целого числа вверх
- ROUND (число, точность) Округление до указанного количества знаков после запятой
- TRUNC (число) Целая часть числа
- HASH(<значение>) Хэш-функция (рандомизация значения)

Числовые функции

- SIN (число) Синус (аргумент задается в радианах)
- COS (число) Косинус (угол определяется в радианах,
- результат в диапазоне от -1 до 1)
- TAN (число) Тангенс (аргумент задается в радианах)
- COT (число) Котангенс

Числовые функции

- ASIN (число) Арксинус (число должно быть в диапазоне от -1 до 1, результат от –π/2 до π/2)
- ACOS (число) Арккосинус (число должно быть в диапазоне от -1 до 1, результат от 0 до π)
- ATAN (число) Арктангенс (возвращает результат в диапазоне от –π/2 до π/2)
- SINH (число) Гиперболический синус
- COSH (число) Гиперболический косинус
- TANH (число) Гиперболический тангенс
- ATAN2 (число1, число2) Арктангенс в градусах, вычисляемый как арктангенс результата деления одного тангенса на другой –
- ATAN(число1/число2). Возвращает результат в диапазоне (–π; π]

Числовые функции

- BIN_AND (число [,число...]) Логическое 'И' на всех аргументах
- BIN_OR (число [,число>..]) Логическое 'ИЛИ' на всех аргументах
- BIN_XOR (число [,число...]) Исключающее 'ИЛИ' на всех аргументах
- BIN_SHL (число, число) Двоичный сдвиг влево
- BIN_SHR (число, число) Двоичный сдвиг вправо

Функции даты и времени

- EXTRACT({ DAY | MONTH | YEAR} FROM <значение>)
- SELECT RequestCD, EXTRACT(DAY FROM IncomingDate) AS IncomingDay, EXTRACT (MONTH FROM IncomingDate) AS IncomingMonth, EXTRACT (YEAR FROM IncomingDate) AS IncomingYear FROM Request WHERE EXTRACT (YEAR FROM IncomingDate) IS DISTINCT FROM 2001

REQUESTCD	INCOMINGDAY	INCOMINGMONTH	INCOMINGYEAR
3	28	2	1998
7	20	10	1998
11	12	1	1999
13	4	9	2000
14	4	4	1999
15	20	9	2000
18	28	12	1999

Функции даты и времени

- DATEADD (количество <временной_отрезок> FOR <значение>)
- DATEADD (<временной_отрезок>, количество, <значение>),
- <временной_отрезок> ::= { YEAR | MONTH | DAY | WEEKDAY | HOUR | MINUTE | SECOND };
- <значение> - значение типа дата, время или дата/время, которое увеличивается или уменьшается.
- SELECT IncomingDate, DATEADD (2 WEEKDAY FOR IncomingDate) AS Exec_Limit FROM Request WHERE FailureCD = 1

Функции даты и времени

- DATEDIFF (<временной_отрезок> FROM <значение1> FOR <значение2>)
- DATEDIFF (<временной_отрезок>, <значение1>, <значение2>)
- SELECT RequestCD, DATEDIFF (WEEKDAY FROM IncomingDate FOR ExecutionDate) AS Interval FROM Request WHERE AccountCD = '115705'

Функция преобразования типа

- CAST (<выражение> AS <тип данных>).
- SELECT * FROM Request WHERE IncomingDate > '01.10.2001';
- CAST(DAY || '. ' || MONTH || '. ' || YEAR AS DATE)
- SELECT DISTINCT NachislMonth, NachislYear, CAST('1.' || NachislMonth || '.' || NachislYear as date) as FirstDay FROM NachislSumma WHERE GazServiceCD = 2

Агрегатные функции

- **AVG** – среднее значение в столбце;
- **SUM** – сумма значений в столбце;
- **MAX** – наибольшее значение в столбце;
- **MIN** – наименьшее значение в столбце;
- **COUNT** – количество значений в столбце.
- COUNT (*) служит для подсчета всех без исключения строк в таблице (включая дубликаты).
- Аргументу всех функций, кроме COUNT (*), может предшествовать ключевое слово DISTINCT (различный), указывающее, что избыточные дублирующие значения должны быть исключены перед тем, как будет применяться функция.

Агрегатные функции

- AVG ({[ALL] столбец | DISTINCT столбец} | <выражение>).
- SUM ({[ALL] столбец | DISTINCT столбец})
- MAX ({[ALL] столбец | DISTINCT столбец}),
- MIN ({[ALL] столбец | DISTINCT столбец}).
- COUNT ({ * | [ALL] столбец | DISTINCT столбец}).

Функция MAX

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	200
7	Сидоров	Март	300

SELECT MAX(Сумма) AS mmm FROM ПОСТУПЛЕНИЯ

mmm
300

SELECT MAX(Сумма) AS mmm2 FROM ПОСТУПЛЕНИЯ
WHERE Фамилия='Петров'

mmm2
200

Функция MIN

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	200
7	Сидоров	Март	300

SELECT MIN(Сумма) AS mm3 FROM ПОСТУПЛЕНИЯ

mm3
100

SELECT MAX(Сумма) AS mm4 FROM ПОСТУПЛЕНИЯ
WHERE Фамилия='Петров'

mm4
200

Функция AVG

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	200
7	Сидоров	Март	300

SELECT AVG(Код) AS akod,AVG(Сумма) AS mm5
FROM ПОСТУПЛЕНИЯ

akod	mm5
4	100

SELECT ‘Константа’ AS kk, 50+MAX(Сумма) AS mm6
FROM ПОСТУПЛЕНИЯ WHERE Фамилия=‘Сидоров’

kk	Mm6
Константа	350

Функция SUM

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	200
7	Сидоров	Март	300

SELECT AVG(Код) AS akod,SUM(Сумма)/10 AS mm7
FROM ПОСТУПЛЕНИЯ

akod	Mm7
4	150

SELECT 112233 AS kk, SUM(Сумма) AS mm8
FROM ПОСТУПЛЕНИЯ WHERE Фамилия='Иванов'

kk	Mm8
112233	600

Функция COUNT

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT COUNT(*) AS ccc
FROM ПОСТУПЛЕНИЯ
```

Ccc
7

```
SELECT COUNT(Сумма) AS ccc1
FROM ПОСТУПЛЕНИЯ
```

ccc1
5

Дополнительные возможности вывода в предложении SELECT

- CASE <выражение> {WHEN <значение1> THEN результат1} [{WHEN <значение2> THEN результат2}] ... [ELSE результат(N+1)]

  END.
- CASE {WHEN <условие_поиска1> THEN результат1} [{ WHEN <условие_поиска2> THEN результат2}]...[ELSE результат(N+1)]

  END.

Пример CASE

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия,CASE Сумма
WHEN 100 THEN ‘МАЛО’
WHEN 200 THEN ‘ХОРОШО’
ELSE ‘НЕИЗВЕСТНО’ END AS Сум FROM ПОСТУПЛЕНИЯ
```

Код	Фамилия	Сум
1	Иванов	МАЛО
2	Петров	ХОРОШО
3	Иванов	ХОРОШО
4	Сидоров	ХОРОШО
5	Иванов	НЕИЗВЕСТНО
6	Петров	НЕИЗВЕСТНО
7	Сидоров	НЕИЗВЕСТНО

Пример CASE

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия,CASE
WHEN Сумма=100 THEN ‘МАЛО’
WHEN Сумма=200 THEN ‘ХОРОШО’
ELSE ‘НЕИЗВЕСТНО’ END AS Сум WHERE
Фамилия=‘Иванов’ FROM ПОСТУПЛЕНИЯ
```

Код	Фамилия	Сум
1	Иванов	МАЛО
3	Иванов	ХОРОШО
5	Иванов	НЕИЗВЕСТНО

Функция COALESCE

- COALESCE (<выражение1> , <выражение2> [, <выражение3>]...) .

1. CASE WHEN <выражение1> IS NOT NULL THEN <выражение1> ELSE <выражение2> END;
2. CASE WHEN <выражение1> IS NOT NULL THEN <выражение1> ELSE COALESCE (<выражение2>,..., <выражениеN>) END.

Пример COALESCE

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия,
COALESCE(Сумма,'Ничего') AS Сум
FROM ПОСТУПЛЕНИЯ
```

Код	Фамилия	Сум
1	Иванов	100
2	Петров	200
3	Иванов	200
4	Сидоров	200
5	Иванов	300
6	Петров	Ничего
7	Сидоров	Ничего

Функция NULLIF

- NULLIF (<выражение1>, <выражение2>),
- CASE WHEN <выражение1> = <выражение2> THEN NULL ELSE <выражение1> END.

- SELECT AccountCD, Fio, NULLIF (Phone,'556893') FROM Abonent

Пример NULLIF

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия,
NULLIF(Сумма,200) AS Сум
FROM ПОСТУПЛЕНИЯ
```

Код	Фамилия	Сум
1	Иванов	100
2	Петров	NULL
3	Иванов	NULL
4	Сидоров	NULL
5	Иванов	300
6	Петров	NULL
7	Сидоров	NULL

Сортировка результатов запроса

- <элемент_сортировки> :: ={{[<таблица>.] столбец | порядковый_номер_столбца | псевдоним_столбца | <выражение>}

[ASC[ENDING] | DESC[ENDING]] [NULLS FIRST | NULLS LAST]}… .

- SELECT AccountCd, PaySum FROM PaySumma WHERE AccountCd IN ('136169','005488','443690') ORDER BY AccountCd, PaySum

Сортировка результатов запроса

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия, Месяц
FROM ПОСТУПЛЕНИЯ ORDER BY Код DESC
```

Код	Фамилия	Месяц
7	Сидоров	Март
6	Петров	Март
5	Иванов	Март
4	Сидоров	Февраль
3	Иванов	Февраль
2	Петров	Январь
1	Иванов	Январь

Сортировка результатов запроса

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Код,Фамилия, Месяц
FROM ПОСТУПЛЕНИЯ ORDER BY Фамилия
```

Код	Фамилия	Месяц
5	Иванов	Март
3	Иванов	Февраль
1	Иванов	Январь
6	Петров	Март
2	Петров	Январь
7	Сидоров	Март
4	Сидоров	Февраль

Предложение GROUP BY

- GROUP BY <элемент_группировки1> [, <элемент_группировки2>]..., где
- <элемент_группировки> := {[<таблица>.] столбец | порядковый_номер_столбца | псевдоним_столбца | <выражение>}.
- SELECT NachislYear, AVG(NachislSum) FROM NachislSumma GROUP BY NachislYear

GROUP BY

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

SELECT Фамилия FROM ПОСТУПЛЕНИЯ
GROUP BY Фамилия

Фамилия
Иванов
Петров
Сидоров

GROUP BY

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

SELECT Фамилия,SUM(Сумма) AS s FROM ПОСТУПЛЕНИ
GROUP BY Фамилия

Фамилия	s
Иванов	600
Петров	200
Сидоров	200

SELECT Месяц,SUM(Сумма) AS s FROM ПОСТУПЛЕНИЯ
GROUP BY Месяц

Месяц	s
Январь	300
Февраль	400
Март	300

GROUP BY

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	100
6	Петров	Март	200
7	Сидоров	Март	100

```
SELECT Фамилия, Сумма,
COUNT(Фамилия) AS Количество
FROM ПОСТУПЛЕНИЯ
GROUP BY Фамилия,Сумма
```

Фамилия	Сумма	Количество
Иванов	100	2
Петров	200	2
Иванов	200	1
Сидоров	200	1
Сидоров	100	1

Предложение HAVING

- HAVING, за которым следует <условие_поиска>:
- <условие_поиска> ::= [NOT] <условие_поиска1> [[AND|OR][NOT] <условие_поиска2>]…,
- SELECT AccountCD, COUNT(*), MIN(Incomingdate) FROM Request GROUP BY AccountCD HAVING COUNT(*) > 1;.

HAVING

Таблица ПОСТУПЛЕНИЯ

Код	Фамилия	Месяц	Сумма
1	Иванов	Январь	100
2	Петров	Январь	200
3	Иванов	Февраль	200
4	Сидоров	Февраль	200
5	Иванов	Март	300
6	Петров	Март	NULL
7	Сидоров	Март	NULL

```
SELECT Фамилия,SUM(Сумма) AS s,
COUNT(Сумма) AS Количество
FROM ПОСТУПЛЕНИЯ
GROUP BY Фамилия HAVING  SUM(Сумма)>300
```

Фамилия	s	Количество
Иванов	600	3

```
SELECT Фамилия,SUM(Сумма) AS s,
COUNT(Сумма) AS Количество
FROM ПОСТУПЛЕНИЯ WHERE Сумма>100
GROUP BY Фамилия HAVING  SUM(Сумма)>300
```

Фамилия	s	Количество
Иванов	500	2