Базы данных

Язык запросов SQL. Команда SELECT

Команда SELECT – выборка данных

Общий синтаксис:

SELECT [{ ALL | DISTINCT }] { *список_вывода* | * }
FROM *имя_таблицы1* [*алиас1*] [, *имя_таблицы2* [*алиас2*].,..]
[WHERE *условие_отбора_записей*]
[GROUP BY { *имя_поля* | *выражение* }.,..]
[HAVING *условие_отбора_групп*]
[UNION [ALL] SELECT …]
[ORDER BY *имя_поля1* | *целое* [ASC | DESC]
[, *имя_поля2* | *целое* [ASC | DESC].,..]];

Примеры:

select * from departs;
select name, post from emp;

Формирование списка вывода (проекция)

Общий синтаксис списка вывода:

[{all | distinct}] { * | *выражение1* [*алиас1*] [, *выражение2* [*алиас2*] .,..]}

Список ввода находится между ключевыми словами **SELECT** и **FROM**.

- Вывести все поля всех записей из таблицы Проекты (Project):

  select * from project;

2. Вывести список сотрудников с указанием их должности и № отдела:

   select depno, name, post
   from emp;

3. Вывести список сотрудников с указанием их должности и зарплаты:

   select name 'ФИО', post 'Должность', salary*0.87 'Зарплата'
   from emp;

Установить другой формат вывода даты:

alter session set nls_date_format = 'dd/mm/yyyy';

Формирование списка вывода (проекция)

1. Вывести должности и оклады сотрудников:

```
select  post,  salary
        from  emp;
```

2. Вывести должности и оклады сотрудников **без повторов**:

```
select  DISTINCT  post,  salary
        from  emp;
```

3. Вывести отделы и должности сотрудников **без повторов**:

```
select  DISTINCT  depno,  post
        from  emp;
```

4. Задание: вывести список сотрудников с указанием ФИО, даты рождения и адреса.

```
select name  'ФИО',  born  'Дата рождения',  adr  'Адрес'
        from  emp;
```

Упорядочение результата: order by

1. Вывести данные из таблицы Проекты в порядке даты начала проекта:

```
select *
    from Project
    order by  dbegin;
```

2. Упорядочить список сотрудников по отделам и по ФИО:

```
select depno, name, post
    from emp
    order by  depno,  name;      -- order by 1,2;
```

3. Упорядочить сотрудников по зарплате (от большей к меньшей):

```
select name  'ФИО',  post  'Должность',  salary  'Зарплата'
    from emp
    order by  3  DESC;
```

4. Упорядочить данные об отделах, должностях и зарплатах:

```
select depno  'Номер отдела',  post  'Должность',  salary  'Зарплата'
    from emp
    order by  1,  3  DESC,  2;
```

Выбор данных из таблицы (селекция)

WHERE – содержит условия выбора отдельных записей. Условие является логическим выражением и может принимать одно из 3-х значений:

- TRUE – истина,
- FALSE – ложь,
- NULL – неизвестное, неопределённое значение (интерпретируется как ложь).

Условие формируется путём применения различных операторов и предикатов.

Операторы сравнения:

= равно,	<>, != не равно,	> больше,
>= больше или равно,	<= меньше или равно,	< меньше.

1. Вывести список сотрудников 2-го отдела:

```
select *  from emp
      where depno = 2;
```

2. Вывести список текущих проектов:

```
select *  from  project
      where  dend > sysdate;
```

-- sysdate – функция, возвращающая текущую дату

Логические операторы

Для формирования условий используются следующие логические операторы:

AND – логическое произведение (И),
OR – логическая сумма (ИЛИ),
NOT – отрицание (НЕ).

Операция И:

a	b	a AND b
0	0	0
0	1	0
1	0	0
1	1	1

Операция ИЛИ:

a	b	a OR b
0	0	0
0	1	1
1	0	1
1	1	1

Операция НЕ:

a	NOT a
0	1
1	0

Выбор данных из таблицы по условию

1. Вывести список сотрудников 2-го отдела с зарплатой больше 30000 рублей:

```
select * from emp
    where depno = 2 AND salary > 30000 ;
```

2. Вывести список сотрудников-мужчин, родившихся после 1979 года:

```
select * from emp
    where born > '31/12/1979' AND sex = 'м';
```

3. Вывести список сотрудников 2-го и 5-го отделов:

```
select * from emp
    where depno=2 OR depno = 5;
```

4. Вывести список сотрудников 2-го и 5-го отделов в зарплатой не менее 30000:

```
select * from emp
    where (depno=2 OR depno = 5) AND salary >= 30000 ;
```

5. Вывести список всех сотрудников, кроме сотрудников 2-го и 5-го:

```
select * from emp
    where NOT (depno=2 OR depno = 5);
```

Выбор данных из таблицы по условию

Задание 1: вывести список текущих проектов стоимостью более 2 млн. рублей.

```
select  *
        from  project
        where  dend > sysdate  AND  cost > 2000000;
```

Задание 2: вывести список сотрудников, работающих в должностях 'инженер' и 'ведущий инженер'.

```
select  *
        from  emp
        where  post = 'инженер'  OR  post = 'ведущий инженер' ;
```

Задание 3: вывести список сотрудников, работающих в должности 'охранник', с зарплатой более 20000 рублей.

```
select  *
        from  emp
        where  post = 'охранник'  AND  salary > 20000;
```

Предикаты формирования условия

Предикат вхождения в список значений:

имя_поля **IN (*значение1* [, *значение2*,...])**

выражение **IN (*значение1* [, *значение2*,...])**

Примеры:

- Список сотрудников отделов 5, 8 и 9:

```
select *
    from emp
    where depno  IN ( 5, 8, 9 ) ;
```

- Список сотрудников, работающих в должностях 'инженер' и 'ведущий инженер' :

```
select  *
    from  emp
    where  post  IN ( 'инженер', 'ведущий инженер' );
```

Предикаты формирования условия

Предикат вхождения в диапазон:

имя_поля **BETWEEN** ***минимальное_значение*** **AND** ***максимальное_значение***
выражение **BETWEEN** ***минимальное_значение*** **AND** ***максимальное_значение***

Минимальное значение должно быть меньше либо равно максимальному.

Примеры:

- Список всех сотрудников со 2-го по 5-й отделы:

```
select *
    from emp
    where depno BETWEEN 2 AND 5 ;
```

- Список сотрудников с чистой зарплатой от 20 до 30 тысяч рублей:

```
select *
    from emp
    where salary*0.87 BETWEEN 20000 AND 30000;
```

Предикаты формирования условия

Предикат поиска подстроки: *имя_поля* LIKE '*шаблон*'

Этот предикат применяется только к полям типа CHAR и VARCHAR.
Возможно использование шаблонов:
'_' – один любой символ,
'%' – произвольное количество любых символов (в т.ч., ни одного).

Примеры:

- Список всех сотрудников-экономистов:

```
select *  from emp
      where post  LIKE  '%экономист%' ;
```

- Список всех инженеров-специалистов (кроме просто инженеров):

```
select  *  from  emp
      where  post  LIKE  'инженер_%' ;
```

Экранировать специальное значение символов '_' и '%' можно так:

```
where  <строка>  LIKE  '_#%%'  ESCAPE  '#';
```

Символ экранирования (escape) может быть любым. В примере первый символ % будет искаться как символ, а второй имеет специальное значение.

Предикаты формирования условия

Предикат поиска неопределенного значения:

значение **IS [NOT] NULL**

Если значения является неопределенным (NULL), то предикат IS NULL выдаст истину, а предикат IS NOT NULL – ложь.

Примеры:

- Список всех сотрудников, у которых нет телефона (номер телефона неопределен):

```
select *
    from emp
    where phone IS NULL;
```

- Список все проекты, у которых определена стоимость:

```
select *
    from project
    where cost IS NOT NULL;
```

Использование предикатов

Задание 1: вывести список сотрудников, которых зовут 'ЮРИЙ'.

```
select *
    from emp
    where name LIKE '%ЮРИЙ%';
```

Задание 2: вывести список проектов стоимостью от 1 до 2 млн. рублей.

```
select *
    from project
    where cost BETWEEN 1000000 AND 2000000;
```

Задание 3: вывести список сотрудников, которые являются начальниками отделов.

```
select *
    from emp
    where post LIKE 'нач%отдел%';
```

Агрегирующие функции

COUNT – подсчёт количества строк (значений). Применяется к записям и полям любого типа. Имеет 3 формата вызова:

- **count (*)** – количество строк результата;
- **count (*имя_поля*)** – количество значений указанного поля, не являющихся *NULL*-значениями.
- **count (distinct *имя_поля*)** – количество разных не-NULL значений указанного поля.

MAX, MIN – определяет максимальное (минимальное) значение указанного поля в результирующем множестве. Применяется к полям любого типа.

SUM – определяет арифметическую сумму значений указанного числового поля в результирующем множестве записей.

AVG – определяет среднее арифметическое значений указанного числового поля в результирующем множестве записей. Не учитывает *NULL*-значения, и сумма значений поля делится на количество определённых значений.

Примеры использования агрегирующих функций

1. Вывести максимальную и минимальную стоимость проектов:

```
select  max(cost) "Максимальная цена",  min(cost)  "Минимальная цена"
        from  project;
```

2. Вывести сумму зарплаты сотрудников 8-го отдела:

```
select  sum(salary)
        from  emp
        where  depno = 8;
```

3. Вывести среднюю зарплату сотрудниц предприятия:

```
select  avg(salary)
        from emp
        where  sex = 'Ж';
```

4. Вывести даты начала работы над первым проектом и завершения работы над последним проектом:

```
select   min(dbegin),  max(dend)
        from  project;
```

Примеры использования функции COUNT

1. Вывести количество сотрудников:

```
select  count(*)
        from  emp;
```

2. Вывести количество сотрудников с телефонами:

```
select  count( phone )
        from  emp;
```

3. Вывести количество разных должностей сотрудников:

```
select   count (DISTINCT  post)
        from  emp;
```

4. Задание: вывести количество сотрудников 6-го отдела.

```
select   count(*)
        from  emp
        where  depno = 6;
```

Группировка данных: предложение GROUP BY

Агрегирующие функции обычно используются совместно с предложением ***GROUP BY***.

Например, следующая команда считает количество сотрудников по отделам:

```
select  depno,  count(*)
        from emp
        group by depno;
```

depno	name	…
1	Белов С.В.	
1	Иванова К.Е.	
1	Седов О.Л.	
2	Волков Н.Е.	
2	Рогов И.Л.	
3	Санина В.П.	
3	Дымова С.Т.	
3	Павлов К.Д.	
3	Орлов Т.Ф.	

depno	count(*)
1	3
2	2
3	4

Примеры использования GROUP BY

1. Вывести минимальную и максимальную зарплату в каждом отделе:

```
select depno, MIN(salary) minsal, MAX(salary) maxsal
    from emp
    group by depno;
```

2. Вывести количество разных должностей в каждом отделе:

```
select depno, COUNT(distinct post) cnt
    from emp
    group by depno;
```

3. Посчитать сумму зарплат в каждом отделе:

```
select depno, SUM(salary) allsal
    from emp
    group by depno;
```

4. Посчитать среднюю зарплату по каждой должности:

```
select post, AVG(salary) avgsal
    from emp
    group by post;
```

Использование GROUP BY

Правило использования *GROUP BY* :

В списке вывода при использовании *GROUP BY* могут быть указаны только **функции агрегирования, константы и поля, перечисленные в *GROUP BY*.**

Если включить в список выбора поля, не указанные в *GROUP BY*, то СУБД не будет выполнять такой запрос и выдаст ошибку "нарушение условия группирования" (not a GROUP BY expression).

Например, нельзя получить сведения о том, у каких сотрудников самая высокая зарплата в своём отделе с помощью такого запроса:

```
select  depno,  name,  max(salary) as max_sal
        from  emp
        group by  depno;
```

Этот запрос синтаксически неверен!

depno	name	salary
1	Белов С.В.	58000
1	Иванова К.Е.	28000
1	Седов О.Л.	41000
2	Волков Н.Е.	40000
2	Рогов И.Л.	32000
3	Санина В.П.	**47000**
3	Дымова С.Т.	29000
3	Павлов К.Д.	**47000**
3	Орлов Т.Ф.	30000

Группировка по нескольким полям

1. Сумма зарплаты по отделам и по должностям:

```
select  depno,  post,  count(*),  sum(salary)
        from  emp
        group by  depno,  post;
```

2. Количество мужчин и женщин по отделам:

```
select  depno,  sex,  count(*)
        from  emp
        group by  depno,  sex;
```

Задание: вывести информацию о зарплате и количестве сотрудников, которые получают такую зарплату.

```
select  salary,  count(*)
        from  emp
        group by  salary;
```