# Лабораторная работа №2. Оператор DML SELECT. Простые запросы

## Краткие теоретические данные.

Для выполнения запросов в SQL существует один единственный оператор – SELECT. Синтаксис оператора SELECT имеет следующий вид:

*SELECT [ ALL| DISTINCT] <Список полей>|\**

*FROM <Список таблиц>*

*[WHERE <Предикат-условие выборки или соединения>]*

*[GROUP BY <Список полей результата>]*

*[HAVING <Предикат-условие для группы>]*

*[ORDER BY <Список полей, по которым упорядочить вывод>];*

**SELECT** – ключевое слово, которое сообщает СУБД, что эта команда –запрос. За ним может следовать способ выборки:

- **ALL -** ключевое слово, означает, что в результирующий набор строк включаются **все строки** (не столбцы), удовлетворяющие условиям запроса, т.е. в результирующий набор могут попасть одинаковые строки.

- **DISTINCT -** ключевое слово,означает, что в результирующий набор включаются только различные строки, то есть дубликаты строк результата не включаются в набор.

- список полей – это список перечисленных через запятую столбцов, которые выбираются запросом из таблиц.

- символ **\*** (звездочка) означает, что в результирующий набор включаются все столбцы из исходных таблиц запроса.

В разделе **FROM** задается перечень исходных отношений (таблиц) запроса. В случае, если указано более одного имени таблицы, неявно подразумевается, что над перечисленными таблицами осуществляется операция **декартова произведения**. Разделы SELECT и FROM являются обязательными, все другие разделы являются необязательными.

Допустимо в раздел SELECT включать не только имена полей и \*.

В разделе **WHERE** задаются условия отбора строк результата или  
условия соединения кортежей исходных таблиц, подобно операции  
условного соединения в реляционной алгебре. В выражении условий раздела WHERE могут быть использованы следующие предикаты:

**- Предикаты сравнения (=, <>, >, >=, <, <=)**, которые имеют  
традиционный смысл.

**- Предикат Between A and B** – принимает значения между А и В. Предикат истинен, когда сравниваемое значение попадает в заданный диапазон, включая границы диапазона. Одновременно в стандарте задан и противоположный предикат Not Between A and B, который истинен тогда, когда сравниваемое значение не попадает в заданный интервал, включая его границы.

- **Предикат вхождения в множество IN** (множество) истинен тогда, когда сравниваемое значение входит в множество заданных значений. При этом множество значений может быть задано простым перечислением или встроенным подзапросом. Одновременно существует противоположный предикат NOT IN (множество), который истинен тогда, когда сравниваемое значение не входит в заданное множество.

**- Предикаты сравнения с образцом LIKE и NOT LIKE**. Предикат LIKE требует задания шаблона, с которым сравнивается заданное значение, предикат истинен, если сравниваемое значение соответствует шаблону, и ложен в противном случае. Предикат NOT LIKE имеет противоположный смысл. Шаблон может содержать % (\* для Access) для обозначения любого числа любых символов; \_ (? для Access) для обозначения любого одного символа.

- Предикат со строками. Для работы со строками могут использоваться встроенные функции, например:

DATALENGTH(<**имя атрибута>**) - возвращает длину строки в байтах;

LEN(<**имя атрибута>**) - возвращает длину строки символах;

- **Предикат сравнения с неопределенным значением IS NULL**. Для выявления равенства значения некоторого атрибута неопределенному значению применяют специальные стандартные предикаты: **<имя атрибута> IS NULL** и <**имя атрибута> IS NOT NULL**

**- Предикаты с временными значениями**

В СУБД реализовано достаточно большое количество функций для работы с данными в формате времени и даты. Наиболее часто используемые приведены ниже:

**DAY** ( *date* )  -  целочисленное представление дня указанной даты. Эта функция эквивалентна функции **DATEPART**(*dd*, *date*).

**MONTH** ( *date* )   -  целочисленное представление месяца указанной даты. Эта функция эквивалентна функции **DATEPART**(*mm*, *date*).

**YEAR** ( *date* )   -  целочисленное представление года указанной даты. Эта функция эквивалентна функции **DATEPART**(*yy*, *date*).

**YEAR(GETDATE()) *–*** возвращает текущее значение даты и времени

**DATEDIFF (year , START\_DATE , END\_DATE) -** возвращает разницу во времени (**year - в годах, day-в днях**)

Функции даты и времени могут использоваться в выражениях SELECT при вычислении значений столбцов, в условии выборки сток, сортировки, группировки, а также условиях выборки групп.

|  |  |  |  |
| --- | --- | --- | --- |
| **tab\_1** | | | |
| **Id** | **Department** | **Name** | **Birthday** |
| 1 | Продаж | Иванов | 06-12-1985 |
| 2 | Продаж | Петрова | 04-23-1976 |
| 3 | Кадров | Сидоров | 08-27-1976 |
| 4 | Кадров | Кунин | 05-13-1983 |
| 5 | Технический | Мартынов | 11-11-1984 |

*SELECT department, name, DATEDIFF (year , Birthday , CONVERT (date,GETDATE( ))) as age*

*FROM tab\_1;*

*- возвращает выборку с названием отдела, фамилией сотрудника, возрастом сотрудника*

*SELECT department, name, birthday*

*FROM tab\_1*

*WHERE DATEDIFF (year , Birthday , CONVERT (date,GETDATE( ))) >40 ORDER BY birthday;*

*- возвращает выборку, содержащую сведения о сотрудниках старше 40 лет отсортированную по дате рождения*

*SELECT department, COUNT(\*)*

*FROM tab\_1*

*GROUP BY department*

*HAVING MAX(DATEDIFF (year , Birthday , CONVERT (date,GETDATE( ))))<40;*

*- возвращает выборку с перечнем отделов, сотрудники которых младше 40 лет;*

В части **ORDER BY** задается список полей упорядочения результата, то  
есть список полей, который определяет порядок сортировки в  
результирующем отношении. Например, если первым полем списка будет  
указан Шифр группы, а вторым Фамилия, то в результирующем отношении  
записи сначала будут расположены в порядке возрастания шифра группы, а  
затем в рамках одной группы записи будут отсортированы по фамилии в  
алфавитном порядке. Также можно определить направление порядка: по убыванию **ASC** – сортировка по возрастанию (стоит по умолчанию), **DESC** – сортировка по убыванию

## Задание к лабораторной работе №2

1. Вывести фамилии студентов, в которых присутствуют буквы "б" и/или "о" в любых комбинациях и порядке
2. Вывести сведения о студентах, фамилии которых начинаются с буквы К, при этом они являются иностранцами (NULL в поле отчества)
3. Вывести сведения о студентах, фамилии которых содержат 8 символов и более
4. Вывести сведения о студентах, фамилии которых содержат или больше, или меньше семи символов (но не ровно семь символов), при этом одна из букв фамилии 'a'
5. Найти всех студентов учащихся на ФПМ очном 1 или 2 курсе, отсортировать результаты по полю отчества
6. Найти всех студентов учащихся на ФПК заочном со средним балом успеваемости больше 6, отсортировать результаты по оценки в убывающем порядке
7. Вывеси список всех преподавателей, которые работают на ФПК (не зависимо от формы обучения) отсортировать по алфавиту фамилии в пределах каждой формы обучения
8. Вывести список всех преподавателей, которые работают на ФПМ, первом курсе и читают дисциплины более 100 часов.
9. Вывести список преподавателей иностранцев, работающих в университете более трех лет на текущий момент
10. Вывести информацию о дисциплинах, читаемых для студентов третьего курса ФПМ
11. Вывести информацию о дисциплинах (курс, форма обучения, Фио преподавателя), читаемых на ФПК, число часов по которым больше 100.
12. Вывести информацию о дисциплинах (факультет, курс, форма обучения, Фио преподавателя), которые ведут преподаватели иностранцы.
13. Вывести список преподавателей старше 30 лет на начало текущего года
14. Вывести список преподавателей, от 35 до 40 лет на настоящий момент, отсортировать их по алфавиту фамилии.
15. Вывести список преподавателей, день рождения у которых в октябре, Вывести в порядке возрастания даты рождения.