**Лабораторное занятие №8.**

**Ограничения на число выводимых строк.**

**Лабораторное (практическое) занятие** рассчитано на 2 часа, относится к теме «»

**Формируемые компетенции:** ОК.1-9 ПК 2.1 - 2.4

**Цель:** Изучить используемые в реляционных СУБД операторы изменения данных. Получить навыки работы с SQL-операторами BETWEEN, IN, LIKE, CONTAINING.

**Методическое и техническое обеспечение:** IBM PC-совместимый компьютер, SQLiteStudio.

## Исходные данные

Студент получает индивидуальный вариант исходных данных с кратким описанием предметной области, который используется при выполнении всех лабораторных работ. При этом каждая очередная лабораторная работа является продолжением выполненной ранее и поэтому они должны обязательно выполняться последовательно. Вариант задания приведен в лабораторной работе №1.

**Теоретические сведения**

Число возвращаемых в результате запроса строк может быть ограничено путем использования предложения WHERE, содержащего условия отбора (предикат, рис.2). Условие отбора для отдельных строк может принимать значения true, false или unnown. При этом запрос возвращает в качестве результата только те строки (записи), для которых предикат имеет значение true.

Типы предикатов, используемых в предложении WHERE:

* сравнение с использованием реляционных операторов
  + = равно
  + <> не равно
  + != не равно
  + > больше
  + < меньше
  + >= больше или равно
  + <= меньше или равно
* BETWEEN
* IN
* LIKE
* CONTAINING
* IS NULL
* ANY
* ALL

**Операции сравнения**

Рассмотрим операции сравнения. Реляционные операторы могут использоваться с различными элементами. При этом важно соблюдать следующее правило: *элементы должны иметь сравнимые типы*. Если в базе данных определены домены, то сравниваемые элементы должны относиться к одному домену.

Что же может быть элементом сравнения? Элементом сравнения может выступать:

* значение поля
* литерал
* арифметическое выражение
* агрегирующая функция
* другая встроенная функция
* значение (значения), возвращаемые подзапросом.

При сравнении литералов конечные пробелы игнорируются. Так, предложение WHERE first\_name = 'Петр ' будет иметь тот же результат, что и предложение WHERE first\_name = 'Петр'.

SELECT first\_name, last\_name, dept\_no

FROM employee

WHERE job\_code = "Admin"

получить список сотрудников (и номера их отделов), занимающих должность администраторов

|  |  |  |
| --- | --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** | **DEPT\_NO** |
| Terri | Leev | 000 |
| Ann | Bennet | 120 |
| Sue Anne | O'Brien | 670 |
| Kelly | Brown | 600 |

SELECT first\_name, last\_name, dept\_no,

job\_country

FROM employee

WHERE job\_country <> "USA"

получить список сотрудников, работающих вне США

|  |  |  |  |
| --- | --- | --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** | **DEPT\_NO** | **JOB\_COUNTRY** |
| Ann | Bennet | 120 | England |
| Roger | Reeves | 120 | England |
| Willie | Stansbury | 120 | England |
| Claudia | Sutherland | 140 | Canada |
| Yuki | Ichida | 115 | Japan |
| Takashi | Yamamoto | 115 | Japan |
| Roberto | Ferrari | 125 | Italy |
| Jacques | Glon | 123 | France |
| Pierre | Osborne | 121 | Switzerland |

**BETWEEN**

Предикат BETWEEN задает диапазон значений, для которого выражение принимает значение true. Разрешено также использовать конструкцию NOT BETWEEN.

SELECT first\_name, last\_name, salary

FROM employee

WHERE salary BETWEEN 20000 AND 30000

получить список сотрудников, годовая зарплата которых больше 20000 и меньше 30000

|  |  |  |
| --- | --- | --- |
| FIRST\_NAME | LAST\_NAME | SALARY |
| Ann | Bennet | 22935.00 |
| Kelly | Brown | 27000.00 |

Тот же запрос с использованием операторов сравнения будет выглядеть следующим образом:

SELECT first\_name, last\_name, salary

FROM employee

WHERE salary >= 20000

AND salary <= 30000

получить список сотрудников, годовая зарплата которых больше 20000 и меньше 30000

|  |  |  |
| --- | --- | --- |
| FIRST\_NAME | LAST\_NAME | SALARY |
| Ann | Bennet | 22935.00 |
| Kelly | Brown | 27000.00 |

Запрос с предикатом BETWEEN может иметь следующий вид:

SELECT first\_name, last\_name, salary

FROM employee

WHERE last\_name BETWEEN "Nelson" AND "Osborne"

получить список сотрудников, фамилии которых начинаются с "Nelson" и заканчиваются "Osborne"

|  |  |  |
| --- | --- | --- |
| FIRST\_NAME | LAST\_NAME | SALARY |
| Robert | Nelson | 105900.00 |
| Carol | Nordstrom | 42742.50 |
| Sue Anne | O'Brien | 31275.00 |
| Pierre | Osborne | 110000.00 |

Значения, определяющие нижний и верхний диапазоны, могут не являться реальными величинами из базы данных. И это очень удобно - ведь мы не всегда можем указать точные значения диапазонов!

SELECT first\_name, last\_name, salary

FROM employee

WHERE last\_name BETWEEN "Nel" AND "Osb"

получить список сотрудников, фамилии которых находятся между "Nel" и "Osb"

|  |  |  |
| --- | --- | --- |
| FIRST\_NAME | LAST\_NAME | SALARY |
| Robert | Nelson | 105900.00 |
| Carol | Nordstrom | 42742.50 |
| Sue Anne | O'Brien | 31275.00 |

В данном примере значений "Nel" и "Osb" в базе данных нет. Однако, все сотрудники, входящие в диапазон, в нижней части которого начало фамилий совпадает с "Nel" (т.е. выполняется условие "больше или равно"), а в верхней части фамилия не более "Osb" (т.е. выполняется условие "меньше или равно" - а именно "O", "Os", "Osb"), попадут в выборку. Отметим, что при выборке с использованием предиката BETWEEN поле, на которое накладывается диапазон, считается упорядоченным по возрастанию.

Предикат BETWEEN с отрицанием NOT (NOT BETWEEN) позволяет получить выборку записей, указанные поля которых имеют значения меньше нижней границы и больше верхней границы.

SELECT first\_name, last\_name, hire\_date

FROM employee

WHERE hire\_date NOT BETWEEN "1-JAN-1989" AND "31-DEC-1993"

получить список самых "старых" и самых "молодых" (по времени поступления на работу) сотрудников

|  |  |  |
| --- | --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** | **HIRE\_DATE** |
| Robert | Nelson | 28-DEC-1988 |
| Bruce | Young | 28-DEC-1988 |
| Pierre | Osborne | 3-JAN-1994 |
| John | Montgomery | 30-MAR-1994 |
| Mark | Guckenheimer | 2-MAY-1994 |

**IN**

Предикат IN проверяет, входит ли заданное значение, предшествующее ключевому слову "IN" (например, значение столбца или функция от него) в указанный в скобках список. Если заданное проверяемое значение равно какому-либо элементу в списке, то предикат принимает значение true. Разрешено также использовать конструкцию NOT IN.

SELECT first\_name, last\_name, job\_code

FROM employee

WHERE job\_code IN ("VP", "Admin", "Finan")

получить список сотрудников, занимающих должности "вице-президент", "администратор", "финансовый директор"

|  |  |  |
| --- | --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** | **JOB\_CODE** |
| Robert | Nelson | VP |
| Terri | Lee | Admin |
| Stewart | Hall | Finan |
| Ann | Bennet | Admin |
| Sue Anne | O'Brien | Admin |
| Mary S. | MacDonald | VP |
| Kelly | Brown | Admin |

А вот пример запроса, использующего предикат NOT IN:

SELECT first\_name, last\_name, job\_country

FROM employee

WHERE job\_country NOT IN

("USA", "Japan", "England")

получить список сотрудников, работающих не в США, не в Японии и не в Великобритании

|  |  |  |
| --- | --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** | **JOB\_COUNTRY** |
| Claudia | Sutherland | Canada |
| Roberto | Ferrari | Italy |
| Jacques | Glon | France |
| Pierre | Osborne | Switzerland |

**LIKE**

Предикат LIKE используется только с символьными данными. Он проверяет, соответствует ли данное символьное значение строке с указанной маской. В качестве маски используются все разрешенные символы (с учетом верхнего и нижнего регистров), а также специальные символы:

* % - замещает любое количество символов (в том числе и 0),
* \_ - замещает только один символ.

Разрешено также использовать конструкцию NOT LIKE.

SELECT first\_name, last\_name

FROM employee

WHERE last\_name LIKE "F%"

получить список сотрудников, фамилии которых начинаются с буквы "F"

|  |  |
| --- | --- |
| **FIRST\_NAME** | **LAST\_NAME** |
| Phil | Forest |
| Pete | Fisher |
| Roberto | Ferrari |
|  |  |

SELECT first\_name, last\_name

FROM employee

WHERE first\_name LIKE "%er"

получить список сотрудников, имена которых заканчиваются буквами "er"

FIRST\_NAMERogerRogerWalter

|  |
| --- |
| LAST\_NAME |
| De Souza |
| Reeves |
| Steadman |

А такой запрос позволяет решить проблему произношения (и написания) имени:

SELECT first\_name, last\_name

FROM employee

WHERE first\_name LIKE "Jacq\_es"

найти сотрудника(ов), в имени которого неизвестно произношение буквы перед окончанием "es" FIRST\_NAMEJacques

|  |
| --- |
| LAST\_NAME |
| Glon |

Что делать, если требуется найти строку, которая содержит указанные выше специальные символы ("%", "\_") в качестве информационных символов? Есть выход! Для этого с помощью ключевого слова **ESCAPE**нужно определить так называемый *escape-символ*, который, будучи поставленным перед символом "%" или "\_", укажет, что этот символ является информационным. Escape-символ не может быть символом "\" (обратная косая черта) и, вообще говоря, должен представлять собой символ, никогда не появляющийся в упоминаемом столбце как информационный символ. Часто для этих целей используются символы "@" и "~".

SELECT first\_name, last\_name

FROM employee

WHERE first\_name LIKE "%@\_%" ESCAPE "@"

получить список сотрудников, в имени которых содержится "\_" (знак подчеркивания)

**CONTAINING**

Предикат CONTAINING аналогичен предикату LIKE, за исключением того, что он не чувствителен к регистру букв. Разрешено также использовать конструкцию NOT CONTAINING.

SELECT first\_name, last\_name

FROM employee

WHERE last\_name CONTAINING "ne"

получить список сотрудников, фамилии которых содержат буквы "ne", "Ne", "NE", "nE"

|  |  |
| --- | --- |
| FIRST\_NAME | LAST\_NAME |
| Robert | Nelson |
| Ann | Bennet |
| Pierre | Osborne |

**IS NULL**

В SQL-запросах NULL означает, что значение столбца неизвестно. Поисковые условия, в которых значение столбца сравнивается с NULL, всегда принимают значение unknown (и, соответственно, приводят к ошибке), в противоположность true или false, т.е.

WHERE dept\_no = NULL

или даже

WHERE NULL = NULL.

Предикат IS NULL принимает значение true только тогда, когда выражение слева от ключевых слов "IS NULL" имеет значение null (пусто, не определено). Разрешено также использовать конструкцию IS NOT NULL, которая означает "не пусто", "имеет какое-либо значение".

SELECT department, mngr\_no

FROM department

WHERE mngr\_no IS NULL

получить список отделов, в которых еще не назначены начальники

|  |  |
| --- | --- |
| DEPARTMENT | MNGR\_NO |
| Marketing | <null> |
| Software Products Div. | <null> |
| Software Development | <null> |
| Field Office: Singapore | <null> |

Предикаты EXIST, ANY, ALL, SOME, SINGULAR мы рассмотрим в разделе, рассказывающем о подзапросах.

***Задание***

Работу следует выполнять в следующем порядке:

1. Изучить синтаксис операторов, на каждый оператор написать по два запроса.

***Отчет о выполнении работы***

Отчет о выполнении необходимо оформить на листах формата A4.

Отчет должен содержать описание и результаты работы, представляемые в следующей последовательности:

1. Перечень запросов к базе данных, выполненных при выполнении этой лабораторной работы с краткими пояснениями и данными о количестве строк, выданных при выполнении каждого запроса.

2. Распечатка всех запросов с краткими комментариями и указанием количества возвращенных строк.