# ПОИСК В БАЗАХ ДАННЫХ

*Методические указания к лабораторной работе*

## 1. ЦЕЛЬ РАБОТЫ

Целью работы является приобретение практических навыков решения задач поиска данных с использования технологии окон данных.

## 2. КРАТКАЯ ТЕОРЕТИЧЕСКАЯ СПРАВКА

Поиск данных является одной из главных задач СУБД. Как известно, в реляционных базах данных для поиска данных применяется оператор SELECT SQL. Применение данного оператора возможно как непосредственно, например, с консоли оператора СУБД, так и составе поискового интерфейса СУБД.

Поисковый интерфейс СУБД необходим для организации интерактивных режимов поиска данных в базах данных. Поисковые интерфейсы основаны на применении поисковых форм и специальных технологий.

В рамках одной поисковой технологии, например, технологии окон данных, можно организовать разнообразные поисковые интерфейсы с применением поисковых форм.

Рассмотрим типовые решения по созданию поисковых интерфейсов.

### 2.1. Применение поисковых форм.

**Поисковая форма** – это окно, содержащее поля для ввода образцов , по которым ищутся данные в базе данных. Поисковая форма также содержит органы управления поиском – командные кнопки, а при необходимости - радио кнопки, элементы выбора и т.д. На рис. 1 показан пример поисковой формы.

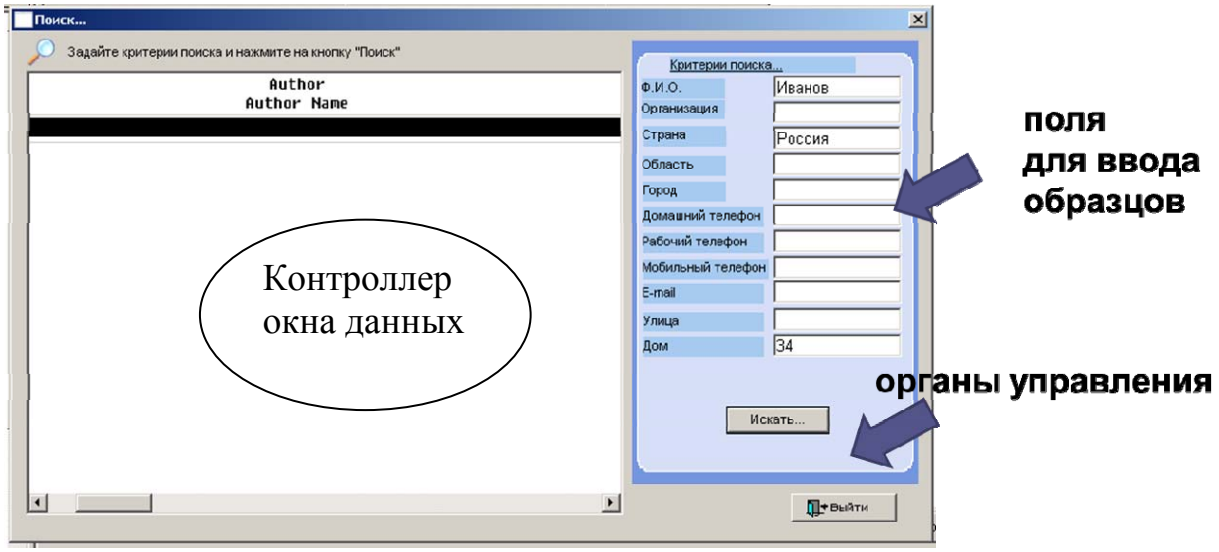


**Рис. 1. Пример поисковой формы**.

Поисковая форма является источником данных для оператора SELECT SQL. Эти данные используются в опции WHERE оператора, образуя выражения вида:

WHERE (<условия>) AND (<условия>) AND …AND(<условия>) (1)

Если хотя бы одно условие в скобках не выполняется, то остальные условия бессмысленно проверять - все выражение примет значение False. Поэтому при использовании поисковых форм следует придерживаться правила:

*вводить в поля поисковых форм только те образцы – атрибуты объекта поиска, значение которых достоверно известно!*

Так в форму на рис. 1 можно вводить данные в любые поля. Поисковый запрос представляет собой сочетание этих данных – образцов согласно выражению (1). Поэтому, если искомый человек по фамилии «Иванов» не живет в доме № 34, то данные о нем не будут найдены.

На рис. 2 показан скрипт обработки события нажатия кнопки «Искать» на рис.1. скрипт описывает формирование оператора SELECT SQL согласно выражению (1). Обратите внимание на то, как задаются условия в опции WHERE. Если в поле поисковой формы ничего не введено, то соответствующее условие в скобке в выражении (1) должно иметь значение True. Это достигается применением символа % для строковых выражений, означающего любой символ. Следующий фрагмент скрипта из рис. 2 демонстрирует применение этого символа:

"WHERE author.author_name LIKE '%"+sle_fio.Text+"%' AND….

Здесь sle_fio.Text - это текст, введенный в поле фамилии формы на рис. 1. Данные в поле author_name таблицы author сравниваются со строкой, формируемой как '%"+sle_fio.Text+"%'. В языке PowerScript, используемом здесь, нельзя использовать подряд кавычки одного типа. Поэтому в строке используются одинарные и двойные кавычки.


**Рис. 2. Скрипт обработки события нажатия кнопки «Искать» на рис.1**

Рассмотрим другие характерные решения в скрипте.

В нем применена функция SyntaxFromSQL(sql, 'Style(Type=Grid) Text()', error_syntaxfromSQL), которая генерирует код окна данных в соответствии с заданным оператором SELECT SQL. Этот оператор является первым аргументом-строкой в вызове функции. Данная строка sql формируется перед обращением к функции.

Второй аргумент функции SyntaxFromSQL – это строка, задающая стиль представления данных в окне данных и тип данных. Используются функции Style() и Text().

Третьим аргументом функции SyntaxFromSQL является строка – сообщение об ошибке.

Код, генерируемый функцией SyntaxFromSQL, далее используется как аргумент в методе Create(syntax, error_create) для создания объекта окна данных для контроллера.

Функция Retrieve(), вызываемая в конце скрипта, работает созданным таким образом объектом окна данных.

## 2.2. Фильтрация в окнах данных

Следующим способом организации поисковых интерфейсов является применение фильтрации в окнах данных. Среди буферов окна данных существует специальный буфер фильтра, куда могут помещаться данные, не удовлетворяющие некоторому выражению - фильтру, задающему условия фильтрации.

Среди функций контроллера окна данных существуют функции SetFilter() и Filter(), которые применяются для фильтрации данных.
Функция SetFilter(f) устанавливает фильтр, задаваемый выражением f, а функция Filter() инициирует выполнение фильтрации.

На рис. 3 показан пример поисковой формы и скрипта фильтрации данных.


**Рис. 3. Поисковая форма и скрипт обработки события нажатия кнопки «Найти»**

В скрипте на рис. 3 формируется строка условия фильтрации, длина которой зависит от того, введены ли данные в поля поисковой формы или нет.

В двух последних строках скрипта задается фильтр и инициируется фильтрация.

## 2.3. Параметрическое извлечение данных

В двух предыдущих примерах применялась функция Retrieve() без аргументов. Эта функция может иметь аргументы, которые задают данные – образцы для поиска. Такие аргументы называются возвращаемыми (Retrieval Arguments) и через них могут передаваться данные в оператор SELECT SQL в его опцию WHERE, задавая этим условия поиска.

Возвращаемые аргументы создаются в объекте окна данных в режиме источника данных, в котором выбираются таблицы и поля для оператора SELECT. Для этого следует выбрать в меню **Design – Retrieval arguments…**

В открывшемся окне указывается имя аргумента и его тип, как показано на рис. 4; они будут являться формальными параметрами для функции Retrieve().

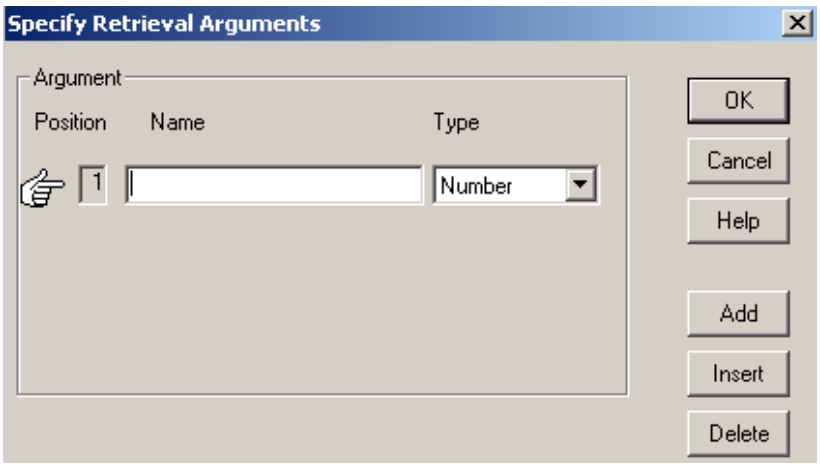


**Рис. 4. Задание возвращаемых аргументов окна данных**

Затем, для включения этих аргументов в оператор SELECT надо открыть вкладку **where** в конструкторе этого оператора **.**

В строке Column указывается столбец таблицы, значение которого будет сравниваться со значением аргумента. В строке Operator выбирается оператор сравнения. В строке Value вставляется имя одного из ранее определенных аргументов. В строке Logical можно указать логический оператор AND или OR, если это сделать будет добавлена еще одна строка в которой, описанным выше способом можно будет определить еще одно условие.

В результате оператор SELECT примет вид:

SELECT {«имя таблицы».«имя столбца», «имя таблицы».«имя столбца»,..}

FROM {«имя таблицы», «имя таблицы»,… }

WHERE {(«имя таблицы».«имя столбца», = **:имя аргумента**)AND/OR..}

Увидеть это можно, открыв вкладку Syntax.

На рис. 5 показан пример задания возвращаемых аргументов в конструкторе оператора SELECT.

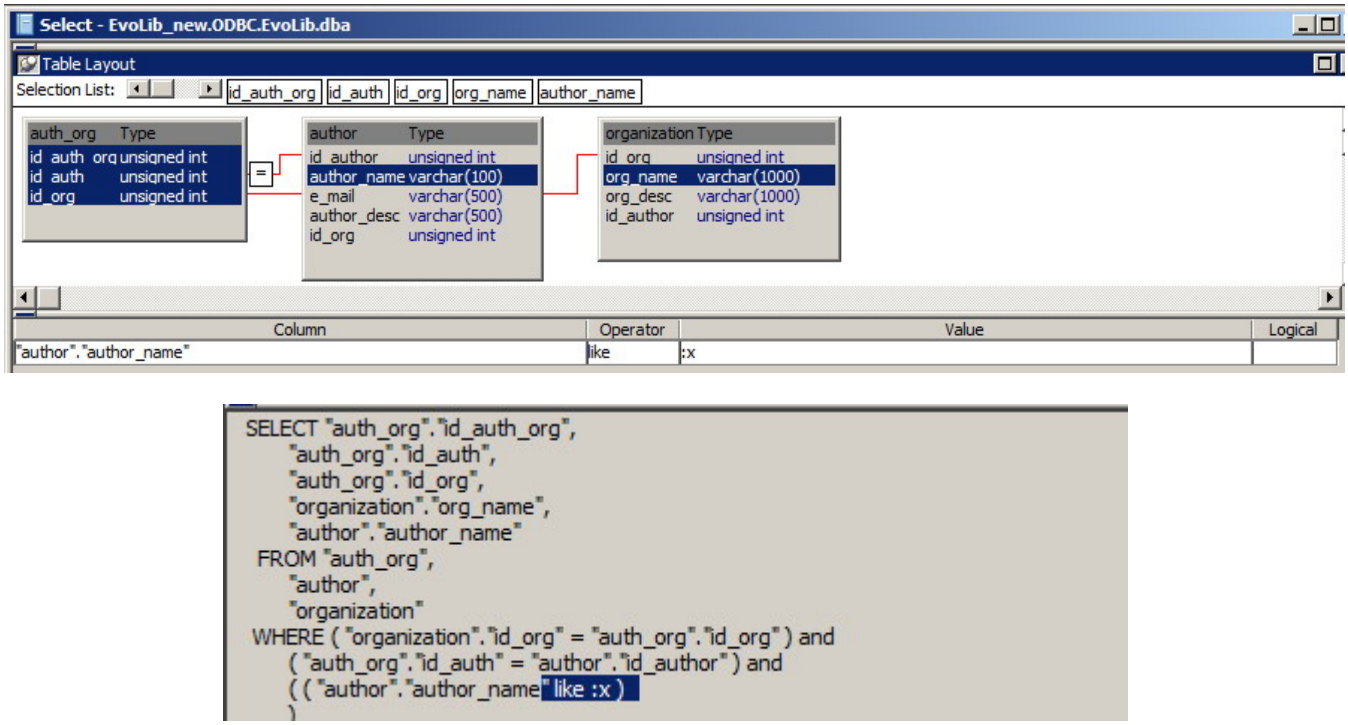


**Рис. 5. Задание возвращаемых аргументов окна данных в конструкторе оператора SELECT.**

Значение возвращаемых аргументов в функцию Retrieve можно передавать как по ссылке, так и непосредственно, например: Retrieve(«Иванов»).

## 3. ОБОРУДОВАНИЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ РАБОТЫ

Работа выполняется на персональном компьютере с операционной системой Windows. В качестве инструментальной СУБД используется SAP-Sybase PowerBuilder версии не ниже 8-й.

## 4. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

4.1. Изучить технологии поиска с применением окон данных.

4.2. На базе данных atm.db решить следующие задачи поиска данных, используя поисковые интерфейсы. Желательно применить все рассмотренные технологии поиска.

4.2.1. Для заданного студента вывести его оценки по заданной дисциплине.
4.2.2. Найти всех студентов, получивших хорошие оценки.
4.2.3. Найти всех студентов с датами рождения в заданном диапазоне.
4.3. Ответить на контрольные вопросы и выполнить упражнения.

## 5. СОДЕРЖАНИЕ ОТЧЕТА

Отчет по работе должен содержать:

- Решения задач п. 4.2 вместе с макетами интерфейсов;
- Результаты выполнения упражнений.

При защите работы студент должен продемонстрировать умение применять технологию окон данных для создания поисковых интерфейсов для таблиц, указанных преподавателем.

## 6. КОНТРОЛЬНЫЕ ВОПРОСЫ И УПРАЖНЕНИЯ

1. Какие характерные объекты принадлежат технологии DataWindow?
2. Почему в скрипте на рис. 2 используется слово LIKE?
3. Есть ли в скрипте на рис. 2 ошибки?
4. Какое имя у контроллера окна данных на рис.1?
5. Имея поисковую форму с полями ввода данных, сколько возвращаемых аргументов нужно задать в соответствующем окне данных?
6. **Упражнение 1**. Постройте оконный интерфейс, в котором будет 1-2 поля для ввода данных и дополнительное поле, в котором будет показан код окна данных, формируемый функцией SyntaxFromSQL для вводимых данных.
7. **Упражнение 2**. Используя решение на рис. 3, модифицируйте скрипт формирования оператора SELECT SQL так, чтобы количество условий поиска в опции WHERE совпадало с числом введенных образцов данных в поисковой форме.

## 7. Литература


1. Электронный ресурс: http://lis.tula.ru/
2. Богатырев М.Ю. разработка и программирование систем управления базами данных. - Тула, изд-во ТулГУ, 2009. - 145 с.
3. Богатырев М.Ю. Введение в систему Power Builder. Методические указания к выполнению лабораторных работ. - Тула, изд-во ТулГУ, 1998. - 36 с.
4. Смит Б. Дж., Шаад Г.У. Power Builder 5.0. Библия разработчика. - К.: Диалектика, 1997. - 544 с.
5. Хайес В.Б. Использование Power Builder 6. - Киев: Вильямс, 1998. - 688 с.