## Методические рекомендации по использованию SQL-ориентированных заданий, выполняемых в среде клиент-серверных систем баз данных

*Щепакина Т.Е.*

*к.п.н., ИИО РАО*

Одной из проблем обучения основам баз данных в школьном курсе информатики является формальный подход к изучению данной. В большинстве существующих СУБД имеются встроенные интерфейсы, в которых пользователь явным образом не использует операции структурированного языка запросов, например, операции SELECT, являющейся основой любой выборки данных.

Для работы с базой данных пользователь может выбрать необходимые команды (подпункты) меню или заполнять поля в формах. По результатам обучения работе с базами данных средствами СУБД MS Access большинство учеников не имеет представления о существовании структурированного языка запросов SQL − стандартного средства доступа к удаленным базам данных, обеспечивающего управление структурой баз данных и манипулирование данными.

Учитель может предложить ученикам сравнить составление SQL-запроса средствами СУБД MS Access и СУБД InterBase.

На рисунке 1 приведено составление простого запроса посредством «Конструктора запросов» и его структуры в виде SQL с помощью графического интерфейса СУБД MS Access.

*Рис. 1. Пример реализации запросов в MS Access (запрос в режиме Конструктора и SQL)*

На рисунке 2 приведен тот же запрос, организованный посредством командного интерфейса в СУБД InterBase.

*Рис.2. Построение SQL-запроса в InterBase Console*

Таким образом, учитель на сравнительном примере демонстрирует ученикам, что синтаксис запроса в режиме SQL достаточно сложный, содержит избыточные квадратные скобки и некоторые лишние параметры, что доказывает невысокую эффективность работы SQL в MS Access. MS Access имеет интерфейс, основанный на меню и формах, что обусловлено необходимостью перестрахования всех неквалифицированных действий пользователей. Поэтому данная СУБД не имеет полных возможностей демонстрации осуществления основных информационных процессов при работе с базами данных.

Целесообразно осуществлять обучение школьников основным операциям применения языка SQL для работы с реляционными базами данных. При этом учителю необходимо акцентировать внимание учеников на использование SQL-запросов на выборку, добавление, изменение, удаление данных и реорганизацию структуры базы данных. Необходимо при использовании SQL-ориентированных заданий при обучении основам баз данных и СУБД учитывать, что запросы на выборку имеют мощный потенциал в процессе обучения информатике, а доступность их использования средствами InterBase делает возможным их применение в процессе изучения школьного курса информатики.

Рассмотрим SQL-операторы ***языка определения данных***и***языка обработки данных***. Учитель может предложить учащимся опорный конспект «Основные операторы языка SQL», приведенный в следующей таблице.

Таблица 1

|  |  |  |
| --- | --- | --- |
| Вид | Оператор языка SQL | Команда пользователя |
| *Язык определения данных**(ЯОД)* | *CREATE TABLE**DROP TABLE**ALTER TABLE**CREATE INDEX**DROP INDEX**ADD PRIMERY KEY**ADD FOREIGN KEY* | Создать таблицуУдалить таблицуИзменить структуру таблицыСоздать индексУдалить индексДобавить первичный ключУстановить связь между таблицами |
| *Язык манипулирования данными**(ЯМД)* | *SELECT**UPDATE**INSERT**DELETE* | Выбрать записьИзменить записьДобавить новую записьУдалить запись |

Рассмотрим примеры использования возможностей операторов ***языка определения данных***, указанных в приведенном выше опорном конспекте. Обучаемому следует начать работу с базой данных с создания структуры таблицы.

*Пример 1.* *Пример 1.* Оператор создания таблицы *Scientist*, имеющей поля: *Code\_Scientist* – идентификатор ученого, *Last\_Name* – фамилия ученого, *Name* – имя ученого, *Second\_Name* – отчество, *Data\_Birth* – дата рождения, *Place\_Birth –* место рождения*; Education* − образование, *Awards* – отличия и вознаграждения ученого, *Publications –* количество публикаций, *Photo -* фото может иметь вид:

|  |  |
| --- | --- |
| /\* Table: Scientist\*/ |  |
| *CREATE TABLE Scientist* | /\*Создать таблицу с именем *Scientist*, содержащую следующие поля: \*/ |
| *(* |  |
| *Code\_Scientist INTEGER NOT NULL,* | /\**Code\_Scientist*, тип целый, не может быть пустым (не иметь никакого значения)\*/ |
| *Last\_Name VARCHAR(25),* | /\* *Last\_Name* буквенного типа длиной 25 символов \*/ |
| *Name VARCHAR(20),* | /\* *Name* буквенного типа длиной 20 символов \*/ |
| *Second\_Name VARCHAR(20),* | /\* *Second\_Name* буквенного типа длиной 20 символов \*/ |
| *Data\_Birth TIMESTAMP,* | /\* *Data\_Birth* типа дата/время\*/ |
| *Place\_Birth VARCHAR(100),* | /\**Place\_Birth* буквенного типа длиной 100 символов\*/ |
| *Education VARCHAR(100),* | /\* *Education* буквенного типа длиной 20 символов \*/ |
| *Scientist\_Grade VARCHAR(60),* | /\* *Scientist\_Grade* буквенного типа длиной 20 символов \*/ |
| *Awards VARCHAR(100 ,* | /\* *Awards* буквенного типа длиной 20 символов \*/ |
| *Publications VARCHAR(50),* | /\* *Publications* буквенного типа длиной 20 символов \*/ |
| *Photo BLOB* | /\* *Photo*, тип – двоичный объект (Binary Large Object)\*/ |
| *PRIMARY KEY (Code\_Scientist)*  | /\*создать первичный ключ по полю *Code\_Scientist* \*/ |
| *);* |  |

*Пример 2.* Рассмотрим использование оператора ALTER для изменения структуры отношений, в данном случае, установление внешнего ключа для таблицы SCIENTIST\_COMP:

*ALTER TABLE SCIENTIST\_COMP* /\*Изменить таблицу SCIENTIST\_COMP \*/

*ADD Foreign KEY (CODE\_SCIENTIST)* /\*добавить внешний ключ по полю CODE\_SCIENTIST \*/

*REFERENCES SCIENTIST (CODE\_SCIENTIST*) /\*значения в данном поле могут быть только такими, которые содержат поле CODE\_SCIENTIST таблицы SCIENTIST \*/ ;

*Пример 3.* Пусть в созданной прежде таблице SCIENTIST следует добавить поле AWARDS, предназначенное для сохранения данных о поощрениях и вознаграждениях данного ученого. Для этого следует записать оператор вида:

*ALTER TABLE SCIENTIST*  /\*Изменить таблицу *SCIENTIST\*/*

*(ADD AWARDS VARCHAR(100)) /\**добавить поле с именем *AWARDS буквенного типа, максимальный размер до 100\*/.*

*Пример 4.* Оператор создание индекса *main\_index* для таблицы *SCIENTIST,* которыйбудет сортировать фамилии в алфавитном порядке и по хронологии рождения, может иметь вид:

*CREATE INDEX main\_index* /\*Создать индекс *main\_index \*/*

*ON SCIENTIST (LAST\_NAME, DATA\_BIRTH DESC)* /\*для таблицы *SCIENTIST* по указанным полям в порядке убывания*\*/.*

Учителю следует подкреплять обучение основам языка структурированных запросов достаточным количеством примеров на применение ***операции выборки*** (SELECT).

*Пример 5.* Получить полную информацию из таблицы Scientist обо всех научных работниках.

Результатом выполнения данного запроса будет копия всей таблицы Scientist. «\*» используется для сокращения списка всех имен столбцов в таблице, на которую делается ссылка в инструкции FROM, в порядке слева направо, так как эти столбцы определены в таблице. Также «\*» удобно использовать в интерактивных запросах. Результат продемонстрирован на рис.3.

*Рис. 3. Результат выполнения запроса (вывод копии таблицы Scientist)*

*Пример 6.* Для каждой ЭВМ получить ее уникальный номер и быстродействие.

Выполнить данное задание можно, составив следующий запрос:

*SELECT Code\_Comp, Speed*

*FROM Comp*

*Пример 7.* Для каждой ЭВМ получить ее уникальный номер и быстродействие.

Выполнить данное задание можно, составив следующий запрос:

*SELECT Code\_Comp, Speed*

*FROM Comp*

*Пример 8.* Получить коды ЭВМ для всех вычислительных машин, являющихся электронными.

На данный вопрос нет короткого ответа, так как характеристика «электронная вычислительная машина» не приводится в данной базе в явном виде. Итак, все вычислительные машины, представленные в базе данных будем считать электронными. Результат можно получить после выполнения следующего запроса:

*SELECT Code\_Comp*

*FROM Comp*

*Пример 9.* Для всех ЭВМ получить их название и площадь, необходимую для установки, вычисленную в см. кв (1 м.кв = 104 см.кв).

На рисунке 2 продемонстрирован результат выполнения данного запроса, в котором спецификация AS AREA\_SM выводит соответствующее имя результатирующего столбца. Таким образом, два столбца результатирующей таблицы будут называться Code\_Comp и Area\_SM соответственно. Если спецификация AS Area\_SM будет опущена, то соответствующий столбец был бы фактически безымянным.

*Рис. 4. Окно запроса к таблице базы данных ComputingHistory*

*Пример 10.* Получить общее число всех ученых, представленных в таблице SCIENTIST.

Для этого можно составить запрос следующего вида:

*SELECT Count (\*) AS N*

*FROM Scientist*

Результатом выполнения предложенного запроса будет таблица с одним столбцом N и одной строкой. Язык SQL поддерживает обычный набор итоговых функций. COUNT (\*) – специальная функция, предназначенная для подсчета всех строк в таблице без единого изъятия дублирования строк.

*Пример 11.* Прокомментировать результаты выполнения запроса на нахождение максимального и минимального быстродействия для ЭВМ, приведенных в базе данных ComputingHistory.

Запрос может быть следующим:

*SELECT MAX (Area) AS MAX\_S,*

*MIN (Area) AS MIN\_S*

*FROM Comp*

*Пример 12.* Из таблицы COMP базы данных известнейших разработок отечественной вычислительной техники ComputingHistory получить все соединения «организация- разработчик – форма представления данных».

Синтаксис запроса и результат его выполнения продемонстрированы на рисунке 3.

*Рис. 5. Результат выполнения запроса на выборку организации-разработчика и формы определения данных*

*Пример 13.* Указать результат следующих SQL-операторов выборки из базы данных ComputingHistory: *SELECT Last\_Name, Awards*

*FROM Scientist*

*WHERE Publications='100 научных работ'*;

*Пример 14.* Указать результат следующих SQL-операторов выборки из базы данных ComputingHistory: *SELECT Last\_Name, Data\_Birth*

*FROM Scientist*

*WHERE Place\_Birth='Россия';*

*Пример 15.* Указать результат следующих SQL-операторов выборки из базы данных ComputingHistory: *SELECT Name, Organization*

*FROM Comp*

*WHERE Year\_Constructing>1952*

*AND Area<100.*

*Пример 16.* Выбрать название и организацию-разработчика для всех ЭВМ, сконструированных после 1957 года.

Результат выполнения задания представлен на рисунке 6.

*Рис. 6. Пример запроса и его выполнения в базе данных ComputingHistory*

 *Пример 17.* Записать оператор SQL для выполнения операции:

а) получения уникального номера, названия ЭВМ и года разработки для всех ЭВМ, сконструированных в институте точной механики и вычислительной техники (ИТМ и ВТ) на основе содержательного наполнения базы данных ComputingHistory (см. рис.7);

б) получения полной информации обо всех ЭВМ, конструирование которых было выполнено в Институте точной механики и вычислительной техники (ИТМ и ВТ) (см. рис.8).

а) б)

*Рис. 7, 8. Результаты выполнения предложенных запросов в консоли InterBase*

*Пример 18.* Выбрать коды и названия ЭВМ, длина машинного слова которых равняется 30 разрядам.

Для выполнения задания можно составить следующий запрос:

*SELECT Code\_Comp, Name*

*FROM Comp*

*WHERE Length\_Words=’30’*

*Пример 19.* Получить все ЭВМ, быстродействие которых находится в диапазоне от 4 000 оп/с до 1 000 000 оп/с включительно.

Структура запроса и результат его выполнения представлены на рисунке 9.

*Рис. 9. Результат выполнения запроса, предложенного в примере №15*

*Пример 20.* Получить название, мощность и быстродействие для всех ЭВМ, которые были разработаны «не в 1961 году» с площадью, необходимой для установки, меньшей 60 кв.г.

При составлении предложенного запроса следует обратить внимание на использование символа <> (не равно). Также важно возможное дублирование аналогичных строк в результате работы оператора SELECT, пока пользователь не запросит это с помощью ключевого слова DISTINCT. Согласно общему правилу относительно уточнения имени в SQL допускаются имена без уточнения в случае, если они не вызовут неоднозначности. Запрос может быть следующим:

*SELECT Name, Power, Speed*

*FROM Comp*

*WHERE Year\_Constructing<>1961*

*AND Area<60*

*Пример 21.* Получить количество ЭВМ, разработкой которых по данным таблицы руководил Лебедев Сергей Алексеевич (уникальный номер в базе данных ComputingHistory - 2).

Запрос может быть следующим:

*SELECT COUNT (Code\_Comp) AS N*

*FROM Scientist\_Comp*

*WHERE Code\_Scientist=2*

*Пример 22.* Получить номера для всех ЭВМ, разработкой которых руководили более одного ученого.

Запрос можно составить следующим образом:

*FROM Scientist\_Comp*

*GROUP BY Code\_omp*

*HAVING COUNT (\*)>1*

При этом инструкция HAVING для групп является тем же самым, что инструкция WHERE для строк.

Операция выборки в SQL − это табличное выражение, которое может быть достаточно сложным. Выражение выборки может содержать несколько компонентов: инструкции SELECT, FROM, WHERE, Group By, HAVING. Учитель может привести пример выборки данных на получение полной информации обо всех ЭВМ средствами SQL и проиллюстрировать выполнение запроса в консоли InterBase.

*Рис. 10. Результат выполнения предложенного запроса в консоли InterBase*

Понимание работы с базами данных формируется на этапе создания запросов с вложенными подзапросами, что отображает нелинейный принцип обработки информации, когда выборка выполняется по нескольким таблицам. Неподдельный интерес у учеников могут вызвать задания на нахождение неочевидного и непредсказуемого заранее ответа. Целесообразным будет приведение учителем примеров составления запросов с использованием подзапросов.

*Пример 1.* Для получения имен ученых, принимавших участие в разработке БЭСМ-6 с уникальным номером 6 можно составить запрос:

*SELECT DISTINCT Last\_Name*

*FROM Science*

*WHERE Code\_Science IN*

*(SELECT Code\_Science*

*FROM Science\_Comp*

*WHERE Code\_Comp=6)*

При этом подзапрос *SELECT Code\_Scientist FROM Scientist\_Comp WHERE Code\_Comp=6* используется для представления множества значений, поиск которых осуществляется с помощью инструкции *IN Condition* (*Condition* – «условие») система вычисляет полностью запрос, выполнив вычисление подзапроса. Предложенное задание можно решить также с помощью операции соединения:

*SELECT s.Last\_name*

*FROM Scientist s, Science\_comp sc*

*WHERE s.code\_Scientist=sc.code\_Scientist*

*AND sc.code\_Comp=6*

*Пример 2.* Запрос на получение фамилий всех ученых, которые не принимали участие в разработке БЭСМ-6, занесенной в базы данных ComputingHistory под уникальным номером 6, будет иметь следующую структуру:

*SELECT DISTINCT Last\_Name*

*FROM Scientist*

*WHERE Code\_Scientist NOT IN*

*(SELECT Code\_Scientist*

*FROM Scientist\_Comp*

*WHERE Code\_Comp=6)*

*Пример 3.* Для получения названия ЭВМ, разработкой которых руководил ученый с уникальным номером 1, следует записать следующий запрос:

*SELECT c.Name*

*FROM Comp c, Scientist\_Comp sc*

*WHERE c.Code\_Comp=sc.Code\_Comp AND*

*Sc.Code\_Science=1*

Для получения названия ЭВМ, разработкой которых руководил Глушков В.М., запрос в консоли InterBase будет иметь синтаксис, представленный на рисунке 4.

*Рис. 11. SQL-запрос и результат его выполнения в консоли InterBase*

*Пример 4.* Получить номера всех ЭВМ, которые имеют быстродействие больше 5 000 операций в секунду или были разработаны под руководством И.С. Брука (в таблице SCIENTIST занесен под уникальным номером 12), или те и другие. Для получения данной информации можно составить такой запрос:

*SELECT Code\_Comp*

*FROM Comp*

*WHERE Speed>5000*

*UNION*

*SELECT Code\_Comp*

*FROM Scientist\_Comp*

*WHERE Code\_Scientist=12*

Следует заметить, что лишние повторяемые строки всегда выключаются из результата безусловных операторов UNION, INTERSECT или EXCEPT (оператор EXCEPT в языке SQL есть аналогом операции MINUS реляционной алгебры).

Работа с языком SQL определяется моделированием анализа и обработки данных, получения новой информации на основе запросов, не очевидной заранее. В качестве примера приведем SQL-запрос на получение номеров для всех ЭВМ, разработкой которых руководили более чем один ученый из приведенных в таблице ученых (Scientist).

*Пример 5.* Получитьномера для всех ЭВМ, разработкой которых руководили более чем один ученый из приведенных в таблице ученых (Scientist).

*SELECT Code\_Comp; /\** выбери значение *Code\_Comp* \*/

*FROM Scientist\_Comp; /\**изтаблицы «*Scientist\_Comp*»\*/

*GROUP BY Code\_Comp;* /\*сгруппированные по совпадающим значениям поля Code\_Scientist \*/

*HAVING COUNT (Code\_Scientist*)>1/\*которые встречаются в таблице чаще 1 раза \*/

Результатом данной выборки будет перечень номеров ЭВМ, имеющих более одного из авторов, перечисленных в таблице Scientist. Эта информация не была доступна в явном виде даже на этапе заполнения базы. Для каждого оператора в предложенном и в последующих примерах запишем в качестве примечания команду на русском языке.

*Пример 6.* Получить номера ЭВМ, ученые-разработчики которых обучались в Московском энергетическом институте.

*SELECT Code\_Comp /\**Выбрать значение в поле *Code\_Comp \*/*

*FROM Scientist \_Comp /\**из таблицы Scientist \_Comp *\*/*

*WHERE code\_Scientist in* /\*где значения поля code\_Scientist находятся в следующем множестве значений: \*/

*(SELECT code\_Scientist /\**Выбрать значение поля *code\_Scientist \*/*

*FROM Scientist /\**из таблицы *Scientist \*/*

*WHERE Education*=’ Московский энергетический институт ’)/\*для записей, в которых поле Education равняется указанному значению \*/.

При работе с клиент-серверными технологиями учителям необходимо ознакомить обучаемых со стандартами системы «клиент-сервер», уделить внимание ***операциям языка обработки данных*** DML, не использующим курсор, среди которых можно выделить INSERT, UPDATE, DELETE. Учителю следует подкреплять обучение основам языка структурированных запросов достаточным количеством примеров на применение операций языка обработки данных при работе с клиент-серверными технологиями.

*Пример 1.* Вставьте строку в таблицу COMP, содержащую информацию про *ЭВМ МИР-2.*

Для того, чтобы вставить строку в таблицу COMP, содержащую информацию про *ЭВМ МИР-2,* необходимо составить запрос:

*INSERT* /\*Вставить\*/

*INTO COMP (CODE\_COMP, NAME, ORGANIZATION, YEAR\_CONSTRUCTING, DATA\_FORM, SPEED, POWER, SQUARES)* /\*в таблицу *COMP* новую запись с  указанием порядка следования полей ... \*/

*VALUES (20, ‘МЕР-2’, ‘Институт кибернетики АН УССР’, 1969, ‘с плавающей запятой’, 20 000, 5, 20)* /\*с перечнем значений полей для новой записи ...\*/.

*Пример 2.* Вставьте несколько строк в дополнительно созданную таблицу TEMP, которые будут содержать данные об уникальном номере, названии и быстродействии для ЭВМ, площадь которых, необходимая для установки, меньше чем 80 кв. м.

Для того, чтобы вставить необходимые строки в дополнительно созданную таблицу TEMP, составим следующий SQL-запрос:

*INSERT* /\*Вставить\*/

*INTO TEMP (CODE\_COMP, NAME, POWER) /\**во временно созданную таблицу TEMP со следующими полями ....*\*/*

*SELECT CODE\_COMP, NAME, POWER /\**с выборкой следующих полей ... *\*/*

*FROM COMP /\** из таблицы COMP\*/

*WHERE SQUARES<80/ \**при условии, что площадь, необходимая для установки, меньше 80 кв.м *\*/*

*Пример 3.* Замените значение поля «Форма данных» для ЭВМ «МЭСМ» на значение «с фиксированной запятой».

Запрос на замену значения формы данных для ЭВМ МЭСМ на значение «с фиксированной запятой» может быть следующим:

*UPDATE COMP*  /\*Обновить данные в таблице *COMP* \*/

*SET DATA\_FORM=’с фиксированной запятой’ /\**Установить в поле *DATA\_FORM* указанное значение ....*\*/*

*WHERE NAME=’МЕСМ’ /\**Для тех записей, где поле *NAME* содержит значение «МЭСМ»*\*/*

*Пример 4.* Удалите все строки таблицы SCIENTIST\_COMP, относящиеся к ЭВМ с уникальным номером 9.

SQL-запрос в данном случае будет иметь вид:

*DELETE /\**Изъять *\*/*

*FROM SCIENTIST\_COMP /\**из таблицы *SCIENTIST\_COMP* записи *\*/*

*WHERE CODE\_COMP=9 /\**в которых поле *CODE\_COMP* содержит значение, равное «9»*\*/*

Следует объяснить учащимся, что удалять, не проверив, нужны ли записи, сведения из базы данных, не желательно без предварительного согласования целесообразности совершения данного действия.

Вышеперечисленные примеры на составление SQL-запросов подтверждают, что общепринятый термин «язык запросов» не совсем достаточно отображает рассматриваемые понятия, поскольку слово «запрос» подразумевает под собой лишь выборку, в то время как с применением этого языка выполняются также операции обновления, вставки и удаления, а также множество других [32], [33].

Таким образом, учителю необходимо в процессе обучения основам баз данных подбирать и формулировать задания и задачи на составление запросов, в том числе сложных. Следует обратить внимание учащихся, что одно и то же задание можно выполнить с помощью разных запросов. Это важно, т.к. если база данных большая, то неграмотно составленный запрос будет обрабатываться в несколько раз медленнее. Понять принципы работы с операцией SELECT и всеми инструкциями можно лишь при практическом выполнении определенных запросов, именно поэтому учителю целесообразно подготавливать достаточное количество заданий с целью выработки у учащихся знаний, умений и навыков работы с реляционными базами данных.

**---------------------------------------------------------------------------------------------------------------------**

[Р](http://rpio.ru/)[оссийский портал информатизации образования](http://portalsga.ru/) [содержит: законодательные и нормативные правовые акты государственного регулирования информатизации образования, федеральные и региональные программы информатизации сферы образования, понятийный аппарат информатизации образования, библиографию по проблемам информатизации образования, по учебникам дисциплин цикла Информатика, научно-популярные, документальные видео материалы и фильмы, периодические издания по информатизации образования и многое другое.](http://portalsga.ru)

