

Работа с базами данных средствами JDBC

Гутников С.Е.

Введение в технологию БД

Эффективное владение информацией в современном мире это необходимый элемент развития в самых разных сферах промышленности, торговли и бизнеса.

Одна из основных областей использования вычислительной техники в настоящее время это **информационные системы (ИС)** предназначенные для обработки информации.

ИС обычно работают с большими объёмами достаточно сложно структурированной информации.

Введение в технологию БД

В качестве примеров ИС можно привести банковские системы, системы резервирования железнодорожных или авиационных билетов, и т.п.

К функциям ИС относят:

- хранение, ввод, редактирование и контроль правильности информации;
- просмотр и поиск информации;
- выборка информации по заданным критериям;
- составление отчётов в различной форме

Введение в технологию БД

Целью любой ИС является обработка данных об объектах реального мира с учётом связей между объектами. В теории баз данных (БД) данные обычно называют **атрибутами**, а объекты – **сущностями**.

БД – это именованная совокупность взаимосвязанных данных находящихся под управлением системы управления базой данных.

Введение в технологию БД

Система управления БД (СУБД) – это совокупность программных и языковых средств, необходимых для создания БД, поддержания БД в актуальном состоянии и организации поиска необходимой информации.

По методам обработки различают централизованные и распределенные БД.

Распределённые БД состоят из нескольких частей, располагающихся на нескольких серверах сети.

Введение в технологию БД

Централизованная БД хранится на одном сервере и пользователи локальной сети могут одновременно обращаться к информации хранящейся в БД.

Кроме разделения БД по методам обработки их различают по используемой модели данных.

В настоящее время, на практике используются три основных модели данных:

Введение в технологию БД

Иерархическая модель данных. БД состоит из набора элементов упорядоченного в виде дерева – исходные элементы порождают другие элементы, при этом каждый потомок имеет только один порождающий элемент.

Сетевая модель данных – это расширение иерархической модели, отличается тем, что каждый потомок может иметь более одного родителя.

Введение в технологию БД

Реляционная модель данных. Основная идея этой модели – представить любой набор данных в виде двумерной таблицы. В простейшем случае это единственная двумерная таблица, но обычно это несколько двумерных таблиц и взаимосвязи между ними.

Наша цель – изучение технологии JDBC, эта технология работает только с так называемыми реляционными СУБД, которые используют реляционную модель данных.

Введение в технологию БД

Основоположником теории реляционных БД считается сотрудник IBM, доктор А. Codd (Кодд), который в 1970 г. предложил использовать для обработки данных аппарат логики предикатов и теории множеств. Он доказал, что любой набор данных можно представить в виде двумерных таблиц особого вида, известных в математике как «отношение». От слова «relation» («отношение») и происходит название модели данных.

Введение в технологию БД

Термин реляционной теории «отношение» имеет более широкий смысл чем таблица, обозначая некоторый абстрактный объект.

В следующей таблице перечислены основные термины используемые при работе с БД.

Введение в технологию БД

Теория БД	Практика
Отношение (Relation)	Таблица (Table)
Кортеж (Tuple)	Запись (Record) – строка таблицы
Атрибут (Attribute)	Поле (Field) – столбец таблицы
Домен (Domain)	Количество полей (столбцов таблицы)
Степень отношения	Количество записей (строк таблицы)

Таблица 15.1 Терминология баз данных

Введение в технологию БД

Далее, познакомимся с операциями, используемыми в реляционной алгебре.

В реляционной алгебре (РА) и операнды и результаты всех действий являются отношениями. Несколько упрощая, будем отождествлять отношения с таблицами. Порядок размещения записей в таблицах-операндах никак не учитывается при выполнении операций, а порядок записей в таблицах-результатах будем считать произвольным.

Введение в технологию БД

1. Операция проектирования: `proj`

В операции участвует единственный операнд - таблица либо с исходными данными, либо полученная в результате операций РА. Определим операцию проектирования следующим образом:

```
proj "список имён столбцов" ТАБЛИЦА
```

В списке имён столбцов должны упоминаться только те имена, которые входят в состав записей таблицы.

Введение в технологию БД

Результат операции `proj` формируется следующим образом:

- а) из таблицы удаляются все столбцы имена которых отсутствуют в списке
- б) из полученной на предыдущем шаге совокупности записей удаляются повторяющиеся, так, что итоговая таблица не содержит одинаковых записей

Введение в технологию БД

2. Операция выбора: `sel`

В операции участвует единственный операнд - таблица либо с исходными данными, либо полученная в результате операций РА.

Определим операцию выбора следующим образом:

```
sel  "условие" ТАБЛИЦА
```

Введение в технологию БД

Итогом операции выбора является совокупность строк таблицы, которые удовлетворяют заданному условию.

Условие может включать операции сравнения и логические операции над одним или несколькими столбцами.

Введение в технологию БД

3. Операция соединение: `join`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию соединения следующим образом:

```
join ТАБЛИЦА1 ТАБЛИЦА2
```

Введение в технологию БД

Соединением двух таблиц является таблица, которая включает в себя слияние каждой строки ТАБЛИЦА1 с каждой строкой ТАБЛИЦА2 у которых совпадают значения всех общих полей, причём сами эти общие поля помещаются лишь однажды.

Введение в технологию БД

4. Операция объединение: `union`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию объединение следующим образом:

```
union ТАБЛИЦА1 ТАБЛИЦА2
```

Введение в технологию БД

Операция объединения определена только над таблицами одного и того же типа, т.е. с одинаковыми именами столбцов.

Результатом является таблица включающая в себя совокупность строк входящих хотя бы в одну из таблиц операндов.

Введение в технологию БД

5. Операция пересечение: `intersection`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию пересечение следующим образом:

```
intersection ТАБЛИЦА1 ТАБЛИЦА2
```

Введение в технологию БД

Операция определена только над таблицами одного и того же типа, т.е. для таблиц с одинаковыми именами столбцов.

Результатом является совокупность строк входящих в каждую таблицу-операнд.

Введение в технологию БД

6. Операция вычитание: `difference`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию вычитание следующим образом:

```
difference ТАБЛИЦА1 ТАБЛИЦА2
```

Введение в технологию БД

Операция определена только над таблицами одного и того же типа, т.е. для таблиц с одинаковыми именами столбцов.

Результатом является совокупность строк входящих в ТАБЛИЦА1 и отсутствующих в ТАБЛИЦА2.

.

Введение в технологию БД

7. Операция деление: `division`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию деление следующим образом:

```
division ТАБЛИЦА1 ТАБЛИЦА2
```

.

Введение в технологию БД

Операция требует, чтобы каждое поле **ТАБЛИЦА2** имело те же имя и тип, что и одно из полей **ТАБЛИЦА1**.

Результатом является таблица, поля которой содержатся в **ТАБЛИЦА1**, но отсутствуют в **ТАБЛИЦА2**. Запись (строка) включается в результат только при условии, что в **ТАБЛИЦА1** она сцеплена с каждой записью из **ТАБЛИЦА2**

.

Введение в технологию БД

8. Операция умножение: `product`

В операции участвует два операнда – таблицы, каждая таблица либо с исходными данными, либо получена в результате операций РА.

Определим операцию умножение следующим образом:

```
product ТАБЛИЦА1 ТАБЛИЦА2.
```

Введение в технологию БД

Результатом является таблица содержащая совокупность записей представляющих конкатенацию каждой строки ТАБЛИЦА1 с каждой строкой ТАБЛИЦА2.

В результирующей таблице каждый столбец сохраняет прежнее имя, дополненное через точку именем таблицы из которой взят: ИмяТаблицы.ИмяСтолбца

Введение в технологию БД

Запросы к БД, как правило, необходимо оптимизировать, выбирая такую последовательность операций, которая минимизирует количество вычислений.

При составлении выражений на языке РА никак не учитываются вычислительные затраты, связанные с выдачей ответов на конкретный вопрос.

Предположим, например, что W это одно из возможных значений поля P таблицы

ТАБЛИЦА1 .

Рассмотрим следующие два запроса:

Введение в технологию БД

```
join (sel P==W ТАБЛИЦА1) ТАБЛИЦА2
```

и

```
sel P==W (join ТАБЛИЦА1 ТАБЛИЦА2)
```

Эти запросы дадут одинаковый результат, но второй запрос может оказаться менее эффективным если результат соединения будет содержать больше записей чем в ТАБЛИЦА1 .

Введение в технологию БД

По своим возможностям многие современные СУБД предоставляют языки запросов существенно превосходящие возможности языка РА. Например, арифметические расчёты невозможно выполнить пользуясь РА. Современные языки запросов также включают в себя операции занесения записей в таблицу, их удаление, корректировку и т.п.

Введение в технологию БД

За рамками нашего обсуждения остались вопросы проектирования связанные с возможной избыточностью данных и понятия нормализации отношений.

Кратко, нормализация – это процесс представления данных в виде простых двумерных таблиц, который позволяет устранить дублирование этих данных и обеспечивает непротиворечивость хранимой в БД информации.

Вопросы нормализации подробно рассматриваются в спецкурсах посвящённых технологиям БД.

Введение в технологию БД

Далее рассмотрим задачу, которую будем использовать при изучении JDBC.

При рассмотрении примера нам понадобится понятие первичного ключа:

В некоторых случаях таблица имеет одно или несколько полей, значения которых однозначно идентифицируют каждую запись в таблице. Такое поле (или комбинация полей) называется первичным ключом (primary key).

Введение в технологию БД

Задача.

Поликлиника дантиста оказывает платные услуги пациентам. Периодически возникает необходимость ответов на следующие вопросы:

Введение в технологию БД

- 1) Сколько пациентов обслужено за месяц/год?
- 2) Какова сумма оказанных услуг за месяц/год?
- 3) Какова сумма неоплаченных счетов за месяц/год?

Необходимо также получать следующую информацию:

- 4) Список пациентов, не обращавшихся за помощью больше года
- 5) Список услуг, которые не были востребованы больше года

Введение в технологию БД

Проанализируем условие этой задачи.

Очевидно, что нам необходимо хранить информацию:

- общие данные пациентов,
- данные об обращении пациентов и об оказанных услугах,
- данные о стоимости услуг и об их оплате

Введение в технологию БД

В принципе, как один из возможных вариантов, все эти данные можно свести в одну единственную таблицу.

Учтем то, что имя, пол и возраст пациента являются необходимой для поликлиники информацией.

Далее, чтобы мы смогли ответить на поставленные в условии вопросы и решить задачу, каждая строка такой таблицы должна содержать хотя бы следующие поля:

Введение в технологию БД

Date –	дата оказания услуги
Name –	наименование услуги
Price –	цена услуги
FIO –	полное имя пациента
BirthDay –	дата рождения пациента
Sex –	пол пациента
isPaied –	оплачена ли услуга

Введение в технологию БД

Далее, если представить, что удобно иметь в распоряжении отдельную таблицу пациентов и таблицу оказываемых услуг, приходим к следующей структуре БД для нашей задачи.

БД состоит из 3-х связанных таблиц:

1) *Service* – таблица содержит список услуг поликлиники с ценами. Поля (столбцы) таблицы:

Введение в технологию БД

`Name` – тип поля «текст», содержит наименование услуги (первичный ключ). Это поле связано с одноименным полем таблицы `HealthCare`.

`Price` – тип поля «деньги», содержит стоимость услуги.

2) `Patients` – таблица содержит список пациентов с паспортными данными. Поля (столбцы) таблицы:

Введение в технологию БД

`FIO` - тип поля «текст», содержит фамилию имя и отчество пациента (первичный ключ). Это поле связано с одноименным полем таблицы `HealthCare`.

`Sex` – тип поля «число», содержит пол пациента, 0 == женский, иначе мужской.

`BirthDate` – тип поля «дата», содержит дату рождения пациента.

Введение в технологию БД

3) `HealthCare` – таблица содержит список оказанных услуг, - проведенное лечение пациентов. Поля (столбцы) таблицы:

`FIO` - тип поля «текст», содержит фамилию имя и отчество пациента. Это поле связано с одноименным полем таблицы `Patients`.

`Name` – тип поля «текст», содержит наименование услуги. Это поле связано с одноименным полем таблицы `Service`.

Введение в технологию БД

`Date` – тип поля «дата», содержит дату оказания услуги.

`isPaied` – тип поля «логический», содержит значение «истина», если пациент оплатил услугу.

Структура описанной БД, показана на следующей схеме:

Введение в технологию БД

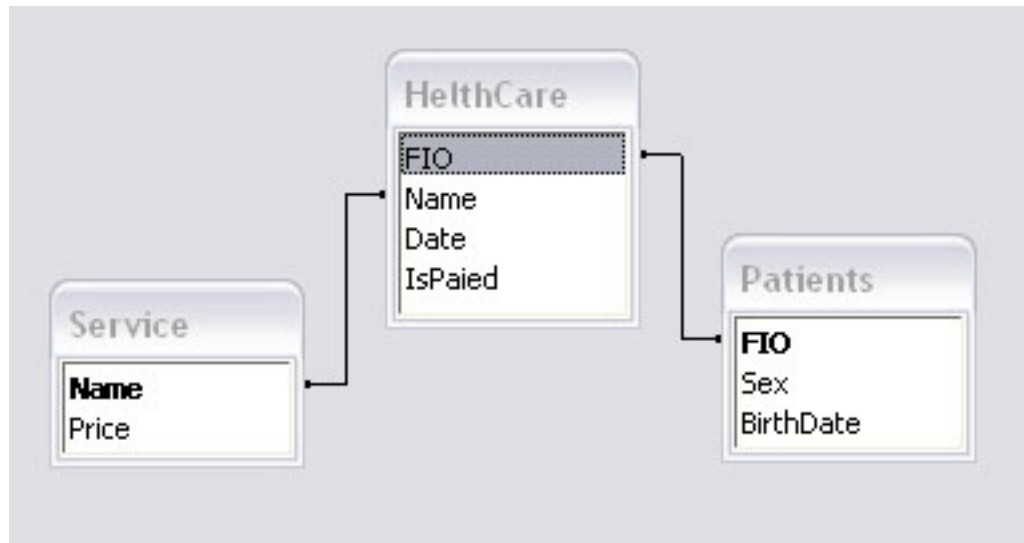


Рисунок 1. Структура БД поликлиники дантиста

Введение в технологию БД

Анализируем дальше нашу задачу.

Очевидно, что при ответе на вопросы 2 и 3, поставленные в задаче, нам будет недостаточно операций РА – потребуется найти сумму числового поля по таблице

Аналогично, при ответе на вопрос 1, нам также будет недостаточно операций РА – нам понадобится определить количество строк в таблице.

Введение в технологию БД

Сформулируем эти дополнительные требования к нашей системе:

Кроме операций РА, необходимо реализовать ещё следующие операции:

а) `Length ТАБЛИЦА`

- подсчёт количества строк в таблице

б) `Sum ТАБЛИЦА.АТРИБУТ`

- сумма по всей таблице указанного числового поля (столбца)

Введение в технологию БД

Кроме этих операций и операций РА, в любую ИС обычно включают методы для:

- создания новой БД
- архивирования существующей БД
- средства просмотра выбранной таблицы
- методы поиска в БД по заданному атрибуту
- методы редактирования БД и т.д.

Введение в технологию БД

В заключение анализа нашей задачи запишем алгоритмы формирования ответов, поставленных в условии.

(Имена таблиц зелёным цветом, имена полей фиолетовым)

Введение в технологию БД

1) Сколько пациентов обслужено за месяц/год?

```
Length (
    proj FIO (
        sel (      Date в указанном
диапазоне)      HealthCare) )
```

Введение в технологию БД

2) Какова сумма оказанных услуг за месяц/год?

```
Sum ( (  
    sel (Date в указанном диапазоне)  
        HealthCare)  
    join Service) .Price
```

Введение в технологию БД

3) Какова сумма неоплаченных счетов за месяц/год?

```
Sum (  
    sel (Date в указанном диапазоне AND  
        isPaied == false) Healthcare)  
join Service) .Price
```

Введение в технологию БД

4) Список пациентов, не обращавшихся за помощью больше года

difference

```
(proj FIO Patients)
```

```
(proj FIO (
```

```
    sel Date в диапазоне последнего года)
```

```
    HealthCare ))
```

Введение в технологию БД

5) Список услуг, которые не были востребованы больше года

difference

```
(proj Name Service )
```

```
(proj Name (
```

```
    sel (Date в диапазоне последнего года)
```

```
        Healthcare ) )
```

Введение в язык запросов SQL

SQL предназначен для использования в качестве языка взаимодействия с реляционной БД.

Язык SQL не содержит тех средств, которые необходимы для разработки законченных приложений.

Стандарт SQL различает три прикладные реализации языка запросов, которые имеют свои особенности. Всегда нужно понимать с какой реализацией вы работаете.

Введение в язык запросов SQL

- 1) Интерактивный или автономный SQL – пользователи непосредственно извлекают информацию из БД или записывают ее в БД.
- 2) Статический SQL – фиксированный, записанный заранее а не генерируемый во время выполнения программы код SQL, который обычно используется в приложениях. Статический SQL по способу включения в приложение делится ещё на встроенный SQL и модульный SQL.

Введение в язык запросов SQL

3) Динамический SQL – код SQL сгенерированный приложением во время исполнения, он заменяет статический тогда, когда код SQL не известен во время разработки приложения и зависит, например, от ответов пользователя во время выполнения программы.

Требования к этим трём реализациям SQL различны и это отражается на языковых конструкциях используемых для каждой из них.

Введение в язык запросов SQL

Эта особенность стандарта SQL, а также то, что различные производители СУБД представляют различные наборы имён типов данных для представления столбцов таблиц часто не даёт возможности создавать независимые от СУБД приложения.

Далее, рассмотрим некоторые вызовы SQL на примере создания БД стоматологической поликлиники из рассмотренного ранее примера. В качестве СУБД будем использовать Apache Derby.

Введение в язык запросов SQL и Apache Derby

Подготовка к работе Apache Derby

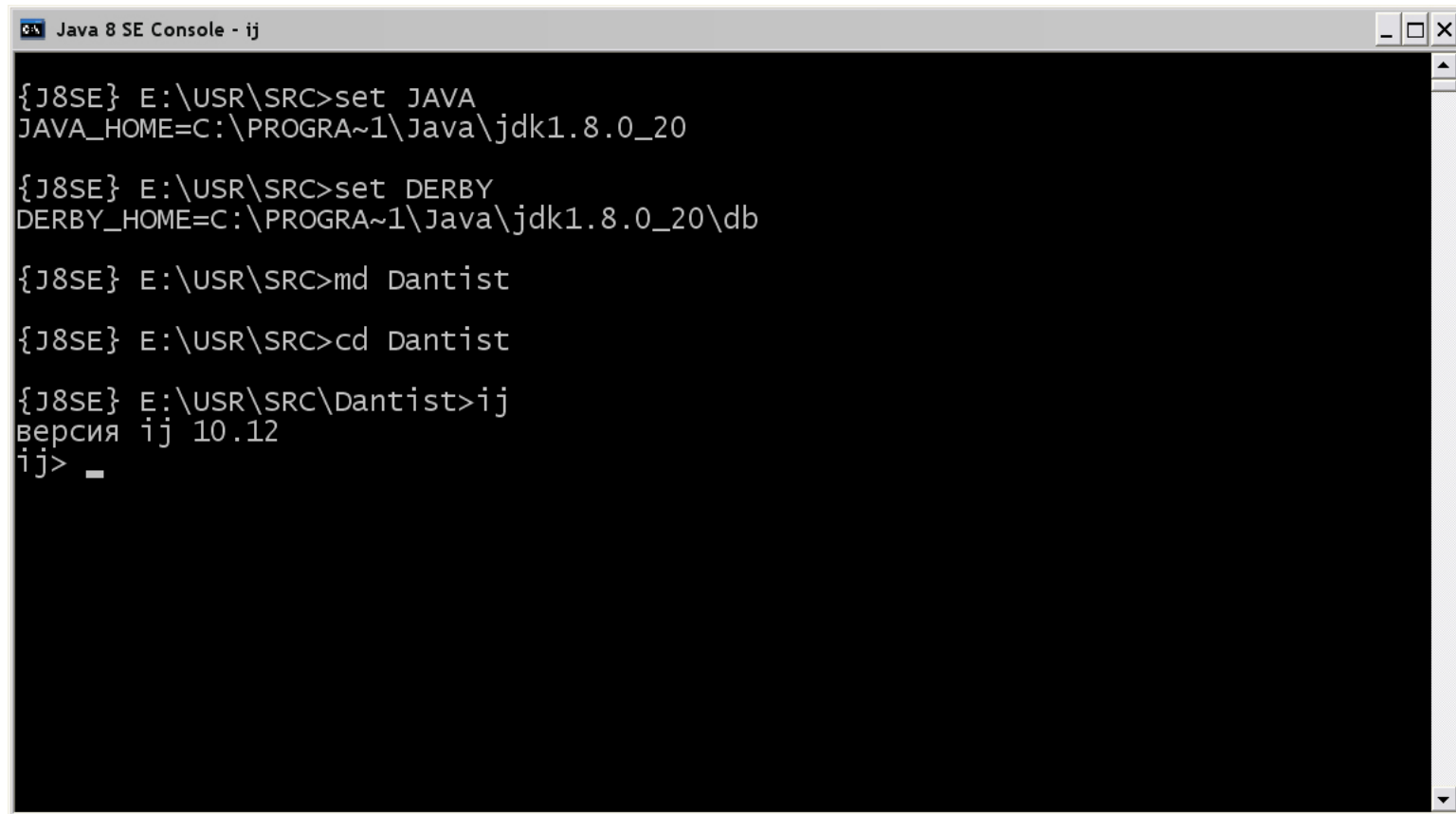
Версия СУБД поставляется с JDK и располагается в каталоге `<jdk>\db`, её рекомендуется обновить с сайта производителя.

Для работы СУБД необходимо установить переменные среды:

JAVA_HOME и
DERBY_HOME

Создадим БД для нашей задачи в каталоге Dantist.

Введение в язык запросов SQL и Apache Derby



```
Java 8 SE Console - ij
{J8SE} E:\USR\SRC>set JAVA
JAVA_HOME=C:\PROGRA~1\Java\jdk1.8.0_20

{J8SE} E:\USR\SRC>set DERBY
DERBY_HOME=C:\PROGRA~1\Java\jdk1.8.0_20\db

{J8SE} E:\USR\SRC>md Dantist

{J8SE} E:\USR\SRC>cd Dantist

{J8SE} E:\USR\SRC\Dantist>ij
версия ij 10.12
ij> _
```

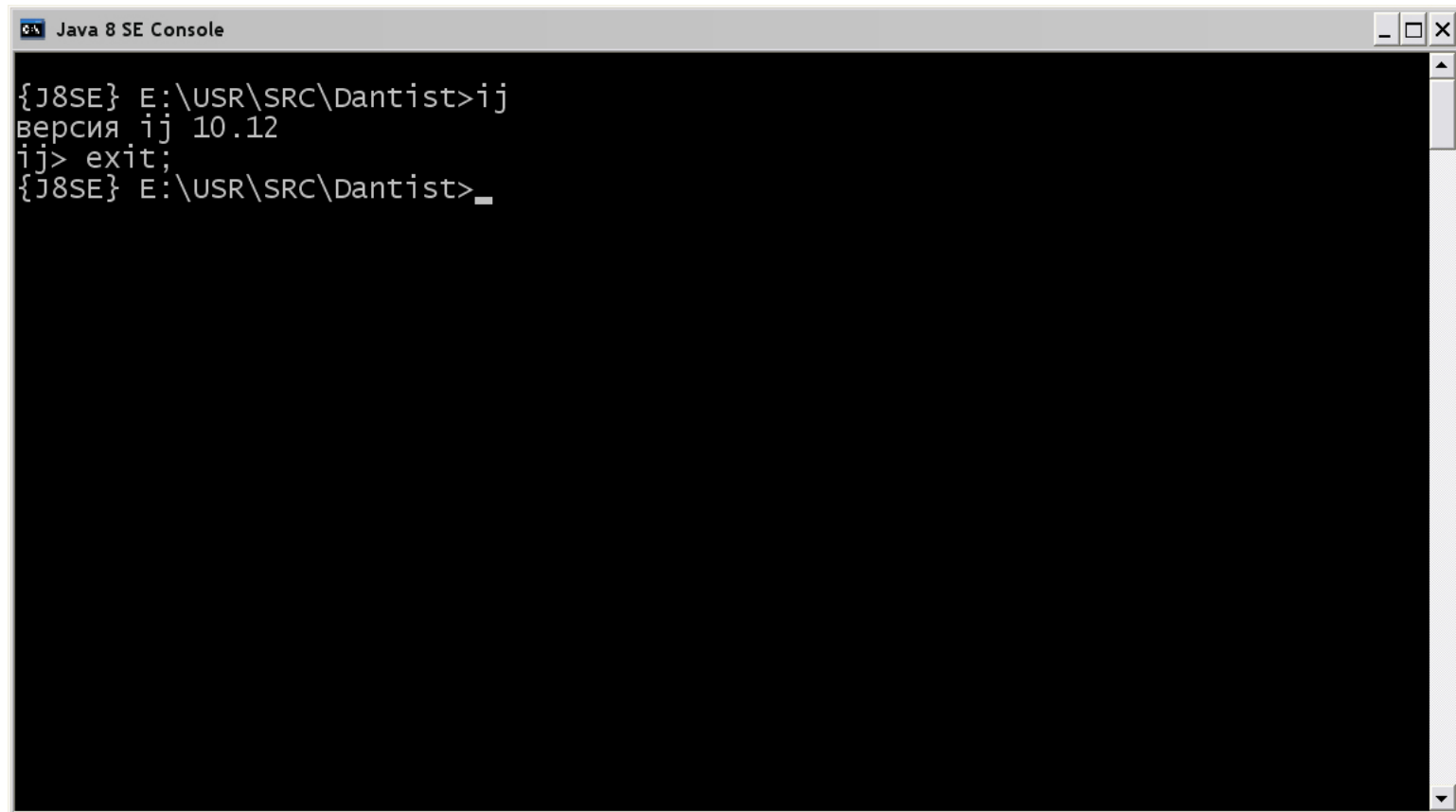
Введение в язык запросов SQL и Apache Derby

Командный процессор СУБД называется `ij`, он принимает команды SQL.

Для завершения его работы используется команда `exit`.

Команды можно вводить в одну или несколько строк, команда начинает выполняться после ввода точки с запятой.

Введение в язык запросов SQL и Apache Derby



```
{J8SE} E:\USR\SRC\Dantist>ij
версия ij 10.12
ij> exit;
{J8SE} E:\USR\SRC\Dantist>_
```

Введение в язык запросов SQL и Apache Derby

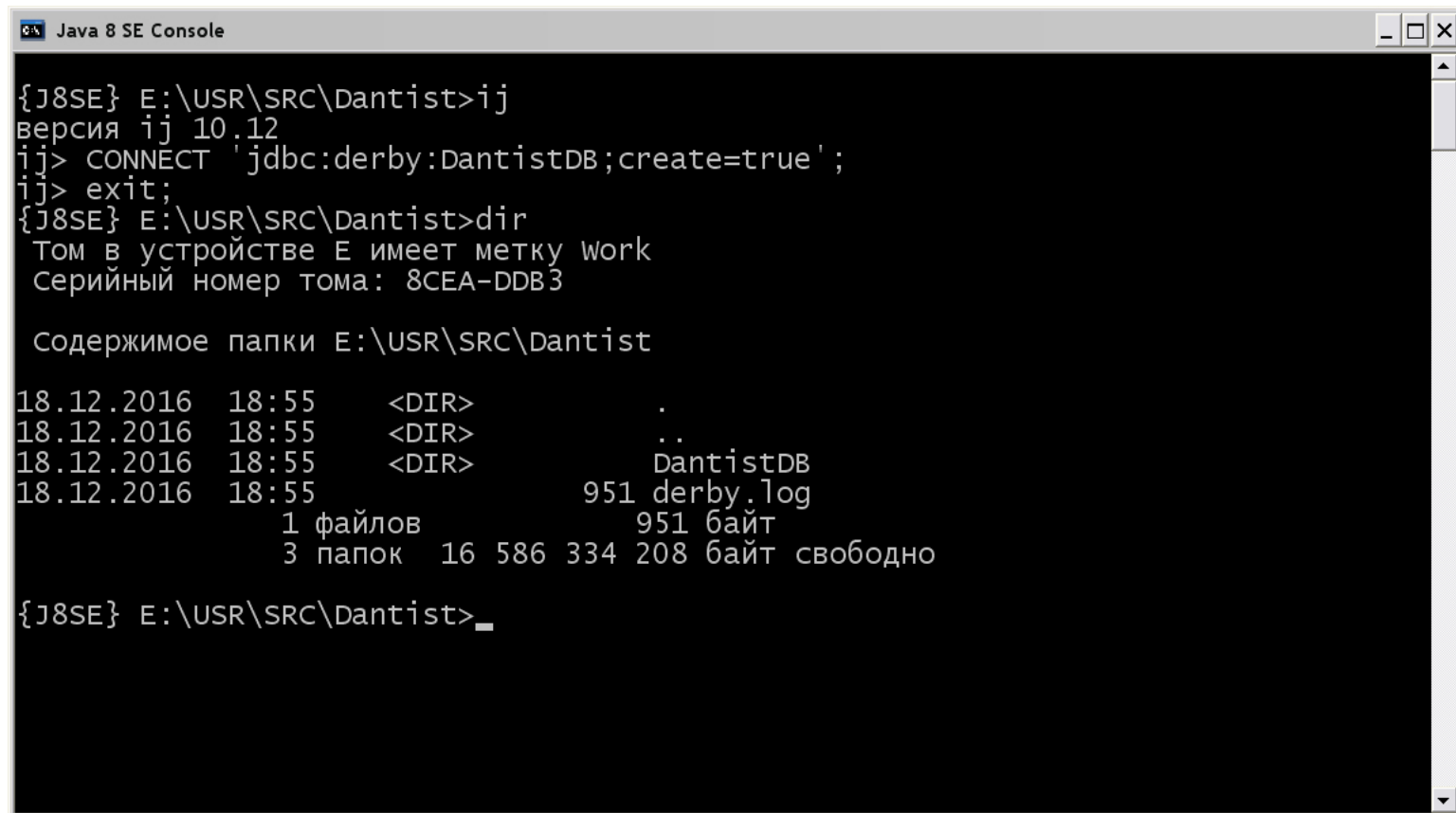
Для создания новой БД или подключения к уже существующей используется команда SQL CONNECT.

Создадим БД для нашей задачи следующей командой:

```
CONNECT 'jdbc:derby:DantistDB;create=true';
```

Строка подключения строится аналогично URL.

Введение в язык запросов SQL и Apache Derby



```
{J8SE} E:\USR\SRC\Dantist>ij
версия ij 10.12
ij> CONNECT 'jdbc:derby:DantistDB;create=true';
ij> exit;
{J8SE} E:\USR\SRC\Dantist>dir
Том в устройстве E имеет метку work
Серийный номер тома: 8CEA-DDB3

Содержимое папки E:\USR\SRC\Dantist

18.12.2016  18:55    <DIR>          .
18.12.2016  18:55    <DIR>          ..
18.12.2016  18:55    <DIR>          DantistDB
18.12.2016  18:55                951 derby.log
                1 файлов                951 байт
                3 папок   16 586 334 208 байт свободно

{J8SE} E:\USR\SRC\Dantist>_
```

Введение в язык запросов SQL и Apache Derby

В примере подключаемся по протоколу JDBC через драйвер `derby` к базе данных с именем `DantistDB`, модификатор `create=true` требует создания новой БД.

После ввода этой команды в текущем каталоге будет создан каталог с именем `DantistDB`.

Для подключения к существующей СУБД модификатор `create` не указываем.

Введение в язык запросов SQL и Apache Derby

Создадим таблицу Service для нашей задачи и заполним её тестовыми данными.

Используем команды SQL:

для создания таблицы – CREATE TABLE

для заполнения таблицы – INSERT

Считаем, что поле Service.Name должно быть уникальным (помечаем это поле как PRIMARY KEY). При попытке добавления строк с неуникальным именем получаем сообщение об ошибке.

Введение в язык запросов SQL и Apache Derby

```
Java 8 SE Console - ij
ij> CONNECT 'jdbc:derby:DantistDB';
ij> CREATE TABLE Service ( Name VARCHAR(32) PRIMARY KEY, Price REAL );
Вставлено/обновлено/удалено строк: 0
ij> INSERT INTO Service VALUES( 'Обследование', 9.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Удаление зуба', 99.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Лечение кариеса', 49.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Лечение кариеса, осложнённое', 89.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Лечение канала зуба', 89.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Отбеливание зубов', 199.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Рентгендиагностика', 3.99 );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Service VALUES( 'Рентгендиагностика', 3.99 );
ОШИБКА 23505: обработка оператора прекращена, поскольку при этом был бы создан дубликат значения ключа в ограничении уникального или главного ключа или в уникальном индексе, указанном при помощи 'SQL161218210014370', заданном в 'SERVICE'.
ij> _
```

Введение в язык запросов SQL и Apache Derby

Проверим введенные данные. Для распечатки таблицы воспользуемся командой SQL – SELECT.

Введение в язык запросов SQL и Apache Derby

```

ij> SELECT * FROM Service ORDER BY Name;
NAME | PRICE
-----|-----
лечение канала зуба | 89.99
лечение кариеса | 49.99
лечение кариеса, осложнённое | 89.99
обследование | 9.99
отбеливание зубов | 199.99
Рентгендиагностика | 3.99
удаление зуба | 99.99

выбрано строк: 7
ij> SELECT * FROM Service ORDER BY Price;
NAME | PRICE
-----|-----
Рентгендиагностика | 3.99
обследование | 9.99
лечение кариеса | 49.99
лечение канала зуба | 89.99
лечение кариеса, осложнённое | 89.99
удаление зуба | 99.99
отбеливание зубов | 199.99

выбрано строк: 7
ij>

```

Введение в язык запросов SQL и Apache Derby

Аналогично создадим таблицу Patients и заполним её тестовыми данными.

Считаем, что поле Patients.FIO должно быть уникальным (помечаем это поле как PRIMARY KEY). При попытке добавления строк с неуникальным именем можем получить сообщение об ошибке.

Введение в язык запросов SQL и Apache Derby

```
Java 8 SE Console - ij
ij> CREATE TABLE Patients (
> FIO VARCHAR(32) PRIMARY KEY, Sex INT, BirthDate DATE );
Вставлено/обновлено/удалено строк: 0
ij> INSERT INTO Patients VALUES( 'Прудник Пётр Иванович', 1, '01.02.1980' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Свирид Дмитрий Алексеевич', 1, '06.05.1977' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Русакова Екатерина Сергеевна', 0, '07.03.1991' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Касаржевская Алла Марковна', 0, '11.10.1983' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Гореликов Тимофей Дмитриевич', 1, '10.01.1990' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Серединский Михаил Алексеевич', 1, '21.06.1979' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Бруцкий Владислав Витальевич', 1, '31.08.1988' );
Вставлена/обновлена/удалена 1 строка
ij> INSERT INTO Patients VALUES( 'Максимов Сергей Леонидович', 1, '27.02.1989' );
;
```

Введение в язык запросов SQL и Apache Derby

Проверим введенные данные. Для распечатки таблицы воспользуемся командой SQL – SELECT.

Введение в язык запросов SQL и Apache Derby

```
Java 8 SE Console - ij
ij> SELECT * from Patients ORDER BY FIO;
FIO                                |SEX      |BIRTHDATE
-----
Бруцкий Владислав Витальевич      |1        |1988-08-31
Гореликов Тимофей Дмитриевич      |1        |1990-01-10
Касаржевская Алла Марковна        |0        |1983-10-11
Максимов Сергей Леонидович        |1        |1989-02-27
Прудник Пётр Иванович             |1        |1980-02-01
Русакова Екатерина Сергеевна      |0        |1991-03-07
Свирид Дмитрий Алексеевич         |1        |1977-05-06
Серединский Михаил Алексеевич     |1        |1979-06-21

выбрано строк: 8
ij>
```


Введение в язык запросов SQL и Apache Derby

Создадим таблицу HealthCare и заполним её тестовыми данными при помощи следующих команд.

```
CREATE TABLE HealthCare (  
    FIO VARCHAR(32) REFERENCES Patients (FIO),  
    Name VARCHAR(32) REFERENCES Service (Name),  
    Date DATE,  
    isPaied BOOLEAN );
```

Введение в язык запросов SQL и Apache Derby

```
INSERT INTO HealthCare VALUES(  
    'Касаржевская Алла Марковна',  
    'Обследование', '01.11.2016',  
    true );
```

```
INSERT INTO HealthCare VALUES(  
    'Бруцкий Владислав Витальевич',  
    'Удаление зуба',  
    '01.11.2016',  
    true );
```

...

Введение в язык запросов SQL и Apache Derby

Распечатаем созданную таблицу. Чтобы поместилось в окне пропустим столбец `isPaied`.

Введение в язык запросов SQL и Apache Derby

```
Java 8 SE Console - ij
ij> SELECT FIO,Name,Date FROM HealthCare ORDER BY Date;
FIO                                | NAME                                | DATE
-----|-----|-----
Бруцкий Владислав Витальевич      | Удаление зуба                      | 2016-11-01
Касаржевская Алла Марковна        | Обследование                      | 2016-11-01
Свирид Дмитрий Алексеевич         | Обследование                      | 2016-11-02
Прудник Пётр Иванович             | Удаление зуба                      | 2016-11-03
Бруцкий Владислав Витальевич      | Лечение кариеса                    | 2016-11-03
Касаржевская Алла Марковна        | Лечение кариеса                    | 2016-11-11
Бруцкий Владислав Витальевич      | Рентгендиагностика                | 2016-11-12
Касаржевская Алла Марковна        | Лечение кариеса                    | 2016-11-13
Бруцкий Владислав Витальевич      | Лечение кариеса, осложнённое      | 2016-11-13
Гореликов Тимофей Дмитриевич      | Обследование                      | 2016-11-21
Русакова Екатерина Сергеевна      | Лечение кариеса                    | 2016-11-22
Максимов Сергей Леонидович        | Лечение кариеса                    | 2016-11-23
Гореликов Тимофей Дмитриевич      | Отбеливание зубов                 | 2016-11-23
Касаржевская Алла Марковна        | Лечение канала зуба                | 2016-11-24
Прудник Пётр Иванович             | Лечение кариеса, осложнённое      | 2016-11-29

выбрано строк: 15
ij>
```

Введение в язык запросов SQL и Apache Derby

Запишем теперь алгоритмы формирования ответов, поставленных в условии на языке запросов SQL и проверим на наших тестовых таблицах.

Таблицы содержат данные только за месяц, поэтому в наших запросах будем использовать более короткие промежутки времени (что легко исправить в реальной задаче, когда нам необходимы данные например за год)

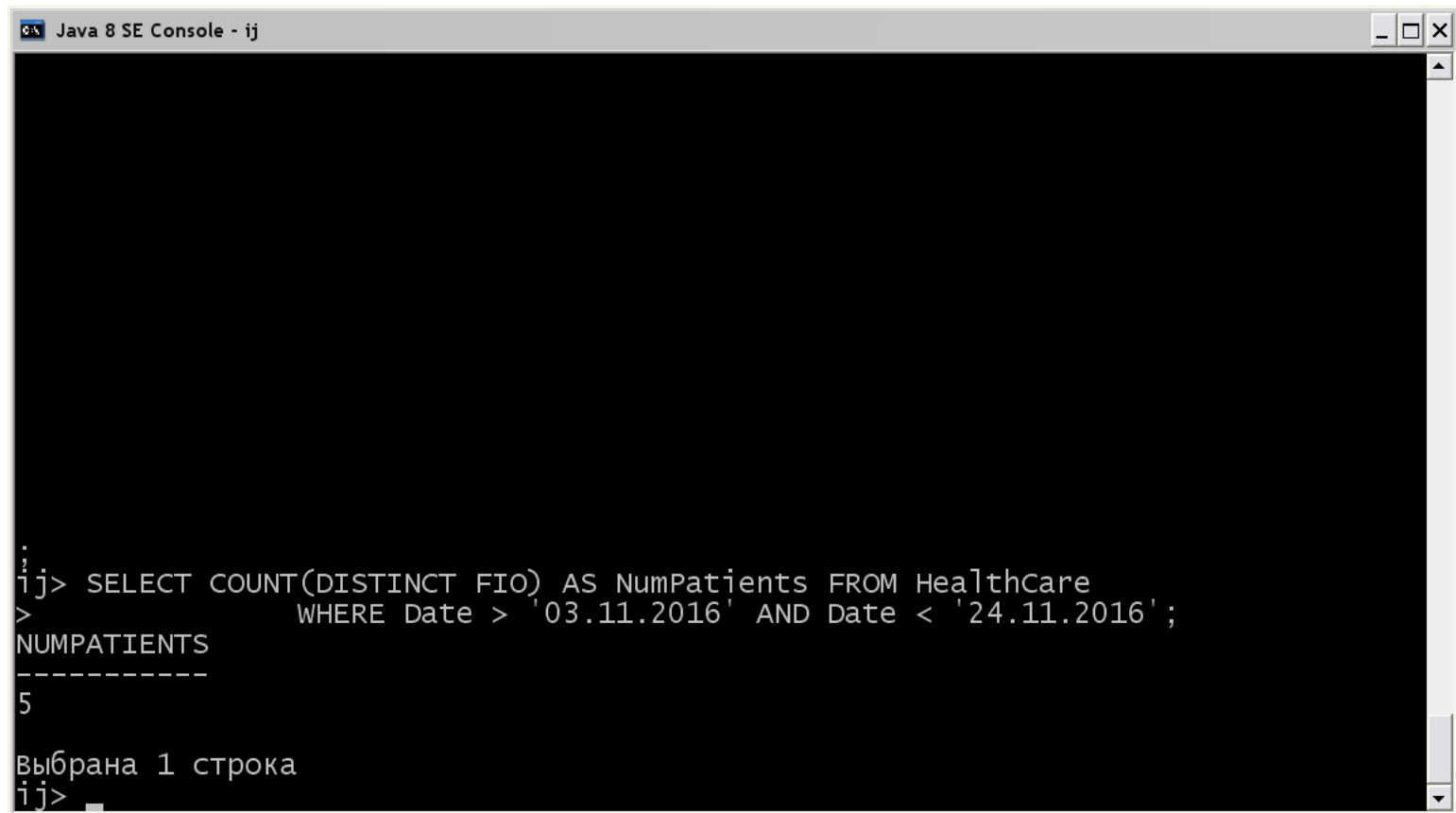
Введение в язык запросов SQL и Apache Derby

1) Сколько пациентов обслужено за месяц/год?

Составим запрос на более короткий период:

```
SELECT COUNT(DISTINCT FIO) AS NumPatients
FROM HealthCare
WHERE    Date > '03.11.2016' AND
        Date < '24.11.2016';
```

Введение в язык запросов SQL и Apache Derby



The screenshot shows a Java 8 SE Console window titled "Java 8 SE Console - ij". The console displays the following SQL query and its result:

```
ij> SELECT COUNT(DISTINCT FIO) AS NumPatients FROM HealthCare
>      WHERE Date > '03.11.2016' AND Date < '24.11.2016';
NUMPATIENTS
-----
5
```

Below the result, the text "Выбрана 1 строка" (1 row selected) is displayed, followed by the prompt "ij>".

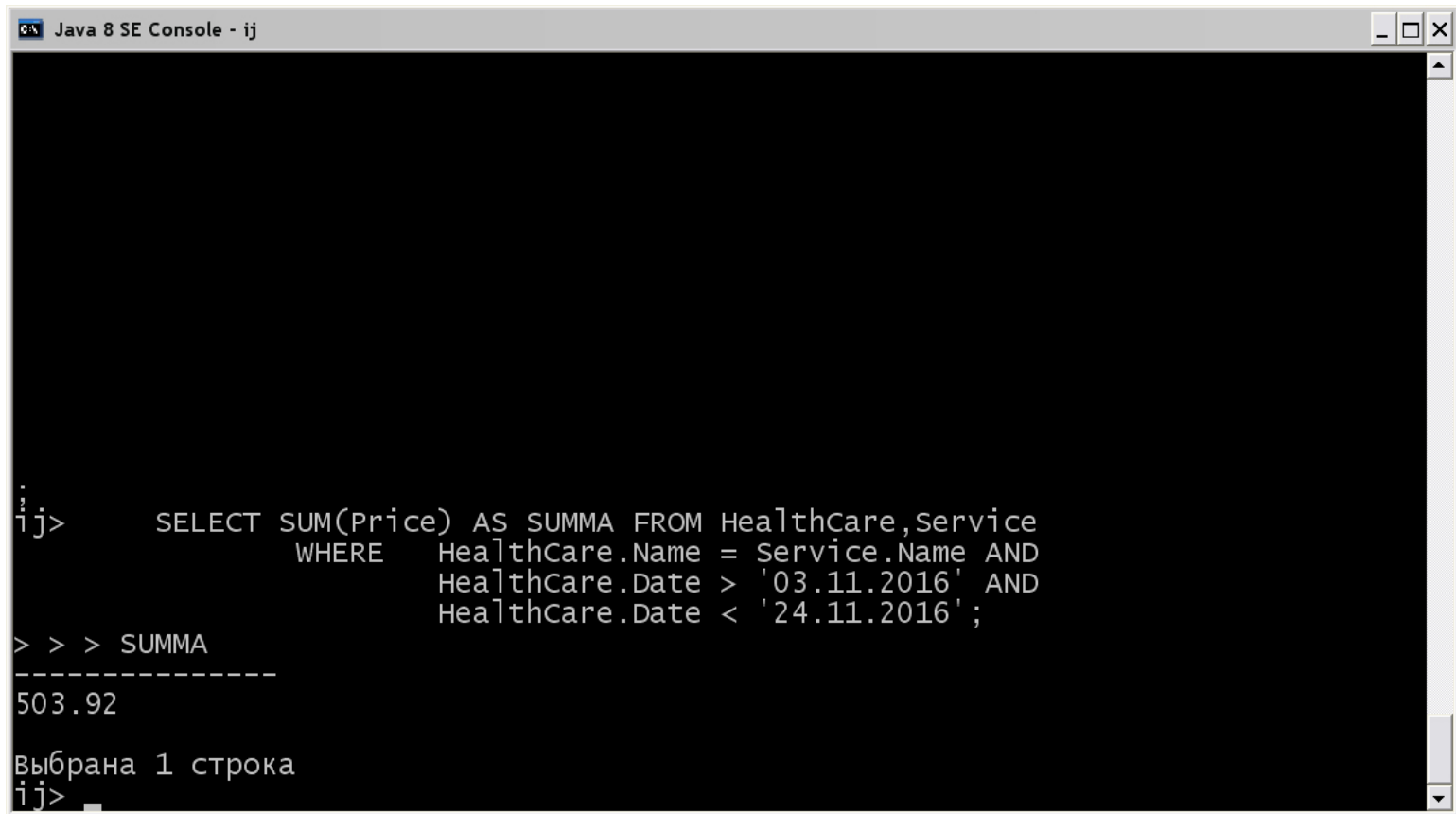
Введение в язык запросов SQL и Apache Derby

2) Какова сумма оказанных услуг за месяц/год?

Составим запрос на более короткий период:

```
SELECT SUM(Price) AS SUMMA FROM HealthCare, Service
WHERE      HealthCare.Name = Service.Name AND
           HealthCare.Date > '03.11.2016' AND
           HealthCare.Date < '24.11.2016';
```


Введение в язык запросов SQL и Apache Derby



```
Java 8 SE Console - ij
ij> SELECT SUM(Price) AS SUMMA FROM HealthCare,Service
      WHERE HealthCare.Name = Service.Name AND
            HealthCare.Date > '03.11.2016' AND
            HealthCare.Date < '24.11.2016';
> > > SUMMA
-----
503.92
Выбрана 1 строка
ij>
```

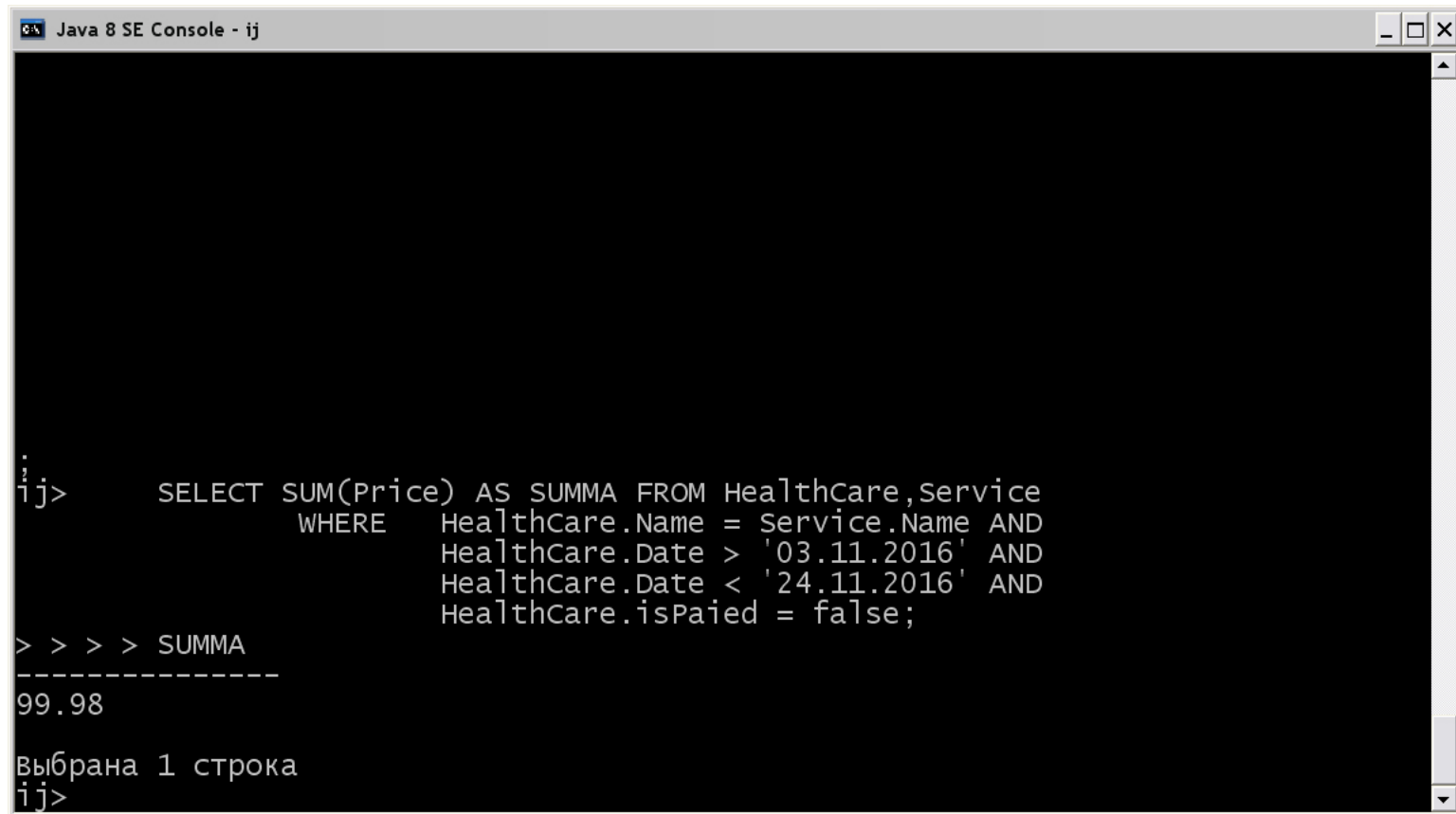
Введение в язык запросов SQL и Apache Derby

3) Какова сумма неоплаченных счетов за месяц/год?

Составим запрос на более короткий период:

```
SELECT SUM(Price) AS SUMMA FROM HealthCare, Service
WHERE      HealthCare.Name = Service.Name AND
           HealthCare.Date > '03.11.2016' AND
           HealthCare.Date < '24.11.2016' AND
           HealthCare.isPaied = false;
```

Введение в язык запросов SQL и Apache Derby



```
Java 8 SE Console - ij
ij> SELECT SUM(Price) AS SUMMA FROM HealthCare,Service
      WHERE      HealthCare.Name = Service.Name AND
                  HealthCare.Date > '03.11.2016' AND
                  HealthCare.Date < '24.11.2016' AND
                  HealthCare.isPaied = false;
> > > > SUMMA
-----
99.98
Выбрана 1 строка
ij>
```

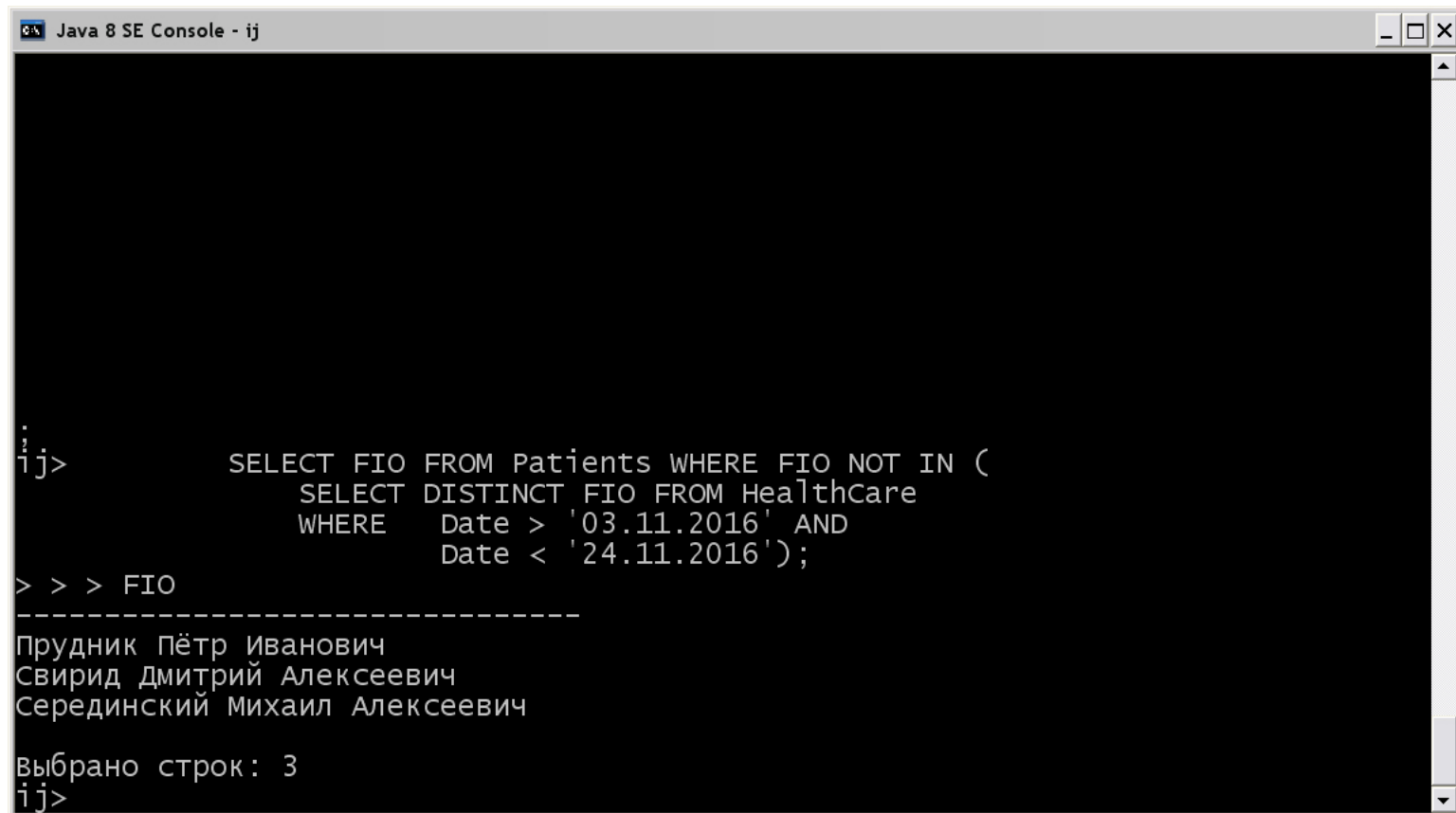
Введение в язык запросов SQL и Apache Derby

4) Список пациентов, не обращавшихся за помощью больше года

Составим запрос на более короткий период:

```
SELECT FIO FROM Patients WHERE FIO NOT IN (  
    SELECT DISTINCT FIO FROM HealthCare  
    WHERE          Date > '03.11.2016' AND  
                   Date < '24.11.2016');
```

Введение в язык запросов SQL и Apache Derby



```
Java 8 SE Console - ij
ij> SELECT FIO FROM Patients WHERE FIO NOT IN (
      SELECT DISTINCT FIO FROM HealthCare
      WHERE Date > '03.11.2016' AND
            Date < '24.11.2016');
> > > FIO
-----
Прудник Пётр Иванович
Свирид Дмитрий Алексеевич
Серединский Михаил Алексеевич

выбрано строк: 3
ij>
```

Введение в язык запросов SQL и Apache Derby

5) Список услуг, которые не были востребованы больше года

Составим запрос на более короткий период:

```
SELECT Name FROM Service WHERE Name NOT IN (  
    SELECT DISTINCT Name FROM HealthCare  
WHERE      Date > '03.11.2016' AND  
           Date < '24.11.2016');
```

Введение в язык запросов SQL и Apache Derby

```
Java 8 SE Console - ij
ij> SELECT Name FROM Service WHERE Name NOT IN (
      SELECT DISTINCT Name FROM HealthCare
      WHERE Date > '03.11.2016' AND
            Date < '24.11.2016');
> > >
NAME
-----
лечение канала зуба
удаление зуба

выбрано строк: 2
ij>
```

Обращение к Apache Derby из программы Java

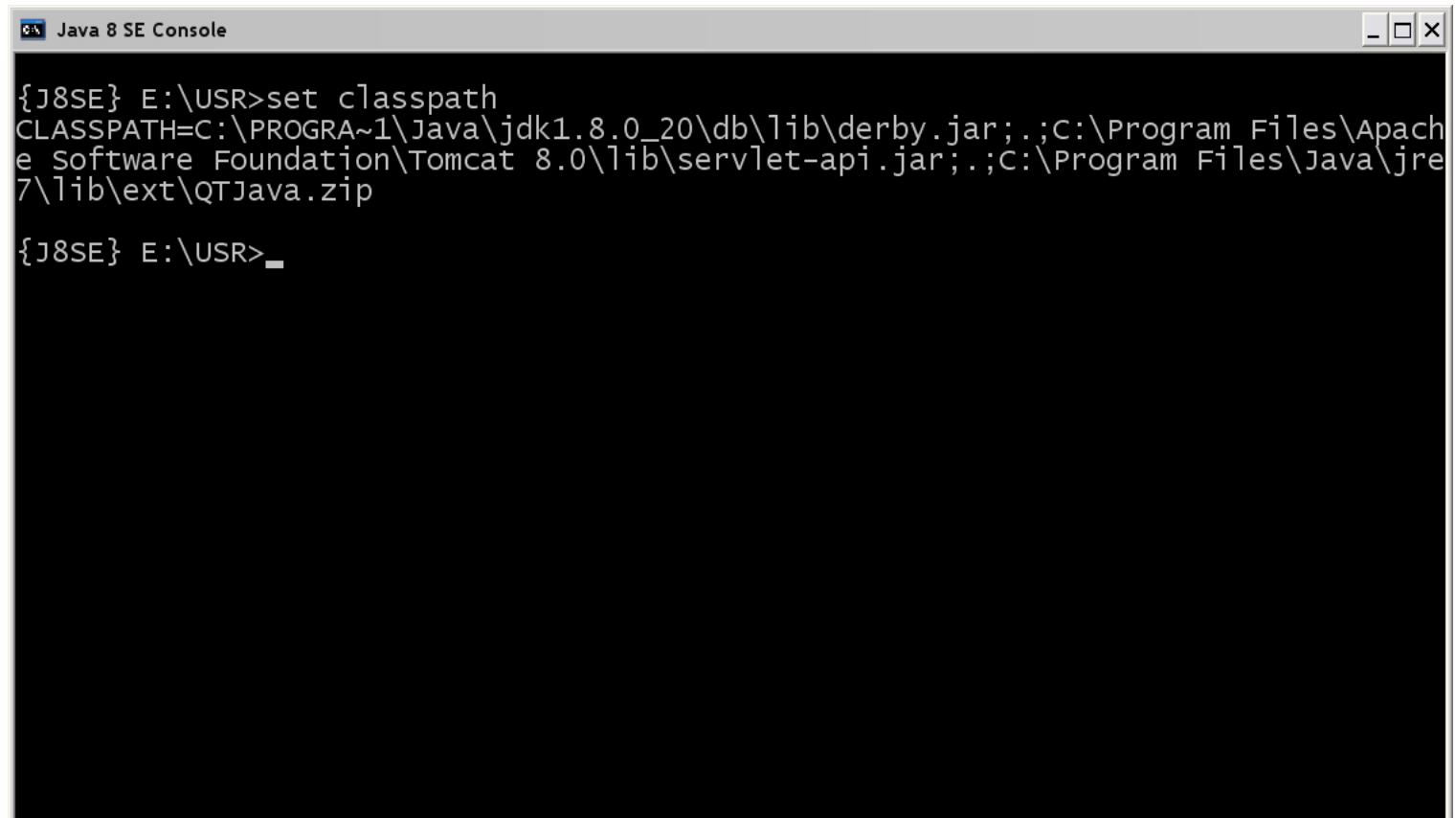
Обратимся к созданной БД из программы Java используя JDBC.

Следующий пример подключается к БД и распечатывает таблицу Patients. Пример использует драйвер JDBC с именем:

`org.apache.derby.jdbc.EmbeddedDriver`

из архива `derby.jar`. Чтобы программа находила его во время выполнения, необходимо включить его в переменную среды `CLASSPATH`. Например,

Обращение к Apache Derby из программы Java



```
{J8SE} E:\USR>set classpath
CLASSPATH=C:\PROGRA~1\Java\jdk1.8.0_20\db\lib\derby.jar;.;C:\Program Files\Apache Software Foundation\Tomcat 8.0\lib\servlet-api.jar;.;C:\Program Files\Java\jre7\lib\ext\QTJava.zip

{J8SE} E:\USR>_
```

Обращение к Apache Derby из программы Java

Аналогично тому, как мы работали с командным процессором Derby, мы можем это сделать из программы Java.

Код создания таблиц приведен в следующем примере (перед запуском создайте каталог C:\Dantist).

Обращение к Apache Derby из программы Java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;

public class DantistDBCreate {
    static String driver = "org.apache.derby.jdbc.EmbeddedDriver";
    static String connect = "jdbc:derby:DantistDB;create=true";
    public static void main(String[] args) {
        // Текущий каталог для Derby
        System.setProperty("derby.system.home", "C:\\\\Dantist" );
        try {
            // Регистрируем драйвер JDBC
            Class.forName( driver );
            // Подключаемся к БД
            Connection conn= DriverManager.getConnection(connect);
```

Обращение к Apache Derby из программы Java

```
// Выполняем запросы
Statement st = conn.createStatement();
st.executeUpdate("CREATE TABLE Service " +
    "(Name VARCHAR(32) PRIMARY KEY, Price REAL )");
st.executeUpdate("CREATE TABLE Patients " +
    "(FIO VARCHAR(32) PRIMARY KEY, Sex INT, " +
    "BirthDate DATE )");
st.executeUpdate("CREATE TABLE HealthCare " +
    "(FIO VARCHAR(32) REFERENCES Patients (FIO), " +
    "Name VARCHAR(32) REFERENCES Service (Name), " +
    "Date DATE, isPaied BOOLEAN )");
st.close();
} catch (Exception e) {
    System.err.println("Run-time error: " + e );
}
}}
```

Обращение к Apache Derby из программы Java

В результате запуска приведенной программы будет создана новая БД в каталоге C:\Dantist с пустыми таблицами.

Заполнить таблицы тестовыми данными можно аналогично тому, как мы это делали в командном процессоре ij.

Код просмотра таблицы Patients приведен в следующем примере.

Обращение к Apache Derby из программы Java

```
import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class DantistDB {

    static String driver =
        "org.apache.derby.jdbc.EmbeddedDriver";
    static String connect = "jdbc:derby:DantistDB";
```

Обращение к Apache Derby из программы Java

```
public static void main(String[] args) {  
    // Текущий каталог для Derby  
    System.setProperty("derby.system.home",  
                        "C:\\\\Dantist");  
  
    try {  
        // Регистрируем драйвер JDBC  
        Class.forName( driver );  
        // Подключаемся к БД  
        Connection conn =  
            DriverManager.getConnection(connect);  
        // Выполняем запрос  
        Statement st = conn.createStatement();  
        ResultSet rec = st.executeQuery(  
            "SELECT * FROM Patients ORDER BY FIO");  
    }  
}
```

Обращение к Apache Derby из программы Java

```
// Просматриваем и печатаем записи
// результирующей таблицы
while (rec.next()) {
    String fio = rec.getString("FIO");
    int sex = rec.getInt("Sex");
    Date birthDate = rec.getDate("BirthDate");
    System.out.println( " FIO: " + fio +
        " Sex: " + sex +
        " BirthDate: " + birthDate );
}
rec.close(); st.close();
} catch (Exception e) {
    System.err.println("Run-time error: " + e );
}}}
```


Обращение к Apache Derby из программы Java

Результат работы программы:

```
FIO: Бруцкий Владислав Витальевич Sex: 1 BirthDate: 1988-08-31
FIO: Гореликов Тимофей Дмитриевич Sex: 1 BirthDate: 1990-01-10
FIO: Касаржевская Алла Марковна Sex: 0 BirthDate: 1983-10-11
FIO: Максимов Сергей Леонидович Sex: 1 BirthDate: 1989-02-27
FIO: Прудник Пётр Иванович Sex: 1 BirthDate: 1980-02-01
FIO: Русакова Екатерина Сергеевна Sex: 0 BirthDate: 1991-03-07
FIO: Свирид Дмитрий Алексеевич Sex: 1 BirthDate: 1977-05-06
FIO: Серединский Михаил Алексеевич Sex: 1 BirthDate: 1979-06-21
```

Литература

- 1) Хорстманн, Кей С., Корнелл, Гарри. Java. Библиотека профессионала, том 2. Расширенные средства, 9-е изд.: Пер. с англ. – М.: ООО «И.Д.Вильямс», 2014. ISBN 978-5-8459-1870-3
- 2) Шкарина Л., Язык SQL: учебный курс. – СПб.: Питер, 2001. ISBN 5-318-00195-5
- 3) Мартин Грабер. Справочное руководство по SQL. – М.: «Лори», 1997. ISBN 5-85582-022-X
- 4) Ульман Дж. Базы данных на паскале/Пер. с англ. – М.: Машиностроение, 1990. ISBN 5-217-00628-5