Частное учреждение образования
Колледж бизнеса и права

УТВЕРЖДАЮ
Заведующий
методическим кабинетом
_____Е.В. Паскал
«___»_____2021

Специальность: 2-40 01 01 «Программное обеспечение информационных технологий»	Дисциплина: «Основы кроссплатформенного программирования»

ЛАБОРАТОРНАЯ РАБОТА № 11
Инструкционно-технологическая карта

Тема: «Построение приложений баз данных с использованием JDBC API»

Цель: Научиться создавать и связывать базы данных инструментами языка Java.

Время выполнения: 6 часов

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Изучить теоретические сведения;
2. Ответить на контрольные вопросы;
3. Откомпилировать примеры программ из раздела «Теоретические сведения»;
4. Выполнить ИДЗ.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ ДЛЯ ВЫПОЛНЕНИЯ РАБОТЫ

Система управления базами данных — это ПО, которое обеспечивает взаимодействие разных внешних программ с данными и дополнительные службы (журналирование, восстановление, резервное копирование и тому подобное), в том числе посредством SQL. То есть программная прослойка между данными и внешними программами с ними работающими. В этой части ответим на вопросы что такое SQL, что такое SQL сервер и создадим первую программу для взаимодействия с СУБД.

Первая программа

Приступим к практической части. Создадим Java-проект с помощью IDE JetBrains IntelliJ IDEA. Заметим, что редакция Ultimate Edition содержит в своём составе замечательный инструмент для работы с SQL и БД — Data Grip. Однако она платная для большинства пользователей. Так что нам для

2

учебных целей остается использовать общедоступную IntelliJ IDEA Community Edition. Итак:

1. Запускаем IDE и создаём новый проект:

Выбираем Java-проект, указываем версию SDK

2. На следующем шаге выбираем в качестве типа консольное приложение:

Указываем имя проекта, пакет и его размещение на диске

3. Отложим на минуту IDE и загрузим с www.h2database.com необходимый JDBC-файл для работы с СУБД H2 (download platform independent ZIP):

4. Заходим внутрь скачанного файла (нас интересует jar-файл по пути h2\bin, который нам далее понадобится, скопируем его):

5. Возвращаемся в IDE и создаём в корне проекта директории:; lib – здесь JAR-библиотека JDBC:

6. Переносим в директорию lib jar-файл из шага 6, и добавим его в проект как библиотеку (ПКМ Add :

7. Переименуем java-файл в src/sql/demo на StockExchange.java (если забыли, мы собираемся эмулировать простую «биржу»), поменяем его содержимое и запустим:

Как правило, существует пять шагов для создания соединения

JDBC. Шаг 1 — Регистрация драйвера базы данных JDBC.

```
Class.forName ("org.h2.Driver");
```

Шаг 2 — Открытие соединения.

```
Connection conn = DriverManager.getConnection ("jdbc:h2:~/test",
"sa","");
```

Шаг 3 — Создание заявления.

```
Statement st = conn.createStatement();
```

Шаг 4 — Выполнение оператора и получение Resultset.

```
Stmt.executeUpdate("sql statement");
```

Шаг 5 — Закрытие соединения.

```
conn.close();
```

Теперь мы умеем подключаться к СУБД и отключаться от неё. Каждый шаг отражается в консоли. При первом подключении к СУБД создаётся файл базы данных stockExchange.mv.db.

Разбор кода

```
import java.sql.*;
public class StockExchange {
 public static final String DB_URL = "jdbc:h2:~/test";
 public static final String DB_Driver = "org.h2.Driver";
 // Database credentials
 static final String USER = "sa";
 static final String PASS = "";
 public static void main(String[] args) {
 try {
 Class.forName(DB_Driver); //Проверяем наличие JDBC драйвера для работы с БД
```

3

```
 Connection connection = DriverManager.getConnection(DB_URL,USER,PASS);
//соединениесБД
 System.out.println("Соединение с СУБД выполнено.");
 connection.close(); // отключение от БД
 System.out.println("Отключение от СУБД выполнено.");
 } catch (ClassNotFoundException e) {
 e.printStackTrace(); // обработка ошибки Class.forName
 System.out.println("JDBC драйвер для СУБД не найден!");
 } catch (SQLException e) {
 e.printStackTrace(); // обработка ошибок DriverManager.getConnection
System.out.println("Ошибка SQL !");
 }
 }
```

}Блок констант:

1. DB_Driver: Здесь мы определили имя драйвера, которое можно узнать, например, кликнув мышкой на подключенную библиотеку и развернув её структуру в директории lib текущего проекта.

2. DB_URL: Адрес нашей базы данных. Состоит из данных, разделённых двоеточием:

3. Протокол=jdbc

4. Вендор (производитель/наименование) СУБД=h2

5. Расположение СУБД, в нашем случае jdbc:h2:~/test, а может содержать путь до файла (c:/JavaPrj/SQLDemo/db/stockExchange). Обработка ошибок:

Вызов методов нашего кода может вернуть ошибки, на которые следует обратить внимание. На данном этапе мы просто информируем о них в консоли. Заметим, что ошибки при работе с СУБД — это чаще всего SQLException.

Хранения данных в СУБД

· Сформулируем начальный свод правил хранения данных для СУБД: · Данные в СУБД организованы в таблицы (TABLE), представляющие собой набор записей.

· Все записи имеют одинаковые наборы полей. Они задаются при создании таблицы.

· Для поля можно выставить значение по умолчанию (DEFAULT). · Для таблицы можно выставить ограничения (CONSTRAINT), описывающие требования к её данным чтобы обеспечить их целостность. Это можно сделать на этапе создания таблицы (CREATE TABLE) или добавить позже (ALTER TABLE … ADD CONSTRAINT).

· Наиболее распространённые CONSTRAINT:

- ✓ Первичный ключ PRIMARY (Id в нашем случае).
- ✓ Уникальное значение поле UNIQUE (VIN для таблицы автотранспорта).

✓ Проверка поля CHECK (значение процентов не может быть больше 100). Одно из частных ограничений на поле – NOT NULL или NULL, запрещающее/разрешающее хранить NULL в поле таблицы.

✓ Ссылка на стороннюю таблицу FOREIGN KEY (ссылка на акцию в таблице курсов акций).

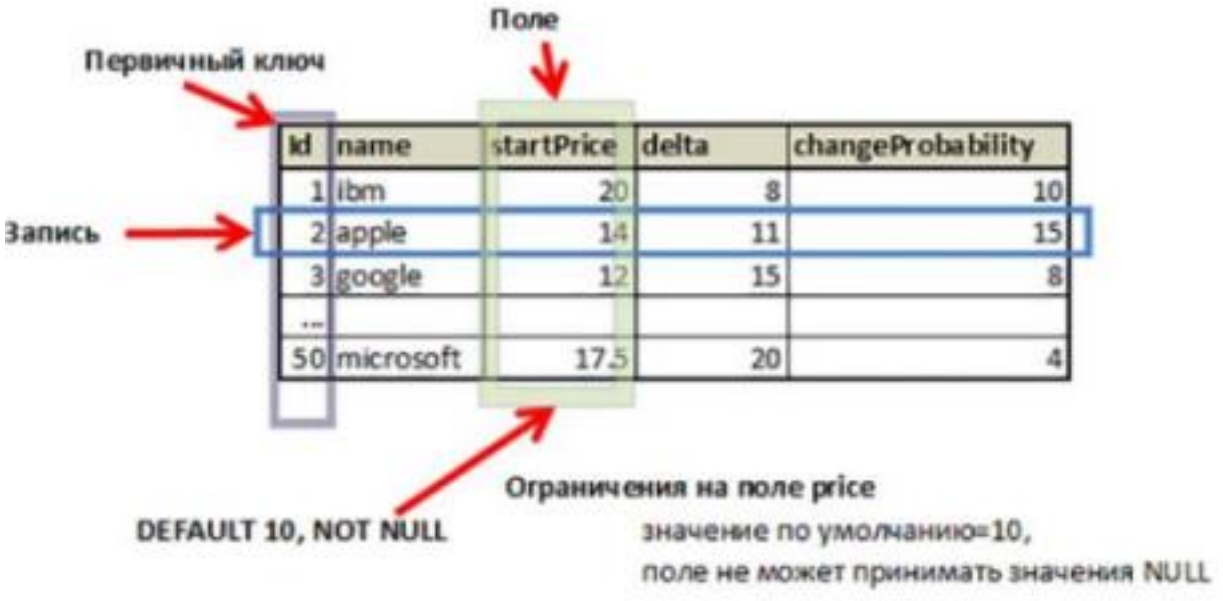
✓ Индекс INDEX (индексирование поля для ускорения поиска значений по нему).

✓ Выполнение модификации записи (INSERT, UPDATE) не произойдёт, если значение её полей противоречат ограничениям (CONSTRAINT). · Каждая таблица может иметь ключевое поле (или несколько), по которой можно однозначно определить запись. Такое поле (или поля, если они формируют составной ключ) образует первичный ключ таблицы — PRIMARY KEY.

✓ Первичный ключ обеспечивает уникальность записи в таблице, по нему создается индекс, что дает быстрый доступ по значению ключа ко всей записи.

✓ Наличие первичного ключа существенно облегчает создание ссылок между таблицами. Далее мы будем использовать искусственный первичный ключ: для первой записи id = 1, каждая следующая запись будет вставляться в таблицу с увеличенным на единицу значением id. Такой ключ часто называют AutoIncrement или AutoIdentity.

Собственно, таблица акций:



Можно ли в таком случае использовать в качестве ключа имя акции? По большому счёту — да, только вот есть вероятность, что какая-то компания выпускает разные акции и именует их только собственным названием. В таком случае уникальности уже не будет.

На практике искусственный первичный ключ используют довольно часто. Согласитесь, использование ФИО в качестве уникального ключа в

таблице, содержащей записи по людям, не обеспечит уникальности. Как и использование комбинации ФИО и даты рождения.

Типы данных в СУБД

Как и в любом другом языке программирования в SQL существует типизация данных. Приведём наиболее распространённые типы данных SQL: Целые типы

SQL-тип	SQL-синонимы	Соответствие в Java	Описание
INT	INT4,INTEGER	java.lang.Integer	4-байтовое целое, -2147483648 … 2147483647
AUTO_INCREMENT	INCREMENT	java.lang.Long	Инкрементальный счётчик, уникальный для таблицы. Если в неё вставляют новое значение, он увеличивается на единицу Сгенерированные значения никогда не повторяются.

Вещественные

DECIMAL(N,M)	DEC, NUMBER	java.math.BigDecimal	Десятичная дробь с фиксированной точностью (N цифр целой части и M — дробной). В основном предназначены для работы с финансовыми данными.
DOUBLE	FLOAT8	java.lang.Double	Вещественное число двойной точности (8 байт).
REAL	FLOAT4	java.lang.Real	Вещественное число одинарной точности (4 байта).

Строковые

VARCHAR(N)	NVARCHAR	java.lang.String	Строка в формате UNICODE длины N. Длина ограничена значением 2147483647 Полностью загружает содержимое строки в память.

Дата и время

TIME		java.time.LocalTime, java.sql.Time	Хранение времени (до наносекунд), при конвертации в DATETIME, в качестве даты выставляется 1 янв 1970.
DATE		java.time.LocalDate, java.sql.Timestamp	Хранение дат в формате yyyy-mm-dd, время выставляется как 00:00

DATETIME	TIMESTAMP	java.time.LocalDateTime, java.sql.Timestamp	Хранение даты + времени (без учёта временных зон).

Стиль написания кода в SQL

- Ключевые и зарезервированные слова, в том числе команды и операторы, нужно писать прописными буквами: CREATE TABLE, CONSTRAINT…
- Имена таблиц, полей и прочих объектов не должны совпадать с ключевыми словами языка SQL, но могут содержать их в себе.
- Имена таблиц должны отражать их назначение. Они записываются строчными буквами. Слова в наименовании отделены друг от друга подчёркиваниями. Слово в конце должно быть во множественном числе: traiders (трейдеры), share_rates (курс акций).
- Имена полей таблиц должны отражать их назначение. Их нужно записывать строчными буквами, слова в наименовании нужно оформлять в стиле Camel Case, а слово в конце нужно использовать в единственном числе: name (наименование), share_rates (курс акций).
- Поля искусственных ключей должны содержать слово id.
- Имена CONSTRAINT должны удовлетворять правилам именования таблиц. Также они должны включать участвующие в них поля и таблицы, начинаться со смыслового префикса: check_ (проверка значения поля), pk_ (первичный ключ), fk_ (внешний ключ), uniq_ (уникальность поля), idx_ (индекс). Пример: pk_traider_share_actions_id (первичный ключ по полю id для таблицы traider_share_actions).

Создание таблиц в СУБД

Перейдём непосредственно к созданию таблиц (CREATE TABLE). Для этого нам желательно иметь предварительные данные:

- имя таблицы
- имена и тип полей
- ограничения (CONSTRAINTS) на поля
- значения по умолчанию для полей (при наличии)
- первичный ключ (PRIMARY KEY) при наличии
- связи между таблицами (FOREIGN KEY)

Не будем изучать досконально все опции команды CREATE TABLE, рассмотрим основы SQL на примере создания таблицы для: CREATE TABLE Employees(

```
id_employee INTEGER not NULL,
surname VARCHAR(255),
name VARCHAR(255),
patronymic VARCHAR(255),
position VARCHAR(255),
phone VARCHAR(255),
address VARCHAR(255), PRIMARY KEY ( id_employee ))
```

Разберём подробнее:

• CREATE TABLE Employees (описание полей) — создание таблицы с

указанным именем, в описании поля разделяются запятой. Любая команда завершается точкой с запятой.

• Описание поля начинается с его имени, далее следует тип, CONSTRAINT и значение по умолчанию.

• id_employee INTEGER not NULL – поле id целого типа — это первичный ключ PRIMARY KEY (id_employee).

Связи между таблицами

Для разбора описания связей между таблицами посмотрим создание:

```
statement.execute("ALTER TABLE Orders ADD FOREIGN KEY (id_employee)
REFERENCES Employees(id_employee)");
```

Ссылку на значения другой таблицы можно задать следующим образом:

```
ALTER TABLE таблица_из_которой_ссылаемся ADD FOREIGN KEY
(поле_которое_ссылается) REFERENCES таблица_на_которую_ссылаемся
(поле_на_которое_ссылаемся)
```

JDBC

Этапы работы с JDBC:

• Class.forName() загружает класс и регистрирует его в DriverManager;

• DriverManager.getConnection() вернёт Connection – соединение с указанной в аргументе метода базой данных и использованием соответствующего JDBC драйвера (который был загружен с использованием Class.forName()).

• createStatement() возвратит нам Statement – объект, на основе которого можно формировать запросы к БД. Существуют также: CallableStatement для вызова собственных СУБД SQL-функций и процедур (их называют хранимыми).

• PreparedStatement облегчающий создание параметризованных и пакетных запросов.

• Имея «на руках» statement, execute() позволит отправить запрос в виде команды языка запросов SQL непосредственно на выполнение СУБД и вернет ответ в виде ResultSet. Для удобства существуют:

• executeQuery() – для чтения данных из СУБД.

executeUpdate() – для модификации данных в СУБД.

• Непосредственно ответ сервера в виде ResultSet можно обработать, итерируя его через first(), last(), next() и так далее. Отдельные поля результаты мы можем получить через геттеры: getInteger(), getString()…

Следует иметь в виду, что после работы с СУБД для экономии ресурсов желательно закрыть за собой (в правильном порядке!) объекты ResultSet, Statement и Connection для экономии ресурсов. Помните, закрывая вышестоящий в последовательности на схеме объект, вы каскадно закроете все порождённые в процессе работы с ним объекты. Таким образом, закрытие соединения (Connection) приведёт к закрытию всех его Statement и всех ResultSet, полученных при их помощи.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что представляет собой СУБД?
2. Какие существуют шаги для создания соединения JDBC
3. Как создаются таблицы СУБД?
4. Какие есть типы данных в СУБД

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Реализовать БД по вариантам. Заполнить с помощью программы.

Варианты для самостоятельного выполнения:

Вариант 1

Табель – год, месяц, ТабНомСотрудника, КодПодразделения, КоличествоОтработДней, ДневнойТариф

Сотрудники – ТабНомСотрудника. ФамилияСотрудника, КодДолжности.

Разряд Должности – КодДолжности, НаименованиеДолжности

Подразделения – КодПодразделения, НаименованиеПодразделения

Вариант 2

Накладные – НомДок, ДатаПост, Поставщик, НоменкНомМатериала, ТабНомМОЛ,

Колич Материалы – НоменклНомМатериала, НаименованиеМатериала, ЕдиницаИзмерения, Цена

МатОтЛица – ТабНомМОЛ, ФамилияМОЛ

Поставщики – НомПоставщика, НаименПоставщика

Вариант 3

Издания – ИндексИздания. НазвИздания, СтоимостьЗаГод, КодГруппыИзданий Подписчики – КодПодписчика, ФамилияПодписчика, Адрес

ГруппыИзданий – КодГруппы, НаименованиеГруппыИзданий, ПроцентСкидки

Подписка – ГодПодписки, ИндексИздания, КодПодписчика, СрокПодписки (мес), МестоДоставки

Вариант 4

Страны – Наименование, Площадь (км2), Население, ГосЯзык, ДенежнаяЕдиница, ЧастьСвета

ЧастиСвета – КодЧастиСвета, НаименованиеЧастиСвета, Полушарие, ПлощадьСушиКвКм

ДенежныеЕдиницы – КодДенЕд, НаименованиеДенЕд

Языки – КодЯзыка, НаименованиеЯзыка, ЯзыковаяГруппа

ЯзыковыеГруппы – КодЯзГруппы, НаименованиеЯзГруппы

Вариант 5

Алмазы – Наименование, СтранаПроисхождения, КогдаНайден, МассаВКаратах Страны – КодСтраны, НаименованиеСтраны, ЧастьСвета

Континенты – КодЧастиСвета, НаименованиеЧастиСвета, Полушарие, ПлощадьСушиКвКм

Вариант 6

Книги – шифр книги, автор, название, год издания, количество экземпляров. Читатели – читательский билет, фамилия, имя, отчество, адрес.

Выданные книги – шифр книги, читательский билет, дата выдачи, дата возвращения, дата фактического возвращения.

Вариант 7

Товары – код товара, наименование товара, количество товара.
Поступление товаров – код товара, дата поступления, цена приобретения товара за единицу, код поставщика.
Продажа товаров – код товара, месяц продажи, проданное количество за месяц, цена продажи товара.
Поставщики – код поставщика, название поставщика, адрес поставщика, телефон поставщика.

Вариант 8

Студенты – шифр студента, фамилия, имя, отчество, курс, группа.
Экзамены – шифр студента, дата, шифр дисциплины, оценка.
Зачеты – шифр студента, дата, шифр дисциплины, зачет.
Дисциплины – шифр дисциплины, название дисциплины, количество часов.

Вариант 9

Склад – код склада, код товара, количество, дата поступления.
Товары – код товара, название товара, срок хранения.
Заявки – код заявки, название организации, код товара, требуемое количество.
Отпуск товаров – код отпуска товаров, код заявки, код товара, отпущенное количество, дата отпуска товара.

Вариант 10

Анкета – номер абитуриента, фамилия, имя, отчество, дата рождения, оконченное среднее учебное заведение (название, номер, населенный пункт), дата окончания учебного заведения, наличие красного диплома или золотой / серебряной медали, адрес, телефон, шифр специальности.
Специальности – шифр специальности, название специальности.
Дисциплины – шифр дисциплины, название дисциплины.
Вступительные экзамены – номер абитуриента, шифр дисциплины, экзаменационная оценка.

Вариант 11

Транспорт – марка автомобиля, государственный номер, расход топлива. Заявки – код заявки, дата заявки, название груза, количество груза, пункт отправления, пункт назначения.
Доставка – № п/п, дата и время отправления, дата и время прибытия, код заявки, государственный номер автомобиля, пройденное расстояние.

Вариант 12

Клиенты – код клиента, фамилия, имя, отчество, телефон, адрес, паспортные данные, залог.
Склад – код оборудования, название, количество, залоговая стоимость, остаток. Прокат – № п/п, клиент, оборудование, дата выдачи, срок возврата, отметка о возврате, оплата проката.

Вариант 13

Клиенты – код клиента, фамилия, имя, отчество, паспорт, телефон, адрес, заработная плата.
Виды кредитов – код кредита, название кредита, процентная ставка, условия предоставления.
Предоставленные кредиты – № п/п, клиент, кредит, дата предоставления, срок, дата возврата, сумма, отметка о возврате.

Вариант 14

Клиенты – код клиента, фамилия, имя, отчество, телефон, адрес, паспорт. Сотрудники – код сотрудника, фамилия, имя, отчество, должность, телефон, адрес, паспортные данные.

Туристические маршруты – код маршрута, название, описание маршрута, страна, стоимость путевки, количество дней, вид транспорта.
Заказы – код заказа, клиент, маршрут, сотрудник (менеджер, оформивший заказ), дата, отметка об оплате.

Вариант 15

Врачи – код врача, ФИО, должность, специализация, стаж работы, адрес, телефон.
Болезни – № п/п, название заболевания, рекомендации по лечению, меры профилактики.
Пациенты – код пациента, ФИО, адрес, телефон, страховой полис, паспорт. Диагноз – № п/п, пациент, заболевание, лечащий врач, дата обращения, дата выздоровления.

Вариант 16

Отделы – название отдела, кол-во прилавков, кол-во продавцов, номер зала.
Сотрудники– фамилия, имя, отчество, отдел, год рождения, год поступления на работу, стаж, должность, пол, адрес, город, телефон.
Должности – название должности, сумма ставки.
Товары – название товара, отдел, страна производитель, условия хранения, сроки хранения.

Вариант 17

Автотранспорт – название транспорта (автобусы, такси, маршрутные такси, прочий легковой транспорт, грузовой транспорт и т.д.), пробег, кол-во ремонтов, характеристика. Маршруты – название маршрута, транспорт, водитель, график работы.
Водители – фамилия, имя, отчество, год рождения, год поступления на работу, стаж, должность, пол, адрес, город, телефон

Вариант 18

Лекарства – название, вид (таблетки, мази, настойки), цена.
Покупатели – фамилия, имя, отчество, адрес, город, телефон.
Продавцы – фамилия, имя, отчество, дата поступления, дата рождения, образование.
Рецепты – номер рецепта, дата выдачи, ФИО больного (покупатель), ФИО врача, диагноз пациента.
Продажа лекарств – дата, лекарство, кол-во, рецепт, продавец

Вариант 19

Товары – код товара, наименование товара, количество товара.
Поступление товаров – код товара, дата поступления, цена приобретения товара за единицу, код поставщика.
Продажа товаров – код товара, месяц продажи, проданное количество за месяц, цена продажи товара.

Вариант 20

Транспорт – марка автомобиля, государственный номер, расход топлива. Заявки – код заявки, дата заявки, название груза, количество груза, пункт отправления, пункт назначения.
Доставка – № п/п, дата и время отправления, дата и время прибытия, код заявки, государственный номер автомобиля, пройденное расстояние.

ДОМАШНЕЕ ЗАДАНИЕ

Индивидуальное задание

ЛИТЕРАТУРА

У. Савитч, язык Java. Курс программирования, Вильямс, 2017.

Преподаватель А.С.Кибисова

Рассмотрено на заседании цикловой комиссии программного обеспечения информационных технологий
Протокол № ____ от «___» _____ 2021
Председатель ЦК _____В.Ю.Михалевич