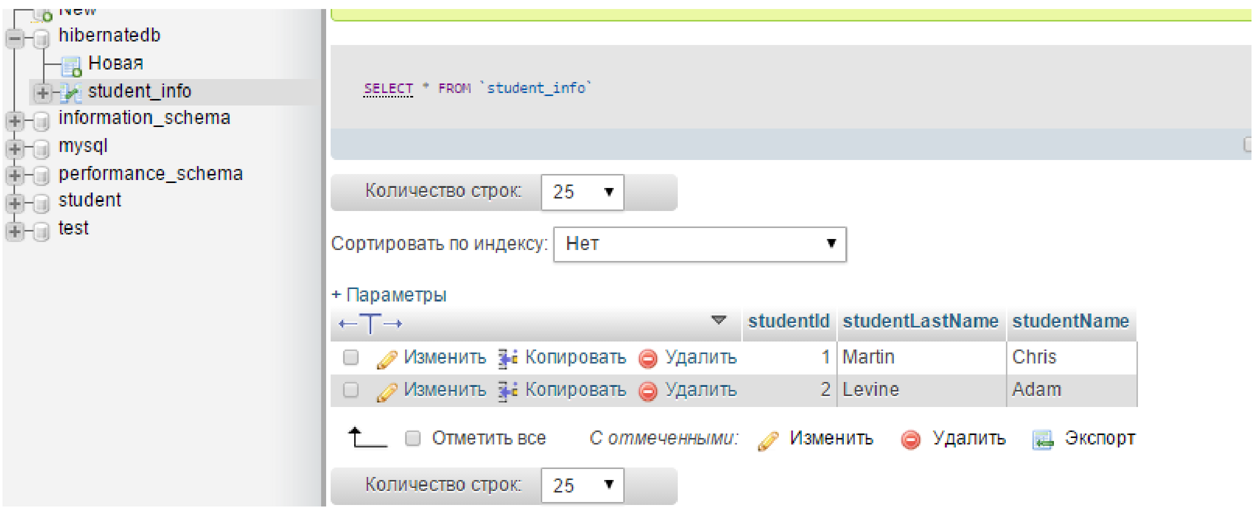
In my GUI application, that saves table with Student first name, last name and id. I used MySQL database. As GUI I used phpMyAdmin page. As ORM framework, I used Hibernate. I included hibernate library and mysql_connector_java.jar. Details of the connection with database are in hibernate.cfg.xml. My project has Student_info class and Test class. Student_info class contains fields as studentName, studentLastName, studentId. Here is piece of code:

```
@Entity //hibernate annotation which makes class Student_Info as an entity
bean
@Table(name="student_info") //to specify the details of the table that will
be used to persist the entity in the database
public class Student_Info {
	private String studentName;
	private String studentLastName;

	@Id  //primary key annotation
	@GeneratedValue(generator = "increment")
	@GenericGenerator(name = "increment", strategy = "increment")
	private int studentId;
	public Student_Info(){}
	public Student_Info(String studentName, String studentLastName, int
studentId){
		this.studentName = studentName;
		this.studentLastName = studentLastName;
		this.studentId = studentId;
	}

The Test class:

public static void main(String[] args) {

		Student_Info st = new Student_Info();
		Student_Info st2 = new Student_Info();
		Student_Info st3 = new Student_Info();
		st.setStudentName("Chris");
		st.setStudentLastName("Martin");
		st2.setStudentName("Adam");
		st2.setStudentLastName("Levine");
		st3.setStudentName("Gwen");
		st3.setStudentLastName("Stefani");

		SessionFactory sessionFactory = new
AnnotationConfiguration().configure().buildSessionFactory();
		Session session = sessionFactory.openSession();
		session.beginTransaction();
		session.save(st);
		session.save(st2);
		session.save(st3);
		session.getTransaction().commit();
		session.close();
```

/* SessionFactory contains all data regarding hibernate configuration file*/

/* to open the database connection call session*/

/* hibernate needs the transaction for functions like insert, delete, update*/



```
SELECT * FROM `student_info`
```

Количество строк: 25

Сортировать по индексу: Нет

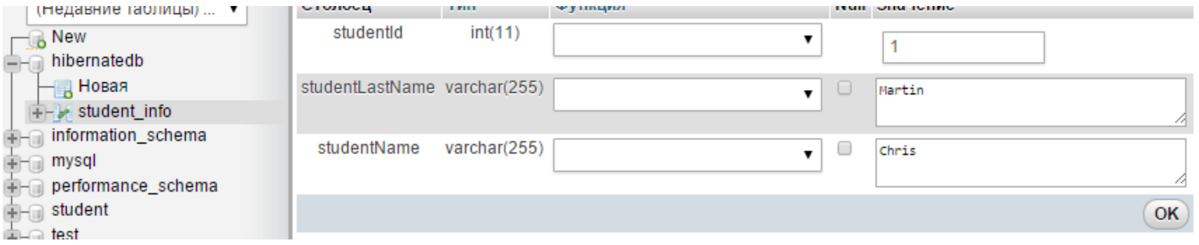
+ Параметры

	studentId	studentLastName	studentName
Изменить Копировать Удалить	1	Martin	Chris
Изменить Копировать Удалить	2	Levine	Adam

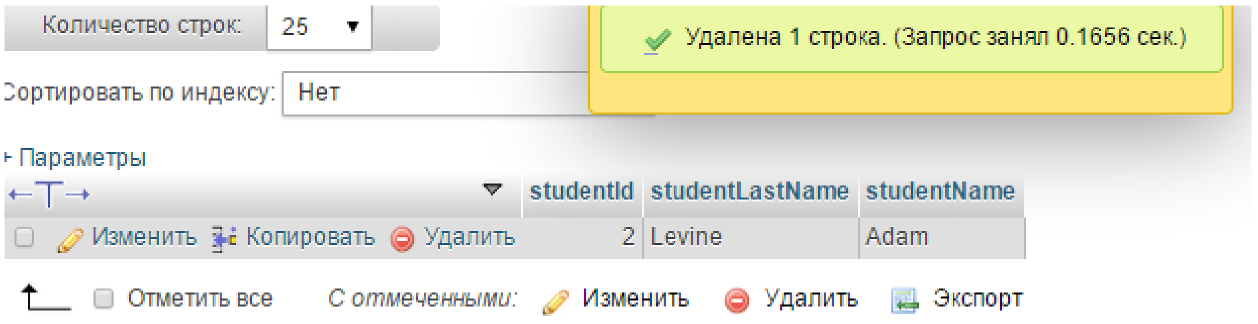
Отметить все С отмеченными: Изменить Удалить Экспорт

Количество строк: 25

My database is HibernateDB and table is student_info.



studentId	int(11)		1
studentLastName	varchar(255)		Martin
studentName	varchar(255)		Chris



Удалена 1 строка. (Запрос занял 0.1656 сек.)

Количество строк: 25

Сортировать по индексу: Нет

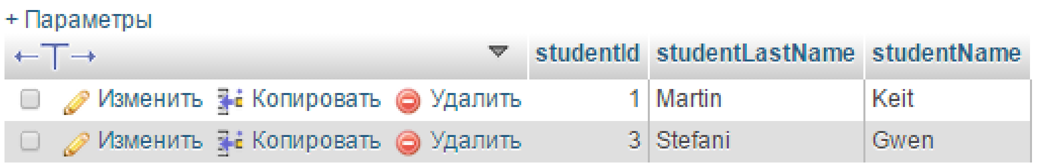
+ Параметры

	studentId	studentLastName	studentName
Изменить Копировать Удалить	2	Levine	Adam

Отметить все С отмеченными: Изменить Удалить Экспорт

I also used functions update and delete.

```
            st.setStudentName("Keit");
            session.update(st);

            session.delete(st2);
```



+ Параметры

	studentId	studentLastName	studentName
Изменить Копировать Удалить	1	Martin	Keit
Изменить Копировать Удалить	3	Stefani	Gwen

From hibernate.cfg.xml file:

```
<property name="connection.driver_class">com.mysql.jdbc.Driver</property>
        <property
name="connection.url">jdbc:mysql://localhost:3306/HibernateDB</property>

/*connection to MySQL database then to my database*/

<mapping class="student.Student_Info"/>
```

/*for class student_info I made table*/

How I understood ORM. ORM is displaying the objects of object-orieneted in the relational databse. All objects with all values, fields.