Module 02: Hibernate Persistence API

Exercise 02.1 – Basic Hibernate

The Setup:

The main objectives of this exercise are to guide you through creating your first hibernate application, and to show you the basics of the development environment.

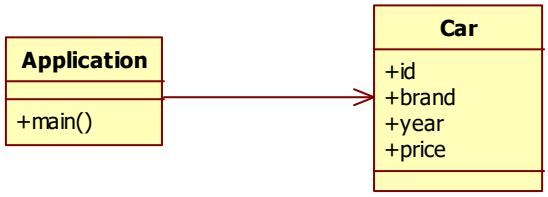
The most important parts here are the username and password (you may need to change them depending on your system), and the mapping class. For most exercises you can just copy hibernate.cfg.xml from a previous exercise and update the mapping classes.

You will also need to add the following to pom.xml

```
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>5.3.7.Final</version>
</dependency>
<dependency>
    <groupId>com.microsoft.sqlserver</groupId>
    <artifactId>mssql-jdbc</artifactId>
    <version>9.2.0.jre8</version>
</dependency>
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.20</version>
</dependency>
```

The Application:

The provided application is very straight forward. The Application class creates several Car objects, and uses Hibernate to persist and then retrieve them.



The Database:

Hibernate will automatically generate the database schema needed for the application. To view the database schema and the objects that were persisted by the application, open MySql "SQL Editor"

Once the program is open you can switch to the test database with:

```
USE cs544;
```

You can see all the tables that have been generated with:

```
SHOW TABLES;
```

You can see all the rows that have been inserted into a table with:

```
SELECT * FROM [table]
```

You can check the table structure (schema) with:

```
DESCRIBE [table]
```

You can remove (drop) a table from the database with:

```
SET foreign_key_checks = 0;

DROP TABLE [table]

SET foreign_key_checks = 1;
```

Please note that you may not have to disable foreign key checks every time.

The Exercise:

Study the code, and once you feel comfortable with what it does, create a similar but slightly more involved application consisting of a Book class and an AppBook class.

The Book class should have the following attributes and should be annotated for persistence:

```
public class Book {
	private int id;
	private String title;
	private String ISBN;
	private String author;
	private double price;
	private java.util.Date publish_date;
}
```

You can use your IDE to automatically generate the Constructors and Getter/Setter methods for your Book class. Hibernate requires that all entity classes have a default (empty) constructor, and a getter and a setter method for each attribute.

Lastly the **AppBook** class should perform the following operations:

- Open a session
- Create 3 books save them to the database
- Close the session

- Open a session
- Retrieve all books and output them to the console
- Close the session

- Open a session
- Retrieve a book from the database and change its title and price
- Delete a book (not the one that was just updated)
- Close the session

- Open a session
- Retrieve all books and output them to the console
- Close the session