```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>net.guides.springboot2</groupId>
    <artifactId>springboot2-jdbc-crud-example</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>
    <name>springboot2-jpa-crud-example</name>
    <description>Demo project for Spring Boot</description>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.5.RELEASE</version>
        <relativePath />
        <!-- lookup parent from repository -->
    </parent>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-jdbc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
  </project>
```

```
create table employees
(
   id integer not null,
   first_name varchar(255) not null,
   last_name varchar(255) not null,
   email_address varchar(255) not null,
   primary key(id)
);
```

```
## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url = jdbc:mysql://localhost:3306/demo?useSSL=false
spring.datasource.username = root
spring.datasource.password = root


## Hibernate Properties
#The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect =
org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto = update

logging.level.org.hibernate.stat=debug
# Show all queries
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
logging.level.org.hibernate.type=trace
```

```
package net.guides.springboot2.jdbc.model;
```

```java
public class Employee {

    private long id;
    private String firstName;
    private String lastName;
    private String emailId;

    public Employee() {

    }

    public Employee(long id, String firstName, String lastName, String
emailId) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
        this.emailId = emailId;
    }

    public long getId() {
        return id;
    }
    public void setId(long id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmailId() {
        return emailId;
    }
    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }
}
```

```java
package net.guides.springboot2.jdbc.repository;
```

```java
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import net.guides.springboot2.jdbc.model.Employee;

@Repository
public class EmployeeJDBCRepository {
    @Autowired
    JdbcTemplate jdbcTemplate;

    class EmployeeRowMapper implements RowMapper < Employee > {
        @Override
        public Employee mapRow(ResultSet rs, int rowNum) throws SQLException
{
            Employee employee = new Employee();
            employee.setId(rs.getLong("id"));
            employee.setFirstName(rs.getString("first_name"));
            employee.setLastName(rs.getString("last_name"));
            employee.setEmailId(rs.getString("email_address"));
            return employee;
        }
    }

    public List < Employee > findAll() {
        return jdbcTemplate.query("select * from employees", new
EmployeeRowMapper());
    }

    public Optional < Employee > findById(long id) {
        return Optional.of(jdbcTemplate.queryForObject("select * from
employees where id=?", new Object[] {
                id
            },
            new BeanPropertyRowMapper < Employee > (Employee.class)));
    }

    public int deleteById(long id) {
        return jdbcTemplate.update("delete from employees where id=?", new
Object[] {
            id
        });
    }

    public int insert(Employee employee) {
        return jdbcTemplate.update("insert into employees (id, first_name,
last_name, email_address) " + "values(?, ?, ?, ?)",
            new Object[] {
                employee.getId(), employee.getFirstName(),
employee.getLastName(), employee.getEmailId()
            });
    }

    public int update(Employee employee) {
        return jdbcTemplate.update("update employees " + " set first_name =
?, last_name = ?, email_address = ? " + " where id = ?",
            new Object[] {
                employee.getFirstName(), employee.getLastName(),
employee.getEmailId(), employee.getId()
            });
    }
}
```

Running

```java
package net.guides.springboot2.jdbc;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

import net.guides.springboot2.jdbc.model.Employee;
import net.guides.springboot2.jdbc.repository.EmployeeJDBCRepository;

@SpringBootApplication
public class Application implements CommandLineRunner {

    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    private EmployeeJDBCRepository employeeRepository;

    @Override
    public void run(String... args) throws Exception {

        logger.info("Inserting -> {}", employeeRepository.insert(new
Employee(10011L, "Ramesh", "Fadatare", "ramesh@gmail.com")));
        logger.info("Inserting -> {}", employeeRepository.insert(new
Employee(10012L, "John", "Cena", "john@gmail.com")));
        logger.info("Inserting -> {}", employeeRepository.insert(new
Employee(10013L, "tony", "stark", "stark@gmail.com")));
```

```
        logger.info("Employee id 10011 -> {}",
employeeRepository.findById(10011L));

        logger.info("Update 10003 -> {}", employeeRepository.update(new
Employee(10011L, "ram", "Stark", "ramesh123@gmail.com")));

        employeeRepository.deleteById(10013L);

        logger.info("All users -> {}", employeeRepository.findAll());
    }

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```