**Spring JDBC**

**Q) How to do CRUD operation in using Spring Jdbc?**

1. **RetrieveAll**

public List<Person> findAll() {

return jdbcTemplate.query("select \* from person", new PersonRowMapper());

}

1. **FindById**

public Person findById(int id) {

return jdbcTemplate.queryForObject("select \* from person where id=?", new Object[] { id },

new BeanPropertyRowMapper<Person>(Person.class));

}

1. **Update**

public int updateById(Person person) {

return jdbcTemplate.update(

"update person "

+ "set name=?,location=? "

+ "where id=?",

new Object[] {person.getName(),person.getLocation(),person.getId() });

}

1. **Insert**

public int insertPerson(Person person) {

return jdbcTemplate.update("insert into person (id,name,location,birth\_date)" + "values(?,?,?,?)",

new Object[] { person.getId(), person.getName(), person.getLocation(),

new Timestamp(person.getBirthDate().getTime()) });

}

}

1. **Delete**

public int deleteById(int id) {

return jdbcTemplate.update("delete from person where id=?",

new Object[] {id});

}

**Custom implementation of the RowMapper**

public Person mapRow(ResultSet rs, int rowNum) throws SQLException {

Person person=new Person();

person.setId(rs.getInt("id"));

person.setName(rs.getString("name"));

person.setLocation(rs.getString("location"));

person.setBirthDate(rs.getTimestamp("birth\_date"));

return person;

}

**Q)What is a RowMapper?**

RowMapper maps the column names of the database to the entity properties.

1)Spring has provided the default RowMapper Implemetation class called BeanPropertyRowMapper.

2)One have to use this RowMapper only when the property name of the entity matches exactly with the database columns.

3) If there is mismatch between the property and database columns then have to use our own Custom RowMapper.

4) Above is the sample code to implement the custom RowMapper in the class.

**Eg: - Below is the sample code for all of the above explanation**

@Repository

public class PersonJdbcDao {

@Autowired

JdbcTemplate jdbcTemplate;

class PersonRowMapper implements RowMapper<Person>{

@Override

public Person mapRow(ResultSet rs, int rowNum) throws SQLException {

Person person=new Person();

person.setId(rs.getInt("id"));

person.setName(rs.getString("name"));

person.setLocation(rs.getString("location"));

person.setBirthDate(rs.getTimestamp("birth\_date"));

return person;

}

}

public List<Person> findAll() {

return jdbcTemplate.query("select \* from person", new PersonRowMapper());

}

public Person findById(int id) {

return jdbcTemplate.queryForObject("select \* from person where id=?", new Object[] { id },

new BeanPropertyRowMapper<Person>(Person.class));

}

public int updateById(Person person) {

return jdbcTemplate.update(

"update person "

+ "set name=?,location=? "

+ "where id=?", new Object[] {person.getName(),person.getLocation(),person.getId() });

}

public int deleteById(int id) {

return jdbcTemplate.update("delete from person where id=?", new Object[] { id });

}

public int insertPerson(Person person) {

return jdbcTemplate.update("insert into person (id,name,location,birth\_date)" + "values(?,?,?,?)",

new Object[] { person.getId(), person.getName(), person.getLocation(),

new Timestamp(person.getBirthDate().getTime()) });

}

}