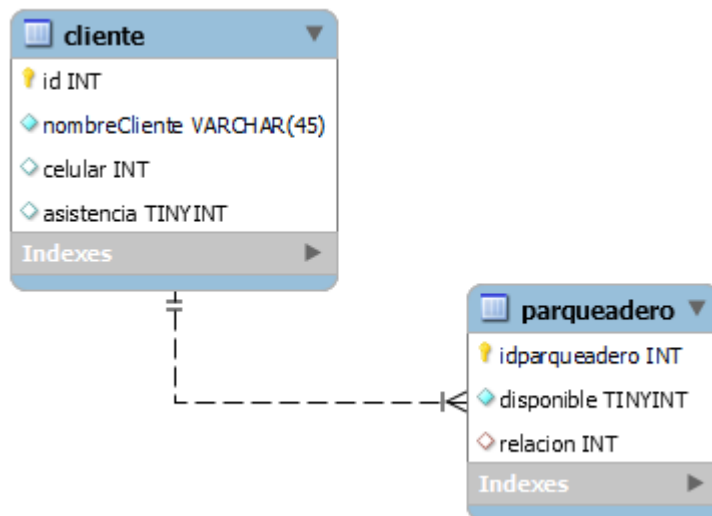


## Spring 3

### 1. Presentación MVC

#### a. Aplicación con persistencia Relacional



Entidades necesarias para crear las bases de datos

```

package modelos.vo;

/**
 *
 * @author edisson
 */
public class Cliente {
    private int id;
    private String nombre;
    private int celular;
    private String password;

    public Cliente(int id, String nombre, int celular) {
        this.id = id;
        this.nombre = nombre;
        this.celular = celular;
    }

    public Cliente(int id, String nombre, int celular, String password) {
        this.id = id;
        this.nombre = nombre;
        this.celular = celular;
        this.password = password;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public int getCelular() {
        return celular;
    }
}

```

trout

```
package modelos.vo;
```

```
/**
```

```
 *
```

```
 * @author edisson
```

```
 */
```

```
public class Parquadero {
```

```
    private int idp;
```

```
    private boolean disponible;
```

```
    private int relacion;
```

```
    public Parquadero(int idp, boolean disponible, int relacion) {
```

```
        this.idp = idp;
```

```
        this.disponible = disponible;
```

```
        this.relacion = relacion;
```

```
    }
```

```
    public int getIdp() {
```

```
        return idp;
```

```
    }
```

```
    public void setIdp(int idp) {
```

```
        this.idp = idp;
```

```
    }
```

```
    public boolean isDisponible() {
```

```
        return disponible;
```

```
    }
```

```
    public void setDisponible(boolean disponible) {
```

```
        this.disponible = disponible;
```

```
    }
```

```
    public int getRelacion() {
```

```
        return relacion;
```

```
    }
```

```
    public void setRelacion(int relacion) {
```

```
        this.relacion = relacion;
```

```
    }
```

```
}
```

```

package configuraciones;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;









/**
 *
 * @author edisson
 */
public class Conexion {
    public Connection getConexion() {

        try {
            Connection conexion = DriverManager.getConnection("jdbc:mysql://localhost:3306/park", "root", "Tirol0.6");
            System.out.println("Conexion establecida");
            return conexion;
        } catch (SQLException err) {
            System.out.println(err+ " Grave");
            return null;
        }
    }
}






```

## b. Pruebas unitarias de la lógica desarrollada

SQL File 3\* x



Limit to 1000 rows



```
1 • INSERT INTO `park`.`parquadero`
2   (`idp`,
3    `disponible`,
4    `relacion`)
5   VALUES
6   ({idp 1},
7    {disponible TRUE},
8    {relacion 3163578490 }});
9
```

<

Output

Action Output

| #   | Time     | Action   | Message           |
|-----|----------|--|-------------------|
| ✓ 1 | 21:22:26 | INSERT INTO `park`.`cliente` (`id`, `nombre`, `celular`, `password`) VALUES (... | 1 row(s) affected |
| ✗ 2 | 21:24:43 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1048. |
| ✗ 3 | 21:26:06 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1452. |
| ✗ 4 | 21:26:52 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1452. |
| ✓ 5 | 21:27:12 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | 1 row(s) affected |

SQL File 3\*

```

1 • INSERT INTO `park`.`parquadero`
2   (`idp`,
3    `disponible`,
4    `relacion`)
5   VALUES
6   ({idp 1},
7    {disponible TRUE},
8    {relacion 3163578490 });
9

```

Output

Action Output

| #   | Time     | Action   | Message           |
|-----|----------|--|-------------------|
| ✓ 1 | 21:22:26 | INSERT INTO `park`.`cliente` (`id`, `nombre`, `celular`, `password`) VALUES (... | 1 row(s) affected |
| ✗ 2 | 21:24:43 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1048. |
| ✗ 3 | 21:26:06 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1452. |
| ✗ 4 | 21:26:52 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | Error Code: 1452. |
| ✓ 5 | 21:27:12 | INSERT INTO `park`.`parquadero` (`idp`, `disponible`, `relacion`) VALUES ({...   | 1 row(s) affected |

## 2. Informe de retrospectiva

los errores en el diseño han hecho que se dificulte mas el desarrollo del producto final, pero nada mas que la practica arranque cada vez más

tomemos decisiones mas acertadas en nuestros desarrollos, practicar, practicar y practicar para tener la posibilidad de pertenecer a un grupo de desarrollo profesional.

### **3. Historias de usuario a desarrollar en el sprint 4. (Trello)**

#### **Historias de usuario**