

**Práctica**

**Programación: Manual técnico**

**2018**

**Mario Burgos Maestre**

Práctica

Programación: Manual técnico

Contenido

[1.1 Introducción 3](#_Toc517569052)

[2.0 Funcionalidad 4](#_Toc517569053)

[2.1 Diagrama E-R 4](#_Toc517569054)

[2.2 Esquema relacional 4](#_Toc517569055)

[2.4 Sentencias SQL 6](#_Toc517569056)

[2.5 Librerías 11](#_Toc517569057)

[3.0 Instalación 12](#_Toc517569058)

[4.0 Clases Java 12](#_Toc517569059)

[1.2 Bibliografía 15](#_Toc517569060)

[Programación (GrupoStudium) 15](#_Toc517569061)

# Introducción

En este documento, describimos varios aspectos para el desarrollo de la aplicación:

Modelo de datos (diagrama E-R, esquema relacional y diagrama en Workbench).

Sentencias SQL de creación de la base de datos y las tablas.

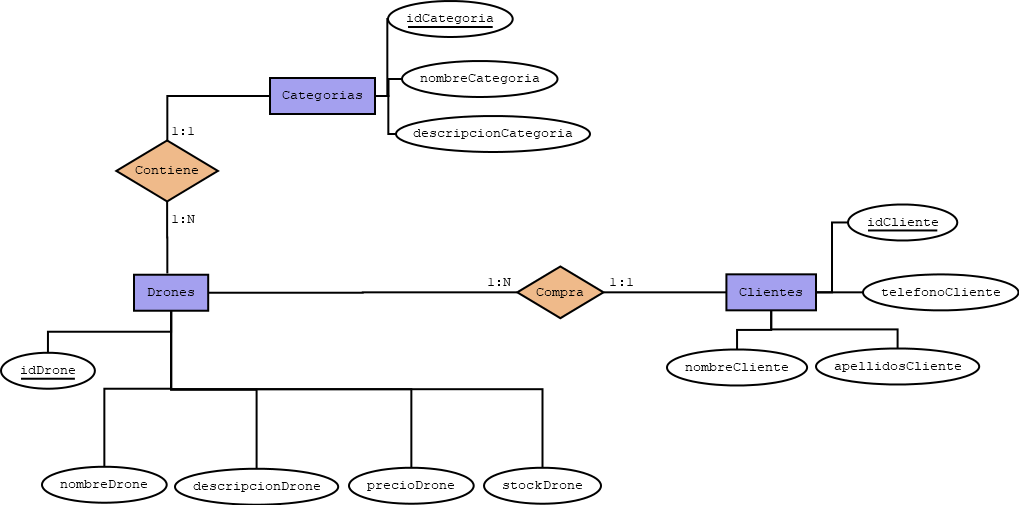
Librerías externas utilizadas para el desarrollo de la aplicación.

Requisitos técnicos imprescindibles para la ejecución del programa.

Explicación de las clases JAVA desarrolladas.

# 2.0 Funcionalidad

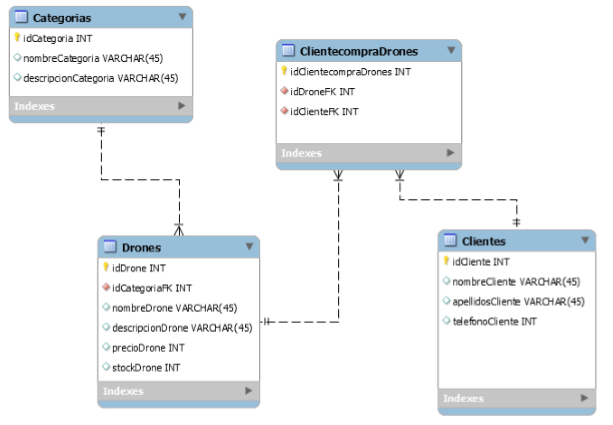
### 2.1 Diagrama E-R



### 2.2 Esquema relacional

|  |  |
| --- | --- |
| Categorias | idCategoria, nombreCantegoria, descripcionCategoria |
| Drones | idDrone, idCategoriaFK, idClienteFK, nombreDrone, descripcionDrone, precioDrone, stockDrone. |
| Clientes | idCliente, nombreCliente, apellidosCliente, teléfonoCliente |
| ClientecompraDrones | idClientecompraDrones, idDroneFK, idClienteFK |

2.3 Diagrama Workbench



### 2.4 Sentencias SQL

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;

-- -----------------------------------------------------

-- Table `mydb`.`Categorias`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Categorias` (

`idCategoria` INT NOT NULL,

`nombreCategoria` VARCHAR(45) NULL,

`descripcionCategoria` VARCHAR(45) NULL,

PRIMARY KEY (`idCategoria`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`Drones`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Drones` (

`idDrone` INT NOT NULL,

`idCategoriaFK` INT NOT NULL,

`nombreDrone` VARCHAR(45) NULL,

`descripcionDrone` VARCHAR(45) NULL,

`precioDrone` INT NULL,

`stockDrone` INT NULL,

PRIMARY KEY (`idDrone`),

INDEX `idCategoriaFK 1\_idx` (`idCategoriaFK` ASC),

CONSTRAINT `idCategoriaFK 1`

FOREIGN KEY (`idCategoriaFK`)

REFERENCES `mydb`.`Categorias` (`idCategoria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`Proveedores`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Proveedores` (

`idProveedor` INT NOT NULL,

`apellidosProveedor` VARCHAR(45) NULL,

`nombreProveedor` VARCHAR(45) NULL,

`correoProveedor` VARCHAR(45) NULL,

`telefonoProveedor` VARCHAR(45) NULL,

PRIMARY KEY (`idProveedor`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`ProveedoresdeDrones`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`ProveedoresdeDrones` (

`ProveedordeDron` INT NOT NULL,

`idProveedorFK` INT NOT NULL,

`idDroneFK` INT NOT NULL,

PRIMARY KEY (`ProveedordeDron`),

INDEX `idDroneFK 3\_idx` (`idDroneFK` ASC),

CONSTRAINT `idProveedorFK 2`

FOREIGN KEY (`idProveedorFK`)

REFERENCES `mydb`.`Proveedores` (`idProveedor`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `idDroneFK 3`

FOREIGN KEY (`idDroneFK`)

REFERENCES `mydb`.`Drones` (`idDrone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`Clientes`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`Clientes` (

`idCliente` INT NOT NULL,

`nombreCliente` VARCHAR(45) NULL,

`apellidosCliente` VARCHAR(45) NULL,

`telefonoCliente` INT NULL,

PRIMARY KEY (`idCliente`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`ClientecompraDrones`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`ClientecompraDrones` (

`idClientecompraDrones` INT NOT NULL,

`idDroneFK` INT NOT NULL,

`idClienteFK` INT NOT NULL,

PRIMARY KEY (`idClientecompraDrones`),

INDEX `idDrones FK2 \_idx` (`idDroneFK` ASC),

INDEX `idCliente FK3\_idx` (`idClienteFK` ASC),

CONSTRAINT `idDrones FK2 `

FOREIGN KEY (`idDroneFK`)

REFERENCES `mydb`.`Drones` (`idDrone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `idCliente FK3`

FOREIGN KEY (`idClienteFK`)

REFERENCES `mydb`.`Clientes` (`idCliente`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`table1`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`table1` (

)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mydb`.`ClientecompraDrones`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`ClientecompraDrones` (

`idClientecompraDrones` INT NOT NULL,

`idDroneFK` INT NOT NULL,

`idClienteFK` INT NOT NULL,

PRIMARY KEY (`idClientecompraDrones`),

INDEX `idDrones FK2 \_idx` (`idDroneFK` ASC),

INDEX `idCliente FK3\_idx` (`idClienteFK` ASC),

CONSTRAINT `idDrones FK2 `

FOREIGN KEY (`idDroneFK`)

REFERENCES `mydb`.`Drones` (`idDrone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `idCliente FK3`

FOREIGN KEY (`idClienteFK`)

REFERENCES `mydb`.`Clientes` (`idCliente`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

### 2.5 Librerías

Para que nuestra aplicación funcione a pleno rendimiento, para ello he utilizado varias librerías externas, que proporcionan una conexión a la base de datos anteriormente creada.

MySQL Connector/J 5.1.35: http://dev.mysql.com/downloads/connector/j/

IText 5.5.5: <http://itextpdf.com/release/iText555>

### 3.0 Instalación

Para poder ejecutar la aplicación correctamente antes debe tener estos requisitos en tu ordenador:

MySQL Server

Base de datos dronesminibd

Java SE Development Kit 8

Archivo JAR de la aplicación

Si cumplimos los requisitos mencionados anteriormente, simplemente tendremos que abrir el terminal y escribir la siguiente sentencia:

**java –jar laberintoJuego.jar**

### 4.0 Clases Java

A continuación haremos una breve explicación sobre las diferentes clases java de nuestra aplicación:

**Inicio.java:** Esta clase es la que contiene el *main* donde se ejecutara el programa. También hace las veces de login, en donde tendremos que identificarnos para poder acceder a la base de datos.

**Sql.java**: Esta clase contiene la conexión a la base de datos dronesminibd.

**Principal.java**: Es nuestro menú principal donde se encuentra las tablas de la base de datos, diferenciamos 3 tipos de tablas**: Clientes, drones y categorias**.

**Vista, controlador, modeloAltaCliente.java**: Estas tres clases java permite realizar una sentencia a la base de datos y realizar un ingreso de un nuevo cliente.

**Vista, controlador, modeloBajaCliente.java:** Estas tres clases java permite realizar una sentencia a la base de datos y eliminar un cliente ya existente dentro de nuestra base de datos.

**Vista, controlador, modeloModificarCliente.java**: Estas tres clases java permite realizar una sentencia a la base de datos y modificar un cliente ya existente.

**Vista, controlador, modeloConsultaCliente.java**: Estas tres clases java permite realizar una sentencia a la base de datos y generar un PDF con los datos de la tabla clientes.

**Vista, controlador, modeloAltaDrones**: Estas tres clases java permite realizar una sentencia a la base de datos y realizar un ingreso de un nuevo drone.

**Vista, controlador, modeloBajaDrone.java:** Estas tres clases java permite realizar una sentencia a la base de datos y eliminar un drone ya existente dentro de nuestra base de datos.

**Vista, controlador, modeloModificarDrone.java**: Estas tres clases java permite realizar una sentencia a la base de datos y modificar un drone ya existente.

**Vista, controlador, modeloConsultaDrone.java**: Estas tres clases java permite realizar una sentencia a la base de datos y generar un PDF con los datos de la tabla drones.

**Vista, controlador, modeloAltaCategorias**: Estas tres clases java permite realizar una sentencia a la base de datos y realizar un ingreso de una nueva categoria.

**Vista, controlador, modeloBajaCategoria.java:** Estas tres clases java permite realizar una sentencia a la base de datos y eliminar una categoría ya existente dentro de nuestra base de datos.

**Vista, controlador, modeloModificarCategoria.java**: Estas tres clases java permite realizar una sentencia a la base de datos y modificar una categoría ya existente.

**Vista, controlador, modeloConsultaCategoria.java**: Estas tres clases java permite realizar una sentencia a la base de datos y generar un PDF con los datos de la tabla categorias.

# 1.2 Bibliografía

### Programación (GrupoStudium)