Integrador

Diseño de Base de Datos I

Profesor:

- Gabriel Arenas

Integrantes:

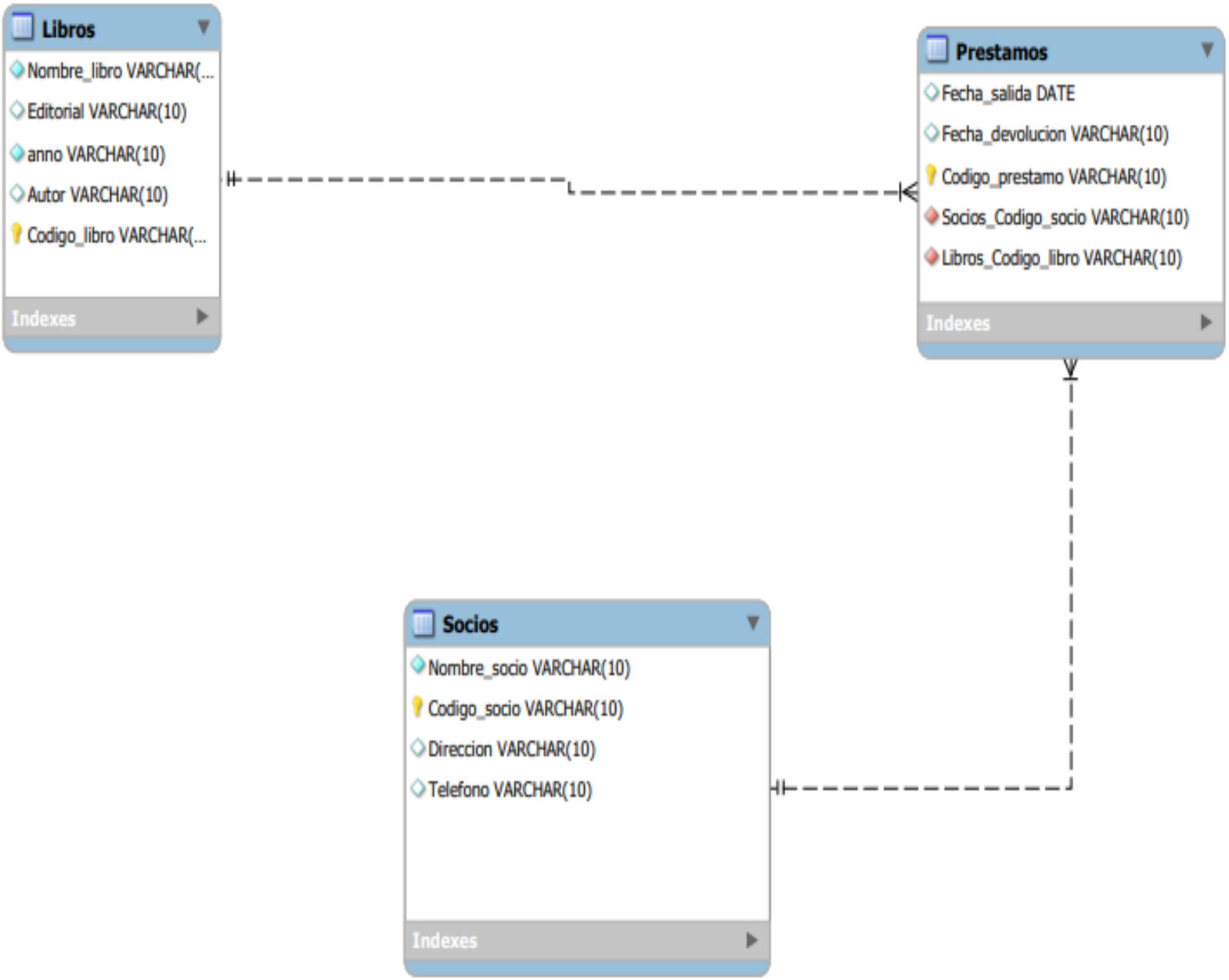
- Merenda Francisco
- Michaux Juan Martin
- Gallo Emanuel
- Ulzurrum Santiago

TEMA II

Enunciado elegido: Movimientos de préstamos de libros en una biblioteca

Una biblioteca de barrio tiene libros que presta a sus socios.
Los libros ingresan a la biblioteca generalmente por donaciones, pero también se compran. Se
desea registrar los movimientos de los libros que pueden ser entradas y salidas.
Las entradas corresponden a devolución de socios o adquisición (por donación o compra).
Las salidas corresponden a préstamos en los que debemos registrar la fecha de devolución o
bajas (por roturas, pérdidas o no devolución).
Se pide mantener los libros (ABM maestro de libros), los socios (ABM de socios) y autores
(ABM de autores) y poder registrar los movimientos de los libros.
Emitir un listado del estado de los libros identificando de los prestados quien los tiene y la fecha de devolución.
Nota: Con la sigla ABM nos referimos a las altas, bajas y modificaciones de los registros de la
correspondiente tabla

Actividad 1-2



ACTIVIDAD 3

```
use mydb;
-- -----------------------------------------------------
-- Table `mydb`.`socios`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`socios` ;

CREATE TABLE IF NOT EXISTS `mydb`.`socios` (
  `Nombre_socio` VARCHAR(10) NOT NULL,
  `Codigo_socio` VARCHAR(10) NOT NULL,
  `Direccion` VARCHAR(10) NULL,
  `Telefono` VARCHAR(10) NULL,
  PRIMARY KEY (`Codigo_socio`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`libros`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`libros` ;

CREATE TABLE IF NOT EXISTS `mydb`.`libros` (
  `Nombre_libro` VARCHAR(10) NOT NULL,
  `Editorial` VARCHAR(10) NULL,
  `anno` VARCHAR(10) NOT NULL,
  `Autor` VARCHAR(10) NULL,
  `Codigo_libro` VARCHAR(10) NOT NULL,
  PRIMARY KEY (`Codigo_libro`))
ENGINE = InnoDB;

CREATE UNIQUE INDEX `Curso_UNIQUE` ON `mydb`.`libros` (`Nombre_libro` ASC)
VISIBLE;

CREATE UNIQUE INDEX `Division_UNIQUE` ON `mydb`.`libros` (`Editorial` ASC) VISIBLE;


-- -----------------------------------------------------
-- Table `mydb`.`prestamos`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`prestamos` ;

CREATE TABLE IF NOT EXISTS `mydb`.`prestamos` (
  `Fecha_salida` DATE NULL,
  `Fecha_devolucion` VARCHAR(10) NULL,
  `Codigo_prestamo` VARCHAR(10) NOT NULL,
  `Socios_Codigo_socio` VARCHAR(10) NOT NULL,
```

```
  `Libros_Codigo_libro` VARCHAR(10) NOT NULL,
  PRIMARY KEY (`Codigo_prestamo`),
  CONSTRAINT `fk_Prestamos_Socios1`
    FOREIGN KEY (`Socios_Codigo_socio`)
    REFERENCES `mydb`.`Socios` (`Codigo_socio`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Prestamos_Libros1`
    FOREIGN KEY (`Libros_Codigo_libro`)
    REFERENCES `mydb`.`Libros` (`Codigo_libro`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE INDEX `fk_Prestamos_Socios1_idx` ON `mydb`.`prestamos`
(`Socios_Codigo_socio` ASC) VISIBLE;

CREATE INDEX `fk_Prestamos_Libros1_idx` ON `mydb`.`prestamos` (`Libros_Codigo_libro`
ASC) VISIBLE;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

ACTIVIDAD 4

MENÚ(CODIGO MENÚ)

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import os
from Libros import load_books, see_books
from Clientes import load_partners
from Prestamos import load_loans

def menu():
    """Función que limpia la pantalla y muestra nuevamente el menu
os.system('clear')
    Si el sistema es Windows se usa cls de otro modo se usa clear
    os.name trae el nombre del kernel. NT para windows y POSIX para
Mac o Linux    " ┌ ─ ┐ | └ ─ ┘ "     ├ ┤
    """
    borrar = 'cls' if os.name == 'nt' else 'clear'
    os.system(borrar)
    print("\t")
    print("└──────────────────────────────── ")
```

```
    print("|SELECCIONE UNA OPCION                |")
    print("└─────────────────────────────────── ")
    print("| 1 - CARGAR LIBRO                    |")
    print("| 2 - CARGAR SOCIOS                   |")
    print("| 3 - CARGAR PRESTAMO                 |")
    print("| 4 - MOSTRAR LIBROS DISPONOBLES      |")
    print("| 5 - PARA SALIR                      |")
    print("└─────────────────────────────────── ")
    print(" ☺ ")
    print("┌──────────────────────────┐")
    print("| Indica opción elegida    |")
    print("└──────────────────────────┘")
```

```
while True:
    # Mostramos el menu
    menu()
    # solicituamos una opción al usuario
    option = input()
    if option == "1":
        print("")
        input("Has pulsado la opción 1...\npulsa una tecla para
continuar")
        load_books()
    elif option == "2":
        print("")
        input("Has pulsado la opción 2...\npulsa una tecla para
continuar")
        load_partners()
    elif option == "3":
        print("")
        input("Has pulsado la opción 3...\npulsa una tecla para
continuar")
        load_loans()
    elif option == "4":
        print("")
        input("Has pulsado la opción 4...\npulsa una tecla para
continuar")
        see_books()
    elif option == "5":
        break
    else:
        print("")
        input("No has pulsado ninguna opción correcta...\npulsa una
tecla para continuar")
```

CLIENTE(CODIGO CLIENTE)

```
import pymysql

def load_partners():
    tupla = []
    db = pymysql.connect(host="localhost", user="root",
passwd="Imperio Romano0008", db="mydb")
    cursor = db.cursor()
    name_partner = str(input("Ingrese nombre del socio: "))
    adress_partner = str(input("Ingrese dirección del socio: "))
    telephone_partner = str(input("Ingrese teléfono del socio: "))

    tupla.append(name_partner)
    tupla.append(adress_partner)
    tupla.append(telephone_partner)

    tupla = tuple(tupla)

    sql = "INSERT INTO socios (Nombre_socio, direccion, telefono)
VALUES {};".format(tupla)
    try:
        # Ejecutamos el comando SQL
        cursor.execute(sql)
        # Se confirman Commit los cambios a la base de dato
        db.commit()
    except:
        # Se deshacen Rollback los cambios si hay errores
        db.rollback()
        # nos desconectamos del servidor
    db.close()
```

LIBROS(CODIGO LIBROS)

```
import pymysql

def load_books():
    db = pymysql.connect(host="localhost", user="root",
passwd="Imperio Romano0008", db="mydb")
    cursor = db.cursor()
    name_book = str(input("Ingrese nombre del libro: "))
```

```
    editorial = str(input("Ingrese editorial del libro: "))
    isbn = str(input("Ingrese ISBN del libro: "))
    year_book = str(input("Ingrese año del libro: "))
    autor_book = str(input("Ingrese autor del libro: "))

    tupla = []

    tupla.append(name_book)
    tupla.append(editorial)
    tupla.append(year_book)
    tupla.append(autor_book)
    tupla.append(isbn)

    tupla = tuple(tupla)

    sql = "INSERT INTO libros (nombre_libro, editorial, anio,
autor, codigo_libro) VALUES {};".format(tupla)
    try:
        # Ejecutamos el comando SQL
        cursor.execute(sql)
        # Se confirman Commit los cambios a la base de dato
        db.commit()
    except:
        # Se deshacen Rollback los cambios si hay errores
        db.rollback()
        # nos desconectamos del servidor
    db.close()

def see_books():
    db = pymysql.connect(host="localhost", user="root",
passwd="Imperio Romano0008", db="mydb")
    cursor = db.cursor()
    cursor.execute("SELECT * FROM libros")
    consulta = cursor.fetchall()
    for i in consulta:
        print(i)
```

PRÉSTAMO(CODIGO PRÉSTAMO)

```
import pymysql

def load_loans():
    tupla = []
    db = pymysql.connect(host="localhost", user="root",
passwd="Imperio Romano0008", db="mydb")
    cursor = db.cursor()
    exit_date = str(input("Ingrese fecha de salida: "))
```

```
    return_date = str(input("Ingrese fecha de devolucion: "))

    tupla.append(exit_date)
    tupla.append(return_date)

    tupla = tuple(tupla)

    sql = "INSERT INTO prestamos (Fecha_salida, Fecha_devolucion)
VALUES {};".format(tupla)
    try:
        # Ejecutamos el comando SQL
        cursor.execute(sql)
        # Se confirman Commit los cambios a la base de dato
        db.commit()
    except:
        # Se deshacen Rollback los cambios si hay errores
        db.rollback()
        # nos desconectamos del servidor
    db.close()
```

TEST(CÓDIGO TEST):

```
def cuenta_registros01():
    import pymysql

    # import PyRsa
    # Abrir Conexión a la Base de Datos
    db = pymysql.connect(host="127.0.0.1", user="root",
passwd="Imperio Romano0008", db="mydb")
    # Preparar objeto cursor con el método cursor()
    cursor = db.cursor()
    # ejecutamos una consulta SQL query con el método execute().
    cursor.execute("SELECT count(*) from libros")
    # Obtenemos resultado de una fila con el método fetchone().
    data = cursor.fetchone()
    print("Nro de Registros en libros: %s " % data)
    # nos desconectamos del servidor
    db.close()

cuenta_registros01()
```