# *Base de Datos – Programmasy*

**<u>Especificación de las tablas:</u>**

Tabla: Usuarios

- UsserID INT PRIMARY KEY
- UsserName VARCHAR (80)
- Email VARCHAR (100)
- Phone VARCHAR (50)
- EstadoID INT
- MatricID INT
- RegistrarID INT

Tabla: Estados

- EstadoID INT PRIMARY KEY
- StatusUsser VARCHAR (20)

Tabla: Cursos

- CoursesID INT PRIMARY KEY
- CourseName VARCHAR (50)
- CourseDescription VARHCAR (100)
- CategoryID INT
- LevelID INT

Tabla: Matriculación

- MatricID INT PRIMARY KEY
- Enrolled INT
- CoursesID INT

Tabla: Niveles

- LevelID INT PRIMARY KEY
- LevelCourse VARCHAR (10)

Tabla: Registro

- RegistrarID INT PRIMARY KEY
- NumParticipants INT
- EventsID INT

Tabla: Eventos

- EventsID INT PRIMARY KEY
- EventName VARCHAR (100)
- EventDescription VARCVHAR (200)
- EventDate DATE
- VersusID INT

Tabla: Versus

- VersusID INT PRIMARY KEY
- VersusParticipants VARCHAR (100)
- VersusCode CHAR (30)
- VersusTime TIME

Tabla: Categorias

- CategoryID INT PRIMARY KEY
- CategoryName VARCHAR (100)
- SuppliersID INT

Tabla: Proveedores

- SuppliersID INT PRIMARY KEY
- SuppliersName VARCHAR (100)
- ProductName VARCHAR (100)
- CategoryProduct VARCHAR (100)

Tabla: Publicidad

- PublicID INT PRIMARY KEY
- PublicDescription VARCHAR (100)
- Payment FLOAT

**Scrip:**

```
create database Programmasy;
use Programmasy;

create table Usuarios(
UsserID INT PRIMARY KEY, UsserName VARCHAR (80), Email VARCHAR
(100), Phone VARCHAR (50), StatusID INT, MatricID INT, RegistrarID INT
);

create table Estados(
StatusID INT PRIMARY KEY, StatusUsser VARCHAR (20)
);

create table Cursos(
CoursesID INT PRIMARY KEY, CourseName VARCHAR
(50),CourseDescription VARCHAR (100),  CategoryID INT, LevelID INT,
```

```
);

create table Matriculados(
MatricID INT PRIMARY KEY, Enrolled INT, CoursesID INT
);


create table Niveles(
LevelID INT PRIMARY KEY, LevelCourse VARCHAR (10)
);

Create table Categorias(
CategoryID INT PRIMARY KEY, CategoryName VARCHAR (100),
SuppliersID INT
);

Create table Registro(
RegistrarID INT PRIMARY KEY, ParticipannNumber INT, EventsID INT
);

create table Eventos(
EventsID INT PRIMARY KEY, EventName VARCHAR (100),
EventDescription VARCHAR (200), EventDate DATE, VersusID INT
);

create table Versus(
VersusID INT PRIMARY KEY, VersusParticipants VARCHAR (100),
VersusCode CHAR (3), VersusTime TIME
);

create table Proovedores(
SuppliersID INT PRIMARY KEY, SuppliersName VARCHAR (100),
ProductName VARCHAR (100), ProductDescription VARCHAR (100)
);
```

```
Create table Publicidad(
PublicID INT PRIMARY KEY, PublicDescription VARCHAR (100), Payment
FLOAT
);
```

**Relación entre tablas:**

```
ALTER TABLE Categorias
ADD CONSTRAINT fk_ASuppliers FOREIGN KEY (SuppliersID)
REFERENCES Proovedores (SuppliersID);

ALTER TABLE Usuarios
ADD CONSTRAINT fk_AStatus FOREIGN KEY (StatusID) REFERENCES
Estados(StatusID);

ALTER TABLE Usuarios
ADD CONSTRAINT fk_ARegistrar FOREIGN KEY (RegistrarID)
REFERENCES Registro (RegistrarID);

ALTER TABLE Usuarios
ADD CONSTRAINT fk_AMatric FOREIGN KEY (MatricID) REFERENCES
Matriculacion (MatricID);

ALTER TABLE Cursos
ADD CONSTRAINT fk_ALevel FOREIGN KEY (LevelID) REFERENCES
Niveles(LevelID);
```

```
ALTER TABLE Cursos
ADD CONSTRAINT fk_ACategory FOREIGN KEY (CategoryID)
REFERENCES Categorias (CategoryID);

ALTER TABLE Eventos
ADD CONSTRAINT fk_AVersus FOREIGN KEY (VersusID) REFERENCES
Versus(VersusID);

ALTER TABLE Matriculados
ADD CONSTRAINT fk_ACourses FOREIGN KEY (CoursesID) REFERENCES
Cursos (CoursesID);

ALTER TABLE Registro
ADD CONSTRAINT fk_AEvents FOREIGN KEY (EventsID) REFERENCES
Eventos (EventsID);
```

**PROCEDIMIENTO ALMACENADO:**

1- DELIMITER $$
CREATE PROCEDURE sp_insertar_estudiante(

IN ins_UserName VARCHAR(80),

IN ins_Email VARCHAR(100),
IN ins_Phone VARCHAR(50)
)
BEGIN
INSERT INTO usuarios(UserName, Email, Phone)
VALUES(ins_UserName, ins_Email, ins_Phone);
END $$
DELIMITER ;

/******************/
PROCEDIMIENTO ALMACENADO PARA REALIZAR CONSULTA DE USUARIO O ESTUDIANTE POR MEDIO DE ID

DELIMITER $$
CREATE PROCEDURE sp_seleccionar_estudiante_por_id(
IN ins_UserID INT)
BEGIN
SELECT * FROM usuarios
WHERE UserID = ins_UserID;

END $$

DELIMITER ;

/******************/

PROCEDIMIENTO ALMACENADO PARA REALIZAR LA CONSULTA DE LOS USUARIOS REGISTRADOS EN LA TABLA

DELIMITER $$

```
CREATE PROCEDURE sp_seleccionar_usuarios()
BEGIN
SELECT UserID, UserName FROM usuarios;
END $$
DELIMITER ;
```

/*****************/

PROCEDIMIENTO ALMACENADO PARA REALIZAR LA ACTUALIZACIÒN DE DATOS DE UN USUARIO O ESTUDIANTE

```
DELIMITER $$
CREATE PROCEDURE sp_actualizar_usuario(
 IN ins_UserID INT(11),
 IN ins_UserName VARCHAR(80),
 IN ins_Email VARCHAR(100),
 IN ins_Phone VARCHAR(50),
 IN ins_StatusID INT(11),
 IN ins_MatricID INT(11),
 IN ins_RegistrarID INT(11)
)
BEGIN
UPDATE usuarios
SET
UserID=ins_UserID,
UserName=ins_UserName,
Email=ins_Email,
Phone=ins_Phone,
StatusID=ins_StatusID,
MatricID=ins_MatricID,
RegistrarID=ins_RegistrarID
WHERE UserID=ins_UserID;
END $$
DELIMITER ;
```

/******************/

PROCEDIMIENTO ALAMACENADO PARA REALIZAR LA ELIMINACIÒN DE UN USUARIO

```
DELIMITER $$
CREATE PROCEDURE sp_eliminar_usuario_por_id(
 IN ins_UserID INT
)
BEGIN
DELETE FROM usuarios WHERE UserID=ins_UserID;
END $$
DELIMITER ;
```