BASE DE DATOS II

EVALUACION PROCESUAL
HITO 4

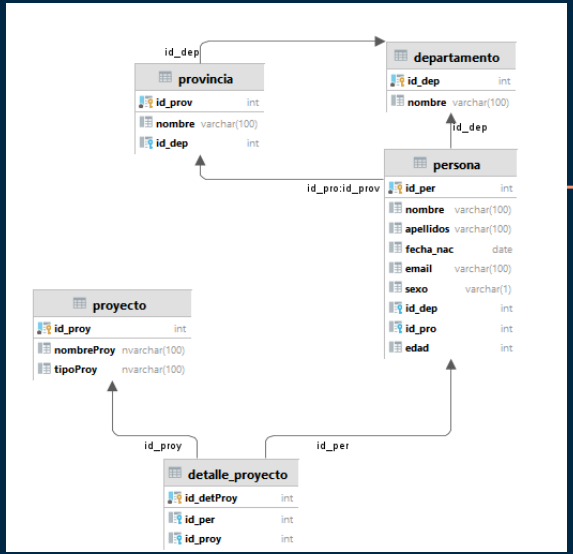
JOSÉ FÉLIX SARMIENTO MAMANI

1.Diseño de base de datos

1.1. Adjuntar SQL CODE con la que se genera la BBD. Nombre: ONG_H4

CREATE TABLE departamento (id_dep INT IDENTITY PRIMARY KEY , nombre VARCHAR (100)) CREATE TABLE provincia (id_prov INT IDENTITY PRIMARY KEY , nombre VARCHAR(100), id_dep INT, FOREIGN KEY (id_dep) REFERENCES departamento (id_dep))
CREATE TABLE persona (id_per INT IDENTITY PRIMARY KEY , nombre VARCHAR(100), apellidos VARCHAR (100), fecha_nac date, edad int, email VARCHAR (100), sexo varchar, id_dep INT, id_pro INT, FOREIGN KEY (id_dep) REFERENCES departamento (id_dep), FOREIGN KEY (id_pro) REFERENCES provincia (id_prov))
CREATE TABLE proyecto (id_proy INT IDENTITY PRIMARY KEY , nombreProy NVARCHAR(100), tipoProy NVARCHAR (100))
CREATE TABLE detalle_proyecto (id_detProy INT IDENTITY PRIMARY KEY , id_per INT, id_proy INT, FOREIGN KEY (id_per) REFERENCES persona(id_per), FOREIGN KEY (id_proy) REFERENCES proyecto(id_proy))

1.2. Modelo lógico de la BBDD creada



2.Manejo de vistas

2.1. Mostrar las personas que viven en Cochabamba

SELECT per.* FROM persona AS per, departamento AS dep WHERE dep.nombre='Cochabamba'

2.2. Mostrar nombre de la persona y el proyecto donde trabajan

SELECT per.nombre, per.apellidos, proy.nombreProy FROM persona AS per, proyecto AS proy

1.Manejo de vistas

2.3. Resolver la consigna

```
CREATE OR ALTER  VIEW prueba AS
SELECT proy.*, departamento_aplicarse=
CASE WHEN proy.tipoProy ='TIPO_A'
THEN 'CBB' WHEN proy.tipoProy ='TIPO_B'
THEN 'LPZ' WHEN proy.tipoProy ='TIPO_C'
THEN 'SCZ' ELSE 'EN PROCESO DE
ANALISIS' END FROM proyecto AS proy;
SELECT * FROM prueba
```

2.4.Crear una vista cualquiera que muestre 5 columnas

```
CREATE OR ALTER  VIEW
[dbo].[pruebaColumna] AS SELECT per.nombre,
per.apellidos, per.edad, per.sexo,
proy.nombreProy AS PROYECTO, dep.nombre AS
DEPARTAMENTO FROM persona AS per,
departamento AS dep INNER JOIN proyecto proy
on proy.id_proy = proy.id_proy SELECT * FROM
dbo.pruebaColumna
```

3.Manejo de funciones

3.1. Procedimiento cuantos proyecto distintos hay.

```
CREATE OR ALTER  FUNCTION canProy() RETURNS INT AS BEGIN DECLARE @quantityPro int =0 SET @quantityPro = (SELECT COUNT(proy.tipoProy) FROM proyecto AS proy) RETURN @quantityPro end; SELECT  dbo.canProy() AS Cantidad_De_Proyectos
```

3.2.Funcion que genere los primero N números impares

```
CREATE OR ALTER  FUNCTION numImpar(@parametro1 INT) RETURNS VARCHAR(100) AS BEGIN DECLARE @nuevo INT ; DECLARE @respuesta VARCHAR(100) = ''; DECLARE @contador INTEGER = 1; SET @nuevo= @parametro1*2; WHILE @contador <= @nuevo BEGIN SET @respuesta = CONCAT(@respuesta, @contador, ', '); SET @contador = @contador + 2; END; RETURN @respuesta; end; SELECT dbo.numImpar(4) AS Resultado
```

3.Manejo de funciones

3.3. Funcion para insertar un registro a la tabla persona

```
CREATE OR ALTER FUNCTION insertarRegistro(@name VARCHAR (50),@lastname VARCHAR (50),@fecha DATE ,@edad INT,@email VARCHAR(100),@sexo VARCHAR,@dep INT,@prov INT) RETURNS VARCHAR(100) AS BEGIN INSERT INTO persona (nombre, apellidos, fecha_nac, edad, email, sexo, id_dep, id_pro) VALUES (@name,@lastname,@fecha,18,'thedarkomgf@gmail.com','M',1,1) RETURN 'se insertó satisfactoriamente el registro. ' END
```

3.4. Función cualquiera, de dos parametros

```
CREATE OR ALTER  FUNCTION dbo.potenciaNumero (@num1 INT, @num2 INT) RETURNS INT AS BEGIN DECLARE @resultado INT=1; DECLARE @aux INT; DECLARE @i int = 0 WHILE @i < @num2 BEGIN SET @i = @i + 1 SET @resultado=@resultado* @num1 END RETURN @resultado END SELECT dbo.potenciaNumero(4, 3) AS Resultado_Potencia;
```