

BASE DE DATOS

MELANIE INGRID VILLCA

LABORATORIO 2 H2

24 de agosto de 2022

```
CREATE DATABASE function aggregation;
```

```
USE function aggregation;
```

```
CREATE TABLE estudiantes
```

```
(
```

```
id est INTEGER AUTO INCREMENT PRIMARY KEY NOT NULL,
```

```
nombres VARCHAR(50),
```

```
apellidos VARCHAR(50),
```

```
edad INTEGER,
```

```
gestion INTEGER,
```

```
fono INTEGER,
```

```
email VARCHAR(100),
```

```
direccion VARCHAR(100),
```

```
genero VARCHAR(10)
```

```
);
```

```
SELECT est.*
```

```
FROM estudiantes AS est;
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email,
```

```
direccion, genero) VALUES
```

```
    ('Miguel' , 'Gonzales Veliz', 20, 2832115, 'miguel@gmail.com',  
'Av. 6 de Agosto', 'masculino'),
```

```
    ('Sandra' , 'Mavir Uria', 25, 2832116, 'sandra@gmail.com', 'Av.  
6 de Agosto', 'femenino'),
```

```
    ('Joel' , 'Adubiri Mondar', 30, 2832117, 'joel@gmail.com', 'Av.  
6 de Agosto', 'masculino'),
```

```
        ('Andrea' , 'Arias Ballesteros', 21, 2832118,  
'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino'),  
        ('Santos' , 'Montes Valenzuela', 24, 2832119,  
'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');
```

```
#Crear la consulta SQL  
#que modifique el campo gestion  
# y que asigne a todos el valor 2022
```

```
UPDATE estudiantes  
SET gestion = '2022'  
WHERE id est > 0;
```

```
CREATE TABLE materias  
(  
id mat INTEGER AUTO INCREMENT PRIMARY KEY NOT NULL,  
nombre mat VARCHAR(100),  
cod mat VARCHAR(100)  
);
```

```
CREATE TABLE inscripcion  
(  
id ins INTEGER AUTO INCREMENT PRIMARY KEY NOT NULL,  
id est INT NOT NULL,  
id mat INT NOT NULL,  
semestre VARCHAR(20),  
gestion INTEGER,  
FOREIGN KEY (id est) REFERENCES estudiantes (id est),  
FOREIGN KEY (id mat) REFERENCES materias (id mat)  
);
```

```
INSERT INTO materias (nombre mat, cod mat) VALUES  
('Introduccion a la Arquitectura', 'ARQ-101'),  
('Urbanismo y Diseno', 'ARQ-102'),  
('Dibujo y Pintura Arquitectonico', 'ARQ-103'),  
('Matematica discreta', 'ARQ-104'),  
('Fisica Basica', 'ARQ-105');
```

```
INSERT INTO inscripcion (id est, id mat, semestre, gestion) VALUES
(1, 1, '1er Semestre', 2015),
(1, 2, '2do Semestre', 2015),
(2, 4, '1er Semestre', 2016),
(2, 3, '2do Semestre', 2016);
```

```
INSERT INTO inscripcion (id est, id mat, semestre, gestion) VALUES
(3, 3, '2do Semestre', 2017),
(3, 1, '3er Semestre', 2017),
(4, 4, '4to Semestre', 2017),
(5, 5, '5to Semestre', 2017);
```

```
SELECT est.*
FROM materias AS est;
```

```
SELECT est.*
FROM inscripcion AS est;
```

```
# Mostrar los nombres y apellidos de los estudiantes inscritos en la
materia
```

```
# ARQ-104, adicionalmente mostrar el nombre de la materia.
```

```
SELECT est.nombres, est.apellidos, mat.nombre mat
FROM inscripcion AS ins
    INNER JOIN estudiantes AS est ON ins.id est=est.id est
    INNER JOIN materias AS mat ON ins.id mat=mat.id mat
WHERE mat.cod mat = 'ARQ-104';
```

```
#Contar cuantos registros tiene la tabla estudiantes
```

```
SELECT COUNT(est.id est) AS 'Cantidad de estudiantes'
FROM estudiantes AS est;
```

```
#Mostrar el promedio de edad en la tabla estudiantes.
```

```
SELECT AVG(est.edad)
FROM estudiantes AS est;
```

```
#Mostrar la máxima edad que se tiene en la tabla estudiantes.
```

```
SELECT MAX(est.edad)
FROM estudiantes AS est;
```

#Mostrar la mínima edad que se tiene en la estudiantes.

```
SELECT MIN(est.edad)
FROM estudiantes AS est;
```

#Determinar la maxima edad de los estudiantes

#cuyo genero sea 'masculino'/'femenino'

#y ademas la edad sea mayor de 20

#Manejo de funciones

#Crear una función que devuelve el máximo valor del campo edad en la tabla estudiantes.

```
CREATE FUNCTION get_max_edad()
RETURNS INTEGER
BEGIN
    RETURN 10;
END;
```

```
SELECT get_max_edad();
```

```
CREATE OR REPLACE FUNCTION get_max_edad()
RETURNS INTEGER
BEGIN
    RETURN (
        SELECT MAX(est.edad)
        FROM estudiantes AS est
    );
END;
```

```
SELECT get_max_edad() AS MaxEdad;
```

#Crear una función que obtenga la menor edad de los estudiantes.

```
CREATE FUNCTION min_edad()
RETURNS INTEGER
BEGIN
    RETURN (
        SELECT MIN(est.edad)
```

```

        FROM estudiantes AS est
    );
END;
SELECT min_edad() AS MinEdad;

```

#Crear una función que obtenga el promedio de las edades.

```

CREATE FUNCTION prom_edad()
    RETURNS INTEGER
    BEGIN
        RETURN (
            SELECT AVG(est.edad)
            FROM estudiantes AS est
        );
    END;
SELECT prom_edad() AS AvgEdad;

```

#Crear una función que obtenga la mayor edad de los estudiantes (cuyo sexo seas masculino o femenino).

```

CREATE FUNCTION get_max_edad_varones()
    RETURNS INTEGER
    BEGIN
        RETURN (
            SELECT MAX(est.edad)
            FROM estudiantes AS est
            WHERE est.genero = 'masculino'
        );
    END;

```

#Mostrar el registro de la tabla estudiantes (nombre y apellidos) donde cuyo id est sea el máximo.

```

CREATE FUNCTION max_id_tabla_est() RETURNS integer
BEGIN
    RETURN (
        SELECT MAX(est.id est)
        FROM estudiantes AS est
    );

```

```
end;
```

```
SELECT max_id_tabla_est() as ID_MAX;
```

```
SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE est.id_est = max_id_tabla_est();
```

```
#Mostrar todos los registros de la tabla estudiantes (nombres y
apellidos)
```

```
#si la suma de las edades de los estudiantes masculino/femenino sea
par.
```

```
#Crear una función que obtenga la suma de las edades de los
estudiantes masculino/femenino
```

```
CREATE FUNCTION suma_edades() RETURNS INTEGER
BEGIN
    RETURN (
        SELECT SUM(est.edad)
        FROM estudiantes AS est
        WHERE est.genero = 'masculino'
    );
end;
```

```
SELECT suma_edades();
```

```
SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE suma_edades() % 2 = 0;
```

```
#Mostrar todos los registros de la tabla estudiantes (nombres y
apellidos)
```

```
# si la suma de las edades de las estudiantes femeninos sea par.
```

```
#Crear una función que obtenga la suma de las edades de las
estudiantes femeninos.
```

```
CREATE FUNCTION suma_edades_femenino() RETURNS INTEGER
BEGIN
```

```

RETURN (
    SELECT SUM(est.edad)
    FROM estudiantes AS est
    WHERE est.genero = 'femenino'
);
end;

```

```
SELECT suma_edades_femenino();
```

```

SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE suma_edades_femenino() % 2 = 0;

```

#Volver parametrizable la anterior función.

```

CREATE FUNCTION suma_edades v1(genero valor VARCHAR(10)) RETURNS
INTEGER
BEGIN
    RETURN (
        SELECT SUM(est.edad)
        FROM estudiantes AS est
        WHERE est.genero = genero valor
    );
END;

```

```
SELECT suma_edades v1('masculino');
```

```

SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE suma_edades v1('masculino') % 2 = 0;

```

#Volver parametrizable la anterior función. V2

```

CREATE FUNCTION suma_edades v2(genero VARCHAR(50)) RETURNS INTEGER
BEGIN
    #Esto sirve para declarar una variable en una funcion
    DECLARE sumaEdad INTEGER DEFAULT 0;

```

```

SELECT SUM(est.edad) INTO sumaEdad
FROM estudiantes AS est
WHERE est.genero = genero;

```

```

RETURN sumaEdad;
END;

```

```

SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE suma_edades_v2('masculino') %2 = 0;

```

```

CREATE OR REPLACE FUNCTION get_promedio_v2(genero VARCHAR(10))
RETURNS INTEGER

```

```

BEGIN
    DECLARE promedio REAL DEFAULT 0;

```

```

    SELECT AVG(est.edad) INTO promedio
    FROM estudiantes AS est
    WHERE est.genero = genero;

```

```

    RETURN promedio;

```

```

END;
SELECT get_promedio_v2('femenino');

```

```

SELECT est.nombres, est.apellidos
FROM estudiantes AS est
WHERE get_promedio_v2('masculino') % 2 = 0;

```

```

#Crear una función que permita concatenar dos columnas.
#Concatena el nombre y apellidos de la tabla estudiante.
#El nombre de la función deberá ser getNombreCompleto
#La función deberá recibir 2 parámetros (nombre y apellidos)

```

```

create function getNombreCompleto(par1 varchar(25),par2 varchar(25))
returns varchar(50)
begin

```



```
declare concatenando varchar(50) DEFAULT '';  
set concatenando = CONCAT(par1, ' - ', par2);  
return concatenando;  
end;
```

```
select getNombreCompleto('Pepito', 'Pep');
```

```
SELECT getNombreCompleto(est.nombres, est.apellidos) AS Fullname  
FROM estudiantes AS est;
```