



Examen Resuelto Hito

3 y 4

- **Nombre Completo:** Diego Emiliano Rivera Tapia
- **Asignatura:** BASE DE DATOS II
- **Carrera:** INGENIERÍA DE SISTEMAS
- **Paralelo:** BDA (1)
- **Docente:** Lic. William R. Barra Paredes
- **Fecha:** 09/12/2019

COCHABAMBA-BOLIVIA

2019

- **Pregunta 1:** Generar la serie fibonacci.
 - El objetivo es sumar todos los números de la serie fibonacci desde una cadena.
 - Es decir usted tendrá solo la cadena generado con los primeros N números de la serie fibonacci y a partir de ellos deberá sumar los números de esa serie.
 - Puede utilizar el siguiente código para generar la serie fibonacci.

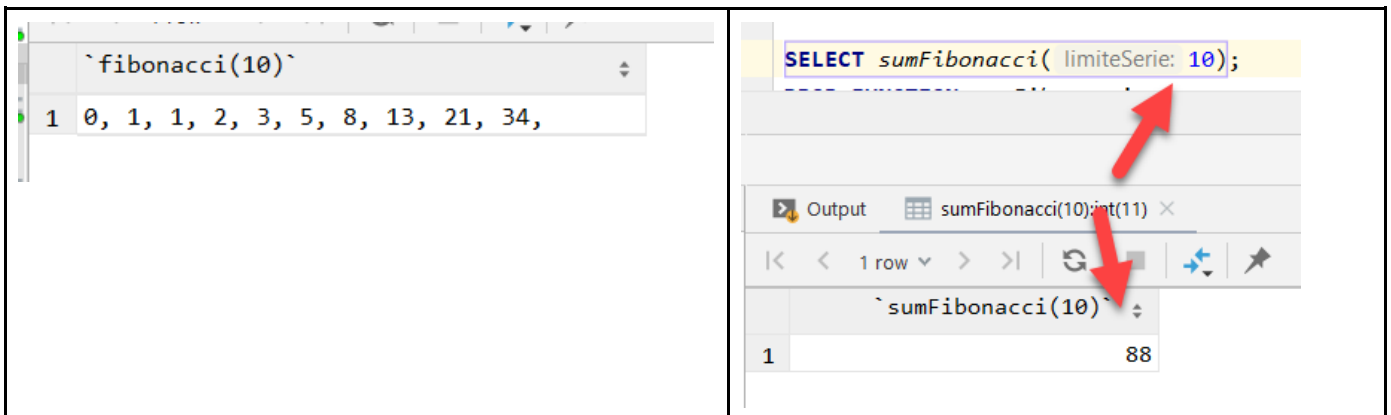
```
CREATE FUNCTION fibonacci(limitSerie INTEGER) RETURNS TEXT
BEGIN
  DECLARE result TEXT DEFAULT "";
  DECLARE n1 INTEGER DEFAULT 0;
  DECLARE n2 INTEGER DEFAULT 1;
  DECLARE serie INTEGER DEFAULT 0;
  DECLARE x INTEGER DEFAULT 2;

  SET result = CONCAT(result, n1, ', ', n2, ', ');

  WHILE x < limitSerie DO
    SET serie = n1 + n2;
    SET n1 = n2;
    SET n2 = serie;
    SET x = x + 1;
    SET result = CONCAT(result, serie, ', ');
  END WHILE;

  RETURN result;
END;
```

- Comportamiento esperado



The left screenshot shows the execution of the `fibonacci(10)` function, resulting in the output: `0, 1, 1, 2, 3, 5, 8, 13, 21, 34,`.

The right screenshot shows the execution of the `sumFibonacci(10)` function. The SQL query `SELECT sumFibonacci(limiteSerie: 10);` is entered. The output window shows the result of the function call: `sumFibonacci(10)` returns `88`. A red arrow points from the query to the output window.

Solución:

```
Drop function Fibonacci_suma;
CREATE FUNCTION Fibonacci_suma(lim INT) RETURNS TEXT
BEGIN
  DECLARE suma BIGINT DEFAULT 0;
  DECLARE res TEXT DEFAULT "";
  DECLARE a INT DEFAULT 0;
  DECLARE b INT DEFAULT 1;
  DECLARE c INT DEFAULT 0;
  DECLARE conta INT DEFAULT 0;

  CASE WHEN lim > 0
  THEN
    REPEAT
      SET c = a + b;
      SET a = b;
```

```

SET b = c;
SET suma = suma + a;
SET res = CONCAT(res, ', ', a);
SET conta = conta + 1;
UNTIL conta > (lim-2)
END REPEAT;
WHEN lim < 0
THEN SET res= 'Ups un Error!!';
END CASE;

RETURN CONCAT(res, ' - LA Suma es: ', suma);
END;
SELECT Fibonacci_suma(10);

```

- **Pregunta 2:** Crear una Vista de nombre **detalle_proyecto_persona** de las personas que no hayan nacido en la ciudad de **COCHABAMBA**.
 - La Vista debe de tener los siguientes campos (**NOMBRES**, **APELLIDOS**, **PROYECTO**, **TIPO_PROYECTO**, **NIVEL_PROYECTO**, **SEXO** y **DEPARTAMENTO**).
 - Consideraciones de la columna **NIVEL_PROYECTO**
 - Si la edad es mayor que 0 y menor igual a 20 mostrar 'MI PRIMER PROYECTO'
 - Si la edad es mayor que 20 y menor igual a 25 mostrar 'MI SEGUNDO PROYECTO'
 - Si la edad es mayor que 25 y menor igual a 30 mostrar 'MI TERCER PROYECTO'
 - Considere la imagen siguiente para tener el contexto de qué datos debe de mostrar cada columna.

NOMBRES	APELLIDOS	PROYECTO	TIPO_PROYECTO	NIVEL_PROYECTO
nombre4	apellidos4	Alfabetizacion	EDUCACION	MI SEGUNDO PROYECTO
nombre4	apellidos4	Creacion de Escuelas	EDUCACION	MI SEGUNDO PROYECTO
nombre5	apellidos5	Apoyo al dibujo	ARQUITECTURA	MI TERCER PROYECTO
nombre6	apellidos6	Apoyo al dibujo	ARQUITECTURA	MI PRIMER PROYECTO

Solución:

```

DROP VIEW detalle_proyecto_persona;
CREATE VIEW detalle_proyecto_persona AS
SELECT PER.nombre AS NOMBRES, PER.apellidos AS APELLIDOS, PROY.nombreProy
AS PROYECTO , PROY.tipoProy AS TIPO_PROYECTO, PER.SEXO AS SEXO,
DEP.nombre AS DEPARTAMENTO,
CASE
WHEN PER.edad > 0 AND PER.edad <= 20 THEN 'MI PRIMER PROYECTO'
WHEN PER.edad > 20 AND PER.edad <= 25 THEN 'MI SEGUNDO PROYECTO'
WHEN PER.edad > 25 AND PER.edad <= 30 THEN 'MI TERCER PROYECTOR'
WHEN PER.edad > 30 THEN 'PAGINA NO ENCONTRADA'
END as NIVEL_PROYECTO
FROM ong.persona AS PER
INNER JOIN ong.detalle_proyecto AS DPROY ON DPROY.id_per = PER.id_per
INNER JOIN ong.proyecto AS PROY ON DPROY.id_proy = PROY.id_proy
INNER JOIN ong.departamento AS DEP ON PER.id_dep = DEP.id_dep
Where DEP.nombre != 'Cochabamba';

```

- **Pregunta 4:** Crear TRIGGERS Before o After para **INSERT, UPDATE y DELETE** aplicado a la tabla PROYECTO.
- Crear una tabla de auditoría de acuerdo al siguiente:

```

CREATE TABLE
auditoria_proyecto
(
operation      CHAR(1) NOT
NULL,
stamp          TIMESTAMP NOT
NULL,
userid         TEXT      NOT NULL,
hostname       TEXT      NOT
NULL,
idProy         INTEGER NOT
NULL,
descProyBefore TEXT      NOT
NULL,
descProyAfter  TEXT      NOT
NULL
);

```

Consideraciones:

Los campos:

descProyBefore : Son los valores antes de la acción (insert - update - delete).

descProyAfter: Son los valores después de la acción (insert - update - delete).

Nota. Debe crear triggers.

- Crear un procedimiento almacenado SP para insertar datos a la tabla de auditoría.
- Considere la siguiente tabla de auditoría para ver el comportamiento esperado.

	stamp	userid	hostname	idProy	descProyBefore	descProyAfter
1 I	2019-12-02 05:20:23	root@localhost	DESKTOP-OI87J52	6	Empty value - INSERT ACTION	Apoyo al dibujo ARQUITECTURA
2 U	2019-12-02 05:23:11	root@localhost	DESKTOP-OI87J52	6	Apoyo al dibujo ARQUITECTURA	Apoyo al dibujo II ARQUITECTURA II
3 D	2019-12-02 05:25:11	root@localhost	DESKTOP-OI87J52	6	Apoyo al dibujo II ARQUITECTURA II	Empty value - Delete action

Solución:

```

CREATE PROCEDURE insertar_detalleProy_en_tablas(in operation char(1), in id_proy
integer, in descProyBefore text, descProyAfter text)
BEGIN
INSERT INTO auditoria_proyecto(operation, stamp, userid, hostname, idProy,

```

```

descProyBefore, descProyAfter) SELECT

operation,now(),user(),@@hostname,id_proy,descProyBefore,descProyAfter;
END;

DROP TRIGGER auditoria_PROYECTO;
CREATE trigger auditoria_PROYECTO
AFTER insert on proyecto
FOR EACH ROW
BEGIN
    DECLARE a TEXT DEFAULT 'Empty value- Insert Action';
    CALL insertar_detalleProy_en_tablas('I',NEW.id_proy,a,concat(NEW.nombreProy,'-
',NEW.tipoproy));
END;
DROP TRIGGER ActualizarProyecto;

CREATE TRIGGER ActualizarProyecto
AFTER update on proyecto
for each row
BEGIN
    call
insertar_detalleProy_en_tablas('U',OLD.id_proy,NEW.nombreProy,concat(old.nombreProy,
'- ',NEW.tipoproy));
END;

DROP TRIGGER EliminarProyecto;
CREATE TRIGGER EliminarProyecto
BEFORE DELETE on proyecto
for each row
begin
    DECLARE a TEXT DEFAULT 'Empty value- DELETE Action';
    call insertar_detalleProy_en_tablas('D',concat(old.id_proy,'-',old.nombreProy),a);
END;

```

- **Pregunta 4:** Crear un TRIGGER BEFORE INSERT para la tabla PERSONA.

```

CREATE TABLE
auditoria_persona
(
    operation CHAR(1) NOT
NULL,
    stamp    TIMESTAMP
NOT NULL,
    userid   TEXT    NOT
NULL,
    hostname TEXT    NOT
NULL,
    fullname TEXT    NOT
NULL,
    edad     INTEGER NOT
NULL,
    sexo     TEXT    NOT
NULL
);

```

Si es día lunes(2) y además la persona es de la ciudad de La Paz no se debe insertar el registro y se debe genera un error indicando lo siguiente.

MESSAGE_TEXT = **'No se admite inserciones en días lunes usuarios de la ciudad de LP';**

Si es otro día insertar los registros en la tabla de auditoria (auditoria_persona).

.

Para saber en que día se encuentra debe de utilizar : **select DAYOFWEEK(now())**

- Debe de crear una función para verificar que es día lunes y que el usuario es de la ciudad de La Paz. la función recibe dos parámetros.

- Dia de la semana(INTEGER).
- El ID de la ciudad (INTEGER)

GER	4	4	nombre4	apellidos4	1996-10-30	24	nombre4@gmail.com	m	2	4
AN	5	5	nombre5	apellidos5	1992-10-30	28	nombre5@gmail.com	m	3	5
D	6	6	nombre6	apellidos6	1999-10-30	19	nombre6@gmail.com	f	3	5
) IN	7	<generated>	nombre4	apellidos4	1996-10-30	24	nombre4@gmail.com	m	2	4
Mon	[45000][1644] No se admite inserciones en dias lunes usuarios de la ciudad de LP [row:7] X									

E	7	8	nombre4	apellidos4	1996-10-30	24	nombre4@gmail.com	m	5	4
---	---	---	---------	------------	------------	----	-------------------	---	---	---

ong.auditoria_proyecto [DBAII] X		ong.auditoria_persona [DBAII] X							
Tx: Auto				DDL		Tab-se...d (TSV)		ong.auditoria_f	
Q <Filter criteria>									
operation	stamp	userid	hostname	fullname	edad	sexo			
1 I	2019-12-02 06:17:49	root@localhost	DESKTOP-OI87J52	nombre4 apellidos4	24	m			