



BASE DE DATOS PRACTICA H4

Nombre: Henry Javier
Huarachi Quispe

Docente: William Barra

MANEJO DE CONCEPTOS

Defina que es lenguaje procedural en MySQL.

- Los procedimientos almacenados MySQL son conjuntos de comandos SQL almacenados en el servidor. Esto nos permite trabajar con grandes cantidades de datos.

DEFINA QUE ES UNA FUNCIÓN EN MYSQL.

- Es una rutina creada para tomar unos parámetros, procesarlos y retornar en un salida

3¿Cuál es la diferencia entre funciones y procedimientos almacenados?

- La diferencia entre una función y un procedimiento almacenado es que la función en si puede usarse de forma manual respecto a un procedimiento almacenado que se ejecuta automáticamente

```
CREATE OR REPLACE TRIGGER ADD_PROY
BEFORE INSERT ON PROYECTO
FOR EACH ROW
BEGIN
    IF NEW.TIPO_PROY='EDUCACION'OR NEW.TIPO_PROY='FORESTACION' OR NEW.TIPO_PROY='CULTURA'
    THEN SET NEW.ESTADO='ACTIVO';
    ELSE
    SET NEW.ESTADO='INACTIVO';
    END IF;
end;
```

```
CREATE OR REPLACE FUNCTION SUMA_FIBONACI(SERIE TEXT)
RETURNS INTEGER
BEGIN
    DECLARE SUMA INTEGER DEFAULT 0;
    DECLARE CONT INTEGER DEFAULT 1;
    DECLARE FINAL INTEGER DEFAULT CHAR_LENGTH(SERIE);

    REPEAT
        SET SUMA =SUMA + SUBSTRING(SERIE,CONT,1);
        SET CONT=CONT+2;
    until CONT > FINAL
    END REPEAT;

    RETURN SUMA;
end;
```

The screenshot shows a MySQL IDE with a database explorer on the left. The main editor displays the following SQL code:

```
88 CREATE FUNCTION suma_serie_fibonacci (serie VARCHAR(255))
89 RETURNS INT
90 BEGIN
91 DECLARE total INT DEFAULT 0;
92 DECLARE n INT DEFAULT 0;
93
94 WHILE LENGTH(serie) > 0 DO
95 SET n = SUBSTRING_INDEX(serie, ',', 1);
96 SET serie = SUBSTRING(serie, LENGTH(n) + 2);
97 SET total = total + n;
98 END WHILE;
99
100 RETURN total;
101 END;
102
103 select suma_serie_fibonacci(serie_fibonacci(10));
104
105 #11 Manejo de vistas Crear una consulta SQL para lo siguiente
```

The output window at the bottom shows the result of the query:

suma_serie_fibonacci(serie_fibonacci(10))
88

4. Cómo se ejecuta una función y un procedimiento almacenado.

The screenshot shows a MySQL IDE with a database explorer on the left. The main editor displays the following SQL code:

```
159
160 CREATE OR REPLACE TRIGGER ADD_PROY
161 BEFORE INSERT ON PROYECTO
162 FOR EACH ROW
163 BEGIN
164 IF NEW.TIPO_PROY='EDUCACION' OR NEW.TIPO_PROY='FORESTACION' OR NEW.TIPO_PROY='CULTURA'
165 THEN SET NEW.ESTADO='ACTIVO';
166 ELSE
167 SET NEW.ESTADO='INACTIVO';
168 END IF;
169 end;
170
171 INSERT INTO PROYECTO(NOMBRE_PROY, TIPO_PROY)
172 VALUES ('ARBOLES','EDUCACION'),
173 ('AMBIENTE','FORESTACION'),
174 ('RECUERDOS','CULTURA');
```

The output window at the bottom shows the result of the query:

fullname	desc_proyecto	departamento	codigo_dep
1 CARLOS SANTOS	1 TIPOPROY2	LA PAZ	lpz
2 STEFANY LURDES	2 TIPOPROY1	POTOSI	pts
3 JUAN PEREZ	3 TIPOPROY1	EL ALTO	<null>
4 GABRIELA PAHE	4 TIPOPROY2	EL ALTO	<null>
5 LUIS SANCHEZ	5 TIPOPROY2	EL ALTO	<null>
6 CARLA MINERVA	6 TIPOPROY1	EL ALTO	<null>

- La función se ejecuta de manera manual en si en una consulta. Muy distinto del procedimiento que se ejecuta por si solo

5. Defina que es una TRIGGER en MySQL.

- Un trigger en si es una función almacenada mas usada para verificar si hay cambios en una tabla en si con las condiciones de after y before además de tener dos variables el old y el new.

```
CREATE OR REPLACE TRIGGER insert_persona_dos
BEFORE INSERT ON persona
FOR EACH ROW
BEGIN
    INSERT INTO persona_dos(id_dep,id_prov,nombre,apellidos,fecha_nac,edad,genero,email)
    VALUES (NEW.id_dep,NEW.id_prov,NEW.nombre,NEW.APELLIDOS,NEW.fecha_nac,NEW.edad,NEW.genero,NEW.email);
END;
```

6. En un trigger que papel juega las variables OLD y NEW

Estas dos variables identifican el antes y el después

Old: Esta variable nos habla sobre como estuvo antes.

New: Esta variable nos muestra el respectivo cambio que tuvo.

```
CREATE OR REPLACE TRIGGER insert_persona_dos
BEFORE INSERT ON persona
FOR EACH ROW
BEGIN
    INSERT INTO persona_dos(id_dep,id_prov,nombre,apellidos,fecha_nac,edad,genero,email)
    VALUES (NEW.id_dep,NEW.id_prov,NEW.nombre,NEW.APELLIDOS,NEW.fecha_nac,NEW.edad,NEW.genero,NEW.email);
END;

INSERT INTO persona(id_dep,id_prov,nombre, apellidos,fecha_nac,edad,genero,email)
VALUES (2,2,'carlos', 'pauca', '1998-11-12',24,'m','carlos@gmail.com');

SELECT * FROM copia_persona;
```


7. En un trigger que papel juega los conceptos(cláusulas) BEFORE o AFTER

Lo que indican estas dos clausulas es si el trigger se ejecutara antes o después.

Before: antes

After. después

```
CREATE OR REPLACE TRIGGER ADD_PROY
BEFORE INSERT ON PROYECTO
FOR EACH ROW
BEGIN
    IF NEW.TIPO_PROY='EDUCACION' OR NEW.TIPO_PROY = 'FORESTACION' OR NEW.TIPO_PROY= 'CULTURA'
    THEN SET NEW.ESTADO='ACTIVO';
    ELSE
        SET NEW.ESTADO='INACTIVO';
    END IF;
end;

INSERT INTO PROYECTO(NOMBRE_PROY, TIPO_PROY)
VALUES ('ARBOLESS', 'EDUCACION'),
('AMBIENTE', 'FORESTACION'),
('RECUERDOS', 'CULTURA'),
('CARNE', 'COCINA');

SELECT PROY.*
```

8. A que se refiere cuando se habla de eventos en TRIGGERS

Los eventos son aquello que hace que el trigger funcione como cuando se modifica algún registro de alguna tabla

```
200
201 CREATE OR REPLACE TRIGGER insert_persona_dos
202 BEFORE INSERT ON persona
203 FOR EACH ROW
204 BEGIN
205     INSERT INTO persona_dos(id_dep,id_prov,nombre,apellidos,fecha_nac,edad,genero,email)
206     VALUES (NEW.id_dep,NEW.id_prov,NEW.nombre,NEW.APELLIDOS,NEW.fecha_nac,NEW.edad,NEW.genero,NEW.email);
207 END;
208
209 INSERT INTO persona(id_dep,id_prov,nombre, apellidos,fecha_nac,edad,genero,email)
210 VALUES (2,2,'carlos', 'paucar', '1998-11-12',24,'m','carlos@gmail.com');
211
212 SELECT * FROM copia_persona;
```

Output hito4.copia_persona

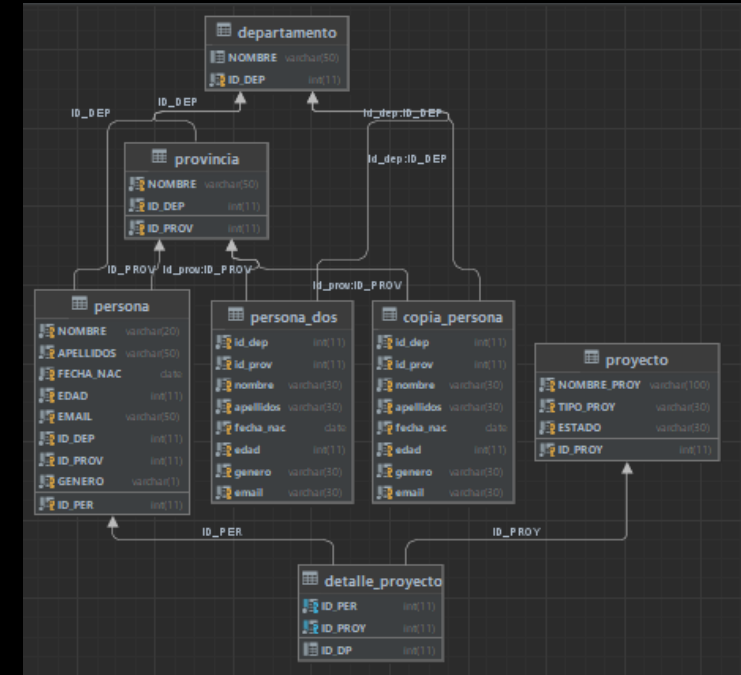
	id_dep	id_prov	nombre	apellidos	fecha_nac	edad	genero	email
1	1	1	Tania	jimenez	1995-02-11	27	F	tania@gmail.com
2	2	2	carlos	paucar	1998-11-12	24	m	carlos@gmail.com


```

INSERT INTO DEPARTAMENTO(NOMBRE)
VALUES ('LA PAZ'),
       ('POTOSI');
INSERT INTO PROVINCIA(NOMBRE, ID_DEP)
VALUES ('VIACHA',1),
       ('UYUNI',2);
INSERT INTO PERSONA (NOMBRE, APELLIDOS, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES ('CARLOS', 'SANTOS', '2000-12-16', 21, 'CARLOS1123@GMAIL.COM', 1, 1, 'M'),
       ('STEFANY', 'LURDES', '1998-04-10', 24, 'TEFFY45@GAMIL.COM', 2, 2, 'F');
INSERT INTO PROYECTO(NOMBRE_PROY, TIPO_PROY)
VALUES('PROYECTO1', 'TIPOPROY1'),
       ('PROYECTO2', 'TIPOPROY2');
INSERT INTO DETALLE_PROYECTO(ID_PER, ID_PROY)
VALUES (1,2),
       (2,1);

```

#10. Crear una función que sume los valores de la copia Fibonacci



PARTE PRACTICA

9. Crear la siguiente Base de datos y sus registros.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database explorer with the 'hito4' database selected. The right pane shows the SQL script editor with the following code:

```
CREATE FUNCTION suma_serie_fibonacci (serie VARCHAR(255))
RETURNS INT
BEGIN
    DECLARE total INT DEFAULT 0;
    DECLARE n INT DEFAULT 0;
    WHILE LENGTH(serie) > 0 DO
        SET n = SUBSTRING_INDEX(serie, ',', 1);
        SET serie = SUBSTRING(serie, LENGTH(n) + 2);
        SET total = total + n;
    END WHILE;
    RETURN total;
END;
```

The script is executed, and the output window shows the result of the function call:

```
select suma_serie_fibonacci(serie: serie_Fibonacci(10));
```

The output window displays the result: 88.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database explorer with the 'hito4' database selected. The right pane shows the SQL script editor with the following code:

```
DECLARE serie VARCHAR(255);
DECLARE n1 INT;
DECLARE n2 INT;
DECLARE n3 INT;

SET n1 = 0;
SET n2 = 1;
SET serie = CONCAT(n1, ',', n2);

WHILE n > 2 DO
    SET n3 = n1 + n2;
    SET serie = CONCAT(serie, ',', n3);
    SET n1 = n2;
    SET n2 = n3;
    SET n = n - 1;
END WHILE;

RETURN serie;
END;
```

The script is executed, and the output window shows the result of the function call:

```
select serie_Fibonacci(10);
```

The output window displays the result: 0,1,1,2,3,5,8,13,21,34.

10.Crear una función que sume los valores de la serie Fibonacci.

```

121 CREATE OR REPLACE VIEW DATOS AS
122 SELECT CONCAT(PERSONA.NOMBRE, ' ', PERSONA.APELLIDOS) AS NOMBRES_Y_APELLIDOS,
123        PERSONA.EDAD AS EDAD,
124        PERSONA.FECHA_NAC AS FECHA_DE_NACIMIENTO,
125        PROYECTO.NOMBRE_PROY AS NOMBRE_DEL_PROYECTO
126 FROM PERSONA
127 INNER JOIN DEPARTAMENTO ON PERSONA.ID_DEP = DEPARTAMENTO.ID_DEP
128 INNER JOIN DETALLE_PROYECTO ON PERSONA.ID_PER = DETALLE_PROYECTO.ID_PER
129 INNER JOIN PROYECTO ON DETALLE_PROYECTO.ID_PROY = PROYECTO.ID_PROY
130
131 WHERE PERSONA.GENERO='F' AND DEPARTAMENTO.NOMBRE='EL ALTO' AND PERSONA.FECHA_NAC='2000-10-10' ;
132
133 SELECT * FROM DATOS;

```

Output hito4.datos

	NOMBRES_Y_APELLIDOS	EDAD	FECHA_DE_NACIMIENTO	NOMBRE_DEL_PROYECTO
1	CARLA MINERVA	22	2000-10-10	PROYECTO1

11. Manejo de vistas

```
CREATE OR REPLACE TRIGGER ADD_PROY
BEFORE INSERT ON PROYECTO
FOR EACH ROW
BEGIN
    IF NEW.TIPO_PROY='EDUCACION' OR NEW.TIPO_PROY='FORESTACION' OR NEW.TIPO_PROY='CULTURA'
    THEN SET NEW.ESTADO='ACTIVO';
    ELSE
    SET NEW.ESTADO='INACTIVO';
    END IF;
end;
```

Output Result 25

ID_PROY	NOMBRE_PROY	TIPO_PROY	ESTADO
1	PROYECTO1	TIPOPROY1	<null>
2	PROYECTO2	TIPOPROY2	<null>
3	EDUCACION PERSONAS ESPECIALES	EDUCACION	<null>
4	PLANTACION DE ARBOLES	FORESTACION	<null>
5	LOS AZTECAS	CULTURA	<null>
6	ARBOLESS	EDUCACION	ACTIVO
7	ARBOLESS	EDUCACION	ACTIVO
8	AMBIENTE	FORESTACION	ACTIVO
9	RECUERDOS	CULTURA	ACTIVO
10	CARNE	COCINA	INACTIVO

12. Manejo de TRIGGERS I.

```

FROM PROYECTO AS PROY;
#13 Manejo de Triggers II El trigger debe de llamarse calculaEdad
CREATE OR REPLACE TRIGGER CALCULAEDAD
BEFORE INSERT ON PERSONA
FOR EACH ROW
BEGIN
    SET NEW.EDAD= TIMESTAMPDIFF(YEAR,NEW.FECHA_NAC,CURDATE());
end;

INSERT INTO PERSONA(NOMBRE, APELLIDOS, FECHA_NAC, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES ('JOSE','CARRASCO','1994-12-17','PEPE12@GMAIL.COM',3,1,'M');

SELECT *FROM PERSONA;
#14 Manejo de TRIGGERS III.

```

Output Result 27

9 rows

ID_PER	NOMBRE	APELLIDOS	FECHA_NAC	EDAD	EMAIL	ID_DEP
5	LUIS	CARRASCO	2000-10-10	22	LUIS123@GMAIL.COM	3
6	CARLA	MINERVA	2000-10-10	22	CARLY123@GMAIL.COM	3
10	JOSE	CARRASCO	1994-12-17	27	PEPE12@GMAIL.COM	3
11	Tania	jimenez	1995-02-11	27	tania@gmail.com	1

13.Manejo de Triggers II.

```
);  
  
CREATE OR REPLACE TRIGGER insert_persona_dos  
BEFORE INSERT ON persona  
FOR EACH ROW  
BEGIN  
    INSERT INTO persona_dos(id_dep,id_prov,nombre,apellidos,fecha_nac,edad,genero,email)  
    VALUES (NEW.id_dep,NEW.id_prov,NEW.nombre,NEW.APELLIDOS,NEW.fecha_nac,NEW.edad,NEW.genero,NEW.email);  
END;  
  
INSERT INTO persona(id_dep,id_prov,nombre, apellidos,fecha_nac,edad,genero,email)  
VALUES (2,2,'carlos', 'paucar', '1998-11-12',24,'m','carlos@gmail.com');  
  
SELECT * FROM copia_persona;
```

Output hito4.copia_persona

id_dep	id_prov	nombre	apellidos	fecha_nac	edad	genero	email
1	1	Tania	jimenez	1995-02-11	27	F	tania@gmail.com
2	2	carlos	paucar	1998-11-12	24	m	carlos@gmail.com

14. Manejo de TRIGGERS II.

15. Crear una consulta SQL que haga uso de todas las tablas

	fullname	desc_proyecto	departamento	codigo_dep
1	CARLOS SANTOS	1 TIPOPROY2	LA PAZ	lpz
2	STEFANY LURDES	2 TIPOPROY1	POTOSI	pts
3	JUAN PEREZ	3 TIPOPROY1	EL ALTO	<null>
4	GABRIELA PAHE	4 TIPOPROY2	EL ALTO	<null>
5	LUIS SANCHEZ	5 TIPOPROY2	EL ALTO	<null>
6	CARLA MINERVA	6 TIPOPROY1	EL ALTO	<null>

```
create or replace view reporte_proyectos as
select concat(per.nombre, ' ', per.apellidos) as fullname,
       concat(dp.id_per, ' ', proy.TIPO_PROY) as desc_proyecto,
       dep.nombre as departamento,
       case
         when dep.nombre = 'la paz' then 'lpz'
         when dep.nombre = 'potosi' then 'pts'
       end as codigo_dep
from persona as per
inner join departamento as dep on per.id_dep = dep.id_dep
inner join detalle_proyecto as dp on per.id_per = dp.id_per
inner join proyecto as proy on dp.id_proy = proy.id_proy;

select *
from reporte_proyectos;
```