

Practica 2 U3



**Saldaña Aguilar Gabriela**

**Materia** BASES DE DATOS

**Profesor** HERNANDEZ CONTRERAS EULER

**Grupo** 2CM10

**Fecha de entrega** 16/02/15

**Nombre del Trabajo:**

**INDICE…**…………………………………………………………..

1.-Marco Teórico……………………………….3

Desarrollo………………………………….4

2.-CONCLUSIONES……………………..9

3.-BIBLIOGRAFÍA…………………………9

**MARCO TEÓRICO**

MySQL Workbench es una herramienta CASE visual unificada para los arquitectos de bases de datos, desarrolladores y administradores de bases. MySQL Workbench proporciona el modelado de datos, desarrollo de SQL y herramientas completas de administración de la configuración del servidor, la administración de usuarios, copia de seguridad, y mucho más.

Modelar simplifica el diseño de bases de datos y mantenimiento, ya que permite, el arquitecto de datos, para visualizar los requisitos y resolver problemas de diseño. Diseño de base de datos basada en modelos es una metodología eficiente para la creación de bases de datos válidos y de buen rendimiento, mientras que proporciona la flexibilidad necesaria para responder a las cambiantes necesidades de datos. Los modelos se utilizan para construir los diagramas ER y bases de datos MySQL físicas.   
  
En esta práctica se vieron nuevas sentencias como lo son : UPDATE y INSERT aumentando sintaxis nueva como los STORE PROCEDURES.

Sintaxis:

UPDATE table\_name  
SET column1=value1,column2=value2,...  
WHERE some\_column=some\_value;

//si no ponemos where clause todos los registros tendran ese valor.

INSERT INTO table\_name  
VALUES (value1,value2,value3,...);

STORE PROCEDURE SINTAXIS

Delimiter char1 char2

CREATE PROCEDURE NOMBRE()

begin

{

Sentencia sql1;

.......

....

..

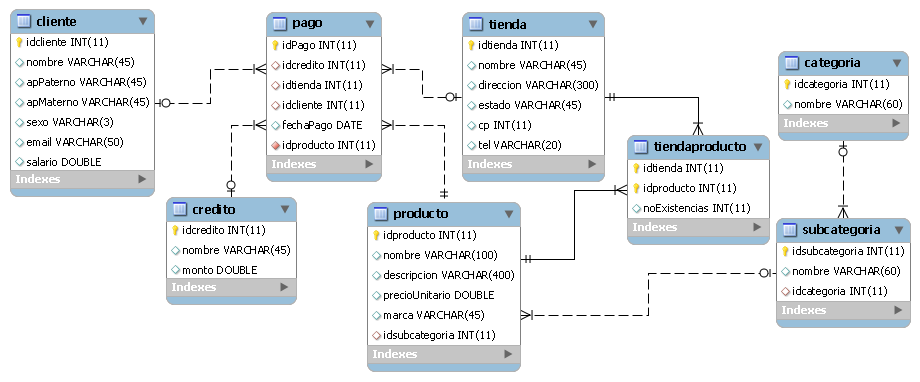
sentencia sqln;

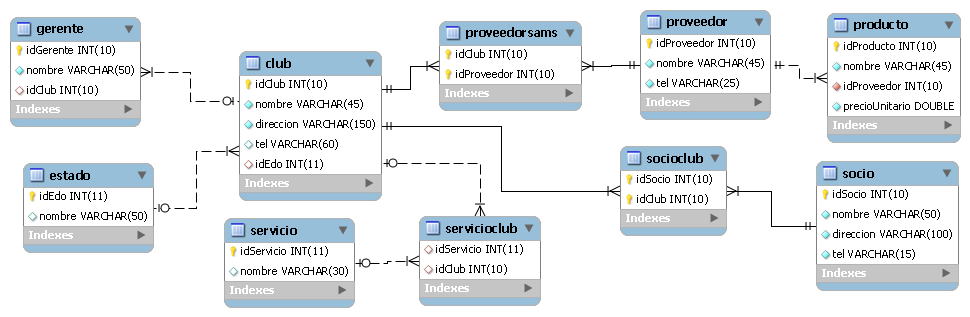
}

end char1 char2

delimiter ;

**TAREAS A REALIZAR:**

**USANDO LA BD DE SAMS Y LA DE ELEKTRA**

****

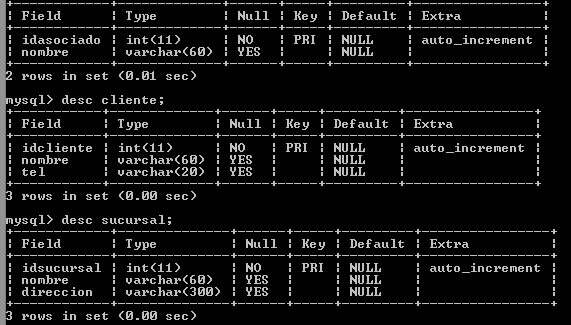
**1.-** **CREAR EL SIGUIENTE ESQUEMA DE BD UTILIZANDO INFORMACIÓN PROVENIENTE DE UNA BD EXTERNA(SAMS Y ELEKTRA).**

1.1Crear una BD

1.2Crear una relación que permita almacenar:

|  |  |  |
| --- | --- | --- |
| cliente  idcliente//auto\_increment  nombre  tel | asociado  idasociado//auto  nombre | sucursal  idsucursal//auto  nombre  dirección |
| create table cliente(  idcliente int not null primary key auto\_increment,  nombre varchar(60),  tel varchar(20)  ); | create table asociado(  idasociado int not null primary key auto\_increment,  nombre varchar(60)  ); | create table sucursal(  idsucursal int not null primary key auto\_increment,  nombre varchar(60),  direccion varchar(300)  ); |

NOTA: auto\_increment es para que los índices se coloquen por si solos.



**2.-** **CONSTRUIR UNA SENTENCIA SQL QUE PERMITA EXTRAER LA INFO DE LA DB SAMS, DE SOCIO E INSERTARLA EN CLIENTE, EN ESTE CASO EL NOMBRE Y TELEFONO**

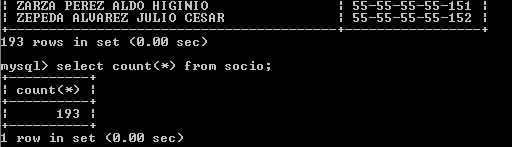
insert into cliente(nombre,tel)—- parámetros a recivir

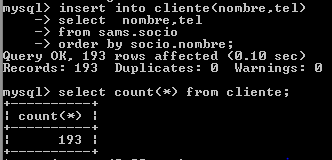
select nombre,tel—- están en una misma tabla

from sams.socio—- de que BD?

order by socio.nombre;-- para ordenar hay que indicar la relacion

primero verificamos que el select nos envie la info deseada, luego la insertamos en la nueva BD.





Estos pasos se realizaran para los demás inserts.

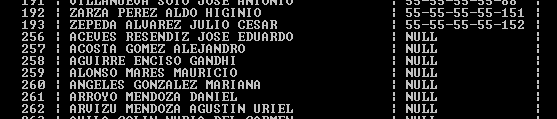
**3.- AHORA SELECCIONAMOS EL NOMBRE DE CLIENTE EN ELEKTRA, YA QUE NO HAY TEL, PERO COMO EL NOMBRE VIENE DIVIDIDO POR APELLIDOS HAY QUE CONCATENAR.**

insert into cliente (nombre)—a donde va a llegar

select CONCAT(apPaterno," ",apMaterno," ",nombre)—esta dentro de la misma tabla pero hay que concatenar

from elektra.cliente—- de que BD?

order by cliente.apPaterno;-- ordenar respecto a un atributo de la relacion dentro de la cual extraemos info



Como se puede apreciar tenemos los clientes que se sacaron de sams(los que tienen tel) y los de elektra (los que no tienen tel); y el incremento en los indices va aumentando automaticamente.

**4.-** **CARGAMOS DATOS A LA RELACION ASOCIADO DE LA RELACION GERENTE DE SAMS .**

insert into asociado (nombre)

select nombre from sams.gerente;//dentro de la misma tabla



**5.-** **USANDO TIENDA DE ELEKTRA Y DE SAMS USANDO CLUB TRAEMOS EL NOMBRE Y LA DIRECCIÓN DE LAS SUCURSALES.**

Como en este caso queremos traer información de diferentes BD y relaciones, ya no de una misma, extraemos primero de una BD y luego de la otra.

insert into sucursal (nombre,direccion)—- Valores a llenar

select nombre,direccion from elektra.tienda;-- Una misma tabla especificando la base de datos en la cual se encuentra la relacion tienda.



insert into sucursal (nombre,direccion)

select nombre,direccion from sams.club;

//FUERON 109

LUEGO DEL select count(\*) from sucursal;



**6.-** **STORE PROCEDURES**

SE PONEN EN CONSULTAS CONCURRIDAS.

SON PROGRAMAS PARA TRABAJOS REPETITIVOS.

**HACER UN PROCEDURE QUE MUESTRE: EL NO DE REGISTROS DE GERENTES,**

**NOMBRE DEL CLUB Y ASOCIADOS DEL EDOMEX Y LOS CLUBES CON EL SERVICIO APPLE SHOP.**

delimiter ##

create procedure uno()

begin

-- numero de registros de gerentes

select count(\*) as Gerentes\_Registrados from sams.gerente;

-- nombre del club y asociados del edoMex.

select g.nombre as ASOCIADO, c.nombre as CLUB

from gerente g,club c, estado e

where g.idclub=c.idclub

and c.idedo=e.idedo

and e.nombre like "M\_xico%";

-- clubes con el servicio de appleshop

select c.nombre as CLUB,s.nombre as SERVICIO

from club c,servicioclub sc, servicio s

where c.idclub=sc.idclub

and sc.idservicio=s.idservicio

and s.nombre like "Apple% Shop%";

end ##

delimiter ;

\*Para llamarlo en seguida usamos

call uno();

\*Para ver que BD tiene algún SP

show procedure status;

\*Para borrar algún SP

drop procedure \_\_\_();



**CONCLUSIONES:**

En esta práctica pude realizar mejor y con menos errores mis búsquedas a pesar de la sencillez de estas, aprendí nueva sintaxis que puede ayudar a realizar una inserción de manera anidada, ya que tenemos los querys anteriores dentro de estos nuevos. Los Store Procedures son herramientas que ayudan a realizar tareas repetitivas por lo que ahorra tiempo al momento de hacer ciertas búsquedas.

**Bibliografía**: PROFESOR EULER.