Apuntes tema 6 parte 1

Orden de ejecucion de una sentencia select

SELECT nomde as ‘DEPARTAMENTOS’
FROM departamentos
WHERE numce= 10

SELECTS

El select y from son siempre obligatorios, se puede omitir el where

SELECT nomde as ‘DEPARTAMENTOS’
FROM departamentos

Con el asterisco se seleccionarian todas las columnas de la tabla departamentos

SELECT * o tambien SELECT departamentos*
FROM departamentos
WHERE numce= 10

SELECT departamentos.nomde **(Seleccionas la columna nomde de la tabla departamento) También se puede utilizar en el where y el from**

Select numde, nomde **(Se puede seleccionar disitnta columnas)**

Ejericio de clase, busca el nombre y extension telefico de los empleados con extension tlefonica 350
Select nomem, extelem
form empleados
where extelem = ‘350’

select nomen
form empleados
where extelem is null **(Donde el valor es nulo)**

select empleados.nomen **(hace referencia la nomen de la tabla empleados, es mejor usarlo así)**

Formas de concatenar campos

where concat(empleados.nomen, ‘’, empleados.ape1em) = ‘Juan Lopez’ **(Sirve para concatenar campos)**

tambien se podrian poner de la siguiente forma
nomen y ape1em por separado

where nomen=”Juan” ape1em=”López”;

where concat(nomem, '', ape1em, '', ape2em, '' numhiem)

where concat_ws(‘’, empleados.nomen, empleados.ape1em) = ‘Juan Lopez’ **(Se utiliza en caso de que uno de los campos sean nulos)**

El concat también se puede usar en el select

select concat(nomem, '', ape1em, '', ape2em, '' numhiem)

NULLS

where ifnull(numhiem, 0) < 1; **(Si los hijos tienen un valor nulo o menor que 0 se muestran)**

o tambien valdria
where numhiem is null or numhiem<1;

select(ifnull(numhiem,0) **(Convierte el número de hijos a 0 si son nulos)**

ifnotnull **(Aquellos que no son nulos)**

where munhiem between 1 and 3 **(Número hijos entre 1 y 3 usando between entre)**

select CONCAT(nomem, '', ape1em, '', ifnull(ape2em, '') ,numhiem,'') as nomcompleto_concat **(Concatena los atributos en una sola línea, el ifnull convierte los nulos ape2em en '' si se una concat_ws no es necesario ponerlo)**

select concat_ws('',nomem, ape1em. ape2em) as nomcompleto_concat_ws **(Lo mismo pero con concat_ws cuando hay valore nulos)**

se puede poner seguido ambos

select CONCAT(nomem, '', ape1em, '', ape2em, '' numhiem) as nomcompleto_concat,
concat_ws('',nomem, ape1em. ape2em) as nomcompleto_concat_ws

not between 1 and 3; **(que no esté entre 1 y 3)**

select CONCAT(nomem, '', ape1em, '', ape2em, '' numhiem) as nomcompleto_concat, comisem **(obtiene en una sola lista los atributos y en una separada la comision)**

where ifnull(comisen, 0) = 0; **(si la comision es nulo es 0 donde la comision vale 0)**

ESPCACIOS

Para quitar espacios se usa ltrim y rtrim (a la izquierda left y derecha right)

where trim(centros.nomcr) = 'Sede Central'; **Quita ambos espacios derecha y izquierda)**

where lower (trim(centros.nomcr)) = 'Sede Central'; **(lower convierte en minuscula al valor)**

select solo por ejemplo
select 'hola' es como system out en java

>= y <= (regla de comparación igual)
<> o != (Distinto de, se usa el primero más comun)

fecinem <='2020/2/5' **(formato siempre año, mes y dia)**

where fecinem <= date_sub(curdate(), interval 1 year);

curdate() es la fecha actual, el date_sub resta el intervalo de un año a la fecha actual curdate() Se puede restar minutos dias, meses etc

DATEDIFF('2020/2/5', '2020/2/5') **compara la dieferencia entre dos fechas, siempre poner la fecha superior primero. si no te saldra un numero negativo**

Procedimientos almacenados Stored Procedures

funciones (functions)

delimiter $$ **crea un delimitador con el símbolo $$, lo cambias en vez de ;**

Estructura procedimiento

create procedure probando(
Aqui van los parametros
in nombre varchar(100) o simplemente nombre varchar(100)
)
begin
(Aquí viene la estrctura de un select, select, from y where) pueden ir varias
(el delimitador va aquí)
end $$
delimiter ;

call probando('Juan lópez') **-- llama al procedimiento donde este juan lópez**

drop procedure probando; **lo elimina**

MUY IMPORTENTE EN LAS SENTECIAS LA ÚLTIMA FILA SIEMPRE TIENE QUE ACABAR EN ; SI NO NO FUNCIONARA LA LLAMADA AL PROCEDIMIENTO

ORDER

ORDER BY - se utiliza para ordenar los resultados de una consulta, según el valor de la columna especificada. Por defecto, se ordena de forma ascendente (ASC) según los valores de la columna.

Select *
form departmaentos
order by depende;

SELECT nombre_columna(s)
FROM nombre_tabla
ORDER BY nombre_columna(s) ASC|DESC (Ascende y descendente)

Si se quiere ordenar por orden descendente se utiliza la palabra DES

nombre	apellido1	apellido2
ANTONIO	PEREZ	GOMEZ
LUIS	LOPEZ	PEREZ
ANTONIO	GARCIA	BENITO

```
SELECT nombre, apellido1
FROM personas
ORDER BY apellido1 ASC
```

Esta es la consulta resultante:

nombre	apellido1
LUIS	LOPEZ
ANTONIO	GARCIA
ANTONIO	PEREZ

Ejemplo de ordenación descendente (DES)

```
SELECT nombre, apellido1
FROM personas
ORDER BY apellido1 DESC
```

Esta es la consulta resultante:

nombre	apellido1
ANTONIO	PEREZ
ANTONIO	GARCIA
LUIS	LOPEZ

Ejemplo uso as y order

select numce as ‘Num’, nomce as ‘Nombre’ **- Selciona los campos y los mostrará con el nombre puesto despues del as**
from centros
order by centros.nomce **- Ordena de forma ascendente y alfabeticamente**

select numce as ‘Num’, nomce as ‘Nombre’
from centros

Si se ejecuta ordenaria el num de forma ascendente

declare nuevocentro int; **(Se declara una variable int de centros)**

set @nuevocentro = (select mac(numce)+1 form centros);

(Establece en la variable @nuevocntro con set la fórmula siguiente)

EJ:

declare nuevocentro int;

set @nuevocentro = (select mac(numce)+1 form centros);

insert into centros()

values();

Esto iria dentro de un procedimiento almacenado

-Para probar el procedimiento
call probando(“Parametros”, @nuevocentro);

select @nuevocentro;

Ejercicio de ejemplo

drop procedure ejercicio14;
delimiter $$
create procedure ejercicio14(
in nombrecentro varchar(60) **(Parametro, le añadimoc el nombre que queramos)**
)
begin
select centros.dirce
from centros
where rtrim(ltrim(centros.nomce)) = nombrecentro; **(Sirve para quitar los espacios y para evitar que no salga el campo vacion cunado se realiza el call)**
end $$
delimiter ;

No entran las condicionales en el examen (if)

declare mivar char(3); **(Declara una variable, va dentro del begin**

Con el set se asigna un valor, set mivar = '123';

(Iria fuera del delimiter)

set @numce = 20; **(@ es una variable global, la variables globales no se declaran)**

call BuscarUnCentro(@numce, @direccion, @ mas los parametros que haya);

select(@numce, @direccion, …) **muestra los resultados**

PARAMETROS

IN parámetro de entrada

Indica que el valor del parámetro debe especificarse cuando se llama al procedimiento almacenado, y el valor del parámetro modificado durante el procedimiento almacenado no se puede devolver, que es el valor predeterminado

Solo necesita pasar los datos al procedimiento almacenado, y no necesita devolver el valor calculado.

Solo se puede usar como parámetro entrante

OUT Parámetro de salida

Este valor se puede cambiar dentro del procedimiento almacenado y se puede devolver

No acepta datos externos entrantes, solo devuelve el valor después del cálculo.

Solo se puede usar como parámetro de despliegue

INOUT parámetros de entrada y salida

Se especifica cuando se llama y se puede cambiar y devolver

Necesita transferir los datos al procedimiento almacenado después de llamar y calcular, y luego devolver el valor de retorno

Se puede usar como parámetros entrantes y salientes

select empleados.extelem into extension **(Almacena el extelem de los empleados en el campo extension)**

order by ape1em, ape2em, nomem; **(Ordenacion de los campos de forma ascendete va despues del where)**

order by ap1em desc, ape2em asc; **(Se pueden ordenar tanto ascendes como descendentes)**

JOIN

Los JOIN son usados en una sentencia SQL para recuperar datos de varias tablas al mismo tiempo. Estas tablas tienen que estar relacionadas de alguna forma, por ejemplo, una tabla usuarios, y otra tabla juegos que contiene también la id del usuario al que pertenece el juego:

select nomem, nomde
form empleados join departamentos **(Anexionamos los empleados y los deaprtmaentos, aparecerán en 2 tablas, sirve para relacionar tablas de forma mas sencilla a traves de sus claves foraneas)**

on empleados.numde = departamentos.numde **(Donde el número de empleados y el número de departamentos coinciden, son las claves foraneas)**
order by nomem;

select nomem, nomde
form empleados join departamentos
on empleados.numde = departamentos.numde
join centros
on departamentos.numce = centro.numce
order by nomem;

(Se pueden concatenar varios join, para varias claves foraneas)

Esto se usa cuando para una clave primaria de una tabla tenga que acceder a otra pk de otra tabla pero tiene primero que pasar por una o más tablas.

RECUERDA

CUANDO PONGAS UN ON SE SUELE HACER REFERENICA A LA CLAVE PRIMARIA QUE COMPARTEN AMBAS TABLAS

CUANDO TE PIDA PARA CADA CASO, TIENES QUE USAR PARAMETROS

LOS PROCEDIENTOS SIEMPRE USAN PARAMETROS, ESOS PARAMETROS DESPUES SE LE ASIGNA SU VALOR CUANDO SE REALIZA EL CALL

Si no se quieren incluir lo extremos se usaría los operandos > <

Cuando se pida devolver algo, en los parametros se utilizaria el out, nunca el in. El out es para valores que sean concretos

EJ:
create procedure ejer(
in centro varchar(60),
out direccion varchar(60) **(Devuelve algo)**
)
begin
select ifnull(dirce, 'No existe el centro') into direccion **(Se almacena en el parametro de salida , se usa siempre en el select)**
from centros
where lower(trim(nomce)) = lower(trim(centro));
end $$

delimiter;

call ejer(‘Sede Central’, @direccion); **(Se llama al parámetro de salida usando @ seguido del nombre, es decir una variable global**

select concat_ws(‘ ’,’el resulrado de la ejecucion es:’, @direccion); **(Sirve como un system out en java, muestra el resultado del parámetro de salida)**

Las comilla para las nombres utilizando la sentencia AS solo son obligatorias cuando haya espacio, aunque se puede usar también en palabras simples

Para seleccionar todos los campos de una tabla, se utiliza esto, ej:
empleados.*

por ejemplo

select empleados.*, departamento.nomde

Cuando hay comparaciones en el mismo campo se usa siempre el operando OR empleados.numde or departamentos.numde

Distinct (Sirve para que no se publiquen campos, va despues de un select)

select DISTINCT CONCAT(nomem, ' ',ape1em, ' ',ape2em) **as nomcompleto, departamentos.nomde**

Cada caso o cualquiera en un enunciado de un ejercicio hay que utilizar procediemientos almacenados