18-7-2023

CODERHOUSE

CURSO SQL

Cesar Gaston Petit Martin

CADENA DE GIMNASIOS

SEGUNDA ENTREGA

Contenido

Definición

El modelo que se va a utilizar es el de una cadena de gimnasios fitness, conformado por diversas sucursales, y que a su vez cuenta muchas actividades, en los distintos turnos.

Objetivo

Crear un sistema de Base De Datos, que no permita llevar el correcto funcionamiento de la cadena de gimnasios y permitir un control de todos los aspectos correspondientes al modelo elegido, a su vez poder llevar una de control y seguimiento.

Necesidades

Podes administrar todas la cadenas del gimnasio , poder llevar un correcto control contables de los clientes y poder controlar la evolución de lo empleados que trabajan.

Diagrama Entidad-Relación (conceptual)

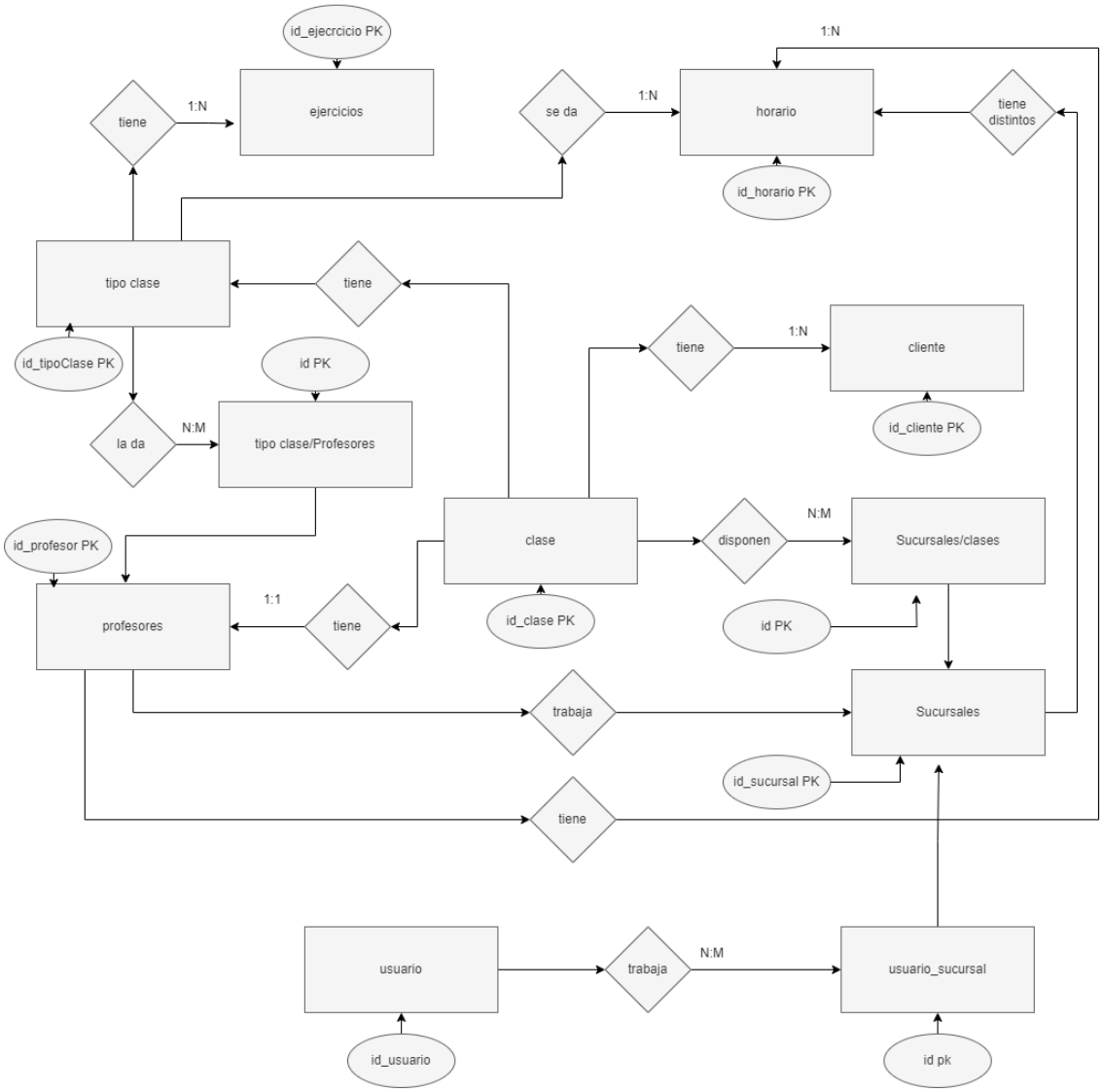
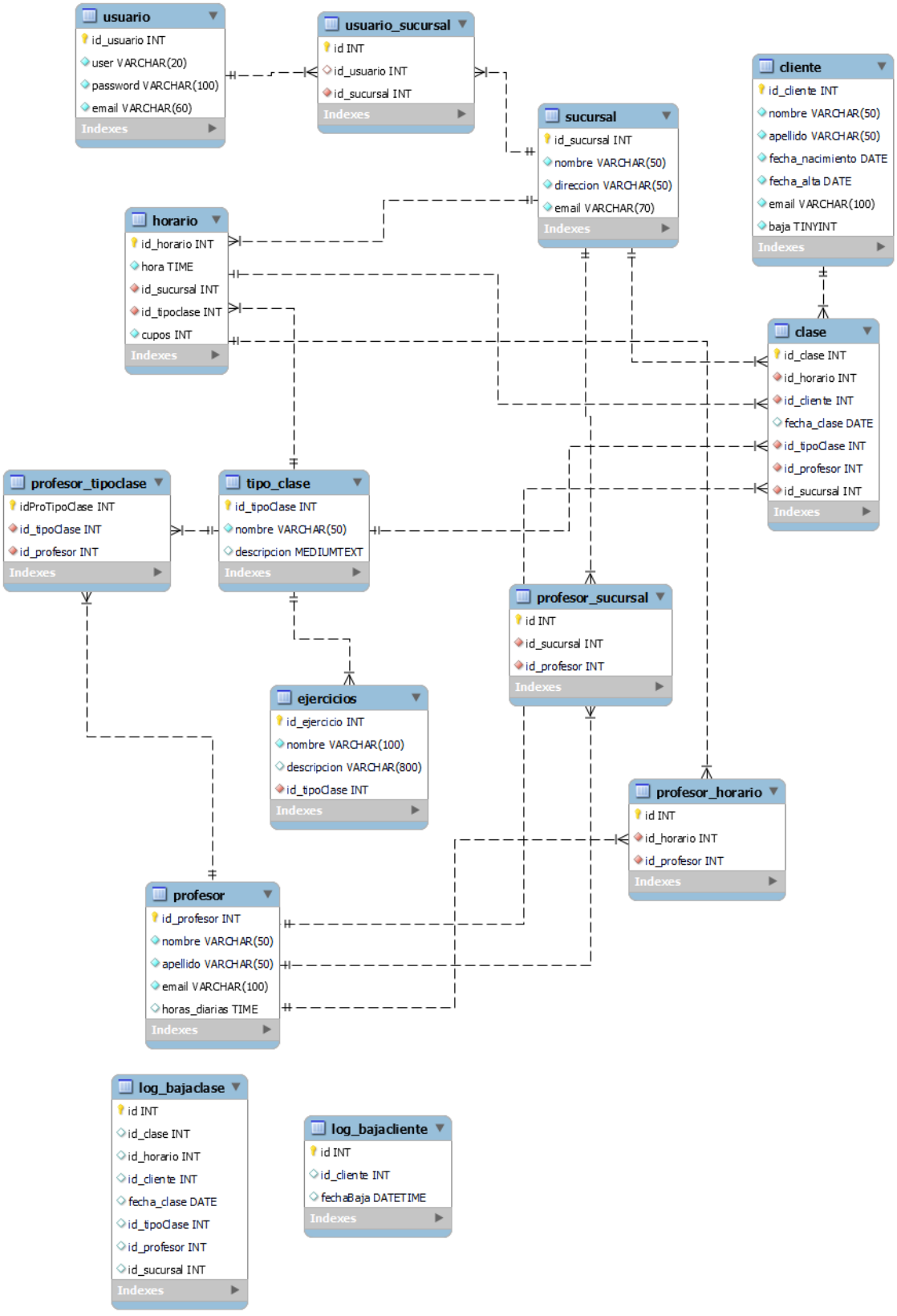


Diagrama Entidad Relación (DER)



VISTAS GENERADAS

VISTA 1

NOMBRE: **v_clases_info**

DEFINICION: ESTA VISTA MUESTRA LA INFORMACIÓN DE LAS CLASES.

ESQUEMA:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `workshop_cesarpetit..v_clases_info `v` AS
    SELECT
        `c`.`id_clase` AS `id_clase`,
        `h`.`hora` AS `hora`,
        `c`.`fecha_clase` AS `fecha_clase`,
        `s`.`nombre` AS `nombre_sucursal`,
        `p`.`nombre` AS `nombre_profesor`,
        `tc`.`nombre` AS `tipo_clase`
    FROM
        ((((`workshop_cesarpetit`.`clase` `c`
        JOIN `workshop_cesarpetit`.`horario` `h` ON ((`c`.`id_horario` = `h`.`id_horario`)))
        JOIN `workshop_cesarpetit`.`sucursal` `s` ON ((`c`.`id_sucursal` = `s`.`id_sucursal`)))
        JOIN `workshop_cesarpetit`.`profesor` `p` ON ((`c`.`id_profesor` = `p`.`id_profesor`)))
        JOIN `workshop_cesarpetit`.`tipo_clase` `tc` ON ((`c`.`id_tipoClase` = `tc`.`id_tipoClase`)))
```

RESULTADO OBTENIDO:

id_clase	hora	fecha_clase	nombre_sucursal	nombre_profesor	tipo_clase
1	08:00:00	2023-02-07	ALMAGRO	Juan	CrossFit
2	08:00:00	2023-02-07	ALMAGRO	Maria	CrossFit
3	08:00:00	2023-02-07	BALVANERA	Laura	CrossFit
4	08:00:00	2023-02-07	BARRACAS	Maria	Funcional
5	08:00:00	2023-02-07	ALMAGRO	Juan	Funcional
6	08:00:00	2023-02-07	BARRACAS	Maria	Spinning
7	08:00:00	2023-02-08	ALMAGRO	Juan	CrossFit
8	08:00:00	2023-02-08	ALMAGRO	Maria	CrossFit
9	08:00:00	2023-02-08	BALVANERA	Laura	CrossFit
10	08:00:00	2023-02-08	BARRACAS	Maria	Funcional
11	08:00:00	2023-02-08	ALMAGRO	Juan	Funcional
12	08:00:00	2023-02-08	BARRACAS	Maria	Spinning

VISTA 2

NOMBRE: v_clases_por_cliente :

DEFINICION: ESTA VISTA MUESTRA LAS CLASES QUE HICIERON LOS CLIENTES

ESQUEMA:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `workshop_cesarpetit`.`v_clases_por_cliente` AS
    SELECT
        `c`.`id_cliente` AS `id_cliente`,
        CONCAT(`c`.`nombre`, ' ', `c`.`apellido`) AS `nombre_completo`,
        COUNT(`cl`.`id_clase`) AS `total_clases`
    FROM
        (`workshop_cesarpetit`.`cliente` `c`
        LEFT JOIN `workshop_cesarpetit`.`clase` `cl` ON ((`c`.`id_cliente` = `cl`.`id_cliente`)))
    GROUP BY `c`.`id_cliente` , `nombre_completo`
```

RESULTADO OBTENIDO:

id_cliente	nombre_completo	total_clases
11	CAROLA CASTELO	0
13	CAROLINA GARCIA	0
1	CESAR PETIT	2
12	DELFINA ARIAS	0
8	ENRIQUE MENTS	0
3	ERNESTO ROLDAN	2
10	ESTELA OCHI	0
7	GASTON CORNIO	0
5	JOAQUIN LON	2
2	JUAN GOMEZ	2
6	LUIS CAPRI	2
15	LUZ APAGADA	0
9	MARIANA SOMAS	0
4	PEDRO PEREZ	2
14	SILVIA LUNA	0

VISTA 3

NOMBRE: v_cupos_horario_sucursal:

DEFINICION: ESTA VISTA MUESTRA LOS CUPOS LIBRES POR CLASE

ESQUEMA:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `workshop_cesarpetit`.`v_cupos_horario_sucursal` AS
    SELECT
        `h`.`id_horario` AS `id_horario`,
        `h`.`hora` AS `hora`,
        `s`.`nombre` AS `nombre_sucursal`,
        `h`.`cupos` AS `total_cupos`,
        (`h`.`cupos` - COUNT(`cl`.`id_clase`)) AS `cupos_disponibles`
    FROM
        ((`workshop_cesarpetit`.`horario` `h`
        JOIN `workshop_cesarpetit`.`sucursal` `s` ON ((`h`.`id_sucursal` = `s`.`id_sucursal`)))
        LEFT JOIN `workshop_cesarpetit`.`clase` `cl` ON ((`h`.`id_horario` = `cl`.`id_horario`)))
    GROUP BY `h`.`id_horario` , `h`.`hora` , `s`.`nombre` , `h`.`cupos`
```

RESULTADO OBTENIDO:

id_horario	hora	nombre_sucursal	total_cupos	cupos_disponibles
1	08:00:00	ALMAGRO	20	14
2	08:00:00	ALMAGRO	20	16
3	08:00:00	BALVANERA	10	8
4	08:00:00	BALVANERA	10	10
5	09:00:00	BARRACAS	20	20
6	10:00:00	RECOLETA	15	15
7	11:00:00	COLEGIALES	20	20
8	12:00:00	VILLA ORTUZAR	20	20
9	13:00:00	RECOLETA	15	15
10	14:00:00	CABALLITO	15	15
11	15:00:00	BARRACAS	10	10
12	16:00:00	BALVANERA	10	10
13	17:00:00	ALMAGRO	20	20
14	18:00:00	BARRACAS	20	20
15	19:00:00	PARQUE CHAC...	20	20

VISTA 4

NOMBRE: vistacanttipoclase:

DEFINICION: ESTA VISTA MUESTRA LA CANTIDAD DE CLASES QUE TIENE EL PROFESOR

ESQUEMA:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `workshop_cesarpetit`.`vistacanttipoclase` AS
    SELECT
        CONCAT(`p`.`nombre`, ' ', `p`.`apellido`) AS `nombre Profesor`,
        `tc`.`nombre` AS `nombre`,
        COUNT(0) AS `Cantidad Clase`
    FROM
        ((`workshop_cesarpetit`.`clase` `c`
        JOIN `workshop_cesarpetit`.`profesor` `p` ON ((`p`.`id_profesor` = `c`.`id_profesor`)))
        JOIN `workshop_cesarpetit`.`tipo_clase` `tc` ON ((`tc`.`id_tipoClase` = `c`.`id_tipoClase`)))
    WHERE
        (`p`.`id_profesor` = 1)
    GROUP BY `tc`.`id_tipoClase`
```

RESULTADO OBTENIDO:

	nombre Profesor	nombre	Cantidad Clase
▶	Juan Perez	CrossFit	2
	Juan Perez	Funcional	2

STORES PROCEDURES

STORES PROCEDURE 1

NOMBRE: sp_agregar_profesor:

PARAMETROS:

P_nombre: NOMBRE DEL PROFESOR

p_apellido : APELLIDO DEL PROFESOR

p_email: EMAIL DEL PROFESOR

p_horasDiarias: HORAS DEL PROFESOR

DEFINICION: AGREGA UN NUEVO PROFESOR AL STAFF

ESQUEMA:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `sp_agregar_profesor`(in p_nombre varchar(50), p_apellido varchar(50), in p_email varchar(100), in p_horasDiarias time)
BEGIN
        INSERT INTO profesor(nombre,apellido,email,horas_diarias)
  values(p_nombre,p_apellido,p_email,p_horasDiarias);
END
```

STORES PROCEDURE 2

NOMBRE: sp_ordenar_clase

PARAMETROS :

Field : NOMBRE DE LA COLUMNA QUE QUIERO FILTRAR,

Orden: ELIGO EL ORDENAMIENTO ASC - DESC

DEFINICION: DEVUELVE LA CLASES ORDENADO POR LOS PARAMETROS SELECCIONADOS.

FUNCIONES

FUNCION 1

NOMBRE: profesoresCantidadHoras

PARAMETROS :

p_horas: PARAMETRO QUE DEFINE LOS PROFESORES QUE TIENENE MAS DE ESAS HORAS.

DEFINICION: DEVUELVE TODOS LO PROFESORES QUE HACEN LAS MISMAS O MAYORES HORAS QUE EL PARAMETRO DE ENTRADA.

ESQUEMA:

```
CREATE DEFINER=`root`@`localhost` FUNCTION `profesoresCantidadHoras`( p_horas time)
RETURNS int
    READS SQL DATA
BEGIN

        declare profesoresHoras int ;
    set profesoresHoras = (select count(*) cantidadProfesores from profesor where
horas_diarias >=p_horas);
    return profesoresHoras;
END
```

FUNCION 2

NOMBRE: mayor_numero

PARAMETROS:

num1: NUMERO 1 PARA COMPARAR

num2: NUMERO 2 PARA COMPARAS

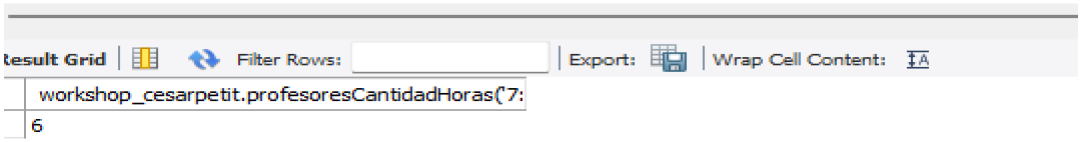
DEFINICION: FUNCION QUE DEVUELVE CUAL ES EL NUMERO MAS GRANDE EN CASO DE IGUALDAD DEVUELVE IGUALES

ESQUEMA:

```
delimiter $$
CREATE FUNCTION mayor_numero(num1 int,num2 int) RETURNS varchar(50)
NO SQL
BEGIN
        declare mayor varchar(50);
        if (num1 > num2)    then
                set mayor = concat('el mayor numero es:',' ',CONVERT(num1, CHAR(20)));
        end if;
  if (num2 > num1) then
                                set mayor = concat('el mayor numero es:',' ',CONVERT(num2,
CHAR(20)));
        end if;
  if (num1 = num2) then
                set mayor = concat("Los numeros son iguales!!!!");
   end if;
  return mayor;
END$$
delimiter ;
```

RESULTADO OBTENIDO:

```
1 •   select workshop_cesarpetit.profesoresCantidadHoras('7:00');
2
```



TRIGGER

TRIGGER 1

NOMBRE: tr_bajaUser

DEFINICION: SE CREO TRIGGER QUE CUANDO SE DA DE BAJA EL USUARIO DE MANERA LOGICA LO GUARDA EN EL LOG

TABLAS QUE IMPACTA: CLIENTE – LOG_CLIENTE

ESQUEMA:

```
delimiter $$
use workshop_CesarPetit $$
CREATE TRIGGER tr_bajaUser
AFTER UPDATE
ON cliente FOR EACH ROW
BEGIN
        IF (OLD.baja <> NEW.baja AND NEW.BAJA = 1) THEN
                INSERT INTO log_bajaCLiente (id_cliente,fechaBaja)
                VALUES (NEW.id_cliente,NOW());
        END IF;
```

END $$

TRIGGER 2

NOMBRE: clase_BEFORE_DELETE

DEFINICION: SE CREO UN TRIGGER QUE AL ELIMINAR LA CLASE DEL CLIENTE LO GUARDA EN EL LOG

TABLAS QUE IMPACTA: CLASE – LOG_BAJACLASE

ESQUEMA:

```
use workshop_CesarPetit $$

CREATE TRIGGER `workshop_CesarPetit`.`clase_BEFORE_DELETE` BEFORE DELETE ON `clase`
FOR EACH ROW

BEGIN

INSERT INTO log_bajaClase


VALUES(OLD.id_clase,OLD.id_horario,OLD.id_cliente,OLD.fecha_clase,OLD.id_tipoClase,OLD.id_profesor,OLD.id_sucursal);

END$$

delimiter ;
```