S.I.G.T.

Base de Datos II

Katsu enterprise (勝つ企業)

Rol	Apellido	Nombre	C.I	Email	Tel/Cel.
Coordinador	Macedo	Fiorella	5.503.612-7	fiorellamacedo22@gmail.com	095 256 351
Sub-Coordinador	Dávila	Oriana	5.074.874-1	orianadavila99@gmail.com	093 308 483
Integrante 1	Pérez	Lautaro	5.468.712-7	pabloramirez199221@gmail.com	097 967 986
Integrante 2	Budes	Agustín	5.121.247-6	agustinbudes@gmail.com	099 431 623

Docente: Romero, Carlos.

Fecha de culminación

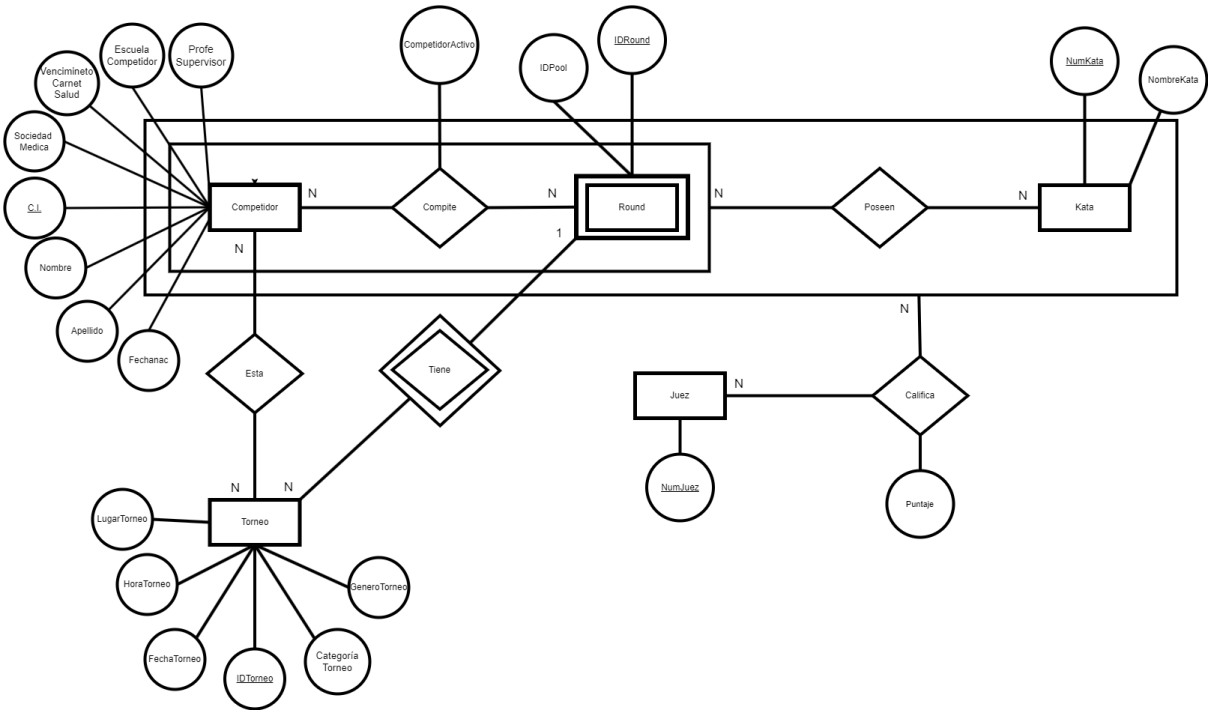
13/11/2023

TERCERA ENTREGA

Índice

Índice....1
Modelo conceptual (DER)....2
Esquema Relacional normalizado....3
RNE....4
Sentencias DDL....5
Gestión de usuarios y permisos....10
Sentencias DCL....12
Diccionario de datos....14
Sugerencias para política de respaldos de las Bases de datos y logs....18
Consultas de base de datos....19
Backups y recuperación....24
Configuración de encoding de tablas....27

Modelo conceptual (DER)



En caso de querer visualizar mejor la imagen, ingrese al siguiente link:

Esquema Relacional normalizado

Esquema relacional normalizado:

ENTIDADES:

Competidor(CI, nombre, apellido, fechanac, sociedadmedica, vencimientocarnetsalud)

Torneo(IDTorneo, generotorneo, categoriatorneo, fechatorneo, horatorneo)

Round(IDRound, IDtorneo, IDpool)

IDTorneo -> Torneo(IDTorneo)

Kata(NumKata, Nombrekata)

Juez(NumJuez)

RELACIONES:

Compite(CI, IDRound, IDTorneo, CompetidorActivo)

CI -> Competidor(CI)
IDTorneo -> Round(IDTorneo)
IDRound -> Round(IDRound)

Esta(CI, IDTorneo)

CI -> Competidor(CI)
IDTorneo -> Torneo(IDTorneo)

Califica(CI, NumKata, IDTorneo, NumJuez, IDRound, PuntuajeJuez)

CI -> Poseen(C)
IDTorneo -> Poseen(IDTorneo)
NumKata -> Poseen(NumKata)
NumJuez -> Juez(NumJuez)
Round -> Poseen(IDRound)

Poseen(CI, NumKata, IDTorneo, Round)

CI -> Compite(CI)
NumKata -> Kata(NumKata)
IDTorneo -> Compite(IDTorneo)
Round -> Compite(IDRound)

RNE

(√ x ∈ competidor) (x ∈ compite si x ∈ esta)

Sentencias DDL

```
/*Entidades*/

create table Competidor(CI int not null,

Nombre varchar (20) not null,

Apellido varchar(20) not null,

FechaNac date not null,

SociedadMedica varchar(30) not null,

VecimientoCarnetSalud date not null,

EscuelaCompetidor VARCHAR(20),

ProfeSupervisor VARCHAR(20),

constraint cp_Competidor primary key (CI));


create table Torneo(

IDTorneo int not null,

CategoriaTorneo varchar(5) not null,

GeneroTorneo varchar(10) not null,

FechaTorneo date not null,

HoraTorneo time not null,

LugarTorneo varchar(20),

constraint cp_Torneo primary key (IDTorneo),

CONSTRAINT restriccion_sexo CHECK (GeneroTorneo in ('Femenino','Masculino')),
```

```
CONSTRAINT restriccion_edad CHECK (CategoriaTorneo in ('12-13','14-15', '16-17',
'+18')));

create table Round(
IDRound int not null,
IDTorneo int not null,
IDPool int not null,
constraint cp_Round primary key (IDRound),
constraint cp_Round_Torneo
foreign key (IDTorneo)
references Torneo(IDTorneo));

create table Kata (NumKata int not null,
NombreKata varchar(50) not null,
constraint cp_Kata primary key (NumKata));
create table Juez (NumJuez int not null,
constraint cp_Juez primary key (NumJuez));

/*Relaciones*/
create table Compite (CI int not null,
IDTorneo int not null,
IDRound int not null,
```

```
CompetidorActivo bool,
constraint cp_Compite primary key (CI, IDTorneo, IDRound),
constraint ce_Compite_Competidor foreign key (CI)
references Competidor(CI),
constraint ce_Compite_Torneo foreign key (IDTorneo)
references Round(IDTorneo),
constraint ce_Compite_Round foreign key (IDRound)
references Round(IDRound));
```

```
create table Esta(CI int not null,

IDTorneo int not null,

constraint cp_Esta primary key (CI, IDTorneo),

constraint ce_Esta_Competidor foreign key (CI)

references Competidor(CI),

constraint ce_Esta_Torneo foreign key (IDTorneo)

references Torneo(IDTorneo));

create table Poseen(CI int not null,

IDRound int not null,

NumKata int not null,

IDTorneo int not null,

constraint cp_Poseen primary key (CI, NumKata, IDTorneo, IDRound),

constraint ce_Poseen_Kata foreign key (NumKata)

references Kata(NumKata),

constraint ce_Poseen_Compite foreign key (CI, IDTorneo, IDRound)

references Compite(CI, IDTorneo, IDRound));
```

```
create table Califica(CI int not null,

IDTorneo int not null,

NumKata int not null,

NumJuez int not null,

Puntaje int not null,

IDRound int not null,

constraint cp_Califica primary key (CI, IDTorneo, NumKata, NumJuez, IDRound),

constraint ce_Califica_Poseen foreign key (CI, IDTorneo, IDRound)

references Poseen(CI, IDTorneo, IDRound),

constraint ce_Califica_Juez foreign key (NumJuez)

references Juez (NumJuez));
```

Gestión de usuarios y permisos

Estudio de los permisos sobre BD, tablas y columnas, considerando los diferentes roles

Competidor

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Torneo

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Round

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Kata

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Juez

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez				
Administrador				

Califica

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓	✓		
Administrador	✓	✓	✓	✓

Poseen

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Compite

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Esta

Rol	Select (ver)	Insert (Ingresar)	Modificar (Update)	Eliminar (Delete)
Juez	✓			
Administrador	✓	✓	✓	✓

Sentencias DCL

create user 'juez'@'%' identified by 'contraseñajuez';
create user ’administrador’@'%' identified by ’adminpassword’;

grant select, insert, update, delete on bdKatsuEnterprise.Competidor TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Torneo TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Round TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Kata TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Juez TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Poseen TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Califica TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Compite TO ’administrador’@'%';

grant select, insert, update, delete on bdKatsuEnterprise.Estan TO ’administrador’@'%';

grant select on basededatos.Kata TO ‘juez’@'%';
grant select (genero) on basededatos.Torneo TO ‘juez’@'%';
grant select (categoria) on basededatos.Torneo TO ‘juez’@'%';
grant insert (Puntaje) on basededatos.Califica TO ‘juez’@'%';

Diccionario de datos

Competidor

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
CI	INT	8	Cédula de Identidad	Primary Key
Nombre	VARCHAR	20	Nombre de Competidor	Not Null
Apellido	VARCHAR	20	Apellido del Competidor	Not Null
FechaNac	DATE	-	Fecha de natalidad	Not Null
SociedadMedica	VARCHAR	30	Sociedad Médica a la que pertenece	Not Null
VencimientoCarnetSalud	DATE	-	Fecha en la que vence el carnet salud	Not Null
Escuela	VARCHAR	30	Escuela donde practica el competidor	Not Null
ProfeSupervisor	VARCHAR	40	Profesor que tiene responsabilidad del alumno	NotNull

Torneo

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	Primary Key
GeneroTorneo	VARCHAR	10	Define si el torneo será de competidores mujeres u hombres	Not Null
FechaTorneo	DATE	-	Fecha (día, mes y año) en el que se realiza el torneo	Not Null
HoraTorneo	TIME	-	Hora (hora:minuto) en la que el torneo empezará	Not Null
CategoriaTorneo	VARCHAR	10	Clasifica el torneo según el rango de edad	Not Null
Lugartorneo	VARCHAR	50	Lugar donde se efectuara el torneo	Not Null

Rounds

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	Primary Key
IDPool	INT	10	Número de Pool	NotNull
IDRound	INT	10	Número de Ronda	notNull

Kata

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
NumKata	INT	**3**	Número identificador del Kata	Primary Key
Nombrekata	VARCHAR	**50**	Nombre del kata	NotNull

Juez

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
NumJuez	INT	1	Numero del Juez	Primary Key

Poseen

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
CI	INT	8	Numero de cedula del competidor	NotNull
NumKata	INT	3	Número identificador del Kata	NotNull
IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	NotNull
IDRound	INT	10	Número de Ronda	NotNull

Compite

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
CI	INT	8	Numero de cedula del competidor	NotNull
IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	NotNull
IDRound	INT	10	Número de Ronda	NotNull
CompetidorActivo	Bool	-	Define si el competidor está realizando el Kata (bool= true)	NotNull

Esta

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
CI	INT	8	Numero de cedula del competidor	NotNull
IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	NotNull

Califica

nombre de atributo	Tipo de dato	Longitud	Descripción	Restricción
CI	INT	8	Numero de cedula del competidor	NotNull
NumKata	INT	3	Número identificador del Kata	NotNull

IDTorneo	VARCHAR	50	Nombre diferenciador de todos los torneos	NotNull
NumJuez	INT	1	Numero del Juez	NotNull
IDRound	INT	10	Número de Ronda	NotNull
Puntaje	Double	-	Puntaje del juez	NotNull

Sugerencias para política de respaldos de las Bases de datos y logs

Frecuencia de respaldo:

- Establece una frecuencia regular de respaldos que sea apropiada para el tipo de datos y la actividad de la base de datos. Puedes considerar respaldos diarios, semanales o incluso más frecuentes dependiendo de la criticidad de los datos.

Automatización del proceso:

- Automatiza el proceso de respaldo tanto como sea posible para reducir errores humanos y garantizar la consistencia en la implementación de la política de respaldos.

Logs de respaldos:

- Lleva un registro detallado de todas las operaciones de respaldo, incluyendo éxito, falla, tiempos de inicio y finalización. Los registros son esenciales para la auditoría y para identificar cualquier problema potencial.

Consultas de base de datos

/* Consulta N°1 */

```
select distinct C.CI, C.Nombre, C.Apellido, C.Escuela, T.IDTorneo, T.FechaTorneo, T.CategoriaTorneo, T.GeneroTorneo

from Competidor C

join Compite Co on C.CI = Co.CI

join Round R on Co.IDTorneo = R.IDTorneo

join Torneo T on R.IDTorneo = T.IDTorneo

where T.FechaTorneo = '2023-10-30'

order by T.IDTorneo, T.FechaTorneo, C.Nombre, C.Apellido;
```

/*----------*/

SELECT DISTINCT: Selecciona columnas específicas de las tablas Competidor (alias C), Compite (alias Co), Round (alias R), y Torneo (alias T). Utiliza DISTINCT para asegurarse de que los resultados sean únicos.

Uniendo las tablas (Competidor, Compite / Compite, Round / Round, Torneo)

JOIN Compite Co ON C.CI = Co.CI

JOIN Round R ON Co.IDTorneo = R.IDTorneo

JOIN Torneo T ON R.IDTorneo = T.IDTorneo

/*----------
*/

/*Consulta N°2*/

```
SELECT k.NumKata, k.NombreKata, COALESCE(COUNT(p.NumKata), 0) AS cantidad

FROM Kata k

LEFT JOIN Poseen p ON k.NumKata = p.NumKata

GROUP BY k.NumKata, k.NombreKata

ORDER BY cantidad DESC;
```

/*----------
*/

LEFT JOIN: Realiza una unión izquierda entre las tablas Kata y Poseen en base a la condición k.NumKata = p.NumKata.

GROUP BY: Agrupa los resultados por el número y nombre del kata.

COALESCE: Se utiliza para manejar casos en los que no hay coincidencias en la tabla derecha (Poseen).

ORDER BY: Ordena los resultados en función de la cantidad de veces que cada kata ha sido poseído (cantidad), en orden descendente (DESC).

/*----------
*/

/* Consulta n°5 */

```
SELECT C.CI AS Cedula, C.Nombre, C.Apellido, T.CategoriaTorneo,
T.GeneroTorneo, R.IDPool AS Pool

FROM Competidor C

JOIN Compite Co ON C.CI = Co.CI

JOIN Round R ON Co.IDRound = R.IDRound AND Co.IDTorneo = R.IDTorneo

JOIN Torneo T ON Co.IDTorneo = T.IDTorneo

WHERE T.CategoriaTorneo = ' '

AND T.GeneroTorneo = ' '

AND R.IDPool IS NOT NULL

ORDER BY R.IDPool, C.CI;

/*----------------------------------------------------------------------------------------------------
*/
```

SELECT: Selecciona las columnas que se mostrarán en los resultados de la consulta

FROM: Indica las tablas desde las cuales se obtendrán los datos.

JOIN: Une las tablas según las condiciones especificadas.

uniendo tablas con tablas (competidor, compite / compite, IdRound / Compite, Torneo)

```
JOIN Compite Co ON C.CI = Co.CI

JOIN Round R ON Co.IDRound = R.IDRound AND Co.IDTorneo = R.IDTorneo

JOIN Torneo T ON Co.IDTorneo = T.IDTorneo
```

/*Consulta n°4*/

SELECT C.Nombre AS NombreCompetidor, C.Apellido AS ApellidoCompetidor, P.NumKata AS NumeroKata, K.NombreKata, R.IDRound AS Ronda, T.IDTorneo, CA.Puntaje

FROM Competidor C

JOIN Poseen P ON C.CI = P.CI

JOIN Round R ON P.IDRound = R.IDRound AND P.IDTorneo = R.IDTorneo

JOIN Torneo T ON R.IDTorneo = T.IDTorneo

JOIN Califica CA ON P.CI = CA.CI AND P.IDRound = CA.IDRound AND P.NumKata = CA.NumKata AND P.IDTorneo = CA.IDTorneo

JOIN Kata K ON P.NumKata = K.NumKata

WHEREC.CI = '1'

AND T.IDTorneo = '1'

ORDER BY R.IDRound;

/*----------*/

Uniendo las tablas (Competidor, Poseen / Poseen, Round / Round, Torneo / Poseen, Califica / Poseen, Kata)

JOIN Poseen P ON C.CI = P.CI

JOIN Round R ON P.IDRound = R.IDRound AND P.IDTorneo = R.IDTorneo

JOIN Torneo T ON R.IDTorneo = T.IDTorneo

JOIN Califica CA ON P.CI = CA.CI AND P.IDRound = CA.IDRound AND P.NumKata = CA.NumKata AND P.IDTorneo = CA.IDTorneo

JOIN Kata K ON P.NumKata = K.NumKata

/*Consulta 7*/

```
SELECT T.IDTorneo, R.IDRound AS Ronda, K.NombreKata

FROM Torneo T

JOIN Round R ON T.IDTorneo = R.IDTorneo

JOIN Poseen P ON R.IDRound = P.IDRound AND R.IDTorneo = P.IDTorneo

JOIN Kata K ON P.NumKata = K.NumKata

WHERE K.NumKata = 'NumeroKata'

ORDER BY T.IDTorneo, R.IDRound;
```

/*----------
*/

/*Consulta 8*/

Backups y recuperación

Se usará un master slave para poder guardar una copia por si luis rompe el servidor principal.

Master - Slave es un proceso en el cual se usa un segundo servidor en algún otro sitio con el fin de servir como respaldo por si al servidor principal le ocurre un accidente, el servidor secundario actuará y lo reemplazará para que no se detenga ningún proceso (el torneo de kata).

Utilizando un script se usará un menú para gestionar los respaldos. y se usará un script para poder configurar cada cuanto se hará una copia.

```
#!/bin/bash
opc=0
fecha=$(date +%Y%m%d-%H-%M-%S-backup_bd)( cambiar a year-month-day-hour-masa-second)

function menu() {
   echo "MENÚ DE GESTIÓN DE RESPALDOS"
   echo "1 - Realizar un respaldo completo"
   echo "2 - Respaldar la estructura de la Base de datos"
   echo "3 - Restaurar base de datos"
   echo "4 - Realizar consulta personalizada"
   echo "5 - Salir"
}

function realizar_respaldo_completo() {
   echo "Ingrese el nombre de la base de datos"
   read nomb
   nombre=$(echo $nomb | tr '[:upper:]' '[:lower:]')
   echo "show databases" | mysql -u root -p > bd.sql
   existe=$(cat bd.sql | grep -c $nombre)
   if [ $existe -ge 1 ]; then
        mysqldump --opt --events --routines --triggers --default-character-set=utf8 -u root -p $nombre > $nombre.sql
      tar -czvf $fecha.tar.gz $nombre.sql
      mv $fecha.tar.gz /root/respaldos/bd/completa/
      rm $nombre.sql
      rm bd.sql
      echo "Respaldo completo realizado con éxito."
   else
```

```
        echo "No existe la base de datos, presione enter para continuar.."
        read enter
        rm bd.sql
    fi
}

function realizar_respaldo_estructura() {
    echo "Ingrese el nombre de la base de datos"
    read nomb
    nombre=$(echo $nomb | tr '[:upper:]' '[:lower:]')
    echo "show databases" | mysql -u root -p > bd.sql
    existe=$(cat bd.sql | grep -c $nombre)
    if [ $existe -ge 1 ]; then
         mysqldump -v --opt --no-data --default-character-set=utf8 -u root -p $nombre >
$nombre.sql
        tar -czvf $fecha.tar.gz $nombre.sql
        mv $fecha.tar.gz /root/respaldos/bd/estructura
        rm $nombre.sql
        rm bd.sql
        echo "Respaldo de la estructura realizado con éxito."
    else
        echo "No existe la base de datos, presione enter para continuar.."
        read enter
        rm bd.sql
    fi
}

function restaurar_basedatos() {
    echo "Ingrese el nombre de la base de datos"
    read nomb
    nombre=$(echo $nomb | tr '[:upper:]' '[:lower:]')
    echo "show databases" | mysql -u root -p > bd.sql
    existe=$(cat bd.sql | grep -c $nombre)
    if [ $existe -ge 1 ]; then
        tar -xzvf /root/respaldos/bd/estructura/$fecha.tar.gz
        mysql -u root -p $nombre < $nombre.sql
        rm $nombre.sql
        rm bd.sql
        echo "Restauración de base de datos realizada con éxito."
    else
        echo "No existe la base de datos, presione enter para continuar.."
        read enter
        rm bd.sql
    fi
```

```
}

function pregunta_personalizada() {
    echo "Ingrese el nombre de la base de datos"
    read nomb
    nombre=$(echo $nomb | tr '[:upper:]' '[:lower:]')
    echo "Ingrese la sintaxis correspondiente"
    read sentencia
    echo "$sentencia" | mysql -u root -p $nombre
    echo "Consulta personalizada realizada con éxito."
}

while [ $opc -ne 6 ]; do
    menu
    echo "Ingrese una opción: "
    read opc
    case $opc in
        1) realizar_respaldo_completo ;;
        2) realizar_respaldo_estructura ;;
        3) restaurar_basedatos ;;
        4) pregunta_personalizada ;;
        5) echo "Adios" ; break ;;
        *) echo "Opción no válida" ; read pausa ;;
    esac
done
```

Este código usara un menú para poder elegir la opción que deseamos:

MENÚ DE GESTIÓN DE RESPALDOS
1 - Realizar un respaldo completo"
2 - Respaldar la estructura de la Base de datos
3 - Restaurar base de datos
4 - Realizar consulta personalizada
5 - Salir"

Este código al igual que el anterior realiza una copia directamente a la tabla de nuestro servidor y llevándola a un respaldo en nuestro CentOS

Configuración de encoding de tablas

se usará utf8mb4

ENGINE= BD_katsu DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

ejemplo:

create table Competidor(CI int not null,

Nombre varchar (20) not null,

Apellido varchar(20) not null,

FechaNac date not null,

SociedadMedica varchar(30) not null,

VecimientoCarnetSalud date not null,

Lugartorneo varchar (50) not null,

constraint cp_Competidor primary key (CI));

ENGINE= BD_katsu DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

Esto en cada una de las tablas para que sea compatible más allá de los caracteres alfanuméricos estándar.