



Proyecto Final SQL

Academia Baile Mambolaye

Alumno: Ernesto Vega

Profesor: Miguel Rodas

Tutor: Juan Rabazzi

Contenido

Pto.		Pág.
1.-	Introducción	3
2.-	Objetivo	3
3.-	Modelo de negocio	3
4.-	Situación problemática	3
5.-	Diagrama E-R	4
6.-	Descripción de tablas	5
7.-	Scripts SQL objetos	7
8.-	Script SQL inserción	7
9.-	Informes generados	8
10.-	Herramientas y tecnologías usadas	13
11.-	Respaldo de información (BackUp)	14
16.-	Scripts DCL & TCL	16

1.- Introducción

Mediante el presente documento contiene toda la información recopilada, scripts generados y todos los requerimientos que han sido solicitados para presentar el Proyecto Final SQL Comisión 31270 de CoderHouse.

Los scripts SQL que se detallan en este proyecto se encuentra en el directorio **Scripts** en el presente [repositorio](#), al inicio de cada descripción se indica el nombre del respectivo archivo con su extensión y dirección URL, esto con el fin de facilitar su ubicación.

2.- Objetivo

Crear una Bases de Datos relacional para llevar un mejor registro de alumnos, actividades, clases impartidas y demás personas involucradas en la Academia Mambolaye.

3.- Modelo de Negocio

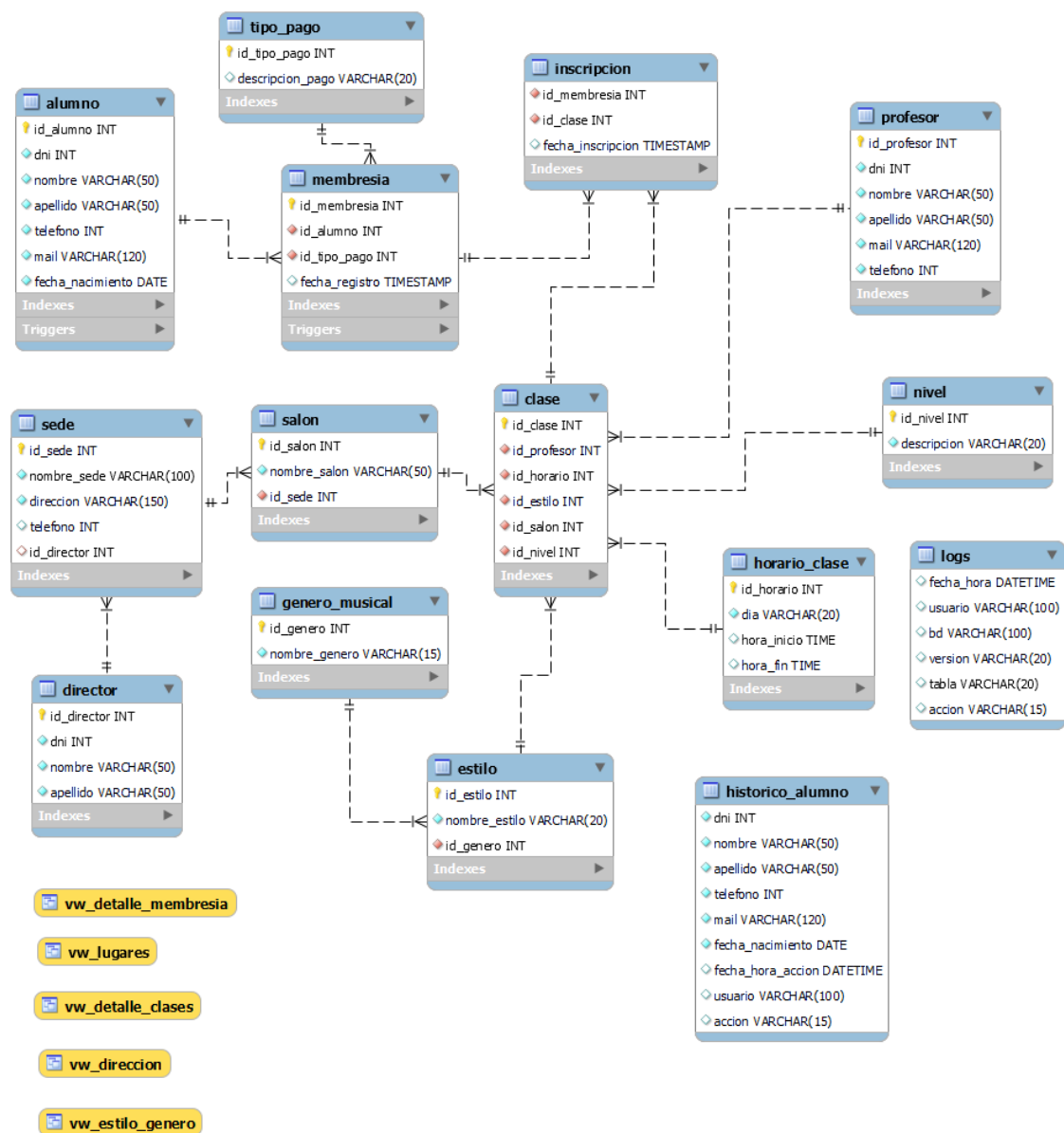
La Academia Mambolaye es una escuela de baile que se dedica a enseñar diversos estilos de baile como Mambo (Salsa on2), Mambo Lady Style, Bachata Estilo Pacheco y próximamente se irán agregando más estilos.

Mambolaye se fundó en Venezuela y actualmente cuenta con un solo espacio de enseñanza en CABA ubicado en Almagro, con miras de expansión. El nombre de la academia está compuesto por dos palabras **Mambo** que es el estilo de baile basado en la Salsa y **Laye** que significa calle, la fusión de estas dos se defino como *el estilo de baile con un toque propio de la persona que solo se obtiene fuera de salones de baile*.

4.- Situación Problemática

Actualmente la academia no cuenta con un sistema de registros para los actores ni actividades que se realizan en ella, se maneja de manera manual y la automatización es casi nula, por ende, con el presente proyecto aplicando un modelo de Bases de Datos Relacional se busca resolver esta problemática y así poder llevar un mejor registro de todo lo que se realiza en la institución.

5.- Diagrama E-R



6.- Descripción de tablas

alumno (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_alumno	int	autoinc	x	x		Datos de los alumnos de la academia
dni	int		x			
nombre_alumno	varchar	50	x			
apellido_alumno	varchar	50	x			
telefono	int		x			
mail	varchar	120	x			
fecha_nacimiento	date	10	x			

tipo_pago (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_tipo_pago	int	autoinc	x	x		Informacion de los metodos de pago
descripcion_pago	varchar	20				

Membresia (TT)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_membresia	int	autoinc	x	x		Datos de la membresia de cada alumno
id_alumno	int		x		x	
id_tipo_pago	int		x		x	
fecha_registro	Timestamp		x		x	

profesor (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_profesor	int	autoinc	x	x		Contiene informacion de los profesores de baile
dni	int		x			
nombre	varchar	50	x			
apellido	varchar	50	x			
mail	varchar	120	x			
telefono	int		x			

genero_musical (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_genero	int	autoinc	x	x		Descripcion del genero musical
nombre_genero	varchar	15	x			

estilo (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_estilo	int	autoinc	x	x		Estilos de baile relacionados con los generos musicales
nombre_estilo	varchar	20	x			
id_genero	int		x		x	

horario_clase (TD)						
Campos	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_horario	int	autoinc	x	x		Tabla de horarios de clases
dia	varchar	20	x			
hora_inicio	Time					
hora_fin	time					

director(TD)						
Campos	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_director	int	autoinc	x	x		Datos de los directores de las sedes
dni	int		x			
nombre	varchar	50	x			
apellido	varchar	50	x			

sede (TD)						
Campos	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_sede	int	autoinc	x	x		Contiene informacion de la(s) sede(s) de la academia
nombre_sede	varchar	100	x			
direccion	varchar	150	x			
telefono	int					
id_director	int				x	

nivel (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_genero	int	autoinc	x	x		Niveles de baile
descripcion	varchar	20	x			

salon (TD)						
Campos	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_salon	int	autoinc	x	x		Contiene datos de salon de practica
nombre_salon	varchar	50	x			
id_sede	int		x		x	

clase (TD)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_clase	int	autoinc	x	x		Contiene informacion sobre las clases que brinda la academia
id_profesor	int		x		x	
id_horario	int		x		x	
id_estilo	int		x		x	
id_salon	int		x		x	
id_nivel	int		x		x	

inscripcion (TH)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
id_registro	int	autoinc	x	x		Contiene datos de la
fecha_registro	datetime	16	x			

tipo_pago	varchar	20	x			inscripcion de cada alumno
id_clase	int		x		x	
id_alumno	int		x		x	

logs (TT)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
fecha_hora	datetime	autoinc	x	x		Contiene informacion sobre las clases que brinda la academia
usuario	varchar	100	x		x	
bd	varchar	100	x		x	
version	varchar	20	x		x	
tabla	varchar	20	x		x	
accion	varchar	15	x		x	

historico_alumnos (TT)						
Campo	Tipo	Longitud	NOT NULL	PK	FK	Descripcion
dni	int		x			Contiene informacion sobre las clases que brinda la academia
nombre	varchar	50	x			
apellido	varchar	50	x			
telefono	int		x			
mail	varchar	120	x			
fecha_nacimiento	date		x			
fecha_hora_accion	datetime		x			
usuario	varchar	100	x			
accion	varchar	15	x			

7.- Scripts SQL Objetos

7.1.- Crear Schema y tablas [Scripts/academia_mambolaye.sql](#)

7.2.- Vistas de la BD [Scripts/ScripVisVega.sql](#)

7.3.- Funciones [Scripts/FormatoVega.sql](#)

7.4.- Stored Procedures [Scripts/ StoredVega.sql](#)

7.5.- Triggers [Scripts/ TriggersVega.sql](#)

8.- Script SQL inserción

Los registros de la BD fueron cargados mediante script ([Scripts/ScriptInsVega.sql](#)), no se dispuso de algún datasets, motivado que el propósito de este proyecto es hacerlo muy cercano a la realidad actual y planes a futuro de la academia.

9.- Informes generados

9.1.- Vistas

Vista 1: vw_direccion

Objetivo: Poder observar las sedes de la academia y sus respectivos directores en una sola consulta.

Descripción: Se realiza una consulta a la tabla *sede* (*nombre_sede*), luego se realiza un join de la concatenación con la tabla *director* (concatenación campos *nombre* y *apellido*), la asociación de los campos se realiza mediante el *id_director* entre ambas tablas

Ejemplo:

	Sede	Director
►	La Huella	Manuel Rosas
	Llano	Beatriz Rosas

Vista 2: vw_estilo_genero

Objetivo: Debido que los estilos de baile pueden variar para un género musical, mediante esta vista se puede obtener el género y sus distintos tipos de estilo.

Descripción: Se realiza consulta a la tabla *estilo* (*nombre_estilo*), posteriormente se ejecuta un join hacia la tabla *genero_musical* (*nombre_genero*), la asociación entre las dos tablas se realiza mediante el campo *id_genero*.

Ejemplo:

Estilo de Baile	Genero Musical
Mambo on2 (General)	Salsa
Mambo on2 (Ladies)	Salsa
Casino	Salsa
Pacheco	Bachata
Urbano	Bachata
Free Style / Urbano	Reggaeton

Vista 3: vw_detalle_clases

Objetivo: La tabla clase está conformada por distintos ID que pertenecen a otras tablas, esto a la vista humana es poco entendible, por ende, esta vista asocia los distintos ID con su respectiva tabla trayendo la descripción de los mismos y así la información sea más acorde a la vista del usuario.

Descripción: Esta vista se basa en la consulta a las tablas *clase* (*id_clase*), *estilo* (*nombre_estilo*), *profesor* (concatenación *nombre* y *apellido*), *horario_clase* (concatenación *dia*, *hora_inicio*, *hora_fin*), *nivel* (*descripción*) y *salón* (*nombre_salon*) las asociaciones para mostrar los valores correspondientes se realizan mediante varios join's de los id's que conforman la tabla clase y que son llaves foráneas de las otras tablas antes mencionadas.

Ejemplo:

id_clase	Estilo Baile	Profesor(a)	Horario	Nivel	Salon
2	Mambo on2 (Ladies)	Beatriz Rosas	Jueves 17:30:00 19:00:00	Inicial	Subteraneo
1	Mambo on2 (General)	Manuel Rosas	Jueves 17:30:00 19:00:00	Basico	Amarillo
3	Casino	Erick Lugo	Jueves 19:00:00 20:30:00	Avanzado	Contemporaneo
4	Casino	Aymara Orta	Jueves 19:00:00 20:30:00	Avanzado	Libertar

Vista 4: vw_lugares

Objetivo: Observar cuales son los salones que se encuentran en cada sede en una sola consulta.

Descripción: Realiza consulta a la tabla *sede* (*nombre_sede*, *dirección*, *teléfono*) y a la tabla *salón* (*nombre_salón*) asociando los distintos valores mediante el *id_sede*.

Ejemplo:

nombre_sede	direccion	telefono	Salon
La Huella	CABA, Bulnes 892	1161989945	Amarillo
La Huella	CABA, Bulnes 892	1161989945	Subteraneo
La Huella	CABA, Bulnes 892	1161989945	Contemporaneo
Llano	Av. San Juan 1555	NULL	Llanito
Llano	Av. San Juan 1555	NULL	Metro
Llano	Av. San Juan 1555	NULL	Libertar

Vista 5: vw_detalle_membresia

Objetivo: La tabla membresía está conformada por ID's que hacen referencia a las otras tablas, esto a la vista del usuario es inentendible o de difícil comprensión, por ello esta vista buscar observar toda la información de los alumnos que han obtenido una membresía y con cual método de pago fue registrado ordenada de manera ascendente.

Descripción: La presente vista realiza una consulta general a la tabla *alumno* y a *tipo_pago* (*descripcion_pago*), y luego realiza un join con la tabla *membresia* asociando los distintos ID's que conforman dicha tabla.

Ejemplo:

id_alumno	dni	nombre	apellido	telefono	mail	fecha_nacimiento	descripcion_pago
1	96036597	Ernie	Vega	1161989945	ernesvein18@gmail.com	1989-06-18	Efectivo
2	30020758	Trula	Coleford	1048063419	tcoleford0@mapquest.com	1999-02-03	Visa
3	30031204	Adelice	Gores	1993368020	agores2@wunderground.com	1999-12-28	MasterCard
4	30054027	Sherwood	Ible	1120795570	sible1@hibu.com	1998-12-17	MasterCard
5	30051640	Sheridan	Ream	1002763570	sream3@cyberchimps.com	1996-07-10	Efectivo
6	30028931	Nance	Benedidick	1612743010	nbenedidick4@blogs.com	1990-08-10	Visa
7	30022815	Hillary	Roloff	1076166546	hroloff5@flickr.com	1993-11-01	Transfer / Deb. Au.
8	30050121	Alecia	Sunley	1131859081	asunley6@wikia.com	1992-08-12	Efectivo
9	30025176	Devon	MacGaughie	1140968817	dmacgaughie7@si.edu	1996-03-08	MasterCard
10	30030484	Cordie	Thorn	1158247105	cthorn8@eventbrite.com	1991-02-25	Visa
11	30046159	Doro	Gillino	1182206525	dgillino9@godaddy.com	1994-05-22	Efectivo

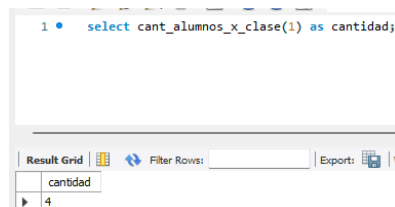
9.2.- Funciones

Función 1: cant_alumnos_x_clase

Objetivo: Contabilizar cuantos alumnos inscritos en una clase especifica indicando el id de la clase.

Descripción: La función recibe por parámetro un valor entero (INT) el cual debe corresponder a una clase y la función realiza un SELECT COUNT de la tabla *inscripción*.

Ejemplo:



```
1 • select cant_alumnos_x_clase(1) as cantidad;
```

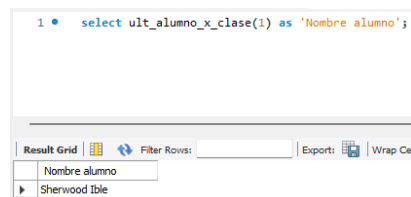
cantidad
4

Función 2: ult_alumno_x_clase

Objetivo: Retornar el nombre del último alumno que se registró a una clase en especifica.

Descripción: Se ingresa un entero que corresponde al ID de la clase a consultar y la función retornar una cadena de texto (VARCHAR) con el nombre y apellido del estudiante.

Ejemplo:



```
1 • select ult_alumno_x_clase(1) as 'Nombre alumno';
```

Nombre alumno
Sherwood Tble

9.3.- Stored Procedures

SP 1: sp_orden_x_campo

Objetivo: Mostrar datos ordenados de una tabla indicada por el usuario, esto permitirá al usuario que trabaje directamente en la BD ahorrar tiempo y código de sentencia.

Descripción: Para hacer uso del SP solo deberá ingresar el nombre de la tabla, la columna y si es en orden ascendente o descendente.

```
1 • CALL academia_mambolaye.sp_orden_x_campo('alumno', 'nombre', 1);
2
```

Ejemplo:

id_alumno	dni	nombre	apellido	telefono	mail	fecha_nacimiento
3	30031204	Adelice	Gores	1993368020	agores2@vunderground.com	1999-12-28
8	30050121	Alecia	Sunley	1131859081	asunley6@wikia.com	1992-08-12
13	30001846	Alick	Bickerdike	1955141625	abickerdikeb@chronoenline.com	1989-09-03
26	30005215	Camel	Corran	1024000113	ccorran@reuters.com	1990-06-06
15	30040083	Corbet	Langstone	1904797776	clangstone@t-online.de	1990-08-24
10	30030484	Cordie	Thorn	1158247105	cthorn8@eventbrite.com	1991-02-25
9	30025176	Devon	MacGaughey	1140968817	dmacgaughey7@si.edu	1996-03-08
28	30036425	Dille	Gaffey	1084015500	dgeffeyq@free.fr	1996-04-24
11	30046159	Doro	Ollino	1182206525	dollino9@godaddy.com	1994-05-22
23	30046271	Elayne	Dunkis	1353671747	edunkis@de.vu	1995-12-12
18	30035585	Elissa	Hardwidge	1510837717	ehardwidge@yandex.ru	1998-03-24
1	96036597	Ernie	Vega	1161989945	ernavein18@gmail.com	1989-06-18
24	30018087	Hetty	Mangeot	1661452277	hmangeot@wikispaces.com	1989-09-05
7	30022815	Hillary	Roloff	1076166546	hroloff5@flickr.com	1993-11-01
21	30037639	Husein	Lukovic	1338813527	hukovicj@google.co.uk	1994-06-13
14	30022092	Kamilah	Flanne	1091384785	kflanne@usatoday.com	1995-08-05
20	30050393	Linc	Gretton	1100077087	lgrettoni@last.fm	1989-08-11
6	30028931	Nance	Benedick	1612743010	nbenedick4@blogs.com	1990-08-10
30	30038173	Phedra	Tague	1973388139	ptague@sfgate.com	1992-12-06
25	30034139	Rainer	Ayer	1851903137	rayern@nymag.com	2000-02-14
31	30038571	Raphael	Minto	1132114412	rmintot@about.com	1990-06-22
5	30051640	Sheridan	Rean	1002763570	sream3@cyberchimps.com	1996-07-10
4	30054027	Sherw...	Ible	1120795570	sible1@hibu.com	1998-12-17
27	30031511	Sybla	Zellmer	1326542944	szellmerp@marriott.com	1997-07-10
2	30020758	Trula	Coleford	1048063419	tcoleford0@mapquest.com	1999-02-03
29	30041200	Tuesday	Cromarty	1376265386	tcromartyr@goo.gl	1997-03-31
22	30001690	Wade	Carnow	1669056697	wcarnowk@shop-pro.jp	1989-12-17
12	30038462	Willabella	Cavan	1166494965	wcavana@nytimes.com	1993-03-25
16	30034364	Wilmette	Crichmer	1283077416	worichmere@bloglines.com	1997-08-06
19	30002525	Winston	Mea	1131866777	wmeah@hincartel.com	1992-04-30

SP 2: sp_delete_alumno

Objetivo: Eliminar alumno que ya no se encuentre en la academia, el SP ayuda a evitar la posibilidad de eliminar registros por equivocación y omisión de la cláusula WHERE. Cuenta con una validación de longitud por si el usuario no ingresa por completo la secuencia de números que conforman un DNI.

Descripción: Se debe ingresa el numero de DNI a eliminar de la tabla alumno.

```
1 • set sql_safe_updates = 0;
2 • CALL sp_delete_alumno(96036597);
3 • select * from alumno where dni = 96036597;
4
5
```

Ejemplo:

id_alumno	dni	nombre	apellido	telefono	mail	fecha_nacimiento
*	NULL	NULL	NULL	NULL	NULL	NULL

9.4.- Triggers

General: Se idearon y crearon varios disparadores para el registro de algunas sentencias DML que el usuario pudiese realizar en las tablas membresías y alumnos.

Nomenclatura de los triggers:

TipoTrigger	BEF = before (antes de ejecutar la acción)
	AFT = after (luego de ejecutar la acción)
AccionDML	DEL = Eliminar registro
	INS = Insertar registro
	UPD = Actualización registro

Sintaxis de uso:

CREATE TRIGGER `<TipoTrigger>_<AccionDML>_<NombreTabla>`

... Bloque de acción del trigger...

TR Logs:

Objetivo: Generar un registro de acciones DML que se puedan realizar sobre las tablas membresía y alumnos.

Descripción: Por cada acción Insert, Update y/o Delete que realice el usuario se realizara un registro en la tabla logs con la siguiente información: Fecha de la acción, nombre de usuario, nombre BD, versión de la BD, nombre tabla (membresía o alumno) y acción (INSERT/UPDATE/DELETE).

Ejemplo:

```

1  -- Borrado previo
2  • delete from alumno
3  where id_alumno = 3;
4  -- Consulta tabla logs
5  • select * from logs;
```

fecha_hora	usuario	bd	version	tabla	accion
2022-08-09 14:49:41	root@localhost	academia_mambolaye	8.0.29	alumno	DELETE

TR Histórico

Objetivo: Generar una copia de los registros que se insertan o actualizan en la tabla alumnos generando de este modo un respaldo de los alumnos agregados y actualizaciones que se puedan ejecutar en la tabla.

Descripción: Por cada acción INSERT y UPDATE que realiza el usuario sobre la tabla alumno se generara un registro en la tabla historico_alumno, con todos los campos hayan sido modificados o no, seguido de fecha_hora_accion, nombre de usuario y breve descripción de la acción que genero ese registro

Ejemplo:

```

1  -- Insert en la tabla alumno
2  • INSERT INTO academia_mambolaye.alumno(id_alumno,dni,nombre,apellido,telefono,mail,fecha_nacimiento)
3  VALUES (1,96036597,'Ernie','Vega',1161989945,'ernesvein18@gmail.com','1989-06-18');
4
5  -- Update en la tabla alumno
6  • UPDATE academia_mambolaye.alumno
7  SET nombre = 'Ernesto'
8  WHERE id_alumno = 1;
9
10 -- Consulta a tabla historico
11 • select * from historico_alumno;
12

```

dni	nombre	apellido	telefono	mail	fecha_nacimiento	fecha_hora_accion	usuario	accion
96036597	Ernie	Vega	1161989945	ernesvein18@gmail.com	1989-06-18	2022-08-09 14:54:08	root@localhost	INSERT
96036597	Ernesto	Vega	1161989945	ernesvein18@gmail.com	1989-06-18	2022-08-09 14:54:12	root@localhost	UPDATE

10.- Herramientas y tecnologías usadas

Draw.io: Se realizo en primera instancia para la creación y moldeado de las tablas, posibles campos que contendrían cada tabla y bosquejo de Diagrama ER.

Microsoft Excel: Se utilizo para el armado de los detalles correspondientes a cada tabla y así poder asegurar el tipo de datos que puede contener cada campo

MySQL Workbench: SGBD usado para la creación de los scripts SQL de todos los objetos de la BD de este proyecto.

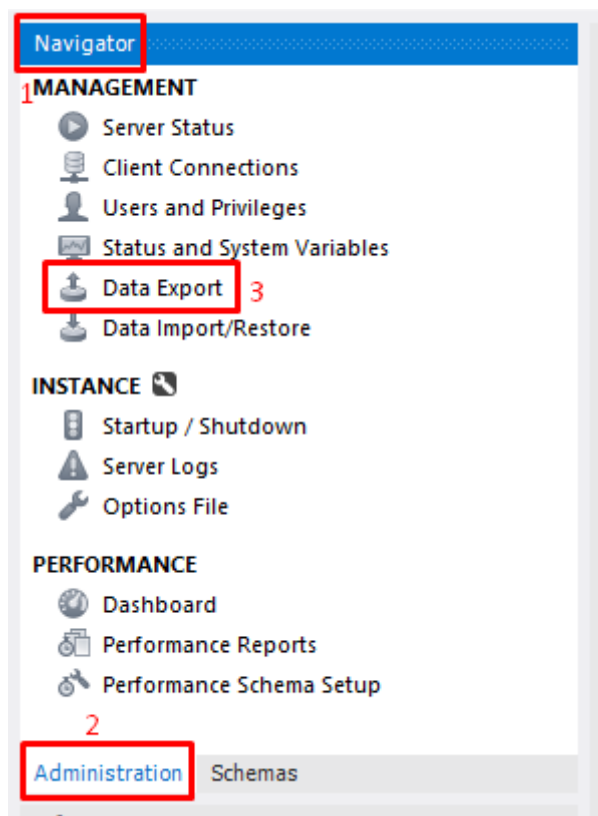
Git: Herramienta de control de versiones local, de este modo pude tener un control de todos los cambios que se pudiese realizar a lo largo del proyecto y poder comparar en caso de algún error.

GitHub: Plataforma basada en sistema de GIT que permite la publicación de repositorios públicos, fue usada para la entrega de los desafíos de toda la comisión.

11.- Respaldo de información (BackUp):

Se realiza un paso a paso del procedimiento a seguir para la creación del SQL de respaldo correspondiente a la BD del proyecto:

Paso 1.- Navigator > Administration > Data Export



Paso 2.- Seleccionar Schema > Dump Data Only > Ingresar Directorio Destino > Start Export

Administration - Data Export

localhost
Data Export

Object Selection | Export Progress

Tables to Export

Exp... Schema

☒ academia_mambolaye 1

☐ examen_practico

☐ fifa

☐ gamblers

☐ sakila

☐ sys

☐ world

Refresh

Exp... Schema Objects

Dump Structure and Data

2 **Dump Data Only**

Dump Structure Only

Objects to Export

☐ Dump Stored Procedures and Functions

☐ Dump Events

Export Options

☒ Export to Dump Project Folder <DIRECTORIO_DESTINO_BACKUP> 3

Each table will be exported into a separate file. This allows a selective restore, but may be slower.

☐ Export to Self-Contained File C:\Users\jernes\Documents\dumps\Dump20220805.sql

All selected database objects will be exported into a single, self-contained file.

☐ Create Dump in a Single Transaction (self-contained file only)

☐ Include Create Schema

Press [Start Export] to start...

4 **Start Export**

Este proceso generará un archivo .sql con el nombre que se le haya indicado el cual contendrá toda la información de la BD seleccionado en el paso 2.

Para este proyecto se generó el archivo [BackupVega.sql](#) que contiene el respaldo de la información del proyecto.

Para la recuperación o importación de la información solo se debe seleccionar la opción **Data Import/Restore** del paso 1 y seleccionar el archivo que contiene la información que se desea recuperar.

12.- Scripts DCL & TCL

- **DCL Data Control Language ([Scripts/SentenciasVega.sql](#))**: Script para la creación de usuarios user_1 con todos los permisos sobre la BD y tablas, y user_2 con solo permiso de lectura, inserción y actualización de registros.
- **TCL Transaction Control Language ([Scripts/TCLVega.sql](#))**: Script del desafío 20 de la cursada donde se ejecutan acciones de borrado con y sin Rollback, y además se realiza la implementación de SAVEPOINT con registros de prueba