WEB_MOVIES

La base de datos de web_movies está pensada para usarse en una aplicación web de streaming de películas. Con el objetivo de almacenar datos de películas y que sean accesibles a cada que se registre en el servicio web.

Índice de archivos

web_movies.sql : Script de creación de la base de datos y las tablas

Insercion.sql: Script de inserción de datos en la base de datos

Objetos.sql: Script de creación de los objetos de la base de datos como vistas, funciones, create procedure y triggers

ProbarObjetos.sql: Contiene sentencias para probar la funcionalidad de cada objeto

TABLAS

Tabla usuario

CONTIENE INFORMACIÓN DEL USUARIO

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
TRUE		id_usuario	BIGINT	TRUE		IDENTIFICADOR DE USUARIO
		nombre	VARCHAR	TRUE	40	NOMBRE DE USUARIO
		apellido	VARCHAR	TRUE	40	APELLIDO DEL USUARIO
		nombre_usuario	VARCHAR	TRUE	40	USERNAME DEL USUARIO
		pais	VARCHAR	TRUE	40	PAIS DEL USUARIO
		email	VARCHAR	TRUE	40	EDAD DE USUARIO
		contraseña	VARCHAR	TRUE	20	CONTRASEÑA DE USUARIO
		edad	INT	TRUE		EDAD DE USUARIO

Tabla película

PELÍCULAS

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
TRUE		id_pelicula	BIGINT	TRUE		IDENTIFICADOR DE PELÍCULA
		titulo	VARCHAR	TRUE	20	NOMBRE DE LA PELÍCULA
		fecha_estreno	DATE	TRUE		FECHA DE ESTRENO DE LA PELICULA
		duracion	INT	TRUE		DURACIÓN DE LA PELÍCULA EN SEGUNDOS
		valoracion	FLOAT	TRUE		VALORACIÓN DE LA PELÍCULA

Tabla director

DIRECTORES

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
TRUE		id_director	BIGINT	TRUE		IDENRIFICADOR DE DIRECTOR
		nombre	VARCHAR	TRUE	40	NOMBRE DEL DIRECTOR
		apellido	VARCHAR	TRUE	40	APELLIDO DEL DIRECTOR
		nacionalidad	VARCHAR	TRUE	40	NACIONALIDAD DEL DIRECTOR

Tabla de genero

INFORMACIÓN SOBRE LOS GÉNEROS

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
PK		id_genero	BIGINT	TRUE		IDENTIFICADOR DEL GÉNERO
		nombre	VARCHAR	TRUE	40	NOMBRE DEL GÉNERO
		descripción	VARCHAR	TRUE	400	DESCRIPCIÓN DEL GÉNERO

Tabla de caracteristicas

CARACTERISTICAS DE CADA PELICULA

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
	TRUE	id_pelicula	BIGINT	TRUE		IDENTIFICADOR DE PELICULA
	TRUE	id_director	BIGINT	TRUE		IDENTIFICADOR DE DIRECTOR
		nacionalidad	VARCHAR	TRUE	40	NACIONALIDAD DE LA PELICULA
		idioma_original	VARCHAR	TRUE	40	IDIOMA ORIGINAL DE LA PELICULA
		productora	VARCHAR	TRUE	160	PRODUCTORA DE LA PELICULA
		sipnosis	VARCHAR	TRUE	600	SINOPSIS DE LA PELICULA

Tabla de valoraciones

CONTIENE LAS VALORACIONES DE LOS USUARIOS EN LAS PELICULAS

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
	TRUE	id_usuario	BIGINT	TRUE		IDENTIFICADOR DE USUARIO
	TRUE	id_pelicula	BIGINT	TRUE		IDENTIFICADOR DE PELÍCULA
		fecha	DATE	TRUE		FECHA EN QUE SE HA VISTO POR EL USUARIO
		valoracione	FLOAT	TRUE		VALORACION DEL USUARIO HACIA PELICULA

Tabla de pelis_vistas

CONTIENE INFORMACION DE CADA VISUALIZACION DEL USUARIO

PK	FK	COLUMN	TYPE	NOT NULL	LEN	NOTES
	TRUE	id_pelicula	BIGINT	TRUE		IDENTIFICADOR DE PELICULA
	TRUE	id_usuario	BIGINT	TRUE		IDENTIFICADOR DE USUARIO
		tiempo_visto	INT	TRUE		TIEMPO DE VISUALIZACION EN SEGUNDOS
		estado	VARCHAR	TRUE	20	fraccion de visualizacion en relacion a la duracion de la pelicula

OBJETOS

VISTAS

Vista 1 (peliculas_generos): Enlaza cada película con su género en el catálogo con un join, pudiendo cada película tener mas de 1 genero enlazada.

Objetivo: clasificar cada película con los géneros.

Tablas que componen: catálogo, pelicula y genero.

Vista 2 (peliculas_top): De pelis_vistas se hace un join a película a través de id_pelicula, las agrupa por id_pelicula y un having para que muestre solo las que tienen visualización. Por último, selecciona algunas columnas de película y un COUNT al dato de visto en pelis_vistas con alias de películas_visadas y las ordena de forma descendiente.

Objetivo: Identificar que películas fueron las más vistas.

Tablas que la componen: pelis_vistas y pelicula.

Vista 3 (fechas_fuertes): De película hay un left join a pelis_vistas a través de id_pelicula, las agrupa por id_pelicula seleccionando id_pelicula, titulo y un formateo al promedio de fechas de visualización con el alias de fechas_fuertes_visadas.

Objetivo: Ver en que fechas tuvo su punto top cada película luego de su estreno.

Tablas que la componen: película y pelis_vistas.

Vista 4 (nacionalidad_peliculas): De película se agrupa por nacionalidad y se selecciona nacionalidad, un COUNT de id_pelicula y suma la duracion.

Objetivo: Ver cuantas películas tiene cada país con su total de duración.

Tablas que la componen: pelicula.

Vista 5 (pelis_por_director): De director enlaza con un join a características con el id_director, agrupa por id_director, selecciona las columnas de director y un COUNT de id_pelicula y por último las ordena ascendientemente.

Objetivo: Ver cuantas películas tiene cada director registrado.

Tablas que la componen: director y características.

Funciones

Función 1 (fecha_diferencia): Utiliza DATEDIFF para diferenciar en días entre dos dates, se concatena con ‘dias’ y devuelve un varchar.

Objetivo: Ver los días en que cada usuario pudo ver la película luego de su estreno

Manipula la tabla de película para obtener el dato de su fecha de estreno.

Función 2 (formatear_duracion): Un int en segundos lo transforma en hora, minuto y segundo

Objetivo: Ver la duración de segundos en horas, minutos y segundos.

No manipula ninguna tabla ni dato mas que las variables y el parámetro int.

Store Procedures

Store Procedure 1 (ordenar_pelicula): Utilizar ifs y variables para ordenar según los parámetros de columna y orden de forma ascendiente o descendiente.

Objetivo: Ordenar una columna.

Aporta Rapidez y legibilidad en visualizar alguna tabla y manipula la tabla de pelicula.

Store Procedure 2 (insertar_pelicula): Insertar con los datos de los parametros en pelicula.

Objetivo: Insertar una fila en la tabla película.

Aporta rapidez a la hora de insertar datos en la fila película, manipulando solo esa tabla.

Store Procedure 3 (usuario_update): Cambiar un dato en algún campo de la tabla usuario teniendo como parámetros el id de usuario, el nombre del campo y el dato nuevo.

Objetivo: Cambiar un dato en la tabla usuario

Aporta flexibilidad a la hora de hacer un update para la tabla usuario.

Triggers

Trigger 1 (new_usuarios):

Antes del primer trigger se crea la tabla new_usuarios.

Se dispara antes de que se inserte una nueva fila en la tabla usuario. Se declara la variable ‘un’ en la que se le guarda si es que hay en la BD el nombre_usuario igual al nuevo nombre_usuario a insertar. En caso de que lo haya saldrá un error de ‘Nombre de Usuario ya en uso. Intente con otro’ y si no se procede a insertar en la tabla new_usuarios los datos y termina el trigger.

Objetivo: Que no haya dos nombre_usuario repetidos en la DB y registrar su fecha de creación.

Localizado en la tabla usuario e interactúa con la tabla new_usuarios.

Trigger 2 (pelis_update):

Antes del segundo trigger se crea la tabla log_pelis.

Se dispara después de un update en la tabla pelicula. Se crean variables de tipo int para cada columna de la tabla pelicula y la fecha y hora de modificación. Un if para comparar si se ha cambiado un dato de cada columna, en caso de que si se inicializa su variable en 1 o sino queda en 0. Y finalmente se inserta en log_pelis las variables declaradas en el trigger más el id_pelicula.

Objetivo: Localizar cada cambio en los datos de cada pelicula.

Se encuentra en la tabla pelicula e interactúa con la tabla log_pelis.

Trigger 3 (valoración_insert): Se dispara después de hacer un insert en la tabla de valoraciones. Se declara la variable promedio en la que se almacena justamente el promedio de los datos de valoraciones en la pelicula en la que se insertó la valoración y luego se actualiza el dato de valoración de la tabla pelicula.

Objetivo: Actualizar los datos de valoración en la tabla pelicula

Se encuentra en la tabla valoraciones