

## Pautas

Lea **atentamente** las pautas para la resolución de los ejercicios planteados y la entrega del trabajo práctico.

- (a) Este TP **debe resolverse individualmente**.
- (b) Genere el archivo `apellido_nombre_unqfaces.sql`, reemplazando las palabras *apellido\_nombre* por los correspondientes al alumno. Ejemplo: `arevalo_gabriela_unqfaces.sql`.
- (c) Guarde todas las consultas pedidas en ese archivo.
- (d) Guarde las descripciones o aclaraciones en el mismo archivo utilizando comentarios de sql. Parte del proceso de corrección se realiza mediante parseo automático, es importante el uso de comentarios en todo lo que no sean sentencias sql.
- (e) Escriba todas las consultas de forma prolija. Es importante que entre cada consulta haya espacios de líneas suficientes. Termine cada consulta con ; (punto y coma).
- (f) Es importante que cada consulta comience en una línea aparte. Esto disminuye la cantidad de errores y facilita la corrección.
- (g) Esta es una instancia de evaluación, aclare todo lo que considere importante, aún lo que considere trivial.
- (h) Envíe el archivo `apellido_nombre_unqfaces.sql` con la resolución del trabajo a `tpi-doc-bd@listas.unq.edu.ar` escribiendo en el Asunto del email `[UNQ - BD] <apellido_nombre> - Entrega TP SQL`, por ejemplo `[UNQ - BD] arevalo_gabriela - Entrega TP SQL`.
- (i) Puede enviar consultas al `tpi-doc-bd@listas.unq.edu.ar` escribiendo en el Asunto del email `[UNQ - BD] Consulta TP Ejercicio(s) XX` identificando XX con el/los número(s) de ejercicio(s) sobre el/los cuales quiere hacer preguntas.
- (j) La fecha límite de entrega es el **día del parcial de normalizacion** inclusive. Si termina el TP antes de la fecha límite, puede enviarlo también.

# 1. Resumen de SQL

La sintaxis del DML de SQL puede resumirse de la siguiente manera:

```
SELECT [ALL | DISTINCT] <atributos>
FROM <tablas>
[WHERE <expresion condicional>]
[GROUP BY <atributo(s)>]
[HAVING <expresion condicional>]
[ORDER BY <columna(s)>]
```

donde:

- <atributos> es la información a obtener de la base de datos.
- FROM <tablas> especifica de qué tablas se obtiene la información buscada.
- WHERE <expresion condicional> expresa una condición que deben cumplir las filas de la consulta resultante.
- GROUP BY <atributo(s)> permite formar consultas agrupadas para extraer información global sobre los grupos formados.
- HAVING <expresion condicional> es condición sobre los grupos formados.
- ORDER BY <columna(s)> ordena por una o varias columnas.
- DISTINCT: No permite la aparición de filas idénticas.
- <expresion condicional>: formada por un conjunto de predicados combinados mediante los operadores lógicos AND, OR y NOT.
- Los predicados utilizados permiten comparar columnas:
  - predicados de comparación: =, <>, >, <, >=, <=.
  - predicado BETWEEN: permite comprobar si un escalar está en un rango.
  - predicado IN: permite comprobar si el valor está dentro de un conjunto.
  - predicado IS NULL: permite comprobar si el valor es nulo.

# 2. Estilo requerido para el código SQL

## 1. Uso de mayúsculas y minúsculas

- a) Palabras reservadas del lenguaje (select, on , where, etc.): MAYÚSCULAS
- b) Nombres de tablas: minúsculas y singular
- c) Nombres de atributos: minúsculas

## 2. Organización del código

- a) Un renglón para todo lo relativo al SELECT
- b) Un renglón para el FROM
- c) Un renglón para cada tabla joineada
- d) Un renglón para el WHERE
- e) Un renglón para cada <expresion condicional> del WHERE

## Ejercicio 1: Películas “ArgenfliX”

Debido al gran éxito que tiene mundialmente Netflix en todo el mundo, un grupo de desarrolladores argentinos idearon la aplicación ArgenfliX, donde los usuarios pueden ver y calificar cualquier tipo de películas y series. El sistema donde se modela ArgenfliX tiene una base de datos con las siguientes relaciones:

```
usuario<nombre_usuario: VARCHAR(35) PK, nombre_y_apellido: VARCHAR(60),  
    contrasena: VARCHAR(16)>
```

```
pelicula<id_pelicula: INT PK, nombre_pelicula: VARCHAR(40), genero: VARCHAR(20),  
    duracion: INT, calificacion: INT, nombre_actor: VARCHAR(50) FK,  
    nombre_director: VARCHAR(50) FK>
```

```
serie<nombre_serie: VARCHAR(30) PK, anho_serie: INT PK, genero: VARCHAR(20),  
    temporadas: INT, calificacion: INT, nombre_actor: VARCHAR(50) FK,  
    nombre_director: VARCHAR(50) FK>
```

```
actor<nombre: VARCHAR(50) PK, edad INT, anhos_activo INT>
```

```
director<nombre_director: VARCHAR(50) PK, edad: INT, nacionalidad: VARCHAR(20)>
```

Se asume que no existen dos actores (ni dos directores) que se llamen igual, y que tanto los actores como los directores pueden actuar y dirigir en series y películas.

La actividad de los usuarios queda registrada en las siguientes tablas:

```
vio_pelicula<nombre_usuario: VARCHAR(35) PK FK, id_pelicula: INT PK FK>
```

```
vio_serie<nombre_usuario: VARCHAR(35) PK FK, nombre_serie: VARCHAR(50) PK FK,  
    anho_serie: INT PK FK>
```

Dada esta BD, se piden las siguientes consultas usando SQL:

### 1. DDL:

- Crear todas las relaciones del ejercicio considerando los tipos de datos de cada atributo, las claves primarias y las claves foráneas entre las diferentes relaciones.
- Una vez creada la base de datos, el analista detecta que necesitamos modelar el concepto de actor fetiche. Se asume que cada director tiene sólo un actor fetiche y que dicho actor se encuentra registrado en el sistema. Escriba el código SQL que lleve adelante dicha modificación y ejecútelo.
- Ejecutar las consultas del documento `argenfliX_test_ddl.sql` para verificar las tablas.
- Una vez que el control haya sido ejecutado exitosamente, insertar los datos contenidos en el documento `argenfliX_dataset.sql`.

### 2. DML:

- Actualizar todas las películas que hayan sido protagonizadas por Steve Guttenberg o Ian Ziering asignándole una calificación de 10, que es la máxima calificación.
- Actualizar todas las películas que hayan sido dirigidas por directores argentinos asignándole una calificación de 10.
- Eliminar todas las series con menos de 3 temporadas.

### 3. Consultas:

- Obtener la cantidad de películas en las que haya actuado el actor Rose McGowan.

- b) Obtener la cantidad películas por nacionalidad del director, y el promedio de puntaje de las mismas. El resultado debe tener sólo los campos <nacionalidad, cantidad, puntaje\_promedio>. Y estar ordenado alfabeticamente por nacionalidad.
- c) Listar el nombre y la cantidad de temporadas de las series que hayan sido dirigidas por directores que hayan dirigido por lo menos alguna película.
- d) Obtener el nombre de los actores que actuaron en películas pero que no lo hicieron en series. Los resultados no deben tener registros repetidos.
- e) Obtener los nombres de las películas, actores y directores de las películas en las que un director dirigió a su actor fetiche.
- f) Obtener los nombres de la serie, el género y el nombre del usuario de las series que hayan sido vistas por los usuarios de nombre "RossGeller85" o "BreakingThrones".
- g) Obtener la tabla de usuarios ordenados de mayor a menor de acuerdo a la cantidad de películas que vieron. La tabla resultante debe tener 2 columnas: el nombre del usuario y la cantidad de películas vistas.
- h) Por cuestiones legales es necesario eliminar al usuario "DarthVader".
- i) Es necesario identificar a los directores que no hayan dirigido ninguna película. Presente al menos dos consultas equivalentes.
- j) De los actores, se requiere identificar al segundo de mayor edad.
- k) Crear una vista que muestre, de las 3 películas más vistas que no sean estadounidenses, cuál es su origen y la cantidad de veces que fue vista por los usuarios del sistema. La vista debe llamarse `otras_mas_vistas`, los campos resultantes deben denominarse `origen_pelicula` y `veces_vista`.