**Base de datos 1**

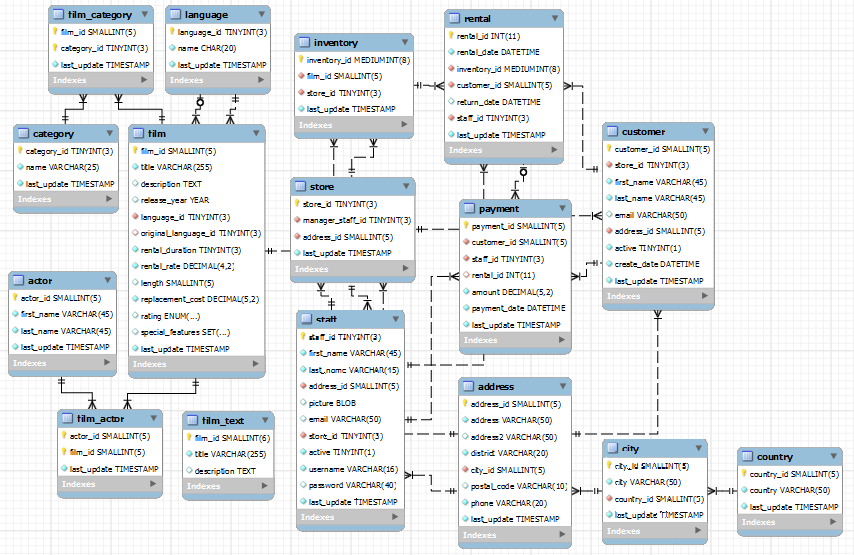
Alumno: Santiago Vietto

Docente: Federico Luis Garofalo

DNI: 42654882

Institución: UCC

Año: 2021



**Ejercicio 5: Gestión de un Video Club**

**Actividades:**

1)\_ Listar el nombre de la película 6, la cantidad de unidades en inventario y la cantidad de veces que fue alquilada. El formato y los valores del resultado de la consulta deben coincidir con la siguiente tabla:

|  |  |
| --- | --- |
| Nombre | AGENT TRUMAN |
| Unidades en inventario | 6 |
| Cantidad de alquileres | 21 |

\_ Solución en clase:

**SELECT** ‘Nombre’, film.title

**FROM** film

**WHERE** film.film\_id = 6

**UNION**

**SELECT** ‘Unidades en inventario’, **COUNT**( \* )

**FROM** inventory

**INNER JOIN** film **ON** inventory.film\_id = film.film\_id

**WHERE** film.film\_id = 6

**UNION**

**SELECT** ‘Cantidad de alquileres’, **COUNT**( \* )

**FROM** rental

**INNER JOIN** inventory **ON** rental.inventory\_id = inventory.inventory\_id

**INNER JOIN** film **ON** inventory.film\_id = film.film\_id

**WHERE** film.film\_id = 6

\_ Tenemos una tabla de 2x3. Al principio solo necesitamos el nombre de la película con ID 6, donde hacemos una SELECT a la tabla film, y que sea el ID 6. Tenemos que hacer que el primer título sea Nombre y para eso lo ponemos entre comillas en el SELECT, y para el segundo título que es AGENT TRUMAN, usamos el film.title. Luego con un UNION unimos la consulta siguiente, que consiste en obtener de la tabla inventario, todas las unidades de la película con ID 6, por eso unimos la tabla inventory con film mediante un INNER JOIN. Y por último unimos la última consulta que consiste en obtener de la tabla de alquileres o rental, la cantidad de alquileres de la película con ID 6, por eso mediante un INNER JOIN unimos desde rental, pasando por inventory, hasta llegar a film. Hacemos un COUNT( \* ), que es básicamente contar todas las tablas.

2)\_ Obtener una lista de los nombres de las categorías juntos a su correspondiente porcentaje del total de alquileres.

**Resultado**: 16 tuplas.

**SELECT** category.name, **COUNT** (\*) \* 100 / (**SELECT** **COUNT**( rental\_id ) **FROM** rental) **AS** ‘Porcentaje’

**FROM** category

**INNER JOIN** film\_category **ON** film\_category.category\_id = category.category\_id

**INNER JOIN** inventory **ON** inventory.category\_id = film\_category.film\_id

**INNER JOIN** rental **ON** rental.inventory\_id = inventory.inventory\_id

**GROUP BY** category.category\_id, category.name

\_ Determinamos las entidades primero, partimos de category hasta rental, es decir, tenemos category, film\_category, film no es necesario porque la tabla pasarela conecta con inventory directamente, y por último rental. Usamos un GROUP BY por ID y no un DISTINCT porque no queremos eliminar los repetidos sino juntarlos. Tenemos que calcular el porcentaje en relación al total, por ende multiplicamos la cantidad de categorías por películas por 100 y lo dividimos en la cantidad de rentas o alquileres totales. A modo de solución alternativa, podemos invertir el recorrido del from con rental hasta llegar a category.

3)\_ Obtener el promedio de ganancia, la ganancia total y el total de alquileres por categoría de película.

**Resultado**: 16 tuplas.

**SELECT** category.name, **AVG** (payment.amount) **AS** “Promedio de ganancia”, **SUM**(payment.amount) **AS** “Ganancia total”, **COUNT**(rental.rental\_id) **AS** “Total de alquileres”

**FROM** category

**JOIN** film\_category **ON** category.category\_id = film\_category.category\_id

**JOIN** film **ON** film\_category.film\_id = film.film\_id

**JOIN** inventory **ON** film.film\_id = inventory.film\_id

**JOIN** rental **ON** inventory.inventory\_id = rental.inventory\_id

**JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** category.category\_id, category.name

\_ Primero determinamos las entidades, y vemos que incluye category, rental y payment, por lo que hacemos el recorrido de category hasta payment. Hacemos un AVG de amount para calcular el promedio de tal alquiler, y un SUM de amount que es la cantidad que se pagó por tal alquiler, y para el total de alquileres contamos los ID de cada renta. A modo de solución alternativa podemos arrancar en el from con payment y llegar hasta category.

4)\_ Obtener nombre y ganancia promedio de todas las películas cuya ganancia promedio supere el promedio de alguna categoría. Ordenar por promedio de mayor a menor.

**Resultado**: 510 tuplas.

**SELECT** film.title, **AVG** ( payment.amount ) **AS** “Ganancia\_promedio”

**FROM** film

**INNER JOIN** inventory **ON** film.film\_id = inventory.film\_id

**INNER JOIN** rental **ON** inventory.inventory\_id = rental.inventory\_id

**INNER JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** film.film\_id, film.title

**HAVING** Ganancia\_promedio > **ANY** (

**SELECT** **AVG**( payment.amount )

**FROM** film\_category

**INNER** **JOIN** inventory **ON** film\_category.film\_id = film.film\_id

**INNER JOIN** rental **ON** rental.inventory\_id = inventory.inevntory\_id

**INNER** **JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** film\_category.category\_id )

**ORDER BY** Ganancia\_promedio **DESC**

\_ Como primer paso identificamos las entidades, y vemos que tenemos que ir desde film a payment, pasando por inventory y rental. En este caso necesitamos el promedio de ganancia por película por eso el AVG a amount. Ahora debemos saber el promedio de ganancia por categoría y solamente quedarnos con aquellas películas que sean más grandes en promedio de ganancia que alguna categoría. Como usamos una función de agregación aplicada a un grupo, usamos un HAVING porque necesitamos agregar la condición de que el promedio sea mayor a algo, y como para calcular el promedio estamos agrupando, tenemos que meterla en el having. Siempre que es una condición del grupo va en el HAVING y si es una condición de los datos va en el WHERE. Usamos el ANY porque necesitamos que la ganancia promedio sea mayor a algún promedio de alguna categoría. Ahora en la sub consulta, para calcular el promedio de ganancia de cada categoría por alquiler, arrancamos desde film\_category hasta renta, no arrancamos desde category porque no usamos ningún dato de esa tabla solo el ID que esta a su vez en la pasarela.

5)\_ Obtener nombre y ganancia promedio de todas las películas cuya ganancia promedio supere el promedio de todas las categoría. Ordenar por promedio de mayor a menor.

**Resultado**: 405 tuplas.

**SELECT** film.title, **AVG** ( payment.amount ) **AS** “Ganancia\_promedio”

**FROM** film

**INNER JOIN** inventory **ON** film.film\_id = inventory.film\_id

**INNER JOIN** rental **ON** inventory.inventory\_id = rental.inventory\_id

**INNER JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** film.film\_id, film.title

**HAVING** Ganancia\_promedio > **ALL** (

**SELECT** **AVG**( payment.amount )

**FROM** film\_category

**INNER** **JOIN** inventory **ON** film\_category.film\_id = film.film\_id

**INNER JOIN** rental **ON** rental.inventory\_id = inventory.inevntory\_id

**INNER** **JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** film\_category.category\_id )

**ORDER BY** Ganancia\_promedio **DESC**

\_ Este ejercicio es igual al 4 solo que como pide que la ganancia promedio supere al promedio de todas las categorías, usamos el ALL.

6)\_ Obtener id, nombre y ganancia promedio de todas las películas cuya ganancia promedio supere el promedio de su categoría. Incluir también el nombre de la categoría a la cual pertenece la película y el promedio de dicha categoría como resultado. (Tipo parcial).

**Resultado**: 454 tuplas.

**SELECT** film.film\_id, film.title, **AVG** ( payment.amount ) **AS** ‘Promedio\_pelicula’, category.name, **AVG**(AVG\_category.promedio) **AS** ‘Promedio\_categoria’

**FROM** film

**INNER JOIN** inventory **ON** film.film\_id = inventory.film\_id

**INNER JOIN** rental **ON** inventory.inventory\_id = rental.inventory\_id

**INNER JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**INNER JOIN** film\_category **ON** film.film\_id = film\_category.film\_id

**INNER JOIN** category **ON** film\_category.category\_id = category.category\_id

**INNER JOIN** (

**SELECT** film\_category.category\_id, **AVG**( payment.amount ) **AS** ‘promedio’

**FROM** film\_category

**INNER** **JOIN** inventory **ON** film\_category.film\_id = film.film\_id

**INNER JOIN** rental **ON** rental.inventory\_id = inventory.inevntory\_id

**INNER** **JOIN** payment **ON** rental.rental\_id = payment.rental\_id

**GROUP BY** film\_category.category\_id

) AVG\_category **ON** film\_category.category\_id = AVG\_category.category\_id

**GROUP BY** film.film\_id, film.title, category.name

**HAVING** Promedio\_pelicula > Promedio\_categoria

\_ Primero vemos las entidades, para obtener el promedio de ganancia de las películas debemos ir desde film a payment pasando por inventory y rental. Hasta acá ya tenemos los datos de la película pero no de la categoría, por eso realizamos otros INNER JOINS desde film\_category a category. Necesitamos todos los amount de cada categoría y esto es lo que tenemos en la subconsulta del ejercicio anterior, por ende le hacemos un INNER JOIN donde a esta tabla le tenemos que asignar si o si un nombre, y la unimos con film\_category pero con el ID de películas.

7)\_ Obtener la cantidad total de películas que se encuentran disponibles en el inventario.

**Resultado**: 4398 películas.

**SELECT** **COUNT**( \* ) **AS** ‘Cantidad\_disponible’

**FROM** inventory

**WHERE** inventory.inventory\_id **NOT IN**(

**SELECT** rental.inventory\_id

**FROM** rental

**WHERE** rental.return\_date **IS** NULL )

\_ Cuando nos referimos a la cantidad de películas que haya en el inventario, es que se hayan devuelto o que no estén actualmente prestadas. Por lo tanto primero identificamos la entidad que es el inventario, luego comparamos que el ID de los inventarios no estén en la subconsulta, que son aquellos alquileres que tienen una fecha de retorno NULL por lo que en el ID de inventario son las que no están devueltas.

8)\_ Obtener el nombre de la categoría y la cantidad total de películas disponibles en inventario para cada una de ellas.

**Resultado**: 16 tuplas.

**SELECT** category.name, **COUNT**( \* ) **AS** ‘Cantidad\_disponible’

**FROM** inventory

**INNER JOIN** film **ON** inventory.film\_id = film.film\_id

**INNER JOIN** film\_category **ON** film.film\_id = film\_category.film\_id

**INNER JOIN** category **ON** film\_category.category\_id = category.category\_id

**WHERE** inventory.inventory\_id **NOT** **IN**(

**SELECT** rental.inventory\_id

**FROM** rental

**WHERE** rental.return\_date **IS** NULL )

**GROUP** **BY** categroy.category\_id, category.name

\_ Para denotar las entidades, vemos que necesitamos del inventario la cantidad de películas y el nombre de categoría, por lo que vamos de inventario a categoría. Y como dice películas disponibles aplicamos la lógica de subconsulta del ejercicio 7.

9)\_ Obtener el nombre de la categoría cuya cantidad total de películas disponibles supere los 300 ejemplares.

**Resultado**: 3 tuplas.

**SELECT** category.name, **COUNT**( \* ) **AS** ‘Cantidad\_disponible’

**FROM** inventory

**JOIN** film **ON** inventory.film\_id = film.film\_id

**JOIN** film\_category **ON** film.film\_id = film\_category.film\_id

**JOIN** category **ON** film\_category.category\_id = category.category\_id

**WHERE** inventory.inventory\_id **NOT** **IN**(

**SELECT** rental.inventory\_id

**FROM** rental

**WHERE** rental.return\_date **IS** NULL )

**GROUP** **BY** categroy.category\_id, category.name

**HAVING** Cantidad\_disponible > 300

\_ La solución es igual a la del ejercicio 8, solo que esta vez agregamos el HAVING porque la cantidad total de películas debe superar los 300 ejemplares.

10)\_ Obtener la cantidad total de películas que aún no se han devuelto por cada categoría.

**Resultado**: 16 tuplas.

**SELECT** category.name, **COUNT**( \* ) **AS** ‘Cantidad\_no\_disponible’

**FROM** inventory

**INNER JOIN** film **ON** inventory.film\_id = film.film\_id

**INNER JOIN** film\_category **ON** film\_category.film\_id = film.film\_id

**INNER JOIN** category **ON** category.category\_id = film.category\_id

**WHERE** inventory.inventory\_id **IN**(

**SELECT** rental.inventory\_id

**FROM** rental

**WHERE** rental.retun\_date **IS** NULL )

**GROUP BY** category.category\_id, category.name

\_ La solución como vemos es igual a la del ejercicio 8 solo que esta vez debemos tomar los ID de las tuplas del inventario de la subconsulta, que son los alquileres que todavía no se devolvieron.