Bases de Datos

Memoria de prácticas - Gestión de una base de datos de contenido multimedia (películas y series)

Grupo: Miércoles B
Hora: 17:00 - 19:00
Turno: Tardes, en todos los integrantes

Integrantes: Jorge Aznar López, Ángel Cañal Muniesa y Abel Chils Trabanco

NIPS, en orden relativo: 721556, 716205, 718997

ÍNDICE:

PARTE 3: OPTIMIZACIÓN, DISEÑO FÍSICO Y TRIGGERS

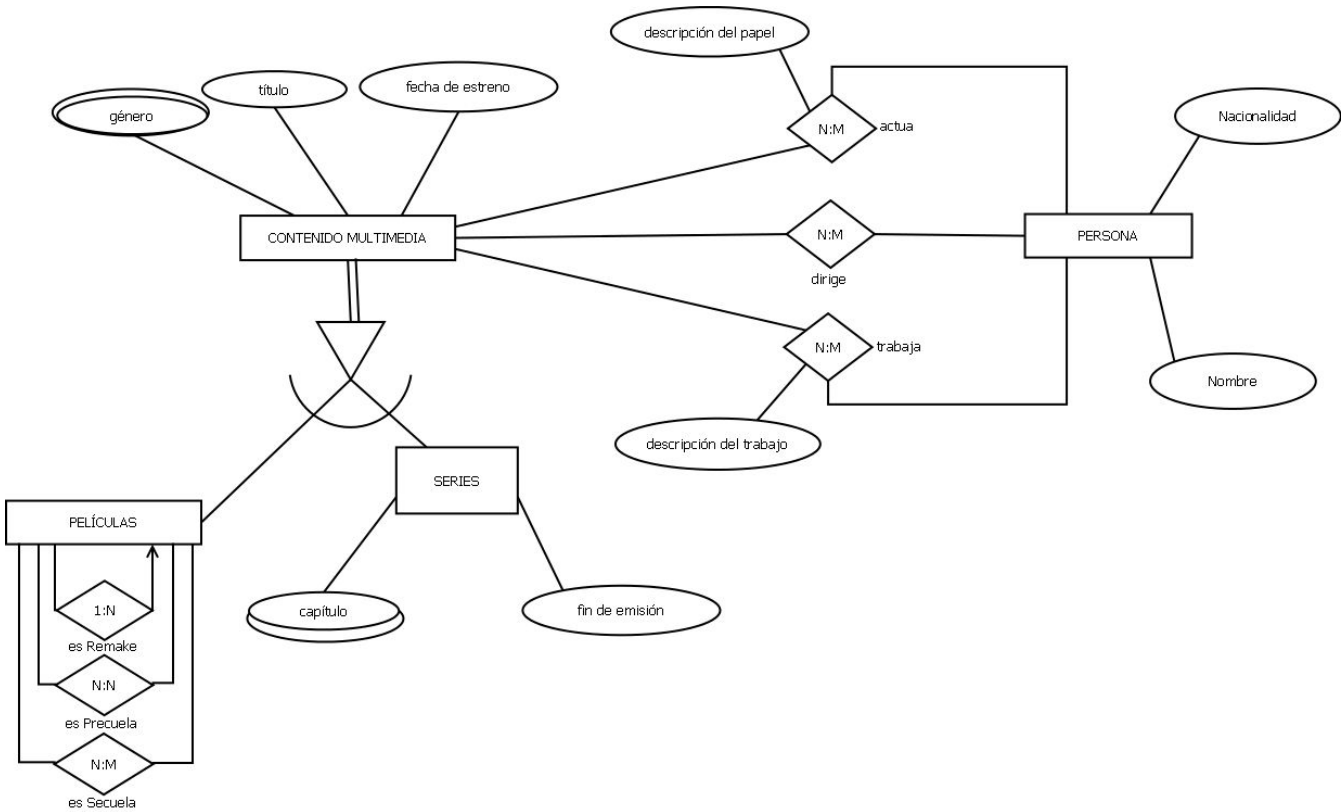
ESQUEMA ENTIDAD-RELACIÓN

A la hora de diseñar el esquema entidad-relación que hará referencia a nuestra base de datos se decidió que las películas y las series, que son un elemento fundamental de la base serán tratadas como una especialización disjunta con participación total en la entidad Contenido_multimedia, en la cual hemos almacenado los atributos Género, Fecha de Estreno y Título, que son aquellos que tanto películas como series comparten, pero a su vez una película no puede ser una serie, ni viceversa.

La creación de la relaciones dirige, trabaja y actúa que relacionan a una persona con la película o serie en la que desempeña una función surgen con la finalidad de poder distinguir la función de dicha persona, es decir, si es un director, actor o trabajador de la película referenciada en la relación, en caso de ser actor se añadirá el atributo papel, pues es fundamental saber qué papel desempeña cada actor en sus películas y si trabaja en ella se añadirá un atributo que describa brevemente su función en la película.

En la relaciones esRemake, esPrecuela y esSecuela existen dos ramas desde la entidad Película debido a que se debe diferenciar la película original del remake, la precuela o la secuela, pero no fue necesario etiquetar a que rama pertenece a cada una dado que la cardinalidad de la relación nos permite saber qué película es la original, ya que solo puede haber una película original pero pueden haber un número 'N' de remakes, precuelas o secuelas (cardinalidad 1:N).

DECISIONES DE DISEÑO



Las principal dificultad con la que encontrada en el proceso de diseño fue la elección del método mediante el cual se reflejase la especialización del contenido multimedia en el modelo relacional, pues existen tres posibles alternativas a contemplar :

1. Incluir toda la información extra de las especializaciones en la relación principal del elemento general (contenido multimedia).

   Consideramos esta opción inviable por la redundancia de datos que crea en la base.

2. Reflejar toda la información de la entidad general en las especialidades eliminando la principal al pasar a modelo relacional.

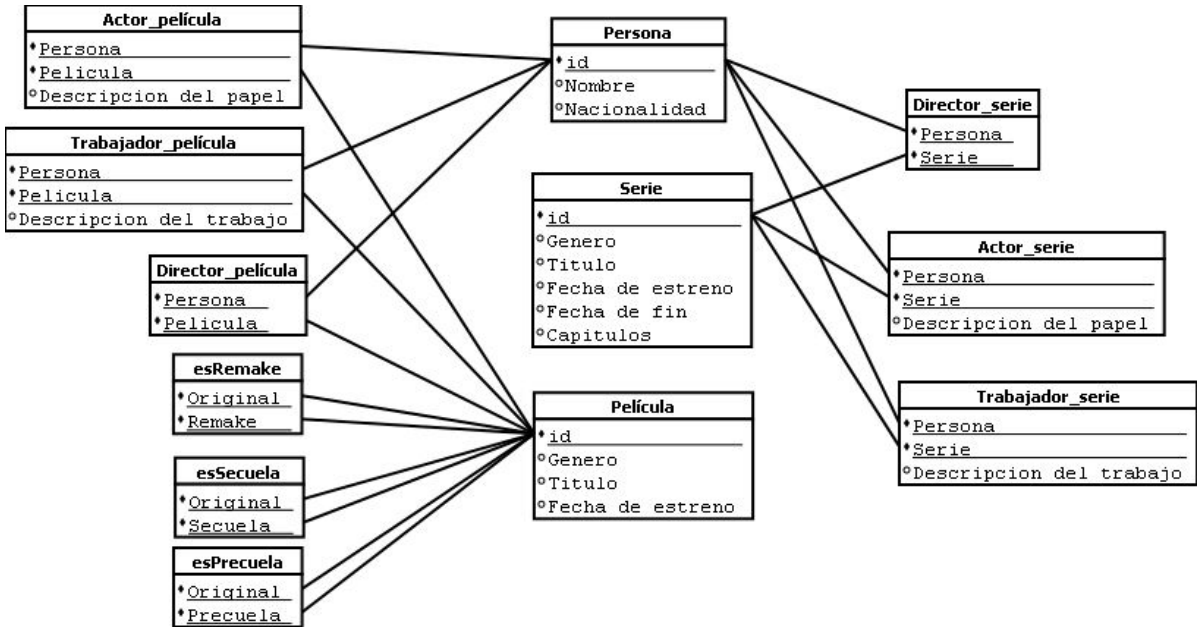
   Elegimos esta opción dado que es la que mejor se adapta al diseño de nuestra base.

3. Crear una relación para cada entidad que participa en la especialización, es decir, tanto como para la entidad general como para sus especializaciones y referenciar las claves de las especializaciones a la relación de la entidad general.

   A pesar de que esta opción también es viable elegimos la segunda pues es la que mejor se adapta a nuestro diseño.

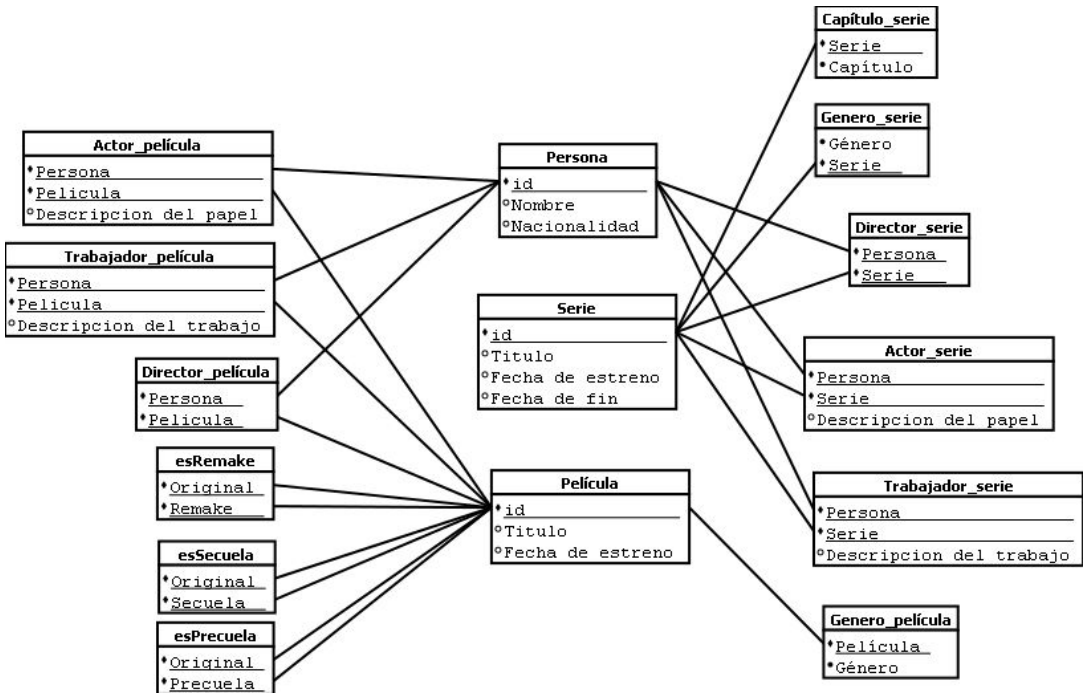
MODELO RELACIONAL Y PROCESO DE NORMALIZACIÓN

Modelo pre-normalizado



1ª Forma normal: Debido a la existencia de los atributos multi-evaluados "Capitulo" y "Genero" en la entidad Serie y al atributo "Genero" en la entidad Pelicula se crearon las tablas Capitulo_serie, Genero_serie con clave primaria compuesta por Serie que referencia a la tabla Serie, y Capitulo/Genero que son propias de la relación y la tabla Genero_pelicula con clave primaria compuesta por Pelicula que referencia a la tabla Pelicula, y Genero que es un atributo propio de la relación.

Modelo en primera forma normal



2ª Forma normal: El modelo relacional se hallaba en segunda forma normal cuando fue analizado, dado que para todo atributo clave A no existía dependencia funcional con cualquier otro atributo también clave B, es decir, ningún atributo A clave depende funcionalmente de parte de la clave.

Heurística: No es necesario analizar las relaciones con clave única, sólo aquellas con clave compuesta (varias claves candidata).

3ª Forma normal: El modelo relacional se hallaba en tercera forma normal cuando fue analizado, dado que para todo atributo clave A no existía dependencia funcional con cualquier otro atributo, o conjunto de atributos, no clave B, es decir, ningún atributo A clave depende funcionalmente de otro atributo de la relación o conjunto de atributos no clave.

Heurística: No es necesario analizar las relaciones con clave única, sólo aquellas con uno o varios atributos no clave.

4ª Formal normal: El modelo relacional se hallaba en cuarta formal normal cuando fue analizado, dado que no existen dependencias multi-evaluadas entre cualquier conjunto de atributos A y otro conjunto de atributos B, es decir, los valores de B sí que dependen de los valores que tome A en un caso determinado.

RESTRICCIONES

Las principales restricciones en los atributos serían las imposibilidades de que la fecha de estreno de una serie fuera posterior a su fecha de fin de emisión y que una película sea secuela y precuela al mismo tiempo de la misma película original, estos casos podrían generar valores erróneos en los atributos de la base.

Población de la base de datos

Para obtener los datos necesarios para poblar la base de datos se decidió realizar consultas sobre la base de datos miniIMDB mediante las cuales se obtuvieran tuplas de datos que se adaptaran a nuestro esquema de tablas.
Estas consultas se encuentran explicadas en la carpeta Poblacion de la base con el nombre Consultas_miniIMDB.sql.
Una vez obtenido los resultados de estas consultas, utilizando el IDE DataGrip(JetBrains) se exportaron a nuestra base de datos.
Y por último a partir de las columnas insertadas en nuestra base de datos generamos los ficheros de insert que se encuentran en la carpeta Datos dentro de la carpeta Poblacion de la base.

Estadísticas de las tablas

El tamaño de las tablas se ha calculado buscando en la tabla user_extents.

Tabla	Tamaño (Kbytes)	Filas
ACTOR_PELICULA	968	51736
CAPITULOS_SERIE	40	157
DIRECTOR_PELICULA	144	4098
ES_PRECUELA	40	0
ES_REMAKE	40	3
ES_SECUELA	40	55
PELICULA	272	3802
PERSONA	1488	38978
SERIE	160	1715

Se puede ver en las tablas que ocupan 40960 bytes que Oracle asigna un tamaño inicial para todas las tablas, que va incrementando según va creciendo.

CONSULTAS

Diseño de las consultas

Para resolver las consultas se decidió valerse de la creación de vistas, en las cuales cada una resuelve una parte del problema y apoyándose en las anteriores se logra resolver la consulta pedida.

Explicación de las consultas

Consulta obligatoria nº1:

Descripción: Porcentaje de películas con entre 5 y 10 actores o actrices.

Script SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERY_1.sql.

Álgebra relacional:

$\pi_{(peliculas\_5\_10\_actores/total\_peliculas)*100}$ - ($f_{count(*)\ as\ peliculas\_5\_10\_actores}$ - $\sigma_{num\_actores >=5\ AND\ num\_actores <=10}$ - $_{(pelicula)}f_{count(*)\ as\ num\_actores}$ - ACTOR_PELICULA), ($f_{count(*)\ as\ peliculas\_5\_10\_actores}$ - PELICULAS)

Explicación: Para resolver esta consulta primero obtenemos los identificadores de las películas con entre 5 y 10 actores, mediante la vista PELICULAS_5_10_ACTORES. Luego contamos el número de filas de la view anterior mediante count_peliculas_5_10_actores, obteniendo así el número de películas con entre 5 y 10 actores, y el número de filas de la tabla película con count_total_peliculas obteniendo el número total de películas . Por último hacemos el producto cartesiano de las dos vistas anteriores por lo que obtenemos en una fila el total de películas y el número de películas con entre 5 y 10 actores, por lo que solo falta dividir este último entre el número total de películas y multiplicarlo por 100 para obtener la solución.

Consulta obligatoria nº2:

Descripción: Directores que han dirigido 3 o más películas el mismo año ordenados de más a menos películas dirigidas en total y el año en cuestión.

Script SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERY_2.sql.

Álgebra relacional:

$$\Pi_{p.fecha\_de\_estreno,\ d.nombre,\ peliculas}(\zeta_{ORDER\ BY\ peliculas\ DESC}(\sigma_{peliculas >= 3}((p.fecha\_de\_estreno,\ d.persona)\ \zeta_{COUNT(*)\ peliculas}(\bowtie_{d.pelicula = p.id}(dirige\ d,\ peliculas\ p)))))$$

Explicación: Primero, se selecciona el id de los directores, qué año y el número de películas que ha estrenado en ese año y si ese número es mayor a 3, finalmente se muestra el nombre del director, qué año y el número de películas que ha estrenado ese año.

Consulta obligatoria nº3:

Descripción: Obtiene los directores para los cuales la última obra en la que han participado ha sido como actor/actriz.

Script SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERY_3.sql.

Álgebra relacional:

PELICULAS_DIRECTOR

(π director, id, fecha_de_estreno) -

(σ director_pelicula.pelicula = pelicula.id) -

(DIRECTOR_PELICULA, PELICULA)

DIR_ULTIMAS_PELICULAS

$(_{(PELICULAS\_DIRECTOR.director)}, f_{MAX(fecha\_de\_estreno)\ AS\ ultima\_pelicula}) -$

(PELICULAS_DIRECTOR)

VIEW DIRECTOR_Y_PELICULA

$(\pi_{id,\ DIR\_ULTIMAS\_PELICULAS.director}) -$

$(\sigma_{DIR\_ULTIMAS\_PELICULAS.ultima\_pelicula\ =\ PELICULAS\_DIRECTOR.FECHA\_DE\_ESTRENO\ AND\ DIR\_ULTIMAS\_PELICULAS.director\ =\ PELICULAS\_DIRECTOR.director}) -$

(PELICULAS_DIRECTOR ,DIR_ULTIMAS_PELICULAS)

ACTORES_DIRECTORES

$(\pi_{ACTOR\_PELICULA.persona\ AS\ actor,pelicula}) -$

$(\sigma_{actor\_pelicula.pelicula\ =\ DIRECTOR\_Y\_PELICULA.id\ AND\ DIRECTOR\_Y\_PELICULA.director\ =\ ACTOR\_PELICULA.PERSONA}) -$

(ACTOR_PELICULA,DIRECTOR_Y_PELICULA)

NOMBRE_DIRECTORES (RESULTADO)

$(\pi_{NOMBRE}) -$

$(\sigma_{PERSONA.ID\ =\ ACTORES\_DIRECTORES.actor}) -$

(ACTORES_DIRECTORES, PERSONA)

Explicación: En primer lugar se obtiene de cada director las películas que ha dirigido así como su fecha de estreno, necesaria para posteriormente saber cuál es su última película en la vista *PELICULAS_DIRECTOR*, posteriormente a partir de esta vista se obtiene entre las películas de cada director la última dirigida por este, se asocia a cada director el id de esta película mediante la vista *DIRECTOR_Y_PELICULA*, se obtienen a parte los actores que participaron en esta película y si alguno coincide con el director se obtiene su identificador, mediante el cual posteriormente a través de la vista *NOMBRE_DIRECTORES* obtendremos el nombre real de cada

director que participó como actor en su última película.

Primera consulta personal:

Descripción: Obtiene los actores y actrices que han participado en al menos la mitad de las películas de la saga a la que pertenece la película "La maldición de la bestia".

Consulta en lenguaje SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERIE_P_1.sql.

Consulta en álgebra relacional:

π_{NOMBRE} -($\bowtie_{PERSONA.ID= PERSONA}$ - ($\pi_{PERSONA}$ -$_{(PERSONA)}$f $_{count(*) >= (f\ count(*)-PELICULA)/2}$ -($\pi_{PERSONA}$ -($\sigma_{PELICULA\ IN}$ (($\pi_{PRECUELA\ AS\ PELICULAS\_SAGA}$ - $\bowtie_{ES\_PRECUELA.ORIGINAL = PELICULA.ID\ AND\ PELICULA.TITULO = 'La\ maldición\ de\ la\ bestia'}$ - ES_PRECUELA , PELICULA) U ($\pi_{SECUELA\ AS\ PELICULAS\_SAGA}$ - $\bowtie_{ES\_SECUELA.ORIGINAL = PELICULA.ID\ AND\ PELICULA.TITULO = 'La\ maldición\ de\ la\ bestia'}$ - ES_SECUELA, PELICULA) U ($\pi_{ID\ AS\ PELICULAS\_SAGA}$ - $\sigma_{TITULO = 'La\ maldición\ de\ la\ bestia'}$ – PELICULA)))))), PERSONA)

Explicación: Para resolver esta consulta primero se obtienen las películas de la saga “La maldición de la bestia”, lo que implica obtener la unión de las precuelas y secuelas de la película junto con esta. Una vez obtenido esto se seleccionan los actores que participan en estas películas. Por último de estos actores se cuenta el número de apariciones en las películas de esta saga y se seleccionan los que aparecen en al menos la mitad.

Segunda consulta personal:

Descripción: La(s) serie(s) con más capítulos y su tiempo de emisión.

Consulta en lenguaje SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERIE_P_2.sql.

Consulta en álgebra relacional:

$$CAPS = (serie)\zeta_{COUNT(*)\ serieCaps}(capitulos\_serie\ c1)$$
$$MAX\_CAPS = \Pi_{totalCaps}(\zeta_{MAX(serieCaps)\ totalCaps}(CAPS))$$
$$\Pi_{duracion,\ s.titulo}(\rho_{s.fin\_de\_emision - s.fecha\_de\_estreno}(\bowtie_{s.id = c1.serie}(serie\ s,\ (\sigma_{seriesCaps=MAX\_CAPS}(CAPS)))))$$

Explicación: Primero, la view seriesCaps (CAPS) selecciona cuantos capítulos tiene cada serie, después, de esos capítulos la view maxCaps (MAX_CAPS) selecciona cuál ha sido el mayor número de capítulos que han tenido las series.
Finalmente la consulta muestra la duración y el título de la(s) serie(s) con más capítulos.

Tercera consulta personal:

Descripción: Obtiene la película con la saga más larga, es decir, aquella con más secuelas/precuelas, el nombre de las películas de la saga y el número de películas que la componen.

Consulta en lenguaje SQL: La consulta en lenguaje SQL se encuentra dentro de la carpeta QUERIES con el nombre QUERIE_P_3.sql.

Consulta en álgebra relacional:

PELICULAS_SAGA

($\pi_{\text{PRECUELA AS PELICULAS\_SAGA,ORIGINAL}}$) -

(ES_PRECUELA)

U

($\pi_{\text{SECUELA AS PELICULAS\_SAGA,ORIGINAL}}$) -

(ES_SECUELA)

NUM_PELIS_SAGA

($_{\text{(ORIGINAL),}}$ $f_{\text{COUNT(PELICULAS\_SAGA) AS NPELIS}}$) –

(PELICULAS_SAGA)

MAX_PELIS_SAGA

($f_{\text{MAX(NPELIS) AS NUMERO\_PELICULAS}}$)-

(NUM_PELIS_SAGA)

PELI_MAX_SAGA

($\pi_{\text{original, NUMERO\_PELICULAS}}$) –

($\sigma_{\text{NUM\_PELIS\_SAGA.NPELIS = MAX\_PELIS\_SAGA.NUMERO\_PELICULAS}}$) -

(NUM_PELIS_SAGA, MAX_PELIS_SAGA)

SAGA_TOTAL

($\pi_{\text{PELI\_MAX\_SAGA.original, NUMERO\_PELICULAS, PELICULAS\_SAGA}}$) –

(σ $_{\text{PELI\_MAX\_SAGA.original = peliculas\_saga.ORIGINAL}}$) –

(PELI_MAX_SAGA, PELICULAS_SAGA)

SAGA (RESULTADO)

(π $_{\text{original AS original, titulo AS titulo, NUMERO\_PELICULAS+1 AS LONGITUD\_SAGA}}$) -

(σ $_{\text{pelicula.id = PELICULAS\_SAGA}}$) –

(SAGA_TOTAL, PELICULA)

U

(π $_{\text{original AS original, titulo AS titulo, NUMERO\_PELICULAS+1 AS LONGITUD\_SAGA}}$) -

(σ $_{\text{pelicula.id = original}}$) –

(SAGA_TOTAL,PELICULA)

Explicación: Para resolver la última consulta personal primero se obtienen todas las precuelas y secuelas de cada película mediante la vista *PELICULAS_SAGA*, se cuentan las precuelas y secuelas para saber cuantas películas componen la saga y se le asocia la cuenta a la película original, posteriormente a través de la vista *PELIS_MAX_SAGA* se obtiene aquella(s) película(s) que más precuelas y secuelas tengan, se obtienen sus identificadores y finalmente a través de la vista *SAGA* se le asocia a cada identificador el título de la película, conservando también en columnas tanto el id de la película original, para evitar confundir sagas en caso de que alguna coincidiese en título y número de películas que la componen, como el ya nombrado número de películas que componen la saga.

TRIGGERS

Trigger nº1

Descripción: El primer trigger que se ha creado se encarga de crear una tabla y guardar en ella el número de actores de cada película, de esta forma en la primera consulta se evita calcular el número de actores que tiene cada película.

Script SQL: El script SQL correspondiente al trigger se encuentra en la carpeta TRIGGERS con el nombre TRIGGER_1.sql y la consulta que lo utiliza se encuentra en la carpeta QUERIES con el nombre QUERY_1_opt.sql

Mejoras en el rendimiento: Después de ejecutar este trigger se obtiene un tercio menos de llamadas recursivas, el número de lecturas a memoria cache ha pasado de 155 a 43, en cambio el número de lecturas físicas pasa de 0 a 5 debido a que esta otra tabla ha de estar muy separada en disco.

Trigger nº2

En este caso, el trigger se encargará de optimizar la consulta 2, y guarda en una tabla el número de películas que ha dirigido en cada año, de esta manera esa consulta reduce notablemente su tiempo de ejecución.

El script SQL correspondiente al trigger se encuentra en la carpeta TRIGGERS con el nombre TRIGGER_2.sql y la consulta que lo utiliza se encuentra en la carpeta QUERIES con el nombre QUERY_2.sql

Tras el uso del trigger, se eliminan todas las llamadas recursivas y las lecturas de disco, de 16 ordenaciones en memoria se pasa a 1 sóla. Por último, los “consistent gets” se reducen un 97.59%, de 8486 a 204. El

tiempo de la consulta se reduce de 970 ms a 830 ms, y podría reducirse aún más si se precalculase el join y en vez del id del director se pusiese su nombre.

Se puede concluir que este trigger reduce ligeramente el tiempo de la consulta, pero sí que reduce el uso de recursos del sistema.

Trigger nº3

Este último trigger tiene como finalidad la creación de una tabla denominada “act_directores” que almacenará los directores que también participaron como actores en una película así como dicha película con la finalidad de optimizar la tercera consulta obligatoria.

El código del trigger se encuentra ubicado en la carpeta TRIGGERS con el nombre TRIGGER_3.sql y la consulta que lo usa se encuentra en la carpeta QUERIES con el nombre QUERY_3_opt.sql

La optimización principal que genera este trigger en la consulta es la disminución de 20387 gets consistentes a 12634, y que pasa de generar un registro de rehacer los logs de 128 bytes a 0 bytes.

Esto genera un aumento del 12% en la eficiencia del trigger, todo ello se puede ver con más detalle junto con la traza en los ficheros stat_3.txt y stat_3_trigger.txt que comparan la salida de la traza con y sin triggers en la carpeta STATS.

ÍNDICES

Índice nº1

Descripción: Sobre la tabla creada en el trigger nº1 para almacenar el número de actores de cada película se crea un índice sobre este número de actores, con el fin de aumentar el rendimiento de la cláusula where en la que se buscan las películas con entre 5 y 10 actores.

Mejoras en el rendimiento: Con respecto a los datos estadísticos obtenidos después del trigger se reduce el número de lecturas a caché de 43 a 32, esto es debido a que al estar indexado, con menos accesos se pueden obtener las películas que se buscan. Por otro lado también se reduce el número de lecturas físicas , que pasa de 5 a 2. Debido a que es el mismo script, el número de llamadas recursivas no se ve alterado.

Índice nº 2

Sobre la tabla del trigger anterior se añade un índice en orden descendente para la columna películas, para optimizar la búsqueda de directores que estrenan más de 3 películas en 1 año.

Mejora el rendimiento de la consulta, reduciendo el tiempo de 830 milisegundos a 650. En total reducimos el tiempo de consulta en un 35%.