



UNIVERSIDAD DE LIMA
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA DE SISTEMAS
ASIGNATURA : INGENIERIA DE DATOS.
SECCIÓN : 601
FECHA : 30/10/2013
TIEMPO : 30 MINUTOS

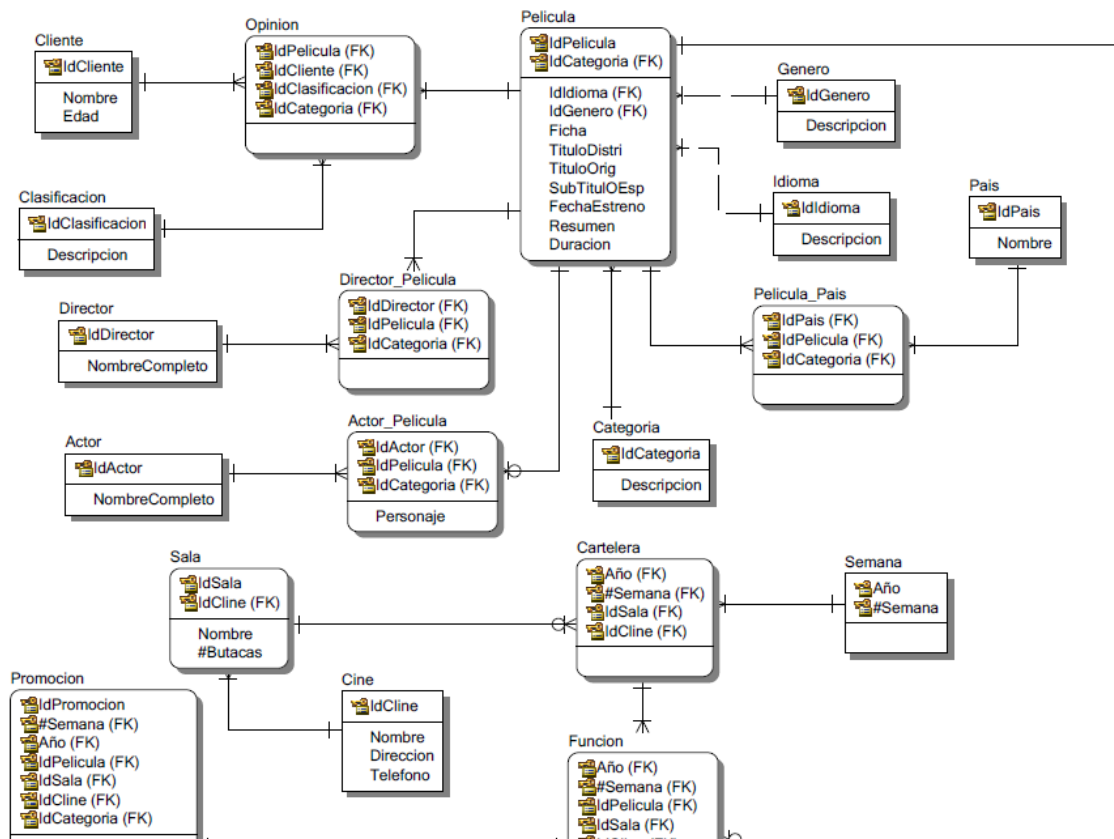


NOTA

Practica Calificada #3

Código: _____ **Nombre**

Modelo de Datos



A partir del modelo de datos, construir las siguientes consultas (2 ptos c/u):

1.- Mostrar la cantidad de películas que fueron estrenadas en el año 2012. Tenga en cuenta que el campo FechaEstreno es un tipo de dato DATE. Paralelizar la consulta con 4 hilos de ejecución.

SQL>

```
select /*+ PARALLEL(p,4) */ count(*)
from pelicula p
where to_char(fechaestreno,'YYYY')='2012';
```

2.- Mostrar un reporte donde se visualice la cantidad de películas por categoría que ha participado el actor Charlton Heston. El reporte debe mostrar la descripción de la categoría y el total de veces que el actor ha participado.

SQL>

```
select c.descripcion, count(*)
from pelicula p join categoria c on (p.idcategoria = c.idcategoria)
join actor_pelicula ap on (ap.idpelicula = p.idpelicula)
join actor a on (a.idactor = ap.idactor)
where a.nombreCompleto = 'Charlton Heston'
group by c.descripcion;
```

3.- Indique cual es la semana del año 2012 que ha tenido más carteleras presentadas.

SQL>

```
select #semana
from(
select #semana, total
from(
select #semana, count(*) as total
from cartelera
where año=2012
group by #semana
)
order by total desc
)
where rownum<=1;
```

4.- Por cada película, liste el título de distribución y la cantidad de directores que han participado en ella.

SQL>

```
select tituloDistri, count(*)
from pelicula p join director_pelicula dp
on p.idpelicula = dp.idpelicula
group by tituloDistri;
```

5.- Liste por cada categoría, la cantidad de películas con una duración mayor a 90 minutos.

SQL>

```
select c.descripcion, count(*)
from categoria c join pelicula p
on c.idcategoria = p.idcategoria
where p.duracion >= 90
group by c.descripcion;
```

6.- Liste las categorías que no tienen alguna película asociada.

SQL>

```
select descripcion
from categoria
where idcategoria not in (select idcategoria from pelicula);
```

7.- Liste el NombreCompleto de cada actor y el personaje que ha tenido en la película cuyo Título Original es: Ben-Hur.

SQL>

```
select a.nombreCompleto, personaje
from pelicula p join actor_pelicula ap
on p.idpelicula = ap.idpelicula
join actor a on a.idactor = ap.idactor
where p.TituloOrig = 'Ben-Hur';
```

8.- Liste el Título Original de cada película del Género 'TERROR' que haya tenido más de 5 clasificaciones con el valor de EXCELENTE.

SQL>

```
select p.TituloOrig, count(*)
from pelicula p join genero g on p.idgenero = g.idgenero
join opinion o on o.idpelicula = p.idpelicula
join clasificacion c on c.idclasificacion = o.idclasificacion
where g.descripcion = 'TERROR' and c.descripcion = 'EXCELENTE'
group by p.TituloOrig
having count(*)>5;
```

9.- Liste las películas (Título Original) que están disponibles para el público el día 30 de Octubre entre las 18:00 h y las 22:00 h.

SQL>

```
select p.tituloorig
from pelicula p join funcion f on p.idpelicula = f.idpelicula
where to_char(dia,'YYYYMMDD')='20131030' and horaInicio=18 and horaFinal=22;
```

10.- Basado en la siguiente consulta:

```
select historicoj_.ID_, historicoj_.PROCINST_ as "PROCINST2_169_",
historicoj_.CLASS_ as "CLASS3_169_", historicoj_.VERSION_ as "VERSION4_169_",
historicoj_.NAME_ as "NAME5_169_",
historicoj_.DESCRIPTION_ as "DESCRIPT6_169_", historicoj_.ACTORID_ as "ACTORID7_169_",
historicoj_.CREATE_ as "CREATE8_169_", historicoj_.START_ as "START9_169_",
historicoj_.END_ as "END10_169_", historicoj_.DUEDATE_ as "DUEDATE11_169_",
historicoj_.PRIORITY_ as "PRIORITY12_169_", historicoj_.ISCANCELLED_ as "ISCANCE13_169_",
historicoj_.ISSUSPENDED_ as "ISSUSPE14_169_", historicoj_.ISOPEN_ as "ISOPEN15_169_",
historicoj_.ISSIGNALLING_ as "ISSIGNA16_169_", historicoj_.ISBLOCKING_ as
"ISBLOCKING17_169_",
historicoj_.TASK_ as "TASK18_169_", historicoj_.TOKEN_ as "TOKEN19_169_",
historicoj_.SWIMLANINSTANCE_ as "SWIMLAN20_169_", historicoj_.TASKMGMTINSTANCE_ as
"TASKMGMT21_169_"
from BPMPORTA.HISTORICO_JBPM_TASKINSTANCE historicoj_ where historicoj_.ID_ = 2096;
```

Se tiene el siguiente plan de ejecución:

```
SQL> select * from table(dbms_xplan.display);

PLAN_TABLE_OUTPUT
-----
Plan hash value: 2686507091

-----
| Id | Operation                | Name                                | Rows | Bytes | Cost (%CPU)| Time     |
-----|-----|-----|-----|-----|-----|-----|
| 0  | SELECT STATEMENT         |                                     |      1 |      69 | 8293  (1)| 00:01:40 |
|* 1  | TABLE ACCESS FULL       | HISTORICO_JBPM_TASKINSTANCE        |      1 |      69 | 8293  (1)| 00:01:40 |
-----

Predicate Information (identified by operation id):
-----

PLAN_TABLE_OUTPUT
-----

      1 - filter("HISTORICOJ_".ID_="TO_NUMBER(:1))

13 rows selected.
```

Implemente algún mecanismo visto en clase con la finalidad de mejorar el tiempo de respuesta de la consulta.

```
SQL> create index IDX_BPMPORTA.HISTORICO_JBPM_TASKINSTANCE on
BPMPORTA.HISTORICO_JBPM_TASKINSTANCE(historicoj_.ID);
```