Examen1:

A) Escribe las instrucciones en SQL para la creación de la tabla "Entrada" y la inserción de una tupla en dicha tabla. (fecha es de tipo date, hora_ini de tipo numérico, se almacena en formato 24 h. o sea, de 0h a 23h y solo se permiten valores enteros desde las 10h a la 1h de la madrugada, sala# es entero de 0 a 19, fila es una letra del alfabeto y columna un entero de 1 a 99).

```
create table Entrada (
    fecha date constraint fecha_no_nulo not null,
    hora_ini int constraint hora_ini_no_nulo not null constraint hora_ini_entre_10_y_1
        check ((hora_ini >= 10 and hora_ini <= 23) or (hora_ini >= 0 and hora_ini <= 1)),
    sala# int constraint sala#_no_nulo not null constraint sala#_entre_0_y_19
        check( sala# >= 0 and sala# <= 19),
    fila char(1) constraint fila_no_nulo not null constraint fila_letra check( fila in ('[A-Z]')),
    columna int constraint columna_no_nulo not null constraint columna_entre_1_y_99
        check( columna >= 1 and columna <= 99),
    constraint clave_primaria primary key ( fecha, hora_ini, sala#, fila, columna),
    constraint clave_externa1 foreing key ( fecha, hora_ini, sala#) referenes Proyección ( fecha, hora_ini, sala#),
    constraint clave_externa2 foreing key ( sala#, fila, columna) referenes Asiento ( sala#, fila, columna)
);

invert into Entrada ( fecha, hora_ini, sala#, fila, columna)
    values( TO_DATE ('22/06/2020', 'dd/mm/yyyy'), 22, 5, 'C', 50);
```

(B) Realizar las siguientes consultas:

a) Proyecciones para las que no se han vendido ninguna entrada.

*AR:

$$\Pi * (\text{Proyección} \bowtie (\Pi_{\text{fecha, hora\_ini, sala\#}}(\text{Proyección}) - \Pi_{\text{fecha, hora\_ini, sala\#}}(\text{Entrada})))$$

*SQL:

```
select * from Proyecciones where in(
  select fecha, hora_ini, sala# from Proyecciones
  menos
  select fecha, hora_ini, sala# from Entrada
);
```

b) Muestra el título de las películas de las que solo existe una copia

*AR:

$$\rho(\text{Copia}) = \text{Cp1}$$
$$\rho(\text{Copia}) = \text{Cp2}$$

$$\Pi_{\text{titulo, P\#}}(\text{Película}) - \Pi_{\text{P\#}}(\sigma_{(\text{CP1.cop\#} != \text{CP2.cop\#} \wedge \text{CP1.P\#} = \text{CP2.P\#})}(\text{Cp1} \times \text{Cp2}))$$

*SQL:

```
select titulo, P# from Película
menos
select P# from copia cp1, copia cp2 where cp1.cop# != cp2.cop# and cp1.p# = cp2.p# ;
```

c) Muestra el título de las películas que se han proyectado en todas las salas

* AR:

$$\Pi_{titulo}(Peliculas \bowtie (\Pi_{p\#, sala\#}(Proyección) \div \Pi_{sala\#}(Sala)))$$

* SQL:

```
select titulo from Peliculas where not exists (
    select sala# from Sala
    minus
    select sala# from Proyección where Película.p# = Proyección.p# );
```

(c) Crea una vista que muestre la cantidad de entradas vendidas por cada proyección que haya vendido más de 40 entradas

```
create view entradas_vendidas as
select cp#, p#, fecha, hora_ini, sala#, count(*) from Proyección, Entrada
where Proyección.fecha = Entrada.fecha and Proyección.hora_ini = Entrada.hora_ini and Proyección.sala# = Entrada.sala#
group by fecha, hora_ini, sala# having count(*) > 40 ;
```

ó

```
create view entradas_vendidas as
select fecha, hora_ini, sala#, count(*) from Entradas
group by fecha, hora_ini, sala#
having count(*) > 40 ;
```

Examen2:

(A) Escribir las sentencias SQL para crear tablas Copia y Prestado teniendo en cuenta que todos los campos son alfanuméricos, salvo ficha que es de tipo date. Y que el año 2012 la biblioteca estuvo cerrada por reformas, por lo que debe evitarse que se introduzcan prestamos durante ese año.

```
create table Copia (
    cop# varchar2(8) constraint copia#_no_nulo not null,
    L# constraint clave_externa_libro
        references libro(L#),
    constraint cop#_L#_cp  primary key (cop#, L#)
);
create table Prestado (
    cop# varchar2(8) constraint cop#_no_nulo not null,
    L# varchar2(8) constraint L#_no_nulo not null,
    ficha date constraint ficha_no_nulo  constraint no_año_2012
        check ((ficha < TO_DATE('01/01/2012', 'dd/mm/yyyy')) or ( ficha > TO_DATE('31/12/2012',
                                                                  'dd/mm/yyyy'))),
    DNI constraint clave_externa_usuario
        references usuario (DNI),
    constraint clave_primaria primary key (cop#, L#, ficha),
    constraint clave_candidata unique (ficha, DNI)
);
```

(B) Realizar las siguientes consultas:

a) Encontrar el nombre de los usuarios que han tomado prestados todos los libros de la biblioteca

*AR:

$$\Pi_{nombre}\left(Usuario \bowtie \left(\Pi_{DNI,\,L\#}(Prestado) \div \Pi_{L\#}(Libro)\right)\right)$$

*SQL: (C)

```
select nombre from Usuario where not exists (
    select L# from Libro
    minus
    select L# from Prestado where Usuario.DNI = Prestado.DNI
);
```

b) Mostrar el título del primer libro que se prestó. (= préstamo más antiguo)

*AR: $\rho(Prestado) = Pres$

$$\Pi_{titulo}\left(Libro \bowtie \left(\Pi_{L\#,\,Prestado.fecha} - \Pi_{Pres.fecha}\left(\sigma_{Pres.fecha > Prestado.fecha}(Pres) \times Prestado\right)\right)\right)$$

4) Crear una vista que muestre: DNI de usuario, título de libros que ha tomado prestado y cuántas veces cada uno de ellos, para aquellos usuarios y títulos de libros prestados más de una vez, ordenadas por DNI y títulos.

```
create view prestamos as
    select DNI, titulo, count(*) from Prestado, Libro
    where Prestado.L# = Libro.L#
    group by DNI, titulo
    having count(*) > 1
    order by DNI, titulo;
```