

[Área personal](#)  [Mis cursos](#)  [bdI-2021](#)  [Evaluaciones](#)  [Evaluación Parcial 2](#)

**Comenzado el** miércoles, 3 de noviembre de 2021, 10:10

**Estado** Finalizado

**Finalizado en** miércoles, 3 de noviembre de 2021, 11:40

**Tiempo** 1 hora 30 minutos

empleado

Pregunta 1

Finalizado

Puntúa como 1,00

## Ejercicio 1

Considere que en la tabla Video del esquema de Películas ([link](#) esquema) se ha agregado un atributo *dif\_distribuidores*, para registrar la cantidad de *distribuidores diferentes* de los que cada video ha recibido entregas.

Justifique cuáles son los eventos a controlar para mantener tal atributo automáticamente actualizado ante operaciones realizadas sobre la BD, escriba en PostgreSQL los encabezados de los triggers necesarios e incluya la implementación completa de la función requerida para el caso de insert.

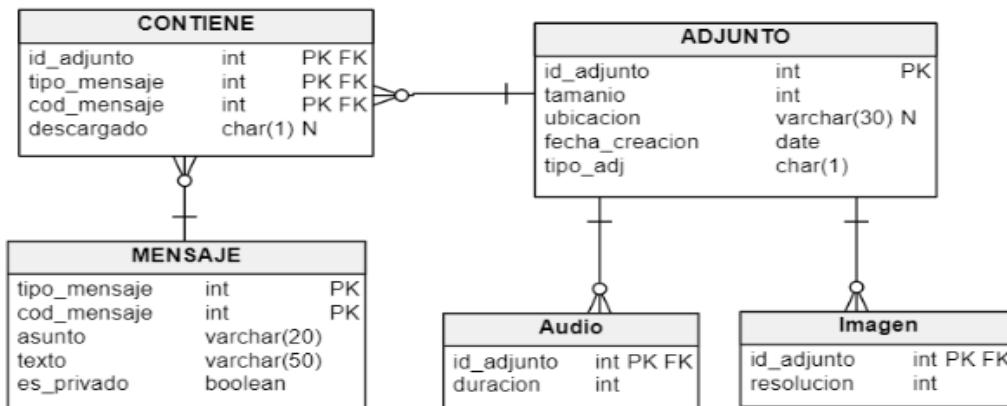
Pregunta 2

Finalizado

Puntúa como 1,00

## Ejercicio 2

Considere que el esquema ([script](#)) corresponde a un sistema de mensajería donde cada mensaje puede contener varios archivos adjuntos de tipo audio (A) o imagen (I). Cada mensaje incluye el asunto y texto del mismo, y si es privado o no.



a) Cree las siguientes vistas en PostgreSQL. Tenga en cuenta de construirlas de modo que, siempre que sea posible, resulten automáticamente actualizables; de lo contrario indique la/s causa/s. En cualquier caso, **justifique** claramente.

- VISTA\_1, que contenga los datos de los mensajes privados con asunto iniciado en 'consulta' que contienen algún archivo adjunto de imagen creado el año actual.
- VISTA\_2, con el identificador y texto de todos los mensajes junto con la cantidad de archivos descargados que tienen adjuntos, si es que poseen.

b) Dada la siguiente vista sobre el esquema dado:

```

create view VISTA_3 as
select a.id_adjunto, a.tipo_adj, a.tamano, a.fecha_creacion, ai.resolucion as dimension
from adjunto a join imagen ai on a.id_adjunto = ai.id_adjunto
union
select a.id_adjunto, a.tipo_adj, a.tamano, a.fecha_creacion, aa.duracion as dimension
from adjunto a join audio aa on a.id_adjunto = aa.id_adjunto;
  
```

Se requiere definir un *trigger instead of* para capturar el evento de borrado de tuplas sobre la vista por medio de una implementación en PostgreSQL que:

- brinde un mensaje descriptivo de error, en caso de no existir lo que se pretende eliminar
- propague adecuadamente la operación hacia las tablas base que conforman la vista, en caso de que la operación pueda proceder

Solución Ej. 2.a)

**Pregunta 3**

Finalizado

Puntúa como 1,00

**Solución Ej. 2.b)**

**Pregunta 4**

Finalizado

Puntúa como 1,00

## Ejercicio 3

Dadas las siguientes definiciones de vistas sobre el esquema de Películas ([link](#) esquema):

```
CREATE VIEW Tarea_9
AS SELECT id_tarea, nombre_tarea, sueldo_minimo, sueldo_maximo
FROM tarea
WHERE nombre_tarea LIKE 'Tarea %9';

CREATE VIEW Tarea_smax
AS SELECT * FROM Tarea_9
WHERE sueldo_maximo < 10000
WITH CASCADED CHECK OPTION;

CREATE VIEW Tarea_smin
AS SELECT * FROM Tarea_9
WHERE sueldo_minimo < 5000
WITH LOCAL CHECK OPTION;
```

- a) Para las siguientes inserciones ejecutadas en el orden dado, indique en cada caso si procede o se rechaza, justificando claramente el motivo.  
b) Luego de ejecutadas las 4 operaciones, indique en cuál/es de los objetos (tabla, vistas) aparecen las nuevas tuplas.

1. insert into Tarea\_smax (id\_tarea, nombre\_tarea, sueldo\_minimo, sueldo\_maximo) values (8009, 'Tarea 8009', 6000, 8000);
2. insert into Tarea\_9 (id\_tarea, nombre\_tarea, sueldo\_minimo, sueldo\_maximo) values (8000, 'Tarea 8000', 3000, 10000);
3. insert into Tarea\_smax (id\_tarea, nombre\_tarea, sueldo\_minimo, sueldo\_maximo) values (9000, 'Tarea 9000', 6000, 7000);
4. insert into Tarea\_smin (id\_tarea, nombre\_tarea, sueldo\_minimo, sueldo\_maximo) values (8010, 'Tarea 8010', 4000, 9000);

Solución Ej. 3.a)

Pregunta 5

Finalizado

Puntúa como 1,00

**Solución Ej. 3.b)**



**Pregunta 6**

Finalizado

Puntúa como 1,00

## Ejercicio 4

La siguiente consulta sobre el esquema Wireless Internet Service Provider del TPE ([link esquema](#)) se ha planteado para obtener la información sobre los equipos correspondientes a clientes que no poseen CUIT, ubicados en ciudades con identificador menor a 3000.

```
SELECT * FROM equipo e
JOIN cliente cl ON (e.id_cliente = cl.id_cliente)
JOIN persona p ON (cl.id_cliente = p.id_persona)
JOIN direccion d ON (d.id_persona = p.id_persona)
JOIN barrio b ON (d.id_barrio = b.id_barrio)
JOIN ciudad c ON (b.id_ciudad = c.id_ciudad)
WHERE p.CUIT is null AND c.id_ciudad < 3000;
```

Analice si la consulta responde a lo solicitado y si considera que representa una consulta optimizada. Si su respuesta es Sí, justifique por qué lo considera así. Si su respuesta es NO, plantee un SQL optimizado, justificando la/s modificación/es introducida/s.

Actividad previa

[◀ Evaluación Parcial 1](#)

Ir a...

Siguiente actividad

[Examen Recuperatorio 1 ►](#)

## Mantente en contacto

Facultad, Pabellón Central Paraje Arroyo Seco. Campus Universitario. (B7001BBO) Tandil.  
Buenos Aires, Argentina

🌐 <https://exa.unicen.edu.ar/>

📞 [\(+54\) \(0249\) 438-5650 Conmutador: int. 2000](tel:+5402494385650)

✉️ [moodle@exa.unicen.edu.ar](mailto:moodle@exa.unicen.edu.ar)



📱 Descargar la app para dispositivos móviles

[Facultad de Ciencias Exactas – UNICEN](#)

Contacto administradores plataforma: E-mail [moodle@exa.unicen.edu.ar](mailto:moodle@exa.unicen.edu.ar) – Tel. [+54 0249 4385650](tel:+5402494385650) int. 2098