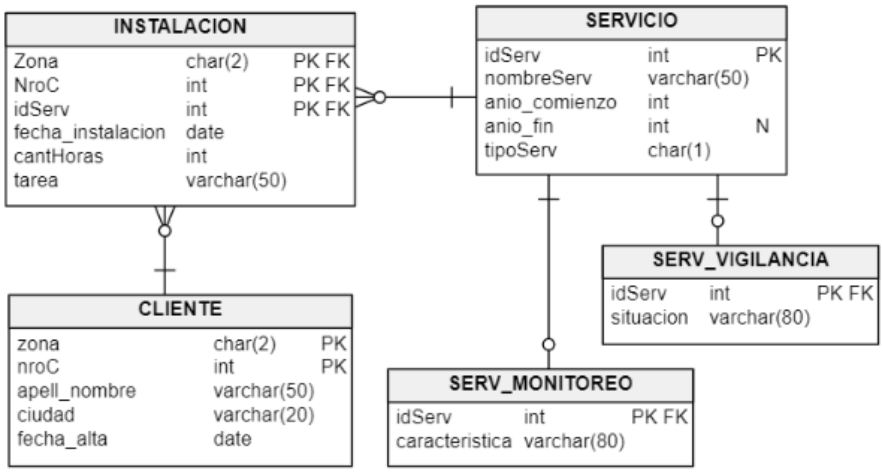
Comenzado el	jueves, 29 de febrero de 2024, 09:39
Estado	Finalizado
Finalizado en	jueves, 29 de febrero de 2024, 12:19
Tiempo empleado	2 horas 40 minutos

Información

Considere que el esquema corresponde a un sistema de gestión de servicios a clientes (script). De cada cliente se registra su identificador, apellido y nombre, ciudad y fecha de alta, además las instalaciones de servicios que posee y sus características. De los servicios se almacena su identificador, nombre, año de comienzo y de fin, más el tipo de servicio, que puede ser de monitoreo (M) o de vigilancia (V).



Pregunta 1

Finalizado

Puntúa como 1,00

Sobre el esquema dado (link), incorpore los siguientes controles en SQL estándar mediante el recurso declarativo más restrictivo y utilizando sólo las tablas y atributos necesarios. Justifique el tipo de restricción definida en cada caso.

a) Los clientes con menos de 3 años de antigüedad pueden tener hasta 3 servicios instalados de cada tipo.

b) La fecha de instalación de cada servicio no puede ser anterior ni posterior a los años de comienzo y de fin, respectivamente, asociadas a dicho servicio.

c) El año de comienzo de los servicios que son de vigilancia debe ser posterior a 2020.

Ejecicio 1 a:
El recurso declarativo mas restrictivo en este caso es un RI Global ya que abarca realizar una consulta en donde se ven involucradas tres tablas (cliente, servicio y instalacion) para satisfacer esa condicion.

```
create assertion as_ej_1_a check (
not exists(select 1
from cliente c
join instalacion i on (i.nroc = c.nroc and i.zona = c.zona)
join servicio s on (s.idServ = i.idServ)
where c.fecha_alta between (current_date - 1095) and current_date
group by tiposerv
having count(*) <= 3));
```

ejercicio 1 b :

El recurso declarativo mas restrictivo en este caso es un RI Global ya que abarca realizar una consulta en dos tablas (instalacion, servicio) para satisfacer esa condicion.

```
create assertion as_ej_1_b check (
not exists (select 1
from instalacion i
join servicio s on i.idserv = s.idserv
where (EXTRACT('YEAR' FROM i.fecha_instalcion) >= s.anio_comienzo)
and (s.anio_fin is null or (EXTRACT('YEAR' FROM i.fecha_instalcion) <= s.anio_fin)));
```

ejercicio 1 c :

El recurso declarativo mas restrictivo en este caso es una RI de tupla ya que solo de debera de chequear por los campos tiposerv y anio_comienzo para satisfacer la condicion.

```
ALTER TABLE servicio ADD CONSTRAINT CK_ej_c
CHECK (tiposerv = 'V' and anio_comienzo > 2020)
```

Comentario:
a) Tipo de declaración: Correcta. Implementación: La condición sobre la fecha de alta es incorrecta y Incorrecto el group by debido a que agrupa solo por tipo de servicio cuando se ensamblan las tres tablas.

b) Tipo de declaración: Correcta. Implementación: Solución correcta.

c) Tipo de declaración: Correcta. Implementación: Falta en la condición OR tipoServ <> 'V'.

Pregunta 2

Finalizado

Puntúa como 1,00

Considere que en la tabla Servicio del esquema dado (link), se ha agregado un atributo cant_clientes en el cual se requiere registrar la cantidad de clientes a los que se ha instalado cada servicio.

Explique e implemente en forma completa y de modo eficiente en PostgreSQL una solución que en cada caso permita:
a) establecer el valor inicial de cant_clientes a partir de los datos ya existentes en la BD.

b) mantener automáticamente actualizado el atributo cant_clientes ante operaciones sobre la BD.

ejercicio 2 a:

```
update servicio s set cant_clientes = (
select count(*)
from instalacion i
where i.idserv = s.idserv) WHERE idserv = s.idserv ;
```

ejercicio 2 b:

```
CREATE OR REPLACE FUNCTION FN_ej_2_b() RETURNS TRIGGER AS $$
BEGIN
IF tg_op = 'INSERT' THEN
UPDATE servicio SET cant_clientes = cant_clientes + 1 WHERE idserv = NEW.idserv;
RETURN NEW;
ELSE
IF tg_op = 'UPDATE' THEN
UPDATE servicio SET cant_clientes = cant_clientes + 1 WHERE idserv = NEW.idserv;
UPDATE servicio SET cant_clientes = cant_clientes - 1 WHERE idserv = OLD.idserv;
RETURN NEW;
ELSE
UPDATE servicio SET cant_clientes = cant_clientes - 1 WHERE idserv = OLD.idserv;
RETURN OLD;
END IF;
END IF;
END $$ LANGUAGE 'plpgsql';
CREATE OR REPLACE TRIGGER TR_ej_2_c
AFTER INSERT OR DELETE OR UPDATE OF idserv
ON instalacion
FOR EACH ROW
EXECUTE FUNCTION FN_ej_2_b();
```

Para el primer punto donde se requiere actualizar todos los campos, realizaremos un update que realizara una consulta en la tabla instalacion por idserv para contar la cantidad de instalaciones que tienen ese servicio. Luego para que sea automaticamente actualizable esta columna lo que se hara es hacer un trigger en la tabla instalacion ya que esta columna depende de los registros en esta tabla por lo que luego de que se realize una transaccion de delete insert o update de idserv en la tabla instalacion se dispara el trigger realizar el update correspondiente dependiendo de la operacion en la tabla servicio.

Comentario:
a) Explicación: Explicación correcta. Solución: Correcta.

b) Explicación: Explicación correcta. Implementación: Cabecera: correcta. Función: Solución correcta.

Pregunta 3

Finalizado

Puntúa como 1,00

Dados los siguientes servicios requeridos sobre el esquema de Películas (unc_esq_peliculas), plantee el SQL que lo resuelve

1. **¿Cuántos distribuidores nacionales han realizado exactamente 10 entregas?**
2. **Listar el/los distribuidor/es con la mayor cantidad de entregas realizadas, indicando cuál es dicha cantidad**

ejercicio 1

```
SELECT count(d.id_distribuidor)
FROM unc_esq_peliculas.distribuidor d
JOIN (SELECT id_distribuidor, count(*) cant_entregas
FROM unc_esq_peliculas.entrega e
GROUP BY id_distribuidor) e ON e.id_distribuidor = d.id_distribuidor
WHERE d.tipo = 'N' AND e.cant_entregas = 10;
```

ejercicio 2

```
SELECT d.*
FROM unc_esq_peliculas.distribuidor d
JOIN( SELECT id_distribuidor, max(id_distribuidor) max_entregas
FROM unc_esq_peliculas.entrega
GROUP by id_distribuidor) e ON e.id_distribuidor = d.id_distribuidor
WHERE e.max_entregas = (SELECT max(id_distribuidor) maximo
FROM unc_esq_peliculas.entrega
GROUP by id_distribuidor
ORDER BY maximo DESC
LIMIT 1);
```

Pregunta **4**

Finalizado

Puntúa como 1,00

Considere que se ha planteado la siguiente consulta sobre el esquema de Películas (unc_esq_peliculas) a fin de recuperar la información sobre las películas en idioma Italiano incluidas en alguna entrega de distribuidores nacionales.

```
SELECT *
FROM pelicula p JOIN renglon_entrega re ON (p.codigo_pelicula =re.codigo_pelicula)
JOIN entrega e ON (e.nro_entrega = re.nro_entrega)
JOIN video v (ON e.id_video = v.id_video)
JOIN distribuidor d ON (d.id_distribuidor =e.id_distribuidor)
JOIN nacional n ON (n.id_distribuidor = d.id_distribuidor)
WHERE d.tipo = 'N' and idioma = 'Italiano';
```

Analice si la consulta permite responder a lo solicitado y si representa una consulta optimizada o no. Si su respuesta es SÍ, justifique por qué, de lo contrario reescriba la consulta justificando las estrategias consideradas para su optimización.

No responde a lo solicitado ya que solo esta queriendo consultar la informacion de las peliculas y al realizar SELECT * esta recuperando informacion no necesaria.
Esta consulta no se encuentra optimizada ya que:
- Proyecta datos demas bastaria con hacer SELECT p.* en lugar SELECT *.
- Realiza un JOIN con video y este no es necesario para poder responder a la consulta solicitada.
- Se podria filtrar la tabla pelicula idioma = 'Italiano' antes de realizar los join, redujendo considerablemente los joins posteriores.
- Se podria evitar el join con la tabla nacional filtrando en distribuidor por tipo = 'N'.

La consulta optimizada quedaria de la siguiente manera:

SELECT p.*

FROM (SELECT * FROM unc_esq_peliculas.pelicula WHERE idioma = 'Italiano') p

JOIN unc_esq_peliculas.renglon_entrega re ON (p.codigo_pelicula = re.codigo_pelicula)

JOIN unc_esq_peliculas.entrega e ON (e.nro_entrega = re.nro_entrega)

JOIN (SELECT * FROM unc_esq_peliculas.distribuidor WHERE tipo = 'N') d ON (d.id_distribuidor =e.id_distribuidor);

Pregunta 5

Finalizado

Puntúa como 1,00

Sobre el esquema dado (link) construya una vista que contenga lo indicado en cada caso y que resulte automáticamente actualizable en PostgreSQL siempre que sea posible (justifique por qué es posible o no):

a) datos de los clientes dados de alta el año actual que poseen únicamente instalaciones de servicios de vigilancia

b) datos completos asociados a cada servicio incluyendo su situación o su característica, según sea su tipo

Para aquella/s vista/s que no soporte/n actualización automática en PostgreSQL, provea una implementación procedural que permita propagar convenientemente operaciones de inserción sobre la vista. Explique brevemente cómo funciona su solución.

Ejercicio a

```
CREATE OR REPLACE VIEW ej_5_a AS
SELECT c.*
FROM cliente c
WHERE EXTRACT('YEAR' FROM c.fecha_alta) = EXTRACT('YEAR' FROM current_date)
AND EXISTS (SELECT 1 FROM instalacion i WHERE i.zona = c.zona AND i.nroc = c.nroc)
AND NOT EXISTS (SELECT 1
FROM instalacion i
JOIN servicio s ON (i.idserv = s.idserv)
WHERE i.zona = c.zona AND i.nroc = c.nroc AND s.tiposerv = 'M');
```

Esta vista sera actualizable ya que:
- Mantiene la key presserved
- No contiene funciones de agregacion
- No contiene la clausula distinc
- No realiza subconsultas en el select

Ejercicio b

```
CREATE OR REPLACE VIEW ej_5_b AS
SELECT s.*, COALESCE (sm.caracteristica,sv.situacion) carc_or_sit
FROM servicio s
LEFT JOIN serv_monitoreo sm ON s.idserv = sm.idserv
LEFT JOIN serv_vigilancia sv ON s.idserv = sv.idserv;
```

Esta vista no sera actualizable ya que contiene funciones de agregacion (COALESCE).

Para que sea actualizable habria que realizar un trigger instead of:

```
CREATE OR REPLACE FUNCTION FN_ej_5_b() RETURNS TRIGGER AS $$
BEGIN
```

```
INSERT INTO servicio (idserv, nombreserv, anio_comienzo, anio_fin, tiposerv) VALUES(NEW.idserv, NEW.nombreserv, NEW.anio_comienzo, NEW.anio_fin, NEW.tiposerv);
IF (NEW.tiposerv = 'V') THEN
INSERT INTO serv_vigilancia (idserv, situacion) VALUES (NEW.idserv, NEW.carc_or_sit);
ELSE
INSERT INTO serv_monitoreo (idserv, caracteristica) VALUES (NEW.idserv, NEW.carc_or_sit);
END IF;
RETURN new;
END $$ LANGUAGE 'plpgsql';
CREATE OR REPLACE TRIGGER TR_ej_5_b
INSTEAD OF INSERT
ON ej_5_b
FOR EACH ROW
EXECUTE FUNCTION FN_ej_5_b();
```

a partir del siguiente trigger cuando se quiera realizar una operacion de insercion sobre la vista en lugar de tomar la consulta literal y dar error ya que no podria realizarla porque no sabria como insertar, lo que hara es reemplazar esa consulta por la ejecucion del trigger permitiendo la insercion y realizando un insert en servicio y otro en la tabla correspondiente serv_monitoreo o serv_vigilancia dependiendo del tiposerv.

Comentario:
a) Falta especificar en el EXISTS JOIN servicio s ON (i.idserv = s.idserv)

..... s.tiposerv = 'V'

```
AND EXISTS (SELECT 1 FROM instalacion i WHERE i.zona = c.zona AND i.nroc = c.nroc)
```

b) Bien la vista. Los triggers INSTEAD OF no retornan RETURN new; (debe ser NULL el retorno)

Actividad previa

◄ Avisos

Ir a...

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

moodle@exa.unicen.edu.ar

Descargar la app para dispositivos móviles