Finales BD1 (Pl. 2011) y BD2 (Pl. 2023)- Ing.

Comenzado el	jueves, 13 de marzo de 2025, 09:30
Estado	Finalizado
Finalizado en	jueves, 13 de marzo de 2025, 12:30
Tiempo empleado	2 horas 59 minutos

Navegación por el cuestionario

i 1 2 3 4 5

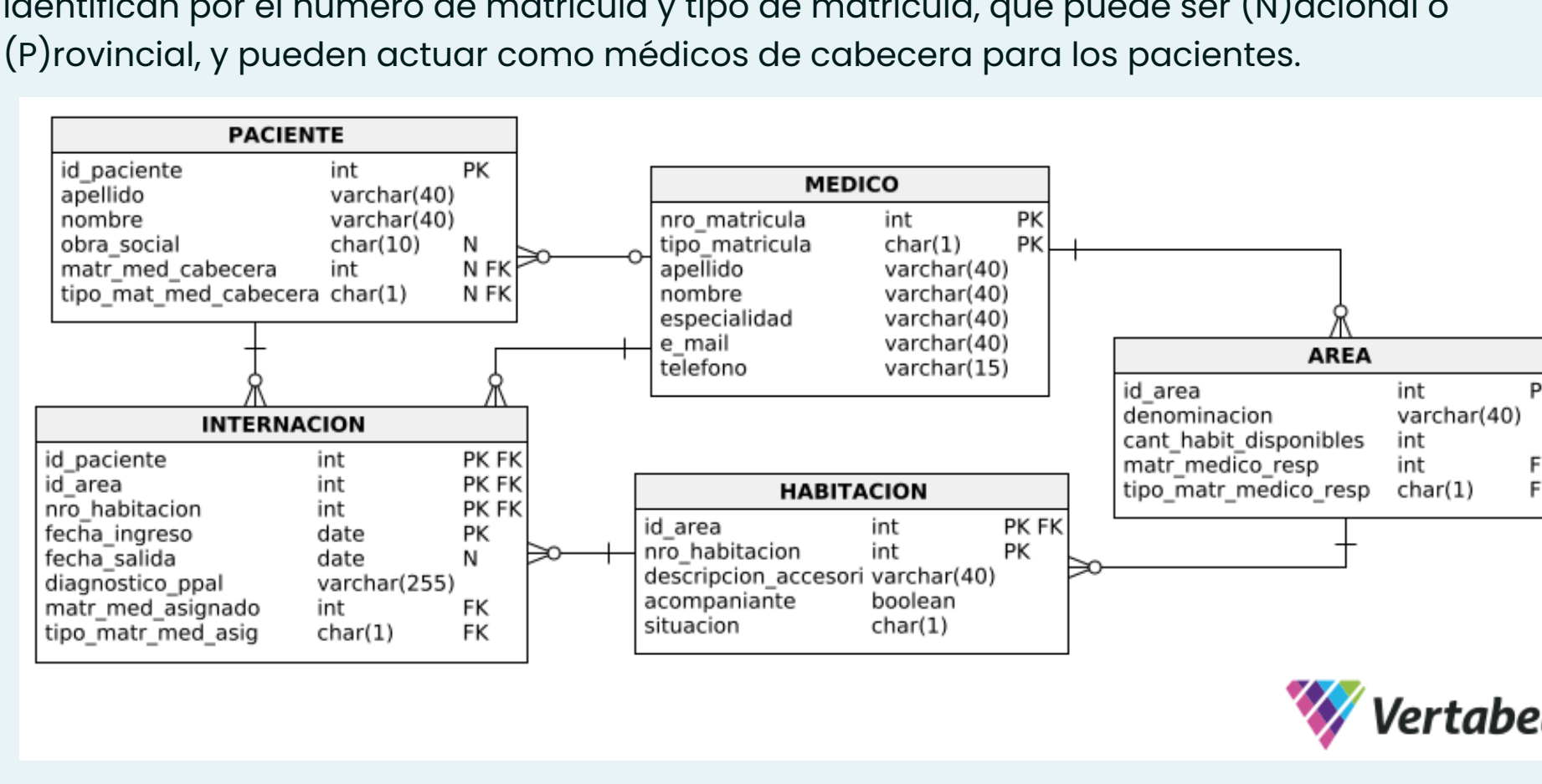
Mostrar una página cada vez

Finalizar revisión

Información

Marcar pregunta

Dado el siguiente esquema relacional en PostgreSQL (script creación), correspondiente a la gestión de internaciones en un centro de salud. Se organiza en torno a áreas médicas, cada una con un médico responsable y la cantidad de habitaciones que se encuentran disponibles (no ocupadas). Cada habitación tiene una descripción de los accesorios que posee, indicación de si dispone o no lugar para acompañante y una situación que indica si la habitación está actualmente (L)libre u (O)cupada. En cada internación de un paciente se asigna una habitación, registrando la fecha de ingreso y de salida (cuando se conoce), el diagnóstico principal y el médico asignado. Los médicos poseen una especialidad y se identifican por el número de matrícula y tipo de matrícula, que puede ser (N)acional o (P)rovincial, y pueden actuar como médicos de cabecera para los pacientes.



Consideraciones

- Cuando se requiere explicar/justificar, hágalo de manera clara y concisa.
- En los ejercicios que solicitan implementaciones en SQL o PL/pgSQL, proponga soluciones lo más eficientes posible.
- Durante el examen únicamente se puede usar PhpPgAdmin y la documentación en línea de PostgreSQL.

Pregunta 1
Finalizado
Se puntúa como 0 sobre 1,00
Marcar pregunta

Ejercicio 1

Considere que se ejecutan las siguientes operaciones sobre la base dada, inicialmente vacía:

INSERT INTO MEDICO (nro_matricula, tipo_matricula, apellido, nombre, especialidad, e_mail, telefono)
VALUES (12345, 'N', 'Gonzalez', 'Maria', 'Cardiologia', 'mgonzalez@email.com', '1122334455'),
(67890, 'P', 'Fernandez', 'Juan', 'Medicina General', 'jfernandez@email.com', '1199887766'),
(54321, 'N', 'Garcia', 'Fernando', 'Clinica Medica', 'fgarcia@email.com', '1177554433');
INSERT INTO PACIENTE (id_paciente, apellido, nombre, obra_social, matr_med_cabecera, tipo_mat_med_cabecera) VALUES
(1, 'Perez', 'Carlos', 'OSDE', 12345, null), (2, 'Lopez', 'Ana', 'SwissMed', null, 'P'),
(3, 'Ortiz', 'Laura', null, 12345, 'N'), (5, 'Ramirez', 'Mario', 'PAMI', 67890, 'P');

1.a) Teniendo en cuenta las diferentes posibilidades de ensamble para listar los datos de los pacientes y de sus médicos de cabecera, provea la/s sentencia/s SQL correspondiente/s y analice en qué caso/s se listarían todas las tuplas de PACIENTE y en qué caso/s no, justificando por qué.

1.b) Luego de ejecutadas las operaciones anteriores, indique si procede o no cada una de las siguientes sentencias y por qué:
b.1) UPDATE PACIENTE SET tipo_mat_med_cabecera = 'N' WHERE id_paciente = 2;
b.2) UPDATE PACIENTE SET tipo_mat_med_cabecera = 'P' WHERE id_paciente = 3;

1.c) Las siguientes expresiones de consulta SQL intentan listar los *datos de los médicos que no están asignados como médicos de cabecera de pacientes.* Determine cuál/es de ellas permiten recuperar correctamente dicha información en cualquier caso; si no fuera así, justifique claramente por qué e indique la corrección necesaria en cada una para obtener lo requerido.

c.1) SELECT * FROM MEDICO
WHERE (nro_matricula, tipo_matricula) NOT IN (
SELECT matr_med_cabecera, tipo_mat_med_cabecera
FROM PACIENTE);

c.2) SELECT * FROM MEDICO M
WHERE tipo_matricula NOT IN (
SELECT tipo_mat_med_cabecera FROM PACIENTE P
WHERE matr_med_cabecera IS NOT NULL and tipo_mat_med_cabecera IS NOT NULL);

c.3) SELECT * FROM MEDICO
WHERE NOT EXISTS
(SELECT matr_med_cabecera, tipo_mat_med_cabecera
FROM PACIENTE);

```
-- 1.a
	-- Natural join
		select p.*, m.*
		from medico m
		join paciente p on m.nro_matricula = p.matr_med_asignado
							and m.tipo_matricula
		-- Esta consulta con natural join no mostraria todas las tup
		-- La consulta muestra solo a los pacientes con id_paciente 

	-- LEFT JOIN
		select p.*, m.*
		from medico m
		left join paciente p on m.nro_matricula = p.matr_med_asignad
							and m.tipo_matricula
		-- Esta consulta con left join actuaria de igual manera que e
```

Comentario:
a) Reg. La primera sentencia estaría bien, pero corresponde a (inner) join, NO natural join como dice (el cual tiene un comportamiento muy diferente en este caso)
LEFT JOIN: falla porque están mal los nombres de atrib (no es asignado, sino cabecera), el resultado no es igual al anterior ya que en este caso incluiría todos los médicos
faltan 2 posibilidades (full y natural join)

b1) justif. incompleta: por qué no es necesario constatar..?, b2) ok

c1) Esta consulta no recuperará la inf. en cualquier caso (ej: si hay alguna FK totalmente nula), c2) bien- (así la consulta falla, los atrib. antes del NOT IN van entre paréntesis), c3) bien

Pregunta 2
Finalizado
Se puntúa como 0 sobre 1,00
Marcar pregunta

Ejercicio 2

Considere las siguientes restricciones sobre el esquema dado debidas a políticas de la institución:

A- Al menos el 50% de las camas de cada área deben tener lugar para acompañante.

B- Los pacientes de la obra social PAMI deben tener un médico de cabecera.

C- El médico asignado a una internación debe tener la misma especialidad que el médico responsable del área correspondiente.

Para cada una de las condiciones anteriores:

2.a) plantee la implementación completa en SQL estándar, mediante el recurso declarativo más restrictivo posible (justifique su elección).

2.b) para aquellas restricciones que no puedan incorporarse en PostgreSQL según lo implementado en 2.a), indique y justifique cada uno de los eventos críticos que deben ser chequeados y provea las declaraciones de los triggers correspondientes.

```
-- 2.a
	-- A)
		alter table habitacion add constraint check_camas
		check (not exists(select 1
							from habitacion h1
							where exists
								(select 1
									from
									where

									grou
		-- Tenemos una restriccion de tabla porque debemos controlar

	-- B)
		alter table paciente add constraint check_medico
```

Comentario:
2.a)

a) Mal, es imposible hacer lo realizado en el group by:

group by h2.id_area having count(h2.*) < (count(h1.*)*0.5))));

b)Mal, dado que falta considerar que no tengan OS. El caso de un paciente sin OS pero con médico es valido y ahora se rechaza.

c) Bien

2.b)

a) Bien pero falto considerar si la habitación cambia de area.

c) Mal, el análisis de eventos no es correcto:

"- **MODIFICACION de clave extranjera (matr_med_asignado, tipo_matr_med_asig) en HABITACION**"

no existen esos atributos en habitación.

Pregunta 3
Finalizado
Se puntúa como 0 sobre 1,00
Marcar pregunta

Ejercicio 3

Provea una implementación completa en PostgreSQL mediante el recurso que considere más adecuado para resolver cada uno de los siguientes requerimientos sobre el esquema dado, y justifique incluyendo los aspectos teóricos correspondientes:

3.a) dada una cierta fecha, brindada por un usuario, se debe verificar si se ha superado el 75% de ocupación de habitaciones en el centro de salud y registrar en una tabla auxiliar (ya creada) si esto sucede, indicando tal fecha y el porcentaje de capacidad alcanzada.

3.b) se debe mantener automáticamente actualizada la cantidad de habitaciones disponibles en cada área a medida que se ocupan o desocupan camas de pacientes internados (considere que los valores actuales de dicho atributo son consistentes con los datos existentes en la base).

```
-- 3.a
	create table aux(
		fecha_ocupacion date not null,
		ocupacion_total float not null);

	create or replace procedure ocupacion_total(fecha date)
	language plpgsql as $$
	begin
		if ((select count(*) from habitacion where situacion ilike '(
			insert into aux (fecha_ocupacion, ocupacion_total)
				values (fecha,
							((select count(*) from habita
		end if;
	end $$;
```

Comentario:
a) Mal

- Que es esto ? if ((select count(*) from habitacion where situacion ilike 'O') >= (select count(*)*0.75 from habitacion)) is true) ... is true ?

- Que pasa si ejecuto dos veces con la misma fecha el procedimiento ?

- No le parece muy ineficiente calcular dos veces el porcentaje ? Una vez en el if y otra para insertar, hace 4 veces un table scan de gusto (se piden soluciones eficientes)

b) Regular.

- Por que si borro automáticamente asume que esa habitación estaba libre ? Disponibles son para usar, y a lo mejor dio de baja una habitación que estaba ocupada y en ese caso no se incrementa nada.

- Que pasa si cambio a una habitacion de áerea ?

Pregunta 4
Finalizado
Se puntúa como 0 sobre 1,00
Marcar pregunta

Ejercicio 4

Implemente las siguientes vistas sobre el esquema dado, de manera que resulten automáticamente actualizables en PostgreSQL siempre que sea posible; en caso contrario, justifique la/s razón/es. En ambos casos, explique las posibilidades de actualización sobre cada vista.

4.a) V1, con identificador y denominación de la/s área/s con la mayor cantidad de internaciones en el año actual.

4.b) V2, con datos de todos los pacientes con obra social, junto con el número total de internaciones que han tenido en los últimos 5 años y el promedio de duración de las mismas (si no registra internaciones, estos valores deben ser 0).

4.c) V3, con apellido y nombre de los médicos de especialidad cardiología asignados únicamente a internaciones realizadas en el área de cardiología.

```
-- 4.1
	create view v1
	as (select a.id_area, a.denominacion
		from area a join internacion i using (id_area)
		group by a.id_area having extract(year from i.fecha_ingreso)
		order by i.count() desc);
	-- Esta vista NO ES AUTOMATICAMENTE ACTUALIZABLE en PostgreSQL porque
	-- No puede realizarse de vista de otra manera que resulte automatica

-- 4.2
	create view v2
	as (select p.*, count(i.*) as "total_internaciones", coalesce(avg(i.1
		from paciente p join internacion i using (id_paciente)
		where p.obra_social is not null)
		group by p.id_paciente having (now()-interval'5 years' >= i.1
```

Comentario:
a) La consulta da error; además no se requiere join (la vista puede plantearse autom. actualizable ya que solo se piden datos de AREA), la condición del having en realidad debe ir en un where (no es condición de grupo), el order by da error. En el caso que funcionara, su planteo sólo listaría un área aunque haya más de un área cumpliendo la condición pedida (se piden la/s área/S...).

b) Reg- La consulta no funciona: sobra ")" luego de null, la condición del having es una condición a nivel del where y estaría al revés el control (se pide de los últimos 5 años). Además no listaría los pacientes sin internaciones (se pide todos)

c) Reg. La consulta da error: select id_aree , pero además no asegura lo pedido, ya que puede incluir a cardiólogos que participaron de alguna internación en otra área distinta a Cardiologia (se pide únicamente).

Pregunta 5
Finalizado
Se puntúa como 0 sobre 1,00
Marcar pregunta

Ejercicio 5

Considerando la consulta SQL expresada en 4.c) sobre el esquema dado:

5.a) Si se conoce que existen índices por cada clave primaria del esquema, qué otro/s índice/s considera necesario/s para responder con mayor eficiencia a la consulta? Justifique.

5.b) Explique cómo resolvería tal consulta el optimizador de consultas, considerando la presencia de índices y otros factores de optimización.

```
5.a) considero necesario un indice denominacion y especialidad que utilice HAS
5.b) respondido tambien en a. El DBMS primero verifica las condiciones y luego
```

Comentario:
a) Correcto.

b) Mal, relata lo que hacer el query pero cómo el optimizador resuelve la consulta.

Finalizar revisión

Ir a...

Siguiente actividad
Final BDI (Plan 2011), BD2 (Plan 2023) - 05/12/2024 ▶

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