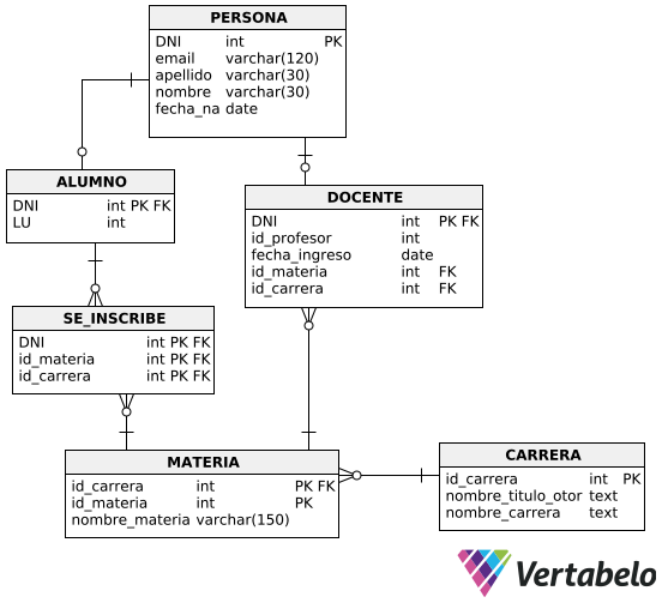
Comenzado el	sábado, 7 de junio de 2025, 14:18
Estado	Finalizado
Finalizado en	sábado, 7 de junio de 2025, 15:13
Tiempo empleado	54 minutos 34 segundos

Pregunta 1

Finalizado

Se puntúa como 0 sobre 1,00



Tiempo estimado 15 min

Para el esquema de la figura y dadas las siguientes definiciones de vistas:

```
CREATE OR REPLACE VIEW v_gmail
AS
SELECT DNI, email, apellido, nombre, fecha_nac
FROM persona
WHERE email like '%gmail%';

CREATE OR REPLACE VIEW v_gmail_mayor
AS
SELECT *
FROM v_gmail
WHERE DNI > 23456789
WITH LOCAL CHECK OPTION;

CREATE OR REPLACE VIEW v_gmail_parcial
AS
SELECT *
FROM v_gmail_mayor
WHERE apellido like 'Bet%'
WITH CASCADED CHECK OPTION;
```

Para las siguientes sentencias ejecutadas de manera independiente señalar las opciones que son VERDADERAS. Nota: Las respuestas incorrectas restan del puntaje total. Tenga cuidado al cortar y pegar las sentencias con las comillas simples ' '.

a. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)
VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))
NO Procede, da error.

b. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

Es Verdadera porque la operación procede e inserta los datos en la tabla persona al cumplir con el WHERE de v_gmail_mayor (LOCAL CHECK OPTION) y no evaluar el WHERE de v_gmail por no tener opción

VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

Procede, inserta los datos en la tabla persona pero no se muestran en la vista v_gmai_ mayor

de chequeo en su definición. Luego, la tupla no se muestra en la vista v_gmail porque email contiene "hotmail" y no "gmail", y por ende no se muestra tampoco en la vista v_gmail_mayor.

c. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

VALUES (33456789, 'cc@gmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

Procede, inserta los datos en la tabla persona y se muestran en la vista v_gmail y en la vista v_gmail_mayor

Es Verdadera porque la operación procede e inserta los datos en la tabla persona al cumplir con el WHERE de v_gmail_mayor (LOCAL CHECK OPTION) y no evaluar el WHERE de v_gmail por no tener opción de chequeo en su definición. Luego, la tupla se muestra en la vista v_gmail porque email contiene "gmail", y luego se muestra en la vista v_gmail_mayor porque el dni 33 millones es mayor a 23456789

d. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

VALUES (33456789, 'cc@ hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

Procede, inserta los datos en la tabla persona y se muestran en todas las vistas

e. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

NO Procede, porque no cumple con la condición de la vista v_gmail

f. INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

Procede, inserta los datos en la tabla persona pero no se muestran en la vista v_gmail

Es Verdadera porque la operación procede e inserta los datos en la tabla persona al cumplir con el WHERE de v_gmail_mayor (LOCAL CHECK OPTION) y no evaluar el WHERE de v_gmail por no tener opción de chequeo en su definición. Luego, la tupla no se muestra en la vista v_gmail porque email contiene "hotmail" y no "gmail"

Respuesta correcta

Las respuestas correctas son:

INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)

VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))

Procede, inserta los datos en la tabla persona pero no se muestran en la vista v_gmail

,

INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)
VALUES (33456789, 'cc@hotmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla persona pero no se muestran en la vista v_gmai_ mayor

,

INSERT INTO v_gmail_mayor (DNI, email, apellido, nombre, fecha_nac)
VALUES (33456789, 'cc@gmail.com', 'Beta', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla persona y se muestran en la vista v_gmail y en la vista v_gmail_mayor

Pregunta 2
Finalizado
Se puntúa como 0 sobre 1,00

Utilizando el esquema unc_esq_voluntario. Cuáles son los 2 coordinadores(nombre) que han tenido a cargo la menor cantidad de voluntarios que hayan realizado cualquier tarea terminada en CLERK.

Tiempo estimado 20 min

- a. Mattheu, Adam, Kevin, Shanta, Payam
- b. Mattheu, Adam
- c. Ninguna de las opciones es correcta
- d. Tj, Hazel
- e. Luisa, Den

Respuesta correcta
SQL que lo resuelve:

```
select coor.nro_voluntario, coor.nombre  as "Coordinador", count(*)
from  voluntario V
join voluntario coor on V.id_coordinador=coor.nro_voluntario
join tarea t on V.id_tarea = t.id_tarea
where t.id_tarea  like '%CLERK'
group by coor.nro_voluntario, coor.nombre
order by 3 asc
```

La respuesta correcta es:
Ninguna de las opciones es correcta

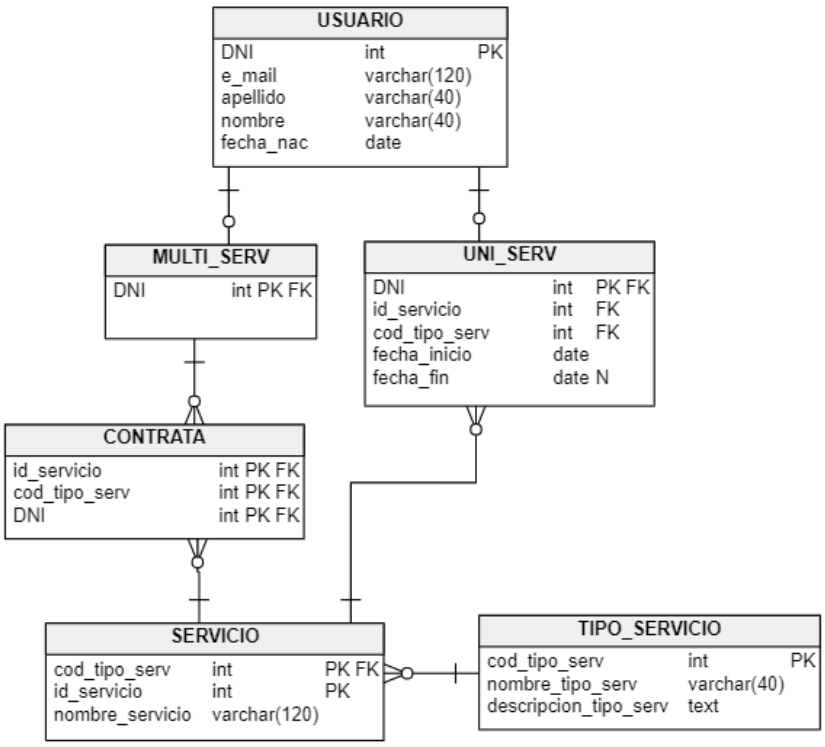
Pregunta 3

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 15 min

Dado el esquema de la figura:



Si las foreign key fueran las que se indican debajo y se encuentran cargados los siguientes datos:

TIPO_SERVICIO		
codigo_tipo_serv	*nombre_tipo_serv*	....
A	nombre1	
B	nombre2	
C	nombre3	
D	nombre4	
E	nombre5	

SERVICIO		
codigo_tipo_serv	*id_servicio*	*nombre_servi*
A	2	servicio1
B	3	servicio2
A	1	servicio3
E	1	servicio4
C	4	servicio5

UNI_SERV			
DNI	*id_servicio*	*cod_tipo_serv*	...
38159753	1	A	
41597842	3	B	
39852456	1	E	
42357852	2	A	
43591573	1	E	

```
FK_SERVICIO_TIPO_SERVICIO
    ON UPDATE RESTRICT
    ON DELETE CASCADE
FK_UNI_SERV_SERVICIO
    ON UPDATE CASCADE
    ON DELETE RESTRICT
```

¿Cuál/es de las siguientes operaciones proceden y cuál/es fallan?

UPDATE UNI_SERV SET codigo_tipo_serv ='E' WHERE id_servicio = 1;	Procede
UPDATE TIPO_SERVICIO SET codigo_tipo_serv ='F' WHERE codigo_tipo_serv ='B';	Procede
DELETE FROM TIPO_SERVICIO WHERE codigo_tipo_serv ='C';	Procede

DELETE FROM SERVICIO WHERE id_servicio = 1;

Procede

DELETE FROM UNI_SERV WHERE DNI = '41597842';

Procede

DELETE FROM TIPO_SERVICIO WHERE codigo_tipo_serv ='B'

Falla

UPDATE SERVICIO SET id_servicio = 5 WHERE codigo_tipo_serv ='E';

Procede

Respuesta parcialmente correcta.

La respuesta correcta es:
UPDATE UNI_SERV SET codigo_tipo_serv ='E' WHERE id_servicio = 1; → Procede,

UPDATE TIPO_SERVICIO SET codigo_tipo_serv ='F' WHERE codigo_tipo_serv ='B'; → Falla,

DELETE FROM TIPO_SERVICIO WHERE codigo_tipo_serv ='C'; → Procede,

DELETE FROM SERVICIO WHERE id_servicio = 1; → Falla,

DELETE FROM UNI_SERV WHERE DNI = '41597842'; → Procede,

DELETE FROM TIPO_SERVICIO WHERE codigo_tipo_serv ='B' → Falla,

UPDATE SERVICIO SET id_servicio = 5 WHERE codigo_tipo_serv ='E'; → Procede

Pregunta 4

Finalizado

Se puntúa como 0 sobre 1,00

Utilizando el esquema unc_esq_peliculas. Cuantos empleados hay cuyo sueldo supera 6.000 y cuyo apellido no contenga la letra A.

Tiempo estimado 20 min

- a. 6696
- b. 6710
- c. Ninguna de las opciones es correcta
- d. 6102
- e. 6987

Respuesta correcta

SQL que lo resuelve:

```
SELECT count(*)
FROM unc_esq_peliculas.empleado
WHERE apellido not like '%A%' AND sueldo > 6000;
```

Si se tomaba:

```
SELECT count(*)
FROM unc_esq_peliculas.empleado
WHERE apellido not like '%A%' AND apellido not like '%a%' AND sueldo > 6000;
```

entonces se podía dar que ninguna de las opciones fuera correcta

La respuesta correcta es:
6696

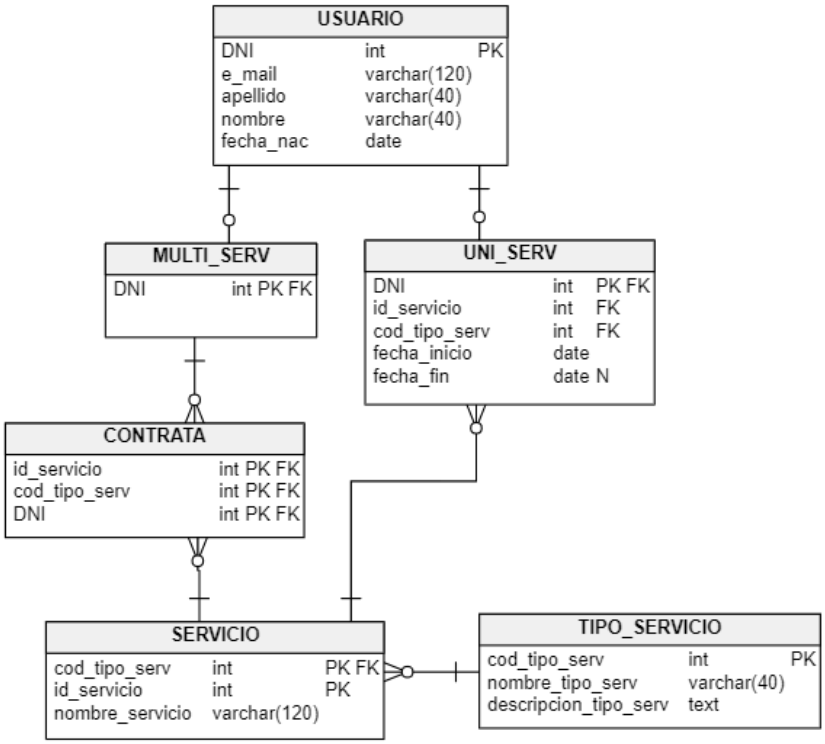
Pregunta 5

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 25 min

Considere el siguiente esquema (script creación solo de tablas) correspondiente a un sistema de contratación de servicios, cuyo diagrama en Vertabelo es el siguiente:



Considere las siguientes sentencias SQL de declaración de claves primarias (PK) y claves extranjeras (FK) sobre el esquema dado:

```
-- Primary keys
```

```
-- Primary keys
ALTER TABLE MULTI_SERV ADD CONSTRAINT PK_MULTI_SERV PRIMARY KEY (DNI);
ALTER TABLE TIPO_SERVICIO ADD CONSTRAINT PK_TIPO_SERVICIO PRIMARY KEY (cod_tipo_serv);
ALTER TABLE UNI_SERV ADD CONSTRAINT PK_UNI_SERV PRIMARY KEY (DNI);
ALTER TABLE USUARIO ADD CONSTRAINT PK_USUARIO PRIMARY KEY (DNI);

-- foreign keys
ALTER TABLE CONTRATA ADD CONSTRAINT FK_CONTRATA_MULTI_SERV
    FOREIGN KEY (DNI)
    REFERENCES MULTI_SERV (DNI);

ALTER TABLE MULTI_SERV ADD CONSTRAINT FK_MULTI_SERV_USUARIO
    FOREIGN KEY (DNI)
    REFERENCES USUARIO (DNI)
        ON DELETE  CASCADE
    ON UPDATE  CASCADE;

ALTER TABLE SERVICIO ADD CONSTRAINT FK_SERVICIO_TIPO_SERVICIO
    FOREIGN KEY (cod_tipo_serv)
    REFERENCES TIPO_SERVICIO (cod_tipo_serv);

ALTER TABLE UNI_SERV ADD CONSTRAINT FK_UNI_SERV_SERVICIO
    FOREIGN KEY (id_servicio, cod_tipo_serv)
    REFERENCES SERVICIO (id_servicio, cod_tipo_serv);
```

Escriba las sentencias sentencias SQL de **declaración de claves primarias (PK), claves alterativas (AK) y claves extranjeras (FK)** que faltan teniendo en cuenta los siguiente:
a) Deben definirse las acciones referenciales para que cada vez que se elimine o modifique (PK) de usuario debe

hacerse también de las tablas subtipo en la jerarquía.
b) El nombre del tipo de servicio debe ser único.

```
-- Primary keys
ALTER TABLE MULTI_SERV ADD CONSTRAINT PK_MULTI_SERV PRIMARY KEY (DNI);
ALTER TABLE TIPO_SERVICIO ADD CONSTRAINT PK_TIPO_SERVICIO PRIMARY KEY (cod_tipo_serv);
ALTER TABLE UNI_SERV ADD CONSTRAINT PK_UNI_SERV PRIMARY KEY (DNI);
ALTER TABLE USUARIO ADD CONSTRAINT PK_USUARIO PRIMARY KEY (DNI);
-- Resuelto
ALTER TABLE CONTRATA ADD CONSTRAINT PK_CONTRATA PRIMARY KEY (id_servicio, cod_tipo_serv, DNI);
ALTER TABLE SERVICIO ADD CONSTRAINT PK_SERVICIO (id_servicio, cod_tipo_serv);

-- Foreign keys
ALTER TABLE CONTRATA ADD CONSTRAINT FK_CONTRATA_MULTI_SERV
    FOREIGN KEY (DNI)
    REFERENCES MULTI_SERV (DNI);
-- Inicio resuelto
ALTER TABLE UNI_SERV ADD CONSTRAINT FK_CONTRATA_MULTI_SERV
    FOREIGN KEY (DNI)
    REFERENCES MULTI_SERV (DNI)
    ON UPDATE CASCADE
    ON DELETE CASCADE;
-- Fin resuelto

ALTER TABLE MULTI_SERV ADD CONSTRAINT FK_MULTI_SERV_USUARIO
    FOREIGN KEY (DNI)
    REFERENCES USUARIO (DNI)
    ON DELETE  CASCADE
    ON UPDATE  CASCADE;

ALTER TABLE SERVICIO ADD CONSTRAINT FK_SERVICIO_TIPO_SERVICIO
    FOREIGN KEY (cod_tipo_serv)
    REFERENCES TIPO_SERVICIO (cod_tipo_serv);

ALTER TABLE UNI_SERV ADD CONSTRAINT FK_UNI_SERV_SERVICIO
    FOREIGN KEY (id_servicio, cod_tipo_serv)
    REFERENCES SERVICIO (id_servicio, cod_tipo_serv);
-- Inicio resuelto
ALTER TABLE UNI_SERV ADD CONSTRAINT FK_UNI_SERV_USUARIO
    FOREIGN KEY (DNI)
    REFERENCES USUARIO (DNI);
-- Fin resuelto

/* Pregunta 5

Escriba las sentencias sentencias SQL de declaración de claves primarias (PK), claves alterativas (AK)
y claves extranjeras (FK) que faltan teniendo en cuenta los siguiente:

a) Deben definirse las acciones referenciales para que cada vez que se elimine o modifique (PK)
de usuario debe hacerse también de las tablas subtipo en la jerarquía.

b) El nombre del tipo de servicio debe ser único.
CONSTRAINT AK_TIPO_SERVICIO UNIQUE (nombre_tipo_serv) NOT DEFERRABLE  INITIALLY IMMEDIATE */
```

```
--Primary Keys
```

```
ALTER TABLE CONTRATA ADD CONSTRAINT PK_CONTRATA PRIMARY KEY (id_servicio,cod_tipo_serv,DNI);
```

```
ALTER TABLE SERVICIO ADD CONSTRAINT PK_SERVICIO PRIMARY KEY (id_servicio,cod_tipo_serv);
ALTER TABLE MULTI_SERV ADD CONSTRAINT FK_MULTI_SERV_USUARIO
    FOREIGN KEY (DNI)
    REFERENCES USUARIO (DNI)
    ON DELETE  CASCADE
    ON UPDATE  CASCADE;
```

```
ALTER TABLE TIPO_SERVICIO ADD CONSTRAINT AK_TIPO_SERVICIO UNIQUE (nombre_tipo_serv);
```

```
-- Foreign Keys

ALTER TABLE UNI_SERV ADD CONSTRAINT FK_UNI_SERV_USUARIO
    FOREIGN KEY (DNI)
    REFERENCES USUARIO (DNI)
    ON DELETE  CASCADE
    ON UPDATE  CASCADE;
```

```
ALTER TABLE CONTRATA ADD CONSTRAINT FK_CONTRATA_SERVICIO
    FOREIGN KEY (id_servicio, cod_tipo_serv)
    REFERENCES SERVICIO (id_servicio, cod_tipo_serv);
```

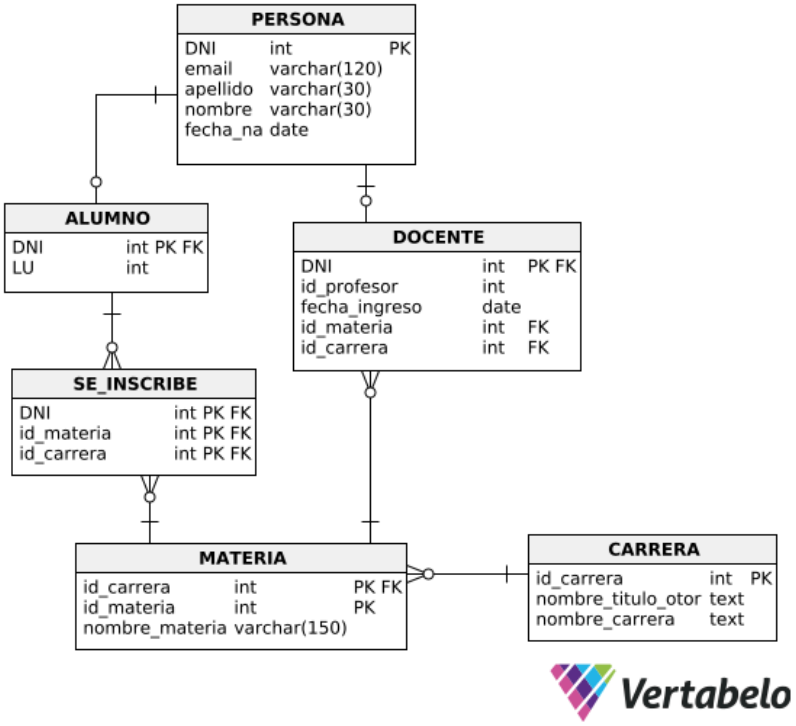
Pregunta 6

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 45 min

Dado el siguiente esquema - Script de creación de tablas



1) De la sentencia declarativa mas restrictiva que controle lo siguiente:

- **Toda materia donde hay alumnos inscriptos, debe haber un docente**

2) En caso de que no pueda ser implementado en PostgreSQL declarativamente, de la solución procedural mas eficiente.

```
/* Pregunta 6

1) De la sentencia declarativa mas restrictiva que controle lo siguiente:
- Toda materia donde hay alumnos inscriptos, debe haber un docente */
-- CHECK Global
CREATE ASSERTION materia_inscriptos_con_docente
CHECK (NOT EXISTS (
       SELECT 1
       FROM SE_INSCRIBE si
       WHERE NOT EXISTS (
            SELECT 1
            FROM DOCENTE d
            WHERE si.id_materia = d.id_materia
)));

-- 2) En caso de que no pueda ser implementado en PostgreSQL declarativamente, de la soluci
-- mas eficiente.
CREATE FUNCTION fn_verificar_materia_con_docente()
RETURNS TRIGGER AS $$
BEGIN
    IF NOT EXISTS (
        SELECT 1
        FROM DOCENTE d
        WHERE NEW.id_materia = d.id_materia)
        THEN RAISE EXCEPTION 'ERROR! La materia a la que se quiere inscribir no tiene docent
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE 'plpqsql';
```

1)

```
CREATE ASSERTION ck_materia_con_inscriptos_tiene_docente
      CHECK ( NOT EXISTS( SELECT 1
                  FROM se_inscribe i
                  WHERE NOT EXISTS (SELECT 1
                              FROM docente d
                              WHERE d.id_materia = i.id_materia
                              AND d.id_carrera = i.id_carrera)));
```

```
--Al ser atributos no nulos, se puede plantear como un NOT IN también
CREATE ASSERTION ck_materia_con_inscriptos_tiene_docente
      CHECK ( NOT EXISTS( SELECT 1
                  FROM se_inscribe
                  WHERE (id_materia, id_carrera)
                         NOT IN (SELECT id_materia, id_carrera
                                     FROM docente)));
```

Nota: Tener en cuenta lo siguiente : Si la materia existe en Se_Inscribe, debe estar presente en Docente. Pero, si una Materia está en Docente , no implica que necesariamente deba haber inscriptos en Se_incribe, y además puede ser dictada por màs de un docente (repetida en la tabla Docente), por esto es que cuando se elimina o se actualiza la materia en un Docente, solo hay que elevar la Exception solo si existe alguna inscripción a la materia "vieja" y si a la vez no queda otro docente dictando dicha materia

```
CREATE OR REPLACE FUNCTION FN_verificar_docente()
RETURNS TRIGGER AS $$
BEGIN
     IF NOT EXISTS (SELECT 1
               FROM DOCENTE D
               WHERE D.id_materia = NEW.id_materia
               AND D.id_carrera = NEW.id_carrera
               ) THEN
               RAISE EXCEPTION 'No hay un docente asignado a la materia % en la carrera %', NEW.id_materia,
NEW.id_carrera;
      END IF;
      RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER TRG_B_INS_UPD_SeInscribe
BEFORE INSERT OR UPDATE ON SE_INSCRIBE
FOR EACH ROW
EXECUTE FUNCTION FN_verificar_docente();
```

```
CREATE OR REPLACE FUNCTION FN_verificar_inscripcion()
RETURNS TRIGGER AS $$
BEGIN
     IF EXISTS (SELECT 1
               FROM se_inscribe SI
               WHERE SI.id_materia = OLD.id_materia
               AND SI.id_carrera = OLD.id_carrera
               )
     AND NOT EXISTS (SELECT 1
               FROM Docente D
               WHERE D.id_materia = OLD.id_materia
               AND D.id_carrera = OLD.id_carrera
               )
     THEN
               RAISE EXCEPTION 'No hay un docente asignado a la materia % en la carrera %', NEW.id_materia,
NEW.id_carrera;
      END IF;
      RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER TRG_B_DEL_UPD_Docente
AFTER DELETE OR UPDATE ON Docente
FOR EACH ROW
EXECUTE FUNCTION FN_verificar_inscripcion();
```

Actividad previa

◀ Práctico 9 - SQL Avanzado

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

Facultad de Ciencias Exactas – UNICEN

Contacto administradores plataforma: E-mail moodle@exa.unicen.edu.ar – Tel. +54 0249 4385650 int. 2098