Es **sumamente importante** que para ***cada esquema*** de BD de ***cada uno de los ejercicios*** realice los siguientes pasos y en el orden que le planteamos (le ayudará mucho a adquirir una metodología para resolverlos).

1. **Primero** completa la **TABLA DE RESTRICCIONES** que se encuentra debajo con lo siguiente:
2. **clasificá** según corresponda cada una de las *restricciones de integridad enunciadas* (de dominio o atributo, de tupla, de tabla o general).
3. **identificá** el ***recurso declarativo más apropiado*** para implementar la restricción de acuerdo al **estándar SQL3**. (**DOMAIN, CHECK o ASSERTION**)
4. Segundo plantea la/las sentencia/s en SQL3 estándar para realizar cada uno de los controles solicitados (ALTER TABLE o CREATE ASSERTION).
5. **Tercero**, crea cada esquema correspondiente y plantea las sentencias de alteración de tablas de las restricciones **que puedan ser soportadas** en ***PostgreSQL***.

| **Restricción** | **Tabla/s** | **Atributo/s** | **Tipo de restricción** |
| --- | --- | --- | --- |
| No puede haber voluntarios de más de 70 años. (V) | voluntario | fecha\_nacimiento | CHECK atributo (involucra solo una celda) |
| Ningún voluntario puede aportar más horas que las de su coordinador. (V) | voluntario | horas\_aportadas, id\_voluntario,  id\_coordinador | CHECK tabla (màs de un registro de la tabla) |
| Las horas aportadas por los voluntarios deben estar dentro de los valores máximos y mínimos consignados en la tarea. (V) | voluntario, tarea | horas\_aportadas, nro\_voluntario, min\_horas\_max\_horas, id\_tarea | CHECK general (más de una tabla involucrada) |
| Todos los voluntarios deben realizar la misma tarea que su coordinador. (V) | voluntario | nro\_voluntario, id\_tarea, id\_coordinador | CHECK registro o tupla (más de una celda) |
| Los voluntarios no pueden cambiar de institución más de tres veces en el año. (V) | histórico | nro\_voluntario, id\_insitucion, fecha\_inicio, fecha\_fin | CHECK registro o tupla (más de una celda) |

**TABLA DE RESTRICCIONES**

**NOTA**: Para resolver los ejercicios que involucren a las **BD de Películas y BD Voluntarios** es necesario crear el esquema de cada una de ellas en tu propio esquema, dado que dichos esquemas son de sólo lectura, entonces no permiten su modificación.

### **Ejercicio 1**

Considere las siguientes restricciones que debe definir sobre el **esquema de la BD de Voluntarios**:

1. No puede haber voluntarios de más de 70 años.

ALTER TABLE esq\_voluntarios\_voluntario ADD CONSTRAINT ck\_voluntario\_edad CHECK (*age*(fecha\_nacimiento)<='70 YEAR');

1. Ningún voluntario puede aportar más horas que las de su coordinador.

ALTER TABLE esq\_voluntarios\_voluntario ADD CONSTRAINT ck\_horas CHECK (NOT *EXISTS*(SELECT 1

FROM esq\_voluntarios\_voluntario v INNER JOIN esq\_voluntarios\_voluntario c ON v.id\_coordinador = c.nro\_voluntario

WHERE v.horas\_aportadas>c.horas\_aportadas));

1. Las horas aportadas por los voluntarios deben estar dentro de los valores máximos y mínimos consignados en la tarea.

CREATE ASSERTION ck\_horas\_aportadas\_entre\_max\_min CHECK

(NOT EXISTS (SELECT \* FROM esq\_voluntarios\_voluntario v INNER JOIN esq\_voluntarios\_tarea t ON v.id\_tarea=t.id\_tarea

WHERE (horas\_aportadas>max\_horas OR horas\_aportadas<min\_horas));

1. Todos los voluntarios deben realizar la misma tarea que su coordinador.

CHECK TABLA (Tabla voluntario. Atributos id\_coordinador, id\_tarea)

ALTER TABLE esq\_voluntarios\_voluntario ADD CONSTRAINT (NOT EXISTS(SELECT 1 FROM esq\_voluntarios\_voluntario v INNER JOIN esq\_voluntarios\_voluntario c ON v.nro\_voluntario=c.id\_coordinador

WHERE v.id\_tarea!=c.id\_tarea));

1. Los voluntarios no pueden cambiar de institución más de tres veces en el año.

CHECK TABLA (Tabla: historico. Atributos: nro\_voluntario, fecha\_inicio.)

ALTER TABLE esq\_voluntarios\_historico ck\_cambio\_insitucion\_menor\_o\_igual\_3 CHECK (NOT EXISTS(SELECT (EXTRACT(YEAR FROM h.fecha\_inicio)), nro\_voluntario FROM esq\_voluntarios\_historico h

GROUP BY (EXTRACT(YEAR FROM h.fecha\_inicio)), nro\_voluntario HAVING COUNT(\*)>3));

1. En el histórico, la fecha de inicio debe ser siempre menor que la fecha de finalización.

CHECK TABLA (Tabla historico. Atributos: fecha\_incio, fecha\_fin

### ALTER TABLE esq\_voluntarios\_historico ck\_fecha\_inicio\_menor\_fecha\_fin CHECK(NOT EXISTS

### (SELECT fecha\_inicio, fecha\_fin FROM esq\_voluntarios\_historico WHERE fecha\_inicio>fecha\_fin));

### 

### **Ejercicio 2**

Considere las siguientes restricciones que debe definir sobre el esquema de la **BD de Películas**:

1. Para cada tarea el sueldo máximo debe ser mayor que el sueldo mínimo.

CHECK TABLA (Tabla tarea. ATributos tarea, sueldo max, sueldo min).

ALTER TABLE esq\_peliculas\_tarea ADD CONSTRAINT ck\_sueldo\_tarea\_mayor\_sueldo\_minimo CHECK

(NOT *EXISTS*(SELECT id\_tarea FROM esq\_peliculas\_tarea GROUP BY id\_tarea HAVING sueldo\_maximo>sueldo\_minimo));

1. No puede haber más de 70 empleados en cada departamento.

CHECK GENERAL (Tablas empleado y departamento. Atributo nombre, empleados)

CREATE ASSERTION ck\_cant\_empleados\_menor\_70 CHECK (NOT EXISTS (SELECT d.nombre, COUNT(id\_empleado) AS CANT\_EMPLEADOS FROM esq\_peliculas\_empleado e INNER JOIN esq\_peliculas\_departamento d ON (e.id\_departamento=d.id\_departamento

AND e.id\_distribuidor=d.id\_distribuidor) GROUP BY d.nombre HAVING COUNT(\*) >70));

1. Los empleados deben tener jefes que pertenezcan al mismo departamento.

CHECK TABLA (Tabla departamento. Atributos jefe\_departamento id\_empleado)

ALTER TABLE esq\_peliculas\_departamento ADD CONSTRAINT ck\_empleados\_jefes\_mismo\_departamento CHECK

(NOT *EXISTS* (SELECT 1 FROM esq\_peliculas\_empleado e INNER JOIN esq\_peliculas\_empleado j ON e.id\_jefe = j.id\_empleado

WHERE e.id\_departamento != j.id\_departamento AND e.id\_distribuidor != j.id\_distribuidor));

1. Todas las entregas, tienen que ser de películas de un mismo idioma.

CHECK GENERAL (TABLAS renglo\_entrega, pelicula. Atributo: nro\_entrega, idioma)

CREATE ASSERTION ck\_entregas\_mismo\_idioma (NOT EXISTS

(SELECT 1 FROM esq\_peliculas\_renglon\_entrega e INNER JOIN esq\_peliculas\_pelicula p ON

e.codigo\_pelicula=p.codigo\_pelicula GROUP BY idioma HAVING COUNT (\*)!=1));

1. No pueden haber más de 10 empresas productoras por ciudad.

Use tres tablas porque si uso el nombre de la empresa puede ser que haya mas de una (en otra ciudad).

CREATE ASSERTION ck\_hasta\_10\_empresas\_productoras\_x\_ciudad (NOT EXISTS

(SELECT 1 FROM esq\_peliculas\_pelicula p INNER JOIN esq\_peliculas\_empresa\_productora ep ON

(ep.codigo\_productora=p.codigo\_productora) INNER JOIN esq\_peliculas\_ciudad c on c.id\_ciudad = ep.id\_ciudad

GROUP BY c.id\_ciudad, ep.codigo\_productora HAVING COUNT(\*)<10));

La subconsulta completa es:

SELECT ep.codigo\_productora, c.nombre\_ciudad, *COUNT*(\*) AS NUM\_PROD FROM esq\_peliculas\_pelicula p INNER JOIN esq\_peliculas\_empresa\_productora ep ON

(ep.codigo\_productora=p.codigo\_productora) INNER JOIN esq\_peliculas\_ciudad c on c.id\_ciudad = ep.id\_ciudad

GROUP BY c.id\_ciudad, ep.codigo\_productora HAVING *COUNT*(\*)<10;

1. Para cada película, si el formato es 8mm, el idioma tiene que ser francés.

ALTER TABLE unc\_esq\_peliculas.pelicula ADD CONSTRAINT ck\_formato\_idioma

CHECK((formato = '8mm' AND idioma = 'Frances') OR formato != '8mm');

1. El teléfono de los distribuidores Nacionales debe tener la misma característica que la de su distribuidor mayorista.

CREATE ASSERTION ck\_telef\_distrib CHECK(NOT EXISTS(

SELECT 1 FROM distribuidor d JOIN nacional n ON d.id\_distribuidor = n.id\_distribuidor JOIN distribuidor di

ON n.id\_distrib\_mayorista = di.id\_distribuidor

WHERE d.LEFT(telefono, 3) != di.LEFT(telefono, 3)));

### **Ejercicio 3**

Para el esquema de la figura que corresponde al ejercicio 1 del Trabajo Práctico 5 cuyo script de creación de tablas lo podes descargar de [aquí](https://drive.google.com/file/d/1FhL5RggPDG1Rr8VJ-sTnfjtSZPAKZeeN/view?usp=sharing)

| Restriccion | Tabla | Atributos | Chequeo |
| --- | --- | --- | --- |
| Controlar que las nacionalidades sean 'Argentina' 'Español' 'Inglés' 'Alemán' o 'Chilena'. | Artículo | nacionalidad | Chequeo de atributo |
| Para las fechas de publicaciones se debe considerar que sean fechas posteriores o iguales al 2010. | Artículo | fecha\_publicacion | Chequeo de atributo |
| Cada palabra clave puede aparecer como máximo en 5 artículos. | Contiene | cod\_palabra, idioma | Chequeo de tabla |
| Sólo los autores argentinos pueden publicar artículos que contengan más de 10 palabras claves, pero con un tope de 15 palabras, el resto de los autores sólo pueden publicar artículos que contengan hasta 10 palabras claves. | Articulo, Contiene | id\_articulo, cod\_palabra, nacionalidad | Chequeo General  (dos tablas) |

1. Controlar que las nacionalidades sean 'Argentina' 'Español' 'Inglés' 'Alemán' o 'Chilena'.

ALTER TABLE p5p1e1\_articulo ADD CONSTRAINT ck\_articulo\_nacionalidad

CHECK (nacionalidad IN( 'Argentina', 'Español', 'Inglés', 'Alemán', 'Chilena'));

Si en la tabla existe alguna fila que contenga otra nacionalidad que no sea una de estas, da error.

| Chequeo atributo porque es solo una celda  Si tuviera que chequear más de una celda, el registro.  Si están involucradas varias tablas es general |
| --- |

1. Para las fechas de publicaciones se debe considerar que sean fechas posteriores o iguales al 2010.

ALTER TABLE p5p1e1\_articulo ADD CONSTRAINT ck\_articulo\_fecha\_publicacion CHECK (*EXTRACT*(YEAR FROM fecha\_publicacion)>=2010);

1. Cada palabra clave puede aparecer como máximo en 5 artículos.

--(necesito solo la tabla contiene)

--Dame las palabras que tienen mas de 5 articulos

--no se puede utilizar un subquery en un check (utilizar manera procedural)

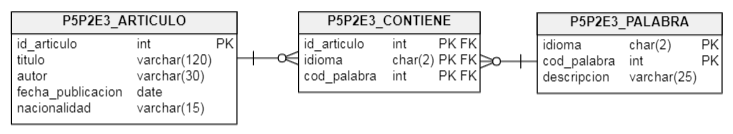
--De manera declarativa no anda en Postgrest:

ALTER TABLE p5p1e1\_contiene ADD CONSTRAINT ck\_contiene\_cod\_palabra CHECK

(NOT *EXISTS* (SELECT cod\_palabra, idioma FROM p5p1e1\_contiene GROUP BY cod\_palabra, idioma HAVING *COUNT*(\*)>5));

1. Sólo los autores argentinos pueden publicar artículos que contengan más de 10 palabras claves, pero con un tope de 15 palabras, el resto de los autores sólo pueden publicar artículos que contengan hasta 10 palabras claves.

**Aclaración:** Si agrupo por id\_articulo cuento las palabras, y si agrupo por idioma y cod\_palabra cuento los artículos)

****

CREATE ASSERTION CK\_CANTIDAD\_PALABRAS CHECK (NOT EXISTS(SELECT 1 FROM p5p2e3\_articulo WHERE (nacionalidad LIKE 'Argentina' AND

id\_articulo IN (SELECT id\_articulo FROM p5p2e3\_contiene GROUP BY id\_articulo HAVING COUNT(\*)>15)) OR

(nacionalidad NOT LIKE 'Argentina' AND id\_articulo IN (SELECT id\_articulo FROM p5p2e3\_contiene GROUP BY id\_articulo

HAVING COUNT(\*)>10) )));

*Las soluciones las podes encontrar* [*aquí*](https://drive.google.com/file/d/1UsMSpjJIwYBIKAyCtUAJ85pbQJZhrxEh/view?usp=sharing)

### **Ejercicio 4**

Para el esquema de la figura cuyo script de creación de tablas lo podes descargar de [aquí](https://drive.google.com/file/d/1AelMX-SfADDU_apCuYNsD6_ET-TV8JkT/view?usp=sharing)

1. La modalidad de la imagen médica puede tomar los siguientes valores RADIOLOGIA CONVENCIONAL, FLUOROSCOPIA, ESTUDIOS RADIOGRAFICOS CON FLUOROSCOPIA, MAMOGRAFIA, SONOGRAFIA.

CHECK ATRIBUTO (Tabla: imagen\_medica. Atributo: modalidad)

ALTER TABLE p5p2e4\_imagen\_medica ADD CONSTRAINT ck\_imagen\_medica\_modalidad CHECK

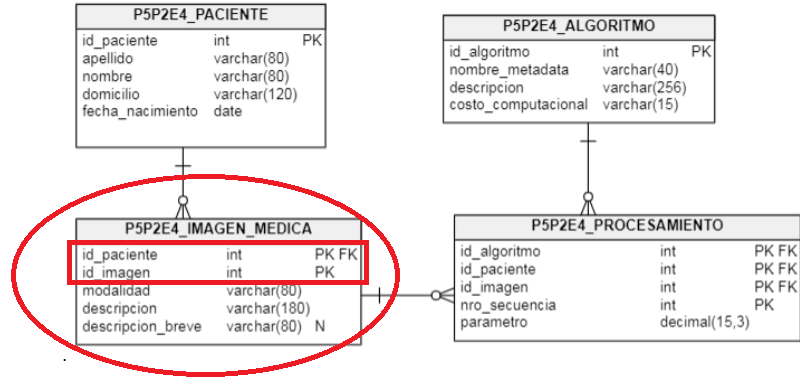
(modalidad IN('RADIOLOGIA CONVENCIONAL','FLUOROSCOPIA', 'ESTUDIOS RADRIOGRAFICOS CON FLUOROSCOPIA', 'MAMOGRAFIA', 'SONOGRAFIA'));

1. Cada imagen no debe tener más de 5 procesamientos.

CHECK TABLA (Tabla: procesamiento. )

ALTER TABLE p5p2e4\_procesamiento ADD CONSTRAINT ck\_procesamiento\_id\_imagen CHECK

(NOT *EXISTS* (SELECT id\_imagen, id\_paciente FROM p5p2e4\_procesamiento GROUP BY id\_imagen, id\_paciente HAVING *count*(\*)>5));



1. Agregue dos atributos de tipo fecha a las tablas Imagen\_medica y Procesamiento, una indica la fecha de la imagen y la otra la fecha de procesamiento de la imagen y controle que la segunda no sea menor que la primera.

CHECK GENERAL (Tablas: imagen\_medica, procesamiento. Atributos: fecha\_imagen, fecha\_procesamiento)

Agrego las columnas:

ALTER TABLE p5p2e4\_imagen\_medica ADD COLUMN fecha\_imagen date;

ALTER TABLE p5p2e4\_procesamiento ADD COLUMN fecha\_procesamiento date;

Luego hago una subconsulta con lo contrario:

SELECT \* FROM p5p2e4\_imagen\_medica i INNER JOIN p5p2e4\_procesamiento p on (i.id\_imagen = p.id\_imagen AND i.id\_paciente=p.id\_paciente)

WHERE fecha\_imagen>fecha\_procesamiento;

Y por último armo el ASSERTION con el NOT EXISTS:

CREATE ASSERTION ck\_imagen\_medica\_procesamiento\_fecha CHECK (NOT EXISTS

(SELECT \* FROM p5p2e4\_imagen\_medica i INNER JOIN p5p2e4\_procesamiento p on (i.id\_imagen = p.id\_imagen AND i.id\_paciente=p.id\_paciente)

WHERE fecha\_imagen>fecha\_procesamiento));

1. Cada paciente sólo puede realizar dos FLUOROSCOPIA anuales.

CHECK TABLA

--Realizo una subconsulta con lo contrario:

SELECT id\_paciente, modalidad, *EXTRACT*(YEAR FROM fecha\_imagen) FROM p5p2e4\_imagen\_medica WHERE modalidad='FLUOROSCOPIA' GROUP BY id\_paciente,modalidad, *EXTRACT*(YEAR FROM fecha\_imagen) HAVING *COUNT*(\*)>2;

--Luego hago el chequeo con el NOT EXISTS:

ALTER TABLE p5p2e4\_imagen\_medica ADD CONSTRAINT ck\_imagen\_medica\_modalidad\_fluoroscopia CHECK (NOT *EXISTS* (SELECT id\_paciente, modalidad, *EXTRACT*(YEAR FROM fecha\_imagen) FROM p5p2e4\_imagen\_medica WHERE modalidad='FLUOROSCOPIA' GROUP BY id\_paciente,modalidad, *EXTRACT*(YEAR FROM fecha\_imagen) HAVING *COUNT*(\*)>2));

1. No se pueden aplicar algoritmos de costo computacional “O(n)” a imágenes de FLUOROSCOPIA

CHECK GENERAL (Tablas: imagen\_medica, procesamiento, algoritmo)

CREATE ASSERTION ck\_imagen\_medica\_procesamiento\_algoritmo\_costo\_fluoroscopia\_on CHECK (NOT EXISTS

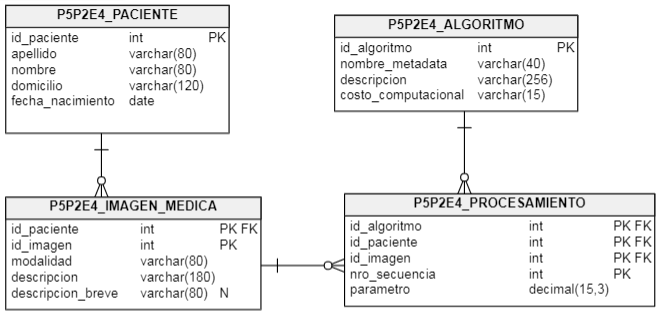
(SELECT m.id\_imagen FROM p5p2e4\_imagen\_medica m INNER JOIN p5p2e4\_procesamiento p ON (m.id\_imagen=p.id\_imagen AND m.id\_paciente=p.id\_paciente)

INNER JOIN p5p2e4\_algoritmo a ON (p.id\_algoritmo=a.id\_algoritmo) WHERE (modalidad='FLUOROSCOPIA' AND costo\_computacional='O(n)')));

--Realizo la subconsulta antes:

SELECT m.id\_imagen FROM p5p2e4\_imagen\_medica m INNER JOIN p5p2e4\_procesamiento p ON (m.id\_imagen=p.id\_imagen AND m.id\_paciente=p.id\_paciente)

INNER JOIN p5p2e4\_algoritmo a ON (p.id\_algoritmo=a.id\_algoritmo) WHERE (modalidad='FLUOROSCOPIA' AND costo\_computacional='O(n)');

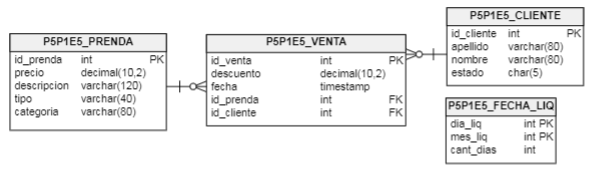
.

### 

Las soluciones las podes encontrar [aquí](https://drive.google.com/file/d/1B8hLSmTM5qr9pVcct3-wY5vKTAMvC8Uo/view?usp=sharing)

### **Ejercicio 5**

Para el esquema de la figura cuyo script de creación de tablas lo podes descargar de [aquí](https://drive.google.com/file/d/1Aa2Kl7OiZ9SlWdibBRxSlxKlJ7nF40Kh/view?usp=sharing)



1. Los descuentos en las ventas son porcentajes y deben estar entre 0 y 100.

CHECK ATRIBUTO (Tabla: venta. Atributo: descuento)

ALTER TABLE p5p2e5\_venta ADD CONSTRAINT ck\_venta\_descuento\_0\_100 CHECK (descuento BETWEEN 0 AND 100);

1. Los descuentos realizados en fechas de liquidación deben superar el 30%.

CHECK GENERAL (Tablas: venta, fecha\_liq. Atributos: id\_venta, descuento, fecha, dia\_liq, mes\_liq

1. Las liquidaciones de Julio y Diciembre no deben superar los 5 días.

CHECK TUPLA (Tabla: fecha\_liq. Atributo: mes\_liq, cant\_dias)

ALTER TABLE p5p2e5\_fecha\_liq ADD CONSTRAINT ck\_fecha\_liq\_mes\_7\_12\_dias\_menor\_5 CHECK

(NOT *EXISTS*(SELECT \* FROM p5p2e5\_fecha\_liq WHERE ((mes\_liq=7 OR mes\_liq=12) AND cant\_dias>5)));

1. Las prendas de categoría ‘oferta’ no tienen descuentos.

CHECK GENERAL (Tablas: prenda, venta. Atributos: id\_prenda, categoria, descuento)

CREATE ASSERTION ck\_venta\_prenda\_categoria\_oferta\_sin\_descuento CHECK

(NOT EXISTS (SELECT \* FROM p5p2e5\_venta v INNER JOIN p5p2e5\_prenda p ON v.id\_prenda = p.id\_prenda WHERE

(categoria='oferta' AND descuento!=0)));

### 