Optimización de consultas

Ejercicio 1

Con el comando EXPLAIN explique la heurística que utiliza para el *query plan* el optimizador de PostgreSQL en cada una de las sentencias:

a) *Listado de las entregas que poseen las película de terror* (esquema Películas)

```
SELECT p.titulo, e.nro_entrega
FROM pelicula p, renglon_entrega re, entrega e
WHERE p.codigo_pelicula = re. codigo_pelicula
AND re.nro_entrega = e. nro_entrega
AND genero = 'TERROR';
```

b) *Listado de los datos de contacto (nombre, apellido, email y teléfono) de todos los voluntarios que hayan desarrollado tareas de hasta 5000 hs (max_horas - min_horas) y que las hayan finalizado antes del 01/01/2024* (esquema Voluntario).

```
SELECT V.nombre, V.apellido, V.e_mail, V.telefono
FROM Voluntario V
WHERE V.nro_voluntario IN (SELECT H.nro_voluntario
        FROM Historico H
        WHERE H.fecha_fin < to_date('1998-07-24', 'yyyy-mm-dd') AND
        H.id_tarea IN (SELECT T.id_tarea
                FROM Tarea T
                WHERE (T.max_horas - T.min_horas) <= 5000));
```

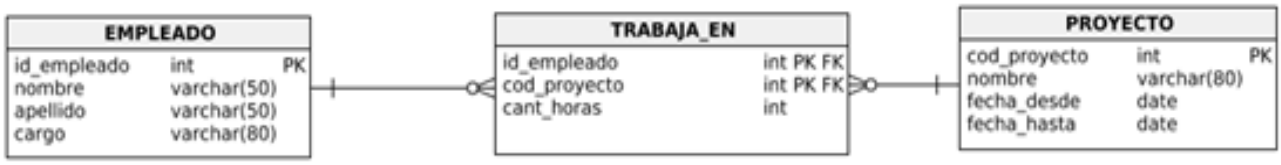
Ejercicio 2

Dada la siguiente consulta sobre el esquema de voluntarios, analice su plan de ejecución. ¿Es posible mejorarla?

```
SELECT * FROM tarea
WHERE id_tarea IN (
        SELECT t.id_tarea
        FROM tarea t inner join voluntario v  on (t.id_tarea = v.id_tarea)
        GROUP BY t.id_tarea
        HAVING COUNT(v.nro_voluntario) > 5);
```

Ejercicio 3

Considere el siguiente esquema de una base de datos de empleados de grupos de investigación:



Suponga la existencia de las siguientes tuplas en las respectivas tablas:

EMPLEADO (1, 'Juan', 'Perez', 'investigador'); (2, 'Rosa', 'Gomez', 'investigador'); (3, 'Bruno', 'Fernandez', 'becario'); (4, 'Ignacio', 'Rodriguez', 'becario'); (5, 'Alejandro', 'Perez', 'investigador'); (6, 'Sebastian', 'Solano', 'investigador');

Optimización de consultas

PROYECTO (1, 'ROBOTICS', '2018-01-01', '2020-01-01'); (2, 'IA', '2017-01-01', '2020-01-01'); (3, 'IMAGE', '2015-01-01', '2017-01-01');

TRABAJA_EN (1, 1, 40); (2, 1, 40); (3, 1, 20); (4, 1, 20); (5, 1, 40); (6, 1, 40); (1, 2, 40); (3, 2, 20); (6, 2, 20); (3, 3, 10); (6, 3, 10);

Con el comando EXPLAIN ANALYZE explique la heurística que utiliza para el *query plan* del optimizador de PostgreSQL en cada una de las sentencias siguientes:

a) Listar los datos de los empleados investigadores que trabajan en algún proyecto

```
SELECT  DISTINCT E.*
FROM  EMPLEADO E  ,
      TRABAJA_EN T,
      PROYECTO P
WHERE E.id_empleado = T.id_empleado
AND P.cod_proyecto = T.cod_proyecto
AND E.cargo = 'investigador';
```

a.1) ¿Es necesario el ensamble con Empleado?
a.2) ¿Es necesario el ensamble con Proyecto?
a.3) ¿Qué implicancias tiene la inclusión del “distinct”?

b) Listar los datos de los proyectos de IA que tienen empleados trabajando

```
SELECT DISTINCT P.*
FROM  TRABAJA_EN T,
      PROYECTO P
WHERE P.cod_proyecto = T.cod_proyecto
AND P.nombre = 'IA';
```

b.1) Analizar para la restricción de clave primaria establecida como (id_empleado, cod_proyecto)
b.2) Analizar para la restricción de clave primaria establecida como (cod_proyecto, id_empleado)

Ejercicio 4

Para cada una de las siguientes consultas sobre el esquema de Películas:

a) Analizar cada caso y justificar la creación o no de índices considerando que cada consulta es muy frecuente en el contexto planteado. Considere también la creación de índices que sirvan para más de una de las consultas.

b) Conociendo que las opciones de estructuras de índices en PostgresSQL son árboles B+ y Hash, analizar cuál de ellas resulta más eficiente en cada situación y por qué.

b.1) SELECT *
FROM entrega

Optimización de consultas

```
        WHERE MONTHS_BETWEEN
        (SYSDATE, fecha_entrega) > 100;

b.2) SELECT *
        FROM distribuidor
        WHERE id_distribuidor > 50;

b.3) SELECT *
        FROM distribuidor
        WHERE tipo = 'N';

b.4) SELECT *
        FROM empleado
        WHERE nombre = 'Juan Lucas'
        AND e_mail = 'Pla@gmail.com';

b.5) SELECT *
        FROM empleado
        WHERE nombre LIKE 'An%'
        AND e_mail = 'Marise@gmail.com';

b.6) SELECT *
        FROM empleado
        WHERE nombre LIKE '%Maria'
        AND e_mail = 'Marise@gmail.com';
```