Parcialito de SQL

1. Considerando los esquemas del dataset de notas visto en clase, resuelva los siguientes puntos escribiendo para cada uno de ellos una única consulta SQL que devuelva el resultado pedido:

   a) Obtener el padrón y apellido de los alumnos con mayor cantidad de materias aprobadas.

   b) Obtener el padrón y apellido de aquellos alumnos que tienen nota en las materias 71.14 y 71.15 y no tienen nota ni en las materias 75.01 ni en 75.15.

   c) Mostrar el padrón, apellido y promedio para aquellos alumnos que tienen nota en más de 3 materias y un promedio de al menos 5.

   d) Para cada nota del alumno más antiguo, mostrar su código de departamento, número de materia y el valor de la nota.

   e) Listar el padrón de aquellos alumnos que, por lo menos, tienen nota en todas las materias que aprobó el alumno de padrón 71000.

   En el Campus en la sección de Talleres encontrará el archivo "base-ejemplo-facultad", que es el dataset que puede utilizar para entender el contenido de la tablas y validar sus respuestas.

2. **BONUS TRACK** *(opcional)*[1] Resuelva el ejercicio 1a del Tema 1 del parcial del primer cuatrimestre de 2019 . Puede encontrar el enunciado en la sección de exámenes del campus, con el nombre de archivo "2019 - 1C - Parcial T1"

Nota: Basta con entregar únicamente las consultas SQL que escribió. No es necesario que incluya las tablas resultado de las consultas en la entrega, aunque si quiere puede hacerlo.

[1] No es obligatorio entregarlo. En caso de hacerlo, no resta nota si está incorrecto aunque puede mejorar la nota del parcialito si está bien resuelto

- Obtener el padrón y apellido de los alumnos con mayor cantidad de materias aprobadas.

```
SELECT alumnos.padron, apellido
FROM alumnos
         JOIN notas ON alumnos.padron = notas.padron
WHERE notas.nota >= 4
GROUP BY alumnos.padron
ORDER BY count(*) DESC
LIMIT 1
```

Técnicamente, esta query devuelve solo un alumno, por más que puede haber varios que tengan el mayor número de materias aprobadas. Para tener en cuenta estos empates, se puede hacer algo como...

```
SELECT padron, apellido
FROM (
         SELECT alumnos.padron,
                apellido,
                rank() OVER (ORDER BY count(*) DESC) AS rnk
         FROM alumnos
                  JOIN notas ON alumnos.padron = notas.
                     padron
         WHERE notas.nota >= 4
         GROUP BY alumnos.padron
     ) subquery
WHERE rnk <= 1
```

(Esta mísma idea se puede utilizar en el resto de los ejercicios donde se piden los máximos, como en el ejercicio **d** donde se necesita encontrar al alumno más antiguo, y este puede ser más de uno, porque se pueden anotar múltiples alumnos en la misma fecha)

- Obtener el padrón y apellido de aquellos alumnos que tienen nota en las materias 71.14 y 71.15 y no tienen nota ni en las materias 75.01 ni en 75.15.

```
SELECT alumnos.padron, apellido
FROM alumnos
         JOIN notas ON alumnos.padron = notas.padron
WHERE ((codigo = 71 AND numero = 14) OR (codigo = 71 AND
    numero = 15))
  AND alumnos.padron NOT IN (
    SELECT alumnos.padron
    FROM alumnos
             JOIN notas ON alumnos.padron = notas.padron
    WHERE ((codigo = 75 AND numero = 01) OR (codigo = 75 AND
        numero = 15))
)
```

- Mostrar el padrón, apellido y promedio para aquellos alumnos que tienen nota en más de 3 materias y un promedio de al menos 5.

```
SELECT alumnos.padron
FROM alumnos
         JOIN notas ON alumnos.padron = notas.padron
GROUP BY alumnos.padron
HAVING (count(alumnos.padron) > 3)
   AND avg(nota) >= 5
```

- Para cada nota del alumno más antiguo, mostrar su código de departamento, número de materia y el valor de la nota.

```
SELECT codigo, numero, nota
FROM notas
WHERE notas.padron IN (
    SELECT padron
    FROM alumnos
    ORDER BY fecha_ingreso ASC
    LIMIT 1
)
```

- Listar el padrón de aquellos alumnos que, por lo menos, tienen nota en todas las materias que aprobó el alumno de padrón 71000.

```
WITH materias AS (
    SELECT codigo, numero
    FROM notas
    WHERE padron = 71000
      AND nota >= 4
)
SELECT padron
FROM (
         SELECT DISTINCT alumnos.padron, codigo, numero
         FROM alumnos
                  JOIN notas ON alumnos.padron = notas.
                     padron
         WHERE (codigo, numero) IN (SELECT codigo, numero
            FROM materias)
         GROUP BY alumnos.padron, notas.codigo, notas.numero
     ) subquery
GROUP BY padron
HAVING count(*) = (SELECT count(*) FROM materias)
```

Para el ejercicio bonus utilizo la expresion `SELECT DISTINCT ON`[1] para quedarme con la fila con la máxima prioridad. Para definir cual es la máxima prioridad, hago uso de `CASE`: al cruzarme a un mercado, le agregro prioridad 3 a la fila, al cruzarme un local le agrego prioridad 2, y así.

```
SELECT DISTINCT ON (cod_prop) dest.cod_prop,
                              calle,
                              altura,
                              dest.cod_destino,
                              desc_destino,
                              superficie
FROM (
         SELECT cod_prop,
                cod_destino,
                CASE
                    WHEN cod_destino = 9 THEN 3
                    WHEN cod_destino = 5 THEN 2
                    WHEN cod_destino = 15 THEN 1
                    END AS priority
         FROM destinos_por_prop
     ) sq
         INNER JOIN destinos_por_prop dest
                    ON sq.cod_destino = dest.cod_destino
                        AND sq.cod_prop = dest.cod_prop
WHERE priority IS NOT NULL
ORDER BY cod_prop, priority DESC
```

[1] https://www.postgresql.org/docs/9.3/sql-select.html#SQL-DISTINCT