**BASES DE DATOS II**

**TRABAJO PRACTICO 1 - QUERYS LINUX**

**PROFESOR: IGNACIO BISSO**

**ignacio\_bisso@yahoo.com**

**ALUMNO: LUCAS VARGAS (37369589/2014) lucasjv92@gmail.com**

**EJERCICIO PARA ENTREGAR (CORREGIDO)**

Fecha Entrega:22-ABR-2017

1 Armar un shell script llamado carga\_inicial.sh que cree las tablas ALUMNO, MATERIA y CURSA y les cargue 10000 alumnos, 20 materias y entre 0 y 4 inscripciones por alumno al azar respectivamente. El script debe borrar las tablas si existen previamente (ver DROP IF EXISTS).

#!/bin/sh

psql $dbname $username << EOF

drop table if exists Alumno;

drop table if exists Materia;

drop table if exists Cursa;

CREATE TABLE Alumno

(

legajo varchar(20) primary key,

nombre varchar(20),

apellido varchar(20),

dni varchar(10),

f\_nacimiento date

);

create table materia

(

codigo varchar(20) ,

nombre varchar(20),

cantidad\_horas int,

promocionable boolean

);

create table cursa

(

legajo varchar(20),

materia varchar(20),

inscripcion\_ts timestamp

);

insert into alumno(legajo,nombre,apellido,dni,f\_nacimiento)

select 10000 + generate\_series(1,10000),

'Juan' || generate\_series(1,10000),

'Perez'||generate\_series(1,10000),

10000 + generate\_series(1,10000),

now()::date + (( random()- 0.5)\* 500 )::integer;

insert into materia (codigo,nombre,cantidad\_horas,promocionable)

select 'MATE'||generate\_series (1, 20),

'Materia'||generate\_series(1,20),

(random() \* 4 + 4)::integer,

((random() \* 1)::integer)::boolean;

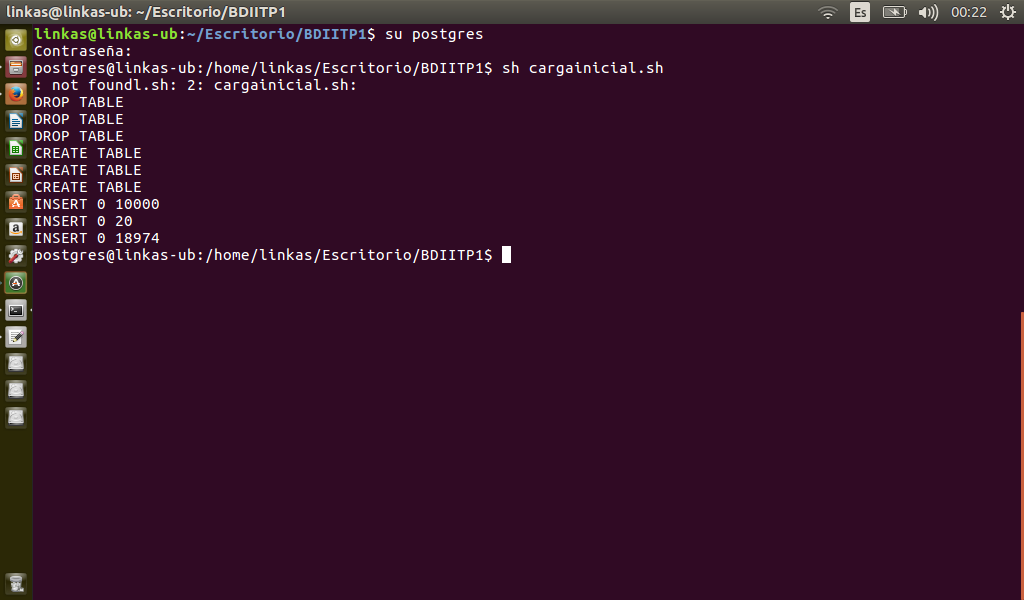
insert into cursa (legajo,materia)--,inscripcion\_ts)

select distinct legajo, codigo\_materia

from (select legajo, generate\_series(1,(random()\*4)::int), 'COD' || ((random()\*19)::int+1) codigo\_materia from alumno) as leg;

EOF

exit 0



2 Armar un shell script con SQL embebido llamado query1.sh que ejecute el query de la guia de laboratorio que devuelve los nombres de los alumnos que cursan 3 materias usando un subquery correlacionado

#!/bin/sh

psql $dbname $username << EOF

select a.nombre

from alumno a

where 3 =

(select count (\*)

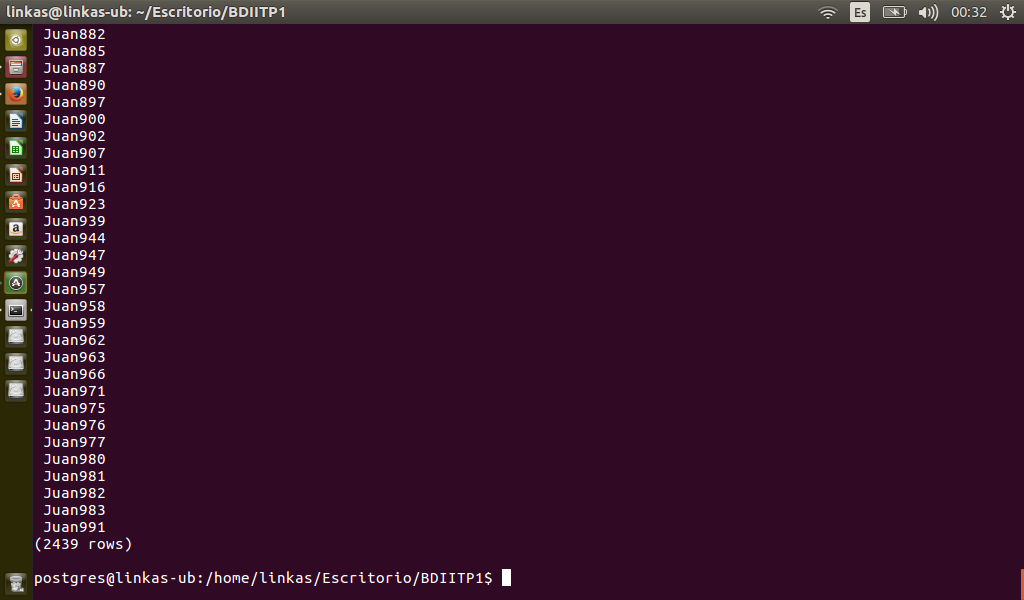
from cursa c

where c.legajo = a.legajo);

EOF

exit 0





3 Armar un shell script con SQL embebido lamado query2.sh que ejecute el query de la guia de laboratorio que devuelve los nombres de los alumnos que cursan 3 materias sin usar un subquery correlacionado

#!/bin/sh

psql $dbname $username << EOF

select nombre

from alumno a

where legajo IN

(select legajo

from cursa

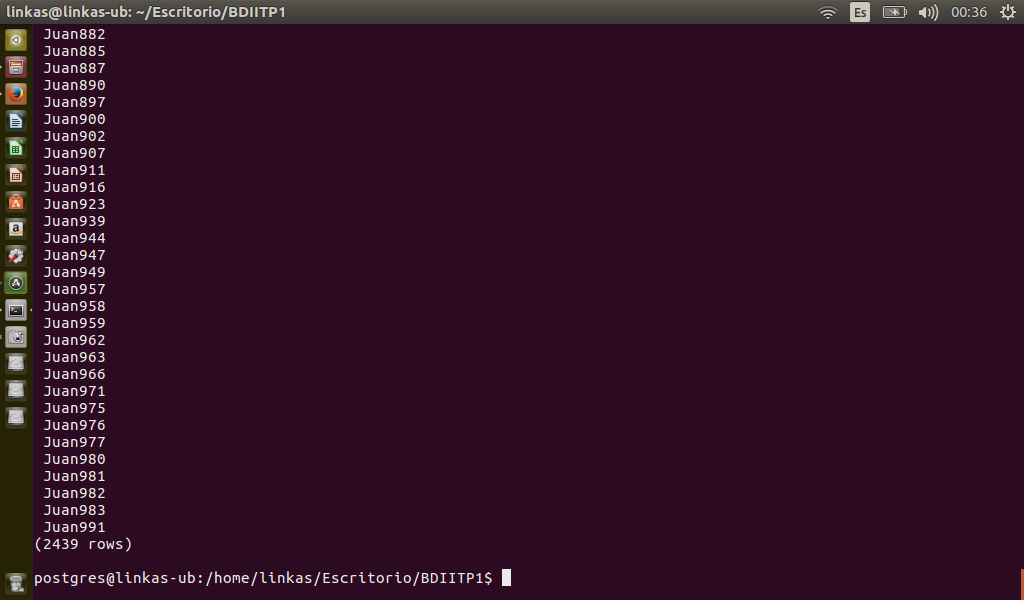
group by legajo

having count(\*) = 3);

EOF

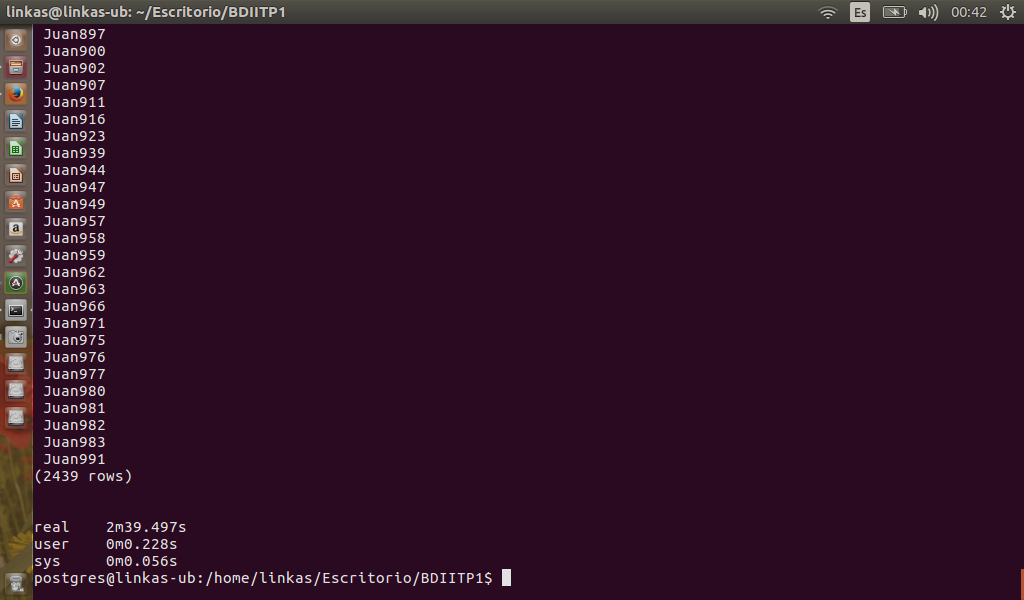
exit 0



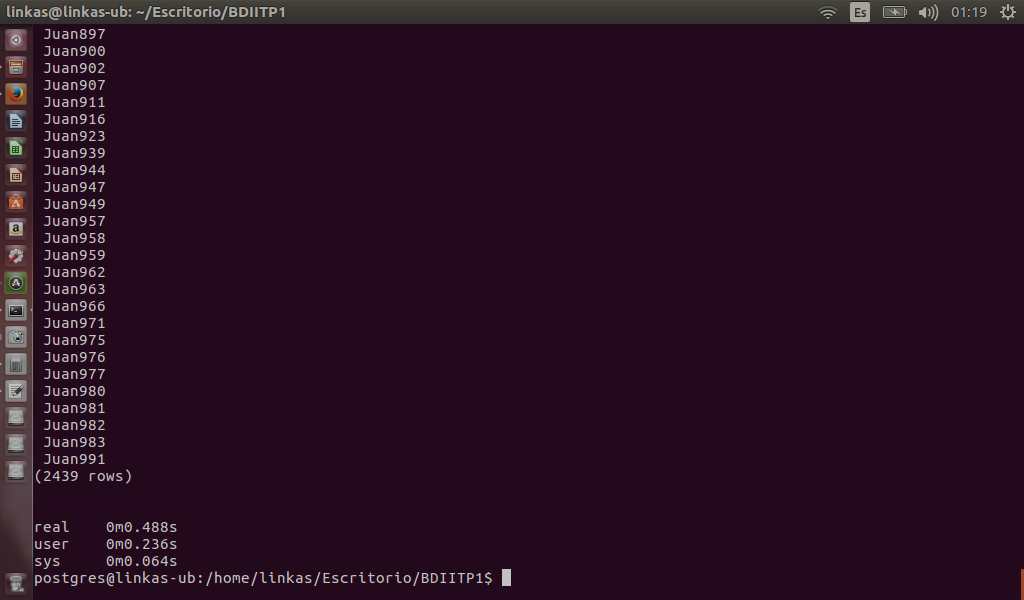


4 Ejecutar los scripts de 2 y 3 anteponiendo el comando time para medir el tiempo de ejecución, comparar y explicar brevemente la causa de la duración de cada query.









Como podemos observar, la query1 tarda mucho más que la query2 debido a que por cada registro de alumno, se debe llamar a la función COUNT en toda la tabla. Como resultado, nos da que las operaciones de lectura de registro que se deben hacer son 10000 \* 18974 = 189740000.

En cambio la query2 al usar GROUP BY legajo HAVING COUNT(\*) = 3 permite de forma rápida leer los legajos que cursan 3 materias, ya que agrupa todos y cuenta la cantidad de cada grupo.

Haciendo la siguiente consulta:

postgres=# select count (\*) from (select legajo from cursa group by legajo having count(\*) = 3 ) as t1;

count

-------

2439

(1 row)

Podemos ver que la cantidad de registros a consultar por cada alumno disminuye, haciendo que el cálculo cambie. 10000\*2439 = 24390000

5 Es posible acelerar alguno de los queries ? En caso de ser posible implementar la mejora en un tercer shell script (query3.sh) y explicar cómo se logró la mejora y el tiempo resultante.

Existe la posibilidad de hacer una query o consulta capaz de mejorar el tiempo en resolver la consulta necesitada. Para ello, por ejemplo, podemos hacer uso de una tabla temporal que vive en memoria RAM para evitar acceder a registros ya leídos anteriormente. En este caso guardamos en la tabla temporal todos los legajos que cursan 3 materias. Luego mostramos el nombre de esos legajos haciendo la relación necesaria.

Tiempo de query1.sh: 2 minutos 39 segundos

Tiempo de query2.sh: 0,488 segundos

Tiempo de query3.sh: 0,469 segundos

Código de query3.sh

#!/bin/sh

psql $dbname $username << EOF

DROP TABLE IF EXISTS alumnosCursan3Materias;

CREATE TEMPORARY TABLE alumnosCursan3Materias(legajo varchar (30));

INSERT INTO tempTable

(

SELECT legajo

FROM cursa

GROUP BY legajo

HAVING count(legajo)=3

);

SELECT nombre

FROM alumno, alumnosCursan3Materias

WHERE alumno.legajo = alumnosCursan3Materias.legajo;

EOF

exit 0

