



Universidad de Guadalajara

Alumno: Sanchez Gomez Edgardo Enrique

Código: 218401479

Asignatura: Administración de Bases de Datos

Práctica: Índices

Fecha de entrega: 17/Febrero/2024

1. Accedemos al servicio con “sqlplus”, e ingresamos el usuario y contraseña para el usuario “edgardom”

```
Windows PowerShell
PS C:\Users\USUARIO> sqlplus edgardom;

SQL*Plus: Release 11.2.0.1.0 Production on Vie Feb 16 19:35:35 2024

Copyright (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production

SQL> CREATE TABLE employees (code NUMBER, name VARCHAR(20));

Table created.

SQL> INSERT INTO employees(name) VALUES ('Pedro');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Juan');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Rodrigo');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Luis');

1 row created.

SQL> INSERT INTO employees SELECT * FROM employees;
```

2. Creamos una tabla para realizar la actividad con “CREATE TABLE employees (code NUMBER, name VARCHAR(20));”

```
Windows PowerShell
PS C:\Users\USUARIO> sqlplus edgardom;

SQL*Plus: Release 11.2.0.1.0 Production on Vie Feb 16 19:35:35 2024

Copyright (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production

SQL> CREATE TABLE employees (code NUMBER, name VARCHAR(20));

Table created.

SQL> INSERT INTO employees(name) VALUES ('Pedro');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Juan');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Rodrigo');

1 row created.

SQL> INSERT INTO employees(name) VALUES ('Luis');

1 row created.

SQL> INSERT INTO employees SELECT * FROM employees;
```

3. Insertamos algunos datos de prueba.

```
Windows PowerShell
PS C:\Users\USUARIO> sqlplus edgardom;

SQL*Plus: Release 11.2.0.1.0 Production on Vie Feb 16 19:35:35 2024

Copyright (c) 1982, 2010, Oracle. All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production

SQL> CREATE TABLE employees (code NUMBER, name VARCHAR(20));

Table created.

SQL> INSERT INTO employees(name) VALUES ('Pedro');
1 row created.

SQL> INSERT INTO employees(name) VALUES ('Juan');
1 row created.

SQL> INSERT INTO employees(name) VALUES ('Rodrigo');
1 row created.

SQL> INSERT INTO employees(name) VALUES ('Luis');
1 row created.

SQL> INSERT INTO employees SELECT * FROM employees;
```

4. Ejecutamos la sentencia “**INSERT INTO employees SELECT * FROM employees**” para insertar elementos más rápido, utilizando los mismos elementos ya pre insertados en la tabla, y repetimos este proceso hasta juntar más de 4 millones de registros en dicha tabla.

```
Windows PowerShell

SQL> INSERT INTO employees SELECT * FROM employees;
4 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
8 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
16 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
32 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
64 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
128 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
256 rows created.

SQL> INSERT INTO employees SELECT * FROM employees;
512 rows created.
```

5. Actualizamos la columna code de todos los registros para que tengan el número de elemento o fila que ocupan con “**UPDATE employees SET code = ROWNUM;**” y ejecutamos “**COMMIT;**” para guardar los cambios.

```
Windows PowerShell
SQL> UPDATE employees SET code = ROWNUM;
8388608 rows updated.
SQL> COMMIT;
Commit complete.

SQL> SET TIMING ON SELECT * FROM employees WHERE code = '523365';
SP2-0158: unknown SET option "SELECT"
SQL> SET TIMING ON;
SQL> SELECT * FROM employees WHERE code = '5244563';

CODE NAME
-----
5244563 Luis

Elapsed: 00:00:05.30
SQL> SELECT * FROM employees WHERE code = '2255456';

CODE NAME
-----
2255456 Rodrigo

Elapsed: 00:00:02.49
SQL> SELECT * FROM employees WHERE code = '8122023';

CODE NAME
-----
8122023 Luis

Elapsed: 00:00:01.67
```

6. Activamos un indicador de tiempo con “**SET TIMING ON;**”, seguido de algunas consultas de prueba para analizar el tiempo que tarda en las consultas sin contar con un índice.

```
Windows PowerShell
SQL> UPDATE employees SET code = ROWNUM;
8388608 rows updated.
SQL> COMMIT;
Commit complete.

SQL> SET TIMING ON SELECT * FROM employees WHERE code = '523365';
SP2-0158: unknown SET option "SELECT"
SQL> SET TIMING ON;
SQL> SELECT * FROM employees WHERE code = '5244563';

CODE NAME
-----
5244563 Luis

Elapsed: 00:00:05.30
SQL> SELECT * FROM employees WHERE code = '2255456';

CODE NAME
-----
2255456 Rodrigo

Elapsed: 00:00:02.49
SQL> SELECT * FROM employees WHERE code = '8122023';

CODE NAME
-----
8122023 Luis

Elapsed: 00:00:01.67
```

7. Creamos un índice para la tabla employees en la columna code con “**CREATE INDEX indemployees ON employees(code);**”, y ejecutamos algunas consultas para comparar la velocidad de búsqueda con el índice creado, y guardamos todo con “**COMMIT;**”

```
Windows PowerShell
Elapsed: 00:00:01.67
SQL> SELECT * FROM employees WHERE code = '8222555';

CODE NAME
-----
8222555 Rodrigo

Elapsed: 00:00:00.34
SQL> CREATE INDEX indemployees ON employees(code);

Index created.

Elapsed: 00:00:32.92
SQL> SELECT * FROM employees WHERE code = '4256589';

CODE NAME
-----
4256589 Pedro

Elapsed: 00:00:00.03
SQL> SELECT * FROM employees WHERE code = '8325144';

CODE NAME
-----
8325144 Luis

Elapsed: 00:00:00.03
SQL> COMMIT;

Commit complete.

Elapsed: 00:00:00.00
SQL>
```