07

Consolidando o seu conhecimento

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Acesse o **SQL Developer** e se conecte com o usuário **USER13**.

2) Caso você ainda não tenha feito, faça o download dos arquivos (https://caelum-online-public.s3.amazonaws.com/1603-oracle-db-performance-parte-2/04/arquivos-aula-4.zip) desta aula, copie o conteúdo do *script* **Criacao_Tabela.sql** e cole na área de edição.

3) Execute, inicialmente, os comandos abaixo:

```
DROP TABLE TEST_RANDOM;
Create table test_random
as
select /*+ append */ * from test_normal WHERE ROWNUM <= 100000 order by dbms_random.random;
UPDATE TEST_RANDOM SET FAIXA = 'baixa' WHERE SAL >= 1000 AND SAL <= 3000;
UPDATE TEST_RANDOM SET FAIXA = 'media' WHERE SAL > 3000 AND SAL <= 6000;
UPDATE TEST_RANDOM SET FAIXA = 'alta' WHERE SAL > 6000 AND SAL <= 7000;
COMMIT;
```

4) Para você ter um relatório, obtenha-o através da consulta abaixo:

```
SELECT FAIXA, COUNT(*) AS NUM_FUNC, SUM(SAL) AS S0MA_SALARIO
FROM TEST_RANDOM GROUP BY FAIXA;
```

5) Verifique o custo desta consulta. No caso do vídeo, o instrutor obteve 196.

6) Abra uma outra área de comandos, com o usuário **USER13**, e nela, crie a ***Materialized View***:

```
CREATE MATERIALIZED VIEW LOG ON TEST_RANDOM
WITH ROWID, SEQUENCE (FAIXA, SAL) INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW TEST_RANDOM_SUM
REFRESH ON COMMIT
ENABLE QUERY REWRITE
AS SELECT FAIXA, COUNT(*) AS NUM_FUNC, SUM(SAL) AS S0MA_SALARIO
FROM TEST_RANDOM GROUP BY FAIXA;
```

7) Se você executar a consulta abaixo:

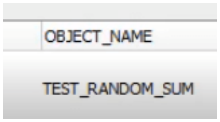
```
SELECT * FROM TEST_RANDOM_SUM;
```

Verá o mesmo resultado do relatório obtido pela SQL original.

8) Verifique o plano de execução da SQL original, após a criação da *Materialized View*:

```
SELECT FAIXA, COUNT(*) AS NUM_FUNC, SUM(SAL) AS S0MA_SALARIO
FROM TEST_RANDOM GROUP BY FAIXA;
```

Verificando o plano de execução da SQL original, após a criação da *Materialized View*, você vê que o custo caiu para 3. E o Oracle reconheceu a existência da *Materialized View* na hora de montar o plano de execução:



9) Crie mais uma área de edição de comandos no SQL Developer, associada ao usuário **USER13**.

10) Inclua 200 novos funcionários na tabela. Para isto, execute:

```
Begin
    For i in 1000000..1000200
    Loop
        Insert into TEST_RANDOM values(
            to_char(i), dbms_random.string('U',30),
            dbms_random.value(1000,7000), 'ND'
        );
        If mod(i, 10000) = 0 then
            Commit;
        End if;
    End loop;
End;
```

11) Recalcule as faixas salariais. Não somente para os novos funcionários, mas sim para todos os outros. E crie também uma nova faixa. Execute:

```
UPDATE TEST_RANDOM SET FAIXA = 'minimo' WHERE SAL >= 1000 AND SAL <= 2000;
UPDATE TEST_RANDOM SET FAIXA = 'baixo' WHERE SAL > 2000 AND SAL <= 4000;
UPDATE TEST_RANDOM SET FAIXA = 'media' WHERE SAL > 4000 AND SAL <= 6000;
UPDATE TEST_RANDOM SET FAIXA = 'alta' WHERE SAL > 6000 AND SAL <= 7000;
COMMIT;
```

12) Para efetuar mais modificações na tabela, exclua todos os funcionários da faixa **Alta**:

```
DELETE FROM TEST_RANDOM WHERE FAIXA = 'alta';
commit;
```

13) Execute a visão materializada. Note que os dados mudaram junto com as atualizações da tabela:

```
SELECT * FROM TEST_RANDOM_SUM;
```