UTL_FILE PERFORMANSI

İçindekiler

UTL_FILE işletim sistemi üzerinde birçok işlem yapmamızı (dosya yazma, okuma, silme, kopyalama vs...) sağlayan bir Oracle paketi. Bu yazıda, UTL_FILE paketini yazma işlemlerinde nasıl daha performanslı kullanabileceğimizi göreceğiz.

UTL_FILE ile yazarken, performans arttırmanın yolu, olabildiğince az I/O yapmaktan geçiyor. Ne kadar çok I/O yapılırsa, diske o kadar sık erişileceğinden performans azalacaktır. Mesela her satırı ayrı ayrı yazmak yerine, bir noktaya kadar biriktirmek ardından veriyi yazmak, UTL_FILE işlem süresini kısaltacaktır.

Çalışmadaki detayları aşağıda bulabilirsiniz. Fakat kısa bir özet geçmek gerekirse;

1. Veritabanında CLOB / BLOB'ların tutulacağı bir tablo yaratıp, raporlar burada saklanabilir. Tablo CLOB olarak saklanacaksa, süre ve boyut olarak iyi bir sonuç elde edemiyoruz. Fakat sıkıştırıp, BLOB olarak saklama imkanı varsa, performans ve alan tüketimi iyi oluyor. Yine de çok data tutulması hâlinde veritabanını son derece hızlı şekilde büyütebilir. Eğer düzenli silme yapılmayacaksa, iyi bir seçim olmayacaktır.

2. Dosyalara yazmaya devam edilebilir. Fakat direkt UTL_FILE ile dosyaya çıkmak yerine değişkenlerde datayı biriktirmek, ardından biriken datayı dosyaya çıkmak performans artışı sağlar.

3. Datayı sıkıştırıp dosyaya yazmak, ya da sıkıştırıp tabloda tutmak mümkün. En zayıf sıkıştırma derecesiyle bile, çok ciddi tasarruflar sağlanabiliyor. Harcanan süre ve boyut arasındaki ilişkiyi iyi hesap etmek gerekli.

4. Sonuçları dosyaya yazarken External Table kullanabilirsiniz. Fakat denemelerimde iyi sonuçlar vermedi. Daha farklı parametrelerle daha iyi sonuçlar elde edilebilir. Fakat yazının ana konusu UTL_FILE olduğu ve External Table'larla iyi sonuç elde edemediğimiçin çalışmadan çıkarttım.

Yaptığım denemelerin sonuçlarını aşağıda bulabilirsiniz:

YÖNTEM	SÜRE (SN)	BOYUT (MB)
1. Standart UTL_FILE Kullanımı	42,36	265,84
2. VARCHAR Ara Değişken Kullanımı	21,40	265,84
3. CLOB Ara Değişken Kullanımı	19,87	265,84
4. CLOB Ara Değişken + Sıkıştırma, Dosyaya Çıkma *	36,12	58,47
5. CLOB Ara Değişken + Sıkıştırma, BLOB Tabloya Yazma *	60,86	60,00
6. CLOB Ara Değişken, CLOB Tabloya Yazma	116,94	216,00

* Sıkıştırma için UTL_COMPRESS paketindeki LZ_COMPRESS prosedürü en düşük sıkıştırma derecesi ile kullanıldı. Daha yüksek derecelerde, boyut ufaltmaktadır. Prosedür Lempel-Ziv algoritmasını kullanmakta ve sıkıştırılan dosyalar RAR, 7zip, vb... güncel programlarla extract edilebilir. (UTL_COMPRESS paketinde extraction için de bir prosedür bulunmaktadır.)

0.ÇALIŞMADA KULLANILMAK ÜZERE DATA OLUŞTURMA

```
-- DATANIN SAKLANACAGI TABLO
CREATE TABLE D_CCEBI.DENEME( ID NUMBER, DUMMY_DATA VARCHAR2(2000) )
TABLESPACE TB_DATA
NOLOGGING;


-- 250BIN SATIRLIK DATA OLUSTURMA KISMI
SET TIMING ON
SET SERVEROUTPUT ON
DECLARE
    V_TEMP  DBMS_SQL.VARCHAR2S;
    i       PLS_INTEGER;
    V_DATA  VARCHAR2(2000);
BEGIN
    V_TEMP(0) := CONCAT( 'Lorem ipsum dolor sit amet, consectetur',
                         'In id mi vestibulum mauris lobortis rutrum' );
    V_TEMP(1) := CONCAT( 'Suspendisse auctor lorem vitae arcu ultricies',
                         'Phasellus quis elit non nisl sollicitudin' );
    V_TEMP(2) := CONCAT( 'Sed posuere nisi eget mauris pretium dictum.',
                         'Sed lorem massa, interdum ac euismod' );
    V_TEMP(3) := CONCAT( 'Vestibulum gravida vehicula tempus. Duis',
                         'Pellentesque consectetur, enim eget vestibulum');
    V_DATA := CONCAT( V_TEMP(0), V_TEMP(1) );
    V_DATA := CONCAT( V_DATA, V_TEMP(2) );
    V_DATA := CONCAT( V_DATA, V_TEMP(3) );
```

```
    FOR i IN 1..250000 LOOP
        INSERT /*+append nologging*/ INTO
            D_CCEBI.DENEME VALUES( i, V_DATA );
        IF MOD(i,10000)=0 THEN
            COMMIT;
        END IF;
    END LOOP;
    COMMIT;
END;
PL/SQL procedure successfully completed.
Elapsed: 00:01:13:48

SQL> CREATE OR REPLACE DIRECTORY CCEBI_TEMP AS '/oracle/oracle1/file_test';
```

1. HERHANGİ BİR İYİLEŞTİRME YAPMADAN STANDART UTL_FILE KULLANIMI

```
-- DIREKT YAZDIRMA
SET TIMING ON
DECLARE
  l_file  UTL_FILE.FILE_TYPE;
BEGIN
    l_file := utl_file.fopen('CCEBI_TEMP','deneme.text','W');
    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        utl_file.put_line(l_file, c.DUMMY_DATA );
    END LOOP;
    utl_file.fclose(l_file);
END;
PL/SQL procedure successfully completed.
Elapsed: 00:00:42:36
```

2. I/O MİKTARINI AZALTMAK İÇİN VARCHAR ARA DEĞİŞKEN KULLANMAK

```
-- DEGISKENDE TUTUP YAZDIRMA
SET TIMING ON
DECLARE
  l_file    UTL_FILE.FILE_TYPE;
  V_BUFFER  VARCHAR(32767) := null;
BEGIN
    l_file := utl_file.fopen('CCEBI_TEMP','deneme.text','W',32767);
    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        IF v_buffer is null THEN
            V_BUFFER := C.DUMMY_DATA;
        ELSIF LENGTH(V_BUFFER) + 1 + LENGTH(C.DUMMY_DATA) <= 32767 THEN
            V_BUFFER := V_BUFFER||CHR(10)||C.DUMMY_DATA;
        ELSE
            utl_file.put_line( l_file, V_BUFFER );
            V_BUFFER := C.DUMMY_DATA;
        END IF;
    END LOOP;
```

```
    utl_file.put_line( l_file, V_BUFFER );
    utl_file.fclose( l_file );
END;
PL/SQL procedure successfully completed.
Elapsed: 00:00:21:40
```

3. I/O MİKTARINI AZALTMAK İÇİN CLOB ARA DEĞİŞKEN KULLANMAK

```
-- CLOB DEGISKENDE TUTUP, TEK SEFERDE YAZDIRMA
SET TIMING ON
DECLARE
  l_file            UTL_FILE.FILE_TYPE;
  V_SMALL_BUFFER    VARCHAR(32767) := null;
  V_BIG_BUFFER      CLOB := 'NULL';
BEGIN

    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        IF V_SMALL_BUFFER is null THEN
            V_SMALL_BUFFER := C.DUMMY_DATA;
        ELSIF LENGTH(V_SMALL_BUFFER)+1+LENGTH(C.DUMMY_DATA) <= 32767 THEN
            V_SMALL_BUFFER := V_SMALL_BUFFER||CHR(10)||C.DUMMY_DATA;
        ELSE
            DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) );
            V_SMALL_BUFFER := C.DUMMY_DATA;
        END IF;
    END LOOP;
    DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) );
    DBMS_XSLPROCESSOR.CLOB2FILE(V_BIG_BUFFER, 'CCEBI_TEMP', 'deneme.text');
    DBMS_LOB.FREETEMPORARY( V_BIG_BUFFER );
END;
PL/SQL procedure successfully completed.
Elapsed: 00:00:19:87
```

4. I/O MİKTARINI AZALTMAK İÇİN CLOB DEĞİŞKEN KULLANIP, SIKIŞTIRMA

```
-- SIKISTIRMA ISLEMINDE KULLANILMAK UZERE IKI PROSEDUR YARATIYORUZ
CREATE OR REPLACE FUNCTION D_CCEBI.c2b( c IN CLOB ) RETURN BLOB
IS
pos PLS_INTEGER := 1;
buffer RAW( 32767 );
res BLOB;
lob_len PLS_INTEGER := DBMS_LOB.getLength( c );
BEGIN
    DBMS_LOB.createTemporary( res, TRUE );
    DBMS_LOB.OPEN( res, DBMS_LOB.LOB_ReadWrite );

    LOOP
    buffer := UTL_RAW.cast_to_raw( DBMS_LOB.SUBSTR( c, 16000, pos ) );
```

```
    IF UTL_RAW.LENGTH( buffer ) > 0 THEN
    DBMS_LOB.writeAppend( res, UTL_RAW.LENGTH( buffer ), buffer );
    END IF;

    pos := pos + 16000;
    EXIT WHEN pos > lob_len;
    END LOOP;

    RETURN res; -- res is OPEN here
END;

CREATE OR REPLACE PROCEDURE D_CCEBI.write_blob_to_file
(V_BLOB BLOB, V_FNAME VARCHAR2, V_DIRNAME VARCHAR2 ) IS
vstart      NUMBER := 1;
bytelen     NUMBER := 32000;
len         NUMBER;
my_vr       RAW(32000);
x           NUMBER;
l_output utl_file.file_type;
BEGIN

-- define output directory
l_output := utl_file.fopen(V_DIRNAME, V_FNAME,'wb', 32760);

vstart := 1;
bytelen := 32000;

-- get length of blob
len := dbms_lob.getlength(V_BLOB);

-- save blob length
x := len;

-- if small enough for a single write
IF len < 32760 THEN
utl_file.put_raw(l_output,V_BLOB);
utl_file.fflush(l_output);
ELSE -- write in pieces
vstart := 1;
    WHILE vstart < len and bytelen > 0
    LOOP
       dbms_lob.read(V_BLOB,bytelen,vstart,my_vr);

       utl_file.put_raw(l_output,my_vr);
       utl_file.fflush(l_output);

       -- set the start position for the next cut
       vstart := vstart + bytelen;

       -- set the end position if less than 32000 bytes
       x := x - bytelen;
       IF x < 32000 THEN
          bytelen := x;
```

```
       END IF;
    END LOOP;
utl_file.fclose(l_output);
END IF;
END;

-- SIKISTIRIP, DOSYA YAZMA
SET TIMING ON
DECLARE
  l_file            UTL_FILE.FILE_TYPE;
  V_SMALL_BUFFER    VARCHAR(32767) := null;
  V_BIG_BUFFER      CLOB := 'NULL';
  V_COMPRESS_DATA   BLOB;
BEGIN

    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        IF V_SMALL_BUFFER is null THEN
            V_SMALL_BUFFER := C.DUMMY_DATA;
        ELSIF LENGTH(V_SMALL_BUFFER)+1+LENGTH(C.DUMMY_DATA) <= 32767 THEN
            V_SMALL_BUFFER := V_SMALL_BUFFER||CHR(10)||C.DUMMY_DATA;
        ELSE
            DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
            V_SMALL_BUFFER := C.DUMMY_DATA;
        END IF;
    END LOOP;
    DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
    V_COMPRESS_DATA:=UTL_COMPRESS.LZ_COMPRESS(
            SRC=> d_ccebi.c2b( V_BIG_BUFFER ),
            quality=>1 );

    d_ccebi.write_blob_to_file(V_COMPRESS_DATA, 'deneme2.lz','CCEBI_TEMP');
    DBMS_LOB.FREETEMPORARY( V_BIG_BUFFER );
END;
PL/SQL procedure successfully completed.
Elapsed: 00:00:36:12
```

5. SIKIŞTIRILAN DATAYI DOSYA YERINE BLOB OLARAK TABLOYA YAZMA

```
-- YAZILACAK TABLO
CREATE TABLE D_CCEBI.RAPOR_TABLOSU
( RAPOR_ID NUMBER, TARIH DATE DEFAULT SYSDATE, RAPOR_VERI BLOB);

-- SIKISTIRIP, TABLOYA KAYDETME
SET TIMING ON
DECLARE
  l_file                UTL_FILE.FILE_TYPE;
  V_SMALL_BUFFER        VARCHAR(32767) := null;
  V_BIG_BUFFER          CLOB := 'NULL';
  V_COMPRESSED_DATA     BLOB;
BEGIN
```

```
    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        IF V_SMALL_BUFFER is null THEN
            V_SMALL_BUFFER := C.DUMMY_DATA;
        ELSIF LENGTH(V_SMALL_BUFFER)+1+LENGTH(C.DUMMY_DATA) <= 32767 THEN
            V_SMALL_BUFFER := V_SMALL_BUFFER||CHR(10)||C.DUMMY_DATA;
        ELSE
            DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
            V_SMALL_BUFFER := C.DUMMY_DATA;
        END IF;
    END LOOP;
    DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
    V_COMPRESSED_DATA:=UTL_COMPRESS.LZ_COMPRESS(
            SRC=> d_ccebi.c2b( V_BIG_BUFFER ),
            quality=>1 );
    INSERT INTO D_CCEBI.RAPOR_TABLOSU
      VALUES( 1, SYSDATE, V_COMPRESSED_DATA );
    COMMIT;
    DBMS_LOB.FREETEMPORARY( V_BIG_BUFFER );
END;
PL/SQL procedure successfully completed.
Elapsed: 00:01:00:86
```

6. DATAYI SIKIŞTIRMADAN DOSYA YERINE CLOB OLARAK TABLOYA YAZMA

```
CREATE TABLE D_CCEBI.RAPOR_TABLOSU
( RAPOR_ID NUMBER,
  TARIH DATE DEFAULT SYSDATE,
  RAPOR_VERI CLOB);

SET TIMING ON
DECLARE
  l_file            UTL_FILE.FILE_TYPE;
  V_SMALL_BUFFER    VARCHAR(32767) := null;
  V_BIG_BUFFER      CLOB := 'NULL';
BEGIN

    FOR c IN ( SELECT DUMMY_DATA FROM D_CCEBI.DENEME ) LOOP
        IF V_SMALL_BUFFER is null THEN
            V_SMALL_BUFFER := C.DUMMY_DATA;
        ELSIF LENGTH(V_SMALL_BUFFER) +1+ LENGTH(C.DUMMY_DATA) <= 32767 THEN
            V_SMALL_BUFFER := V_SMALL_BUFFER||CHR(10)||C.DUMMY_DATA;
        ELSE
            DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
            V_SMALL_BUFFER := C.DUMMY_DATA;
        END IF;
    END LOOP;
    DBMS_LOB.WRITEAPPEND( V_BIG_BUFFER, LENGTH( V_SMALL_BUFFER )+1,
                                  V_SMALL_BUFFER||CHR(10) ) ;
    INSERT INTO D_CCEBI.RAPOR_TABLOSU VALUES( 1, SYSDATE, V_BIG_BUFFER );
```

```
    COMMIT;
    DBMS_LOB.FREETEMPORARY( V_BIG_BUFFER );
END;
PL/SQL procedure successfully completed.
Elapsed: 00:01:56:94
```

Verilen örneklerin hepsinde alt satıra geçmek için **chr(10)** kullandığımı farketmişsinizdir. Ancak Windows tabanlı işletim sistemi kullanıyorsanız, chr(13)||chr(10) kullanmanız gerekir. Kodunuzun platform bağımsız çalışması için statik olarak chr(10) yazmak yerine, aşağıdaki gibi bir yöntemi tercih edebilirsiniz:

```
v_eol:= CASE
            WHEN DBMS_UTILITY.PORT_STRING LIKE 'IBMPC%'
            THEN CHR(13)||CHR(10)
            ELSE CHR(10)
        END;
```

Son olarak kodlardaki bir hatamı farkedip, beni uyaran **Ümit Karaoğul**'a teşekkür ederim.

7. KAYNAKLAR

- **TUNING PL/SQL FILE I/O**
  (http://www.oracle-developer.net/display.php?id=425 **)**

- **How to convert a CLOB to BLOB**
  (http://forums.oracle.com/forums/thread.jspa?messageID=1587220 **)**

- **Oracle UTL_FILE**
  (http://www.psoug.org/reference/utl_file.html **)**