ISBN: 979-756-061-6

KOMPRESI DATA PADA ORACLE 9i

Irving Vitra Paputungan
Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia
Jl. Kaliurang Km 14 Yogyakarta Telp. (0274) 895287 ext. 122, Faks. (0274) 895007 ext. 148
E-mail: ipink@engineer.com, irving@fti.uii.ac.id

Abstraksi

Most decision support systems usually involve large amounts of data stored in a few very large tables. As these systems evolve, the demand on disk space can grow quickly. In today's environment, data warehouses of hundreds of terabytes have become increasingly common.

This paper will show how The Compression in Oracle 9i Release 2 works. In order to build and manage the database well.

Keywords:

1. Pendahuluan

Beberapa Sistem Informasi yang berbasis Sistem Pendukung Keputusan, sebagian besar mereka membutuhkan banyak sekali *tablespace* untuk datanya. Dalam perkembangan sistem tersebut, kebutuhan akan *discspace* juga semakin banyak. Sekarang ini, *data warehousing* yang menggunakan data mencapai *terrabytes* data sudah mulai bermunculan.

Untuk membantu mengatasi isu yang berkembang seperti itu, ada *feature* baru di dalam Oracle 9i yang diperkenalkan tentang Kompresi *Tabelspace*. Dengan *feature* tersebut, *tablespace* untuk data dapat dikurangi dan performance dari database dapat ditingkatkan.

Feature dari Oracle 9i ini bekerja dengan mengeliminasi data-data terduplikasi yang ditemukan dalam tabel-tabel pada database. Kompresi ini bekerja pada level database blok. Ketika database terdefinisikan secara terkompres, maka database tersebut akan memesan space pada tiap blok database untuk menyimpan satu dari beberapa kemunculan data yang sama pada blok tersebut. Space ini dinamakan tabel simbol. Data yang tertata untuk kompresi tersebut disimpan hanya pada tabel simbol dan bukan pada baris-baris database tersebut sendiri. Ini berarti bahwa, akan ada satu atau lebih pointer yang muncul untuk menunjukkan data sebenarnya pada tabel simbol, disamping data itu sendiri.

Hasil lain dari tabel kompresi ini adalah sebuah transparasi untuk user dan para pengembang aplikasi. Pada pengembang mampu mengakses tabel-tabel terkompres tersebut tanpa menghiraukan kondisinya (terkompres atau tidak), sehingga query-query SQL tidak perlu berubah ketika sudah didefinisikan diawal pembuatan tabel. Tabel terkompres ini didefinisikan oleh pada administrator atau perancang database, dengan melibatkan sedikit pengembang maupun pemakai.

2. Pembuatan Tabel Terkompres

Untuk membuat tabel terkompres, dibutuhkan kata kunci yang itu COMPRESS ketika akan membuat tabel (CREATE TABLE). Kata kunci tersebut menunjukkan bahwa database pada Oracle akan langsung terformat secara kompres sebisa mungkin. Misalnya:

```
CREATE TABLE MAHASISWA (
  NO_MHS       VARCHAR2(8) NOT NULL,
  NAMA         VARCHAR2(50) NOT NULL,
  TGL_LAHIR    DATE NOT NULL,
  KD_ASAL      VARCHAR2(3) NOT NULL
)
COMPRESS
;
```

Atau dapat juga dideklarasikan pada sebuah tabel yang sudah terbuat, menggunakan kata kunci ALTER, misalnya:

```
ALTER TABLE MAHASISWA COMPRESS;
```

Untuk melihat apakah tabel yang dibuat terkompresi atau tidak, dapat dilihat melalui kamus data tabel yaitu USER_TABLES pada kolom COMPRESSION.

```
SELECT TABLE_NAME, COMPRESSION FROM
USER_TABLES;

TABLE_NAME           COMPRESSION
-------------------  -----------
MAHASISWA_TEMP       DISABLED
MAHASISWA            ENABLED
```

Atribut kompres dapat juga didefinisikan pada level *tablespace*, baik ketika pembuatan *tablespace* baru (CREATE TABLESPACE) maupun memodifikasi *tablespace* yang sudah ada (ALTER TABLESPACE). Atribut COMPRESS mempunyai properti semacam parameter penyimpanan. Maksudnya, ketika sebuah tabel didefinisikan pada sebuah *tablespace*, maka tabel tersebut akan mempunyai properti yang sama

ISBN: 979-756-061-6

seperti *tablespace* tersebut. Untuk melihat sebuah *tablespace* terkompresi atau tidak, dapat dilihat pada kamus data DBA_TABLESPACES kolom DEF_TAB_COMPRESSION.

```
SELECT TABLESPACE_NAME,
DEF_TAB_COMPRESSION
FROM DBA_TABLESPACES;

TABLESPACE_NAME     DEF_TAB_COMPRESSION
---------------     -------------------
DATA_MHS            DISABLED
INDEX_MHS           DISABLED
```

Walaupun seperti itu, tabel juga masih dapat dimodifikasi kondisinya tanpa menghiraukan kondisi *tablespace*-nya, seperti perintah sebelumnya.

3. Input Data pada Tabel Terkompresi

Ketika sebuah tabel didefinisikan terkompres, sebenarnya belum terjadi sebuah kompresi data sampai diinputkan data ke dalam tabel tersebut. Perintah atau kata kunci yang disebutkan atau dilakukan diatas hanya memodifikasi pada level kamus data. Lebih jauh lagi, untuk memastikan bahwa data benar-benar terkompres, dibutuhkan sebuah metode yang tepat dalam memasukkan atau menyisipkan data ke dalam tabel. Kompresi data hanya terjadi ketika sejumlah data yang sangat besar diinputkan ke dalam tabel melalui satu dari empat proses berikut:

a. Langsung melalui SQL*Loader
b. Serial INSERT menggunakan APPEND
c. Paralel INSERT
d. Melalui perintah CREATE TABLE ... AS SELECT

Cara yang pertama, yaitu melalui SQL*Loader, merupakan cara yang paling baik untuk mengisikan data ke dalam sebuah tabel jika data yang tersedia adalah file yang berukuran kecil.

```
$sqlldr              IRVING/IRVING@DBMHS
control=mahasiswa.ctl direct=true
```

Jika data yang akan diisikan terdapat pada sebuah tabel yang sedang dipakai, maka metode kedua dan ketiga dapat dipakai. Sebagai contoh, anggap saja input data terdapat pada tabel tak terkompres yang sedang dipakai, yaitu MAHASISWA. Menggunakan metode serial INSERT, perintah yang dapat dituliskan yaitu:

```
INSERT /*+ APPEND */
INTO MAHASISWA_TEMP
SELECT * FROM MAHASISWA;
```

Atau dapat menggunakan metode paralel INSERT untuk mentransfer data ke dalam tabel yang terkompres.

```
ALTER SESSION ENABLE PARALLEL DML;

INSERT /*+PARALLEL(MAHASISWA_TEMP,4)*/
INTO MAHASISWA_TEMP
SELECT * FROM MAHASISWA;
```

Akan tetapi, jika metode paralel INSERT digunakan, maka langkah sebelumnya yang harus dilakukan adalah dengan mengaktifkan pararel DML untuk session tersebut, menggunakan perintah ALTER SESSION ENABLE PARALLEL DML.

Jika input data berupa file berukuran kecil, dapat digunakan sebuah eksternal tabel, yang kemudian isikan data ke dalam tabel yang sudah terkompres seperti halnya ketika data ada di dalam tabel yang sedang dipakai.

Dapat juga digunakan perintah CREATE TABLE ... AS SELECT untuk membuat tabel yang terkompres dan mengisikan data kedalamnya dalam sekali perintah.

```
CREATE TABLE MAHASISWA_TEMP
COMPRESS
AS SELECT * FROM MAHASISWA;
```

Jika metode-metode yang tadi dijelaskan tidak digunakan, maka data yang terdapat di dalam tabel tetap akan dalam kondisi tak terkompres, walaupun tabel yang didefinisikan adalah tabel terkompres. Misalnya saja, jika menggunakan path SQL*Loader yang lama, atau perintah INSERT yang biasa, data tidak akan terkompres.

4. Penggunaan Tabel Terkompres

Penentuan bilamana sebuah tabel akan dikompres atau tidak, tergantung dari aplikasi yang akan dibangun, tetapi selalu direkomendasikan kompresi. Seperti yang telah dijelaskan sebelumnya, data yang terdapat pada tabel terkompresi akan terkompresi jika melalui salah satu dari empat metode tersebut. Jika data diisikan tanpa menggunakan metode tersebut makan akan tetap tak terkompresi.

Pada sistem OLTP (*Online Transaction Processing*), data biasanya diisikan secara reguler. Sebagai hasilnya, tabel-tabel yang dibuat tidak akan mengkompresi data-datanya. Tabel terkompresi akan bekerja dengan baikpada tabel-tabel yang beratribut '*read-only*' tetapi sering dibaca. Misalnya data-data yang terdapat pada data warehousing merupakan kandidat terkuat untuk dikompres.

Lebih jauh lagi, mengupdate data pada tabel yang terkompresi mungkin membutuhkan baris-baris untuk tidak dikompres, yang ini merupakan kontradiksi dari pengertian kompresi data. Maka dari itu, tabel-tabel yang membutuhkan untuk selalu diupdate secara konsisten tidak bisa dibuat secara terkompresi.

ISBN: 979-756-061-6

Pada akhirnya, harus dipikirkan bagaimana pengaruh dari penghapusan kolom dalam penggunaan tabel terkompresi. Ketika terjadi penghapusan data pada tabel terkompres, database akan mengkosongkan sejumlah space yang sedang dipakai oleh kolom dalam blok database. Space kosong tersebut dapat digunakan kembali pada pengisian data dilain waktu. Akan tetapi, ketika sebuah baris kembali diisi melalui cara konvensional, maka tidak akan terkompres lagi, bukan memasuki baris yang sudah dibebaskan sebelumnya (pada baris terkompres). Penggunaan DELETE dan INSERT dalam jumlah yang banyak akan menyebabkan terjadinya fragmentasi dan akhirnya akan membuang lebih banyak space dibandingkan dengan penggunaan kompresi tabel.

5. Pengkompresan Tabel yang Belum Terkompres

Jika tabel-tabel yang ada merupakan tabel yang belum terkompresi, perintah ALTER TABLE ... MOVE dapat digunakan untuk mengkompresnya. Misalnya, pada tabel SALES_HISTORY_TEMP dapat dikompres dengan cara:

```
ALTER TABLE MAHASISWA_TEMP
MOVE COMPRESS;
```

Atau juga perintah tersebut bisa digunakan untuk menghilangkan kondisi kompresi sebuah tabel.

```
ALTER TABLE MAHASISWA_TEMP
MOVE NOCOMPRESS;
```

Yang perlu diperhatikan adalah penggunaan perintah tersebut membutuhkan sebuah penguncian secara EXCLUSIVE pada tabel, dimana akan mencegah suatu operasi DML ketika perintah tersebut dieksekusi. Masalah ini juga dapat dihindari menggunakan Online Table Redefinition yang merupakan *feature* dari Oracle 9i.

6. Pengkompresan Sebuah VIEW

VIEW merupakan perwujudan dari sebuah query yang sangat panjang. VIEW juga dapat dikompres sama seperti halnya dengan tabel. Caranya:

```
CREATE MATERIALIZED VIEW ASAL_MAHASISWA
COMPRESS
AS SELECT M.NO_MHS, M.NAMA, K.NAMA_KOTA
FROM MAHASISWA M, KOTA K
WHERE M.KD_ASAL = K.KD_ASAL;
```

Sebuah VIEW yang didasarkan penggabungan beberapa tabel merupakan kandidat kuat untuk dilakukan sebuah pengkompresan, terutama pada VIEW yang jarang terjadi duplikasi item data. Kompresi sebuah VIEW dapat juga dilakukan dengan perintah ALTER MATERIALIZED VIEW.

```
ALTER MATERIALIZED VIEW ASAL_MAHASISWA
COMPRESS;
```

Ketika perintah ini digunakan, perhatikan bahwa kompresi hanya akan terjadi pada waktu VIEW tersebut di refresh kembali.

7. Pengkompresan Tabel yang Terpartisi

Untuk kompresi sebuah tabel yang terpartisi, ada banyak sekali pilihan untuk melakukannya. Maksudnya, kompresi bisa dilakukan pada level tabel atau juga pada level partisi. Misalnya, pada Listing 1 (Lampiran). Pembuatan tabel tersebut membentuk empat partisi. Ketika pernyataan COMPRESS diberikan, maka secara otomatis empat partisi tersebut akan terkompres. Dengan demikian, adanya keuntungan ini (kompresi pada level partisi), dapat digunakan sebagai pilihan untuk mengkompres sebagian partisi dan lainnya tetap tak terkompres. Contoh pada Listing 2 menunjukkan bagaimana melakukan kompresi pada level partisi

Pada Listing 2, tabel SALES_Q1_03 dan SALES_Q2_03 sudah terkompresi, dan dua yang lainnya tidak terkompresi. Perhatikan bahwa atribut kompresi untuk level partisi diberikan mengesampingkan definisi dari tabel-tabel unutk partisi tersebut. Jika atribut kompresi tidak diberikan untuk sebuah partisi, maka partisi akan mewarisi kondisi dari definisi level tabel. Juga pada Listing 2, ketika atribut kompresi tidak diberikan pada SALES_Q3_03 dan SALES_Q4_03, maka kedua partisi tersebut akan tetap dalam kondisi NOCOMPRESS, mengikuti kondisi definisi tabel.

Tabel yang terpartisi memberikan keuntungan yang lain ketika dalam kondisi terkompress. Salah satunya adalah kemampuan memisahkan operasi-operasi DML kedalam partisi yang terpisah dari read-only data. Contohnya, pada Listing 2, data sales terpartisi melalui SALE_DATE, semacam tiap bagian pada sales history yang partisinya terpisah. Pada conoh ini, data sales Q1 dan Q2 2003 tidak dapat dimodifikasi, oleh karenanya diletakkan dalam partisi SALES_Q1_03 dan SALES_Q2_03. Data sales pada Q3 dan Q4 masih bisa dimodifikasi, karena masih dalam kondisi tidak terkompres.

Jika pada bagian akhir dari Q3 2003, data pada SALES_Q3_03 menjadi read-only, partisi ini dapat dikompres menggunakan perintah ALTER TABLE ... MOVE PARTITION.

Untuk melihat kondisi partisi yang terkompres atau tidak, dapat digunakan perintah USER_TAB_PARTITIONS pada kamus data.

ISBN: 979-756-061-6

8. Pengujian dan Pembahasan

Alasan terbesar untuk menggunakan tabel terkompres adalah meningkatkan efisiensi penyimpanan. Sebuah tabel dengan kondisi terkompres akan memakan jauh lebih sedikit space dibandingkan dengan yang tidak terkompres. Untuk mengilustrasikannya, akan dicoba sebuah pengujian menggunakan dua tabel, MAHASISWA sebagai tabel tak terkompres, dan MAHASISWA_TEMP sebagai yang terkompres. Data sejumlah dua juta baris pada sebuah file berukuran kecil diisikan ke tabel ini melalui SQL*Loader. Ternyata setelah data semua dimasukkan, pada tabel yang terkompres hanya membutuhkan space separuh dari tabel yang terkompres. Hasil analisis ditunjukkan pada Listing 3 (Lampiran). Selain daripada itu, selain mengefisiensikan space penyimpanan, kompresi ini akan meningkatkan performance. Query pada tabel yang terkompres mampu diselesaikan oleh I/O lebih cepat, karena blok yang ada hanya sedikit. Untuk mengujinya, digunakan sebuah query pada kedua tabel tersebut, dan dilakukan pengecekan langkah per langkah (SQLTRACE/TKPROF). Analisis ditunjukkan pada Listing 4 (Lampiran). Hasilnya, query pada tabel terkompres menunjukkan operasi logical dan physical yang lebih sedikit dan juga lebih cepat.

9. Kesimpulan

Kompresi tabel yang diperkenal oleh Oracle 9i ini, sangat baik sekali digunakan, terutama untuk mengefisiensikan space penyimpanan dan meningkatkan performance dari query yang dibuat, walaupun untuk pemasukan data ke dalam tabel terkompres akan membutuhkan waktu lebih lama daripada tabel yang tidak terkompres. Ini dikarenakan operasi-operasi yang terjadi selama proses kompresi ketika data dimasukkan.

Referensi

[1] Hutabarat, Benaridho. I. (2004). Oracle 8i/9i: Performance Tuning. Yogyakarta: Andi.
[2] Mishra, Sanjay. (2004). http://www.oracle.com/technology/oramag/oracle/04-mar/o24tech_data.html.
[3] Sakti, Nufransa Wira. (1999). Belajar Sendiri menggunakan SQL *Plus.
[4] Millsap, Cary. (2004). http://www.oracle.com/technology/oramag/oracle/04-jan/o14tech_perf.html

LAMPIRAN

Listing 1: Kompresi semua Partisi

```
CREATE TABLE MAHASISWA (
  NO_MHS       VARCHAR2(8) NOT NULL,
  NAMA         VARCHAR2(50) NOT NULL,
  TGL_LAHIR    DATE NOT NULL,
  KD_ASAL      VARCHAR2(3) NOT NULL
)
COMPRESS
PARTITION BY RANGE (TGL_LAHIR) (
  PARTITION PERTAMA
    VALUES LESS THAN (TO_DATE('01-APR-1980','DD-MON-YYYY')),
  PARTITION KEDUA
    VALUES LESS THAN (TO_DATE('01-JUN-1981','DD-MON-YYYY')),
  PARTITION KETIGA
    VALUES LESS THAN (TO_DATE('01-OCT-1982','DD-MON-YYYY')),
  PARTITION KEEMPAT
    VALUES LESS THAN (TO_DATE('01-JAN-1983','DD-MON-YYYY'))
)
;
```

Listing 2: Kompresi sebagian Partisi

```
CREATE TABLE MAHASISWA (
  NO_MHS       VARCHAR2(8) NOT NULL,
  NAMA         VARCHAR2(50) NOT NULL,
  TGL_LAHIR    DATE NOT NULL,
  KD_ASAL      VARCHAR2(3) NOT NULL
)
COMPRESS
PARTITION BY RANGE (TGL_LAHIR) (
  PARTITION PERTAMA
    VALUES LESS THAN (TO_DATE('01-APR-1980','DD-MON-YYYY')) COMPRESS,
  PARTITION KEDUA
    VALUES LESS THAN (TO_DATE('01-JUN-1981','DD-MON-YYYY')) COMPRESS,
  PARTITION KETIGA
    VALUES LESS THAN (TO_DATE('01-OCT-1982','DD-MON-YYYY')),
  PARTITION KEEMPAT
    VALUES LESS THAN (TO_DATE('01-JAN-1983','DD-MON-YYYY'))
)
;
```

ISBN: 979-756-061-6

Listing 3: Analisa Efisiensi Penyimpanan

```
ANALYZE TABLE MAHASISWA COMPUTE STATISTICS;
ANALYZE TABLE MAHASISWA_TEMP COMPUTE STATISTICS;

SELECT TABLE_NAME, BLOCKS, NUM_ROWS, COMPRESSION
FROM USER_TABLES
WHERE TABLE_NAME LIKE 'MAHA%';

TABLE_NAME                BLOCKS   NUM_ROWS   COMPRESSION
------------------        ------  --------   -----------
MAHASISWA                  12137    1000000   DISABLED
MAHASISWA_TEMP              6188    1000000   ENABLED
```

Listing 4: Perbandingan Query Tabel terkompres dan tak Terkompres

Hasil TKPROF pada Tabel tak terkompresi:

```
SELECT SALE_DATE, COUNT(*) FROM MAHASISWA GROUP BY KD_ASAL;

call     count   cpu elapsed  disk      query    current   rows
------- ------  ---- -------  ----- ---------- ----------  -----
Parse        1  0.00    0.01     0          0          0      0
Execute      1  0.00    0.00     0          0          0      0
Fetch        2  5.22   13.76  10560      12148          0      1
------- ------  ---- -------  ----- ---------- ----------  -----
total        4  5.22   13.78  10560      12148          0      1
```

Hasil TKPROF pada Tabel terkompresi:

```
SELECT SALE_DATE, COUNT(*) FROM MAHASISWA_TEMP GROUP BY KD_ASAL;

call     count   cpu elapsed  disk      query    current   rows
------- ------  ---- -------  ----- ---------- ----------  -----
Parse        1  0.00    0.00     0          0          0      0
Execute      1  0.00    0.00     0          0          0      0
Fetch        2  5.27    7.20  6082       6091          0      1
------- ------  ---- -------  ----- ---------- ----------  -----
total        4  5.27    7.20  6082       6091          0      1
```