ORACLE
PCTFREE ve PCTUSED DEĞERLERİ

İçindekiler

1.Giriş

Bu çalışmada, PCTFREE ve PCTUSED değerleri üzerine testler yapacağız. Yapılan testler, özellikle yedek almak için yaratılan tablolar düşünülerek değerlendirilmelidir.

PCTFREE değeri bir veri block'unun sonraki update vb. işlemler için ne kadar boş yer bırakılacağını belirlemek için kullanılır. PCTFREE değerini bir trigger olarak düşünebiliriz. Örneğin PCTFREE değeri 10 olan bir blokta, blok %90 oranında doluysa, bu blok freelist'ten çıkartılır ve bloğa artık değer girilmez. Kalan %10'luk kısım ise sonra gerçekleşecek update işlemleri için rezerve edilir.

PCTUSED değeri ise bir tablonun hangi durumda boş sayılacağını belirler. Örneğin PCTUSED değeri 40 olan bir block için, %40'ın altında bir doluluk varsa (yani %60'ı boşsa), bu block freelist'e geri koyulur. Bunun anlamı bu block'a tekrar data girilebileceğidir. Doluluk oranı, block içindeki satırların silinmesi ya da daha az yer kaplayan değerlerle satırların update edilmesi sonucunda azalabilir.

2. Performans, Disk Alanına Karşı

Disk alanından maksimum yararlanmak için *PCTUSED* değerini yüksek verebiliriz. Bu değer ne kadar yüksekse, block freelist'te o kadar çok kalacaktır. Bu sayede aynı block'a daha çok satır yazılabilir. Fakat bu ekstra I/O anlamına gelir.

Düşük bir *PCTUSED* değeri ise block'un freelist'ten çabuk çıkmasına neden olur; bu da daha çok block kullanılmasına ve dolayısıyla yer israfına neden olacaktır. Fakat Oracle için yeni block oluşturmak, block kontrolü yapmadan daha az maliyetlidir. Dolaysıyla performans artışı sağlanacaktır.

Tabloların *PCTUSED* ve *PCTFREE* değerleriyle dinamik olarak oynayabiliriz:

```
SQL> ALTER TABLE D_CCEBI.DENEME PCTFREE 10 PCTUSED 80;
```

Yapacağınız güncelleme *DBA_TABLES* view'ına hemen yansımaz. Kontrol etmek için aşağıdaki sorguyu kullanmanız daha iyi olur:

```
-- DEGISTIRILEN PCTFREE VE PCTUSED DEGERLERINI GORMEK ICIN
SELECT  S.PCTFREE$ PCT_FREE, S.PCTUSED$ PCT_USED, T.*
FROM    SYS.TAB$ S, DBA_OBJECTS O, DBA_TABLES T
WHERE   S.OBJ#=O.OBJECT_ID AND T.TABLE_NAME=O.OBJECT_NAME AND
        O.OWNER='D_CCEBI' AND O.OBJECT_NAME='DENEME';
```

Bizim yapacağımız testlerde *PCTUSED* değerinin önemi olmayacak. Çünkü yapacağımız testlerde, yedek işlemlerinde kullanılmak üzere tablo oluşturmaktan bahsedeceğiz. Bu tablolara daha sonra update ve delete işlemleri yapılacak olsa, PCTUSED değerini de hesaba katmak gerekecekti. Fakat böyle bir durumun olmadığını kabul ediyoruz. Tablo bir kere yaratılacak ve o şekilde kalacak diye düşünüyoruz. (Aslında tablonun sabit kalması gerekmiyor; sadece Insert işlemlerinin yapıldığı log tabloları için de aynı durum geçerlidir.)

3.Testler

Yapacağımız testlerde kullanacağımız tablo, 1.06GB boyutunda. Default olarak PCTFREE değeri 10; PCTUSED değeri 40 olarak ayarlanmış. Bu tabloyu yedekleyerek farklı kombinasyonlar deneyeceğiz.

TEST - 1 : PCTFREE 10 ve PCTUSED 40 (*NOCOMPRESS*)
Önce compress olmaksızın, tablonun birebir özelliklerine sahip bir kopyasını oluşturuyoruz:

```
SET TIMING ON
CREATE TABLE D_CCEBI.DENEME1
TABLESPACE DATA_TS
NOLOGGING
NOCOMPRESS
PARALLEL (DEGREE 8)
PCTFREE 10
PCTUSED 40
AS
SELECT * FROM D_CCEBI.KAYNAK;
Table created.
Elapsed: 00:01:46.48
```

Kopya tablo boyutu 1.06 GB. (Bu değer KAYNAK tablosuyla aynı.) Blok sayısı, 138.800. (Zaten 8K ile çarparsak 1.06GB çıkıyor ve bu da normal.) Tabloya full select cost'u ise 4297.

TEST-2 : PCTFREE 0 ve PCTUSED 95 (*NOCOMPRESS*)
Şimdi de tablonun hiç update edilmeyeceğini düşünerek, PCTFREE değerini 0 veriyoruz. PCTUSED değerini de 95 verelim. Aslında PCTUSED değerinin bir önemi yok; çünkü tabloya girilen satırlar bir daha silinmeyecek. Satır silinmeyeceği için freelist'e tekrar düşmesi de söz konusu değil. Freelist'e hiç düşmeyecek bir tablonun PCTUSED değerinin pek anlamı kalmıyor.

```
SET TIMING ON
CREATE TABLE D_CCEBI.DENEME2
TABLESPACE DATA_TS
NOLOGGING
NOCOMPRESS
PARALLEL (DEGREE 8)
PCTFREE 0
PCTUSED 95
AS
SELECT * FROM D_CCEBI.KAYNAK;
Table created.
Elapsed: 00:02:16.14
```

Süre farkettiğiniz gibi uzadı. Testi iki defa yaptım; ikisi de sonucu böyle çıkarttı. Update işlemlerinde var olan block'ların kullanılması yerine, yeni bir block extend etmek Oracle için daha kolay. Sıfır bir tabloda da bu kural geçerli olabilir.

Boyut olarak 976 MB'a düştük. Herhangi bir compress işlemi uygulamadan 109MB (yaklaşık %10 civarı) kazanç elde ettik. Elde edeceğimiz kazanç beklenen bir değer. Çünkü her bloğun %10'unu boş bırakmak yerine hepsini doldurursanız, %10 yerden tasarruf sağlamanız normal. Block sayımız da 138.800'den 124.904'e (%10 civarı) düşüyor.

Tabloya full select cost'u 3875.

TEST-3 : PCTFREE 10 ve PCTUSED 40 (*COMPRESS*)

```
SET TIMING ON
CREATE TABLE D_CCEBI.DENEME3
TABLESPACE DATA_TS
NOLOGGING
COMPRESS
PARALLEL (DEGREE 8)
PCTFREE 10
PCTUSED 40
AS
SELECT * FROM D_CCEBI.KAYNAK;
Table created.
Elapsed: 00:02:14.93
```

Tablo boyutu 452 Mb'a düştü. Kullanılan block sayısı ise 57.808. Compress'in faydalarını daha önce de biliyoruz; o yüzden sürpriz bir sonuç olmadı. Süre compress işlemine göre biraz uzun; daha düşük çıkmasını bekliyordum. O sırada istatistik geçmemin etkisi olabilir. Tabloya full select cost'u 1837.

TEST-4 : PCTFREE 0 ve PCTUSED 95 (*COMPRESS*)

COMPRESS tablolardaki yapıyı ele almadan önce bazı şeyleri gözönüne almak gerekiyor. Bir tabloda PCTFREE değeri 10 olursa, bir block %90'a kadar doldurulabilir. Kalan %10'luk kısım update işlemleri için rezerver edilir. Fakat compress alıştığımız anlamda satırlara sahip değil. Aynı block içindeki aynı veriye sahip satır ve sütunlar, symbol table olarak adlandırılan datablock'un basına kaydedilir. Sıkıştırılmamış bir tabloda satırı değiştirdiğinizde %10'luk alan yeterli olabilir, yeni block açılması belki icap etmez. Fakat sıkıştırılmış (compressed) tabloda, daha önce sadece bir işaretçi ile gösterilen satır, artık uncompress edilmeli ve bu şekilde saklanmalıdır. Bu yüzden update gören tablolarda compress tavsiye edilmiyor. Çünkü sık update'in döndüğü bir ortamda sıkıştırma yaparsanız, hem performans hem de yerden kaybedersiniz.

Denememizi yaparsak;

```
SET TIMING ON
CREATE TABLE D_CCEBI.DENEME4
TABLESPACE DATA_TS
NOLOGGING
COMPRESS
PARALLEL (DEGREE 8)
PCTFREE 0
PCTUSED 95
AS
SELECT * FROM D_CCEBI.KAYNAK;
Table created.
Elapsed: 00:02:03.65
```

Tablo 406 Mb'a düştü. 52000 block söz konusu. Kazanım yine %10 civarında. Emin olmamakla birlikte tekrarın çok daha sık olduğu sıkıştırılmış bir tabloda PCTFREE değerinin daha da etkin olacağını düşünüyorum. Bunu şuna dayandırıyorum:

Sıkıştırılmış tablolarda, her block içinde bir symbol table var. '**A**' block'unda toplamda 200 adet satır olsun. Bu 200 satırda, birbirinden farklı 10 adet satır olduğunu düşünelim. Bu 10 satır kaydedilir ve Symbol table kullanılarak diğer satırlar, bu satırları refere eder. '**B**' block'unda da 5 adet farklı olmak üzere toplam 200 satır olduğunu düşünelim. Eğer 5 adet farklı satırı, '**A**' block'unun symbol table'ında yazacak kadar yerimiz olsaydı, '**B**' block'una hiç gerek kalmayabilirdi. *PCTFREE* değeri ne kadar düşük olursa, symbol table daha çok farklı değer taşıyan satırı refere

edebilecektir. Böyle bir durumda da sıkıştırmanın daha etkin olacağını düşünüyorum. (Bunları test etmedim, yanılıyor olabilirim.) Tabloya full select cost'u 1660.

TEST-5 : PCTFREE 0 ve PCTUSED 95 (*COMPRESS ve INITIAL EXTENT*)
Burada oyunun kurallarını performans için biraz bozuyoruz. Daha önceki testlerde tablonun yapısını bilmediğimizi düşünerek default bir çalışma yaptık. Şimdi ise compress edilerek yaratılacak tablonun 400 MB civarında olacağını ve ortalama satır boyutunun 100KB civarında olacağını bildiğimizi düşünelim.

```
SET TIMING ON
CREATE TABLE D_CCEBI.DENEME5
TABLESPACE DATA_TS
NOLOGGING
COMPRESS
STORAGE (INITIAL 200M NEXT 64K)
PARALLEL (DEGREE 8)
PCTFREE 0
PCTUSED 95
AS
SELECT * FROM D_CCEBI.KAYNAK;
Table created.
Elapsed: 00:01:43.42
```

Dikkat ettiğiniz gibi süre daha kısa oldu. Boyut yine 406 MB. Tabloya full select cost'u 1660.

TEST-6 : PCTFREE 0 NOCOMPRESS TABLODA UPDATE
Test amaçlı bütün tablolarımızda *KAYIT_ACIKLAMA* sütunu 120 karakterlik varchar2 tipli bir alan var. Çoğu satırda bu sütunun 20-30 karakterden fazlası kullanılmıyor. Bir update ile bütün satırları güncelliyoruz.

Önce PCTFREE değeri 10 olan, standart yapılı *DENEME1* tablosunda işlemi gerçekleştiriyoruz:

```
UPDATE D_CCEBI.DENEME SET KAYIT_ACIKLAMA='Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Nam imperdiet condimentum quam. Aliquam sed
quam viverra fusce';
commit;
```

Şimdi de PCTFREE değerinin 0 olduğu sıkıştırılmamış *DENEME2* tablosunda bu denemeyi yapıyoruz:

```
UPDATE D_CCEBI.DENEME2 SET KAYIT_ACIKLAMA='Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Nam imperdiet condimentum quam. Aliquam sed
quam viverra fusce';
commit;
```

Kayıt sayısı her iki tabloda da 10 milyon üzerinde. Update işlemi her iki tablo içinde 1 saat 16 dakikada tamamlanıyor. İşlem sonucunda PCTFREE değeri 0 olan tablonun daha büyük boyuta erişeceğini düşünüyordum. Fakat sonuç beklediğimden farklı oldu.

Update sonrası, PCTFREE değeri 10 olan tablonun boyutu 2.07 GB olmuş. PCTFREE değeri 0 olan tablo ise 1.99GB. Hâlbuki tam tersinin olması mantıklı. Belki bütün satırları update etmek yerine belirli bir kısmını update etseydik durum farklı olabilirdi. Ya da update esnasında kullandığımız ifade 120 karakter yerine daha az sayıda olsaydı, sonuçlar farklı olabilirdi. Aşağıda bir başka deneme yaptım:

```
CREATE TABLE D_CCEBI.OBJS_1
TABLESPACE DATA_TS
NOLOGGING
NOCOMPRESS
PCTFREE 10
PCTUSED 40
AS
SELECT * FROM DBA_OBJECTS;

CREATE TABLE D_CCEBI.OBJS_2
TABLESPACE DATA_TS
NOLOGGING
NOCOMPRESS
PCTFREE 0
PCTUSED 95
AS
SELECT * FROM DBA_OBJECTS;

UPDATE D_CCEBI.OBJS_2
SET SUBOBJECT_NAME='BU BIR DENEMEDIR.',OWNER='DENEME OWNER'
WHERE OBJECT_ID < 10000;
UPDATE D_CCEBI.OBJS_1
SET SUBOBJECT_NAME='BU BIR DENEMEDIR.',OWNER='DENEME OWNER'
WHERE OBJECT_ID < 10000;
COMMIT;
```

OBJS_1 tablosunun update öncesi ve sonrası bir farkı olmadı. Daha önce 8192KB iken; update sonrası yine 8192KB'da kaldı. OBJS_2 tablosu ise update öncesi 7168 KB iken, update sonrası 8192 KB'a çıktı. PCTFREE değeri ufak olan tablonun daha fazla büyüyebileceğini gördük. Ancak her zaman böyle olmuyor. Yapılan update'in sütunun ne kadarını işgal edeceği de önemli.

4. Test Sonuçlarını Yorumlamak

Yaptığımız testlerin sonuçlarını derlersek;

1. Yedek alınacak bir tabloda Update işleminin kesinlikle olmayacağından emin olabiliyorsanız; PCTFREE değerini sıfır ya da sıfıra çok yakın vererek alan kazanabilirsiniz. PCTFREE değerinin az olması daha az block kullanılması, yerden daha çok tasarruf ve sorgu cost'larının düşmesi anlamına geliyor.
2. Compress her zaman büyük kâr sağlayan bir özelliktir. Ancak update söz konusu ise, astarı yüzünden pahalıya gelen bir durumla karşılaşırız.
3. Tabloya dair ön bilginiz varsa (–ki istatistik özelliklerinden tabloyabilirsiniz) initial extent belirtmeniz size fayda sağlayacaktır. Next değerini ise ortalama satır boyutuna uyacak şekilde belirtebilirsiniz. Altı bir değer extent işleminin sık tekrarına neden olur. Üstü bir değer ise yer israfına neden olur.
4. Bu test yedek tablolarıyla ilgiliydi. PCTFREE'yi 0 yapabilmemiz bundan kaynaklanıyor. Update varsa, bu hiç akıllıca bir çözüm olmayacaktır. Her update yeni block'larla sonuçlanabilir.