VIEW'LERDE SQL HINT KULLANIMI

1. Çalışma Verisi Hazırlama

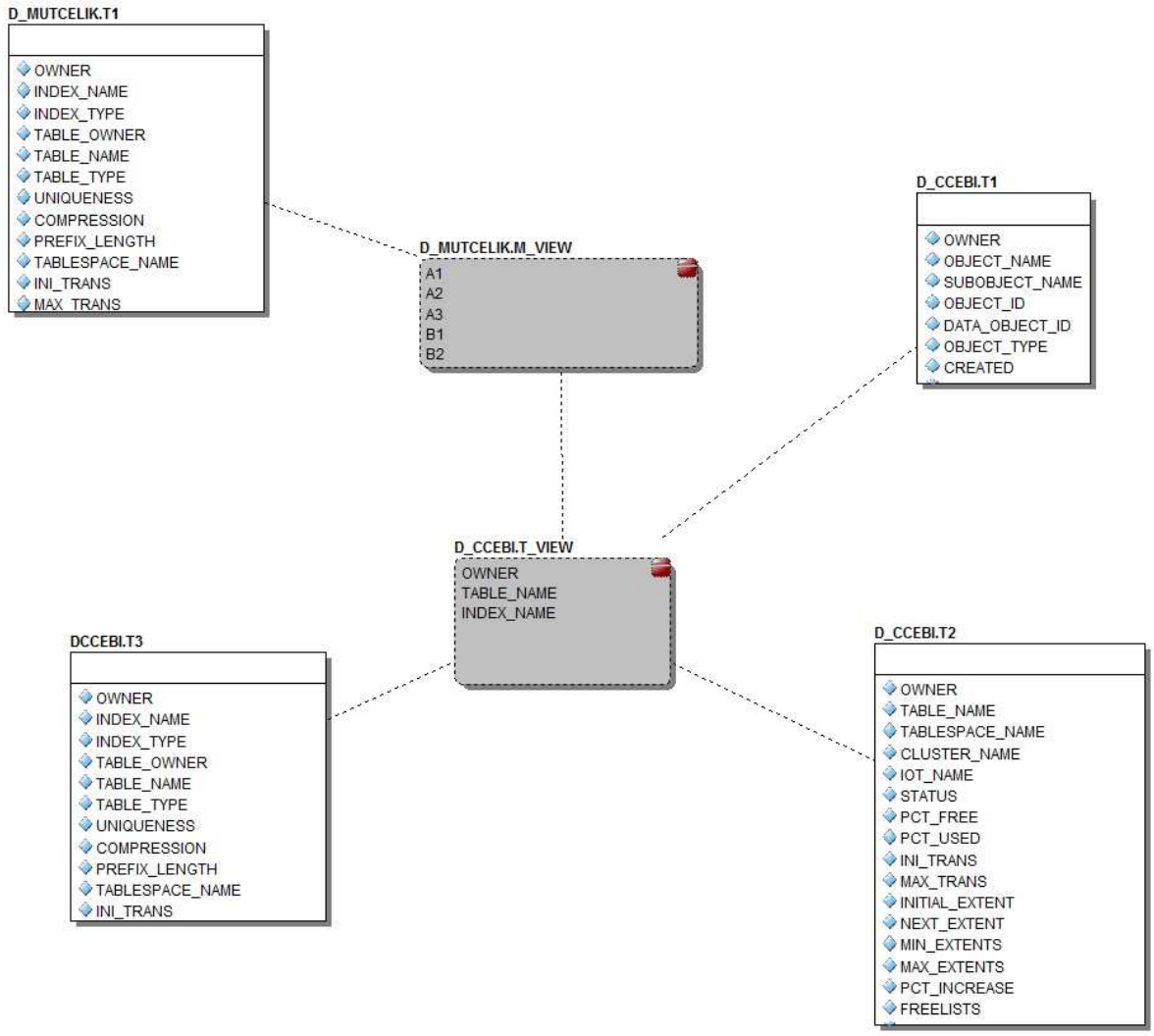
View'larda hint kullanımı problemli olabiliyor. Oracle kompleks yapıdaki view'lerin içine 'default' olarak hint'i yaymıyor. Kompleks view'larda hint'leri kullanabilmek için global ya da local hint kullanma yöntemleri var. Local hint yöntemi view yazılırken sorguda hint kullanılması şeklinde oluyor. Global hint kullanımını ise yazı içerisinde göreceğiz.

Global index denemesi için örnek bazı tablo ve view'lar yaratıyoruz:

```
-- D_CCEBI.T1 TABLOSU
CREATE TABLE D_CCEBI.T1
NOLOGGING
TABLESPACE TB_DATA
AS SELECT * FROM DBA_OBJECTS;
CREATE INDEX D_CCEBI.T1_OBJINFO_IDX
ON D_CCEBI.T1( OWNER, OBJECT_NAME )
TABLESPACE TB_INDEX;


-- D_CCEBI.T2 TABLOSU
CREATE TABLE D_CCEBI.T2
NOLOGGING
TABLESPACE TB_DATA
AS SELECT * FROM DBA_TABLES;
CREATE INDEX D_CCEBI.T2_TBLINFO_IDX
ON D_CCEBI.T2( OWNER, TABLE_NAME )
TABLESPACE TB_INDEX;


-- D_CCEBI.T3 TABLOSU
CREATE TABLE D_CCEBI.T3
NOLOGGING
TABLESPACE TB_DATA
AS SELECT * FROM DBA_INDEXES;
CREATE INDEX D_CCEBI.T3_INDXINFO_IDX
ON D_CCEBI.T3( OWNER, TABLE_NAME )
TABLESPACE TB_INDEX;


-- D_CCEBI.T_VIEW (D_CCEBI.T1, D_CCEBI.T2 VE D_CCEBI.T3 BIRLESIMI)
CREATE OR REPLACE VIEW D_CCEBI.T_VIEW
AS
SELECT  T1.OWNER, T2.TABLE_NAME, T3.INDEX_NAME
FROM    D_CCEBI.T1, D_CCEBI.T2, D_CCEBI.T3
WHERE   T1.OWNER = T2.OWNER AND T1.OBJECT_NAME = T2.TABLE_NAME AND
        T2.OWNER = T3.OWNER AND  T2.TABLE_NAME = T3.TABLE_NAME;


-- D_MUTCELIK.T1 TABLOSU
CREATE TABLE D_MUTCELIK.T1
NOLOGGING
TABLESPACE TB_DATA
AS SELECT * FROM DBA_INDEXES;
CREATE INDEX D_MUTCELIK.T1_INDXINFO_IDX
ON D_MUTCELIK.T1( OWNER, TABLE_NAME )
TABLESPACE TB_INDEX;
```

```
-- D_MUTCELIK KULLANICISINA D_CCEBI.T_VIEW ICIN SELECT HAKKI VERILIYOR
GRANT SELECT ON D_CCEBI.T_VIEW TO D_MUTCELIK;

-- D_MUTCELIK.M_VIEW (D_MUTCELIK.T1 VE D_CCEBI.T_VIEW BIRLESIMI)
CREATE OR REPLACE VIEW D_MUTCELIK.M_VIEW
AS
SELECT  D.OWNER AS A1, D.TABLE_NAME AS A2, D.INDEX_NAME AS A3,
        T.OWNER AS B1, T.TABLE_NAME AS B2, T.INDEX_NAME AS B3
FROM    D_MUTCELIK.T1 D, D_CCEBI.T_VIEW T
WHERE   D.OWNER = T.OWNER AND D.TABLE_NAME = T.TABLE_NAME;
```

Tablo ve view'lar arasındaki ilişkiyi aşağıda görebilirsiniz:



(Dökümana sığması için tabloları biraz ufalttım. Normalde hepsinde 50 civarı sütun var.)

2. Hint verilerek View'larda Execution Plan'ın Değiştirilmesi

Optimizer elindeki verilerden (istatistik, histogram, tablo boyutu, index'lerin yapısı vs...) yola çıkarak, maliyeti (cost) en düşük yolu tercih eder. Ama maliyetin düşük olması, her zaman doğru planı kullanmayı garantilemez. Bazı durumlarda düşük cost'a rağmen yanlış

bir plan seçilmiş olabilir ve bu da ifadenin çok daha uzun süre çalışmasına neden olur. Böyle olursa SQL Hint'lerle ifadeye müdahele etmemiz gerekebilir. Hint'ler sayesinde öyle çalışma, böyle çalış; index kullanma, tabloya full git; ya da full yerine index kullan gibi çeşit çeşit belirtme yapabiliriz. Yalnız hint'leri view'larla kullanmak daha zordur. Çünkü siz belirtmediğiniz sürece, SQL hint'lerin alt nesnelere (tablo ya da view'lara) nüfuz etme kabiliyeti yoktur.

Şimdi hint'lerin etkisiyle ilgili bazı testler yapacağız. Önce herhangi bir müdahelede bulunmadan planın ilk hâline bakıyoruz:

```
SELECT * FROM D_MUTCELIK.M_VIEW;
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation               | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT        |                |    37 |  8103 |   179   (3)| 00:00:03 |
|*  1 |  HASH JOIN              |                |    37 |  8103 |   179   (3)| 00:00:03 |
|*  2 |   HASH JOIN             |                |  1105 |   146K|    81   (3)| 00:00:01 |
|*  3 |    HASH JOIN            |                |  1521 |   126K|    46   (3)| 00:00:01 |
|   4 |     TABLE ACCESS FULL   | T3             |  4550 |   226K|    34   (0)| 00:00:01 |
|   5 |     INDEX FAST FULL SCAN| T2_TBLINFO_IDX |  6334 |   210K|    11   (0)| 00:00:01 |
|   6 |    TABLE ACCESS FULL    | T1             |  4480 |   223K|    34   (0)| 00:00:01 |
|   7 |   INDEX FAST FULL SCAN  | T1_OBJINFO_IDX | 74859 |  6067K|    97   (2)| 00:00:02 |
-------------------------------------------------------------------------------------------
```

Hint vererek bu planı değiştirmeye çalışalım:

```
SELECT /*+INDEX( M_VIEW T3_INDXINFO_IDX) */
* FROM D_MUTCELIK.M_VIEW;
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation               | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT        |                |    37 |  8103 |   179   (3)| 00:00:03 |
|*  1 |  HASH JOIN              |                |    37 |  8103 |   179   (3)| 00:00:03 |
|*  2 |   HASH JOIN             |                |  1105 |   146K|    81   (3)| 00:00:01 |
|*  3 |    HASH JOIN            |                |  1521 |   126K|    46   (3)| 00:00:01 |
|   4 |     TABLE ACCESS FULL   | T3             |  4550 |   226K|    34   (0)| 00:00:01 |
|   5 |     INDEX FAST FULL SCAN| T2_TBLINFO_IDX |  6334 |   210K|    11   (0)| 00:00:01 |
|   6 |    TABLE ACCESS FULL    | T1             |  4480 |   223K|    34   (0)| 00:00:01 |
|   7 |   INDEX FAST FULL SCAN  | T1_OBJINFO_IDX | 74859 |  6067K|    97   (2)| 00:00:02 |
-------------------------------------------------------------------------------------------
```

Plan değişmiyor. Çünkü hint'in kompleks yapılarda aşağıya işleme imkanı yok(muş). Fakat aşağıdaki gibi yazarsanız, hint'leri kabul ettirebiliyorsunuz:

```
SELECT
/*+INDEX( M_VIEW.D T1_INDXINFO_IDX)
FULL( M_VIEW.T.T1)
FULL( M_VIEW.T.T2 )
INDEX( M_VIEW.T.T3 T3_INDXINFO_IDX)
*/ * FROM D_MUTCELIK.M_VIEW;
```

```
-----------------------------------------------------------------------------------------------------
| Id  | Operation                          | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |                 |    37 |  8103 |  2492   (1)| 00:00:30 |
|   1 |  TABLE ACCESS BY INDEX ROWID       | T3              |     1 |    51 |     2   (0)| 00:00:01 |
|   2 |   NESTED LOOPS                     |                 |    37 |  8103 |  2492   (1)| 00:00:30 |
|*  3 |    HASH JOIN                       |                 |   154 | 25872 |  2337   (1)| 00:00:29 |
|*  4 |     HASH JOIN                      |                 |  4601 |   381K|  2104   (1)| 00:00:26 |
|   5 |      TABLE ACCESS BY INDEX ROWID   | T1              |  4480 |   223K|  2051   (1)| 00:00:25 |
|   6 |       INDEX FULL SCAN              | T1_INDXINFO_IDX |  4480 |       |    29   (0)| 00:00:01 |
|   7 |      TABLE ACCESS FULL             | T2              |  6334 |   210K|    52   (0)| 00:00:01 |
|   8 |     TABLE ACCESS FULL              | T1              | 74859 |  6067K|   232   (1)| 00:00:03 |
|*  9 |    INDEX RANGE SCAN                | T3_INDXINFO_IDX |     2 |       |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------------
```

Açıklamak için INDEX(**M_VIEW.T.T3 T3_INDXINFO_IDX**) hint'inden yola çıkalım. Aslında hint ile yaptığımız işlem şöyle oluyor: M_VIEW içinde T diye bir view ya da tablo var. T diye anılan tablo ya da view içinde T3 diye bir tablo var. T3 tablosunda T3_INDXINFO_IDX diye bir index var; SQL'i çalıştıracağın zaman bunu kullan.

Burada T olmasının nedeni, M_VIEW'i tanımlarken alias kullanmamızdan kaynaklanıyor. D_MUTCELIK.M_VIEW'i hatırlarsak;

```
CREATE OR REPLACE VIEW D_MUTCELIK.M_VIEW
AS
SELECT  D.OWNER AS A1, D.TABLE_NAME AS A2, D.INDEX_NAME AS A3,
        T.OWNER AS B1, T.TABLE_NAME AS B2, T.INDEX_NAME AS B3
FROM    D_MUTCELIK.T1 D, MARDATA.T_VIEW T
WHERE   D.OWNER = T.OWNER AND D.TABLE_NAME = T.TABLE_NAME;
```

Hint'lerde tablo/view isimleri değil, alias tanımları önemli oluyor. Eğer view içindeki alt tablo/view'lara yönelik hint'leri kullanacaksanız, alias'lara göre yazmanız gerekiyor.

Son olarak belirtmemde fayda var; Oracle Global Index'leri tercih etmemizi öneriyor. Fakat view içinde view, onun içinde bir başka view, sonra yine bir alt view gibi durumlarda, bunu yapmak gerçekten güç. Bu yüzden view'ları olabildiğince basit tutup, içiçe yapılardan sakınmak hem optimizer hem de bizlerin işini kolaylaştıracaktır.

3. Kaynak

- Oracle9i Database Performance Tuning Guide and Reference Release 2 / Optimizer Hints
  http://download.oracle.com/docs/cd/B10501_01/server.920/a96533/hintsref.htm