Rapport
TP 6

Optimisation par vues matérialisées

Réalisé en binôme par :
BENSALAH Kawthar / ABBACI Khaled
Numero du binôme : 22
Master 2 IL - Groupe 1
USTHB 2019/2020

TP 6

Objectif du TP

Optimisation par vues matérialisées

Introduction

À travers ce TP, nous allons comprendre l'utilité de l'implémentation des métadonnées des dimensions sous forme de vues matérialisées dans l'optimisation des requêtes.

Réponse 1

- **Activation des options autotrace et timing de oracle :**

```
SQL> set timing on;
SQL> set autotrace on explain;
SQL>
```

- **Vider les buffers :**

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.22
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.04
SQL>
```

- **Ecriture et exécution de la requête R1 qui donne le nombre d'opérations réalisées dans les agences de la wilaya d'Alger :**

```
SQL> select DA.CodeWilaya,
  2          sum(FO.NbOperationR+FO.NbOperationV)
  3          as NbOperation
  4     from FOperation FO, DAgence DA
  5    where FO.NumAgence = DA.NumAgence
  6    and DA.CodeWilaya = 16
  7    group by DA.CodeWilaya;

CODEWILAYA NBOPERATION
---------- -----------
        16        8982
```

- **Examination du temps d'exécution :**

```
Ecoulé : 00 :00 :00.35
```

- **Examination du plan d'exécution :**

```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 1233101489

-------------------------------------------------------------------------------
---
| Id  | Operation            | Name       | Rows  | Bytes | Cost (%CPU)| Time
  |
-------------------------------------------------------------------------------
---
|   0 | SELECT STATEMENT     |            |     1 |    19 |   929   (2)| 00:00:1
2 |
|   1 |  SORT GROUP BY NOSORT|            |     1 |    19 |   929   (2)| 00:00:1
2 |
|*  2 |   HASH JOIN          |            | 12701 |   235K|   929   (2)| 00:00:1
2 |
|*  3 |    TABLE ACCESS FULL | DAGENCE    |   256 |  2048 |    30   (0)| 00:00:0
1 |
|   4 |    TABLE ACCESS FULL | FOPERATION |   609K|  6549K|   896   (2)| 00:00:1
1 |
-------------------------------------------------------------------------------
---

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("FO"."NUMAGENCE"="DA"."NUMAGENCE")
   3 - filter("DA"."CODEWILAYA"=16)

SQL>
```

Remarque 1 : On remarque l'exploitation de la dimension DAgence et du fait FOperation pendant l'exécution de la requête.

Réponse 2

- **Création de la vue matérialisée VMWilaya :**

```
SQL> CREATE MATERIALIZED VIEW VMWilaya
  2       BUILD IMMEDIATE REFRESH COMPLETE ON DEMAND
  3       enable query rewrite
  4       AS select DA.CodeWilaya, DA.NomWilaya,
  5                 sum(FO.NbOperationR+FO.NbOperationV) as NbOperation
  6          from DAgence DA, FOperation FO
  7         where FO.NumAgence = DA.NumAgence
  8         group by DA.CodeWilaya, DA.NomWilaya
  9         order by DA.CodeWilaya;

Vue matérialisée créée.

Ecoulé : 00 :00 :01.13
SQL>
```

Réponse 3

- **Vider les buffers :**

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.25
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.02
SQL>
```

- **Réexécution de la requête R1 :**

```
SQL> select DA.CodeWilaya,
  2          sum(FO.NbOperationR+FO.NbOperationV)
  3          as NbOperation
  4    from FOperation FO, DAgence DA
  5   where FO.NumAgence = DA.NumAgence
  6   and DA.CodeWilaya = 16
  7   group by DA.CodeWilaya;

CODEWILAYA NBOPERATION
---------- -----------
        16        8982
```

- **Examination du temps d'exécution :**

```
Ecoulé : 00 :00 :00.13
```

Remarque 2 : On remarque une baisse du temps d'exécution de la requête R1 (00:00:00.13) par rapport à celui examiné avant la création de VMWilaya (00:00:00.35).

- **Examination du plan d'exécution :**

```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 3295964862

-------------------------------------------------------------------------------------
----------

| Id  | Operation                     | Name     | Rows  | Bytes | Cost (%CPU)|
Time     |

-------------------------------------------------------------------------------------
----------

|   0 | SELECT STATEMENT              |          |     1 |    26 |     3   (0)|
00:00:01 |

|   1 |  SORT GROUP BY NOSORT         |          |     1 |    26 |     3   (0)|
00:00:01 |

|*  2 |   MAT_VIEW REWRITE ACCESS FULL| VMWILAYA |     1 |    26 |     3   (0)|
00:00:01 |

-------------------------------------------------------------------------------------
----------


Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter("VMWILAYA"."CODEWILAYA"=16)

Note
-----
   - dynamic sampling used for this statement (level=2)

SQL>
```

Remarque 3: On remarque bien que l'optimiseur exploite la vue matérialisée VMWilaya au lieu de DAgence et FOperation pendant l'exécution de la requête.

Conclusion 1 : L'exploitation de la vue matérialisée VMWilaya par l'optimiseur a réduit le temps d'exécution de la requête R1.

Réponse 4

- Création de la vue matérialisée VMontantVMensuel :

```
SQL> CREATE MATERIALIZED VIEW VMMontantVMensuel
  2       BUILD IMMEDIATE REFRESH COMPLETE ON DEMAND
  3       enable query rewrite
  4       AS select DT.Mois,
  5                 sum(FO.MontantV) as MVMensuel
  6          from DTemps DT, FOperation FO
  7         where FO.CodeTemps = DT.CodeTemps
  8         group by DT.Mois
  9         order by DT.Mois;

Vue matérialisée créée.

Ecoulé : 00 :00 :00.99
SQL>
```

Réponse 5

- **Vider les buffers :**

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.24
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.04
SQL>
```

- **Ecriture et exécution de la requête R2 qui donne les montants versés annuels :**

```
SQL> select DT.Année,
  2         sum(FO.MontantV) as MVAnnuel
  3  from DTemps DT, FOperation FO
  4  where FO.CodeTemps = DT.CodeTemps
  5  group by DT.Année
  6  order by DT.Année;

ANNÉE             MVANNUEL
--------------- ----------
2015            4226461684
2016            4210595166
2017            4219235397
2018            4230932776
```

- **Examination du temps d'exécution :**

```
Ecoulé : 00 :00 :00.53
```

- **Examination du plan d'exécution :**

```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 2545120435

-------------------------------------------------------------------------------
-----
| Id  | Operation              | Name       | Rows  | Bytes | Cost (%CPU)| Time
    |
-------------------------------------------------------------------------------
-----
|   0 | SELECT STATEMENT       |            |     4 |   104 |   931   (5)| 00:00
:12 |
|   1 |  SORT ORDER BY         |            |     4 |   104 |   931   (5)| 00:00
:12 |
|   2 |   HASH GROUP BY        |            |     4 |   104 |   931   (5)| 00:00
:12 |
|*  3 |    HASH JOIN           |            |  1460 | 37960 |   929   (4)| 00:00
:12 |
|   4 |     TABLE ACCESS FULL  | DTEMPS     |  1460 | 13140 |     5   (0)| 00:00
:01 |
|   5 |     VIEW               | VW_GBC_5   |  1460 | 24820 |   924   (5)| 00:00
:12 |
|   6 |      HASH GROUP BY     |            |  1460 | 13140 |   924   (5)| 00:00
:12 |
|   7 |       TABLE ACCESS FULL| FOPERATION |   609K|  5358K|   897   (2)| 00:00
:11 |
-------------------------------------------------------------------------------
-----
```

```
Predicate Information (identified by operation id):
---------------------------------------------------

   3 - access("ITEM_1"="DT"."CODETEMPS")
```

Remarque 4 : Après avoir analyser le plan d'exécution de la requête R2, On remarque que la vue matérialisée VMMontantVMensuel n'a pas été exploitée car les attributs **Année** et **Mois** sont considérés comme des chaînes de caractères uniquement sans qu'il y ait de lien hiérarchique entre eux.

Réponse 6

- **Création des métadonnées de la dimension DTemps :**

```
SQL> create dimension DIMTemps
  2  level NTemps1 is (DTemps.CodeTemps)
  3  level NTemps2 is (DTemps.Mois)
  4  level NTemps3 is (DTemps.Année)
  5  hierarchy HTemps1 (NTemps1 child of NTemps2 child of NTemps3)
  6  attribute NTemps1 determines (DTemps.Jour, DTemps.LibJour)
  7  attribute NTemps2 determines (DTemps.LibMois)
  8  ;

Dimension créée.

Ecoulé : 00 :00 :00.09
SQL>
```

- **Création des métadonnées de la dimension DAgence :**

```
SQL> create dimension DIMAgence
  2  LEVEL NAgence1 IS DAgence.NumAgence
  3  LEVEL NAgence2 IS DAgence.CodeBanque
  4  LEVEL NAgence3 IS DAgence.CodeVille
  5  LEVEL NAgence4 IS DAgence.CodeWilaya
  6  HIERARCHY HAgence1(NAgence1 child of NAgence2 CHILD OF NAgence3  CHILD OF NAgence4)
  7  ATTRIBUTE NAgence1 DETERMINES DAgence.NomAgence
  8  ATTRIBUTE NAgence2 DETERMINES DAgence.NomBanque
  9  ATTRIBUTE NAgence3 DETERMINES DAgence.NomVille
 10  ATTRIBUTE NAgence4 DETERMINES DAgence.NomWilaya;

Dimension créée.

Ecoulé : 00 :00 :00.01
SQL>
```

- **Création des métadonnées de la dimension DClient :**

```
SQL> create dimension DIMClient
  2  LEVEL NClient1 IS DClient.NumClient
  3  ATTRIBUTE NClient1 determines (DClient.NomClient, DClient.DNClient);

Dimension créée.

Ecoulé : 00 :00 :00.01
SQL>
```

- **Création des métadonnées de la dimension DTypeCompte :**

```
SQL> create dimension DIMTypeCompte
  2  level NTypeCompte1 is (DTypeCompte.CodeType)
  3  attribute NTypeCompte1 determines (DTypeCompte.LibType);

Dimension créée.

Ecoulé : 00 :00 :00.01
SQL>
```

Réponse 7

- **Modification de la session afin de permettre l'exploitation des dimensions dans l'amélioration des temps d'exécutions :**

```
SQL> Alter session set query_rewrite_integrity = trusted;

Session modifiée.

Ecoulé : 00 :00 :00.01
SQL>
```

Réponse 8

- **Vider les buffers :**

```
Ecoulé : 00 :00 :00.01
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.26
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.06
SQL>
```

- **Réexécution de la requête R2 :**

```
SQL> select DT.Année,
  2         sum(FO.MontantV) as MVAnnuel
  3  from DTemps DT, FOperation FO
  4  where FO.CodeTemps = DT.CodeTemps
  5  group by DT.Année
  6  order by DT.Année;

ANNÉE              MVANNUEL
--------------- ----------
2015            4226461684
2016            4210595166
2017            4219235397
2018            4230932776
```

- **Examination du temps d'exécution :**

```
Ecoulé : 00 :00 :00.19
```

- **Examination du plan d'exécution :**

```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 2666923579

-------------------------------------------------------------------------------
---------------------

| Id  | Operation                      | Name              | Rows  | Bytes | Co
st (%CPU)| Time     |

-------------------------------------------------------------------------------
---------------------

|   0 | SELECT STATEMENT               |                   |     4 |   140 |
 12  (34)| 00:00:01 |

|   1 |  SORT ORDER BY                 |                   |     4 |   140 |
 12  (34)| 00:00:01 |

|   2 |   HASH GROUP BY                |                   |     4 |   140 |
 12  (34)| 00:00:01 |

|*  3 |    HASH JOIN                   |                   |   136 |  4760 |
 10  (20)| 00:00:01 |

|   4 |     MAT_VIEW REWRITE ACCESS FULL| VMMONTANTVMENSUEL |    48 |  1056 |
  3   (0)| 00:00:01 |

|   5 |     VIEW                       |                   |   136 |  1768 |
  6  (17)| 00:00:01 |

|   6 |      HASH UNIQUE               |                   |   136 |  1768 |
  6  (17)| 00:00:01 |

|   7 |       TABLE ACCESS FULL        | DTEMPS            |  1460 | 18980 |
  5   (0)| 00:00:01 |

-------------------------------------------------------------------------------
---------------------
```

```
Predicate Information (identified by operation id):
---------------------------------------------------

   3 - access("from$_subquery$_005"."MOIS"="VMMONTANTVMENSUEL"."MOIS")

Note
-----
   - dynamic sampling used for this statement (level=2)

SQL>
```

Conclusion 2 : On remarque une baisse du temps d'exécution de la requête R2 (00:00:00.19) par rapport à celui examiné avant la création des métadonnées des dimensions (00:00:00.53). L'alimentation des métadonnées concernant les hiérarchies a montré le lien entre les attributs **Année** et **Mois**, ce qui a permis à l'optimiseur d'exploiter la vue matérialisée VMMontantVMensuel afin de réduire le temps d'exécution. En effet, comme l'année est composée de 12 mois,

l'optimiseur calcule la somme des montants versés mensuels en utilisant VMMontantVMensuel au lieu de calculer la somme de tous les montants versés existants.

Réponse 9

- Création de la vue matérialisée VMontantVVille :

```
SQL> CREATE MATERIALIZED VIEW VMMontantVVille
  2       BUILD IMMEDIATE REFRESH COMPLETE ON DEMAND
  3       enable query rewrite
  4       AS select DA.CodeVille, DA.NomVille,
  5                 sum(FO.MontantV) as MontantVille
  6            from FOperation FO, DAgence DA
  7           where FO.NumAgence = DA.NumAgence
  8           group by DA.CodeVille, DA.NomVille
  9           order by DA.CodeVille;

Vue matérialisée créée.

Ecoulé : 00 :00 :01.33
SQL>
```

Réponse 10

- Vider les buffers :

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.23
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.03
SQL>
```

- **Ecriture et exécution de la requête R3 qui donne les montants versés par Wilaya :**

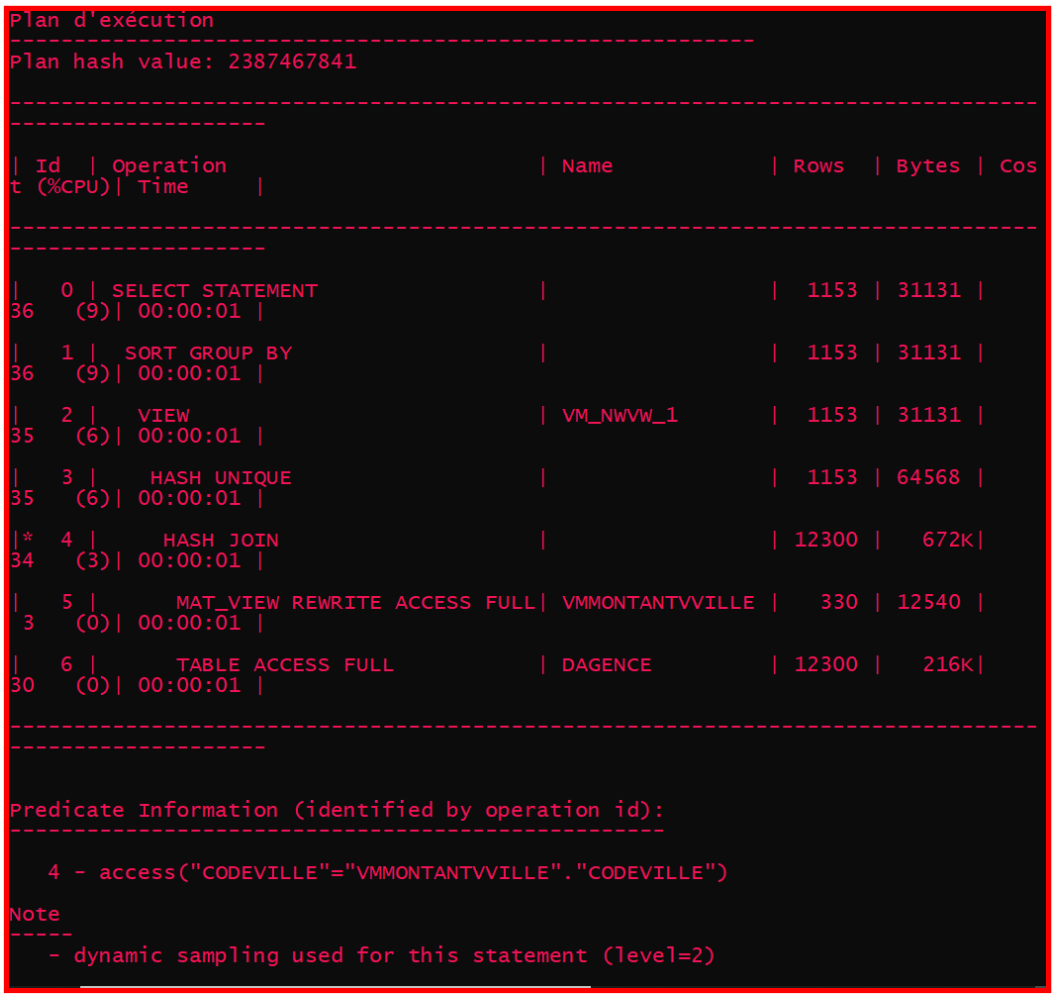
```
SQL> select DA.CodeWilaya, DA.NomWilaya,
  2             sum(FO.MontantV) as MontantWilaya
  3         from FOperation FO, DAgence DA
  4        where FO.NumAgence = DA.NumAgence
  5        group by DA.CodeWilaya, DA.NomWilaya
  6        order by DA.CodeWilaya;

CODEWILAYA NOMWILAYA                                           MONTANTWILAYA
---------- -------------------------------------------------- -------------
         1 JISYKJBS                                             289946702
         2 RTTWDMFH                                             263163008
         3 CYRSXXJE                                             229453489
         4 MZIKVSMC                                             201279652
         5 BZFBFROK                                             402881655
         6 NPHWCJFN                                             180015419
         7 LCDPNTGV                                             365100662
```

Réponse 11

- **Examination du temps d'exécution :**

```
48 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.61
```

- **Examination du plan d'exécution :**

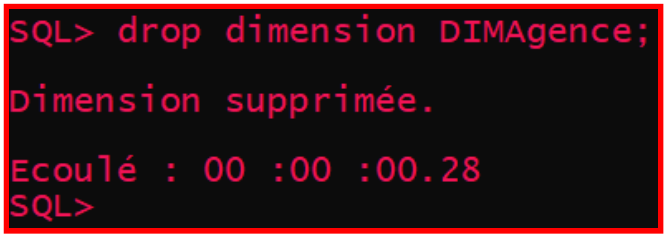


```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 2387467841

-------------------------------------------------------------------------------
--------------------

| Id  | Operation                      | Name            | Rows  | Bytes | Cos
t (%CPU)| Time     |

-------------------------------------------------------------------------------
--------------------

|   0 | SELECT STATEMENT               |                 |  1153 | 31131 |
36   (9)| 00:00:01 |

|   1 |  SORT GROUP BY                 |                 |  1153 | 31131 |
36   (9)| 00:00:01 |

|   2 |   VIEW                         | VM_NWVW_1       |  1153 | 31131 |
35   (6)| 00:00:01 |

|   3 |    HASH UNIQUE                 |                 |  1153 | 64568 |
35   (6)| 00:00:01 |

|*  4 |     HASH JOIN                  |                 | 12300 |   672K|
34   (3)| 00:00:01 |

|   5 |      MAT_VIEW REWRITE ACCESS FULL| VMMONTANTVVILLE |   330 | 12540 |
 3   (0)| 00:00:01 |

|   6 |      TABLE ACCESS FULL         | DAGENCE         | 12300 |   216K|
30   (0)| 00:00:01 |

-------------------------------------------------------------------------------
--------------------


Predicate Information (identified by operation id):
---------------------------------------------------

   4 - access("CODEVILLE"="VMMONTANTVVILLE"."CODEVILLE")

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Remarque 5 : On remarque que l'optimiseur a exploité la vue materialisée VMMontantVVille pendant l'exécution de la requête R3.

Réponse 12

- **Suppression des métadonnées de la dimension DAgence :**



```
SQL> drop dimension DIMAgence;

Dimension supprimée.

Ecoulé : 00 :00 :00.28
SQL>
```

- **Vider les buffers :**

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.21
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.03
SQL>
```

- **Réexécution de la requête R3 :**

```
SQL> select DA.CodeWilaya, DA.NomWilaya,
  2             sum(FO.MontantV) as MontantWilaya
  3          from FOperation FO, DAgence DA
  4         where FO.NumAgence = DA.NumAgence
  5         group by DA.CodeWilaya, DA.NomWilaya
  6         order by DA.CodeWilaya;

CODEWILAYA NOMWILAYA                                          MONTANTWILAYA
---------- -------------------------------------------------- -------------
         1 JISYKJBS                                               289946702
         2 RTTWDMFH                                               263163008
         3 CYRSXXJE                                               229453489
         4 MZIKVSMC                                               201279652
         5 BZFBFROK                                               402881655
         6 NPHWCJFN                                               180015419
         7 LCDPNTGV                                               365100662
         8 HBJGBNVT                                               253293603
```

- **Examination du temps d'exécution :**

```
48 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.87
```

- **Examination du plan d'exécution :**

```
Plan d'exécution
----------------------------------------------------------
Plan hash value: 1774511753

-------------------------------------------------------------------------------------
----
| Id  | Operation             | Name       | Rows  | Bytes | Cost (%CPU)| Time
   |
-------------------------------------------------------------------------------------
----
|   0 | SELECT STATEMENT      |            |  1630 | 60310 |   956   (5)| 00:00:
12 |
|   1 |  SORT GROUP BY        |            |  1630 | 60310 |   956   (5)| 00:00:
12 |
|*  2 |   HASH JOIN           |            | 12300 |   444K|   955   (4)| 00:00:
12 |
|   3 |    VIEW               | VW_GBC_5   | 12300 |   216K|   924   (5)| 00:00:
12 |
|   4 |     HASH GROUP BY     |            | 12300 |   120K|   924   (5)| 00:00:
12 |
|   5 |      TABLE ACCESS FULL| FOPERATION |   609K|  5953K|   897   (2)| 00:00:
11 |
|   6 |    TABLE ACCESS FULL  | DAGENCE    | 12300 |   228K|    30   (0)| 00:00:
01 |
-------------------------------------------------------------------------------------
----

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("ITEM_1"="DA"."NUMAGENCE")

SQL>
```

Remarque 6 : On remarque qu'après la suppression des métadonnées de la dimension DAgence, le temps d'exécution a augmenté et l'optimiseur n'a pas exploité la vue matérialisée VMMontantVVille pour exécuter la requete R3 car avec la suppression, les liens hiérarchiques entre les attributs **NumAgence**, **CodeBanque**, **CodeVille** et **CodeWilaya** ont été détruits.

Réponse 13

- **Conclusions :**

L'exploitation de la vue matérialisée a pour but d'optimiser le temps d'exécution des requêtes. Comme dans le cas de la vue matérialisée VMWilaya et la requête R1 vu que les résultats sont précalculés dans cette vue.

Les tables des dimensions contiennent des attributs définissants des niveaux hiérarchiques. L'implémentation des métadonnées de ces derniers sous forme de vues matérialisées permet d'optimiser le temps d'exécution des requêtes. Dans ce TP, l'optimiseur s'est appuyé sur les vues matérialisées VMMontantVMensuel et VMMontantVVille afin de réduire le coût d'exécution de la requête R2 et R3 respectivement en se basant sur la relation entre les attributs Année et Mois de la dimension DTemps et les attributs NumAgence, CodeBanque, CodeVille et CodeWilaya de la dimension DAgence respectivement.

Réponse 14

Le paramètre de session **query_rewrite_integrity** définit le niveau d'intégrité de la réécriture de la requête. En mode par défaut (Enforced) l'optimiseur utilise uniquement les données des vues matérialisées et les relations basées sur des contraintes de clé primaire, unique ou étrangère. En mode **trusted**, l'optimiseur approuve également les contraintes de clé déclarées mais non primaires ou uniques et les relations des données spécifiées à l'aide des dimensions. Ce mode offre de plus grandes capacités de réécriture de requête, mais crée également le risque de résultats incorrects si l'une des relations entre les attributs des dimensions déclarées est incorrecte[1].

[1] Documentation Oracle : (https://docs.oracle.com/database/121/DWHSG/qrbasic.htm#DWHSG8496)

Conclusion

Après la réalisation de ce TP, nous pouvons dire que l'exploitation des dimensions par le biais des vues matérialisées est très utile et permet un gain de temps assez important et ceci via les méthodes de réécriture des requêtes.