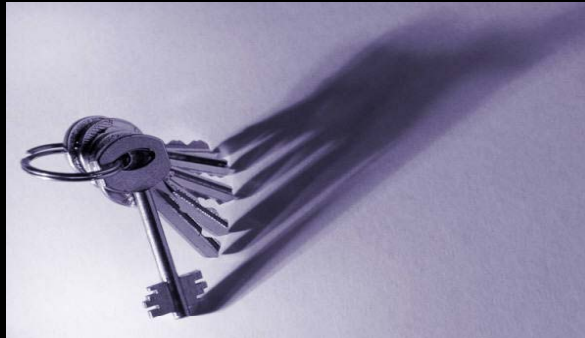


The Overview of Materialized View



Getting the most out of MetaLink

최창권

한국 오라클 제품지원실

ORACLE

안녕하십니까?

한국 오라클에서 주최하는 Technical iSeminar “Mview Overview” 에 참석해 주신 여러분께
감사 드립니다.

저는 이번 세미나를 진행하게 될 한국오라클 제품지원실에 근무하는 최창권 입니다.

본 seminar 는 Replication 환경에서 사용하는 Mview 와 DW 환경에서 사용하는 Mview 에
대해서 알아 보기로 하겠습니다.

Mview 를 사용함으로써 인해 Query speed 를 보다 빠르게 사용할 수 있게 하는 Query Rewrite
기능 및 I/O 를 분산하여 Base Table 에 대한 Load 를 줄일 수 있는 데 대해서 목적을
가지고 있습니다.

Agenda

- What is Mview ?
- Refresh method and option
- Mview for Replication
- Mview for DW
- Query Rewrite
- Troubleshooting.
- Reference

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 seminar 는

먼저 Mview 가 무엇인지 ?

그리고 Mview 의 Refresh method 및 Option 에 대해서 알아보도록 하겠습니다.

이후 Replication 에서 사용되는 Mview 및 DW 에서 사용되는 Mview 에 대해서 알아 보도록 하겠습니다.

그리고 Query Rewrite 기능에 대해서 알아보며 , Troubleshooting 에 대해서 알아보고자 합니다.

What is Mview ?

- VIEW

- 논리적인 Table
- Object 로 존재하지 않는다.
- DBA_SEGMENTS 에 존재하지 않는다.

- Mview

- 물리적인 Table 을 포함한 view
- View 의 결과를 물리적으로 Table 에 저장한다.
- DBA_SEGMENTS 에서 확인이 가능함.
- Table 과 Mview 는 동일한 이름으로 생성.
- ORACLE 8i 이전 : Snapshot 으로 제공됨
- 8i 에서 Snapshot 의 기능을 발전시켜 Mview 로 소개됨

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

먼저 Mview 는 무엇인가?

이전에 VIEW 에 대해서 먼저 확인해 볼 필요성이 있습니다.

VIEW 의 경우 물리적인 Object 가 아닌 단지 논리적인 Definition 입니다.

이에 반해 MVIEW 의 경우 물리적인 Table 을 포함하는 View 의 형태입니다.

이는 실제로 DBA_Objects 를 조회하게 되면 다음과 같은 형태로 Table 을 포함하여 생성이 되게 됩니다.

```
SQL> Select OWNER,Object_NAME,Object_TYPE,STATUS
2 FROM DBA_Objects
3 WHERE Object_NAME='DEPTV' ;
```

OWNER	Object_NAME	Object_TYPE	STATUS
SCOTT	DEPTV	Table	VALID
SCOTT	DEPTV	MATERIALIZED VIEW	VALID

위의 결과와 같이 Table 은 Mview 와 동일한 이름으로 생성이 됩니다.

8i 이전에서는 주로 Replication 환경에서 Snapshot 으로 사용이 되었습니다.

이후 8i 에서 Mview 로 통합되어 Optimizer 의 위에서 동작할 수 있도록 하는 Local site 의 Summary Table 을

Reference 할 수 있는 기능을 제공합니다.

The Need for Materialized Views

- Remote 의 Table 에 대한 Data 를 Access 하지 않고 Local Table 에서만 Query 를 수행하여 Network Overload 를 줄이기 위해서
- Large Database 의 Query Performance 를 증대하기 위해서
- Database 의 Table 에 대한 Query 를 수행하기 전 Join 및 통계 를 위한 Data 를 미리 수집하기 위해서
- Detail Table에서 Query 를 수행하지 않고 Mview 에 대해 Query 를 수행하기 위해서

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

Mview 를 사용하는 이유는 먼저 Replication 의 관점에서 보면 , Master Node 의 Data 를 복제하여 Network 을 통하여 Access 하여 Network 에 대한 Overload 를 줄여주기 위하여 Mview 를 생성합니다. 따라서 Remote Node 에서만 Query 를 수행하기 때문에 Master Table 에 대한 Network Load 를 줄일 수가 있습니다 .

그리고 DW 관점에서 보자면 Large Database 에 대한 Query Performance 를 증대할 수 있으며, Join 방식 형태의 Query 나 통계를 수집하기 위한 Data 를 Mview 로 미리 생성해 놓음으로써 Query 에 대한 Cost 를 줄일 수가 있습니다.

또한 Base Table에 대한 Query 를 수행 시 관련되어 있는 Mview 의 Rewrite 기능을 이용하여 Mview 의 통계정보를 직접 이용할 수가 있습니다.

Refresh Mode

- On commit
 - Commit 발생시 자동으로 Refresh.
 - Master Table 에 대하여 concurrent 한 변경작업이 일어나는 경우 부적합.
- On demand (default)
 - Package 를 이용하여 manual 하게 Refresh
 - DBMS_MVIEW
 - Dbms_Refresh : Refresh Group 을 생성했을 경우
 - Dbms_job : JOB 에 등록을 했을 경우

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

Refresh mode 로는

On Commit 방식 과 On Demand 방식이 있습니다.

이중 On Commit 방식은 Refresh Group 등에 대한 고려가 필요 없으며, Commit 이 발생함과 동시에 자동으로 Refresh 가 됩니다.

이는 Mview 가 Reference 하고 있는 Master Table 에 동시에 많은 Session 들에 의한 변경작업이 있을 경우 Recursive 하게 Mview 까지 반영을 해야 하므로 Performance Issue 를 발생할 수 있습니다.

그리고 On Demand 방식은 Create Mview 시의 Default Option 으로 DBMS_MVIEW 나 Refresh Group 을 생성하여 Dbms_Refresh 또는 Job 으로 등록을 하여 Dbms_job 등에 의해서 Manual 하게 Refresh 를 수행하는 작업이 필요합니다.

Refresh Options

- Complete
 - Refresh 시점에 Mview 에 대한 Table TRUNCATE
 - 전체 Data 를 다시 Query 해서 적용
- Fast
 - Base Table 에 대한 Mview log 를 이용하여 변경된 Data 를 적용
- Force (default)
 - 먼저 FAST 방식으로 시도하지만
 - FAST Refresh 방식이 실패할 경우 COMPLETE Refresh 로 시도
- Never
 - Refresh 를 전혀 수행하지 않음.

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

그럼 Mview 가 Refresh 되는 Option 에 대해서 알아 보도록 하겠습니다.

Refresh 가 되는 Option 으로는 다음과 같이 4가지가 소개가 됩니다.

먼저 Complete 의 경우는 이는 Mview 를 새롭게 생성하는 것과 다를 바가 없습니다.
이는 먼저 Mview Table 에 Truncate 를 하여 전체 Data 를 모두 삭제한 후 Master Table 의 Data 를 재 Query 하여 Refresh 하는 방법입니다.

그리고 가장 많이 사용되는 Option 으로 Fast 방식이 있습니다.
Fast 방식은 Master Table 의 Mview log 를 이용하여 Master Table 에 변경된 Data 만을 Mview 에 적용하는 방식입니다.

그리고 Force 방식과 Never 방식이 있습니다.
Force 방식은 Refresh 시에 먼저 REFRESH FAST 방식으로 시도를 하지만 REFRESH FAST 방식이 실패했을 경우 Complete 방식으로 Refresh 를 시도하게 됩니다.

Never 는 Refresh 를 전혀 수행하지 않는 것을 의미합니다.

Restrictions on Fast Refresh(1)

- SYSDATE , ROWNUM 을 포함하지 않아야 함.
- RAW , LONG RAW type 을 지원하지 않음.
- Select List 에 subQuery 는 지원되지 않음.
- Select 절에 RANK 함수 등 analytical function 은 허용되지 않음.
- MODEL 절을 포함하지 말 것.
- SubQuery 를 가지는 Having 절을 포함하지 말 것.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

REFRESH FAST 를 수행중의 제약사항은 다음과 같은 내용이 있습니다.

1. Select List 에서 SYSDATE , ROWNUM 을 포함하지 않아야 합니다.
2. Datatype 중 RAW , LONG RAW Type 을 지원하지 않습니다.
3. Select List 에 subQuery 는 지원이 되지 않습니다.
4. Select 절에 RANK 등의 분석 함수는 허용이 되지 않습니다.
5. MODEL 절에 대해서는 허용이 되지 않습니다.
6. SubQuery 를 가지는 HAVING 절을 포함해서는 안됩니다.

Restrictions on Fast Refresh (2)

- ANY , ALL , NOT EXISTS 등을 가지는 Nested Query 를 포함하지 말 것.
- CONNECT BY 절을 포함하지 말 것.
- 각기 다른 site 의 여러 Table 을 포함하지 말 것.
- On-commit Mview 는 Remote Table 을 참조해서는 안됨.
- Nested Mview 의 경우 Join 이나 집합연산을 가져야 함.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

7. ANY, ALL , NOT EXISTS 등을 가지는 Nested Query 를 포함하지 않아야 합니다.
8. CONNECT BY 절에 대해서는 허용이 되지 않습니다.
9. 각각 다른 Site 의 여러 Table 에 대해서는 지원이 되지 않습니다.
10. ON-commit Mview 의 경우 Remote Table 을 참조해서는 안됩니다.
11. Nested Mview 의 경우 Join 이나 집합연산 에 대한 Mview 이어야 합니다.

Mview for Replication

- 분산 Database 환경에서 Data 를 복제하기 위해서 사용
- 각각의 site 의 변경된 Data를 동기화 하기 위해서 사용
- Remote 의 Master Table 을 access 하지 않고 Local site 의 Mview 로 access 함으로 인해 network overload 해소
- Master site 로의 connection 이 끊어져 있는 상황에서 Local 에서 Data 접근 가능
- Remote Data Mart 에 유용

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

Replication 상에서 Mview 를 사용하는 이유는

1. 분산 Database 환경에서 Data 를 복제하여 Local Table 에 Data 를 저장하기 위함입니다.
이는 Read Only 형태로 복제가 가능하며 , Udatable 형태로 복제가 가능합니다.
Read Only의 경우는 Remote 에서는 전혀 변경을 할 수 없으며, 단지 Master Site 에서 변경된 Data 를 Sync 하여 참조하는 형태입니다.
다음으로 Udatable Mview 의 경우 Remote 에서도 Update 가 가능합니다.
이는 Read Only 의 Data 복제하는 방법과 같은 맥락으로 복제가 되지만 Remote Site 에서는 변경된 Data 를 따로 저장하는 USLOG\$_MVNAME 이 생성되는 점과 변경된 Data 를 Push 하는 Mechanism 이 있다는 점의 차이가 있습니다.
2. 각각의 변경된 Data 에 동기화가 가능합니다.
3. Master Site 에 대한 Data 를 Remote 의 Table 로 저장하기 때문에 Network 을 이용하여 Access 해야하는 부담을 줄이며 , Network 에 대한 Overload 를 해소할 수 있습니다.
또한 Network 이 끊기거나 상대 Node 의 Instance Fail 이 일어나더라도 Mview Site 의 Local Table 에 Access 하여 Data 에 대한 Query 가 가능합니다.
4. Remote Data mart 등에 유용하게 사용이 됩니다.

Type of Mview for Replication

- Read only Mview
 - Mview 에 대한 변경작업이 불가능
 - Master site 의 Mview log 를 이용하여 변경된 Data 반영
- Updatable Mview
 - Mview site 에서 변경작업이 가능
 - USLOG\$ Table 에 변경 Data를 저장
 - Master site 에 변경된 Data 를 반영후 Mview log 의 정보를 Refresh
 - Uslog\$ Table 의 정보 purge.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

Replication 을 위한 Mview 는 다음과 같이 Read Only Mview , Updatable Mview 가 있습니다.

먼저 READ ONLY Mview 의 경우 Master Table 에서만 변경작업이 가능하며 Mview Site 에서는 변경작업 자체가 불가능합니다.

그리고 Updatable Mview 의 경우, Master Site 뿐만 아니라 Mview Site 에서 모두 변경작업이 가능합니다.

Mview Site 에서 변경작업이 일어날 때 Data 가 반영되는 순서를 보자면,

먼저 Mview Site 에서 변경작업이 일어나게 되면 변경된 Data 는 USLOG\$ Table 에 저장됩니다.

이후 Refresh 를 수행했을 경우 혹은 Deferred Transaction 에 대해서 Master Site 에 Uslog\$ Table 의 Data 를 전송하게 되며, Master Site 에 전송된 Data 는 Master Table 및 Mview log Table 에 실제로 반영이 됩니다.

이후 Mview Site 에서는 해당 Data 에 대한 mlog\$ Table 의 정보를 Refresh 하여 Uslog\$ Table 과 비교하여 해당 Data가 정상적으로 반영이 되어 있는 것을 확인 후 Uslog\$ Table 정보를 Purge 하는 형태의 구조를 가지고 있습니다.

Mview for DW

- SUM 및 AVG 등의 집합 연산에 대한 통계 Data 를 Mview 에 저장하여 미리 연산을 하는 역할
- Join 에 대한 Query 를 미리 수행
- 중요한 Query 에 대해서 Join 및 집합 연산에 대한 cost 를 줄이는 역할

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 DW 환경에서의 Mview 를 소개하고 있습니다.

DW 환경에서는 SUM 등의 집합연산을 통한 SQL 문장을 많이 사용하게 됩니다.
또한 여러 Table 을 Join 하여 통계정보를 수집하는 SQL 문장들을 많이 발행하게 됩니다.

하지만 해당 Query 들은 전체의 Data 에 대한 정보를 기반으로 통계가 산출되기 때문에 많은 COST 가 필요로 하며 , 해당 Query 들로 인하여 전반적인 Database 의 Performance 에 영향을 줄 수 있습니다.

이에 Mview 를 사용하여 SUM 및 AVG 등의 집합연산을 하는 통계 Data 를 미리 Table 에 저장하여 미리 연산을 할 수 있습니다.

또한 Join 등이 실행되는 SQL 문장에 대해서 Join Operation 을 수행하여 Mview 내에 저장 가능합니다.

그리고 Join 연산 및 집합 연산에 대한 Mview 들을 Nested Mview 로 생성이 가능합니다.

Types of Mview for DW

- Materialized Views with Aggregates
- Materialized Views Containing Only Joins
- Nested Materialized Views

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 DW 환경에서 사용되는 Mview 의 TYPE 에 대해서 소개하고 있습니다.

DW 환경에서의 Mview type 은 다음과 같이 3가지 형태로 존재합니다.

1. 집합함수를 가지는 Mview 의 형태
2. 여러 Table 에 대한 Join 의 형태를 지닌 Mview 의 형태.
3. 그리고 이를 조합한 형태의 Nested Mview 의 형태

가 있습니다.

Restrictions on Fast Refresh on Materialized Views with Aggregates

- All restrictions from “General Restrictions on Fast Refresh”
- 모든 base Table 에 대해서 Mview log 가 생성되어 있어야 한다.
 - Mview 에서 reference 하는 모든 column 이 포함되어야 한다.
 - 반드시 rowid 와 INCLUDING NEW VALUES 절이 포함되어야 한다.
- Fast Refresh 의 경우 SUM, COUNT, AVG, STDDEV, VARIANCE, MIN, MAX function 만을 지원한다.
- COUNT(*) 가 반드시 포함되어야 한다.
- AVG(expr) 를 사용시 반드시 COUNT(expr) 이 존재해야 한다.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

먼저 집합 연산을 가지는 Mview 에 대해서 알아 보도록 하겠습니다.

본 SLIDE 는 집합연산을 가지는 Mview 를 사용하고자 하는 경우 REFRESH FAST 시의 제약사항을 소개하고 있습니다.

1. General Restriction of Fast Refresh 의 제약사항이 적용되며,
2. 모든 Base Table 에 대해서는 Mview log 가 생성되어 있어야 합니다.
이때 Mview log 는 반드시 Rowid 와 INCLUDING NEW VALUES 절이 포함되어야 하며,
Mview 에서 Reference 하는 모든 Column 이 포함되어 있어야 합니다.
3. FAST Refresh 의 경우 SUM, COUNT , AVG , STDDEV (편차) , VARIANCE (분산) , MIN , MAX function 만을 지원합니다.
4. COUNT(*) 이 반드시 포함되어 있어야 합니다.
5. AVG(expr) function 을 사용하는 경우에는 COUNT(expr) 이 같이 존재해야 합니다.

Restrictions on Fast Refresh on Materialized Views with Aggregates

- VARIANCE(expr) or STDDEV(expr) function 이 사용되는 경우 COUNT(expr) 과 SUM(expr) 이 반드시 존재해야 한다.
- Select List 에는 GROUP BY column 이 존재해야 함.
- 각각의 집합연산을 하는 column 에 대해서 COUNT function이 포함되어야 한다.
 - AVG(amount_sold) , COUNT(amount_sold)
- **Only-insert** 에 대해서만 지원되는 경우
 - MIN or MAX 연산을 가지는 Mview
 - SUM(expr) 는 있으나 COUNT(expr) 는 없는 경우
 - COUNT(*) 이 없는 경우

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

6. VARIANCE(expr) or STDDEV(expr) function 이 사용되는 경우 COUNT(expr) 과 SUM(expr) 이 반드시 존재해야 합니다.
7. GROUP BY 가 사용되는 경우 관련 Column 은 Select List 에 존재해야 합니다.
8. AVG(expr) 등의 집합연산을 수행하는 경우 COUNT(expr) function 이 포함되어야 합니다.
9. 다음의 경우에도 Fast Refresh 를 지원을 하지만 단지 Insert Operation 에 대해서만 지원이 됩니다.
MIN or MAX Function 을 가지는 Materialized Views,
SUM(expr) 은 있으나 COUNT(expr) 이 없는 경우,
COUNT(*) 이 없는 경우

Creation of Mview log for Materialized Views with Aggregates

```
CREATE MATERIALIZED VIEW LOG ON products
WITH SEQUENCE, rowid
(prod_id, prod_name, prod_desc, prod_subcategory, prod_subcat_desc, prod_
category, prod_cat_desc, prod_weight_class, prod_unit_of_measure, prod_pack_
size, supplier_id, prod_status, prod_List_price, prod_min_price)
INCLUDING NEW VALUES;
```

```
CREATE MATERIALIZED VIEW LOG ON sales
WITH SEQUENCE, rowid
(prod_id, cust_id, time_id, channel_id, promo_id, quantity_sold, amount_sold)
INCLUDING NEW VALUES;
```

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 집합연산을 가지는 Mview 를 생성시의 Mview log 를 생성하는 실제 예를 소개하고 있습니다.

본 예에서 가장 먼저 확인할 수 있는 사항은

1. Mview Log 는 반드시 Rowid 와 INCLUDING NEW VALUES 절을 가져야 합니다.
2. Mview Log 는 Mview 가 Reference 하는 Column을 모두 포함해야 합니다.

따라서 SLIDE 에서 소개되는 Mview Log 는 집합연산을 가지는 Mview 를 생성하기 위한 조건에 부합한다고 할 수 있습니다.

Materialized Views with Aggregates

- Example

```
CREATE MATERIALIZED VIEW product_sales_mv
BUILD IMMEDIATE
REFRESH FAST
ENABLE Query Rewrite
AS Select p.prod_name,
SUM(amount_sold) AS dollar_sales,
COUNT(*) AS cnt, COUNT(amount_sold) AS cnt_amt
FROM sales s, products p
WHERE s.prod_id = p.prod_id
GROUP BY prod_name;
```

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

다음으로 이전 SLIDE 에서 생성된 Mview Log 를 이용하여 집합연산을 가지는 Mview 를 생성하는 예를 소개하고 있습니다.

본 SLIDE 에서 CHECK 해야 하는 점은 REFRESH FAST 방식으로 Mview 가 생성된다는 점입니다.

따라서 집합 연산을 가지는 Mview 를 FAST Refresh 하고자 하는 경우의 제약사항에 부합해야 합니다.

1. GROUP BY 절에서 prod_name Column 이 사용이 되었기 때문에 반드시 Select List 에 존재해야 합니다.
2. Amount_sold 에 대한 SUM function 을 사용하였기 때문에 COUNT(amount_sold) 가 존재해야 합니다.
3. COUNT(*) 절을 가지고 있습니다.

따라서 본 SLIDE 의 Mview 문장은 모든 조건이 부합되기 때문에 FAST Refresh 방식의 Mview를 생성하는 데 부합한 조건을 가지고 있습니다.

Materialized Views with Aggregates

- Example

```
CREATE MATERIALIZED VIEW product_sales_mv
BUILD DEFERRED
REFRESH COMPLETE ON DEMAND
ENABLE Query Rewrite
AS
Select
p.prod_name,
SUM(amount_sold) AS dollar_sales
FROM sales s, products p
WHERE s.prod_id = p.prod_id
GROUP BY p.prod_name;
```

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 COMPLETE REFRESH 방식의 Mview 에 대한 예를 들고 있습니다.

SLIDE 에서 소개되는 Mview 는 COMPLETE Refresh 방식으로 생성이 되었지만 REFRESH FAST 방식으로도 생성이 가능합니다.

하지만

제약조건의 내용 중

COUNT(*) 가 Select List 가 없으며 ,

SUM(amount_sold) function 을 사용하고 있지만 COUNT(amount_sold) function 이 존재하지 않습니다.

따라서 SLIDE 에서 소개되는 Mview 를 FAST Refresh 방식으로 생성했을 경우 해당 Base Table 에는 오직 INSERT 만이 존재해야 하는 Mview 입니다.

따라서 전체적인 DML 에 대하여 Refresh 를 해야 하는 상황에서 COMPLETE Refresh 방식의 Mview 를 생성한 예를 보여주고 있습니다.

Materialized Views with Aggregates

- Example

```
CREATE MATERIALIZED VIEW SUM_sales
REFRESH FAST ON COMMIT
AS
Select s.prod_id, s.time_id,
COUNT(*) AS COUNT_grp,
SUM(s.amount_sold) AS SUM_dollar_sales,
COUNT(s.amount_sold) AS COUNT_dollar_sales,
SUM(s.quantity_sold) AS SUM_quantity_sales,
COUNT(s.quantity_sold) AS COUNT_quantity_sales
FROM sales s
GROUP BY s.prod_id, s.time_id;
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 이전에 언급되었던 FAST Refresh 방식과 동일한 구조의 Mview 를 소개하고 있습니다.

단지 Refresh MODE 로 ON DEMAND 만 지원되는 것이 아니며 , ON COMMIT MODE 로 생성이 가능한 예를 보여 주고 있습니다.

REFRESH FAST 의 제약조건을 다시 확인하자면

1. GROUP BY절이 가지는 Column s.prod_id , s.time_id 를 모두 Select List 에 포함하고 있으며,
2. COUNT(*) 이 Select List 에 존재하며 ,
3. SUM(s.amount_sold) 와 함께 COUNT(a.amount_sold) 가 존재하며 ,
4. SUM(s.quantity_sold) 와 함께 COUNT(a.quantity_sold) 가 존재합니다.

따라서 집합연산을 가지는 Mview 에 대한 FAST Refresh 방식에 모두 부합이 됩니다.

Restrictions on Fast Refresh on Materialized Views with Joins Only

- All restrictions from “General Restrictions on Fast Refresh”
- GROUP BY 절을 가질 수 없다.
- Select List 에 모든 Table 의 Rowid 가 반드시 기술이 되어야 한다.
- Mview log 는 반드시 Rowid 를 이용하여 생성이 되어야 한다.
- 만약 outer Join 이 있다면 inner Table에 Unique Constraint 가 존재해야 하며 Where 절을 가질 수 없다.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

다음으로 Join 만을 가지는 MVIEW 에 대해서 소개를 하도록 하겠습니다.

본 SLIDE 는 Join 만을 가지는 Mview 를 FAST Refresh 방식으로 생성하고자 하는 경우의 제약사항에 대해서 소개를 하고 있습니다.

1. FAST Refresh 의 일반적인 사항에 부합해야 하며,
2. GROUP BY 절을 가질 수 없습니다.
3. Select List 에는 Join 되는 모든 Table의 Rowid 가 반드시 존재해야 하며,
4. Mview Log 는 반드시 Rowid 를 이용하여 생성이 되어야 합니다.
5. 만약 Outer Join 이 있다면 Inner Table에 Unique Constraint 가 존재해야 하며 Where 절을 가질 수 없습니다.

Materialized Views Containing Only Joins

- Example

```
CREATE MATERIALIZED VIEW LOG ON sales WITH ROWID;  
CREATE MATERIALIZED VIEW LOG ON times WITH ROWID;  
CREATE MATERIALIZED VIEW LOG ON customers WITH ROWID;
```

```
CREATE MATERIALIZED VIEW detail_sales_mv  
REFRESH FAST  
AS  
Select  
s.rowid "sales_rid", t.rowid "times_rid", c.rowid "customers_rid",  
c.cust_id, c.cust_last_name, s.amount_sold,  
s.quantity_sold, s.time_id  
FROM sales s, times t, customers c  
WHERE s.cust_id = c.cust_id(+) AND  
s.time_id = t.time_id(+);
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

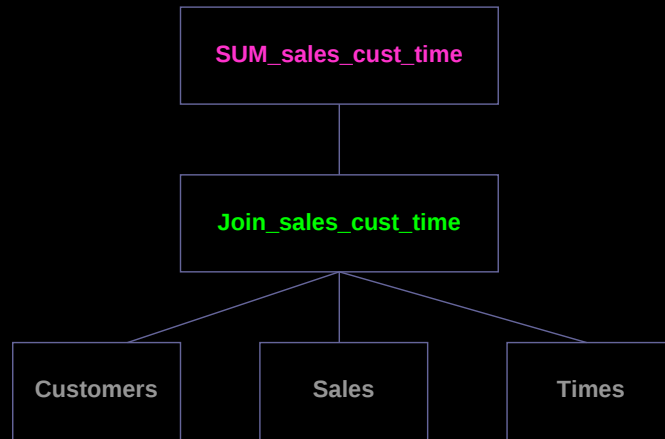
본 SLIDE 는 FAST Refresh 방식으로 Join 만을 가지는 Mview 에 대한 예를 소개하고 있습니다.

먼저 Mview Log는 Rowid 를 이용하여 생성이 되었으며,

1. Mview 는 관련된 Table 에 대한 Rowid 를 가지고 있습니다.
2. GROUP BY 절이 포함되지 않았습니다.

따라서 Join 절을 가지는 Mview 의 FAST Refresh 방식의 제약사항에 부합됩니다.

Nested Materialized Views



ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 여러 Table 을 이용하여 Join 만을 가지는 Mview 를 생성하고 이 Mview 를 Reference 하여 Mview Log 를 생성하여 집합 연산을 하는 Mview 를 생성하는 Nested Mview 의 구조를 보여 주고 있습니다.

이는 Join 이 미리 수행되어 있는 Mview 에 대하여 집합연산을 가지는 Mview 를 생성함으로 인해 많은 COST 를 요구하는 Join 및 집합연산을 Mview 로 생성하여 Storage 에 저장함으로 인해 Performance 향상을 기대할 수 있는 모델입니다.

Nested Materialized Views (1)

```
/* create the materialized view logs */
CREATE MATERIALIZED VIEW LOG ON sales WITH ROWID;
CREATE MATERIALIZED VIEW LOG ON customers WITH ROWID;
CREATE MATERIALIZED VIEW LOG ON times WITH ROWID;

/*create materialized view Join_sales_cust_time as fast Refreshable at
COMMIT time */
CREATE MATERIALIZED VIEW Join_sales_cust_time
REFRESH FAST ON COMMIT AS
Select c.cust_id, c.cust_last_name, s.amount_sold, t.time_id,
t.day_number_in_week, s.rowid srid, t.rowid trid, c.rowid crid
FROM sales s, customers c, times t
WHERE s.time_id = t.time_id AND
s.cust_id = c.cust_id;
```

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

먼저 Join 만을 가지는 Mview 를 다음과 같이 생성합니다.
이 구조는 이전에 소개되었던 것과 같은 구조를 가지고 있습니다.

Nested Materialized Views (2)

```
/* create materialized view log on Join_sales_cust_time */
CREATE MATERIALIZED VIEW LOG ON Join_sales_cust_time
WITH ROWID (cust_name, day_number_in_week, amount_sold)
INCLUDING NEW VALUES;

/* create the single-Table aggregate materialized view
   SUM_sales_cust_time on
Join_sales_cust_time as fast Refreshable at COMMIT time */
CREATE MATERIALIZED VIEW SUM_sales_cust_time
REFRESH FAST ON COMMIT
AS
Select COUNT(*) cnt_all, SUM(amount_sold) SUM_sales,
COUNT(amount_sold)
cnt_sales, cust_last_name, day_number_in_week
FROM Join_sales_cust_time
GROUP BY cust_last_name, day_number_in_week;
```

ORACLE

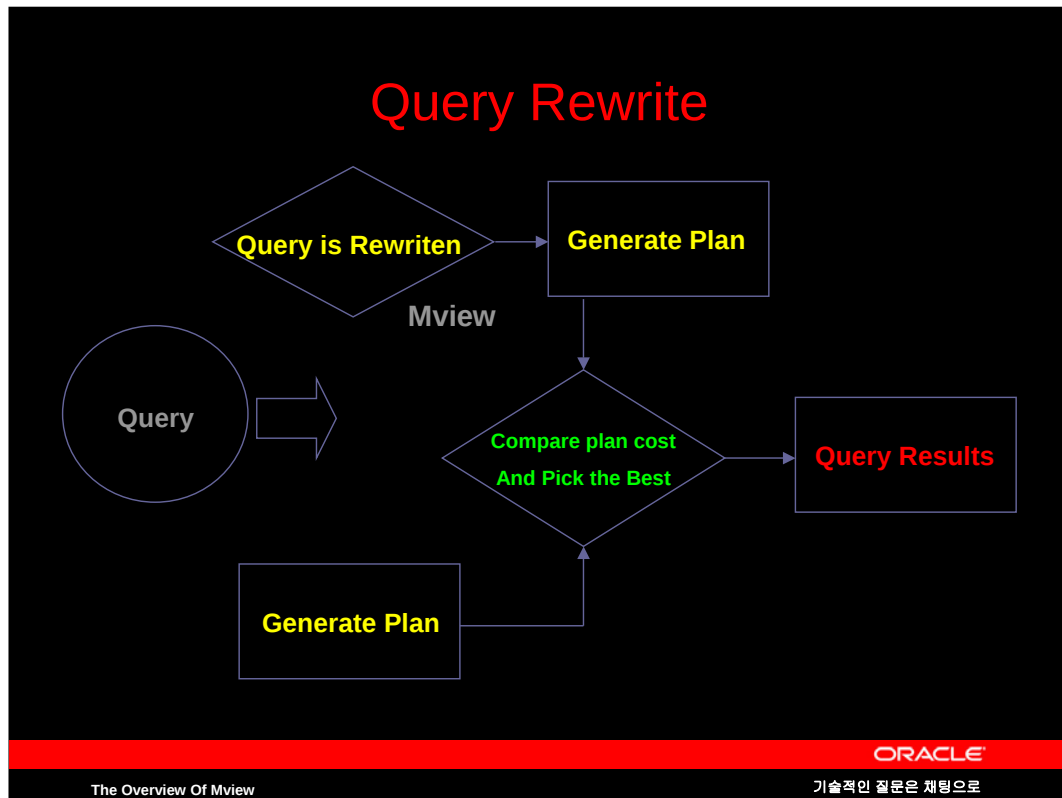
The Overview Of Mview

기술적인 질문은 채팅으로

이후 Join 만을 가지는 Mview 에 다시 Mview Log 를 생성하게 됩니다.

Mview Log 를 생성하는 구절에서는 Rowid 와 INCLUDING NEW VALUES 를 포함하고 있으며 Nested Mview 로 생성될 Reference Column 을 모두 포함하고 있습니다.

이후 Mview 를 생성하는 구조로 되어 있습니다.



본 SLIDE 는 Query Rewrite 의 기능을 그림으로 설명하고 있습니다.

사용자가 SQL 문장을 실행하게 되면 이는 해당 SQL 의 Plan 이 생성되게 되며, 또한 Mview 가 있는 경우 해당 Mview 에 대한 Query Rewrite 를 실행하게 됩니다.

이후 사용자가 실행한 SQL 문장에 대한 Plan 과 Rewrite 된 SQL 문장의 Plan 에 대한 Cost 를 비교하여 Optimizer 는 보다 Cost 가 작은 Plan 을 선택하여 Query 결과를 Return 하게 됩니다.

Parameters for Query Rewrite

- **OPTIMIZER_MODE**
 - ALL_ROWS (default), FIRST_ROWS, or CHOOSE
- **Query_Rewrite_ENABLED**
 - TRUE (default),
- **Query_Rewrite_INTEGRITY**
 - STALE_TOLERATED, TRUSTED, or ENFORCED(default)

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 Query Rewrite 를 하는 경우 Setting 되는 Initialization Parameter 를 소개하고 있습니다.

Setting 되어야 할 Parameter 는 다음과 같습니다.

OPTIMIZER_MODE : All_rows 나 First_rows 로 지정을 하거나 CHOOSE 로 지정이 되어야 하며 통계치 정보를 가지고 있어야 합니다.

Query_Rewrite_ENABLED : TRUE

Query_Rewrite_INTEGRITY

TRUSTED : 모든 Mview 의 Data 가 Base Table 과 일치한다고 판단함.

ENFORCED (default) : Base Table 과의 Consistent 를 Check.

STALE_TOLERATED : Base Table 과 Consistent 하지 않는 View 에 대해서도 Query Rewrite 를 허용.

Privilege for Query Rewrite

- GRANT Query Rewrite
 - 자기 자신의 Mview 에 대해서 Query Rewrite 권한을 제공
- GRANT GLOBAL Query Rewrite
 - Mview 에서 reference 하는 Object 가 자신의 Object 가 아니더라도 Query Rewrite 권한을 제공

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 Query Rewrite 를 수행하고자 할 경우 필요한 권한에 대해서 소개를 하고 있습니다.

권한으로는 Query Rewrite 와 GLOBAL Query Rewrite 가 있습니다.

Query Rewrite 권한은 자기 자신의 Mview 및 Base Table 에 대한 Query Rewrite 할 수 있는 권한이며 , GLOBAL Query Rewrite 권한은 자기 자신의 Mview 및 Base Table 뿐만 아니라 Reference 하는 Object 가 자신의 Object 가 아니더라도 Query Rewrite를 할 수 있는 권한입니다.

일반적으로 Query Rewrite 권한을 부여시 같이 부여하는 경우가 많습니다.

Example of Query Rewrite

```
TEST: SQL> Select s.prod_id, s.time_id,  
2 COUNT(*) AS COUNT_grp,  
3 SUM(s.amount_sold) AS SUM_dollar_sales,  
4 COUNT(s.amount_sold) AS COUNT_dollar_sales,  
5 SUM(s.quantity_sold) AS SUM_quantity_sales,  
6 COUNT(s.quantity_sold) AS COUNT_quantity_sales  
7 FROM sales s  
8 GROUP BY s.prod_id, s.time_id ;
```

Execution Plan

```
-----  
0  Select STATEMENT Optimizer=CHOOSE (Cost=760 Card=319489 Byte  
   s=27795543)  
  
1  0 Table ACCESS* (FULL) OF 'SUM_SALES' (Cost=760 Card=319489  
   :Q134000  
   Bytes=27795543)
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 Query Rewrite 의 예제를 보고 있습니다.

Select 문장은 Mview 를 생성시의 문장으로써 같은 문장을 수행하였을 경우 해당 Table 에 대해서 FULL Scan 을 하여 많은 비용을 들이는 형태로 Optimizer 가 동작하지 않고 Mview 의 Table 을 FULL scan 하는 형태로 Plan 을 선택하여 보다 적은 비용으로 같은 결과를 얻어 낼 수 있습니다.

Example of Query Rewrite

```
SQL> Select s.prod_id, s.time_id,  
2 COUNT(*) AS COUNT_grp,  
3 SUM(s.amount_sold) AS SUM_dollar_sales,  
4 COUNT(s.amount_sold) AS COUNT_dollar_sales,  
5 SUM(s.quantity_sold) AS SUM_quantity_sales,  
6 COUNT(s.quantity_sold) AS COUNT_quantity_sales  
7 FROM sales s  
8 GROUP BY s.prod_id, s.time_id  
9 having SUM(s.amount_sold) > 1000 ;
```

Execution Plan

```
-----  
0  Select STATEMENT Optimizer=CHOOSE (Cost=760 Card=15974 Bytes  
   =1389738)  
  
1  0 Table ACCESS* (FULL) OF 'SUM_SALES' (Cost=760 Card=15974 B :Q132000  
   ytes=1389738)
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 예제는 Mview 의 Sql 문장에서 Having 절을 추가하여 조건을 부여한 결과입니다.

이 역시 Optimizer 가 Base Table 을 Access 하지 않고 Mview 를 Access 하여 해당 결과를 Return 하는 것을 볼 수 있습니다.

Example of Query Rewrite

```
SQL> Select s.prod_id,  
2   SUM(s.amount_sold) AS SUM_dollar_sales  
3   FROM sales s  
4   GROUP BY s.prod_id  
5   having SUM(s.amount_sold) > 1000 ;
```

Execution Plan

```
-----  
0   Select STATEMENT Optimizer=CHOOSE (Cost=1179 Card=319489 Bytes=8306714)  
  
1  0   FILTER*                                     :Q146001  
2  1   SORT* (GROUP BY) (Cost=1179 Card=319489 Bytes=8306714) :Q146001  
3  2   SORT* (GROUP BY) (Cost=1179 Card=319489 Bytes=8306714) :Q146000  
4  3   Table ACCESS* (FULL) OF 'SUM_SALES' (Cost=760 Card=3 :Q146000  
    19489 Bytes=8306714)
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 결과는

Mview 내의 모든 Column 을 조회하지 않고 특정 Field 만을 조회하는 결과로서 역시 같은 형태로 Optimizer 가 Mview 의 Query Rewrite 기능을 사용하는 결과를 보이고 있습니다.

HINT For Query Rewrite

```
SQL> set autotrace trace exp
SQL> Select /*+ NOREWRITE*/ s.prod_id, s.time_id,
2 COUNT(*) AS COUNT_grp,
3 SUM(s.amount_sold) AS SUM_dollar_sales,
4 COUNT(s.amount_sold) AS COUNT_dollar_sales,
5 SUM(s.quantity_sold) AS SUM_quantity_sales,
6 COUNT(s.quantity_sold) AS COUNT_quantity_sales
7 FROM sales s
8 GROUP BY s.prod_id, s.time_id ;
```

Execution Plan

```
-----
0  Select STATEMENT Optimizer=CHOOSE (Cost=29289 Card=1016271 Bytes=20325420)

1  0  PARTITION RANGE (ALL)
2    1  SORT (GROUP BY) (Cost=29289 Card=1016271 Bytes=20325420)
3      2  Table ACCESS (FULL) OF 'SALES' (Cost=1973 Card=1016271 Bytes=20325420)
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 내용은 Query Rewrite 에 대한 HINT 를 사용하는 예 를 보여 주고 있습니다.

Query Rewrite HINT 로는

Rewrite
NOREWRITE

가 있습니다.

Rewrite Hint 는 지정한 Mview 에 대해서 Query Rewrite 기능을 사용하는 HINT 이며 , 예제로 보여주고 있는 내용은 해당 SQL 문장이 Query Rewrite 기능을 이용하여 PLAN 을 풀어가지만, NOWRITE HINT 를 이용하여 Mview 를 Access 하지 않고 Base Table 을 Access 하는 형태의 PLAN을 선택하는 결과를 보여주고 있습니다.

Restrictions of Query Rewrite

- 만약 Local Table 과 Remote Table 이 있다면, Local Table 에 대해서만 Rewrite 기능이 사용됨.
- Non-sys user 의 Detail Table 과 Mview 에 대해서만 사용됨.
- 만약 Mview 에 GROUP BY 절이 사용되었다면 관련된 column 및 표현식은 반드시 Select List 에 존재해야 함.
- AVG(AVG(x)) 및 AVG(x) + AVG(y) 등의 연산은 허용이 되지 않음.
- CONNECT BY 절은 허용되지 않음.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 Slide 에서는 Query Rewrite 의 기능의 제약사항에 대해서 보여주고 있습니다.

1. Local Table 과 Remote Table 이 관련되어 있다면 , Remote Table 에 대해서는 Query Rewrite 기능이 적용되지 않으며, Local Table 에 대해서만 Rewrite 기능을 적용하게 됩니다.
2. SYS User 의 Mview 나 Detail Table 에 대해서는 Query Rewrite 기능이 적용되지 않습니다.
3. 만약 Mview 에 GROUP By 절이 사용되었다면 관련되어 있는 Column 이나 , 표현식은 반드시 수행하는 SQL 문장에 포함이 되어 있어야 합니다.
4. AVG (AVG (x)) 및 AVG(x) + AVG (y) 등의 연산에는 Query Rewrite 기능이 적용되지 않습니다.
5. CONNECT BY 절은 허용되지 않습니다.

DBMS_MVIEW.EXPLAIN_MVIEW

- FAST Refresh 및 Rewrite 가 가능한지 확인.
- Mv_capabilities_Table 에 기록한다.
- \$OH/rdbms/admin/utlxmv.sql : MV_CAPABILITIES_Table 생성
- DBMS_MVIEW.EXPALIN_MVIEW ('MV_NAME')

```
Select mvname, capability_name , possible  
FROM mv_capabilities_Table ;
```

MVNAME	CAPABILITY_NAME	POSSIBLE
<< truncated ... >>		
SUM_SALES	Refresh_FAST	Y
SUM_SALES	Rewrite	Y
SUM_SALES	Refresh_FAST_AFTER_INSERT	Y
SUM_SALES	Rewrite_FULL_TEXT_MATCH	Y
SUM_SALES	Rewrite_PARTIAL_TEXT_MATCH	Y
SUM_SALES	Rewrite_GENERAL	Y

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

다음으로 Mview 에 대해서 Fast Refresh 가 가능한 지 혹은 Query Rewrite 기능이 사용 가능한지를 알아 볼 수 있는 DBMS_MVIEW.EXPLAIN_MVIEW 에 대해서 알아보도록 하겠습니다.

EXPLAIN_MVIEW 를 수행하게 되면 해당 정보는 MV_CAPABILITIES_Table 에 기록이 됩니다.
따라서 MV_CAPABILITIES_Table 을 생성할 필요가 있습니다.
MV_CAPABILITIES_Table 을 생성하기 위한 Script 는 utlxmv.sql 을 수행해 주어야 합니다.

Slide 에 보여주는 결과는 Query Rewrite 뿐만 아니라 Fast Refresh 까지 가능하다는 것을 보여 주고 있습니다.

DBMS_MVIEW.EXPLAIN_MVIEW

```
CREATE MATERIALIZED VIEW sal_dept
BUILD IMMEDIATE REFRESH FAST ON DEMAND
ENABLE Query Rewrite
AS
Select a.deptno , SUM(a.sal)
FROM emp a, dept b
WHERE a.deptno = b.deptno
GROUP BY a.deptno;
```

SQL> Select capability_name,possible from mv_capabilities_Table ;

CAPABILITY_NAME	POSSIBLE
Refresh_FAST_AFTER_INSERT	Y
Refresh_FAST_AFTER_ONETAB_DML	N
Refresh_FAST_AFTER_ONETAB_DML	N
Refresh_FAST_AFTER_ANY_DML	N

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

예를 들어 다음과 같은 Mview 가 있다고 가정을 합니다.

```
CREATE MATERIALIZED VIEW sal_dept
BUILD IMMEDIATE REFRESH FAST ON DEMAND
ENABLE Query Rewrite
AS
Select a.deptno , SUM(a.sal)
FROM emp a, dept b
WHERE a.deptno = b.deptno
GROUP BY a.deptno;
```

이 경우는 FAST Refresh 의 제약 조건 중 Select List 에 COUNT(*) 절이 포함되지 않으며, COUNT(a.sal) 절이 포함되어 있지 않기 때문에 이는 Only Insert 에만 Fast Refresh 가 되는 Mview 형태임을 알 수 있으며 , Query Rewrite 기능을 사용할 수 있음을 알 수 있습니다.

DBMS_MVIEW.EXPLAIN_REWRITE

- 관련 Query 가 Rewrite 기능을 사용가능한지 확인
- Rewrite_Table 에 기록한다.
- \$OH/rdbms/admin/utlxrw.sql : Rewrite_Table 생성

```
SQL> Select message from Rewrite_Table order by sequence;
```

```
MESSAGE
```

```
-----  
QSM-01009: materialized view, SUM_SALES, matched Query text
```

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서 보여주는 DBMS_MVIEW.EXPLAIN_REWRITE package 는 관련 Query 가 특정 Mview 에 대하여 Query Rewrite 기능을 사용 가능한지를 알아 보는 Package 입니다.

이는 Rewrite_Table 에 정보를 기록하며, Rewrite_Table 을 생성하기 위해서는 utlxrw.sql 을 수행하여야 합니다.

다음의 결과는 SUM_SALES Mview 의 SQL 문장과 같으며, Query Rewrite 를 사용 가능하다는 것을 보여주고 있습니다.

만약 이 결과 중 Query Rewrite 를 사용할 수 없다는 결과가 나온다면, 10g 에서 제공되는 DBMS_ADVISOR.TUNE_MVIEW 를 이용하여 해당 Mview 를 TUNING 할 수 있는 정보를 얻어 낼 수 있습니다.

DBMS_ADVISOR.TUNE_MVIEW

- 10g 에서 소개.
- Privilege : advisor
- USER_TUNE_MVIEW

```
set autotrace off
```

```
declare task_cust_mv VARCHAR2(20):= 'tune_cust_mv';
begin
  DBMS_ADVISOR.TUNE_MVIEW(task_cust_mv,
    'CREATE MATERIALIZED VIEW cust_mv
    ENABLE Query Rewrite AS
    Select s.prod_id, s.cust_id, SUM(s.amount_sold) SUM_amount
    FROM sales s, customers cs
    WHERE s.cust_id = cs.cust_id
    GROUP BY s.prod_id, s.cust_id' );
end;
/
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 DBMS_ADVISOR.TUNE_MVIEW 에 대해서 소개하고 있습니다.

DBMS_ADVISOR.TUNE_MVIEW Package 는 10g 에서 소개가 되었으며, 해당 Package 를 사용하기 위해서는 ADVISOR 권한이 필요합니다.

예제로 위의 SLIDE 와 같이 Mview 를 생성하고자 하는 경우에 대해서 TUNE_MVIEW 를 수행하게 되면 TUNING 된 정보를 USER_TUNE_MVIEW 에 결과를 저장하게 됩니다.

결과로는 Mview 뿐만 아니라 Mview Log 에 대한 정보까지 함께 보여주게 됩니다.

USER_TUNE_MVIEW

```
SQL> Select * from user_tune_MVIEW ;
```

TASK_NAME	ACTION_ID	SCRIPT_TYPE
-----------	-----------	-------------

STATEMENT

tune_cust_mv	3 IMPLEMENTATION	CREATE MATERIALIZED VIEW LOG ON "SH"."CUSTOMERS" WITH ROWID, SEQUENCE ("CUST_ID") INCLUDING NEW VALUES
--------------	------------------	--

tune_cust_mv	4 IMPLEMENTATION	ALTER MATERIALIZED VIEW LOG FORCE ON "SH"."CUSTOMERS" ADD rowid, SEQUENCE ("CUST_ID") INCLUDING NEW VALUES
--------------	------------------	--

tune_cust_mv	5 IMPLEMENTATION	CREATE MATERIALIZED VIEW LOG ON "SH"."SALES" WITH ROWID, SEQUENCE ("PROD_ID","CUST_ID","AMOUNT_SOLD") INCLUDING NEW VALUES
--------------	------------------	--

tune_cust_mv	6 IMPLEMENTATION	ALTER MATERIALIZED VIEW LOG FORCE ON "SH"."SALES" ADD rowid, SEQUENCE ("PROD_ID","CUST_ID","AMOUNT_SOLD") INCLUDING NEW VALUES
--------------	------------------	--

tune_cust_mv	7 IMPLEMENTATION	CREATE MATERIALIZED VIEW SH.CUST_MV REFRESH FAST WITH ROWID ENABLE Query Rewrite AS Select SH.SALES.PROD_ID C1, SH.CUSTOMERS.CUST_ID C2, SUM("SH"."SALES"."AMOUNT_SOLD") M1, COUNT("SH"."SALES"."AMOUNT_SOLD") M2, COUNT(*) M3 FROM SH.SALES,SH.CUSTOMERS WHERE SH.CUSTOMERS.CUST_ID = SH.SALES.CUST_ID GROUP BY SH.SALES.PROD_ID, SH.CUSTOMERS.CUST_ID
--------------	------------------	---

tune_cust_mv	8 UNDO	DROP MATERIALIZED VIEW SH.CUST_MV
--------------	--------	-----------------------------------

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 DBMS_ADVISOR.TUNE_MVIEW 를 수행 후 USER_TUNE_MVIEW 의 결과를 보여주고 있습니다.

이 예제에서 확인할 수 있는 사항은

현재 해당 Mview 에 대한 Mview Log 가 생성이 되어 있지 않음을 보여 주고 있으며, 집합 연산을 하는 Mview 를 생성하는 내용이므로 Mview Log 를 Rowid Option 과 함께 Column 정보 및 INCLUDING NEW VALUES 를 사용하여 Mview Log 를 생성해 주는 것을 Guide 하고 있으며, Mview 를 생성하는 Script 를 제공하고 있습니다.

이 기능을 이용하게 되면 제약사항이 많은 Mview 를 보다 효과적으로 생성할 수 있습니다.

After Tuning Mview

```
SQL> set autotrace off
declare task_cust_mv VARCHAR2(20):= 'tune_cust_mv';
SQL> 2   begin
3       DBMS_ADVISOR.TUNE_MVIEW(task_cust_mv,
4       'CREATE MATERIALIZED VIEW SH.CUST_MV
5       REFRESH FAST WITH ROWID
6       ENABLE Query Rewrite
7       AS
8       Select SH.SALES.PROD_ID a1, SH.CUSTOMERS.CUST_ID a2,
9       SUM(SH.SALES.AMOUNT_SOLD) c1,
10      COUNT(SH.SALES.AMOUNT_SOLD) c2,
11      COUNT(*) c3
12      FROM SH.SALES, SH.CUSTOMERS
13      WHERE SH.CUSTOMERS.CUST_ID = SH.SALES.CUST_ID
14      GROUP BY SH.SALES.PROD_ID, SH.CUSTOMERS.CUST_ID' );
15  end;
```

다음의 내용은 USER_TUNE_MVIEW 의 결과를 바탕으로

Mview Log 을 생성한 이후 다음과 같이 하고 Tuning 된 Mview Script 를 이용하여 DBMS_ADVISOR.TUNE_MVIEW 를 실행하게 되면 이미 OPTIMAL 하다는 다음과 같은 MESSAGE 를 받게 됩니다.

이는 Error message 가 아니며, 단지 Message 입니다.

Result After Tuning Mview

- 해당 Mview statement 는 이미 tuning 이 되었음을 알리는 message

```
declare task_cust_mv VARCHAR2(20):= 'tune_cust_mv';
```

```
*
```

```
ERROR at line 1:
```

```
ORA-13600: error enCOUNTERed in Advisor
```

```
QSM-03116: The materialized view is already optimal and cannot be tuned any further
```

```
ORA-06512: at "SYS.DBMS_SYS_ERROR", line 86
```

```
ORA-06512: at "SYS.PRVT_ACCESS_ADVISOR", line 202
```

```
ORA-06512: at "SYS.PRVT_TUNE_MVIEW", line 1042
```

```
ORA-06512: at "SYS.DBMS_ADVISOR", line 754
```

```
ORA-06512: at line 3
```

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 는 이미 TUNING 된 Mview 를 DBMS_ADVISOR 를 이용하여 Tuning 하고자 하는 경우 보여 주는 Message 입니다.

QSM-3116 에서 이는 이미 OPTIMAL 한 Tuning 된 값이며 , 더 이상 Tuning 할 수 없다는 Message 를 보여주고 있습니다.

Truobleshooting

- Mview log 가 너무 커진 경우
 - 원인)
 - Refresh 작업이 너무 오랫동안 진행되지 못했을 때
 - Replication 환경에서 network failure 로 인해 Refresh 가 안되었을 경우
 - Replication 환경에서 Mview 가 drop 되었지만 Master site 에 반영이 되지 않는 경우
 - SOLUTION)
 - Refresh 작업을 모두 수행하여 Mview log 를 purge 한다.
 - Extent 가 너무 많이 발생한 경우에는 해당 Mview log 를 truncate 해 주어 FULL Table scan 시 적은 block 을 access 하게 한다.

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 Mview 를 운영 중 Mview Log 가 너무 커진 경우에 대한 원인과 대처 방법에 대해서 소개하고 있습니다.

Mview Log 는 Mview 가 Refresh 가 정상적으로 수행한 경우 Purge 가 되게 됩니다.

하지만 Mview 를 너무 오랫동안 Purge 하지 않게 되면 Mview Log 는 Mview 에 반영되지 않는 정보를 Purge 할 때까지 저장하게 됩니다.

이의 원인으로는 Manual 하게 Mview 에 대한 Refresh 작업이 이루어지지 않았을 경우, Replication 환경에서 Network 문제로 인하여 통신이 두절되었을 경우가 있습니다.

이와 같은 경우 Manual 하게 Refresh 를 해 주거나 Network 문제를 해결하여 Mview Log 를 Purge 해 주어야 합니다.

하지만 너무 많은 Extent 가 발생하게 되면 Refresh 시에 Mview Log를 FULL scan 함에 따라 Performance 에 영향을 줄 수 가 있습니다.

따라서 너무 많은 Extent 가 발생한 경우에는 해당 Mview Log 를 Truncate 하여 Extent 수를 줄여줌에 따라 Performance 를 향상시킬 수 있습니다.

Truobleshooting

- Mview log Table 의 extent 가 너무 많아 Refresh Performance 가 저하될 경우
 - LOCK Table scott.emp IN EXCLUSIVE MODE;
 - CREATE Table scott.templog AS Select * FROM scott.mlog\$_emp;
 - TRUNCATE scott.mlog\$_emp;
 - INSERT INTO scott.mlog\$_emp Select * FROM scott.templog;
 - DROP Table scott.templog;
- Mview log 를 수동으로 purge 하는 방법
 - DBMS_SNAPSHOT.PURGE_SNAPSHOT_FROM_LOG (SNAPOWNER => 'SCOTT', SNAPNAME => 'EMP', SNAPSITE => 'SNAP_SITE')
 - DBMS_SNAPSHOT.UNREGISTER_SNAPSHOT

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

본 SLIDE 에서는 이전 SLIDE 에서 언급되었던 Mview Log 를 Truncate 하는 방법을 언급하고 있습니다.

1. LOCK Table scott.emp IN EXCLUSIVE MODE;
2. CREATE Table scott.templog AS Select * FROM scott.mlog\$_emp;
3. TRUNCATE scott.mlog\$_emp;
4. INSERT INTO scott.mlog\$_emp Select * FROM scott.templog;
5. DROP Table scott.templog;

와 같은 방법으로 Mview Log 를 Truncate 할 수 있습니다.

그리고 Mview 가 Drop 되었을 시 Master Site 에 반영이 되지 않았을 경우

```
DBMS_SNAPSHOT.PURGE_SNAPSHOT_FROM_LOG (SNAPOWNER => 'SCOTT', SNAPNAME => 'EMP',  
SNAPSITE => 'SNAP_SITE')  
DBMS_SNAPSHOT.UNREGISTER_SNAPSHOT(SNAPOWNER => 'SCOTT', SNAPNAME => 'EMP',  
SNAPSITE => 'SNAP_SITE')
```

Package 를 이용하여 Mview Log Purge 및 Master Site 에 등록된 Mview 를 Unregister 할 수 있습니다.

Troubleshooting

- ORA-32314

- REFRESH FAST of "SCOTT"."DEPT_SAL" unsupported after deletes/updates
- 원인) insert 가 아닌 operation 이 수행됨
- SOLUTION) REFRESH COMPLETE

- ORA-12033

- cannot use filter columns from materialized view log on "SCOTT"."EMP"
- 원인) Mview 에서 reference 하는 column이 Mview log 에 포함이 되지 않은 경우
- SOLUTION) Mview log 에 Mview 가 reference 하는 column 을 추가.

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 SLIDE 는 Mview Refresh 시에 ORA-32314 Error 가 발생하는 경우로 Insert-only REFRESH FAST 를 지원하는 Mview 에 대한 Base Table 에 Insert 가 아닌 Update , Delete 등의 Operation 이 수행된 경우에 발생할 수 있습니다.

이에 대한 Solution 으로는 Complete 로 Refresh 해 주는 방법이 있습니다.

ORA-12033 이 발생하는 경우는 총합을 가지는 Mview 생성시 Reference 되는 Column이 Mview Log 에 포함되지 않았을 경우 발생하는 Error 입니다.

이에 대한 Solution 으로는 Mview Log 를 관련된 Column 을 추가하여 생성해 주어야 합니다.

Troubleshooting

- ORA-12015

- cannot create a fast Refresh materialized view from a complex Query
- 원인) REFRESH FAST restriction 에 위배되는 경우
- SOLUTION) REFRESH COMPLETE

- ORA-23413

- Table "SCOTT"."EMP" does not have a materialized view log
- 원인) Mview log 가 없는 경우
- SOLUTION) recreate Mview log and complete Refresh Mview

ORACLE

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 SLIDE 는 Mview 를 REFRESH FAST Mode 로 생성시 ORA-12015 Error 가 발생한 경우입니다.

이 경우는 REFRESH FAST Restriction 을 위배한 경우에 발생하게 되며 이는 REFRESH COMPLETE 형태로 Mview 를 생성해야 합니다.

운영 중이던 Mview 를 Refresh 하는 상황에서 ORA-23413 Error 가 발생하는 경우는 Mview 가 Reference 하는 Mview Log 가 삭제되었기 때문입니다.

이는 Mview Log 를 생성한 후 Mview 를 REFRESH COMPLETE 를 수행해야 합니다.

또는 Mview 를 다시 생성하셔야 합니다.

Troubleshooting

- **ORA-12052**

- cannot fast Refresh materialized view
SH.DETAIL_SALES_MV
- 원인) rowid 가 포함되지 않은 경우
- SOLUTION) Mview 생성시 base Table 의 rowid 를 Select List 에 넣어 주어야 한다. 또는 REFRESH COMPLETE 로 생성.

- **ORA-12032**

- cannot use rowid column from materialized view log on
"SH"."CUSTOMERS"
- 원인) Mview log 에 rowid가 없는 경우
- SOLUTION) recreate Mview log WITH ROWID and recreate Mview

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

다음의 SLIDE 는 Join 절을 가지는 Mview 를 REFRESH FAST Mode 로 생성시 ORA-12052 Error 가 발생한 경우입니다.

이 경우는 Join 절을 가지는 Mview 를 생성시 REFRESH FAST Restriction 을 위배한 경우에 발생하게 되며 이는 Mview 의 Select List 에 Rowid 를 추가해 주거나 REFRESH COMPLETE 형태로 Mview 를 생성해야 합니다.

Join 절을 가지는 Mview 를 REFRESH FAST Mode 로 생성시 ORA-12032 Error 가 발생한 경우입니다.

이 경우는 Join 절을 가지는 Mview 를 생성시 Mview Log 에서 Rowid 를 참조할 수 없는 경우에 발생하게 되며, 이는 Mview Log 생성시 WITH ROWID 형태로 생성을 해야 합니다.

Reference

- Data Warehousing Concept
- Advanced Replication Concept
- NOTE : 76673.1
- NOTE : 252727.1
- NOTE : 236233.1

ORACLE®

The Overview Of Mview

기술적인 질문은 채팅으로

DW Warehousing Concept

Advanced Replication Concept

[Note 76673.1](#)

Title: What are Materialized Views?

[Note 252727.1](#)

Title: Oracle10g: Using DBMS_ADVISOR.TUNE_MVIEW to Optimize Materialized View for Query Rewrite

[Note 236233.1](#)

Title: Materialized View Refresh : Log Population and Purge