即時マテリアライズド・ビュー

マテリアライズド・ビュー (Materialized View：MV) **は、データベース内のベース・テーブルとして、クエリの結果を保存します。クエリはベース・テーブルからの大量のデータを集約するため、通常はその計算結果を保存するためにマテリアライズド・ビューを使用します。マテリアライズド・ビューは、ビューやテーブルといった他のデータベース・オブジェクトと比べて、オプティマイザとの対話に優れているという利点があります。またマテリアライズド・ビューは、コストのかかるクエリの結果を保存して、あらかじめ集中的に作業を進めることにより、パフォーマンスの向上を実現します。**

SQL Anywhere 10 **の場合、マテリアライズド・ビューはビューがベースとしているベース・テーブルが変更されると、即座に** 'Stale' **になります。マテリアライズド・ビューのコンテンツを更新するためには、定期的に手作業でビューをリフレッシュする必要があります。マテリアライズド・ビューをリフレッシュすると、マテリアライズド・ビューのコンテンツは削除され、新しいクエリの結果と入れ替わってしまいます。**SQL Anywhere 11 **を使用すれば、マテリアライズド・ビューが自動的に最新データを保持するように設定できます。このため、マテリアライズド・ビューがベースとする基本テーブルが** (**挿入／更新／削除によって**) **変更されると、マテリアライズド・ビューも更新され、これらの変更内容が反映されます。**

即時ビューのサンプル

このサンプルでは、マテリアライズド・ビューを即座にメンテナンスするデモを実演します。この機能により、開発者がアプリケーションでマテリアライズド・ビューを使用する際の負担を軽減するとともに、最適化を行う際にオプティマイザが実体化ビューの利点を生かすことができます。

1. Interactive SQL **を起動し、**ODBC **から** SQL Anywhere 11 Demo **データベースに接続します。**

Windows **の場合は、コマンド・プロンプトを開きます。**

Linux **の場合は、ターミナルを開きます。**SQL Anywhere **のサンプルがインストールされているかの確認が完了していない場合は、**
$SQLANY11/samples/sample_config32.sh **のファイルを実行して確認します。**

以下のコマンドを実行します。

dbisql -c "dsn＝SQL Anywhere 11 Demo"

2. **各製品の売上高に従い、サイズと年度ごとにマテリアライズド・ビューを作成します (ユニット、ドル)。このビューを、手作業でリフレッシュするビューに指定します。**Interactive SQL **で、**[F5] **を押して以下の文を実行します。**

```
CREATE MATERIALIZED VIEW groupo.mv_manualrefresh AS
    SELECT year( s.shipdate) AS yr, p.name AS name, p.size AS size,
          sum(s.quantity) AS total_quantity,
          sum(p.unitprice) AS total_price,
          count(*) AS num_records
    FROM salesorderitems s, products p
    WHERE p.id = s.productid
    GROUP BY name, size, yr;

CREATE UNIQUE INDEX mvidx ON groupo.mv_manualrefresh( yr,
name, size );
ALTER MATERIALIZED VIEW groupo.mv_manualrefresh MANUAL
REFRESH;
```

3. **各製品の売上高に応じて、サイズと年度ごとにもう一つマテリアライズド・ビューを作成します (ユニット、ドル)。ここでは、自動的にリフレッシュ可能なビューに指定します (キーワード** IMMEDIATE REFRESH **を使用することに注意してください)。**

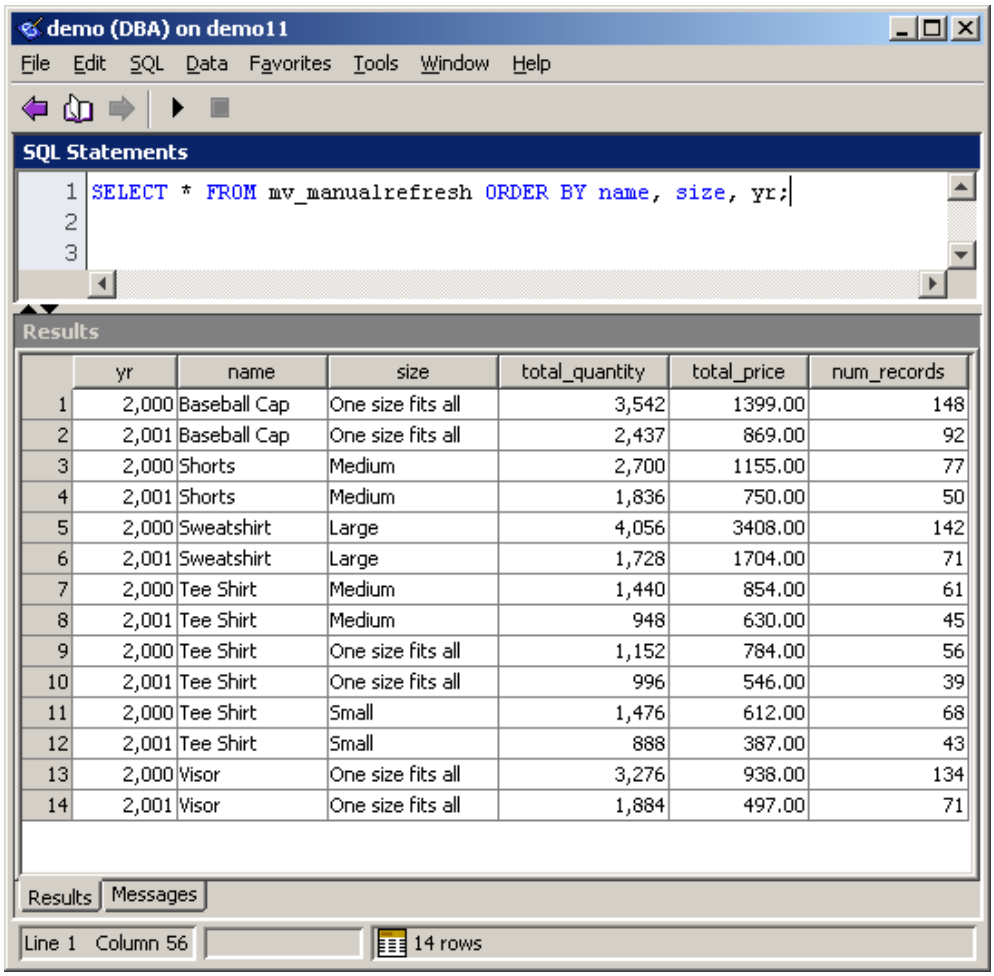
```
CREATE MATERIALIZED VIEW groupo.mv_autorefresh AS
    SELECT year( s.shipdate) AS yr, p.name AS name, p.size AS size,
          sum(s.quantity) AS total_quantity,
          sum(p.unitprice) AS total_price,
          count(*) AS num_records
    FROM salesorderitems s, products p
    WHERE p.id = s.productid
    GROUP BY name, size, yr;

CREATE UNIQUE INDEX mvidx ON groupo.mv_autorefresh( yr, name,
size );
ALTER MATERIALIZED VIEW groupo.mv_autorefresh IMMEDIATE
REFRESH;
```

4. **以下の** SQL **文を実行してマテリアライズド・ビューにデータを移植します。**

```
REFRESH MATERIALIZED VIEW groupo.mv_manualrefresh;
REFRESH MATERIALIZED VIEW groupo.mv_autorefresh;
```

5. **以下のクエリを実行して、ビューのコンテンツを取り出します。**

```
SELECT * FROM mv_manualrefresh ORDER BY name, size, yr;
```



	yr	name	size	total_quantity	total_price	num_records
1	2,000	Baseball Cap	One size fits all	3,542	1399.00	148
2	2,001	Baseball Cap	One size fits all	2,437	869.00	92
3	2,000	Shorts	Medium	2,700	1155.00	77
4	2,001	Shorts	Medium	1,836	750.00	50
5	2,000	Sweatshirt	Large	4,056	3408.00	142
6	2,001	Sweatshirt	Large	1,728	1704.00	71
7	2,000	Tee Shirt	Medium	1,440	854.00	61
8	2,001	Tee Shirt	Medium	948	630.00	45
9	2,000	Tee Shirt	One size fits all	1,152	784.00	56
10	2,001	Tee Shirt	One size fits all	996	546.00	39
11	2,000	Tee Shirt	Small	1,476	612.00	68
12	2,001	Tee Shirt	Small	888	387.00	43
13	2,000	Visor	One size fits all	3,276	938.00	134
14	2,001	Visor	One size fits all	1,884	497.00	71

6. **以下の** SQL **バッチを実行して、データベースに販売注文を数件追加します。**

```
BEGIN
   DECLARE lastid INTEGER;
   INSERT INTO salesorders
      VALUES (DEFAULT, 101, today(), 'r1', 'Eastern', 299);
   SELECT @@identity INTO lastid;
   INSERT INTO salesorderitems VALUES (lastid, 1, 300, 12,
today());
   INSERT INTO salesorderitems VALUES (lastid, 2, 400, 29,
today());
   INSERT INTO salesorderitems VALUES (lastid, 3, 500, 8, today());
   COMMIT;
END;
```

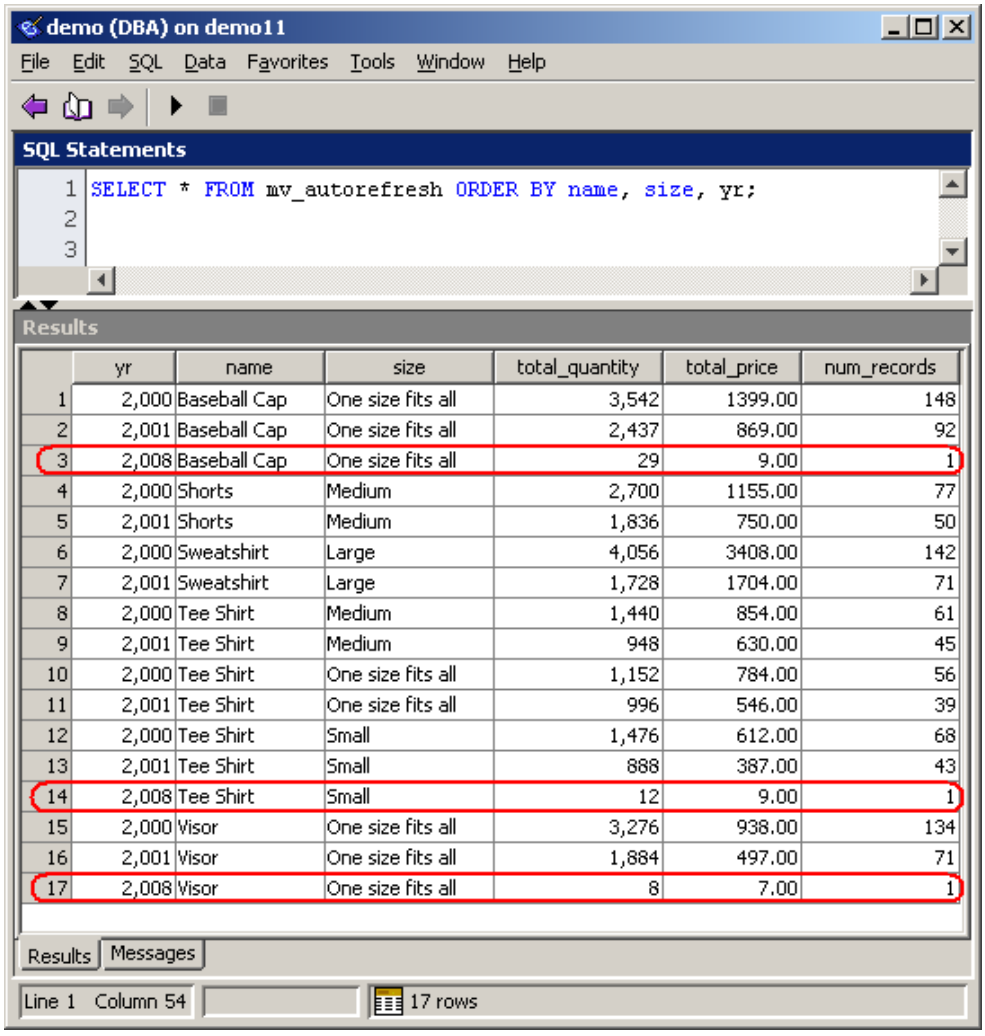
7. **新しい注文を追加した場合の影響を確認するために、以下のコマンドを実行して、マニュアル・ビュー** mv_manualrefresh **のコンテンツを確認します。**

   SELECT * FROM mv_manualrefresh ORDER BY name, size, yr;

   新しい販売注文を追加しても、結果一覧には何も変化がないことを確認してください。

8. **以下のコマンドを実行して、即時ビュー** mv_autorefresh **のコンテンツを確認してください。**

   SELECT * FROM mv_autorefresh ORDER BY name, size, yr;



	yr	name	size	total_quantity	total_price	num_records
1	2,000	Baseball Cap	One size fits all	3,542	1399.00	148
2	2,001	Baseball Cap	One size fits all	2,437	869.00	92
3	2,008	Baseball Cap	One size fits all	29	9.00	1
4	2,000	Shorts	Medium	2,700	1155.00	77
5	2,001	Shorts	Medium	1,836	750.00	50
6	2,000	Sweatshirt	Large	4,056	3408.00	142
7	2,001	Sweatshirt	Large	1,728	1704.00	71
8	2,000	Tee Shirt	Medium	1,440	854.00	61
9	2,001	Tee Shirt	Medium	948	630.00	45
10	2,000	Tee Shirt	One size fits all	1,152	784.00	56
11	2,001	Tee Shirt	One size fits all	996	546.00	39
12	2,000	Tee Shirt	Small	1,476	612.00	68
13	2,001	Tee Shirt	Small	888	387.00	43
14	2,008	Tee Shirt	Small	12	9.00	1
15	2,000	Visor	One size fits all	3,276	938.00	134
16	2,001	Visor	One size fits all	1,884	497.00	71
17	2,008	Visor	One size fits all	8	7.00	1

マテリアライズド・ビューは自動的に更新され、新しい販売注文の追加が反映されていることを確認してください。

結論

SQL Anywhere Ver.11 **は、ベース・データ・テーブルの変更を検知すると、即時マテリアライズド・ビューを自動的にリフレッシュします。即時ビューを作成する場合は、**ALTER MATERIALIZED VIEW … IMMEDIATE REFRESH **の文を実行して、マテリアライズド・ビューのリフレッシュ方式を 即時 に変更します。または、** Sybase Central **管理ツールを使用して、グラフィカルに実行することも可能です。ただし、即時ビューで作業する場合は、一定の制約について注意が必要なため、詳細についてはオンライン・マニュアルを確認してください。**