66

DBMS_DBCOMP

The DBMS_DBCOMP package performs block comparison to detect lost writes or database inconsistencies between a primary database and one or more physical standby databases. It contains one procedure, DBCOMP, which can be executed at any time. (It does not require that the DB_LOST_WRITE_PROTECT initialization parameter be enabled.)

This chapter contains the following sections:

- Using DBMS_DBCOMP
- DBMS_DBCOMP Security Model
- DBCOMP Procedure

Related Topics

- *Oracle Data Guard Concepts and Administration*

Using DBMS_DBCOMP

The DBMS_DBCOMP.DBCOMP procedure assumes that there is one primary database and one or more physical standby databases. The databases should be at least mounted before block comparison.

> **Note:**
>
> Logical standby databases, far sync instances, and cascaded standbys cannot be the target database for the DBMS_DBCOMP.DBCOMP procedure.

While this procedure is running, you can monitor progress of the compare operation by querying the V$SESSION_LONGOPS view. For example:

```
SQL> SELECT target_desc, sofar, totalwork FROM V$SESSION_LONGOPS WHERE opname
= 'BlockCompare';
```

The output might look as follows:

```
TARGET_DESC                    SO FAR    TOTALWORK
------------------------------ --------- ----------
Compared Blocks                23914     340639
Lost Writes                    0         0
```

The results can be interpreted as follows:

- The Compared Blocks value is the number of blocks that were scanned (including those that did not use compare callback functions). The number is the sum of all secondary processes (that is, each secondary process is responsible for a connection between primary and standby). For example, if there are two secondary processes and each secondary process has compared 1000 blocks, then the value of SOFAR should be 2000.

- The Lost Writes value is the number of lost writes found at both the primary and the standby. Also, it is the sum of all secondary processes. For Lost Writes, the value of SOFAR should be always equal to the value of TOTALWORK.
- The value of SOFAR should be always less than or equal to the value of TOTALWORK.
- When the block compare secondary processes finish, the value of SOFAR should be equal to the value of TOTALWORK.

DBMS_DBCOMP Security Model

The DBMS_DBCOMP package runs with invoker's rights and requires the SYSDBA or SYSDG or SYSBACKUP privilege.

Summary of the DBMS_DBCOMP Subprogram

The DBMS_DBCOMP package contains one procedure, DBCOMP.

Subprogram	Description
DBCOMP procedure	The DBMS_DBCOMP.DBCOMP procedure performs block comparison to detect lost writes or database inconsistencies between a primary database and one or more physical standby databases. The DBCOMP procedure can be executed on the primary or on the physical standby while the databases are mounted or open.

DBCOMP Procedure

Use DBCOMP procedure to compare blocks on a primary database and one or more physical standby databases.

Syntax

```
DBMS_DBCOMP.DBCOMP (
  datafile IN varchar2,
  outputfile IN varchar2,
  block_dump IN boolean);
```

Parameters

Parameter	Description
datafile	This can be a data file name or a data file number. Specify 'ALL' to compare all data files.
outputfile	This value is used as a prefix in the name of the output file. By default, output files are placed in the dbs/ directory, but the output file argument can contain a relative directory path or an absolute directory path. See Usage Notes for descriptions of the information contained in the output file.

Parameter	Description
`block_dump`	This is a Boolean parameter to specify whether block content should be dumped into output files if a pair of blocks of primary and standby databases is not identical. This parameter does not guarantee that all different block pairs are dumped. The default value is `FALSE`.

Usage Notes

- The output file contains the following columns:
  - **ID**: The block type. For instance, type 02 is undo block type, and 06 is KTB-managed data block type.
  - **TOTAL**: The total number of blocks in the specific block type.
  - **CORR**: The number of corrupted blocks.
  - **SKIPPED**: The number of block pairs that are skipped (ignored) without comparison, such as direct loaded blocks, RMAN optimized blocks, and flashback optimized blocks
  - **DIFFV**: The number of blocks that have different version numbers between primary and standby.
  - **SAMEV**: The number of block pairs that have the same version number between primary and standby.
  - **SAMEV&C**: The number of block pairs that have the same version and the same checksum.
  - **ENCERR**: The number of block pairs that have encryption related errors (usually because the Wallet is not open).
  - **LWLOC**: The number of lost writes on the local database.
  - **LWRMT**: The number of lost writes on the remote databases.
  - **DIFFPAIR** : The number of block pairs that have the same version but different checksum, and which the block compare callback function has concluded are different.

Example 66-1 Example

The following is an example of using the `DBMS_DBCOMP.DBCOMP` procedure.

```
declare
  Datafile_Name_or_Number varchar2(1000);
  Output_File varchar2(1000);
begin
  dbms_output.enable(1000000);
  Datafile_Name_or_Number:= 'all' ;
  Output_File:='BlockCompareAll_';
  sys.dbms_dbcomp.DBCOMP(Datafile_Name_or_Number, Output_File, true);
end;
```