162
DBMS_REPAIR

The DBMS_REPAIR package contains data corruption repair procedures that enable you to detect and repair corrupt blocks in tables and indexes. You can address corruptions where possible and continue to use objects while you attempt to rebuild or repair them.

This chapter contains the following topics:

- Overview
- Security Model
- Constants
- Operating Notes
- Exceptions
- Examples
- Summary of DBMS_REPAIR Subprograms

> **See Also:**
>
> For detailed information about using the DBMS_REPAIR package, see *Oracle Database Administrator's Guide*

DBMS_REPAIR Overview

The DBMS_REPAIR package is intended for use by database administrators only. It is not intended for use by application developers.

DBMS_REPAIR Security Model

The package is owned by SYS. Execution privilege is not granted to other users.

DBMS_REPAIR Constants

The DBMS_REPAIR package defines several enumerated constants that should be used for specifying parameter values. Enumerated constants must be prefixed with the package name. For example, DBMS_REPAIR.TABLE_OBJECT.

The following table lists the parameters and the enumerated constants.

Table 162-1 DBMS_REPAIR Parameters with Enumerated Constants

Parameter	Option	Type	Description
object_type	• TABLE_OBJECT • INDEX_OBJECT • CLUSTER_OBJECT	BINARY_INTEGER	-
action	• CREATE_ACTION • DROP_ACTION • PURGE_ACTION	BINARY_INTEGER	-
table_type	• REPAIR_TABLE • ORPHAN_TABLE	BINARY_INTEGER	-
flags	• SKIP_FLAG • NOSKIP_FLAG	BINARY_INTEGER	-
object_id	• ALL_INDEX_ID := 0	BINARY_INTEGER	Clean up all objects that qualify
wait_for_lock	• LOCK_WAIT := 1 • LOCK_NOWAIT := 0	BINARY_INTEGER	Specifies whether to try getting DML locks on underlying table [[sub]partition] object

> **Note:**
>
> The default table_name will be REPAIR_TABLE when table_type is REPAIR_TABLE, and will be ORPHAN_KEY_TABLE when table_type is ORPHAN_TABLE.

DBMS_REPAIR Operating Notes

The procedure to create the ORPHAN_KEYS_TABLE is similar to the one used to create the REPAIR_TABLE.

```
CONNECT / AS SYSDBA;
EXEC DBMS_REPAIR.ADMIN_TABLES('ORPHAN_KEYS_TABLE', DBMS_REPAIR.ORPHAN_TABLE,
                              DBMS_REPAIR.CREATE_ACTION);
EXEC DBMS_REPAIR.ADMIN_TABLES('REPAIR_TABLE', DBMS_REPAIR.REPAIR_TABLE,
                              DBMS_REPAIR.CREATE_ACTION);
DESCRIBE ORPHAN_KEYS_TABLE;
DESCRIBE REPAIR_TABLE;
SELECT * FROM ORPHAN_KEYS_TABLE;
SELECT * FROM REPAIR_TABLE;
```

The DBA would create the repair and orphan keys tables once. Subsequent executions of the CHECK_OBJECT Procedure would add rows into the appropriate table indicating the types of errors found.

The name of the repair and orphan keys tables can be chosen by the user, with the following restriction: the name of the repair table must begin with the 'REPAIR_' prefix, and the name of the orphan keys table must begin with the 'ORPHAN_' prefix. The following code is also legal:

```
CONNECT / AS SYSDBA;
EXEC DBMS_REPAIR.ADMIN_TABLES('ORPHAN_FOOBAR', DBMS_REPAIR.ORPHAN_TABLE,
                              DBMS_REPAIR.CREATE_ACTION);
EXEC DBMS_REPAIR.ADMIN_TABLES('REPAIR_ABCD', DBMS_REPAIR.REPAIR_TABLE,
                              DBMS_REPAIR.CREATE_ACTION);
DESCRIBE ORPHAN_FOOBAR;
DESCRIBE REPAIR_ABCD;
SELECT * FROM ORPHAN_FOOBAR;
SELECT * FROM REPAIR_ABCD;
```

When invoking the CHECK_OBJECT Procedure the name of the repair and orphan keys tables that were created should be specified correctly, especially if the default values were not used in the ADMIN_TABLES Procedure or CREATE_ACTION.

Other actions in the ADMIN_TABLES Procedure can be used to purge/delete the REPAIR_TABLE and the ORPHAN_KEYS_TABLE.

DBMS_REPAIR Exceptions

The table in this topic describes the exceptions raised by the DDBMS_REPAIR subprograms.

Table 162-2 DBMS_REPAIR Exceptions

Exception	Description	Action
942	Reported by DBMS_REPAIR.ADMIN_TABLES during a DROP_ACTION when the specified table doesn't exist.	-
955	Reported by DBMS_REPAIR. CREATE_ACTION when the specified table already exists.	-
24120	An invalid parameter was passed to the specified DBMS_REPAIR procedure.	Specify a valid parameter value or use the parameter's default.
24122	An incorrect block range was specified.	Specify correct values for the BLOCK_START and BLOCK_END parameters.
24123	An attempt was made to use the specified feature, but the feature is not yet implemented.	Do not attempt to use the feature.
24124	An invalid ACTION parameter was specified.	Specify CREATE_ACTION, PURGE_ACTION or DROP_ACTION for the ACTION parameter.
24125	An attempt was made to fix corrupt blocks on an object that has been dropped or truncated since DBMS_REPAIR.CHECK_OBJECT was run.	Use DBMS_REPAIR.ADMIN_TABLES to purge the repair table and run DBMS_REPAIR.CHECK_OBJECT to determine whether there are any corrupt blocks to be fixed.
24127	TABLESPACE parameter specified with an ACTION other than CREATE_ACTION.	Do not specify TABLESPACE when performing actions other than CREATE_ACTION.
24128	A partition name was specified for an object that is not partitioned.	Specify a partition name only if the object is partitioned.
24129	An attempt was made to pass a table name parameter without the specified prefix.	Pass a valid table name parameter.
24130	An attempt was made to specify a repair or orphan table that does not exist.	Specify a valid table name parameter.
24131	An attempt was made to specify a repair or orphan table that does not have a correct definition.	Specify a table name that refers to a properly created table.

Table 162-2 (Cont.) DBMS_REPAIR Exceptions

Exception	Description	Action
24132	An attempt was made to specify a table name is greater than 30 characters long.	Specify a valid table name parameter.

DBMS_REPAIR Examples

This topic shows examples of DBMS_REPAIR usage.

```
/* Fix the bitmap status for all the blocks in table mytab in schema sys */

EXECUTE DBMS_REPAIR.SEGMENT_FIX_STATUS('SYS', 'MYTAB');

/* Mark block number 45, filenumber 1 for table mytab in sys schema as FULL.*/

EXECUTE DBMS_REPAIR.SEGMENT_FIX_STATUS('SYS', 'MYTAB', TABLE_OBJECT,1, 45, 1);
```

Summary of DBMS_REPAIR Subprograms

This table lists the DBMS_REPAIR subprograms and briefly describes them.

Table 162-3 DBMS_REPAIR Package Subprograms

Subprogram	Description
ADMIN_TABLES Procedure	Provides administrative functions for the DBMS_REPAIR package repair and orphan key tables, including create, purge, and drop functions
CHECK_OBJECT Procedure	Detects and reports corruptions in a table or index
DUMP_ORPHAN_KEYS Procedure	Reports on index entries that point to rows in corrupt data blocks
FIX_CORRUPT_BLOCKS Procedure	Marks blocks software corrupt that have been previously detected as corrupt by CHECK_OBJECT
ONLINE_INDEX_CLEAN Function	Performs a manual cleanup of failed or interrupted online index builds or rebuilds
REBUILD_FREELISTS Procedure	Rebuilds an object's freelists
SEGMENT_FIX_STATUS Procedure	Fixes the corrupted state of a bitmap entry
SKIP_CORRUPT_BLOCKS Procedure	Sets whether to ignore blocks marked corrupt during table and index scans or to report ORA-1578 when blocks marked corrupt are encountered

ADMIN_TABLES Procedure

This procedure provides administrative functions for the DBMS_REPAIR package repair and orphan key tables.

Syntax

```
DBMS_REPAIR.ADMIN_TABLES (
   table_name  IN   VARCHAR2,
   table_type  IN   BINARY_INTEGER,
   action       IN   BINARY_INTEGER,
   tablespace   IN   VARCHAR2  DEFAULT NULL);
```

Parameters

Table 162-4 ADMIN_TABLES Procedure Parameters

Parameter	Description
`table_name`	Name of the table to be processed. Defaults to `ORPHAN_KEY_TABLE` or `REPAIR_TABLE` based on the specified `table_type`. When specified, the table name must have the appropriate prefix: `ORPHAN_` or `REPAIR_`.
`table_type`	Type of table; must be either `ORPHAN_TABLE` or `REPAIR_TABLE`. See "Constants".
`action`	Indicates what administrative action to perform. Must be either `CREATE_ACTION`, `PURGE_ACTION`, or `DROP_ACTION`. If the table already exists, and if `CREATE_ACTION` is specified, then an error is returned. `PURGE_ACTION` indicates to delete all rows in the table that are associated with non-existent objects. If the table does not exist, and if `DROP_ACTION` is specified, then an error is returned. When `CREATE_ACTION` and `DROP_ACTION` are specified, an associated view named `DBA_<table_name>` is created and dropped respectively. The view is defined so that rows associated with non-existent objects are eliminated. Created in the `SYS` schema. See "Constants".
`tablespace`	Indicates the tablespace to use when creating a table. By default, the `SYS` default tablespace is used. An error is returned if the tablespace is specified and if the action is not `CREATE_ACTION`.

CHECK_OBJECT Procedure

This procedure checks the specified objects and populates the repair table with information about corruptions and repair directives.

Validation consists of block checking all blocks in the object.

Syntax

```
DBMS_REPAIR.CHECK_OBJECT (
   schema_name       IN  VARCHAR2,
   object_name       IN  VARCHAR2,
   partition_name    IN  VARCHAR2       DEFAULT NULL,
   object_type       IN  BINARY_INTEGER DEFAULT TABLE_OBJECT,
   repair_table_name IN  VARCHAR2       DEFAULT 'REPAIR_TABLE',
   flags             IN  BINARY_INTEGER DEFAULT NULL,
   relative_fno      IN  BINARY_INTEGER DEFAULT NULL,
   block_start       IN  BINARY_INTEGER DEFAULT NULL,
   block_end         IN  BINARY_INTEGER DEFAULT NULL,
   corrupt_count     OUT BINARY_INTEGER);
```

Parameters

Table 162-5   CHECK_OBJECT Procedure Parameters

Parameter	Description
`schema_name`	Schema name of the object to be checked.
`object_name`	Name of the table or index to be checked.
`partition_name`	Partition or subpartition name to be checked. If this is a partitioned object, and if `partition_name` is not specified, then all partitions and subpartitions are checked. If this is a partitioned object, and if the specified partition contains subpartitions, then all subpartitions are checked.
`object_type`	Type of the object to be processed. This must be either `TABLE_OBJECT` (default) or `INDEX_OBJECT`. See "Constants".
`repair_table_name`	Name of the repair table to be populated. The table must exist in the `SYS` schema. Use the `ADMIN_TABLES` Procedure to create a repair table. The default name is `REPAIR_TABLE`.
`flags`	Reserved for future use.
`relative_fno`	Relative file number: Used when specifying a block range.
`block_start`	First block to process if specifying a block range. May be specified only if the object is a single table, partition, or subpartition.
`block_end`	Last block to process if specifying a block range. May be specified only if the object is a single table, partition, or subpartition. If only one of `block_start` or `block_end` is specified, then the other defaults to the first or last block in the file respectively.
`corrupt_count`	Number of corruptions reported.

Usage Notes

You may optionally specify a DBA range, partition name, or subpartition name when you want to check a portion of an object.

DUMP_ORPHAN_KEYS Procedure

This procedure reports on index entries that point to rows in corrupt data blocks. For each such index entry encountered, a row is inserted into the specified orphan table.

If the repair table is specified, then any corrupt blocks associated with the base table are handled in addition to all data blocks that are marked software corrupt. Otherwise, only blocks that are marked corrupt are handled.

This information may be useful for rebuilding lost rows in the table and for diagnostic purposes.

Syntax

```
DBMS_REPAIR.DUMP_ORPHAN_KEYS (
   schema_name       IN  VARCHAR2,
   object_name       IN  VARCHAR2,
   partition_name    IN  VARCHAR2       DEFAULT NULL,
   object_type       IN  BINARY_INTEGER DEFAULT INDEX_OBJECT,
   repair_table_name IN  VARCHAR2       DEFAULT 'REPAIR_TABLE',
   orphan_table_name IN  VARCHAR2       DEFAULT 'ORPHAN_KEYS_TABLE',
   flags             IN  BINARY_INTEGER DEFAULT NULL,
   key_count         OUT BINARY_INTEGER);
```

Parameters

Table 162-6   DUMP_ORPHAN_KEYS Procedure Parameters

Parameter	Description
`schema_name`	Schema name.
`object_name`	Object name.
`partition_name`	Partition or subpartition name to be processed. If this is a partitioned object, and if `partition_name` is not specified, then all partitions and subpartitions are processed. If this is a partitioned object, and if the specified partition contains subpartitions, then all subpartitions are processed.
`object_type`	Type of the object to be processed. The default is `INDEX_OBJECT` See "Constants".
`repair_table_name`	Name of the repair table that has information regarding corrupt blocks in the base table. The specified table must exist in the `SYS` schema. The `ADMIN_TABLES` Procedure is used to create the table.
`orphan_table_name`	Name of the orphan key table to populate with information regarding each index entry that refers to a row in a corrupt data block. The specified table must exist in the `SYS` schema. The `ADMIN_TABLES` Procedure is used to create the table.
`flags`	Reserved for future use.
`key_count`	Number of index entries processed.

FIX_CORRUPT_BLOCKS Procedure

This procedure fixes the corrupt blocks in specified objects based on information in the repair table that was previously generated by the CHECK_OBJECT Procedure.

Prior to effecting any change to a block, the block is checked to ensure the block is still corrupt. Corrupt blocks are repaired by marking the block software corrupt. When a repair is effected, the associated row in the repair table is updated with a fix timestamp.

Syntax

```
DBMS_REPAIR.FIX_CORRUPT_BLOCKS (
   schema_name       IN  VARCHAR2,
   object_name       IN  VARCHAR2,
   partition_name    IN  VARCHAR2       DEFAULT NULL,
   object_type       IN  BINARY_INTEGER DEFAULT TABLE_OBJECT,
   repair_table_name IN  VARCHAR2       DEFAULT 'REPAIR_TABLE',
   flags             IN  BINARY_INTEGER DEFAULT NULL,
   fix_count         OUT BINARY_INTEGER);
```

Parameters

Table 162-7 FIX_CORRUPT_BLOCKS Procedure Parameters

Parameter	Description
`schema_name`	Schema name.
`object_name`	Name of the object with corrupt blocks to be fixed.
`partition_name`	Partition or subpartition name to be processed. If this is a partitioned object, and if `partition_name` is not specified, then all partitions and subpartitions are processed. If this is a partitioned object, and if the specified partition contains subpartitions, then all subpartitions are processed.
`object_type`	Type of the object to be processed. This must be either `TABLE_OBJECT` (default) or `INDEX_OBJECT`. See "Constants".
`repair_table_name`	Name of the repair table with the repair directives. Must exist in the `SYS` schema.
`flags`	Reserved for future use.
`fix_count`	Number of blocks fixed.

Related Topics

- CHECK_OBJECT Procedure
  This procedure checks the specified objects and populates the repair table with information about corruptions and repair directives.

ONLINE_INDEX_CLEAN Function

This function performs a manual cleanup of failed or interrupted online index builds or rebuilds.

This action is also performed periodically by SMON, regardless of user-initiated cleanup.

This function returns `TRUE` if all indexes specified were cleaned up and `FALSE` if one or more indexes could not be cleaned up.

Syntax

```
DBMS_REPAIR.ONLINE_INDEX_CLEAN (
   object_id      IN BINARY_INTEGER DEFAULT ALL_INDEX_ID,
   wait_for_lock  IN BINARY_INTEGER DEFAULT LOCK_WAIT)
 RETURN BOOLEAN;
```

Parameters

Table 162-8 ONLINE_INDEX_CLEAN Function Parameters

Parameter	Description
`object_id`	Object id of index to be cleaned up. The default cleans up all object ids that qualify.

Table 162-8 (Cont.) ONLINE_INDEX_CLEAN Function Parameters

Parameter	Description
wait_for_lock	This parameter specifies whether to try getting DML locks on underlying table [[sub]partition] object. The default retries up to an internal retry limit, after which the lock get will give up. If LOCK_NOWAIT is specified, then the lock get does not retry.

REBUILD_FREELISTS Procedure

This procedure rebuilds the freelists for the specified object.

All free blocks are placed on the master freelist. All other freelists are zeroed.

If the object has multiple freelist groups, then the free blocks are distributed among all freelists, allocating to the different groups in round-robin fashion.

Syntax

```
DBMS_REPAIR.REBUILD_FREELISTS (
   schema_name    IN VARCHAR2,
   object_name    IN  VARCHAR2,
   partition_name IN VARCHAR2 DEFAULT NULL,
   object_type    IN BINARY_INTEGER DEFAULT TABLE_OBJECT);
```

Parameters

Table 162-9 REBUILD_FREELISTS Procedure Parameters

Parameter	Description
schema_name	Schema name.
object_name	Name of the object whose freelists are to be rebuilt.
partition_name	Partition or subpartition name whose freelists are to be rebuilt. If this is a partitioned object, and partition_name is not specified, then all partitions and subpartitions are processed. If this is a partitioned object, and the specified partition contains subpartitions, then all subpartitions are processed.
object_type	Type of the object to be processed. This must be either TABLE_OBJECT (default) or INDEX_OBJECT. See"Constants".

SEGMENT_FIX_STATUS Procedure

With this procedure you can fix the corrupted state of a bitmap entry. The procedure either recalculates the state based on the current contents of the corresponding block or sets the state to a specific value.

Syntax

```
DBMS_REPAIR.SEGMENT_FIX_STATUS (
   segment_owner   IN VARCHAR2,
   segment_name    IN VARCHAR2,
   segment_type    IN BINARY_INTEGER DEFAULT TABLE_OBJECT,
   file_number     IN BINARY_INTEGER DEFAULT NULL,
   block_number    IN BINARY_INTEGER DEFAULT NULL,
   status_value    IN BINARY_INTEGER DEFAULT NULL,
   partition_name  IN VARCHAR2 DEFAULT NULL,);
```

Parameters

Table 162-10 SEGMENT_FIX_STATUS Procedure Parameters

Parameter	Description
`schema_owner`	Schema name of the segment.
`segment_name`	Segment name.
`partition_name`	Optional. Name of an individual partition. `NULL` for nonpartitioned objects. Default is `NULL`.
`segment_type`	Optional Type of the segment (for example, `TABLE_OBJECT` or `INDEX_OBJECT`). Default is `NULL`.
`file_number`	(optional) The tablespace-relative file number of the data block whose status has to be fixed. If omitted, all the blocks in the segment will be checked for state correctness and fixed.
`block_number`	(optional) The file-relative block number of the data block whose status has to be fixed. If omitted, all the blocks in the segment will be checked for state correctness and fixed.
`status_value`	(optional) The value to which the block status described by the `file_number` and `block_number` will be set. If omitted, the status will be set based on the current state of the block. This is almost always the case, but if there is a bug in the calculation algorithm, the value can be set manually. Status values: • 1 = block is full • 2 = block is 0-25% free • 3 = block is 25-50% free • 4 = block is 50-75% free • 5 = block is 75-100% free The status for bitmap blocks, segment headers, and extent map blocks cannot be altered. The status for blocks in a fixed hash area cannot be altered. For index blocks, there are only two possible states: 1 = block is full and 3 = block has free space.

SKIP_CORRUPT_BLOCKS Procedure

This procedure enables or disables the skipping of corrupt blocks during index and table scans of the specified object.

When the object is a table, skip applies to the table and its indexes. When the object is a cluster, it applies to all of the tables in the cluster, and their respective indexes.

> **Note:**
>
> When Oracle performs an index range scan on a corrupt index after DBMS_REPAIR.SKIP_CORRUPT_BLOCKS has been set for the base table, corrupt branch blocks and root blocks are not skipped. Only corrupt non-root leaf blocks are skipped.

Syntax

```
DBMS_REPAIR.SKIP_CORRUPT_BLOCKS (
   schema_name  IN VARCHAR2,
   object_name  IN VARCHAR2,
   object_type  IN BINARY_INTEGER DEFAULT TABLE_OBJECT,
   flags        IN BINARY_INTEGER DEFAULT SKIP_FLAG);
```

Parameters

Table 162-11 SKIP_CORRUPT_BLOCKS Procedure Parameters

Parameter	Description
`schema_name`	Schema name of the object to be processed.
`object_name`	Name of the object.
`object_type`	Type of the object to be processed. This must be either `TABLE_OBJECT` (default) or `CLUSTER_OBJECT`. See "Constants".
`flags`	If `SKIP_FLAG` is specified, then it turns on the skip of software corrupt blocks for the object during index and table scans. If `NOSKIP_FLAG` is specified, then scans that encounter software corrupt blocks return an `ORA-1578`. See"Constants".