287
UTL_COMPRESS

The UTL_COMPRESS package provides a set of data compression utilities.

This chapter contains the following topics:

- Constants
- Exceptions
- Operational Notes
- Summary of UTL_COMPRESS Subprograms

UTL_COMPRESS Constants

The maximum number of handles for piecewise operations can be defined by a constant.

```
UTLCOMP_MAX_HANDLE   CONSTANT  PLS_INTEGER := 5;
```

UTL_COMPRESS Exceptions

This table describes exceptions raised by UTL_COMPRESS subprograms.

Table 287-1   UTL_COMPRESS Exceptions

Exception	Description
BUFFER_TOO_SMALL	The compressed representation is too big.
DATA_ERROR	The input or output data stream was found to be an invalid format.
INVALID_ARGUMENT	One of the arguments was an invalid type or value.
INVALID_HANDLE	Invalid handle for piecewise compress or uncompress.
STREAM_ERROR	An error occurred during compression or uncompression of the data stream

UTL_COMPRESS Operational Notes

Certain operational notes apply to UTL_COMPRESS.

- It is the caller's responsibility to free the temporary LOB returned by the LZ* functions with DBMS_LOB.FREETEMPORARY call.
- A BFILE passed into LZ_COMPRESS* or lZ_UNCOMPRESS* has to be opened by DBMS_LOB.FILEOPEN.
- Under special circumstances (especially if the input has already been compressed) the output produced by one of the UTL_COMPRESS subprograms may be the same size, or even slightly larger than, the input.

- The output of the `UTL_COMPRESS` compressed data is compatible with `gzip`(with `-n` option)/ `gunzip` on a single file.

Summary of UTL_COMPRESS Subprograms

This table lists the `UTL_COMPRESS` subprograms and briefly describes them.

Table 287-2   *UTL_COMPRESS Package Subprograms*

Subprogram	Description
ISOPEN Function	Checks to see if the handle to a piecewise (un)compress context is open or closed
LZ_COMPRESS Functions and Procedures	Compresses data using Lempel-Ziv compression algorithm
LZ_COMPRESS_ADD Procedure	Adds a piece of compressed data
LZ_COMPRESS_CLOSE	Closes and finishes piecewise compress operation
LZ_COMPRESS_OPEN	Initializes a piecewise context that maintains the compress state and data
LZ_UNCOMPRESS Functions and Procedures	Accepts compressed input, verifies it to be a valid and uncompresses it
LZ_UNCOMPRESS_EXTRACT Procedure	Extracts a piece of uncompressed data
LZ_UNCOMPRESS_OPEN Function	Initializes a piecewise context that maintains the uncompress state and data
LZ_UNCOMPRESS_CLOSE Procedure	Closes and finishes the piecewise uncompress

ISOPEN Function

This function checks to see if the handle to a piecewise (un)compress context is open or closed.

Syntax

```
UTL_COMPRESS.ISOPEN(
   handle in binary_integer)
 RETURN BOOLEAN;
```

Parameters

Table 287-3   ISOPEN Function Parameters

Parameter	Description
`handle`	The handle to a piecewise uncompress context.

Return Values

`TRUE` if the given piecewise handle is opened, otherwise `FALSE`.

Examples

```
IF (UTL_COMPRESS.ISOPEN(myhandle) = TRUE) then
   UTL_COMPRESS.LZ_COMPRESS_CLOSE(myhandle, lob_1);
END IF;
```

Alternatively:

```
IF (UTL_COMPRESS.ISOPEN(myhandle) = TRUE) THEN
   UTL_COMPRESS.LZ_UNCOMPRESS_CLOSE(myhandle);
END IF;
```

LZ_COMPRESS Functions and Procedures

These functions and procedures compress data using Lempel-Ziv compression algorithm.

Syntax

This function accept a `RAW` as input, compress it and return the compressed `RAW` result and metadata:

```
UTL_COMPRESS.LZ_COMPRESS (
   src       IN           RAW,
   quality   IN           BINARY_INTEGER DEFAULT 6)
 RETURN RAW;
```

This function accept a `BLOB` as input, compress it and returns a temporary BLOB for the compressed data:

```
UTL_COMPRESS.LZ_COMPRESS (
  src       IN           BLOB,
  quality   IN           BINARY_INTEGER DEFAULT 6)
 RETURN BLOB;
```

This procedure returns the compressed data into the existing BLOB(dst) which is trimmed to the compressed data size:

```
UTL_COMPRESS.LZ_COMPRESS (
  src      IN            BLOB,
  dst      IN OUT NOCOPY BLOB,
  quality  IN            BINARY_INTEGER DEFAULT 6);
```

This function returns a temporary `BLOB` for the compressed data:

```
UTL_COMPRESS.LZ_COMPRESS (
   src     IN            BFILE,
   quality IN            BINARY_INTEGER DEFAULT 6)
 RETURN BLOB;
```

This procedure will return the compressed data into the existing BLOB(dst) which is trimmed to the compressed data size:

```
UTL_COMPRESS.LZ_COMPRESS (
   src     IN            BFILE,
   dst     IN OUT NOCOPY BLOB,
   quality IN            BINARY_INTEGER DEFAULT 6);
```

Parameters

Table 287-4 LZ_COMPRESS Function and Procedures Parameters

Parameter	Description
`src`	Data (`RAW`, `BLOB or BFILE`) to be compressed.
`dst`	Destination for compressed data
`quality`	An integer in the range 1 to 9, 1=fast compression, 9=best compression, default=6

Usage Notes

- `quality` is an optional compression tuning value. It allows the `UTL_COMPRESS` user to choose between speed and compression quality, meaning the percentage of reduction in size. A faster compression speed will result in less compression of the data. A slower compression speed will result in more compression of the data. Valid values are [1..9], with 1=fastest and 9=slowest. The default 'quality' value is 6.

LZ_COMPRESS_ADD Procedure

This procedure adds a piece of compressed data.

Syntax

```
UTL_COMPRESS.LZ_COMPRESS_ADD (
   handle IN            BINARY_INTEGER,
   dst    IN OUT NOCOPY  BLOB,
   src    IN            RAW);
```

Parameters

Table 287-5 LZ_COMPRESS_ADD Procedure Parameters

Parameter	Description
`handle`	The handle to a piecewise compress context.
`dst`	The opened `LOB` from `LZ_COMPRESS_OPEN` to store compressed data.
`src`	The input data to be compressed.

Exceptions

- `invalid_handle` - out of range invalid or unopened handle.
- `invalid_argument` - `NULL` handle, src, dst, or invalid `dst`.

LZ_COMPRESS_CLOSE

This procedure closes and finishes piecewise compress operation.

Syntax

```
UTL_COMPRESS.LZ_COMPRESS_CLOSE (
   handle IN            BINARY_INTEGER,
   dst    IN OUT NOCOPY  BLOB);
```

Parameters

Table 287-6 LZ_COMPRESS_CLOSE Procedure Parameters

Parameter	Description
handle	The handle to a piecewise compress context.
dst	The opened LOB from LZ_COMPRESS_OPEN to store compressed data.

Exceptions

- invalid_handle - out of range invalid or uninitialized handle.
- invalid_argument - NULL handle, dst, or invalid dst.

LZ_COMPRESS_OPEN

This function initializes a piecewise context that maintains the compress state and data.

Syntax

```
UTL_COMPRESS.LZ_COMPRESS_OPEN (
   dst       IN OUT NOCOPY BLOB,
   quality   IN            BINARY_INTEGER DEFAULT 6)
 RETURN BINARY_INTEGER;
```

Parameters

Table 287-7 LZ_COMPRESS_OPEN Function Parameters

Parameter	Description
dst	User supplied LOB to store compressed data.
quality	Speed versus efficiency of resulting compressed output. • Valid values are the range 1..9, with a default value of 6. • 1=fastest compression, 9=slowest compression and best compressed file size.

Return Values

A handle to an initialized piecewise compress context.

Exceptions

- invalid_handle - invalid handle, too many open handles.
- invalid_argument - NULL dst or invalid quality specified.

Usage Notes

Close the opened handle with LZ_COMPRESS_CLOSE

- once the piecewise compress is completed
- in the event of an exception in the middle of process

because lack of doing so will cause these handles to leak.

LZ_UNCOMPRESS Functions and Procedures

This procedure accepts as input a RAW, BLOB or BFILE compressed string, verifies it to be a valid compressed value, uncompresses it using Lempel-Ziv compression algorithm, and returns the uncompressed RAW or BLOB result.

Syntax

This function returns uncompressed data as RAW:

```
UTL_COMPRESS.LZ_UNCOMPRESS(
   src   IN   RAW)
  RETURN RAW;
```

This function returns uncompressed data as a temporary BLOB:

```
UTL_COMPRESS.LZ_UNCOMPRESS(
   src   IN   BLOB)
  RETURN BLOB;
```

This procedure returns the uncompressed data into the existing BLOB(dst), which will be trimmed to the uncompressed data size:

```
UTL_COMPRESS.LZ_UNCOMPRESS(
   src   IN   BLOB,
   dst   IN OUT NOCOPY BLOB);
```

This function returns a temporary BLOB for the uncompressed data:

```
UTL_COMPRESS.LZ_UNCOMPRESS(
   src   IN BFILE)
 RETURN BLOB;
```

This procedure returns the uncompressed data into the existing BLOB(dst). The original dst data will be overwritten.

```
UTL_COMPRESS.LZ_UNCOMPRESS(
   src   IN   BFILE,
   dst   IN OUT NOCOPY BLOB);
```

Parameters

Table 287-8   LZ_UNCOMPRESS Function and Procedures Parameters

Parameter	Description
src	Compressed data.
dst	Destination for uncompressed data.

LZ_UNCOMPRESS_EXTRACT Procedure

This procedure extracts a piece of uncompressed data.

Syntax

```
UTL_COMPRESS.LZ_UNCOMPRESS_EXTRACT(
   handle  IN          BINARY_INTEGER,
   dst     OUT NOCOPY  RAW);
```

Parameters

Table 287-9 LZ_UNCOMPRESS_EXTRACT Function Parameters

Parameter	Description
`handle`	The handle to a piecewise uncompress context.
`dst`	The uncompressed data.

Exceptions

- `no_data_found` - finished uncompress.
- `invalid_handle` - out of range invalid or uninitialized handle.
- `invalid_argument` - NULL handle.

LZ_UNCOMPRESS_OPEN Function

This function initializes a piecewise context that maintains the uncompress state and data.

Syntax

```
UTL_COMPRESS.LZ_UNCOMPRESS_OPEN(
   src   IN   BLOB)
  RETURN BINARY_INTEGER;
```

Parameters

Table 287-10 LZ_UNCOMPRESS_OPEN Function Parameters

Parameter	Description
`src`	The input data to be uncompressed.

Return Values

A handle to an initialized piecewise compress context.

Exceptions

- `invalid_handle` - invalid handle, too many open handles.
- `invalid_argument` - `NULL` `src`.

Usage Notes

Close the opened handle with `LZ_UNCOMPRESS_CLOSE`

- once the piecewise uncompress is completed
- in the event of an exception in the middle of process

because lack of doing so will cause these handles to leak.

LZ_UNCOMPRESS_CLOSE Procedure

This procedure closes and finishes the piecewise uncompress.

Syntax

```
UTL_COMPRESS.LZ_UNCOMPRESS_CLOSE(
   handle   IN    BINARY_INTEGER);
```

Parameters

Table 287-11   LZ_UNCOMPRESS_CLOSE Procedure Parameters

Parameter	Description
handle	The handle to a piecewise uncompress context.

Exceptions

- invalid_handle - out of range invalid or uninitialized handle.
- invalid_argument - NULL handle.