302 UTL_SMTP

The `UTL_SMTP` package is designed for sending electronic mails (e-mails) over Simple Mail Transfer Protocol (SMTP) as specified by RFC821.

This chapter contains the following topics:

See Also:

Oracle Database Development Guide

UTL_SMTP Overview

The `UTL_SMTP` protocol consists of a set of commands for an e-mail client to dispatch e-mails to an SMTP server. The `UTL_SMTP` package provides interfaces to the SMTP commands.

For many of the commands, the package provides both a procedural and a functional interface. The functional form returns the reply from the server for processing by the client. The procedural form checks the reply and raises an exception if the reply indicates a transient (400-range reply code) or permanent error (500-range reply code). Otherwise, it discards the reply.

Note that the original SMTP protocol communicates using 7-bit ASCII. Using `UTL_SMTP`, all text data (in other words, those in `VARCHAR2`) is converted to US7ASCII before it is sent to the server. Some implementations of SMTP servers that support SMTP extension 8BITMIME [RFC1652] support full 8-bit communication between client and server. The body of the DATA command can be transferred in full 8 bits, but the rest of the SMTP command and response must be in 7 bits. When the target SMTP server supports 8BITMIME extension, users of multibyte databases may convert their non-US7ASCII, multibyte `VARCHAR2` data to `RAW` and use the `WRITE_RAW_DATA` subprogram to send multibyte data using 8-bit MIME encoding.

`UTL_SMTP` provides for SMTP communication as specified in RFC821, but does not provide an API to format the content of the message according to RFC 822 (for example, setting the subject of an electronic mail). You must format the message appropriately. In addition, UTL_SMTP does not have the functionality to implement an SMTP server for an e-mail clients to send e-mails using SMTP.

UTL_SMTP Security Model

This package is an invoker's rights package. The invoking user must have the connect privilege granted in the access control list assigned to the remote network host to which the user must connect.

Note:

For more information on managing fine-grained access, see *Oracle Database Security Guide*

UTL_SMTP Constants

UTL_SMTP defines several constants to use when specifying parameter values.

These are shown in the following table.

Table 302-1 UTL_SMTP Constants

Name	Type	Value	Description
ALL_SCHEMES	VARCHAR2(256)	'CRAM-MD5 PLAIN LOGIN'	List of all authentication schemes UTL_SMTP supports, in order of their relative security strength. The subset of the schemes in ALL_SCHEMES (namely, PLAIN and LOGIN) in which cleartext passwords are sent over SMTP must be used only in SMTP connections that are secured by Secure Socket Layer / Transport Layer Security (SSL/TLS).
NON_CLEARTEXT_PASSWORD_SCHEMES	VARCHAR2(256)	'CRAM-MD5'	List of authentication schemes that UTL_SMTP supports and in which no cleartext passwords are sent over SMTP. They can be used in SMTP connections that are not secured by SSL/TLS. Note that these schemes may still be weak when used in an insecure SMTP connection.

UTL_SMTP Types

UTL_SMTP uses a CONNECTION record type and REPLY_REPLIES record types.

CONNECTION Record Type

This is a PL/SQL record type used to represent an SMTP connection.

Syntax

```
TYPE connection IS RECORD (
    host              VARCHAR2(255),
    port              PLS_INTEGER,
    tx_timeout        PLS_INTEGER,
    private_tcp_con   utl_tcp.connection,
    private_state     PLS_INTEGER);
```

Fields

Table 302-2   CONNECTION Record Type Fields

Field	Description
`host`	Name of the remote host when connection is established. `NULL` when no connection is established.
`port`	Port number of the remote SMTP server connected. `NULL` when no connection is established.
`tx_timeout`	Time in seconds that the `UTL_SMTP` package waits before timing out in a read or write operation in this connection. In read operations, this package times out if no data is available for reading immediately. In write operations, this package times out if the output buffer is full and no data is to be sent into the network without being blocked. `0` indicates not to wait at all. `NULL` indicates to wait forever.
`private_tcp_con`	Private, for implementation use only. You should not modify this field.
`private_state`	Private, for implementation use only. You should not modify this field.

Usage Notes

The read-only fields in a connection record are used to return information about the SMTP connection after the connection is successfully made with the OPEN_CONNECTION Functions. Changing the values of these fields has no effect on the connection. The fields `private_tcp_con` and `private_state` for implementation use only. You should not modify these fields.

REPLY_REPLIES Record Types

These are PL/SQL record types used to represent an SMTP reply line. Each SMTP reply line consists of a reply code followed by a text message. While a single reply line is expected for most SMTP commands, some SMTP commands expect multiple reply lines. For those situations, a PL/SQL table of reply records is used to represent multiple reply lines.

Syntax

```
TYPE reply IS RECORD (
  code    PLS_INTEGER,
  text    VARCHAR2(508));

TYPE replies IS TABLE OF reply INDEX BY BINARY_INTEGER;
```

Fields

Table 302-3 REPLY, REPLIES Record Type Fields

Field	Description
`code`	3-digit reply code
`text`	Text message of the reply

UTL_SMTP Reply Codes

SMTP servers send reply codes that indicate message or server status.

The following is a list of the SMTP reply codes.

Table 302-4 SMTP Reply Codes

Reply Code	Meaning
`211`	System status, or system help reply
`214`	Help message [Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user]
`220`	`<domain>` Service ready
`221`	`<domain>` Service closing transmission channel
`250`	Requested mail action okay, completed
`251`	User not local; forwards to `<forward-path>`
`252`	OK, pending messages for node `<node>` started. Cannot `VRFY` user (for example, info is not local), but takes message for this user and attempts delivery.
`253`	OK, `<messages>` pending messages for node `<node>` started
`354`	Start mail input; end with `<CRLF.CRLF>`
`355`	Octet-offset is the transaction offset
`421`	`<domain>` Service not available, closing transmission channel (This can be a reply to any command if the service knows it must shut down.)
`450`	Requested mail action not taken: mailbox unavailable [for example, mailbox busy]
`451`	Requested action terminated: local error in processing
`452`	Requested action not taken: insufficient system storage
`453`	You have no mail.
`454`	TLS not available due to temporary reason. Encryption required for requested authentication mechanism.
`458`	Unable to queue messages for node `<node>`
`459`	Node `<node>` not allowed: `reason`
`500`	Syntax error, command unrecognized (This may include errors such as command line too long.)
`501`	Syntax error in parameters or arguments
`502`	Command not implemented

Table 302-4 (Cont.) SMTP Reply Codes

Reply Code	Meaning
`503`	Bad sequence of commands
`504`	Command parameter not implemented
`521`	`<Machine>` does not accept mail.
`530`	Must issue a `STARTTLS` command first. Encryption required for requested authentication mechanism.
`534`	Authentication mechanism is too weak.
`538`	Encryption required for requested authentication mechanism.
`550`	Requested action not taken: mailbox unavailable [for, mailbox not found, no access]
`551`	User not local; please try `<forward-path>`
`552`	Requested mail action terminated: exceeded storage allocation
`553`	Requested action not taken: mailbox name not allowed [for example, mailbox syntax incorrect]
`554`	Transaction failed

UTL_SMTP Operational Notes

An SMTP connection is initiated by a call to `OPEN_CONNECTION` Functions which returns a SMTP connection.

After a connection is established, the following calls are required to send a mail:

HELO Function and Procedure - identify the domain of the sender

MAIL Function and Procedure- start a mail, specify the sender

RCPT Function - specify the recipient

OPEN_DATA Function and Procedure- start the mail body

WRITE_RAW_DATA Procedure - write the mail body (multiple calls allowed)

CLOSE_DATA Function and Procedure - close the mail body and send the mail

The SMTP connection is closed by calling the `QUIT` Function and Procedure.

Related Topics

- OPEN_CONNECTION Functions
  These functions open a connection to an SMTP server.
- QUIT Function and Procedure
  This subprogram terminates an SMTP session and disconnects from the server.

UTL_SMTP Exceptions

This table lists the exceptions that can be raised by the interface of the `UTL_SMTP` package.

The network error is transferred to a reply code of 421- service not available.

Table 302-5 UTL_SMTP Exceptions

Exception	Description
`INVALID_OPERATION`	Raised when an invalid operation is made. In other words, calling API other than the WRITE_DATA Procedure, the WRITE_RAW_DATA Procedure or the CLOSE_DATA Function and Procedure after the OPEN_DATA Function and Procedure is called, or calling `WRITE_DATA`, `WRITE_RAW_DATA` or `CLOSE_DATA` without first calling `OPEN_DATA`.
`TRANSIENT_ERROR`	Raised when receiving a reply code in 400 range
`PERMANENT_ERROR`	Raised when receiving a reply code in 500 range

UTL_SMTP Rules and Limits

The API imposes no imitation or range-checking. However, you must be aware of the limitations on various elements of SMTP. Sending data that exceed these limits may result in errors returned by the server.

The following table describes the size limitations encountered by the UTL_SMTP subprograms.

Table 302-6 SMTP Size Limitation

Element	Size Limitation
`user`	Maximum total length of a user name is 64 characters
`domain`	Maximum total length of a domain name or number is 64 characters
`path`	Maximum total length of a reverse-path or forward-path is 256 characters (including the punctuation and element separators)
`command line`	Maximum total length of a command line including the command word and the `<CRLF>` is 512 characters
`reply line`	Maximum total length of a reply line including the reply code and the `<CRLF>` is 512 characters
`text line`	Maximum total length of a text line including the `<CRLF>` is 1000 characters (but not counting the leading dot duplicated for transparency)
`recipients buffer`	Maximum total number of recipients that must be buffered is 100 recipients

UTL_SMTP Examples

This example illustrates how `UTL_SMTP` is used by an application to send e-mail. The application connects to an SMTP server at port 25 and sends a simple text message.

```
DECLARE
  c UTL_SMTP.CONNECTION;

  PROCEDURE send_header(name IN VARCHAR2, header IN VARCHAR2) AS
  BEGIN
    UTL_SMTP.WRITE_DATA(c, name || ': ' || header || UTL_TCP.CRLF);
  END;
```

```
BEGIN
  c := UTL_SMTP.OPEN_CONNECTION('smtp-server.acme.com');
  UTL_SMTP.HELO(c, 'foo.com');
  UTL_SMTP.MAIL(c, 'sender@foo.com');
  UTL_SMTP.RCPT(c, 'recipient@foo.com');
  UTL_SMTP.OPEN_DATA(c);
  send_header('From',    '"Sender" <sender@foo.com>');
  send_header('To',      '"Recipient" <recipient@foo.com>');
  send_header('Subject', 'Hello');
  UTL_SMTP.WRITE_DATA(c, UTL_TCP.CRLF || 'Hello, world!');
  UTL_SMTP.CLOSE_DATA(c);
  UTL_SMTP.QUIT(c);
EXCEPTION
  WHEN utl_smtp.transient_error OR utl_smtp.permanent_error THEN
    BEGIN
      UTL_SMTP.QUIT(c);
    EXCEPTION
      WHEN UTL_SMTP.TRANSIENT_ERROR OR UTL_SMTP.PERMANENT_ERROR THEN
        NULL; -- When the SMTP server is down or unavailable, we don't have
              -- a connection to the server. The QUIT call raises an
              -- exception that we can ignore.
    END;
    raise_application_error(-20000,
      'Failed to send mail due to the following error: ' || sqlerrm);
END;
```

Summary of UTL_SMTP Subprograms

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

Table 302-7 UTL_SMTP Package Subprograms

Subprogram	Description
AUTH Function and Procedure	Sends the `AUTH` command to authenticate to the SMTP server
CLOSE_CONNECTION Procedure	Closes the SMTP connection, causing the current SMTP operation to terminate
CLOSE_DATA Function and Procedure	Closes the data session
COMMAND Function and Procedure	Performs a generic SMTP command
COMMAND_REPLIES Function	Performs a generic SMTP command and retrieves multiple reply lines
DATA Function and Procedure	Sends the e-mail body
EHLO Function and Procedure	Performs the initial handshake with SMTP server using the `EHLO` command
HELO Function and Procedure	Performs the initial handshake with SMTP server using the `HELO` command
HELP Function	Sends `HELP` command
MAIL Function and Procedure	Initiates an e-mail transaction with the server, the destination is a mailbox
NOOP Function and Procedure	`NULL` command
OPEN_CONNECTION Functions	Opens a connection to an SMTP server

Table 302-7 (Cont.) UTL_SMTP Package Subprograms

Subprogram	Description
OPEN_DATA Function and Procedure	Sends the DATA command
QUIT Function and Procedure	Terminates an SMTP session and disconnects from the server
RCPT Function	Specifies the recipient of an e-mail message
RSET Function and Procedure	Terminates the current e-mail transaction
STARTTLS Function and Procedure	Sends STARTTLS command to secure the SMTP connection using SSL/TLS
VRFY Function	Verifies the validity of a destination e-mail address
WRITE_DATA Procedure	Writes a portion of the e-mail message
WRITE_RAW_DATA Procedure	Writes a portion of the e-mail message with RAW data

AUTH Function and Procedure

This subprogram sends the AUTH command to authenticate to the SMTP server. The UTL_SMTP package goes through the user's choices of authentication schemes, skips any that is not supported by the SMTP server and uses the first supported.

To determine the schemes the SMTP server supports from its EHLO reply, the user must call the EHLO Function and Procedure. Otherwise, UTL_SMTP uses the first scheme in the list.

Syntax

```
UTL_SMTP.AUTH (
   c          IN OUT NOCOPY connection,
   username   IN            VARCHAR2,
   password   IN            VARCHAR2,
   schemes    IN            VARCHAR2 DEFAULT NON_CLEARTEXT_PASSWORD_SCHEMES)
 RETURN reply;

UTL_SMTP.AUTH (
   c          IN OUT NOCOPY connection,
   username   IN            VARCHAR2,
   password   IN            VARCHAR2,
   schemes    IN            VARCHAR2 DEFAULT NON_CLEARTEXT_PASSWORD_SCHEMES);
```

Parameters

Table 302-8 AUTH Function and Procedure Parameters

Parameter	Description
c	SMTP connection
username	Username
password	Password
schemes	Space-separated list of authentication schemes UTL_SMTP is allowed to use in the preferred order. See the ALL_SCHEMES and NON_CLEARTEXT_PASSWORD_SCHEMES constants for suggestions.

Return Values

Table 302-9 AUTH Function and Procedure Function Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

- Currently only `PLAIN`, `LOGIN` and `CRAM-MD5` authentication schemes are supported by `UTL_SMTP`.
- Since the SMTP server may change the authentication schemes it supports after the SMTP connection is secured by SSL/TLS after the `STARTTLS` command (for example, adding `PLAIN` and `LOGIN`), the caller must call the EHLO Function and Procedure again for `UTL_SMTP` to update the list after the STARTTLS Function and Procedure is called.

Examples

```
DECLARE
  c utl_smtp.connection;
BEGIN
  c := utl_smtp.open_connection(
      host => 'smtp.example.com',
      port => 25,
      wallet_path => 'file:/oracle/wallets/smtp_wallet',
      wallet_password => 'password',
      secure_connection_before_smtp => FALSE);
  UTL_SMTP.STARTTLS(c);
  UTL_SMTP.AUTH(
      c => c,
      username => 'scott',
      password => 'password'
      schemes  => utl_smtp.all_schemes);
END;
```

CLOSE_CONNECTION Procedure

This procedure closes the SMTP connection, causing the current SMTP operation to terminate. Use this procedure only to cancel an e-mail in the middle of the data session.

To end the SMTP connection properly, use the QUIT Function and Procedure.

Syntax

```
UTL_SMTP.CLOSE_CONNECTION (
   c     IN OUT NOCOPY connection);
```

Parameters

Table 302-10 CLOSE_CONNECTION Procedure Parameters

Parameter	Description
`c`	SMTP connection

CLOSE_DATA Function and Procedure

This subprogram ends the e-mail message by sending the sequence `<CR><LF>.<CR><LF>` (a single period at the beginning of a line).

Syntax

```
UTL_SMTP.CLOSE_DATA (
   c     IN OUT NOCOPY connection)
RETURN reply;

UTL_SMTP.CLOSE_DATA (
   c     IN OUT NOCOPY connection);
```

Parameters

Table 302-11 CLOSE_DATA Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection

Return Values

Table 302-12 CLOSE_DATA Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

The calls to `OPEN_DATA`, `WRITE_DATA`, `WRITE_RAW_DATA` and `CLOSE_DATA` must be made in the right order. A program calls `OPEN_DATA` to send the `DATA` command to the SMTP server. After that, it can call `WRITE_DATA` or `WRITE_RAW_DATA` repeatedly to send the actual data. The data is terminated by calling `CLOSE_DATA`. After `OPEN_DATA` is called, the only subprograms that can be called are `WRITE_DATA`, `WRITE_RAW_DATA`, or `CLOSE_DATA`. A call to other subprograms results in an `INVALID_OPERATION` exception being raised.

`CLOSE_DATA` must be called only after `OPEN_CONNECTION`, `HELO` or `EHLO`, `MAIL`, and `RCPT` have been called. The connection to the SMTP server must be open and a mail transaction must be active when this routine is called.

Note that there is no function form of `WRITE_DATA` because the SMTP server does not respond until the data-terminator is sent during the call to `CLOSE_DATA`.

COMMAND Function and Procedure

This subprogram performs a generic SMTP command.

Syntax

```
UTL_SMTP.COMMAND (
   c     IN OUT NOCOPY    connection,
   cmd   IN                VARCHAR2,
   arg   IN                VARCHAR2  DEFAULT NULL)
RETURN reply;

UTL_SMTP.COMMAND (
   c      IN OUT NOCOPY    connection,
   cmd    IN               VARCHAR2,
   arg    IN               VARCHAR2  DEFAULT NULL);
```

Parameters

Table 302-13 COMMAND Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`cmd`	SMTP command to send to the server
`arg`	Optional argument to the SMTP argument. A space is inserted between `cmd` and `arg`.

Return Values

Table 302-14 COMMAND Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

This function is used to invoke generic SMTP commands. Use `COMMAND` if only a single reply line is expected. Use `COMMAND_REPLIES` if multiple reply lines are expected.

For `COMMAND`, if multiple reply lines are returned from the SMTP server, it returns the last reply line only.

COMMAND_REPLIES Function

This function performs a generic SMTP command and retrieves multiple reply lines.

Syntax

```
UTL_SMTP.COMMAND_REPLIES (
   c      IN OUT NOCOPY    connection,
   cmd    IN               VARCHAR2,
   arg    IN               VARCHAR2  DEFAULT NULL)
RETURN replies;
```

Parameters

Table 302-15 COMMAND_REPLIES Function Parameters

Parameter	Description
`c`	SMTP connection
`cmd`	SMTP command to send to the server
`arg`	Optional argument to the SMTP argument. A space is inserted between `cmd` and `arg`.

Return Values

Table 302-16 COMMAND_REPLIES Function Return Values

Return Value	Description
`replies`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types)

Usage Notes

This function is used to invoke generic SMTP commands. Use `COMMAND` if only a single reply line is expected. Use `COMMAND_REPLIES` if multiple reply lines are expected.

For `COMMAND`, if multiple reply lines are returned from the SMTP server, it returns the last reply line only.

DATA Function and Procedure

This subprogram specifies the body of an e-mail message.

Syntax

```
UTL_SMTP.DATA (
   c     IN OUT NOCOPY connection
   body  IN  VARCHAR2 CHARACTER SET ANY_CS)
RETURN reply;

UTL_SMTP.DATA (
   c     IN OUT NOCOPY connection
   body  IN VARCHAR2 CHARACTER SET ANY_CS);
```

Parameters

Table 302-17 DATA Function and Procedure Parameters

Parameter	Description
`c`	SMTP Connection
`body`	Text of the message to be sent, including headers, in [RFC822] format

Return Values

Table 302-18 DATA Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

The application must ensure that the contents of the body parameter conform to the MIME(RFC822) specification. The `DATA` routine terminates the message with a `<CR><LF>.<CR><LF>` sequence (a single period at the beginning of a line), as required by RFC821. It also translates any sequence of `<CR><LF>.<CR><LF>` (single period) in body to `<CR><LF>..<CR><LF>` (double period). This conversion provides the transparency as described in Section 4.5.2 of RFC821.

The `DATA` subprogram must be called only after `OPEN_CONNECTION`, `HELO` or `EHLO`, `MAIL` and `RCPT` have been called. The connection to the SMTP server must be open, and a mail transaction must be active when this routine is called.

The expected response from the server is a message beginning with status code 250. The 354 response received from the initial `DATA` command is not returned to the caller.

EHLO Function and Procedure

This subprogram performs the initial handshake with SMTP server using the `EHLO` command.

Syntax

```
UTL_SMTP.EHLO (
   c       IN OUT NOCOPY connection,
   domain  IN)
RETURN replies;

UTL_SMTP.EHLO (
   c       IN OUT NOCOPY connection,
   domain  IN);
```

Parameters

Table 302-19 EHLO Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`domain`	Domain name of the local (sending) host. Used for identification purposes.

Return Values

Table 302-20 EHLO Function and Procedure Return Values

Return Value	Description
`replies`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types).

Usage Notes

The `EHLO` interface is identical to `HELO` except that it allows the server to return more descriptive information about its configuration. [RFC1869] specifies the format of the information returned, which the PL/SQL application can retrieve using the functional form of this call. For compatibility with `HELO`, each line of text returned by the server begins with status code 250.

Related Functions

HELO Function and Procedure

HELO Function and Procedure

This subprogram performs the initial handshake with SMTP server using the `HELO` command.

Syntax

```
UTL_SMTP.HELO (
   c       IN OUT NOCOPY   connection,
   domain  IN              VARCHAR2)
RETURN reply;

UTL_SMTP.HELO (
   c       IN OUT NOCOPY   connection,
   domain  IN              VARCHAR2);
```

Parameters

Table 302-21 HELO Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`domain`	Domain name of the local (sending) host. Used for identification purposes.

Return Values

Table 302-22 HELO Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

RFC 821 specifies that the client must identify itself to the server after connecting. This routine performs that identification. The connection must have been opened through a call to OPEN_CONNECTION Functions before calling this routine.

The expected response from the server is a message beginning with status code 250.

Related Functions

EHLO Function and Procedure

HELP Function

This function sends the `HELP` command.

Syntax

```
UTL_SMTP.HELP (
   c          IN OUT NOCOPY   connection,
   command    IN              VARCHAR2 DEFAULT NULL)
RETURN replies;
```

Parameters

Table 302-23 HELP Function Parameters

Parameter	Description
`c`	SMTP connection
`command`	Command to get the help message

Return Values

Table 302-24 HELP Function Return Values

Return Value	Description
`replies`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types)

MAIL Function and Procedure

This subprogram initiate a mail transaction with the server. The destination is a mailbox.

Syntax

```
UTL_SMTP.MAIL (
   c            IN OUT NOCOPY   connection,
   sender       IN              VARCHAR2,
   parameters   IN              VARCHAR2 DEFAULT NULL)
RETURN reply;

UTL_SMTP.MAIL (
   c            IN OUT NOCOPY   connection,
   sender       IN              VARCHAR2,
   parameters   IN              VARCHAR2 DEFAULT NULL);
```

Parameters

Table 302-25 MAIL Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`sender`	E-mail address of the user sending the message.
`parameters`	Additional parameters to `mail` command as defined in Section 6 of [RFC1869]. It must follow the format of "XXX=XXX (XXX=XXX ....)".

Return Values

Table 302-26 MAIL Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTI_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

This command does not send the message; it simply begins its preparation. It must be followed by calls to `RCPT` and `DATA` to complete the transaction. The connection to the SMTP server must be open and a `HELO` or `EHLO` command must have already been sent.

The expected response from the server is a message beginning with status code 250.

NOOP Function and Procedure

This subprogram issues the `NULL` command.

Syntax

```
UTL_SMTP.NOOP (
   c  IN OUT NOCOPY connection)
RETURN reply;

UTL_SMTP.NOOP (
   c  IN OUT NOCOPY connection);
```

Parameter

Table 302-27 NOOP Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection

Return Values

Table 302-28 NOOP Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

- This command has no effect except to elicit a successful reply from the server. It can be issued at any time after the connection to the server has been established with `OPEN_CONNECTION`. The `NOOP` command can be used to verify that the server is still connected and is listening properly.
- This command replies with a single line beginning with status code 250.

OPEN_CONNECTION Functions

These functions open a connection to an SMTP server.

Syntax

```
UTL_SMTP.OPEN_CONNECTION (
   host                           IN  VARCHAR2,
   port                           IN  PLS_INTEGER DEFAULT 25,
   c                              OUT connection,
   tx_timeout                     IN  PLS_INTEGER DEFAULT NULL,
   wallet_path                    IN  VARCHAR2 DEFAULT NULL,
   wallet_password                IN  VARCHAR2 DEFAULT NULL,
   secure_connection_before_smtp  IN  BOOLEAN DEFAULT FALSE,
   secure_host                    IN  VARCHAR2 DEFAULT NULL)
 RETURN reply;

UTL_SMTP.OPEN_CONNECTION (
   host                          IN  VARCHAR2,
   port                          IN  PLS_INTEGER DEFAULT 25,
   tx_timeout                    IN  PLS_INTEGER DEFAULT NULL,
   wallet_path                   IN  VARCHAR2 DEFAULT NULL,
   wallet_password               IN  VARCHAR2 DEFAULT NULL,
   secure_connection_before_smtp IN  BOOLEAN DEFAULT FALSE,
   secure_host                   IN  VARCHAR2 DEFAULT NULL)
RETURN connection;
```

Parameters

Table 302-29 OPEN_CONNECTION Functions Parameters

Parameter	Description
`host`	Name of the SMTP server host
`port`	Port number on which SMTP server is listening (usually 25)
`c`	SMTP connection

Table 302-29 (Cont.) OPEN_CONNECTION Functions Parameters

Parameter	Description
`tx_timeout`	Time in seconds that the `UTL_SMTP` package waits before timing out in a read or write operation for this connection. In read operations, this package times out if no data is available for reading immediately. In write operations, this package times out if the output buffer is full and no data is to be sent into the network without being blocked. `0` indicates not to wait at all. `NULL` indicates to wait forever.
`wallet_path`	Directory path that contains the Oracle wallet for SSL/TLS. The format is `file:<directory-path>` If you want to use the operating system certificate store to act in place of the Oracle wallet, then set the `path` parameter to `system:` (include the colon). Doing so greatly improves performance in the database.
`wallet_password`	Password to open the wallet. When the wallet is auto-login enabled, the password can be set to `NULL`. If you set `path` to `system:`, then omit the password by setting it to `NULL`.
`secure_connection_before_smtp`	If `TRUE`, a secure connection with SSL/TLS is made before SMTP communication. If `FALSE`, no connection is made.
`secure_host`	The host name to be matched against the common name (CN) of the SMTP server's certificate when a secure connection is used. It can also be a domain name like "`*.example.com`". If `NULL`, the SMTP host name to connect to will be used.

Return Values

Table 302-30 OPEN_CONNECTION Functions Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

- The expected response from the server is a message beginning with status code 220.
- The version of `OPEN_CONNECTION` that returns `UTL_SMTP.CONNECTION` record checks the reply code returned by an SMTP server when the connection is first established. It raises an exception when the reply indicates an error. Otherwise, it discards the reply. If you want to examine the reply, invoke the version of `OPEN_CONNECTION` that returns `REPLY`.
- `tx_timeout` is intended to govern both the read operations and the write operations. However, an implementation restriction prevents `tx_timeout` from governing write operations in the current release.

Examples

```
DECLARE
  c utl_smtp.connection;
BEGIN
  c := UTL_SMTP.OPEN_CONNECTION(
   host => 'smtp.example.com',
   port => 465,
   wallet_path => 'file:/oracle/wallets/smtp_wallet',
   wallet_password => 'password',
   secure_connection_before_smtp => TRUE);
END;
```

OPEN_DATA Function and Procedure

This subprogram sends the `DATA` command after which you can use `WRITE_DATA` and `WRITE_RAW_DATA` to write a portion of the e-mail message.

Syntax

```
UTL_SMTP.OPEN_DATA (
   c      IN OUT NOCOPY connection)
RETURN reply;

UTL_SMTP.OPEN_DATA (
   c      IN OUT NOCOPY connection);
```

Parameters

Table 302-31 OPEN_DATA Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`data`	Portion of the text of the message to be sent, including headers, in RFC822 format.

Return Values

Table 302-32 OPEN_DATA Function and Procedure Function Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTI_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

- The calls to `OPEN_DATA`, `WRITE_DATA`, `WRITE_RAW_DATA` and `CLOSE_DATA` must be made in the right order. A program calls `OPEN_DATA` to send the `DATA` command to the SMTP server. After that, it can call `WRITE_DATA` or `WRITE_RAW_DATA` repeatedly to send the actual data. The data is terminated by calling `CLOSE_DATA`. After `OPEN_DATA` is called, the only subprograms that can be called are `WRITE_DATA`, `WRITE_RAW_DATA`, or `CLOSE_DATA`. A call to other subprograms results in an `INVALID_OPERATION` exception being raised.
- `OPEN_DATA` must be called only after `OPEN_CONNECTION`, `HELO` or `EHLO`, `MAIL`, and `RCPT` have been called. The connection to the SMTP server must be open and a mail transaction must be active when this routine is called.

QUIT Function and Procedure

This subprogram terminates an SMTP session and disconnects from the server.

Syntax

```
UTL_SMTP.QUIT (
   c  IN OUT NOCOPY connection)
RETURN reply;

UTL_SMTP.QUIT (
   c  IN OUT NOCOPY connection);
```

Parameter

Table 302-33 QUIT Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection

Return Values

Table 302-34 QUIT Function and Procedure Function Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

The `QUIT` command informs the SMTP server of the client's intent to terminate the session. It then closes the connection established by `OPEN_CONNECTION` which must have been called before executing this command. If a mail transaction is in progress when `QUIT` is issued, it is canceled in the same manner as `RSET`.

The function form of this command returns a single line beginning with the status code 221 on successful termination. In all cases, the connection to the SMTP server is closed. The fields `REMOTE_HOST` and `REMOTE_PORT` of `c` are reset.

Related Functions

RSET Function and Procedure

RCPT Function

This subprogram specifies the recipient of an e-mail message.

Syntax

```
UTL_SMTP.RCPT (
   c           IN OUT NOCOPY     connection,
   recipient   IN                VARCHAR2,
   parameters  IN                VARCHAR2 DEFAULT NULL)
```

```
RETURN reply;

UTL_SMTP.RCPT (
   c            IN OUT NOCOPY     connection,
   recipient    IN                VARCHAR2,
   parameters   IN                VARCHAR2 DEFAULT NULL);
```

Table 302-35   RCPT Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`recipient`	E-mail address of the user to which the message is being sent
`parameters`	Additional parameters to RCPT command as defined in Section 6 of [RFC1869]. It must follow the format of "XXX=XXX (XXX=XXX ....)".

Return Values

Table 302-36   RCPT Function and Procedure Function Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

To send a message to multiple recipients, call this routine multiple times. Each invocation schedules delivery to a single e-mail address. The message transaction must have been begun by a prior call to `MAIL`, and the connection to the mail server must have been opened and initialized by prior calls to `OPEN_CONNECTION` and `HELO` or `EHLO` respectively.

The expected response from the server is a message beginning with status code 250 or 251.

RSET Function and Procedure

This subprogram terminates the current mail transaction.

Syntax

```
UTL_SMTP.RSET (
   c  IN OUT NOCOPY connection)
RETURN reply;

UTL_SMTP.RSET (
   c  IN OUT NOCOPY connection);
```

Parameters

Table 302-37   RSET Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection

Return Values

Table 302-38 RSET Function and Procedure Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTl_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

- This command allows the client to cancel an e-mail message it was in the process of composing. No mail is sent. The client can call `RSET` at any time after the connection to the SMTP server has been opened by means of `OPEN_CONNECTION` until `DATA` or `OPEN_DATA` is called. Once the e-mail data has been sent, it is too late to prevent the e-mail from being sent.
- The server responds to `RSET` with a message beginning with status code 250.

Related Functions

QUIT Function and Procedure

STARTTLS Function and Procedure

This subprogram sends the `STARTTLS` command to secure the SMTP connection using SSL/TLS.

SSL/TLS requires an Oracle wallet which must be specified when the connection was opened by the OPEN_CONNECTION Functions.

Syntax

```
UTL_SMTP.STARTTLS (
   c             IN OUT NOCOPY  connection,
   secure_host   IN     VARCHAR2 DEFAULT NULL)
RETURN reply;

UTL_SMTP.STARTTLS (
   c             IN OUT NOCOPY connection,
   secure_host   IN     VARCHAR2 DEFAULT NULL);
```

Parameters

Table 302-39 STARTTLS Function and Procedure Parameters

Parameter	Description
`c`	SMTP connection
`secure_host`	The host name to be matched against the common name (CN) of the SMTP server's certificate. It can also be a domain name like `"*.example.com"`. If `NULL`, the SMTP host name to connect to will be used.

Return Values

Table 302-40 STARTTLS Function and Procedure Return Values

Return Value	Description
`reply`	SMTP reply

Usage Notes

The `STARTTLS` command must only be issued on an unencrypted connection and when the SMTP server indicates the support of the command in the reply of the `EHLO` command. The wallet to be used for encryption must have been specified when the initial SMTP connection was opened by the `OPEN_CONNECTION` function.

Examples

```
DECLARE
  c utl_smtp.connection;
BEGIN
  c := utl_smtp.open_connection(
     host => 'smtp.example.com',
     port => 25,
     wallet_path => 'file:/oracle/wallets/smtp_wallet',
     wallet_password => 'password',
     secure_connection_before_smtp => FALSE);
  utl_smtp.starttls(c);
END
```

VRFY Function

This function verifies the validity of a destination e-mail address.

Syntax

```
UTL_SMTP.VRFY (
   c          IN OUT NOCOPY connection
   recipient  IN VARCHAR2)
RETURN reply;
```

Parameters

Table 302-41 VRFY Function Parameters

Parameter	Description
`c`	SMTP connection
`recipient`	E-mail address to be verified

Return Values

Table 302-42 VRFY Function Return Values

Return Value	Description
`reply`	Reply of the command (see REPLY_ REPLIES Record Types in UTI_SMTP Types). In cases where there are multiple replies, the last reply is returned.

Usage Notes

The server attempts to resolve the destination address `recipient.` If successful, it returns the recipient's full name and fully qualified mailbox path. The connection to the server must have already been established by means of `OPEN_CONNECTION` and `HELO` or `EHLO` before making this request.

Successful verification returns one or more lines beginning with status code 250 or 251.

WRITE_DATA Procedure

This procedure writes a portion of the e-mail message. A repeat call to `WRITE_DATA` appends data to the e-mail message.

Syntax

```
UTL_SMTP.WRITE_DATA (
   c      IN OUT NOCOPY connection,
   data   IN VARCHAR2 CHARACTER SET ANY_CS);
```

Parameters

Table 302-43 WRITE_DATA Procedure Parameters

Parameter	Description
`c`	SMTP connection
`data`	Portion of the text of the message to be sent, including headers, in [RFC822] format

Usage Notes

- The calls to the OPEN_DATA Function and Procedure, WRITE_DATA Procedure, WRITE_RAW_DATA Procedure and CLOSE_DATA Function and Procedure must be made in the correct order. A program calls `OPEN_DATA` to send the `DATA` command to the SMTP server. After that, it can call `WRITE_DATA` or `WRITE_RAW_DATA` repeatedly to send the actual data. The data is terminated by calling `CLOSE_DATA`. After `OPEN_DATA` is called, the only subprograms that can be called are `WRITE_DATA`, `WRITE_RAW_DATA`, or `CLOSE_DATA`. A call to other subprograms results in an `INVALID_OPERATION` exception being raised.
- The application must ensure that the contents of the body parameter conform to the MIME(RFC822) specification. The `DATA` routine terminates the message with a `<CR><LF>.<CR><LF>` sequence (a single period at the beginning of a line), as required by RFC821. It also translates any sequence of `<CR><LF>.<CR><LF>` (single period) in the body

to <CR><LF>..<CR><LF> (double period). This conversion provides the transparency as described in Section 4.5.2 of RFC821.

- The OPEN_DATA Function and Procedure, WRITE_DATA Procedure, WRITE_RAW_DATA Procedure and CLOSE_DATA Function and Procedure must be called only after OPEN_CONNECTION Functions, HELO Function and Procedure, or EHLO Function and Procedure, MAIL Function and Procedure, and RCPT Function have been called. The connection to the SMTP server must be open and a mail transaction must be active when this routine is called.
- Note that there is no function form of the WRITE_DATA Procedure because the SMTP server does not respond until the data-terminator is sent during the call to CLOSE_DATA Function and Procedure.
- Text (VARCHAR2) data sent using WRITE_DATA is converted to US7ASCII before it is sent. If the text contains multibyte characters, each multibyte character in the text that cannot be converted to US7ASCII is replaced by a '?' character. If 8BITMIME extension is negotiated with the SMTP server using the EHLO subprogram, multibyte VARCHAR2 data can be sent by first converting the text to RAW using the UTL_RAW package, and then sending the RAW data using WRITE_RAW_DATA.

WRITE_RAW_DATA Procedure

This procedure writes a portion of the e-mail message. A repeat call to WRITE_RAW_DATA appends data to the e-mail message.

Syntax

```
UTL_SMTP.WRITE_RAW_DATA (
   c     IN OUT NOCOPY connection
   data  IN RAW);
```

Parameters

Table 302-44   WRITE_RAW_DATA Procedure Parameters

Parameter	Description
c	SMTP connection
data	Portion of the text of the message to be sent, including headers, in [RFC822] format

Usage Notes

- The calls to the OPEN_DATA Function and Procedure, WRITE_DATA Procedure, WRITE_RAW_DATA Procedure and CLOSE_DATA Function and Procedure must be made in the correct order. A program calls OPEN_DATA to send the DATA command to the SMTP server. After that, it can call WRITE_DATA or WRITE_RAW_DATA repeatedly to send the actual data. The data is terminated by calling CLOSE_DATA. After OPEN_DATA is called, the only subprograms that can be called are WRITE_DATA, WRITE_RAW_DATA, or CLOSE_DATA. A call to other subprograms results in an INVALID_OPERATION exception being raised.
- The application must ensure that the contents of the body parameter conform to the MIME(RFC822) specification. The DATA routine terminates the message with a <CR><LF>.<CR><LF> sequence (a single period at the beginning of a line), as required by RFC821. It also translates any sequence of <CR><LF>.<CR><LF> (single period) in the body

to `<CR><LF>..<CR><LF>` (double period). This conversion provides the transparency as described in Section 4.5.2 of RFC821.

- The OPEN_DATA Function and Procedure, WRITE_DATA Procedure, WRITE_RAW_DATA Procedure and CLOSE_DATA Function and Procedure must be called only after OPEN_CONNECTION Functions, HELO Function and Procedure, or EHLO Function and Procedure, MAIL Function and Procedure, and RCPT Function have been called. The connection to the SMTP server must be open and a mail transaction must be active when this routine is called.
- Note that there is no function form of the WRITE_DATA Procedure because the SMTP server does not respond until the data-terminator is sent during the call to CLOSE_DATA Function and Procedure.