35

How to Load XML Data

The main way to load XML data into Oracle XML DB is to use SQL*Loader.

- Overview of Loading XMLType Data Into Oracle Database
  You can load `XMLType` data with SQL*Loader, using either the conventional method or the direct-path method, regardless of how it is stored (object-relational or binary XML storage).
- Load XMLType Data Using SQL*Loader
  SQL*Loader treats `XMLType` columns and tables like object-relational columns and tables. All methods for loading LOB data from the primary datafile or from a `LOBFILE` value apply also to loading `XMLType` columns and tables when the `XMLType` data is stored as a LOB.

Related Topics

- Overview of How To Use Oracle XML DB
  An overview of the various ways of using Oracle XML DB is presented.

Overview of Loading XMLType Data Into Oracle Database

You can load `XMLType` data with SQL*Loader, using either the conventional method or the direct-path method, regardless of how it is stored (object-relational or binary XML storage).

Starting with Oracle9i release 1 (9.0.1), the Export-Import utility and SQL*Loader support `XMLType` as a column type. Starting with Oracle Database 10g, SQL*Loader also supports loading `XMLType` tables.

> **Note:**
>
> For *object-relational storage* of XML data, if the data involves *inheritance* (extension or restriction) of XML Schema types, then SQL*Loader does *not* support direct-path loading.
>
> That is, if an XML schema contains a `complexType` element that extends or restricts another `complexType` element (the base type), then this results in some SQL types being defined in terms of other SQL types. In this case, direct-path loading is not supported for object-relational storage.

Oracle XML DB Repository information is *not* exported when user data is exported. Neither the resources nor any information are exported.

Related Topics

- Export and Import of Oracle XML DB Data
  You can use Oracle Data Pump to export and import `XMLType` tables for use with Oracle XML DB.

> **See Also:**
>
> *Oracle Database Utilities*
>
> - Export and Import of Oracle XML DB Data
> - *Oracle Database Utilities*

Load XMLType Data Using SQL*Loader

SQL*Loader treats `XMLType` columns and tables like object-relational columns and tables. All methods for loading LOB data from the primary datafile or from a `LOBFILE` value apply also to loading `XMLType` columns and tables when the `XMLType` data is stored as a LOB.

> **See Also:**
>
> *Oracle Database Utilities*

> **Note:**
>
> You cannot specify a SQL string for LOB fields. This is true even if you specify `LOBFILE_spec`.

`XMLType` data can be present in a control file or in a LOB file. In the former case, the LOB file name is present in the control file.

Because `XMLType` data can be quite large, SQL*Loader can load LOB data from either a primary datafile (in line with the rest of the data) or from LOB files, independent of how the data is stored (the underlying storage can, for example, still be object-relational).

- Load XMLType LOB Data Using SQL*Loader
  To load internal LOBs, Binary Large Objects (BLOBs), Character Large Objects (CLOBs), and National Character Large Object (NCLOBs), or `XMLType` columns and tables from a primary datafile, use standard SQL*Loader formats.
- Load XMLType Data Directly from a Control File Using SQL*Loader
  You can load `XMLType` data directly from a control file. SQL*Loader treats `XMLType` data like any scalar type.
- Loading Large XML Documents Using SQL*Loader
  You can use SQL*Loader to load large amounts of XML data into Oracle Database.

Load XMLType LOB Data Using SQL*Loader

To load internal LOBs, Binary Large Objects (BLOBs), Character Large Objects (CLOBs), and National Character Large Object (NCLOBs), or `XMLType` columns and tables from a primary datafile, use standard SQL*Loader formats.

- Predetermined size fields

- Delimited fields
- Length-value pair fields

These formats are described in the following sections and in more detail in *Oracle Database Utilities*

- Load LOB Data Using Predetermined Size Fields
  Predetermined size fields constitute a very fast and conceptually simple SQL*Loader format for loading LOBs.
- Load LOB Data Using Delimited Fields
  The delimited fields format handles LOBs of different sizes within the same column (datafile field). However, this added flexibility can affect performance, because SQL*Loader must scan through the data, looking for the delimiter string.
- Load XML Columns Containing LOB Data from LOBFILEs
  LOB data can be lengthy enough that it makes sense to load it from a LOBFILE instead of from a primary datafile.
- Specify LOBFILEs
  You can specify LOBFILEs either statically (you specify the name of the file) or dynamically (you use a FILLER field as the source of the filename).

Load LOB Data Using Predetermined Size Fields

Predetermined size fields constitute a very fast and conceptually simple SQL*Loader format for loading LOBs.

> **Note:**
>
> Because the LOBs you are loading might not be of equal size, you can use whitespace to pad the LOB data to make the LOBs all of equal length within a particular data field.

Load LOB Data Using Delimited Fields

The delimited fields format handles LOBs of different sizes within the same column (datafile field). However, this added flexibility can affect performance, because SQL*Loader must scan through the data, looking for the delimiter string.

As with single-character delimiters, when you specify string delimiters, you should consider the character set of the datafile. When the character set of the datafile is different than that of the control file, you can specify the delimiters in hexadecimal (that is, hexadecimal string). If the delimiters are specified in hexadecimal notation, then the specification must consist of characters that are valid in the character set of the input datafile. In contrast, if hexadecimal specification is not used, then the delimiter specification is considered to be in the client (that is, the control file) character set. In this case, the delimiter is converted into the datafile character set before SQL*Loader searches for the delimiter in the datafile.

Load XML Columns Containing LOB Data from LOBFILEs

LOB data can be lengthy enough that it makes sense to load it from a LOBFILE instead of from a primary datafile.

In LOBFILEs, LOB data instances are still considered to be in fields (predetermined size, delimited, length-value), but these fields are not organized into records (the concept of a record does not exist within LOBFILEs). Therefore, the processing overhead of dealing with records is avoided. This type of organization of data is ideal for LOB loading.

There is no requirement that a LOB from a LOBFILE fit in memory. SQL*Loader reads LOBFILEs in 64 KB chunks.

In LOBFILEs the data can be in any of the following types of fields, any of which can be used to load XML columns:

- A single LOB field into which the entire contents of a file can be read
- Predetermined size fields (fixed-length fields)
- Delimited fields (that is, `TERMINATED BY` or `ENCLOSED BY`)

  The clause `PRESERVE BLANKS` is not applicable to fields read from a LOBFILE.
- Length-value pair fields (variable-length fields) .

  To load data from this type of field, use the `VARRAY`, `VARCHAR`, or `VARCHAR2` SQL*Loader data types.

Specify LOBFILEs

You can specify LOBFILEs either statically (you specify the name of the file) or dynamically (you use a FILLER field as the source of the filename).

In either case, when the EOF of a LOBFILE is reached, the file is closed and additional attempts to read data from that file produce results equivalent to reading data from an empty field.

You should not specify the same LOBFILE as the source of two different fields. If you do so, then typically, the two fields read the data independently.

Load XMLType Data Directly from a Control File Using SQL*Loader

You can load `XMLType` data directly from a control file. SQL*Loader treats `XMLType` data like any scalar type.

For example, consider a table containing a `NUMBER` column followed by an `XMLType` column that is stored object-relationally. The control file used for this table can contain the value of the `NUMBER` column followed by the value of the `XMLType` instance.

SQL*Loader accommodates `XMLType` instances that are very large. You also have the option to load such data from a LOB file.

Loading Large XML Documents Using SQL*Loader

You can use SQL*Loader to load large amounts of XML data into Oracle Database.

1. List in a data file, say `filelist.dat`, the locations of the XML documents to be loaded.
2. Create a control file, say `load_data.ctl`, with commands that process the files listed in the data file.
3. Invoke the SQL*Loader shell command, `sqlldr`, passing it the name of the control file.

This is illustrated in Example 35-1, Example 35-2, and Example 35-3. File `filelist.dat` lists XML files that contain purchase orders for the year 2002.

If your application uses indexes or constraints then processing of these can impact loading performance. You can temporarily disable this processing using PL/SQL subprograms `disableIndexesAndConstraints` and `enableIndexesAndConstraints` in package `DBMS_XMLSTORAGE_MANAGE`.

> **See Also:**
>
> - *Oracle Database Utilities* for information about shell command `sqlldr`
> - *Oracle Database PL/SQL Packages and Types Reference* for information about `DBMS_XMLSTORAGE_MANAGE` subprograms `disableIndexesAndConstraints` and `enableIndexesAndConstraints`

Example 35-1 Data File filelist.dat: List of XML Files to Load

```
2002/Jan/AMCEWEN-20021009123335370PDT.xm
2002/Jan/AWALSH-2002100912333570PDT.xml
2002/Jan/CJOHNSON-20021009123335170PDT.xml
2002/Jan/LSMITH-20021009123335500PDT.xml
2002/Jan/PTUCKER-20021009123335430PDT.xml
2002/Jan/SBELL-2002100912333280PDT.xml
2002/Jan/SKING-2002100912333560PDT.xml
2002/Jan/SMCCAIN-20021009123335470PDT.xml
2002/Jan/TFOX-2002100912333520PDT.xml
2002/Jan/VJONES-2002100912333350PDT.xml
2002/Jan/WSMITH-2002100912333450PDT.xml
2002/Feb/AMCEWEN-20021009123335600PDT.xml
2002/Feb/AMCEWEN-20021009123335701PDT.xml
2002/Feb/DAUSTIN-20021009123335811PDT.xml
2002/Feb/EABEL-2002100912333791PDT.xml
2002/Feb/PTUCKER-20021009123335721PDT.xml
2002/Feb/PTUCKER-20021009123335821PDT.xml
2002/Feb/SBELL-2002100912333771PDT.xml
2002/Feb/SMCCAIN-20021009123335681PDT.xml
2002/Feb/WSMITH-20021009123335650PDT.xml
2002/Feb/WSMITH-20021009123335741PDT.xml
2002/Feb/WSMITH-20021009123335751PDT.xml
...
```

Example 35-2 Control File load_datra.ctl, for Loading Purchase-Order XML Documents

```
load data
infile 'filelist.dat'
append
into table PURCHASEORDER
xmltype(XMLDATA)
(
 filename filler char(120),
 XMLDATA  lobfile(filename) terminated by eof
)
```

Example 35-3 Loading XML Data Using Shell Command sqlldr

```
sqlldr load_data.ctl
```

For direct-path loading, use this instead:

```
sqlldr load_data.ctl direct=y
```

Related Topics

- Overview of How To Use Oracle XML DB
  An overview of the various ways of using Oracle XML DB is presented.
- Loading Large XML Files Using SQL*Loader
  You can use SQL*Loader to load large amounts of XML data into Oracle Database.