SQL*Loader

Introduction to SQL*Loader

SQL*Loader is an Oracle utility used to load data from external files into Oracle database tables. It is widely used for bulk data loading due to its speed and flexibility.

Why Use SQL*Loader?

Handles large volumes of data efficiently.

Allows flexible data transformations during loading.

Supports various file formats and complex loading scenarios.

Offers logging and error-handling mechanisms.

SQL*Loader Components

Control File (.ctl)

A control file contains metadata that guides SQL*Loader on how to interpret and load data.

Specifies the input data file(s).

Defines table and column mappings.

Controls error handling and logging.

Supports conditional loading and data transformations.

Data File (.csv, .dat, .txt)

Contains the actual data to be loaded.

Can be delimited (comma, tab, pipe) or fixed-width format.

Log File (.log)

Records detailed execution logs, including successfully loaded rows and errors.

Bad File (.bad)

Stores records that fail to load due to data format or constraint violations.

Discard File (.dsc)

Contains records rejected based on filtering criteria in the control file.

Control File Syntax Breakdown

Header Section

Defines the input data file and table to load data into.

Field Definitions

Specifies how columns in the data file map to table columns.

Supports data type conversions and transformations.

Handling NULLs

TRAILING NULLCOLS allows missing column values to be treated as NULL.

Error Handling

ERRORS option specifies the maximum number of errors before stopping.

SKIP option allows skipping initial records.

DISCARDFILE option enables conditional record rejection.

SQL*Loader Parameters

USERID - Specifies Oracle database credentials.

CONTROL - Specifies the control file location.

DATA - Specifies the data file location.

LOG - Specifies the log file location.

BAD - Specifies the bad file location.

DISCARD - Specifies the discard file location.

SKIP - Skips the specified number of records.

DIRECT - Enables direct path loading.

PARALLEL - Enables parallel execution for faster loads.

Performance Optimization

Use direct path load for large datasets.

Disable indexes and constraints before loading and re-enable after.

Use parallel processing for improved performance.

Increase buffer size with the READSIZE parameter.

Error Handling and Debugging

Check .log, .bad, and .dsc files for error details.

Use SQLLDR exit codes to handle errors in scripts.

Validate data format before loading.

Use Cases for SQL*Loader

Migrating large datasets between environments.

Loading data from external systems.

Automating batch data processing.

Alternatives to SQL*Loader

Oracle Data Pump - More advanced, supports direct exports and imports.

External Tables - Allows querying external files directly.

ETL Tools (Informatica, Talend, etc.) - GUI-based, more suited for complex transformations.