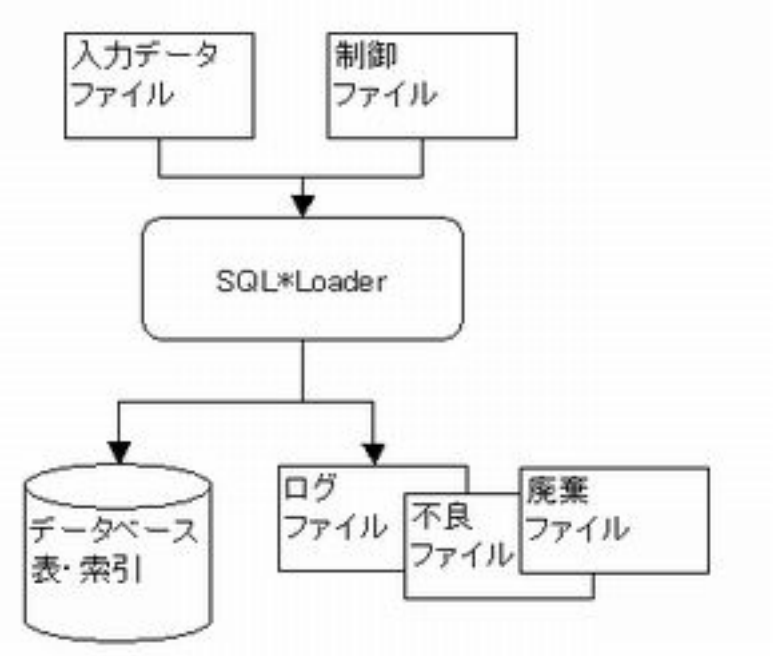
Oracle Database10g SQL*Loader

[Oracle][Oracle Database10g]

概要

・SQL*Loader を使用して、外部ファイルのデータを Oracle データベースの表にロードできる
・一般的な SQL*Loader セッションでは、SQL*Loader の動作を制御する制御ファイルと 1 つ以上のデータ・ファイルが入力用に使用される
・SQL*Loader の出力先は、データがロードされる Oracle データベース、ログ・ファイル、不良ファイルで、廃棄ファイルに出力される場合もある



SQL*Loader のパラメータ

・SQL*Loader は、sqlldr コマンドを指定すると起動します。また、オプションで、セッション特性を確立するパラメータを指定した場合も起動します。
・常に、値がほとんど変らない同じパラメータを使用する場合は、コマンドラインではなく、次の方法でパラメータを指定すると効率的
  ・パラメータ・ファイルとしてグループ化。その後、PARFILE パラメータを使用して、そのパラメータ・ファイルの名前をコマンドラインで指定。
  ・一部のパラメータは、OPTIONS 句を使用して、制御ファイル内に指定。

データのロード方法

以下の 2 つの方法

方法	内容
従来型パス・ロード	データベースの表に対して（1 つ以上の）SQL INSERT 文が実行されます
ダイレクト・パス・ロード	データ・ブロックをフォーマットし、データ・ブロックを直接データ・ファイルに書き込むため、オーバーヘッドが大幅に削減

■従来型パス・ロード

・従来型パス・ロード（デフォルト）では、SQL INSERT 文とバインド配列バッファを使用して、データをデータベース表にロード
・バッファ・リソースに関して他のすべてのプロセスと同等の処理が行われるため、競合が発生
・SQL 文が生成され、 Oracle に渡されてから実行されるため、オーバーヘッドが発生
・挿入が発生すると、常に、Oracle データベースで空き領域のあるブロック（ディスク内に散在して、部分的に書込み可能なブロック）が検索され、そこにデータが書き込まれる
・大量データのロード速度を大幅に低下させることがあります。

■ダイレクト・パス・ロード

・バインド配列バッファに書き込むかわりに、SQL INSERT 文を使用して、バインド配列を Oracle データベースに渡す。
・ダイレクト・パス・ロードは、ダイレクト・パス API を使用して、ロードされるデータをサーバーのロード・エンジンに渡す。
・ロード・エンジンは、渡されたデータから列配列構造体を作成
・ロード・エンジンは、列配列構造体を使用して Oracle データ・ブロックをフォーマットし、索引キーを作成します。新しくフォーマットされたデータベース・ブロックを直接データベースに書き込む
・I/O を伴う処理がオーバーラップするため、ロード・パフォーマンスが向上。

ダイレクト・パス・ロードの指定

・SQL*Loader をダイレクト・パス・ロード・モードで起動するには、次の形式で、コマンドラインまたはパラメータ・ファイル（使用している場合）の DIRECT パラメータに true を設定します。

```
DIRECT=true
```

SQL*Loader 制御ファイル

・制御ファイルは、SQL*Loader が解釈できる言語で記述されたテキスト・ファイル

■ 3 つのセクション

セクション	内容
第 1 セクション	セッション全体の情報
第 2 セクション	1 つ以上の INTO TABLE ブロックで構成、それぞれのブロックには、表名、その表の列などの、データがロードされる表についての情報
第 3 セクション	オプションで、このセクションがある場合は、入力データを記述

■制御ファイルの内容

・SQL*Loader の DDL 構文図

■コメント

・コメントはファイル中のコマンド部分のどこにでも記述できますが、データの部分には記述できません。

```
-- This is a comment.
```

■データ・ファイルの指定

・ロードするデータを含むデータ・ファイルを指定するには、INFILE キーワードにファイル名を続け、必要な場合はファイル処理オプション文字列を続ける
・ァイル名が指定されない場合は、デフォルトで制御ファイル名の拡張子を .dat にしたものが採用
・ードするデータを制御ファイル内にも記述した場合は、ファイル名にアスタリスク（*）を指定

制御ファイルにデータがある場合

```
INFILE *
```

デフォルトの拡張子 .dat を持つファイル sample にデータがある場合

```
INFILE sample
```

フルパスに指定されたファイル datafile.dat にデータがある場合

```
INFILE 'c:/topdir/subdir/datafile.dat'
```

■フィールド・リストの内容

位置指定

・POSITION に、データ・フィールドの位置を指定します

```
( { start | * [+integer] } [{ : | - } end] )
```

パラメータ	内容
start	開始位置です。論理レコードの先頭バイト位置は 1
end	終了位置。start-end と表記することも、start:end と表記することもできます。
*	対象となるデータ・フィールドが前のフィールドの直後にあることを示す
integer	オフセットを使用。前フィールドの終了位置直後の位置から現行のフィールドをオフセット

データ型の指定

TERMINATED FIELD

・フィールドの開始位置から最初のデリミタ文字までのデータが読み込まれます
・TERMINATED BY WHITESPACE を指定すると、最初に空白文字（スペース、タブ、空白、LF、改ページまたは改行）が現れるまでデータが読み込まれます

```
TERMINATED [BY] { WHITESPACE | X'hexstr' | 'string' | EOF }
```

ENCLOSED フィールド

・空白以外の文字が検出されるまで、空白文字はスキップされます

```
ENCLOSED [BY] [ 'string' | X'hexstr' ] [AND] [ 'string' | X'hexstr' ]
```

```
TERMINATED BY ','                       a data string,
ENCLOSED BY '"'                         "a data string"
TERMINATED BY ',' ENCLOSED BY '"'       "a data string",
ENCLOSED BY '(' AND ')'                 (a data string)
```

入力データおよびデータ・ファイル

・制御ファイルに指定された 1 つ以上のファイルなどから、SQL*Loader にデータが読み込まれます
・レコード形式は、INFILE パラメータを使用して制御ファイルに指定することができる。デフォルトはストリーム・レコード形式

データファイル形式

・固定レコード形式
・可変レコード形式
・ストリーム・レコード形式

■固定レコード形式

```
INFILE datafile_name "fix n"
```

■可変レコード形式

```
INFILE "datafile_name" "var n"
```

■ストリーム・レコード形式

```
INFILE datafile_name ["str terminator_string"]
```

SQL*Loader の事例

・Oracle Database のインストール時に、$ORACLE_HOME/rdbms/demo ディレクトリに事例ファイルがインストールされる

通常、各事例は次の種類のファイルで構成

・制御ファイル（ulcase5.ctl など）
・データ・ファイル（ulcase5.dat など）

・セットアップ・ファイル（ulcase5.sql など）

ストリームレコード形式のロード

■テーブルの作成

・適当にテーブルを作成する

```
SQL> show user
ユーザーは "EXAM" です。
SQL> create table apache_access_log (
  2    ip_address     char(20)     ,
  3    user_inf       char(20)     ,
  4    user_id        char(20)     ,
  5    req_date       char(60)     ,
  6    request        varchar2(512),
  7    status_cd      char(10)     ,
  8    res_size       char(10)     ,
  9    referer        varchar2(512),
 10    user_agent     varchar2(512)
 11  )
 12 /

表が作成されました。
```

■制御ファイルの作成

・apache_access_log.ctl

```
-- appache access logfile load sample control file
load data                           -- 新しくデータロードが開始される
infile  'access_log'                -- ロードするデータが入っているファイル名
badfile 'access_log.bad'            -- 拒否レコードが書き込まれるファイル名
discardfile 'access_log.dsc'        -- 廃棄レコードが書き込まれるファイル名
 insert                                        --  すでにデータが存在する場合のオプション
APPEND,REPLACE,TRUNCATE
 into table exam.apache_access_log    --
 (  ip_address   position(*) char terminated by whitespace
  , user_inf     position(*) char terminated by whitespace
  , user_id      position(*) char terminated by whitespace
  , req_date     position(*) char enclosed by '[' and ']'
  , request      position(*) char enclosed by '"' and '"'
  , status_cd    position(*) char terminated by whitespace
  , res_size     position(*) char terminated by whitespace
  , referer      position(*) char enclosed by '"' and '"'
  , user_agent   position(*) char enclosed by '"' and '"'
 )
```

■実行

・apache のアクセスログを、access_log.dat に変更して、以下を実行

```
$sqlldr exam/abc123 apache_access_log.ctl

SQL*Loader: Release 10.2.0.1.0 - Production on Sat Jul 11 22:05:04 2009

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Commit point reached - logical record count 64
Commit point reached - logical record count 128
               :
```

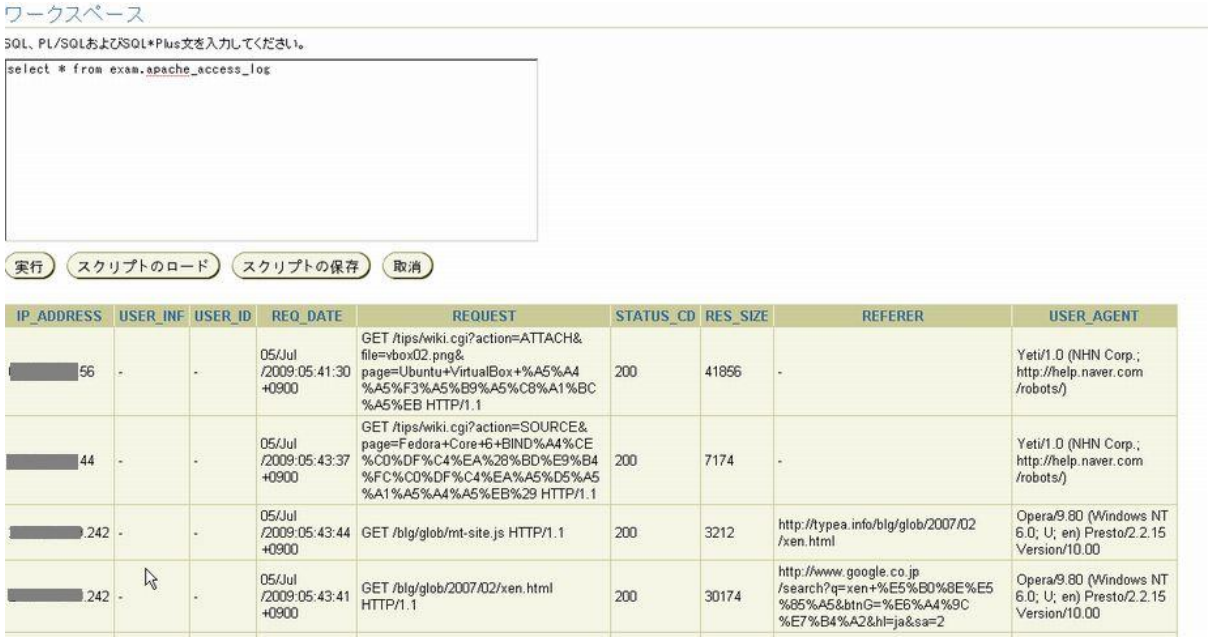
・ダイレクト・パス・ロードで実行の例

```
$ sqlldr exam/abc123 apache_access_log.ctl  direct=true
```

```
SQL*Loader: Release 10.2.0.1.0 - Production on Sun Jul 12 00:18:28 2009
Copyright (c) 1982, 2005, Oracle.  All rights reserved.
Load completed - logical record count 5872.
```

■取り込まれた

```
select * from exam.apache_access_log;
```



IP_ADDRESS	USER_INF	USER_ID	REQ_DATE	REQUEST	STATUS_CD	RES_SIZE	REFERER	USER_AGENT
[redacted]56	-	-	05/Jul /2009:05:41:30 +0900	GET /tips/wiki.cgi?action=ATTACH&file=vbox02.png&page=Ubuntu+VirtualBox+%A5%A4%A5%F3%A5%B9%A5%C8%A1%BC%A5%EB HTTP/1.1	200	41856	-	Yeti/1.0 (NHN Corp.; http://help.naver.com/robots/)
[redacted]44	-	-	05/Jul /2009:05:43:37 +0900	GET /tips/wiki.cgi?action=SOURCE&page=Fedora+Core+6+BIND%A4%CE%C0%DF%C4%EA%28%BD%E9%B4%FC%C0%DF%C4%EA%A5%D5%A5%A1%A5%A4%A5%EB%29 HTTP/1.1	200	7174	-	Yeti/1.0 (NHN Corp.; http://help.naver.com/robots/)
[redacted].242	-	-	05/Jul /2009:05:43:44 +0900	GET /blg/glob/mt-site.js HTTP/1.1	200	3212	http://typea.info/blg/glob/2007/02/xen.html	Opera/9.80 (Windows NT 6.0; U; en) Presto/2.2.15 Version/10.00
[redacted].242	-	-	05/Jul /2009:05:43:41 +0900	GET /blg/glob/2007/02/xen.html HTTP/1.1	200	30174	http://www.google.co.jp/search?q=xen+%E5%B0%8E%E5%85%A5&btnG=%E6%A4%9C%E7%B4%A2&hl=ja&sa=2	Opera/9.80 (Windows NT 6.0; U; en) Presto/2.2.15 Version/10.00

ユーザーエージェントを調べてみる

```
select user_agent,count(user_agent) from exam.apache_access_log
group by user_agent
order by 2 desc
```

USER_AGENT	COUNT(USER_AGENT)
Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)	3191
Mozilla/5.0 (compatible; Yahoo! Slurp/3.0; http://help.yahoo.com/help/us/ysearch/slurp)	2263
Yeti/1.0 (NHN Corp.; http://help.naver.com/robots/)	2186
DoCoMo/2.0 N905i(c100;TB;W24H16) (compatible; Googlebot-Mobile/2.1; +http://www.google.com/bot.html)	1470
Mozilla/5.0 (Windows; U; Windows NT 6.0; ja; rv:1.9.0.11) Gecko/2009060215 Firefox/3.0.11 (.NET CLR 3.5.30729)	1395
Mozilla/5.0 (Windows; U; Windows NT 5.1; ja; rv:1.9.0.11) Gecko/2009060215 Firefox/3.0.11	955
WordPress/2.7	917
Mozilla/5.0 (Windows; U; Windows NT 5.1; ja; rv:1.9.0.11) Gecko/2009060215 Firefox/3.0.11 (.NET CLR 3.5.30729)	858
Mediapartners-Google	850
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)	765
Baiduspider+(+http://www.baidu.jp/spider/)	665
Mozilla/5.0 (compatible; DotBot/1.1; http://www.dotnetdotcom.org/, crawler@dotnetdotcom.org)	647
Mozilla/5.0 (Twiceler-0.9 http://www.cuil.com/twiceler/robot.html)	538
Mozilla/4.0 (compatible;)	505
Mozilla/5.0 (Windows; U; Windows NT 5.1; ja; rv:1.9.0.10) Gecko/2009042316 Firefox/3.0.10	469
Mozilla/5.0 (Windows; U; Windows NT 5.1; ja; rv:1.9.0.10) Gecko/2009042316 Firefox/3.0.10 (.NET CLR 3.5.30729)	441