Information Management

IBM

Educate. Inform. Entertain.
Growing the DB2 Community Since 2009
www.DB2NightShow.com

The DB2Night Show Episode #88

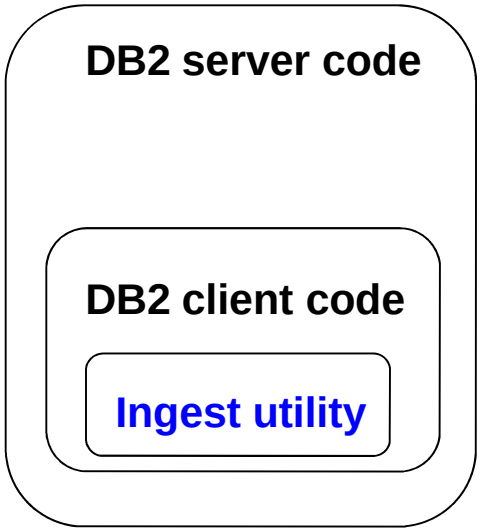
Real Time Warehousing with IBM InfoSphere Warehouse V10

Pat Bates, WW Technical Sales for Big Data and Warehousing, jpbates@us.ibm.com

June 26, 2012

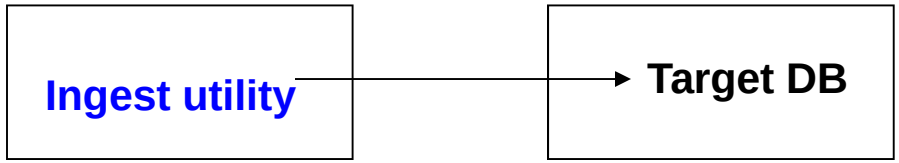
Deployment options

Client-side utility



Topology options

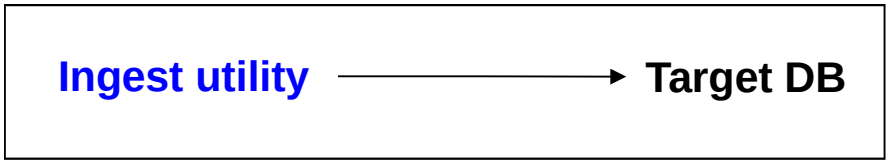
1. Remote client



server hosting DB2 client

servers hosting DB2 DPF

2. Local client



server hosting DB2 client and DB2 DPF

- **Supported environments**
  - Serial
  - DPF
  - pureScale

Input types and formats

- **Input types: file or named pipe (can also specify multiple files or multiple pipes)**
- **Input formats**
  - DELIMITED
    - Equivalent to "OF DEL" on the IMPORT and LOAD commands
    - Fields are separated by a one-byte character (default is comma)
    - Records are always varying length (delimited by CRLF or LF)
    - Field definition list is optional and defaults to specified or implied table columns
  - POSITIONAL
    - Equivalent to "OF ASC" on the IMPORT and LOAD commands
    - Fields are at fixed offsets in each record and have a fixed length
    - Records can be varying length (delimited by CRLF or LF) or fixed length
    - Field definition list is required
    - Allows numbers to be specified in binary
- **INPUT CODEPAGE parameter allows specifying codepage of input data**

Input types and formats -- examples

```
-- Input records are sent over a named pipe.
INGEST FROM PIPE my_pipe FORMAT DELIMITED
   INSERT INTO my_table;

-- Input records are delimited by CRLF and fields are delimited by a
  vertical bar.
INGEST FROM FILE my_file.del FORMAT DELIMITED '|'
   INSERT INTO my_table;

-- Input records are delimited by CRLF with fields in fixed positions.
INGEST FROM FILE my_file.del FORMAT POSITIONAL
   (
      $field1 POSITION(1:12) INTEGER EXTERNAL,
      $field2 POSITION(13:20) CHAR(8)
   )
   INSERT INTO my_table;

-- Input records are fixed length (no CRLF) with fields in fixed
  positions.
INGEST FROM FILE my_file.del FORMAT POSITIONAL RECORDLEN 20
   (
      $field1 POSITION(1:12) INTEGER EXTERNAL,
      $field2 POSITION(13:20) CHAR(8)
   )
   INSERT INTO my_table;
```

Field definition lists and SQL expressions

```
-- Compute column TOTAL_PRICE from two input fields.
INGEST FROM FILE my_file FORMAT DELIMITED
  (
      $prod_ID INTEGER EXTERNAL,
      $prod_name CHAR(8),
      $unused_field CHAR(1),
      $base_price DECIMAL(5,2) EXTERNAL,
      $shipping_cost DECIMAL(5,2) EXTERNAL
   )
   INSERT INTO my_table(prod_ID, prod_name, total_price)
      VALUES($prod_ID, $prod_name, $base_price + $shipping_cost);

-- Input records have three 2-character fields for year, month, and day.
-- Ingest them into a DATE column.
INGEST FROM FILE my_file FORMAT DELIMITED
  (
      $month CHAR(2),
      $day CHAR(2),
      $year CHAR(2)
   )
   INSERT INTO my_table(date_column)
      VALUES( DATE('20' || $year || '-' || $month || '-' || $day) );
```

Field options

- **Numbers can be specified in various formats**
  - ASCII/binary (all numeric types)
  - big/little endian (all numeric types except decimal)
  - packed/zoned (decimal only)
  - radix point other than dot (all numeric types)
- **Strings can be specified**
  - with or without delimiters (equivalent to chardelx in import and load)
  - With trimming of blanks on the left, right, both, or neither (equivalent to striptblanks or keepblanks in import and load)
- **Security labels can be specified in binary, name, or string format**
- **A character to represent the default value can be specified.**

Field options -- example

```
-- Examples of field options.
-- Field $prod_ID is in binary big endian format.
-- Field $price is in binary packed decimal format.
-- Field $desc is enclosed by exclamation marks and the utility is to
-- trim blanks from the right.
-- If the $location field starts with 'D', set the corresponding
-- column to its default value.
INGEST FROM FILE my_file FORMAT POSITIONAL
  (
      $prod_ID  POSITION(1:8)   INTEGER BIG ENDIAN,
      $price    POSITION(9:11)  DECIMAL(5,2) PACKED,
      $desc     POSITION(12:21) CHAR(10) RTRIM OPTIONALLY ENCLOSED BY
  '!',
      $location POSITION(22:31) CHAR(10) DEFAULTIF 'D',
      $seclabel POSITION(32)    DB2SECURITYLABEL STRING
   )
   INSERT INTO my_table
      VALUES($prod_ID, $price, $desc, $location, $seclabel);
```

SQL statements

- **The INGEST command can specify a subset of the INSERT, UPDATE, DELETE, or MERGE statements.**

```
-- Update records in the table.
INGEST FROM FILE my_file.del FORMAT DELIMITED
   (
      $key_fld1 INTEGER EXTERNAL,
      $key_fld2 INTEGER EXTERNAL,
      $data_fld1 CHAR(8),
      $data_fld2 CHAR(8),
      $data_fld3 CHAR(8)
   )
   UPDATE my_table SET (data_col1, data_col2, data_col3) =
      ($data_fld1, $data_fld2, $data_fld3)
   WHERE (key_col1 = $key_fld1) AND (key_col2 = $key_fld2);
```

SQL Statement Examples

```
-- Merge input records into the table.
INGEST FROM FILE my_file.del FORMAT DELIMITED
   (
      … same field definition list as before …
   )
   MERGE INTO my_table
   ON (key_col1 = $key_fld1) AND (key_col2 = $key_fld2)
   WHEN NOT MATCHED THEN
      INSERT VALUES($key_fld1, $key_fld1, $data_fld1, $data_fld2,
  $data_fld3)
   WHEN MATCHED THEN
      UPDATE SET (data_col1, data_col2, data_col3) =
         ($data_fld1, $data_fld2, $data_fld3);

-- Delete records from the table that correspond to input records.
INGEST FROM FILE my_file.del FORMAT DELIMITED
   (
      … same field definition list as before …
   )
   DELETE FROM my_table
   WHERE (key_col1 = $key_fld1) AND (key_col2 = $key_fld2);
```

Restart -- Example

```
INGEST FROM FILE my_file.del FORMAT DELIMITED
   RESTART NEW 'My ingest job' -- or omit and use default job ID
   INSERT INTO my_table;
```


```
-- Restart the failed job from the last commit point.
-- (This also cleans up the restart data for this job in the
-- restart table.)
INGEST FROM FILE my_file.del FORMAT DELIMITED
   RESTART CONTINUE 'My ingest job'
   INSERT INTO my_table;
-- ***** OR *****
-- We don't want to restart -- clean up the restart data.
INGEST FROM FILE my_file.del FORMAT DELIMITED
   RESTART TERMINATE 'My ingest job'
   INSERT INTO my_table;
```

Comparison to IMPORT and LOAD – Table types

Table type	Import	Load	Ingest
Created global temporary table	no	no	no
Declared global temporary table	no	no	no
Detached table	no	no	no
Multidimensional clustering (MDC) table		yes	yes
Materialized query table (MQT) that is maintained by user	yes	yes	yes
Nickname	relational except ODBC	no (SQL02305N)	yes
Range-clustered table (RCT)	yes	no	yes
Range-partitioned table	yes	yes	yes
Summary table	no	yes	yes
Typed table	yes		no
Typed view	yes		no
Untyped (regular) table	yes	yes	yes
Updatable view	yes	no (SQL02305N)	yes

Comparison to IMPORT and LOAD – Column types

Column data type	Import	Load	Ingest
Numeric: SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, DECFLOAT	yes	yes	yes
Character: CHAR, VARCHAR, NCHAR, NVARCHAR, plus corresponding FOR BIT DATA types	yes	yes	yes
Graphic: GRAPHIC, VARGRAPHIC	yes	yes	yes
Long types: LONG VARCHAR, LONG VARGRAPHIC	yes	yes	yes
Date/time: DATE, TIME, TIMESTAMP(p)	yes	yes	yes
DB2SECURITYLABEL	yes	yes	yes
LOBs from files: BLOB, CLOB, DBCLOB, NCLOB	yes	yes	no
inline LOBs	yes	yes	no
XML from files	yes	yes	no
inline XML	no	no	no
distinct type (note 1)	yes	yes	yes
structured type	no	no	no
reference type	yes	yes	yes

Notes:

1. **Supported if based on a supported built-in type.**

Comparison to IMPORT and LOAD

Input types and formats

Input type	Import	Load	INGEST
cursor	no	yes	no
device	no	yes	no
file	yes	yes	yes
pipe	no	yes	yes
multiple input files, multiple pipes, etc	no	yes	yes

Input format	Import	Load	INGEST
ASC (including binary)	yes, except binary	yes	yes
DEL	yes	yes	yes
IXF	yes	yes	no
WSF (worksheet format)	yes, but will be discontinued in V10.1	no	no

When to use INGEST rather than LOAD

- **Use INGEST when any of the following is true**
  - You need other applications to update the table while it is being loaded
  - The input file contains fields you want to skip over
  - You need to specify an SQL statement other than INSERT
  - You need to specify an SQL expression (to construct a column value from field values)
  - You need to recover and continue on when the utility gets a recoverable error

When to use LOAD rather than INGEST

- **Use LOAD when any of the following is true**
  - You don't need other applications to update the table while it is being loaded
  - You need to load a table that contains XML or LOB columns
  - You need to load from cursor or load from a device
  - You need to load from a file in IXF format
  - You need to load a GENERATED ALWAYS column or SYSTEM_TIME column with the data specified in the input file

Using Adaptive Compression

- **Default for new tables created under DB2 V10 when compression is enabled**
  - `CREATE TABLE tb1 ….  COMPRESS YES` (adaptive compression)
  - `CREATE TABLE tb1 ….  COMPRESS YES ADAPTIVE` (adaptive compression)
  - `CREATE TABLE tb1 ….  COMPRESS YES STATIC` (classic compression)

- **For existing tables**
  - use the ALTER statement to switch from classic to adaptive compression

```
ALTER TABLE <tablename> COMPRESS YES ADAPTIVE
```

  - The new algorithm takes effect for applicable data
  - Note that existing data will not be “table” compressed until the next scheduled REORG

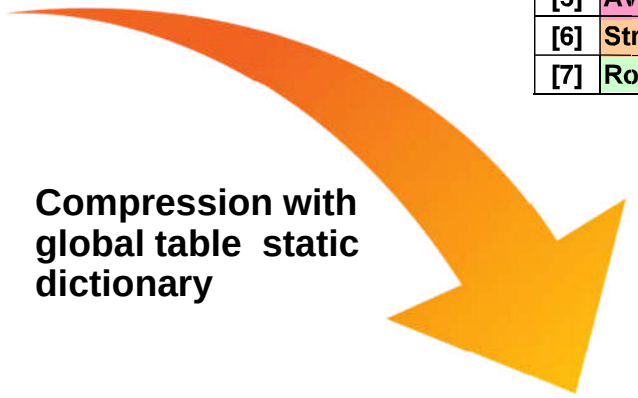
- **Adaptive compression will compress the same data types that are compressed with static compression**

How Does Adaptive Compression Work?

- **Step 1: Compression with static table level dictionary**

Christine	Haas	(408) 463-1234	555 Bailey Avenue	San Jose	California	95141
John	Thompson	(408) 463-5678	555 Bailey Avenue	San Jose	California	95141
Jose	Fernandez	(408) 463-1357	555 Bailey Avenue	San Jose	California	95141
Margaret	Miller	(408) 463-2468	555 Bailey Avenue	San Jose	California	95141
Bruce	Kwan	(408) 956-9876	4400 North 1st Street	San Jose	California	95134
James	Geyer	(408) 956-5432	4400 North 1st Street	San Jose	California	95134
Linda	Hernandez	(408) 956-9753	4400 North 1st Street	San Jose	California	95134
Theodore	Mills	(408) 927-8642	650 Harry Road	San Jose	California	95134
Susan	Stern	(408) 927-9630	650 Harry Road	San Jose	California	95134
James	Polaski	(415) 545-1423	425 Market Street	San Francisco	California	94105
John	Miller	(415) 545-5867	425 Market Street	San Francisco	California	94105
James	Walker	(415) 545-4132	425 Market Street	San Francisco	California	94105
Elizabeth	Brown	(415) 545-8576	425 Market Street	San Francisco	California	94105
Sarah	Johnson	(415) 545-1928	425 Market Street	San Francisco	California	94105

[1]	California 9
[2]	San
[3]	Jose
[4]	Francisco
[5]	Avenue
[6]	Street
[7]	Road



Christine	Haas	(408) 463-1234	555 Bailey [5]	[2][3]	[1]	95141
John	Thompson	(408) 463-5678	555 Bailey [5]	[2][3]	[1]	95141
[3]	Fernandez	(408) 463-1357	555 Bailey [5]	[2][3]	[1]	95141
Margaret	Schneider	(408) 463-2468	555 Bailey [5]	[2][3]	[1]	95141
Bruce	Kwan	(408) 956-9876	4400 North 1st [6]	[2][3]	[1]	95134
James	Geyer	(408) 956-5432	4400 North 1st [6]	[2][3]	[1]	95134
Linda	Hernandez	(408) 956-9753	4400 North 1st [6]	[2][3]	[1]	95134
Theodore	Mills	(408) 927-8642	650 Harry [7]	[2][3]	[1]	95134
Susan	Stern	(408) 927-9630	650 Harry [7]	[2][3]	[1]	95134
James	Polaski	(415) 545-1423	425 Market [6]	[2][4]	[1]	94105
John	Miller	(415) 545-5867	425 Market [6]	[2][4]	[1]	94105
James	Walker	(415) 545-4132	425 Market [6]	[2][4]	[1]	94105
Elizabeth	Miller	(415) 545-8576	425 Market [6]	[2][4]	[1]	94105
Sarah	Johnson	(415) 545-1928	425 Market [6]	[2][4]	[1]	94105

- **Table-level compression symbol dictionary containing globally recurring patterns**
- **Table-level dictionary can only be rebuilt during classic table REORG**
  - Involves re-compressing all data

Data Warehouse Compression Results

230GB raw size - Most of the data in a single table

- **Graph – Storage Savings**
- **Increase in savings by Adaptive Compression**
  - 3x Compression with Static Compression using reorg
  - 5.6x Compression with Automatic dictionary creation and Adaptive Compression
  - 7.4x Compression with Adaptive Compression and full reorg