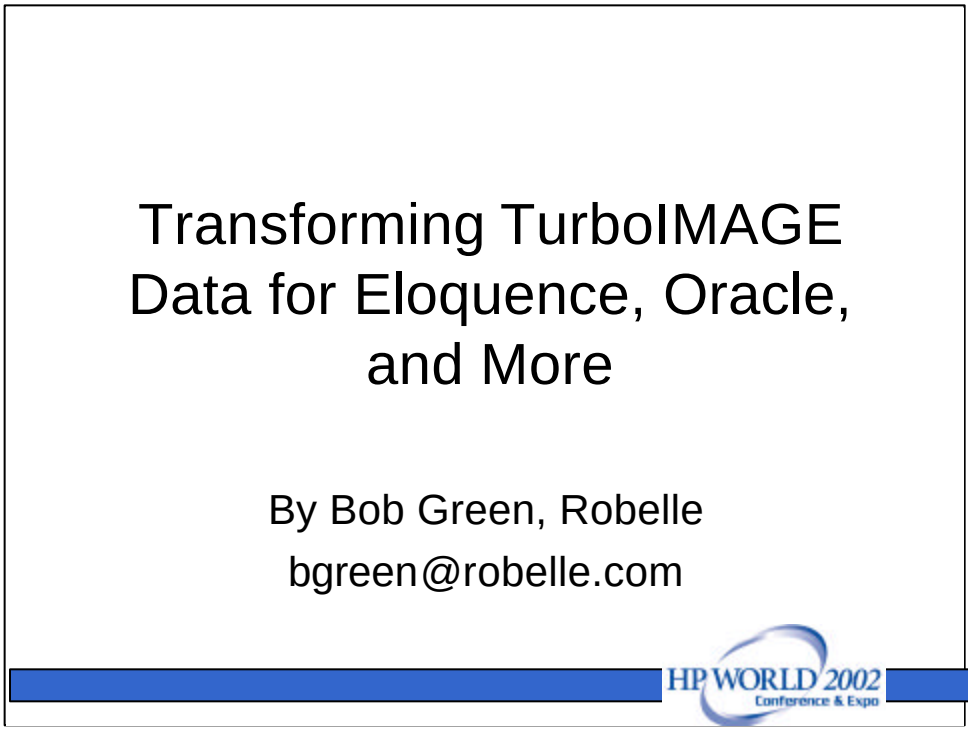


Transforming Your TurboIMAGE Data

Practical tips on how to transform TurboIMAGE data so that it can be used in other databases:

Image to HP Eloquence

Image to Oracle on HP-UX

Image to SQL Server

Image to mySQL

Image to PostgreSQL

Special Cases:

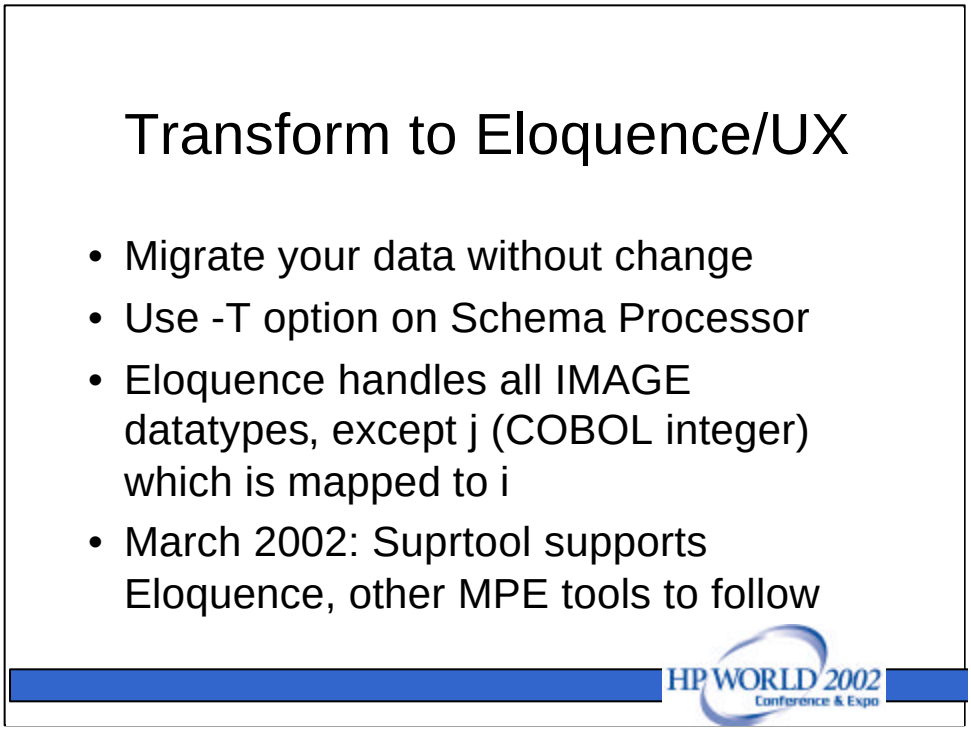
Combining Image Note Fields into Large SQL Text Fields

Presenter:

Bob Green is the founder of Robelle, HP vendor since 1977.

Email: bgreen@robelle.com

Web: www.robelle.com/library/papers/transform.pdf



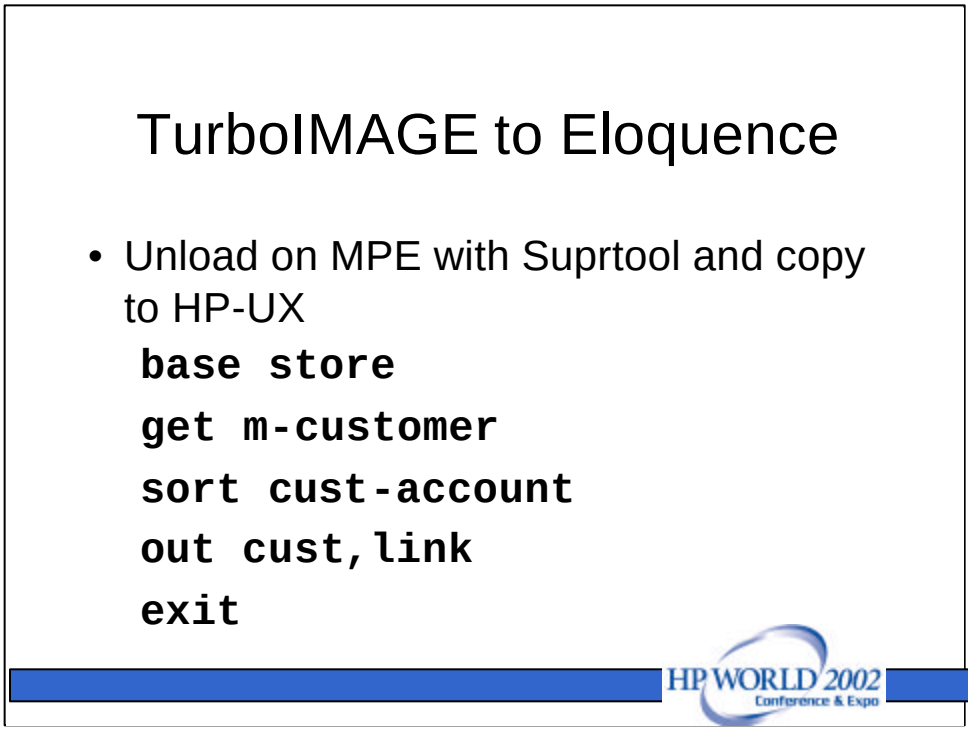
HP Eloquence is an IMAGE-like database that runs on HP-UX, LINUX and Windows NT/2K. It is owned by HP Germany and supported by a 3rd party, Marxmeier:

> HP Eloquence home page: www.hp-eloquence.com/
> Marxmeier Software AG: www.marxmeier.com/
> Email contact at Marxmeier: info@marxmeir.com

Eloquence 7 maps datatype J is mapped to I because J is not supported by the database kernel. It doesn't really matter, as J is only a COBOL documentation thing. You would need to do all database edits with COBOL programs with PIC 9999 without sign to enforce this, since IMAGE doesn't enforce the J edits.

R is mapped to E because HP-UX has only IEEE floating point, not the old floating-point format of the original 1970 HP 3000, but hopefully you are not using that anymore anyway! If you find that you are, then Suprtool can convert from Oldreal to IEEE real.

Robelle converted Suprtool for HP-UX to support Eloquence and completed the project in March 2002. Other vendors have announced plans to support Eloquence and HP has blessed it as a migration option for small to medium-sized databases.



To migrate data without any conversion issues, use Suprtool. This is different from most migrations, because we do **not** need to convert the data to character format. We can transfer in the original native BINARY formats, since MPE and HP-UX use exactly the same data formats and you can have Suprtool on both servers.

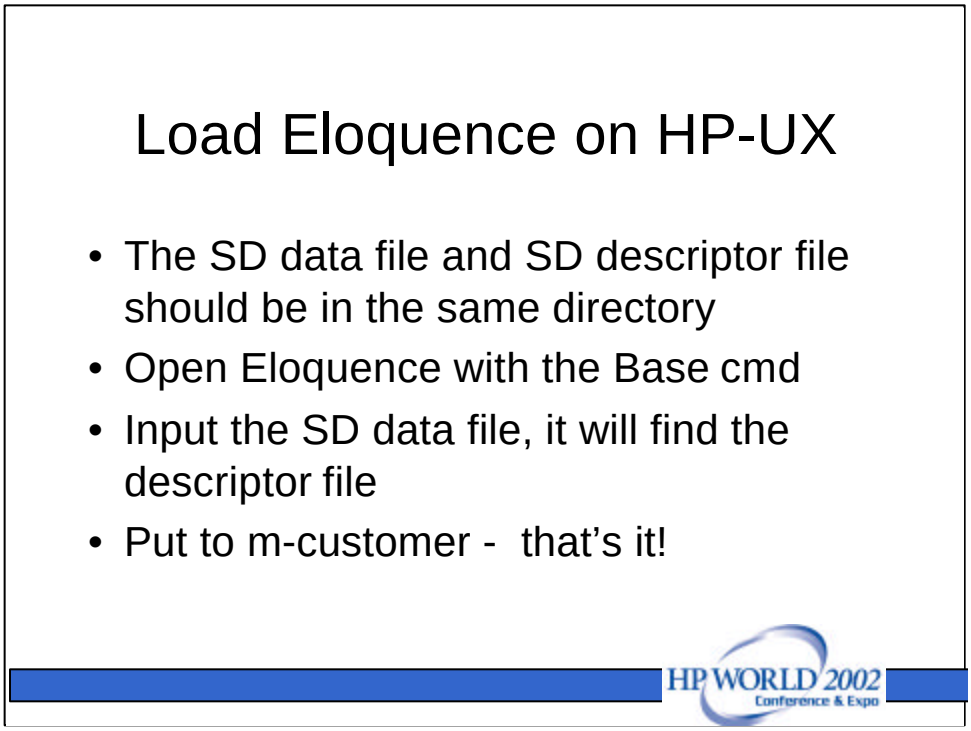
Once you dump a dataset to a self-describing LINK file, convert the SD file into 2 files for UNIX. The first file is the data file, the second is the descriptor file:

```
:run sdunix.pub.robelle;info="cust custsd nolf"
```

Then copy the 2 files to the HP-UX server using FTP:

```
:ftp.arpa.sys  hpuxbox.robelle.com
user dbadmin password
binary
put cust /users/data/store.cust
put custsd /users/data/store.cust.sd
quit
```

If you are going to use DCSCOPY specify the LF option on sdunix instead.



On HP-UX, inport the self-describing file into Eloquence using Suprtool:

```
$ suprtool
base hpuxbox.robelle.com:eloqdb/storedb
input store.cust
put m-customer
exit
```

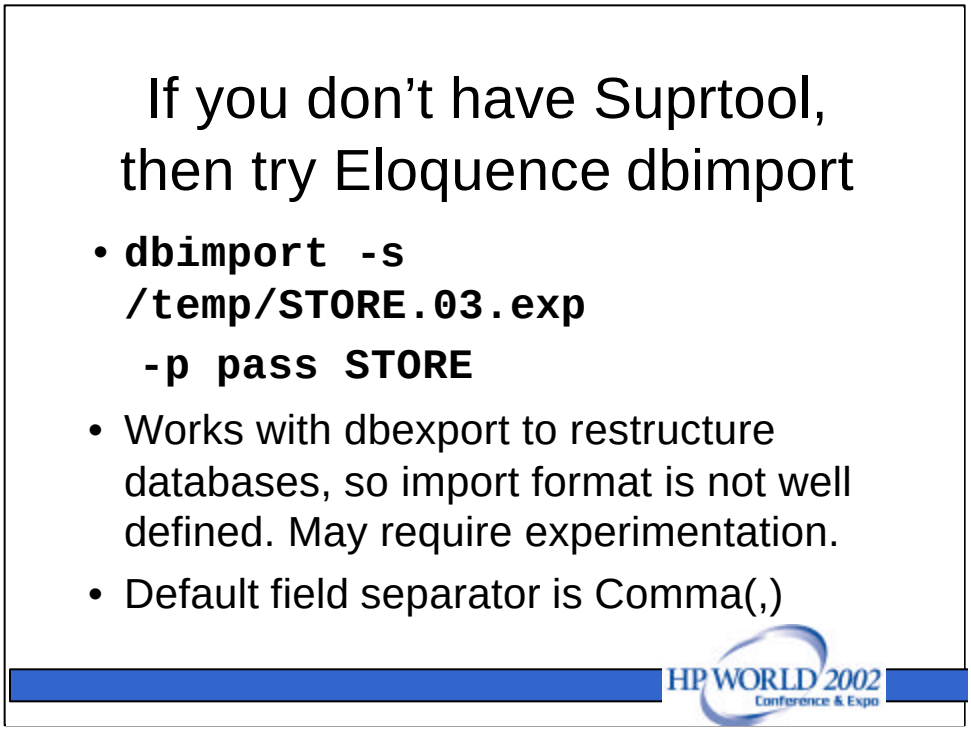
That's it!

Since the Eloquence database has more open options than TurboIMAGE, the Base command has 3 forms:

base [server][:service/]database

base :service/database

base database



Because the Eloquence dbimport was written to go with the dbexport program and to facilitate data restructuring, files to be imported must have a name in this format:

databasename.datasetnumber.exp

dbimport [options] database [dataset [,dataset...]]

-u userid -p passwd

-i path (dbimport looks for all files that match the name format); default path is the current working directory.)

-s file (import from single file instead of all the files in a directory.)

-f sep (specifies a different field separator; default is comma.)

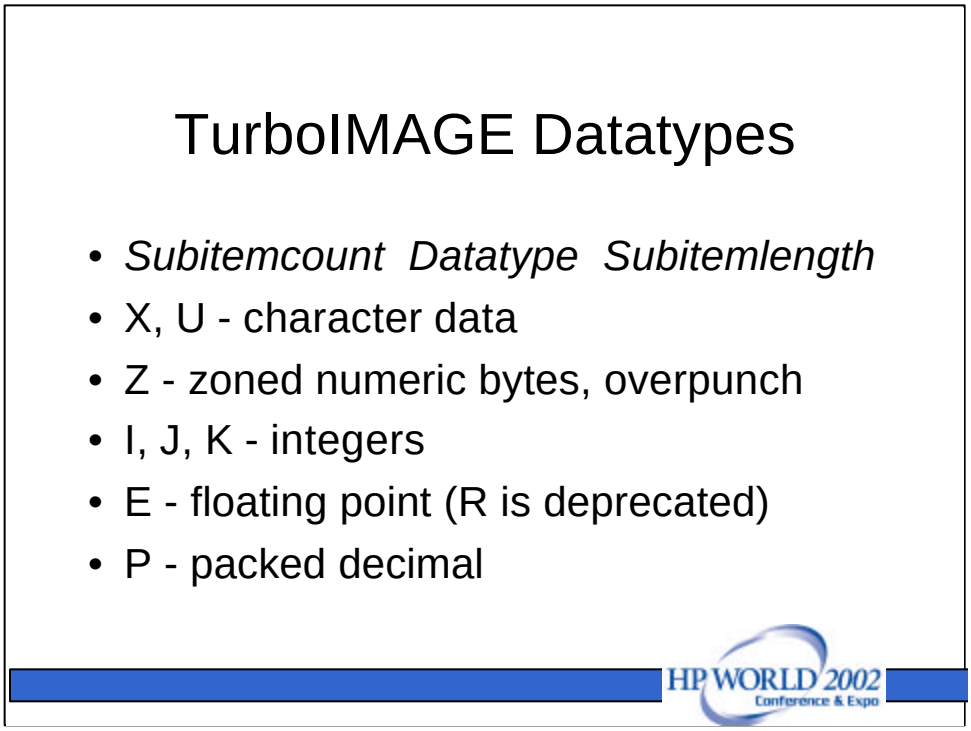
-v (verbose mode.)

For example,

```
dbimport -v -i sadexp -p secret SAD
```

finds all the SAD database export files and imports them. For example, SAD.03.exp, SAD.04.exp, etc.

The format of the EXP files is not well documented. Best way to learn the format is to export some Eloquence data and look at the EXP files created.



A typical data item definition is J2, where J is the Datatype and 2 is the SubItemLength, or 5X8 where 5 is the Subitemcount and 8 is the SubitemLength. The type designators are E, I, J, K, P, R, U, X, and Z.

E = ieee floating point. sub-item length is in halfwords (R is old floating point, you should not be using that).

I = signed integer, sub-item length is in halfwords

J = signed integer, but conforms to COBOL standards (i.e. S9999 should not have any values greater than 9999). sub-item length is in halfwords

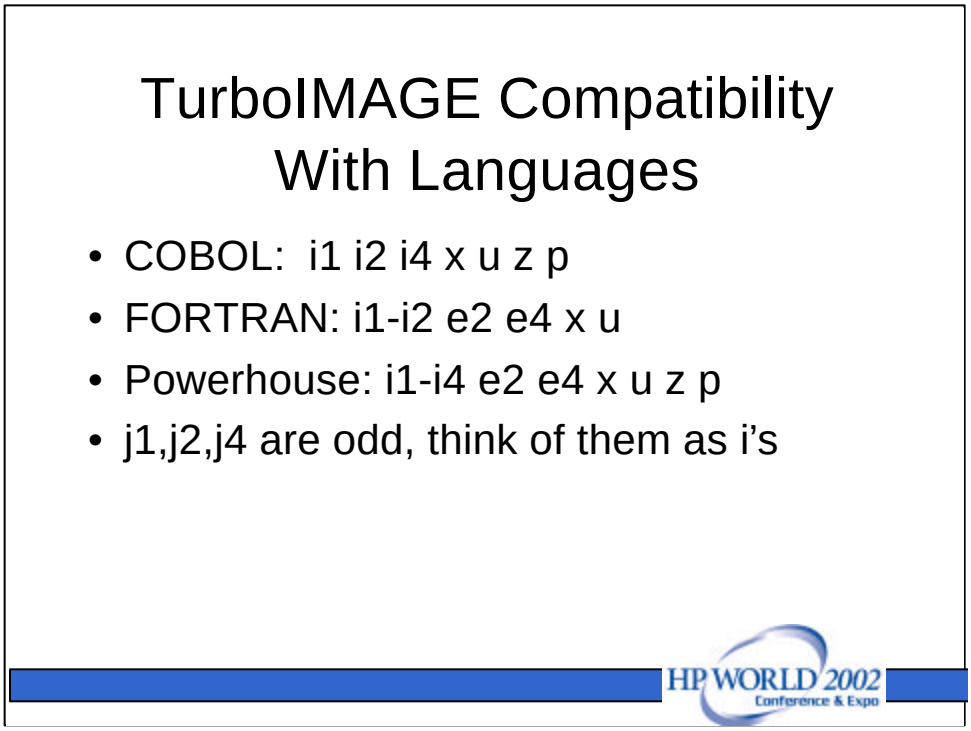
K = unsigned integer, no negative value. 1 halfword = 0-65K, 2 halfwords= 0-2 Billion, sub-item length is in halfwords

P = packed decimal, sub-item length is in nibbles, 2 to 28, with one digit used for the sign (note: TurboIMAGE will let you create a P48 or even larger, but COBOL will not process it).

U = uppercase ASCII chars, **IMAGE does not enforce uppercase!**

X = any ASCII characters, sub-item length is in bytes.

Z = zoned decimal number. sub-item length is in bytes. Overpunch sign.



Although TurboIMAGE does not place any restrictions on the reasonableness of item datatypes (i.e., you can define J25 if you wish) and does not validate values before inserting them into the database, most TurboIMAGE databases use only the data types that can be processed by the programming languages on the HP 3000.

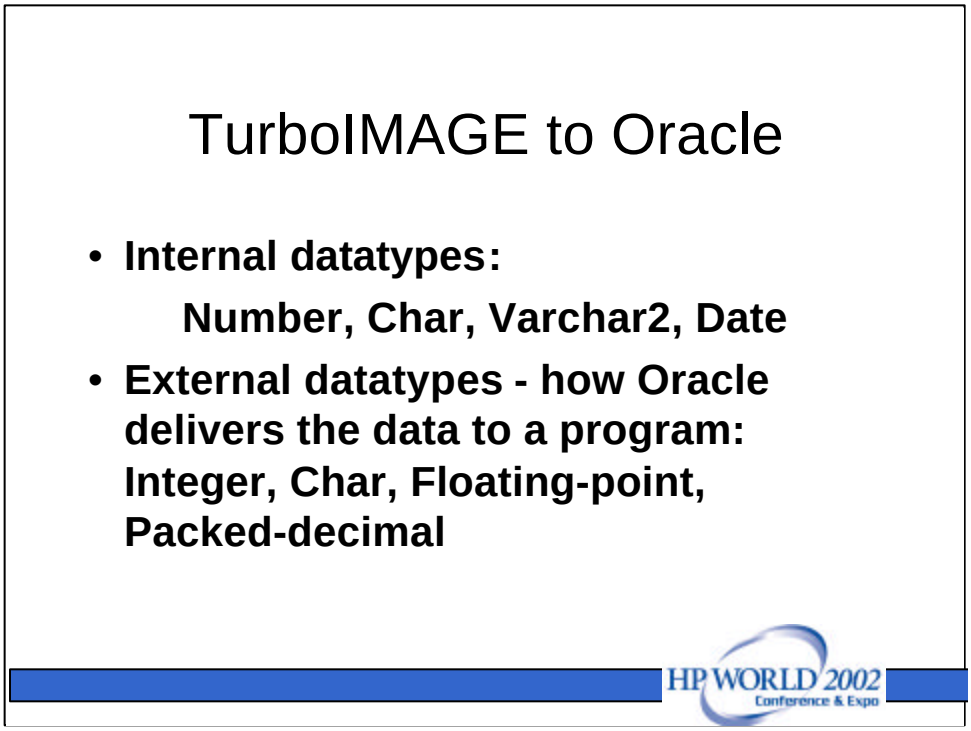
I3 is a 6 byte integer, supported only in Powerhouse and Suprtool. I4 is an 8-byte integer that is supported in COBOL, Suprtool and Powerhouse only. Expect possible migration problems with these fields.

J1 is 2-byte integer for COBOL PIC S9999; I.e. it should not have any values greater than 9999 or less than -9999, but IMAGE does not enforce this, so you can treat it as I1 in Fortran and Powerhouse. Same for J2 and J4. J3 will probably work in Powerhouse.

K1 Logical, 2 bytes, unsigned integer, define as Logical in Fortran, not supported in COBOL. Suprtool and Powerhouse support K2 and K3.??

Zn Zoned-Decimal, n bytes, s(n) Display in COBOL, overpunched

Pn Packed-Decimal, n/2 bytes, s9(n-1) Comp-3 in COBOL, not supported in Fortran. Maximum N in HP COBOL is 19 (18 digits plus a sign).



Oracle has internal datatypes and external datatypes. The internal datatypes are the format that Oracle actually stores the data in. However, these internal datatypes, might not be recognized by any of the standard programming languages. For example, Number is a variable-length format, with one byte used to store the exponent and up to 20 bytes to store the mantissa.You cannot process such a number directly in Fortran, COBOL, or C. Therefore, Oracle also has external datatypes, which are the formats that Oracle is willing to transform into for calling code written in languages that include C, COBOL, Fortran and Java.

Oracle 7 Internal Datatypes:

CHAR for X/U fields of up to 255 characters.

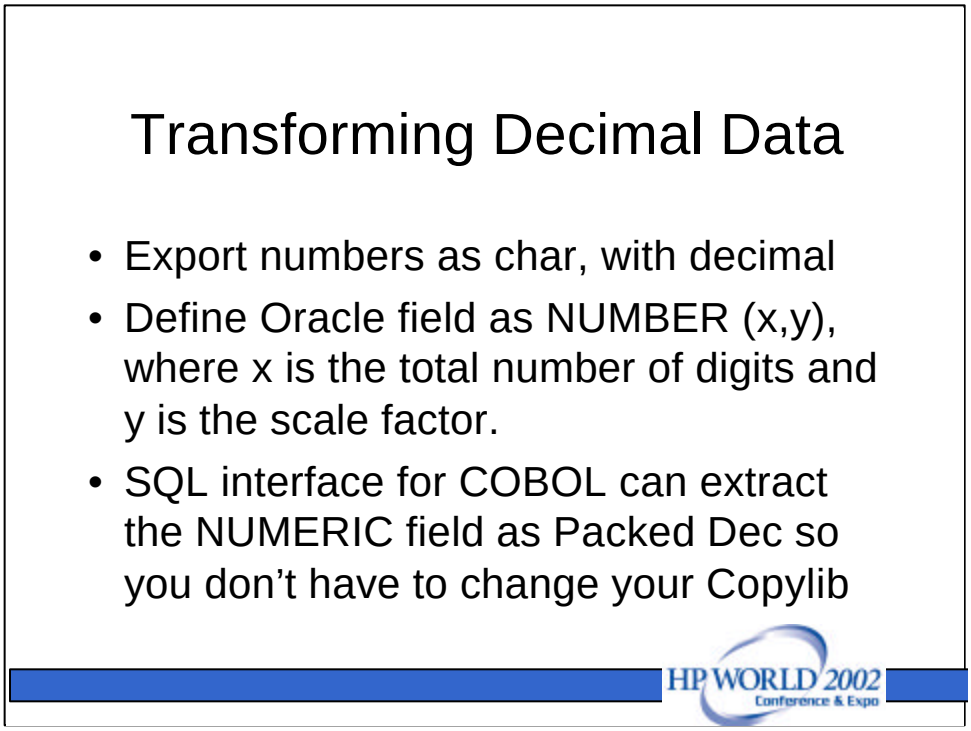
VARCHAR2 for X/U fields up to 2000 characters.

NUMBER for I/J/K/P/Z fields, up to 38 digits.

DATE for any field that contained a date value, such as SHIP-DATE X8, or DUE-DATE J2, or ORDER-DATE Z8. The Oracle date can also hold the time.

Oracle 8i and 9i Internal Datatypes:

CHAR can hold up to 2000 characters, VARCHAR2 can hold up to 4000 characters. So it appears that converting our datatypes to Oracle is straightforward.



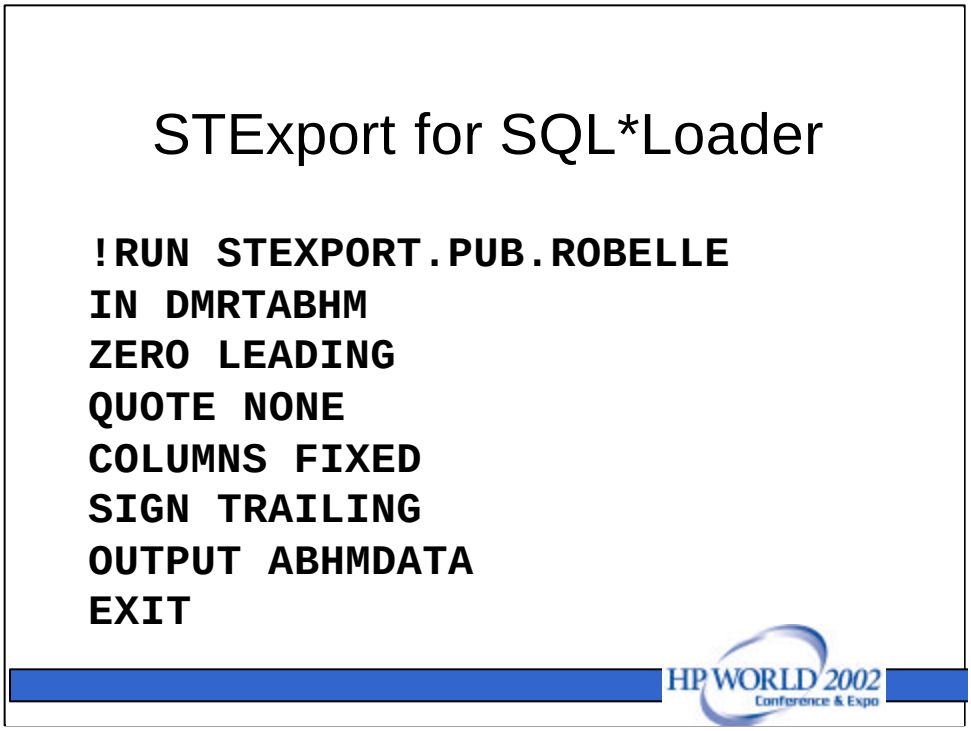
When exporting Image datasets to other sources, one of the common transfer file formats used is the "comma separated values" or CSV format. This is generally a flat file with one record per line, quotes around fields and commas between fields.

```
"cust-id","comment"
"12A","want web delivery"
```

Suprtool and other tools have options to generate CSV files. You will find Suprtool's Item command handy for defining the decimal place in your TurboIMAGE fields.

You will want the export file to have an explicit decimal place "." in numeric values where appropriate, since Oracle understands decimal places and remembers them. With a Display format, you only need to include an explicit decimal point as in 1234.89. STExport does that automatically when properly configured (Item command in Suprtool and Decimal Period in STExport).

SQL*Loader takes care of the alignment with the column definition e.g. NUMBER(12,2). If the file contains more decimals than the column definition, SQL*Loader rounds it up. For example, if you try to load 1234.5398 into NUMBER(12,2), the table will contain 1234.54. Negative values must have a leading sign (Sign Leading in STExport).

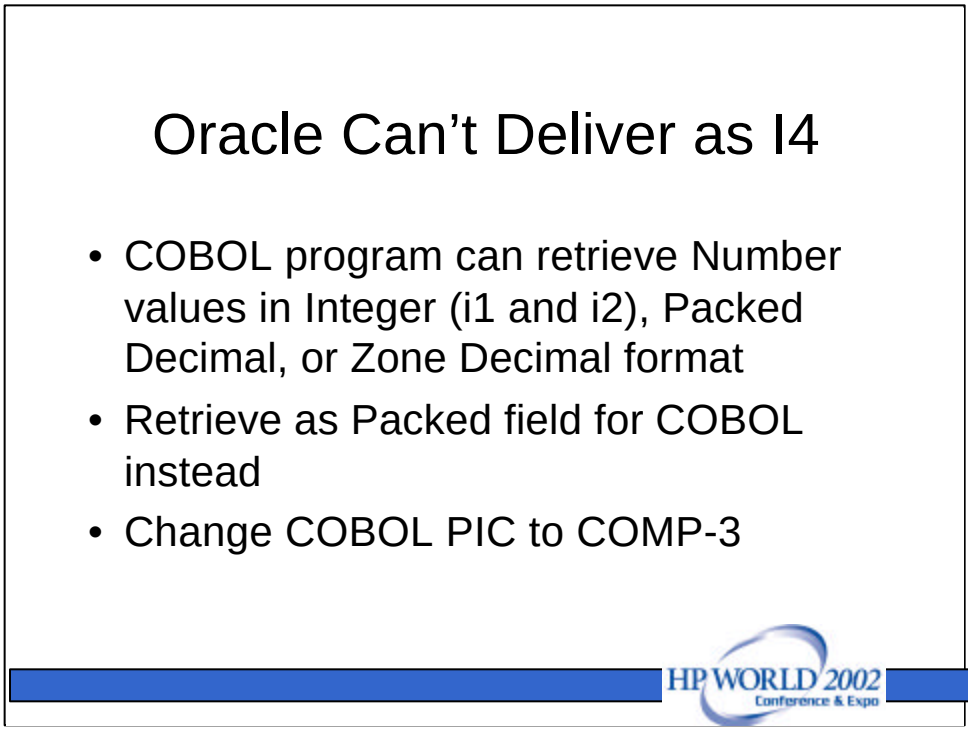


If you used Suprtool to extract a dataset into a file, you can use Suprtool's STExport to prepare the data in a format that the SQL*Loader will accept. STExport lets him define the format of numeric data, including leading zeros and the position of the sign.

The resulting files are sent using the HP e3000's FTP client to the NT computer where the data mart resides.

```
!COMMENT *** FTP OUTPUT FILES TO DATAMART ***
!RUN FTP.ARPA.SYS
open 123.456.789.012
user <<login string and password>>
ascii
exitOnError
cd /isdmdata
cd macsdata_in
put ABHMDATA.pub.hsmacs ABHMDATA.txt
dir
quit
```

In SQL*Loader, use the Load Data command with the `FIELDS TERMINATED BY ","` clause to insert the data into your table. This is just one example, with fixed-length fields. STExport and SQL*Loader also have options to use variable-length fields.



What if you have a field that looks like this in COBOL on MPE?

```
05 EXT-PRICE S9(10)V9(2) COMP.
```

In TurboIMAGE, this would be a J4, which converts to Number as the Oracle internal datatype.

But Oracle does not have an external datatype of 64-bit integer!

So you will have to use Packed-Decimal as the external datatype.

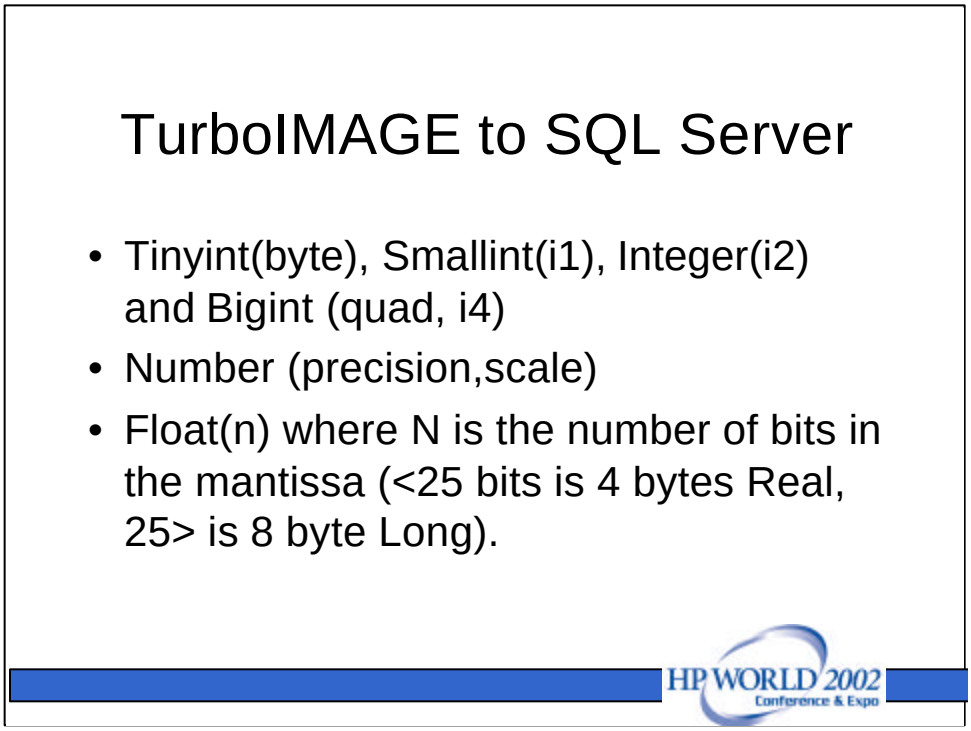
Which means changing the COBOL definition to

```
05 EXT-PRICE S9(10)V9(2) COMP-3.
```

Now your programs are different and any files you create with this data will probably need new data definitions.

Note: If your internal Number field has decimal positions, then you will need to always convert it to an external Packed datatype (COMP-3). If you tried to convert it to an external Integer datatype, you would lose the decimal places. This is true regardless of the size of the field.

This does not sound too bad, unless you want to keep the same source on MPE and UNIX. Or if you have hundreds of tasks that may process this data!



For integer values without a decimal place, use one of the Integer datatypes: Tinyint (values 0 to 255), Smallint (-32,768 to +32,767), Integer (-2B to + 2B), Bigint (for very large values, up to 18 digits; introduced in SQL Server 2000 -- ensure that your Windows COBOL compiler supports Bigint).

NUM or NUMBER or DEC = numbers with decimal places. You specify a precision and scale for the values. Precision is the maximum total digits in the values, with 38 the largest allowed by SQL Server. Scale is the number of places to the right of the decimal. The maximum number of digits that can be placed to the left of the decimal is precision-scale. For example, DEC(7,2) means the same as S9(5)V9(2) in COBOL. NUMERIC or FLOAT is the datatype for any value with a decimal place. NUMERIC in SQL Server is much like NUMERIC in Oracle, although it does not have the odd "negative scale factors" of Oracle (scale factor-3 in Oracle actually multiplies the value by 1000!).

FLOAT(n) = approximate numeric values in floating point format. Supported in 4 byte and 8 byte formats. A floating point number has an exponent and a mantissa. FLOAT(n) specifies number of bits for the mantissa, which can be up to 53. 1 through 24 specify a single precision real (4 bytes) and 25 through 53 specify Double Precision (8 bytes). Same as e2 and e4 in TurboIMAGE.

Other SQL Server datatypes that you will find useful are MONEY and DATETIME.

What about Compound Items?

- TurboIMAGE has 5x10 for an array of five elements, each with 10 characters.
- This is called a compound item
- Not supported in SQL databases
- Convert each element to a separate column: address1, address2, address3…

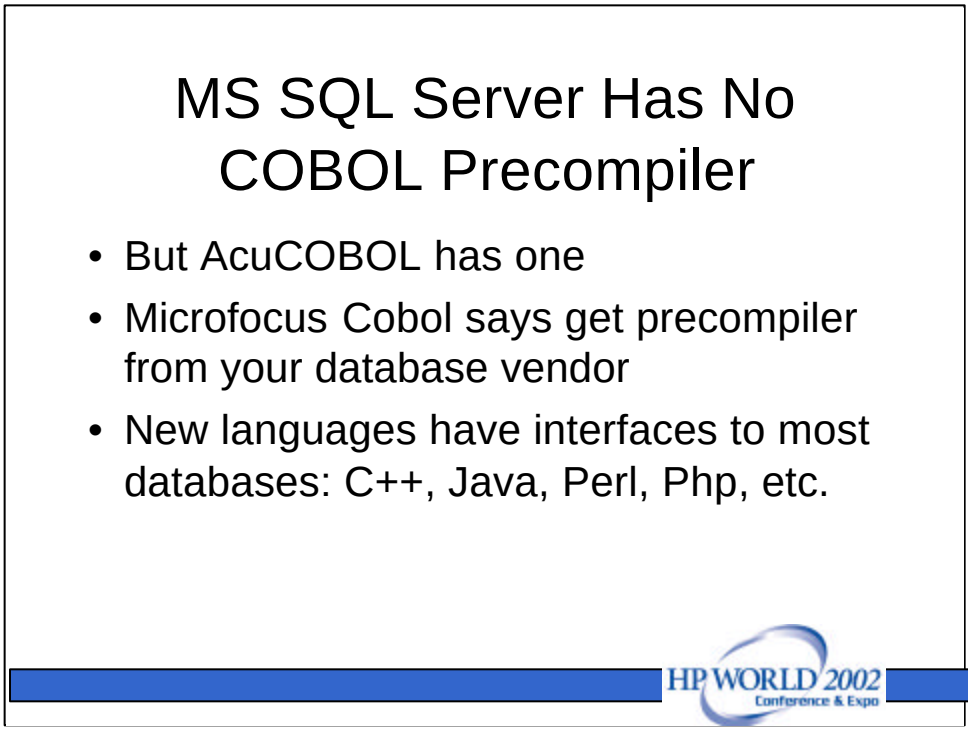

```
M-CUSTOMER       Master                    Set# 1
   Entry:                        Offset
      CITY                    X12     1
      CREDIT-RATING           J2     13 <<0.00>>
      CUST-ACCOUNT            Z8     17
      CUST-STATUS             X2     25
      NAME-FIRST              X10    27
      NAME-LAST               X16    37
      STATE-CODE              X2     53
      STREET-ADDRESS         2X25    55
      ZIP-CODE                X6    105
Capacity: 211 (7)  Entries: 12  Bytes: 110
```

The repeated item STREET-ADDRESS (2X25) converts into ADDRESS1 and ADDRESS2 in SQL, since SQL does not have repeated data items.

CUST-ACCOUNT is only Z for hashing, so convert it into INT.

Since CREDIT-RATING values have 2 decimal places, convert into the NUMBER datatype, which understands decimal places.

Since we are redefining fields, why not expand ZIPCODE from X(6) to VARCHAR (16) so it can handle extended and foreign postal codes?



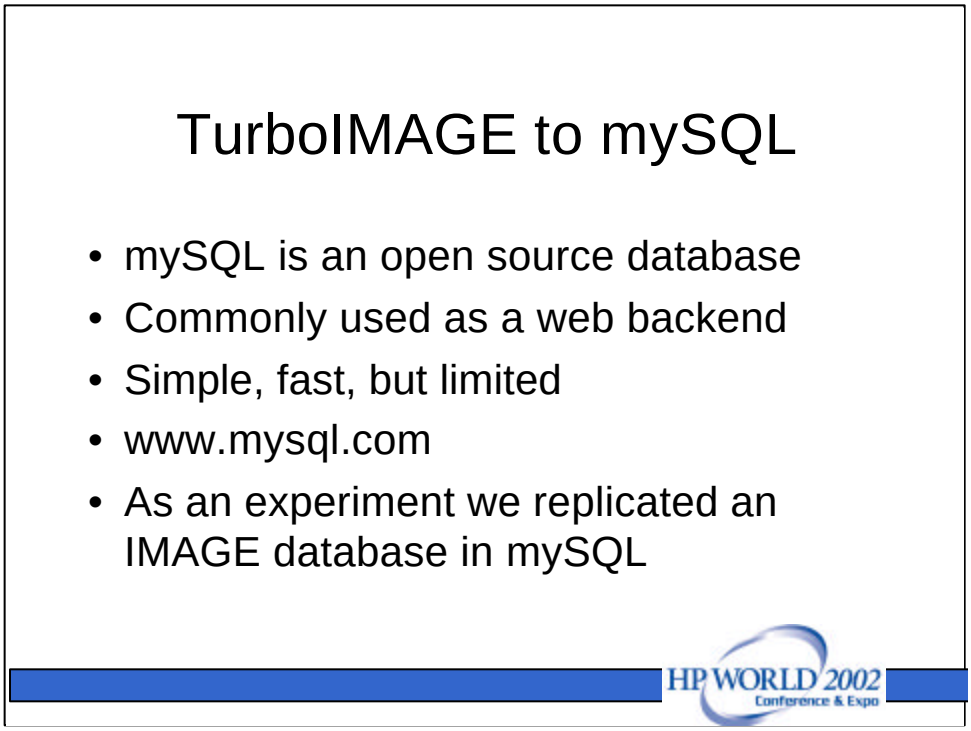
With Oracle, the package comes with pre-processor for Fortran and COBOL, but SQL Server only provides a pre-compiler for C++. What if you want to do SQL Server functions in your COBOL program? You need to look to your compiler vendor.

For example, AcuSQL has an Embedded SQL (ESQL) precompiler that lets you embed standard SQL directly into ACUCOBOL program.

`www.acucorp.com/Solutions/access3.html`
`www.acucorp.com/Solutions/acusql.html`

However MicroFocus COBOL says "COBOL precompilers for the different databases should be obtained from the appropriate database vendor."

I looked for a FORTRAN precompiler for SQL Server, but did not find one, so that problem is left to the reader. Keep in mind the database you select may not have precompilers for all your programming languages.



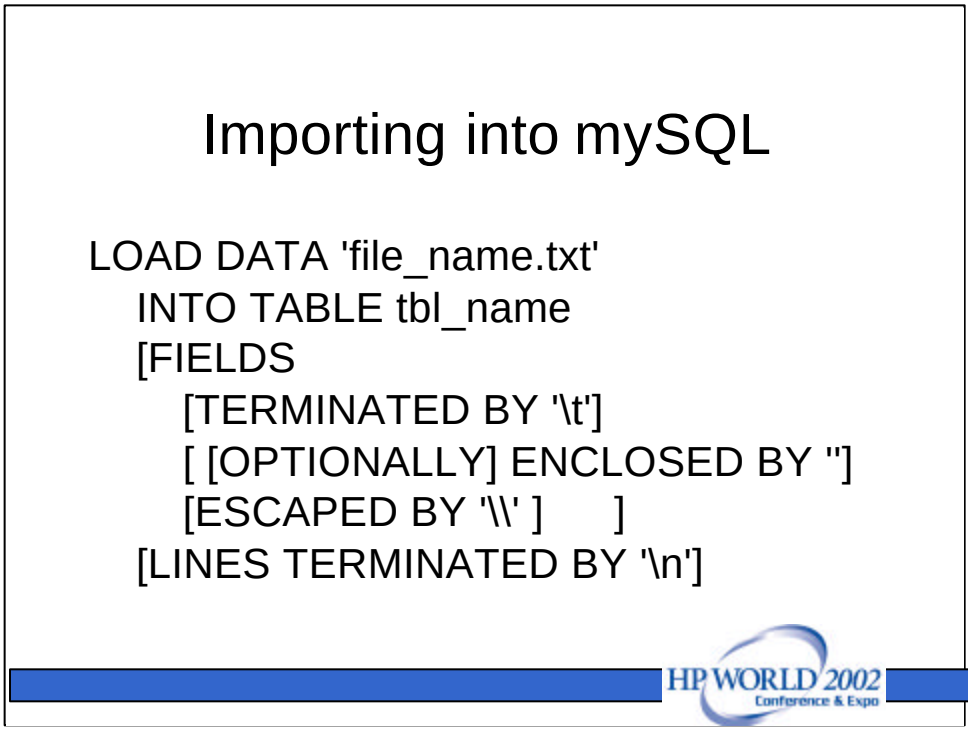
mySQL is an Open Source database that is commonly used as the backend database server for many Web applications on Linux and Unix platforms as well as Windows machines.

The source and or binaries can be obtained from www.mysql.com or www.sourceforge.net and many other download mirrors around the globe.

As an experiment we built a mySQL database that looked like an Image database, building a simple Master dataset and a single detail dataset. The byte type fields in Image were created as char fields. The I2 or J2 fields were created as int fields.

Having done this, we extracted data from the HP 3000 database using Suprtool and then used default STExport settings to output a file that was comma delimited, with each text field enclosed in quotes.

We then attempted to import the comma-delimited file into mySQL.

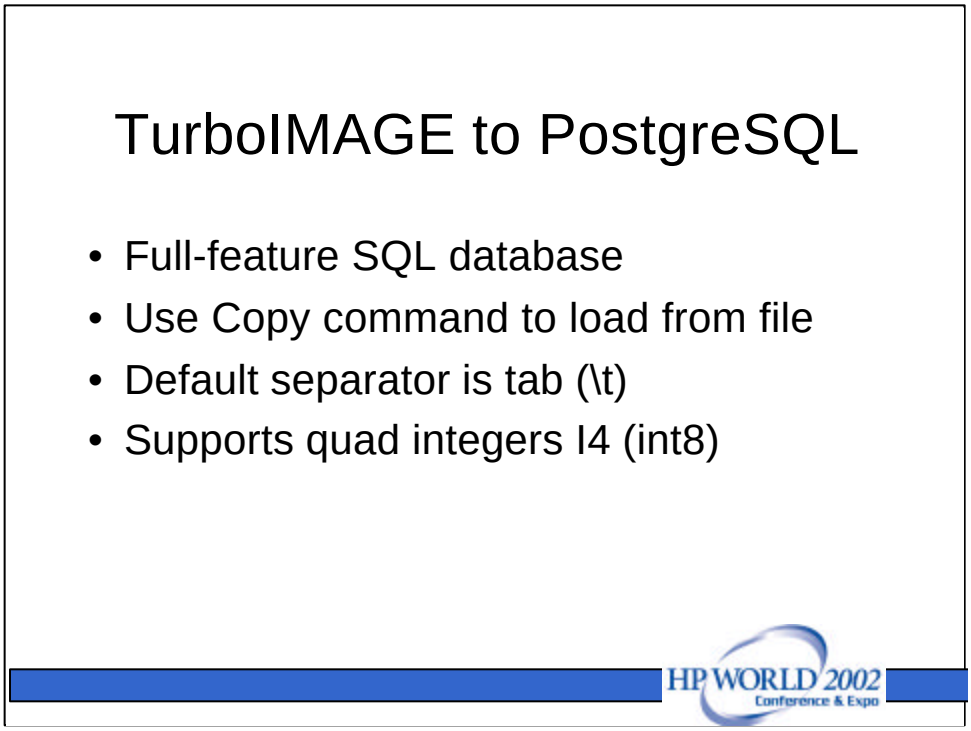


In investigating how to import data into mySQL, we first tried the mySQLImport program, but it didn't seem as robust and we could not figure out how to tell it what delimiters to use. In looking at the documentation, we thought that the LOAD_FILE command might work, but further investigation showed that this command opens the file and returns the contents as a string. This feature is only used by Text and Blob columns of mySQL.

We finally had success with the **LOAD_DATA** statement. If you don't specify a `FIELDS` clause, the command acts as follows: Look for line boundaries at newlines; break lines into fields at tabs; do not expect fields to be enclosed within any quoting characters; interpret occurrences of tab, newline, or `\' preceded by `\' as literal characters that are part of field values.

If you specify a `FIELDS` clause you can change the delimiter from Tab to comma (or another character).

We had trouble with the comma delimiter, because our actual data contained commas, so we used question mark as the field delimiter (or we could have used Tab). In order to import fields exported with “?” as the delimiter, we used `fields terminated by '?'` in the LOAD_DATA command.

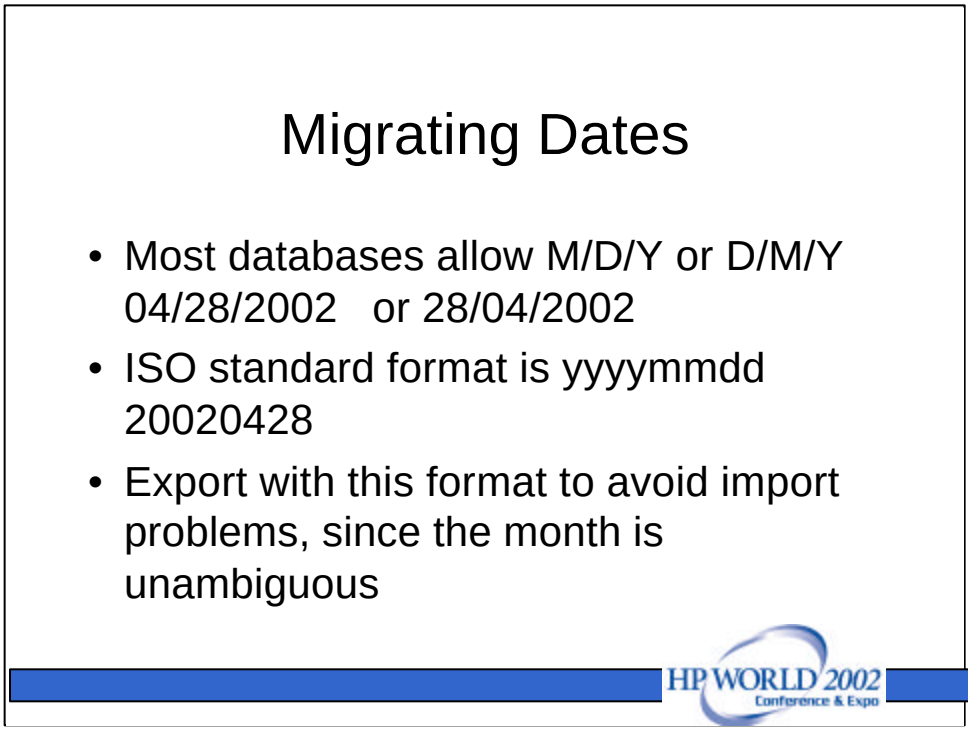


PostgreSQL is an Object-Relational database management system that supports almost all SQL constructs, including subselects, transactions, and user-defined types and functions. It is free to download, use, and modify, and it runs under many OS types, including Unix and Windows.

For an experiment, we created an address table and loaded it from a Comma Delimited data file.The next step was to populate the table with the data from the text file. This was easy to do by using the COPY command, which loads large amounts of data (either character or Binary) from flat-text files. You tell COPY which file to load by specifying `FROM '/directory/file'`. By default, COPY uses a tab ("\t") character as a delimiter between fields, I had to change this behavior by including `USING DELIMETERS ','`

The COPY command, however, has a few potential pitfalls. Such as if you don't have enough columns in the file, you will get an error, but if you have too many columns you will get a warning only and the extra columns are ignored. Also remember that COPY is executed as a transaction, meaning that a single error in the data causes an undo of the entire import operation. As always it is good practice to read over the intricacies of the COPY command in the PostgreSQL help docs (http://www.postgresql.org/idocs/).

The PostgreSQL data types are like other SQL databases (Number, Int, Varchar, Date, etc.). The Money datatype is deprecated in favor of Number because it is US-oriented.



Most SQL databases allow you to import dates in either Month/Day/Year format or Day/Month/Year, but there are still configuration commands that you must get correct.

The ISO standard format for dates is actually

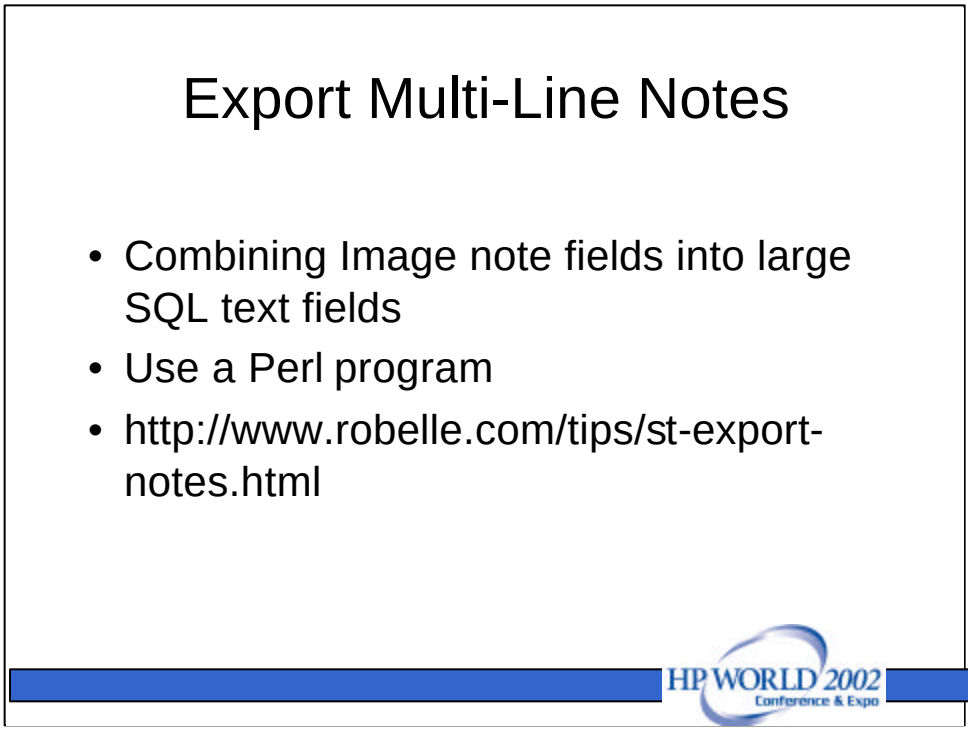
YYYYMMDD (20020428)

If you export date fields in this format, you should have minimal problems loading into your SQL database.

On the other hand, if you have an HP 3000 data field in the form

MMDDYY 042802

and export it in that format, you will need to configure your import carefully to ensure that your SQL database contains the correct date.



There is one type of field that cannot easily be exported and loaded into SQL: Notes fields, which span multiple records even though they are logically one field. They usually look like the following:

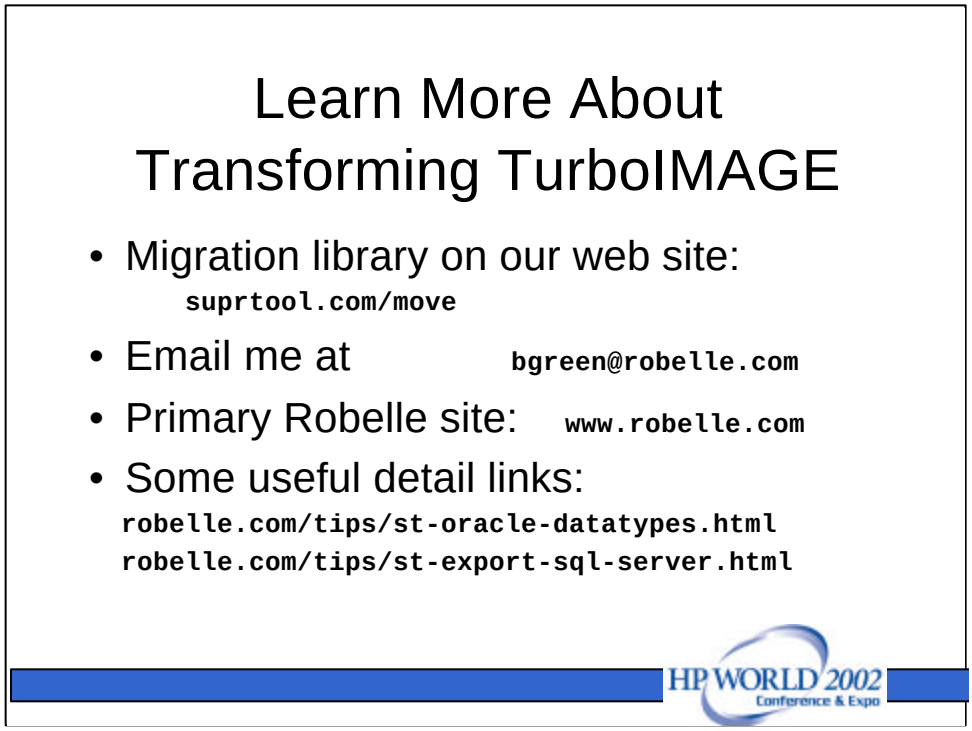
```
cust-id  seq-num  comment

12A      001      want web delivery but
12A      002      limited by bandwidth,
12A      003      so use FTP.
88X      001      Send doc in PDF format.
```

We want to merge the related records into a single record. For example:

```
    "12A","want web delivery but limited by
bandwidth, so use FTP."
    "88X","Send doc in PDF format."
```

Although Suprtool cannot produce this format directly, it can front-end the database extract portion, and let a straight-forward Perl script do the merging. If you have this problem, refer to the web link above and the Perl program that it contains (this program can be configured for many common problems).



This paper is just a summary of the many details that you will need to learn in order to migrate your HP 3000 data.

At Robelle, we have created a large number of application notes on this topic, adding more as we learn more. The primary migration web page is

http://www.suprtool.com/move

which links to numerous other pages.

Our corporate web page is at

http://www.robelle.com

If you want to add your own experiences or you have any questions about this paper, email me at

bgreen@robelle.com