

## Debugging Siebel EIM results (Part I)

Author: Ioannis Xanthopoulos  
Siebel Contractor  
[ix@ixanos.com](mailto:ix@ixanos.com)

This article presents a practical approach of debugging EIM processes using Ixanos (<http://www.ixanos.com>).

The rough procedure of migrating data into the Siebel database using EIM is:

- 1) populate the staging tables
- 2) pre-processing activities
- 3) tune the database to prepare for the data load, create auxiliary db objects
- 4) identify and populate the appropriate interface tables
- 5) populate the interface table(s)
- 6) determine the EIM logging level
- 7) start the EIM task
- 8) check the results
- 9) check the EIM logs
- 10) clean up and post-processing activities

Errors may appear in any of the high-level stages identified above, but in this article we will only focus on stage 8 errors.

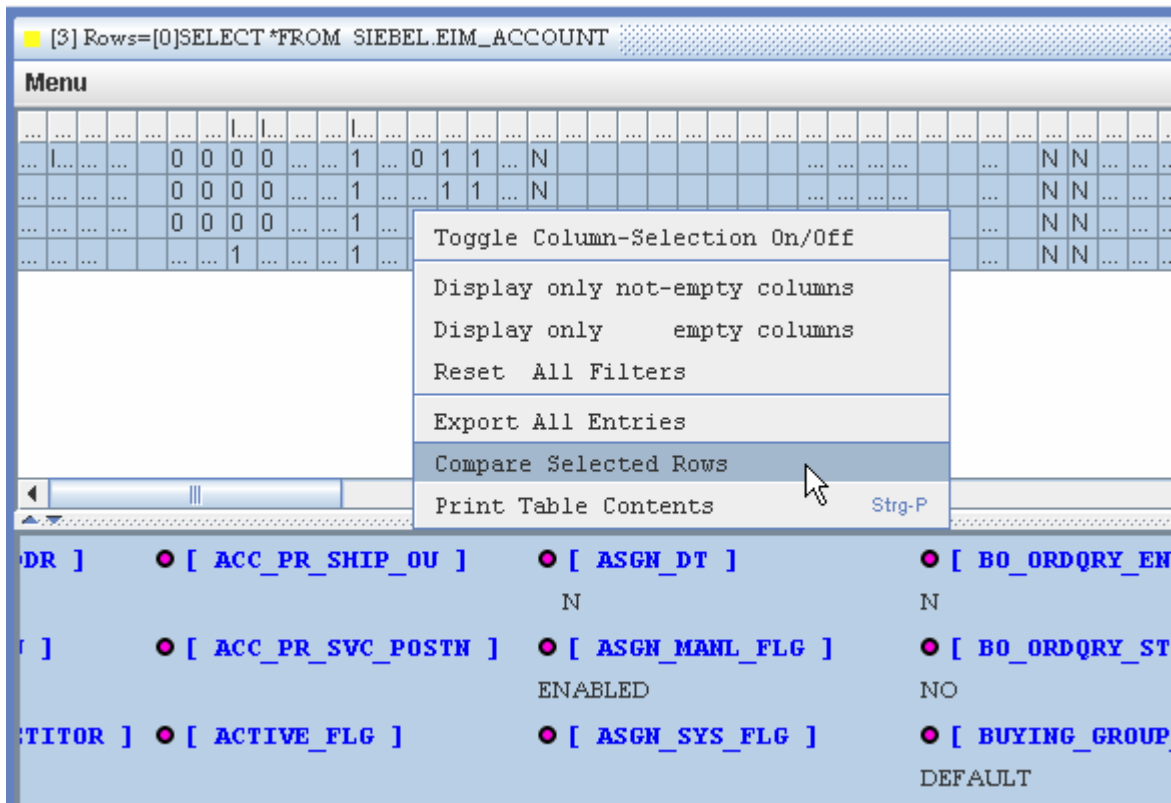
Typically, errors in this step emanate from incorrectly defined foreign keys in the interface tables. In other words, the values entered in the interface tables could not be resolved to match the foreign keys in the target database. Other error sources are duplicate or partly populated rows, faulty LOV values, numeric vs alphanumeric values, not expected attributes (suppressed in the .ifb file) etc.

In order to swiftly identify the source of errors in this stage, it is very helpful to select some rows that have been successfully processed and some that haven't and to compare them in order to identify all value-differences. Most of the times a list of the different values in a given interface table column is self-evident and the problem can be solved swiftly.

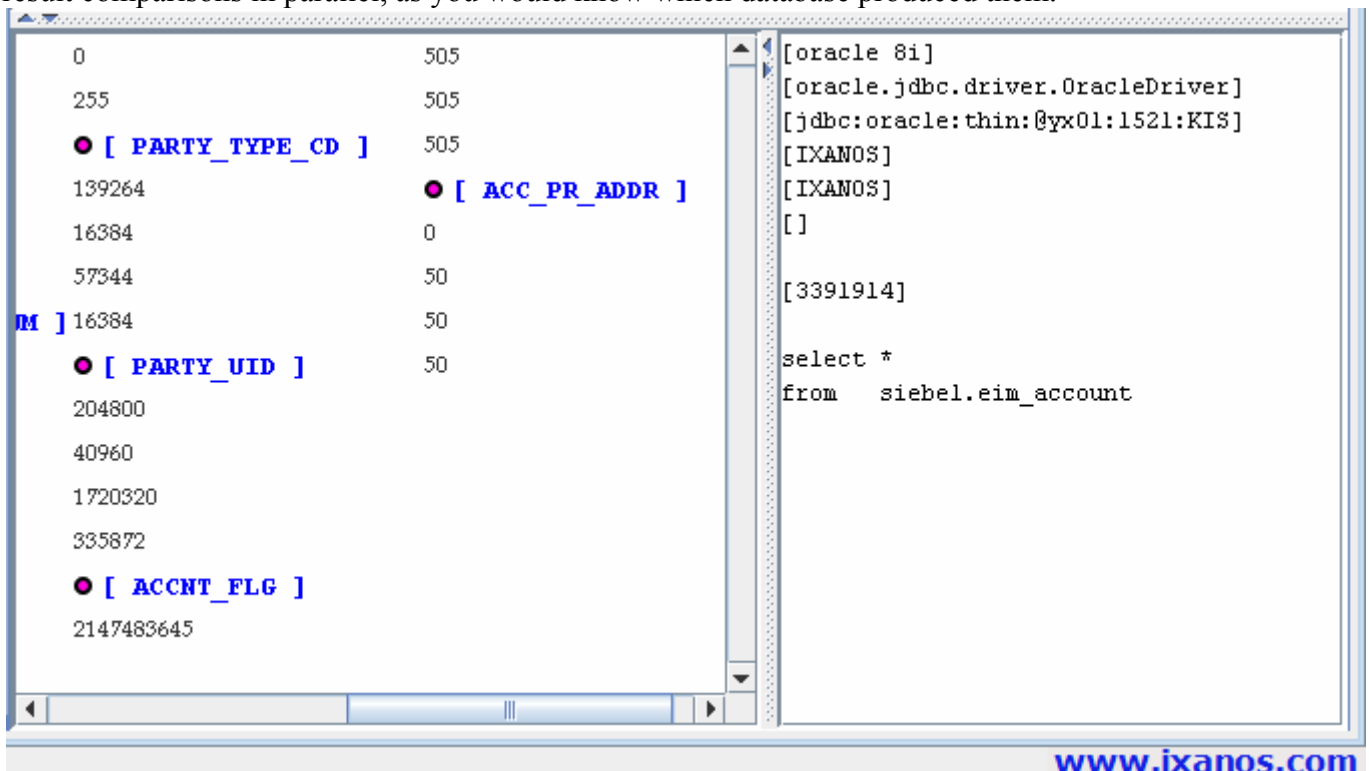
To select both successful as well as failed rows from the interface table, we can use a query similar to the one that follows. In case the interface involves more than one interface tables we could involve the remaining tables as well. Further on, the staging tables could also be a good comparison source.

```
select *
from siebel.eim_asset
where if_row_batch_num = 1000
and if_row_stat = 'IMPORTED'
fetch first 5 rows only
union
select *
from siebel.eim_asset
where if_row_batch_num = 1000
and if_row_stat <> 'IMPORTED'
fetch first 5 rows only
```

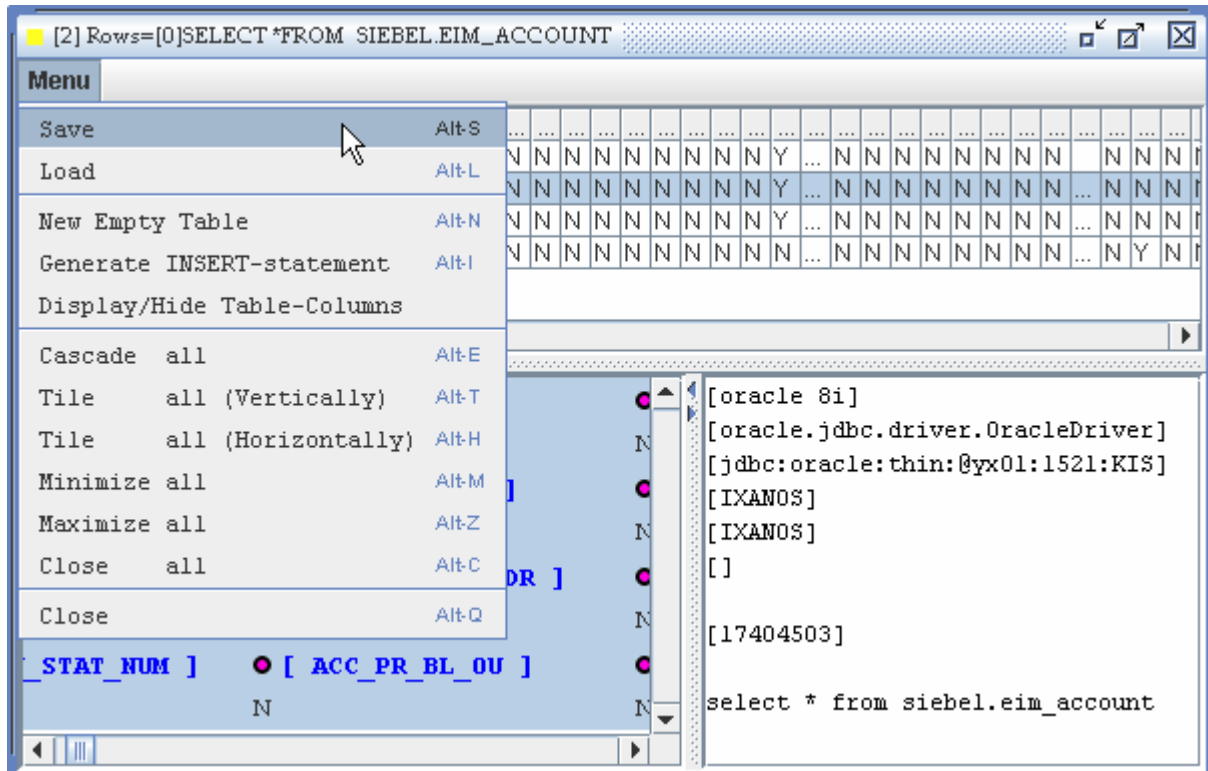
In the displayed results we can then simply utilize Ixanos and right-click and select the option “compare selected rows” and analyze the differences.



The differences are then displayed in a separate list. This list will only contain the columns that have different values in the rows that we have selected. These columns will be presented in bold letters and will have all its different values listed directly underneath them. A snapshot of such a list could look as follows. The right panel shows the connection details, ignoring passwords and allows you to have several result comparisons in parallel, as you would know which database produced them.



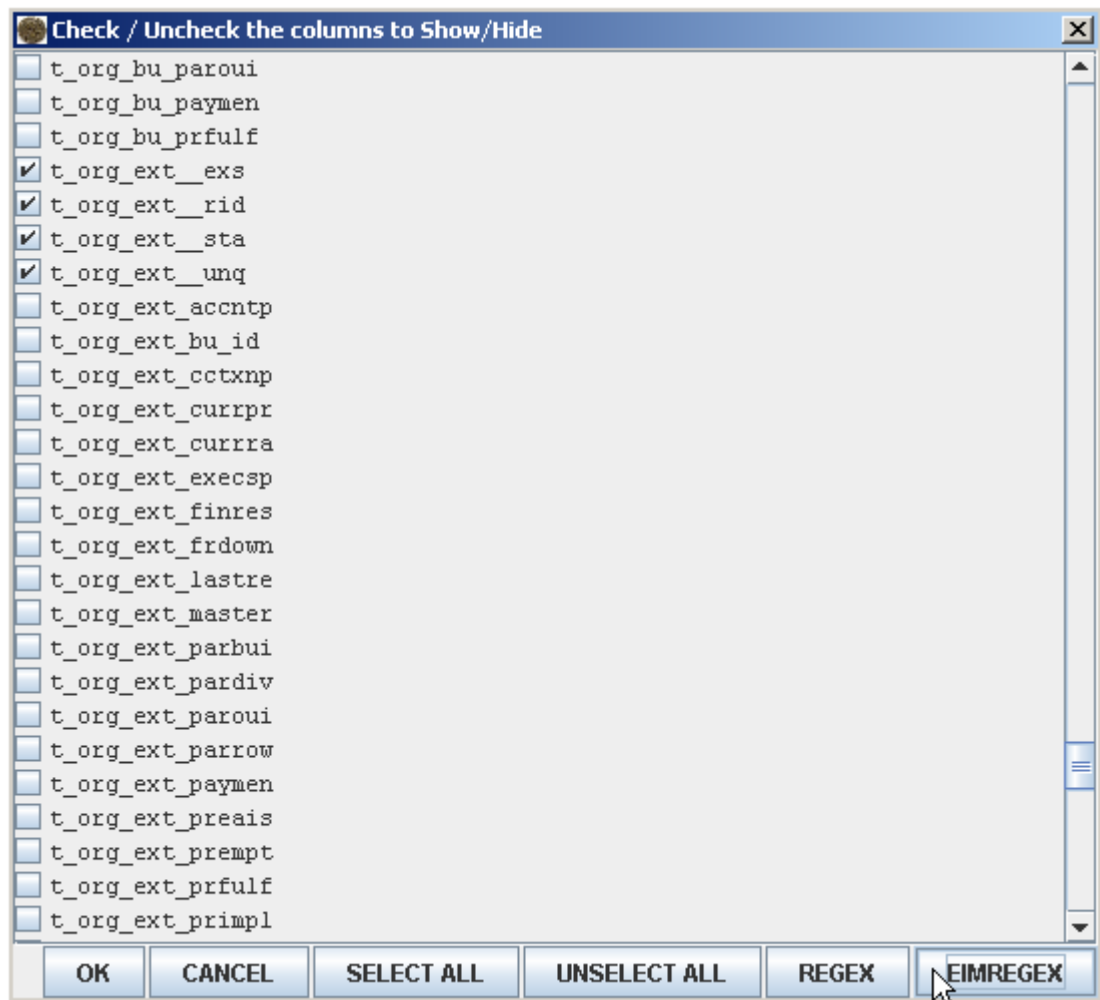
Further on, you may store these results on disc and port them to any other computer for a comparison or review with your team or to debug them at a later point in time. In case you have results from previous runs of the very same interface you can also load those into Ixanos and compare them to the current results. Both save and load functionality is provided in Ixanos and allows you to store any SQL-query results that you want.



Practical experience shows that sometimes EIM tasks run flawless in an environment and fail when ported on another environment. Different environments typically correspond to different databases. In this case we would need to setup a database link to run the query above and collect successful and failed rows from both environments simultaneously. Then again using Ixanos, we can simply connect to both databases, run the select statement against each of them and then drag and drop the results into a new table or one of the existing tables. And finally just use the Ixanos option “**compare selected rows**” and analyze the differences.

Another approach could of course be to store the results of the query of both environments on disc and load them into Ixanos and compare them. This would make sense in case one of the environments is not available and you want to run the comparison later.

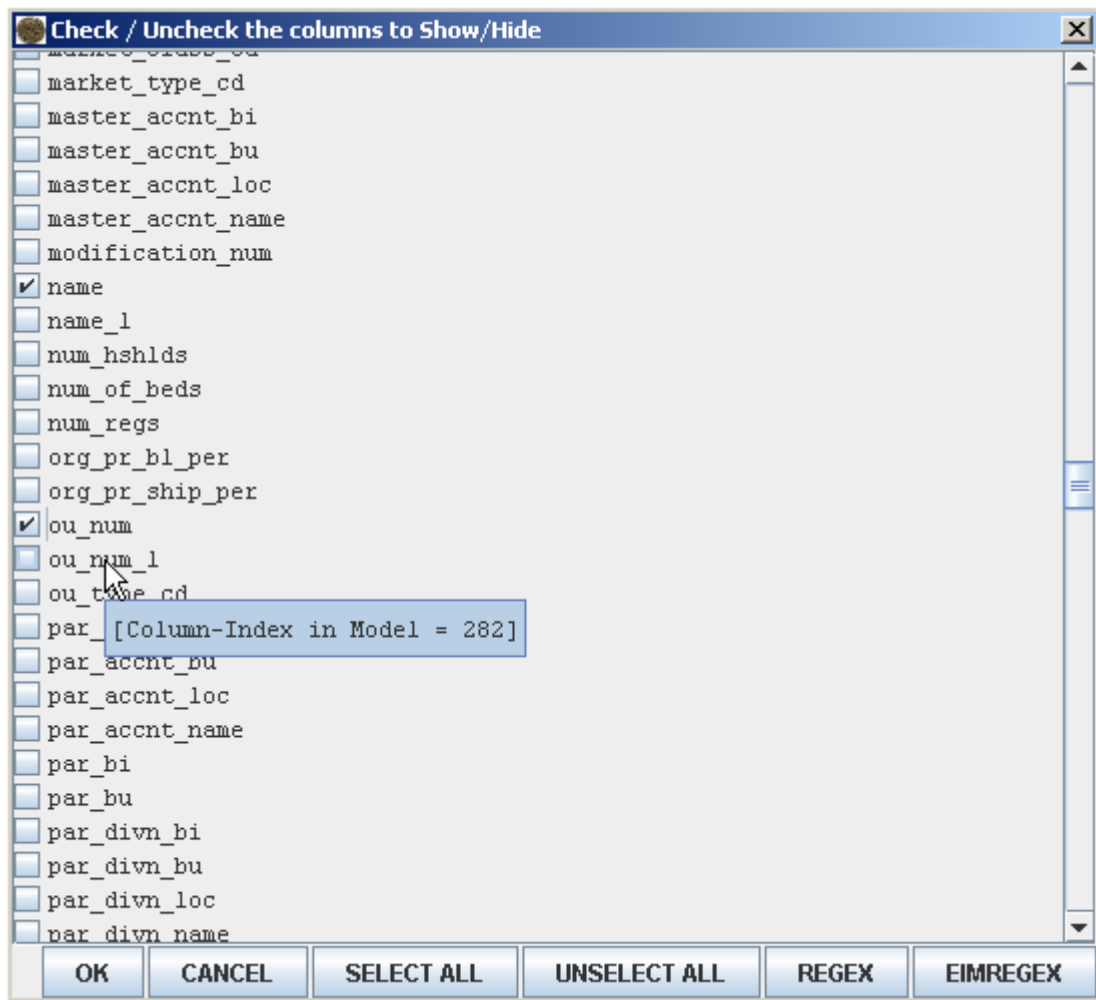
Another possible scenario is that of the EIM-run being a completely new one, in which case you have no results to compare to. In this case it is advisable to concentrate on the control-T\_ columns in order to see which have been resolved and which not. This can be easily done using Ixanos, by just filtering them out of the interface table using the integrated menu option EIMREGEX as can be seen below:



Once the control-T\_ columns ( \_\_EXS, \_\_UNQ, \_\_RID, \_\_STA) have been selected you may also wish to display the user key of the underlying base table. This can be done either by simply enabling the selected columns manually or by using the REGEX option.

These filtering options are complementing each other and will not erase the T\_ columns selected previously.

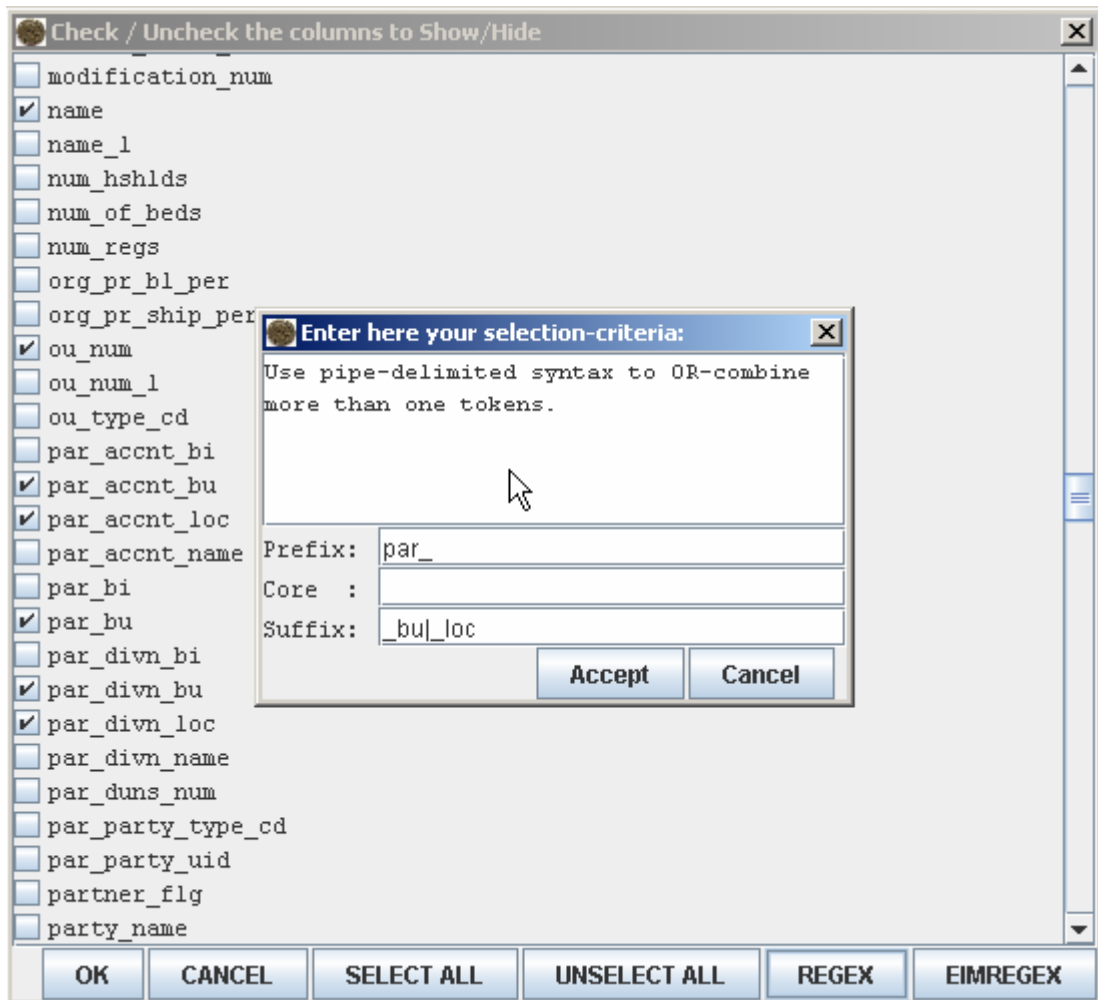
An example can be seen here:



Finally, to also include all columns that fulfil certain criteria, for example all columns starting with PAR\_ and ending either with \_BU or with \_LOC you can use the REGEX button and define the exact columns you wish to include by defining their 3 parts (prefix, core and suffix). All three parts can contain multiple values that will be OR-combined to determine which columns will be included in the result-set.

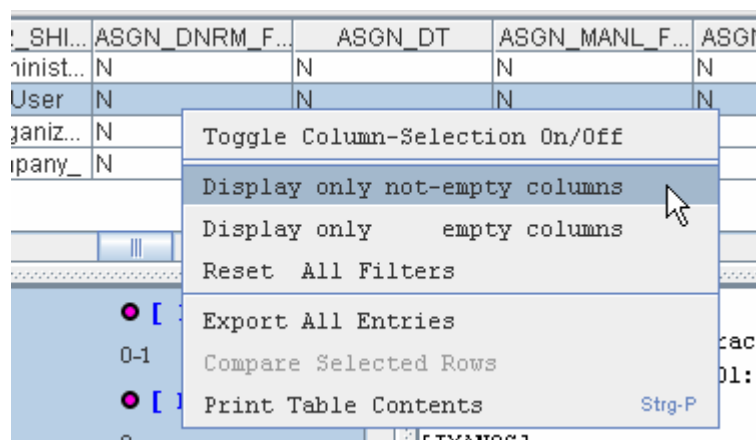
In the example below we select all columns that start with PRE\_ and end with \_BU or with \_LOC. This would include columns like PAR\_BU and PAR\_DIVN\_LOC.

Pay attention to the pipe-delimiter in the definition of the suffix in the input mask (bu|loc). Also note that Ixanos uses lower-case for all columns. This is done to have more meaningful sorting of the column names.



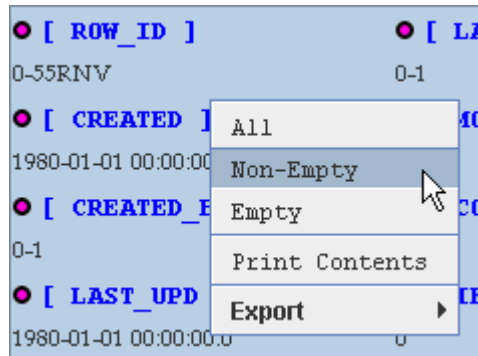
In case we still haven't understood the reason why the EIM task fails, we would have to look at all non empty columns of the interface table or perhaps just look at all the empty ones to verify that we have indeed populated all columns as indicated in the .ifb file.

Of course some Siebel interface tables have more than 200 columns and observing their values may be a very frustrating exercise. In this case we can simply utilize Ixanos and ask him to present to us all columns that have at least one row in the result set that has a value or we can have Ixanos present to us all columns that have no values in all selected rows.



Finally, we may wish to concentrate on a single row of the result set. Ixanos present the result set in two views. The first one shows all columns of the row as a list applet. The second one displays all columns of

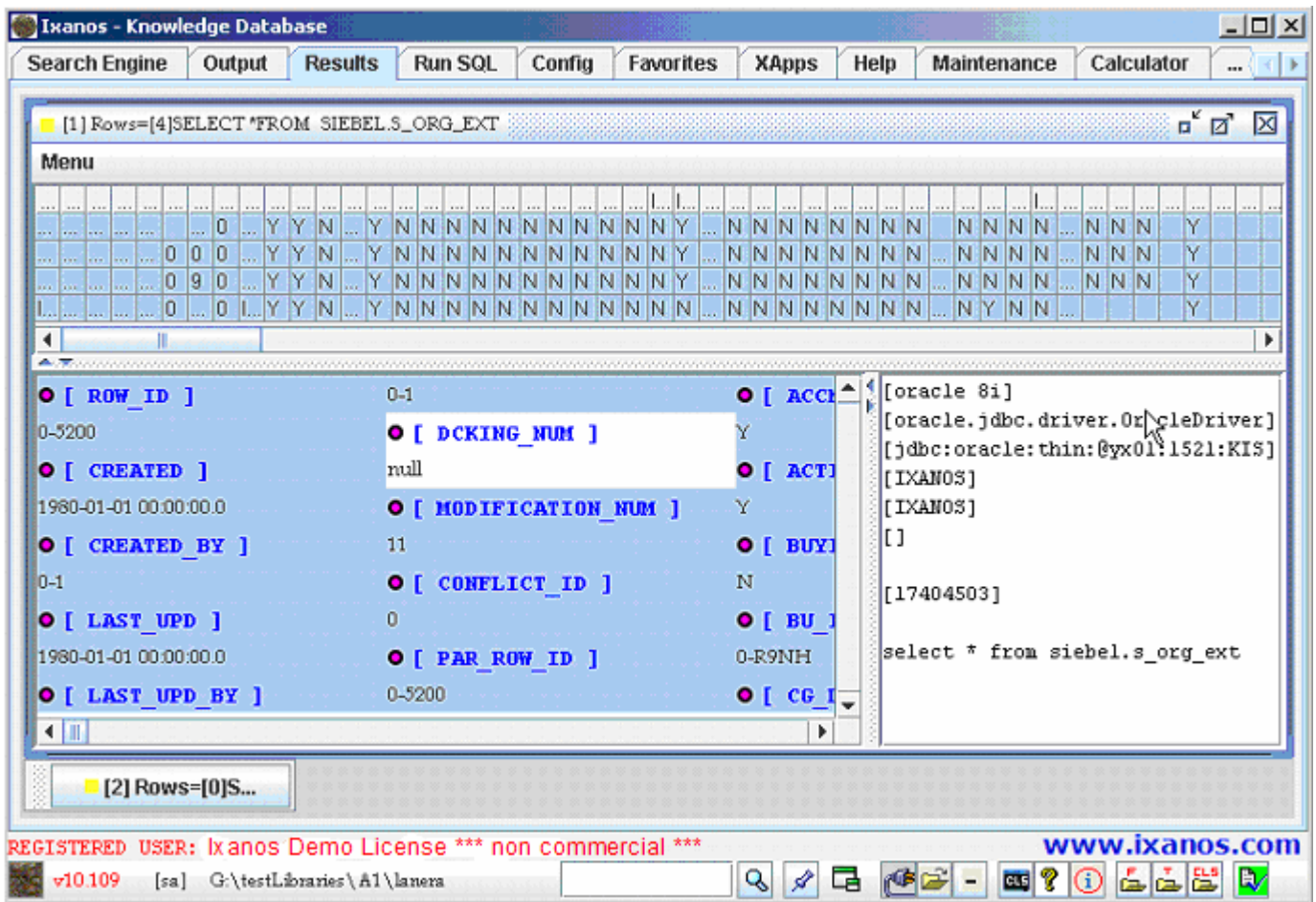
the row as a form applet. On this form applet you have again the option to display only non-empty columns (ie columns that have a value) or to display all columns that are empty.



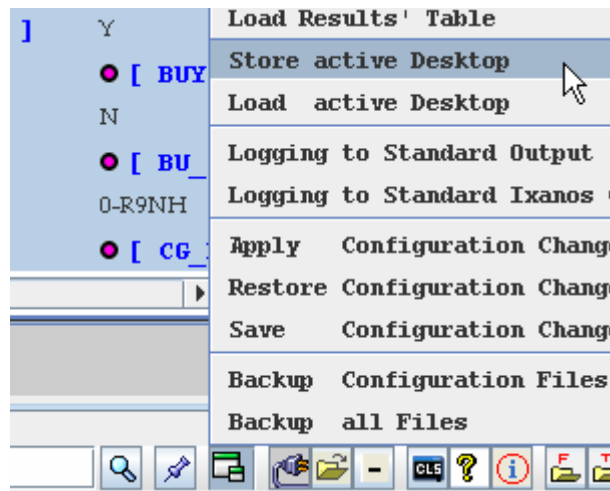
A typical list- and form-applet view of an Ixanos SQL results' set can be seen here. As you can see for the classic table S\_ORG\_EXT browsing the column values in the list applet may be difficult. Then again the form applet gives you a much easier way of viewing the data.

Note, that the filled columns are highlighted in blue as opposed to the empty ones (like DCKING\_NUM) which are not highlighted.

The [Results]-view of Ixanos is a desktop which can contain as many result screens as you need for your analysis. In the image below we can see two screens. One is maximized and visible and the other is minimized as a button in the lower left corner.



Many times during EIM-troubleshooting you will find yourself having ten to twenty such open results' windows that you wish to store so as to use them again next day. This can easily be done using Ixanos by utilizing the save desktop /load desktop functionality:



As I have personally worked in more than ten EIM projects in west Europe so far and was involved in companies across sectors (telco, banking, pharma, public sector, utilities, automotive), I have seen most of the issues that occur during EIM troubleshooting. All these issues have been dealt by Ixanos easily and made my troubleshooting experience a very smooth one. I hope this article sheds some light in this sometimes frustrating exercise and I hope to have also shown you how to make your task more pleasant using Ixanos.

Regards  
Ioannis Xanthopoulos  
[ix@ixanos.com](mailto:ix@ixanos.com)  
[www.ixanos.com](http://www.ixanos.com)