AE097\_SG Performance review

July 2014

Disclaimer

Please **refer to the Compliance Officer in your area for content** for this section, which **may be compulsory for legal and regulatory reasons** depending on the subject of the document and its target audience.

Table of Contents

[1. Overview 4](#_Toc392509931)

[2. the Performance test scenario & workaround: 5](#_Toc392509932)

[2.1. The baseline : 5](#_Toc392509933)

[2.2. the results : 5](#_Toc392509934)

[2.2.1 On the scheduling side 5](#_Toc392509935)

[2.2.2 On oracle side 5](#_Toc392509936)

[2.3. the AE097\_SG WORKAROUND: 6](#_Toc392509937)

[2.3.1 AE097\_SG SQL reviews: 6](#_Toc392509938)

[2.3.1.1. Oracle Objects CREATION: 6](#_Toc392509939)

[2.3.1.2. Existing SQL modification 7](#_Toc392509940)

[2.3.1.3. Existing SQL hints modification 12](#_Toc392509941)

[2.3.2 AE097\_SG Program review 12](#_Toc392509942)

[2.3.2.1. AE Steps modification 12](#_Toc392509943)

[2.3.2.2. Poluate the GTT table 12](#_Toc392509944)

[2.3.2.3. AE Peoplecode modification 13](#_Toc392509945)

[2.3.2.4. AE parameter review 14](#_Toc392509946)

[2.3.3 Other improvement areas: 16](#_Toc392509947)

# Overview

The aim of this document is to give a quick overview of the suggestions made by the performance team to improve time execution of RFI’s extractions (AE097\_SG).

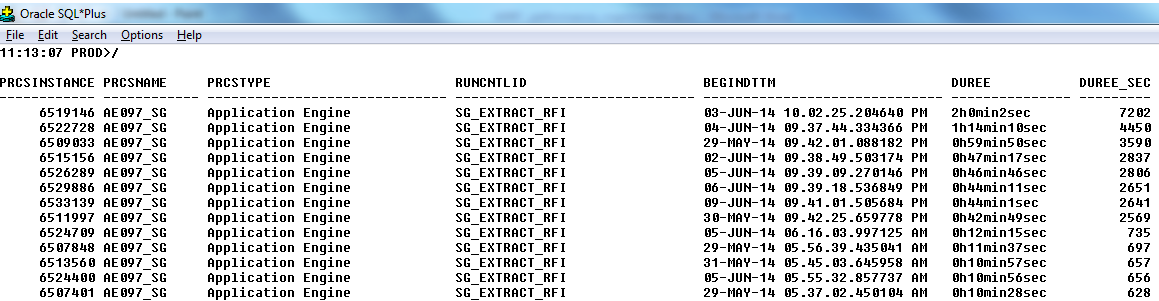
This is not specification document and does not intend to be a migration template. Development team should handle AE097\_SG performance suggestions and follow the whole migration process, based on SGCIB development rules and standards, before it’s effective on COE production environment.

# the Performance test scenario & workaround:

## The baseline :

In order to reproduce performance issue on AE097\_SG, Production snapshot was applied on Performance environment (OPARCOEP53) in order to run RFI Extraction for D+1, D+2 closure period.

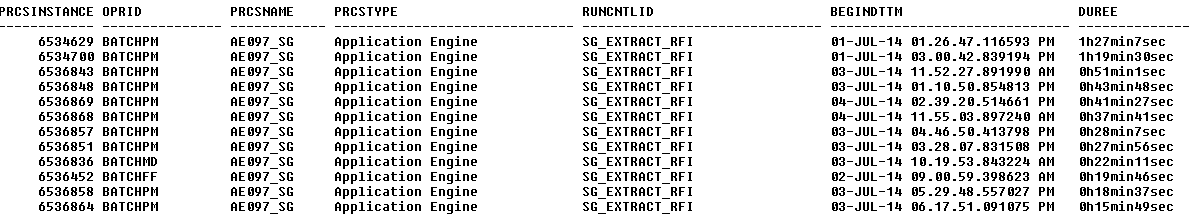
As an example, here are time executions for RFI Extraction for June 2014 closure



## the results :

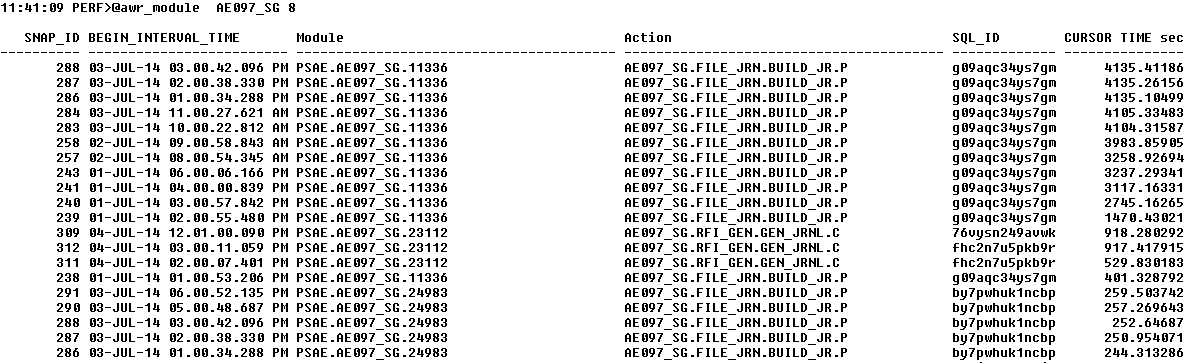
### On the scheduling side

The extraction below shows different AE097\_SG launches on OPACOEP53, along with improvement modifications :

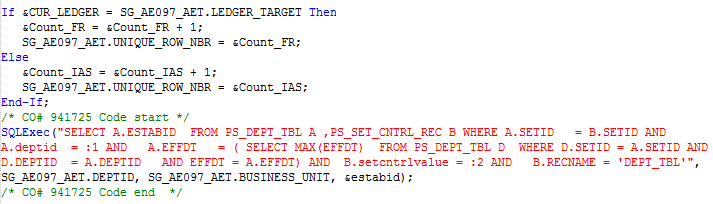


### On oracle side

As you can see below, the time spent by RFI Extraction on the Oracle side is on the step AE097\_SG.FILE\_JRN.BUILD\_JR (see below) :



The SQL is encapsulated in a SQLEXEC statement inside a Peoplecode (example Below) :



## the AE097\_SG WORKAROUND:

For each process execution, AE097\_SG is looping on the SQL\_EXEC inside a ‘Do Select’ statement in order to retrieve ESTABID information, which is harmful in terms of performance.

The global rework done is to remove this SQLEXEC call and retrieve the ESTABID information earlier in the program.

In order to make this modification effective, the steps below were necessary:

* SQL review :
  + New oracle objects Creation : table , index
  + Existing SQL modification
  + Existing SQL hint’s modification
* AE Program review :
  + AE Steps modification
  + AE Peoplecode modification
  + AE parameter review

### AE097\_SG SQL reviews:

### Oracle Objects CREATION:

In order to make the ESTABID columns information available to the AE097\_SG program, this information is retrieved and stored in an Oracle Global Temporary Table. Storing this information in a GTT table will make the table and data maintenance easier: data in the table is local to the session and Oracle creates as many GTT instances in it’s memory as AE097\_SG occurrences running.

Also, an index is necessary to fasten access to this GTT.

Below is the script to create GTT Table and it’s index:

CREATE GLOBAL TEMPORARY TABLE "COEPRD00"."PS\_AE097\_DEPT\_TMP\_SG"

( "SETCNTRLVALUE" VARCHAR2(20) NOT NULL ENABLE,

"DEPTID" VARCHAR2(10) NOT NULL ENABLE,

"ESTABID" VARCHAR2(12) NOT NULL ENABLE

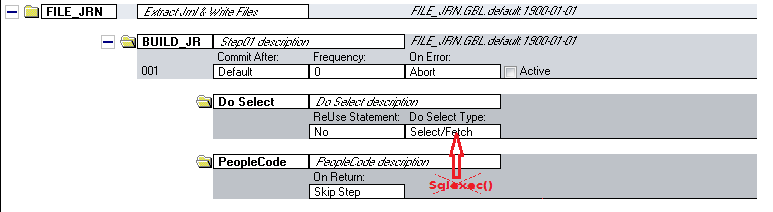
) ON COMMIT DELETE ROWS

CREATE INDEX "COEPRD00"."PS\_AE097\_DEPT\_TMP\_SG" ON "COEPRD00"."PS\_AE097\_DEPT\_TMP\_SG" ("DEPTID", "SETCNTRLVALUE")

### Existing SQL modification

As mentioned previousely, the SQLEXEC to remove is encapsulated inside a Peopesoft ‘Do Select’ statement.

The SQLEXEC will be removed and the ESATBID information will be captured in the ‘Do select’ (see example below)



PS:

The same logic is used on several step of the application Engine to retrieve the ESTABID information, so the same logic will be applied to remove the Sqlexec calls. Below is the list of the steps that need to be changed:

**AE097\_SG.FILE\_JRN.BUILD\_JR.D**

**AE097\_SG.FILE\_TRA.JRNL\_TRA.D**

**AE097\_SG.FILE\_TRA.BAL\_TRA.D**

**AE097\_SG.FILE\_OTH.BLD\_OTH.D**

AE097\_SG.FILE\_JRN.BUILD\_JR.D (After) :

%Select(JOURNAL\_LINE , DEPTID , OBJECT\_CODE , PRODUCT , FOREIGN\_AMOUNT , FOREIGN\_CURRENCY , ALTACCT , VALUE\_DT\_SG , BO\_ID\_SG , FO\_ID\_SG , OPERATION\_CD\_SG , PROD\_TYPE\_BO\_SG , CRE\_LOGIC\_ID\_SG , RDJ\_DT\_SG , SIGN, LEDGER,**ESTABID**)

SELECT **/\*+ ORDERED \*/** N.JOURNAL\_LINE

, N.DEPTID

, N.OBJECT\_CODE

, N.PRODUCT

, DECODE(SIGN(N.FOREIGN\_AMOUNT)

, -1

, N.FOREIGN\_AMOUNT\*-1000

, N.FOREIGN\_AMOUNT\*1000)

, N.FOREIGN\_CURRENCY

, DECODE(n.BUSINESS\_UNIT

,'G0001'

, N.ALTACCT

, N.ACCOUNT)

, %DateOut(N.VALUE\_DT\_SG)

, N.BO\_ID\_SG

, N.FO\_ID\_SG

, N.OPERATION\_CD\_SG

, N.PROD\_TYPE\_BO\_SG

, N.CRE\_LOGIC\_ID\_SG

, %DateOut(nvl(N.RDJ\_DT\_SG, N.JOURNAL\_DATE))

, DECODE(SIGN(N.FOREIGN\_AMOUNT)

, -1

, 1

, 0)

, N.LEDGER

**, TRIM(d.estabid)**

FROM **/\* PS\_JRNL\_HEADER a JOIN \*/** PS\_JRNL\_LN N **/\* ON ( N.BUSINESS\_UNIT = a.BUSINESS\_UNIT AND N.JOURNAL\_ID = a.JOURNAL\_ID AND N.JOURNAL\_DATE = a.JOURNAL\_DATE AND N.UNPOST\_SEQ = a.UNPOST\_SEQ AND a.JRNL\_HDR\_STATUS IN ('P' , 'U') ) \*/** JOIN PS\_RFI\_ACCOUNTS\_SG ACCT ON ( N.LEDGER = ACCT.LEDGER %Bind(ACCT\_ALT\_CLAUS\_SG, NOQUOTES)

AND ACCT.OPRID = %OperatorId

AND ACCT.RUN\_CNTL\_ID = %runcontrol )

**LEFT OUTER JOIN PS\_AE097\_DEPT\_TMP\_SG d**

**ON ( d.deptid = n.deptid AND d.setcntrlvalue = n.business\_unit )**

WHERE **n**.BUSINESS\_UNIT = %Bind(BUSINESS\_UNIT)

AND **n**.JOURNAL\_ID = %Bind(JOURNAL\_ID)

AND N.JOURNAL\_DATE = %Bind(JOURNAL\_DATE)

AND N.UNPOST\_SEQ = %Bind(UNPOST\_SEQ)

AND N.LEDGER %Bind(LEDGER\_LIST\_SG, NOQUOTES)

AND N.FOREIGN\_AMOUNT <> 0 %Bind(DEPT\_EXC\_CLAUS\_SG, NOQUOTES)

ORDER BY N.JOURNAL\_LINE

AE097\_SG.FILE\_TRA.JRNL\_TRA.D (After) :

%Select (ALTACCT, FOREIGN\_CURRENCY , RDJ\_DT\_SG, VALUE\_DT\_SG, FOREIGN\_AMOUNT, SIGN, **ESTABID**)

SELECT **/\*+ ORDERED \*/** DECODE(N.BUSINESS\_UNIT

,'G0001'

,DECODE( N.ALTACCT

, ' '

, N.ACCOUNT

, N.ALTACCT )

,N.ACCOUNT)

, N.FOREIGN\_CURRENCY

, %DateOut(N.RDJ\_DT\_SG)

, %DateOut(N.VALUE\_DT\_SG )

, SUM(N.FOREIGN\_AMOUNT)\*-1000

, DECODE(SIGN(FOREIGN\_AMOUNT)

, -1

, 1

, 0)

, **TRIM(d.estabid)**

FROM PS\_JRNL\_LN N

, PS\_RFI\_EXTRACT\_SG A

, PS\_RFI\_ACC\_TRF\_SG ACCT

**, PS\_AE097\_DEPT\_TMP\_SG D**

WHERE N.BUSINESS\_UNIT = %Bind (BUSINESS\_UNIT)

AND N.BUSINESS\_UNIT = A.BUSINESS\_UNIT

AND N.BUSINESS\_UNIT = ACCT.BUSINESS\_UNIT

AND N.LEDGER %Bind(LEDGER\_LIST\_SG, NOQUOTES)

AND ACCT.OPRID = %OperatorId

AND ACCT.RUN\_CNTL\_ID = %runcontrol

AND N.JOURNAL\_ID = A.JOURNAL\_ID

AND N.JOURNAL\_DATE = A.JOURNAL\_DATE

AND N.UNPOST\_SEQ = A.UNPOST\_SEQ %Bind(FROM\_DT\_CLAUSE\_SG, NOQUOTES)

AND (A.PROCESS\_DATE BETWEEN TO\_DATE('0101' || To\_Char(%Bind(AS\_OF\_DATE),'yyyy'),'DDMMYYYY') AND TRUNC(%Bind(DATE\_PROCESSED)))

AND A.PROCESS\_INSTANCE > 0 %Bind(ACCT\_ALT\_CLAUS\_SG, NOQUOTES)

AND N.DEPTID = %Bind(DEPTID)

AND N.FOREIGN\_AMOUNT <> 0

AND N.JOURNAL\_DATE >= %Bind(JOURNAL\_DATE\_FROM)

**AND N.BUSINESS\_UNIT = D.setcntrlvalue(+)**

**AND N.DEPTID = D.DEPTID(+)**

GROUP BY DECODE(N.BUSINESS\_UNIT ,'G0001',DECODE( N.ALTACCT , ' ' , N.ACCOUNT , N.ALTACCT ),N.ACCOUNT), N.FOREIGN\_CURRENCY, N.RDJ\_DT\_SG, N.VALUE\_DT\_SG, DECODE(SIGN(FOREIGN\_AMOUNT) , -1 , 1 , 0) **, TRIM(d.estabid)**

AE097\_SG.FILE\_TRA.BAL\_TRA.D (After):

%Select (CURRENCY\_CD , FISCAL\_YEAR, FOREIGN\_AMOUNT, DATE\_EXTRACT\_SG, ALTACCT, SIGN,**ESTABID**)

SELECT N.CURRENCY\_CD

, N.FISCAL\_YEAR

, SUM (YTD\_TRAN\_AMT) \* 1000

, %DateOut( To\_Date('3112' || To\_Char(To\_Number(To\_Char(To\_Date(%Bind(AS\_OF\_DATE) ,'YYYY-MM-DD') ,'yyyy'))-1) ,'DDMMYYYY'))

, DECODE(N.BUSINESS\_UNIT

,'G0001'

,N.ALTACCT

,N.ACCOUNT)

, DECODE(SIGN(SUM (YTD\_TRAN\_AMT) \* 1000)

, -1

, 1

, 0)

**, TRIM(d.estabid)**

FROM PS\_%Bind(RECNAME, NOQUOTES) N

, PS\_RFI\_ACC\_TRF\_SG ACCT

**, PS\_AE097\_DEPT\_TMP\_SG D**

WHERE N.BUSINESS\_UNIT = %Bind(BUSINESS\_UNIT)

AND N.BUSINESS\_UNIT = ACCT.BUSINESS\_UNIT

AND ACCT.OPRID = %OperatorId

AND ACCT.RUN\_CNTL\_ID = %runcontrol

AND N.LEDGER %Bind(LEDGER\_LIST\_SG, NOQUOTES) %Bind(ACCT\_ALT\_CLAUS\_SG, NOQUOTES)

AND N.DEPTID = %Bind(DEPTID)

AND N.FISCAL\_YEAR = To\_Number(To\_Char(%Bind(AS\_OF\_DATE),'yyyy'))-1

AND N.ACCOUNTING\_PERIOD = 12

**AND N.BUSINESS\_UNIT = D.setcntrlvalue(+)**

**AND N.DEPTID = D.DEPTID(+)**

GROUP BY N.CURRENCY\_CD, N.FISCAL\_YEAR ,DECODE(N.BUSINESS\_UNIT ,'G0001',N.ALTACCT,N.ACCOUNT), **TRIM(d.estabid)**

AE097\_SG.FILE\_OTH.BLD\_OTH.D (After) :

%Select(BUSINESS\_UNIT, ALTACCT, CURRENCY\_CD, DEPTID, YTD\_TRAN\_AMT, SIGN, LEDGER, **ESTABID**)

SELECT N.BUSINESS\_UNIT

, N.ACCOUNT

, N.CURRENCY\_CD

, N.DEPTID

, DECODE(SIGN(SUM(N.YTD\_TRAN\_AMT))

, -1

, SUM(N.YTD\_TRAN\_AMT)\*-1000

, SUM(N.YTD\_TRAN\_AMT)\*1000)

, DECODE(SIGN(SUM(N.YTD\_TRAN\_AMT))

, -1

, 1

, 0)

, N.LEDGER

**, TRIM(d.estabid)**

FROM PS\_%Bind(RECNAME, NOQUOTES) N

, PS\_RFI\_ACCOUNTS\_SG ACCT

**, PS\_AE097\_DEPT\_TMP\_SG D**

WHERE N.BUSINESS\_UNIT = %Bind(BUSINESS\_UNIT)

AND N.ACCOUNTING\_PERIOD = %Bind(ACCOUNTING\_PERIOD)

AND N.FISCAL\_YEAR= %Bind(FISCAL\_YEAR)

AND N.LEDGER %Bind(LEDGER\_LIST\_SG, NOQUOTES)

AND N.LEDGER = ACCT.LEDGER

AND ACCT.OPRID = %OperatorId

AND ACCT.RUN\_CNTL\_ID = %runcontrol

AND N.ACCOUNT = ACCT.ALTACCT %Bind(DEPT\_EXC\_CLAUS\_SG, NOQUOTES)

AND NOT (N.CURRENCY\_CD = BASE\_CURRENCY

AND N.YTD\_TRAN\_CR = 0

AND N.YTD\_TRAN\_DR = 0

AND N.YTD\_BASE\_AMT = 0 )

**AND N.BUSINESS\_UNIT = D.setcntrlvalue(+)**

**AND N.DEPTID = D.DEPTID(+)**

GROUP BY N.BUSINESS\_UNIT, N.ACCOUNT, N.CURRENCY\_CD, N.DEPTID , N.LEDGER , **TRIM(d.estabid)**

ORDER BY N.BUSINESS\_UNIT, N.ACCOUNT, N.CURRENCY\_CD, N.DEPTID, N.LEDGER

### Existing SQL hints modification

For the Step **AE097\_SG.FILE\_JRN.BUILD\_JR.D,** existing hint :

/\*+ PARALLEL(N 8) use\_hash(N a)\*/

Has been changed to

/\*+ ORDERED \*/

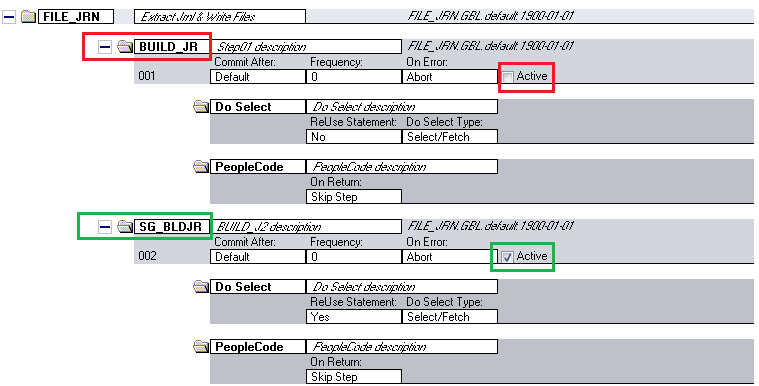
PS: please respect the order of the tables as written in the Step.

### AE097\_SG Program review

### AE Steps modification

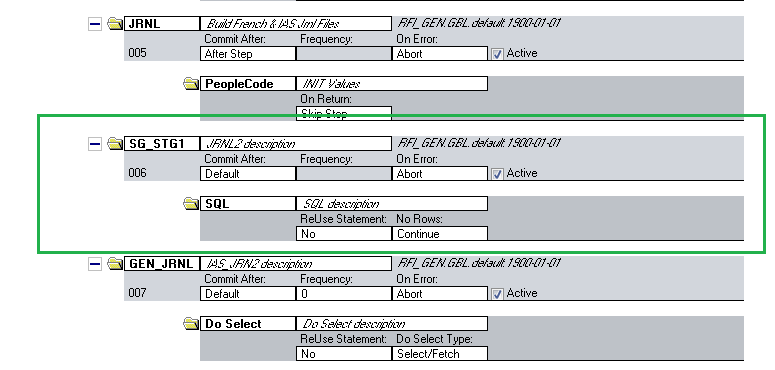
For readability purpose and fast switching between old / new code, AE steps that were modified were duplicated, and old steps where flagged as ‘inactive’. Also the new step are called SG\_...

below is an example:



### Poluate the GTT table

As mentioned at the beginning of the document, all the needed ESTABID values needed by the program are poplated in a GTT table. An AE step is added in order to populate this table? The step name is RFI\_GEN.SG\_STG1. It has been added just after step RFI\_GEN.JRNL.P ( see screenshot below)



### AE Peoplecode modification

As mentioned below the SQLExec inside the Peoplecode that retrieves ESTABID value is removed as the ESTABID is retrieved earlier in the ‘Do Select’ statement.

The modification is done on 4 Peoplecode steps :

**AE097\_SG.FILE\_JRN.BUILD\_JR.P**

**AE097\_SG.FILE\_TRA.JRNL\_TRA.P**

**AE097\_SG.FILE\_TRA.BAL\_TRA.P**

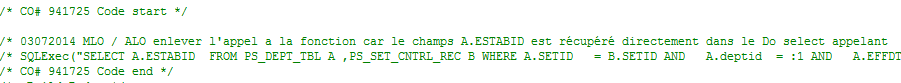
**AE097\_SG.FILE\_OTH.BLD\_OTH.P**

Example on the step **AE097\_SG.FILE\_JRN.BUILD\_JR.P**

**Before :**

****

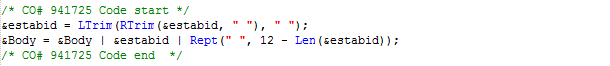
**After** :



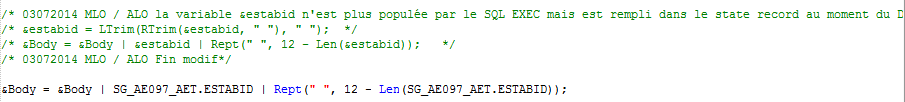
Also some operations done in the peoplecode on the value of the ESTBID, were done directly in the ‘Do select’ statement.

As an example the operation to retrieve left and right space in ESTABID, done in Peoplecode ( using syntax LTrim(RTrim(&estabid, " "), " "); ) are now done in during SQL selection .

Before:

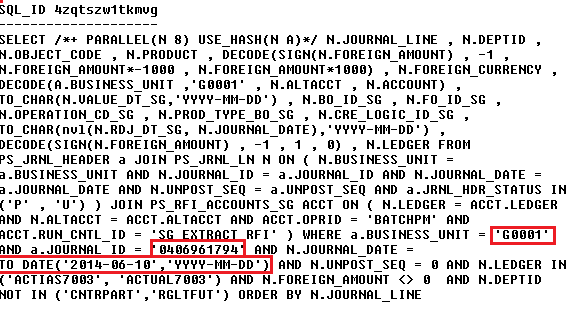


After:

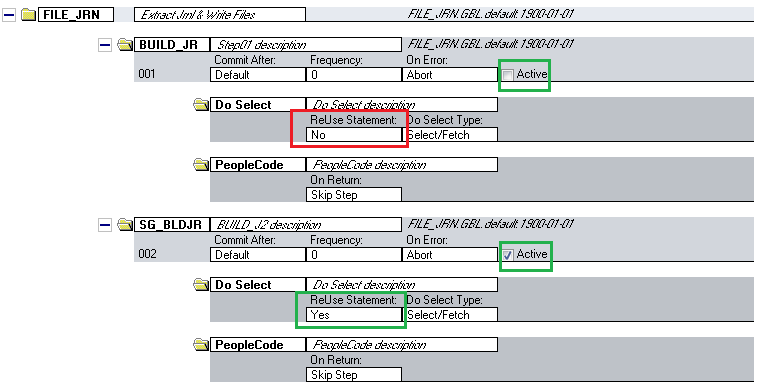


### AE parameter review

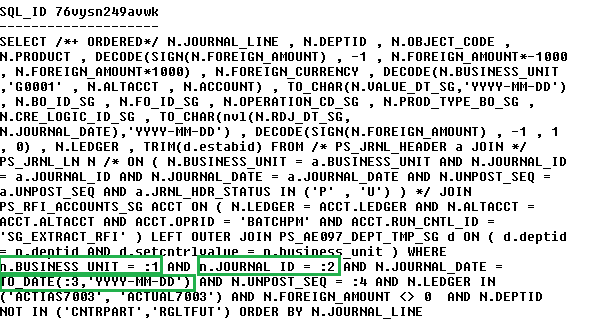
During performance test the step AE097\_SG.FILE\_JRN/BUILD\_JR.D is Called as many time as Journals to Extract. This step is containing litterals, so Oracle is generating as many exection plan as occurrence, causing overhead on Execution plan parsing and also, sometime, generating a bad execution plan.



The solution is to bind the SQL STATEMENT through the tools by setting ‘reuse statement’ to ‘Yes’ on the Step level ( see screenshots below )

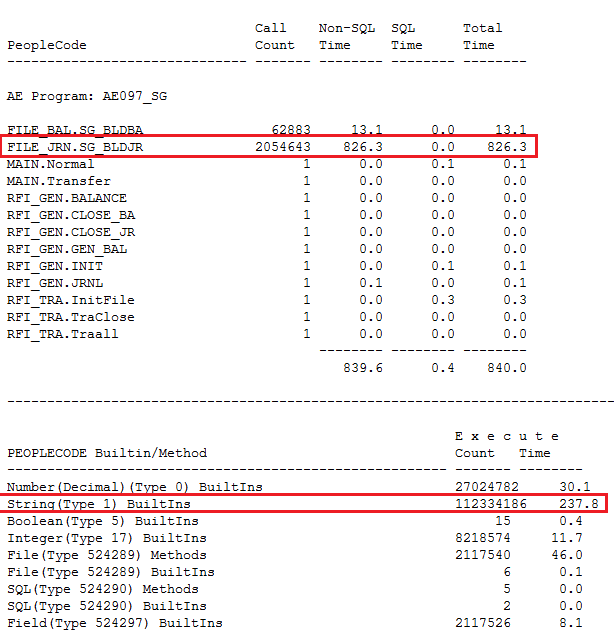


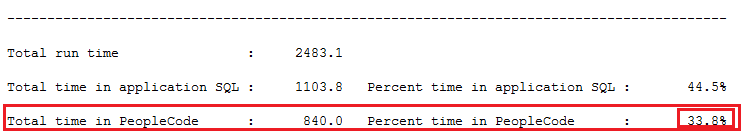
This make Peoplesoft generate SQL with Bind variable instead of hardcoded values. Below is the step as executed after on the database side :



### Other improvement areas:

During the performance tests, a significant AE097\_SG processing time is spent on on-SQL





On large files processing/extraction, Application Engine programs are not very efficient, also linear call of Peoplecode is time consuming. There are still some Peoplecode manipulating functions that can be removed and call earlier in the SQL statements. As an example, all the peoplecode Ltrim(Rtrim(… operation can be done during SQL extractions using the Oracle TRIM function

Below some extraction from the AE097\_SG Peoplecode :

&Body = LTrim(RTrim(&unique\_row\_nbr, " "), " ");

&foriegn\_cur = LTrim(RTrim(SG\_AE097\_AET.FOREIGN\_CURRENCY, " "), " ");

&Body = &Body | LTrim(RTrim(&sign, " "), " ");

&deptid = LTrim(RTrim(SG\_AE097\_AET.DEPTID, " "), " ");

&altacct = LTrim(RTrim(SG\_AE097\_AET.ALTACCT, " "), " ");

&estabid = LTrim(RTrim(&estabid, " "), " ");

Etc…