Creating Simple Inquiries and Using the EOD Report Generator

March 10, 2016

Purpose

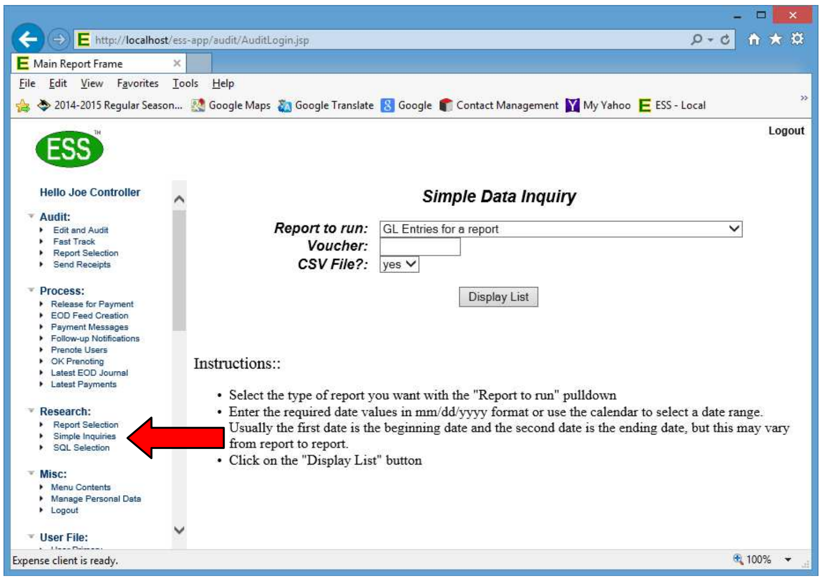
The purpose of this guide is three-fold:

1. Provide instructions and information for creating and running Simple Inquiries.
2. Provide instructions on how to create an EOD report batch using the EOD Report Generator to create a feed for another application.
3. Using a Simple Inquiry element as part of the EOD Report Generator for producing customized EOD feeds.

We will cover both the operational and technical aspects to the EOD Report Generation process. If you want to jump to the operations secton right away, go to page 12, " Operating the Report Generator EOD process"

Simple Inquiries

One of the benefits of an automated expense report process is the wealth of information that can be mined from the database. You can create custom reports to improve your process, enforce your travel policy or just provide you with information for decision making. While using a full-blown report writer that you are familiar with is preferable, ESS does come with a miniature report writing tool that you can use as an alternative. It is the "Simple Inquiries" option that is located in the Audit module.



Simple Data Inquiry

Report to run: GL Entries for a report
Voucher:
CSV File?: yes

Instructions::

- Select the type of report you want with the "Report to run" pulldown
- Enter the required date values in mm/dd/yyyy format or use the calendar to select a date range. Usually the first date is the beginning date and the second date is the ending date, but this may vary from report to report.
- Click on the "Display List" button

Expense client is ready.

Simple Inquiries are generated by combining paramenters passed into the *ReportGenerator.jsp* from *reportGenerator.html* with specifications defined in the *reports.xml* file.   You can create custom reports by modifying the *reports.xml* file and/or by using an alternative to *reportGenerator.html*.

The advantages of creating your SQL statements as simple inquiries are:

- They can be launched from the ESS menus with a simple menu selection.
- Different inquiries can be assigned for usage to different work functions. For example, a query that is OK for auditors to run may not be appropriate for a normal traveler.
- The inquiry can be parameterize with requested parameters specified as screen prompts when the query is selected from the menu.
- The "My World" function will segregate data by reporting lines based on the USER table.
- A summary option is available.
- CSV files are emailed to the person running the query.
- The queries can also be used to create an A/P and G/L posting feed.

Simple Inquiries are a powerful drill down into your expense report data.

Comma Separated File

A comma separated file (CSV) is a standard output format that many systems support for input and output. The user has the option of generating a CSV for any report they run by selecting "yes" to the "CSV File?" prompt. As part of the report, ESS emails the file to the user. Among other uses, a CSV file is importable into Excel.

Creating a simple data inquiry

To create a simple data inquiry, you need to modify the *reports.xml* file that is located with other configuration files in the *xmls/en* folder*

A simple data inquiry is basically an SQL statement that is run across the ESS database. Anything you can do with a single SQL statement can be done with a simple data inquiry. This includes utilizing the *union* operator and "selects on selects" so the queries can reach any level of complexity that you can handle. The query is specified in a query element in the main reports element in the *reports.xml* file. Here is an example of a query that lists out all the rows in the USER table:

```
<userlistAudit>
<title>Users for Database</title>
<pulldown>List of users</pulldown>
<sql>
SELECT PERS_NUM "Pers #" ,LNAME "Last", FNAME "First", EMAIL "E-mail",
MANAGER "Mgr #", COMPANY "Company" FROM USER ORDER BY PERS_NUM;
</sql>
<users>;AUDITOR;</users>
<myworldcheck>no</myworldcheck>
<usedates>no</usedates>
<field1name></field1name>
<field2name></field2name>
<ignoredaterange>Yes</ignoredaterange>
===== Ignore from here to the end ======
<sqlcheck1plus></sqlcheck1plus>
<sqlcheck2plus></sqlcheck2plus>
<sqlcheck3plus></sqlcheck3plus>
<sqlcheck1minus></sqlcheck1minus>
<sqlcheck2minus></sqlcheck2minus>
<sqlcheck3minus></sqlcheck3minus>
<encrypteditems></encrypteditems>
<datetypeoverride></datetypeoverride>
</userlistAudit>
```

Let's take a detailed look at the query:

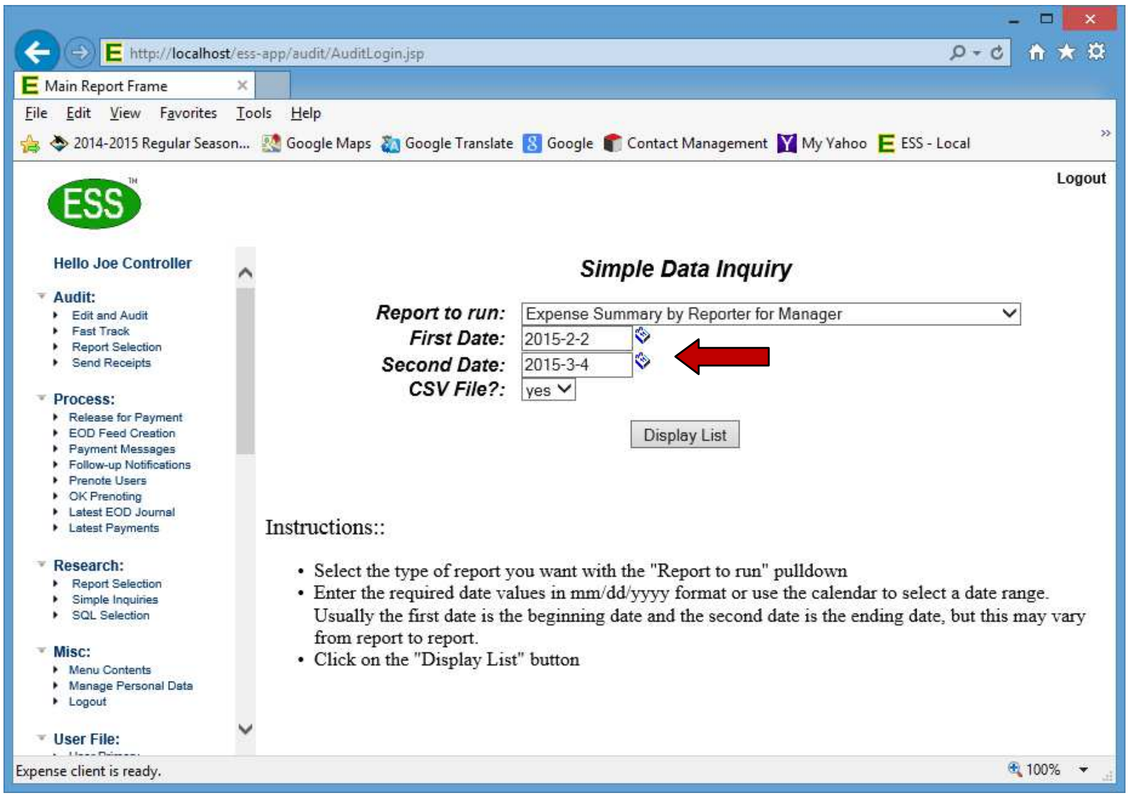
1. The query is specified in a element called "userlistAudit". Each query element must be unique.
2. The next five elements describe how the query is to execute:
   a. title - This is the title that will appear at the top of the report.
   b. pulldown - This is the description that will appear in the menu pulldown.
   c. sql - This is the SQL statement that will be executed. More about this in the next section.
   d. users - This controls who will be able to run this. This corresponds to the PRIMARY FUNCTION field in the User Primary screen.
   e. myworldcheck - if set to yes, only data from people who report to the person running the query (plus themselves) will be used to generate query

results. This is based on the REPORT and USER tables. Since this a report that will

f. usedates - This indicates if a data range should be displayed to the user. Since we are not using a date range in this query we will set this to a blank
g. field1name - The name of the first selection field. If blank, it will not be display.
h. field2name - same as above
i. ignoredaterange - This should be set to Yes for a simple query.
j. startatcolumn - Indicates which column to start printing from. Useful with the myworld function and summing. In simplest queries always set to 1.

3. The rest of the items should be blank for a simple query.

Let's take a look at another query that implements a date range. The query uses the period first (i.e., beginning) and second (i.e., ending) dates to let the user specify the period that the query covers.



Simple Data Inquiry

Report to run: Expense Summary by Reporter for Manager
First Date: 2015-2-2
Second Date: 2015-3-4
CSV File?: yes

Instructions::

- Select the type of report you want with the "Report to run" pulldown
- Enter the required date values in mm/dd/yyyy format or use the calendar to select a date range. Usually the first date is the beginning date and the second date is the ending date, but this may vary from report to report.
- Click on the "Display List" button

The date prompts are invoked with the *usedates* element that will add a prompt to the launch screen.

Here is the query:

```
<reports_processed>
  <title>Reports processed over the time period</title>
  <pulldown>Reports processed - enter beginning &amp; ending upload
dates</pulldown>
  <sql>
  SELECT VOUCHER, NAME, RE_AMT, UP_DATE FROM REPORT WHERE
REPORT.RP_STAT = 'H4' AND UP_DATE &gt;= $begdate$ AND UP_DATE &lt;=
$enddate$ ORDER BY VOUCHER;
  </sql>
  <usedates>yes</usedates>
  <users>;AUDITOR;</users>
  <myworldcheck>no</myworldcheck>
===== Ignore from here to the end ======
  <ignoredaterange>Yes</ignoredaterange>
  <datetypeoverride></datetypeoverride>
  <sqlcheck1plus></sqlcheck1plus>
  <sqlcheck2plus></sqlcheck2plus>
  <sqlcheck3plus></sqlcheck3plus>
  <sqlcheck1minus></sqlcheck1minus>
  <sqlcheck2minus></sqlcheck2minus>
  <sqlcheck3minus></sqlcheck3minus>
  <encrypteditems></encrypteditems>
  <startatcolumn>1</startatcolumn>
</reports_processed>
```

Here are the important declarations:

1. The query is specified in the "reports_processed" element.
2. The following elements drive the query:
   a. title - This is the title that will appear at the top of the report.
   b. pulldown - This is the description that will appear in the menu pulldown.
   c. sql - This is the SQL statement that will be executed. More about this in the next section.
   d. usedates - a 'yes' here means that dates will be prompted for and supplied in the *$begdate$* and *$enddate$* replacement parameters.
   e. users - This report is available to the basic function Auditor.
   f. myworldcheck - all reports will be showing regardless of reporting responsibilities.
3. The *$begdate$* and *$enddate$* macro handle all formating and quoting requirements based on the SQL setup of your database.
4. Notice the use of *&lt;* and *&gt;* for the less-than and greater-than conditions.

Common Parameters

Parameters, which in the ESS simple inquiries are easily recognized by the dollar sign nomenclature let you create queries that accept input from the user and from the system to let queries be created that are dynamic from run to run. This is a list of the most used parameters:

- $begdate$ - date specified by the user with the beginning date prompt. Formatted by the system to match the database. Does not need to be quoted.
- $enddate$ - date specified by the user with the ending date prompt. See above.
- $field1$ - value from the field1name user prompt.
- $field2$ - value from the field2name user prompt.
- $persnum$ - gets replaced by user's persnum.
- $email$ - gets replace by user's email address.
- $company$ - gets replace by the user's company number. Good for systems that have more than one company to make sure data from one company is not seen by another company.

With the exception of the dates, the above need to be quoted in the SQL statement if required.

Summary Queries

Summary queries have been deprecated from ESS 8.9 onward. Though they still function on ESS 8.9, it is recommended that the same queries be done via SQL union queries.

All Query Elements

A Simple Inquiry consists of a uniquely named element in *reports.xml* with the following sub-elements defined:

- **title** – enter the title of the report. The title will appear at the top of the report. Sometimes the *title* element will include a quick description of what the date fields mean.
- **pulldown** – text that will appear in the dropdown list.
- **sql** – enter the SQL statement that will be used to retrieve the basic rows and columns for the report. Any SQL statement supported by your database can be used. There are macros for beginning date and ending date, as well as, four general macros that will be explained below.
- **field1name** – Field name for selection/sort value. The value replaces $field1$ in the SQL sub-element.
- **field2name** – Field name for selection.sort value. The value replaces $field2$ in the SQL sub-element.
- **datetypeoverride** – Allows a different date type, from the one specified in the *system.xml SQL* element, to be used to check the date range. Example, the register table, store the *REPDATE* in Web format. Valid entries are 'YYYY-MM-DD' (MySQL, MS SQLServer) 'MM/DD/YYYY' (xBase), 'DD-MMM-YYYY' (Oracle), and 'WEB' (Web format). If left blank will use the format specified in *system.xml* which covers 99% of the cases as that corresponds to the date format used by the database.
- **users** – the users who will be able to run this report. This is based on the function in the user profile. Values must be seperated and boxed by semi--colons. Values are: ;AUDITOR;ADMIN;APPROVER;REPORTER;GL_ADMIN;
- **mywordcheck** – if *yes* the user will only see items for themselves and users who eventually report to them. A *no* will let the user see whatever the SQL statement returns.
- **startatcolumn** - Column which to start displaying, Lets you specify columns that will not be displayed. Principally used for the myworld check to eliminate the first column from being displayed.
- **usedates** - Controls the beginning and ending date prompts on the Simple Inquiry select screen. A *Yes* will cause the prompts to be displayed

A Simple Inquiry element may also used the following sub-elements which have been deprecated from ESS 8.9 onward:

- sqlcheck1plus – A secondary SQL statement that must evaluate as true for the row to be included on the report. Not required. Information from result rows, columns one and two, returned by the SQL statement, specified in the *sql* element, is passed into this statement. If blank, this check is not performed and the condition is true. [deprecated]
- sqlcheck2plus – see above. [deprecated]
- sqlcheck3plus – see above. [deprecated]
- sqlcheck1minus – A secondary SQL statement that must evaluate as true for the row to be included on the report. Not required. Information from result rows, columns one and two, returned by the SQL statement, specified in the *sql* element, is passed into this statement. . If blank, this check is not performed and the condition is true. [deprecated]
- sqlcheck2minus – see above. [deprecated]
- sqlcheck3minus – see above. [deprecated]
- encrypteditems – List of items in the SQL statement that are scrambled with the key specified in the SQL select of the *system.xml* file. At most sites this will be blank. [Deprecated]
- ignoredaterange – A “Yes” ignores the date range passed into the ReportGenerator.jsp from the HTML screen. A ‘No’ allows the first column not to be the date that is checked to be between a range. [deprecated]
- primarytitle – Title of the first column on a summary report. [deprecated]
- secondary1title – Title of the second column on a summary report. [deprecated]
- secondary2title – Title of the third column on a summary report. [deprecated]
- amounttitle – Title of the amount column on a summary report. [deprecated]
- summaryreport – A *yes* will aggregate data based on the first three columns and provide a report total. All summary reports must have four columns. If you don't need three aggregate columns, the SQL statement should supply a blank column. [deprecated]

The REGISTER Table

If you write a query against the REGISTER table, you will need to use the *datetypeoverride* element if you are using the REPDATE column. Here is an example from the "List of Register Items" query:

```
<register>
  <title>List of Register Items</title>
  <pulldown>List of register items</pulldown>
  <sql>
  SELECT REPDATE, RTRIM(REFERENCE) "Ref #" ,RTRIM(OWNER) "Owner",
RTRIM(STATUS) "Status", REPDATE "Date" FROM REGISTER ORDER BY REFERENCE;
  </sql>
  <ignoredaterange>No</ignoredaterange>
  <datetypeoverride>WEB</datetypeoverride>
  <users>;AUDITOR;</users>
  <myworldcheck>no</myworldcheck>
'.......
</register>
```

The query will list out the REGISTER for a period of time and is available to an auditor.

The "My World" Check

The My World check is available for usage when a site uses the manager column in the user table to indicate who the reporters manager is. This lets the system determine all the reporters for a specific individual by creating an organization chart underneath that person. To use the My World check, see the *myworld* sub-element text to *Yes*.

All Parameters

The following Parameters are available to the SQL statement defined in the *sql* element:

- $begdate$ - Used to prep the SQL with the date format string specified in the *sql* element in the *system.xml* file. The date format string generally carries the $date$ macro. Begin and end date integrity are maintain with execution presidence.
- $enddate$ - Used to prep the SQL with the date format string specified in the *sql* element in the *system.xml* file. The date format string generally carries the $date$ macro. Begin and end date integrity are maintain with execution presidence.
- $date$ - Used internally. If used directly by an inquiry will resolve as the ending date string.
- $begdatestr$ - beginning date in MM/DD/YYYY format. (use $begdate$)
- $enddatestr$ - ending date in MM/DD/YYYY format. (use $enddate$
- $begdatesql$ - beginning date in YYYY-MM-DD format. (use $begdate$)
- $enddatesql$ - ending date in YYYY-MM-DD format. (use $enddate$)
- $field1$ - value from the field1name entry.
- $field2$ - value from the field2name entry.
- $persnum$ - gets replaced by user's persnum.
- $email$ - gets replaced by user's email address.
- $company$ - gets replaced by the user's company number. Good for systems that have more than one company to make sure data from one company is not seen by another company.

Normally the *report.xml* is located in the xmls language folder. However, since it is specified by a sub-element in the configuration element of the *system.xml* folder, it can be moved to an area that is accessible to alternative personnel.

The default *reports.xml* comes with several example inquiries.

EOD Report Generator and Posting File

ESS provides two methods of creating files that will update systems such as your A/P, G/L, banking, project tracking, etc. systems. They are:

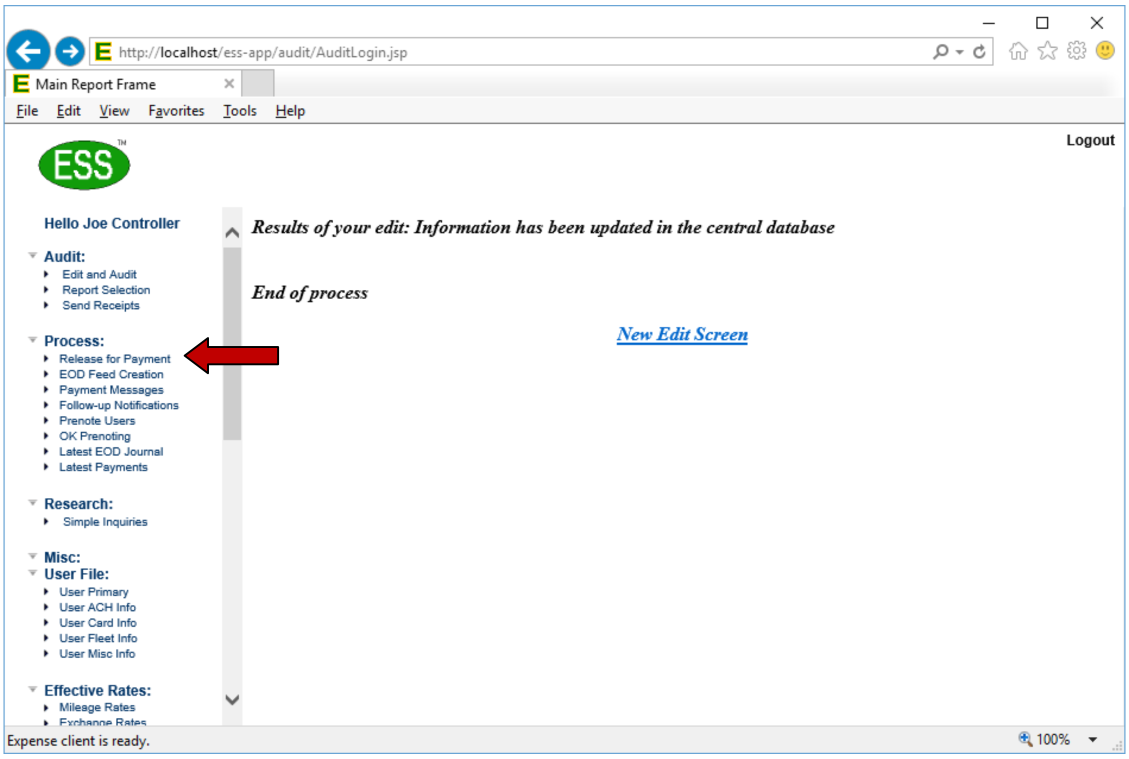
- EOD Feed Creation Process - This used to create feeds that have a level of complexity to them, such as, NACHA, SAP, QuickBooks, etc. where there is not always a one to one correspondence between the system entry and the data elements in ESS. ESS comes standard with the NACHA feed. It also comes standard with a generic A/P and a generic G/L feed. Other feeds are customized to specific processing requirements. This type of feed can be created by programming, in Java, and implements the *ess.FeedAPI.class*. To run the feed as part of the EOD File Creation process, you need to create a *feed* sub-element in the *endofday* element in the system.xml file with the full class name as the text (e.g., *essNACHAFeed*). Contact *service@expenseservices.com* for more information on creating this type of feed.

- Report Generator Feed (RGF) - This can be used when a one to one mapping opportunity exists between the elements in the ESS database and the information required by the receiving system can be delivered in a CSV format. This feed uses a Simply Inquiry definition in the report.xml file to create a list of entries for an expense report.

The rest of this document deals with running and creating your own feed with the RGF.

Operating the Report Generator EOD process

To actually create a feed file start by accessing the "Release for Payment" menu option.

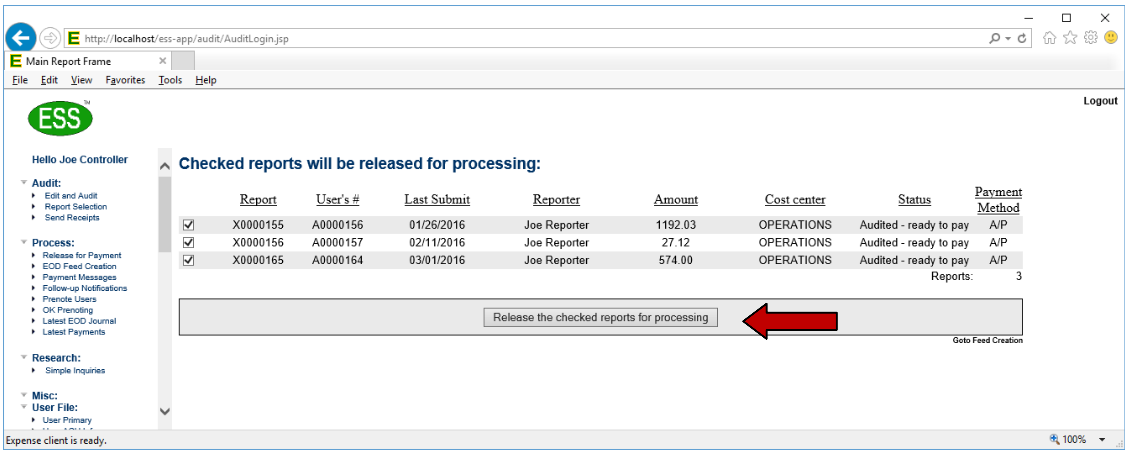


Results of your edit: Information has been updated in the central database

End of process

Expense client is ready.

The "Release for Payment" option is used to create an End of Day (EOD) batch. An EOD batch is a collection of reports that have been promoted to the paid (i.e., H4) and have the same upload date (i.e., UP_DATE).

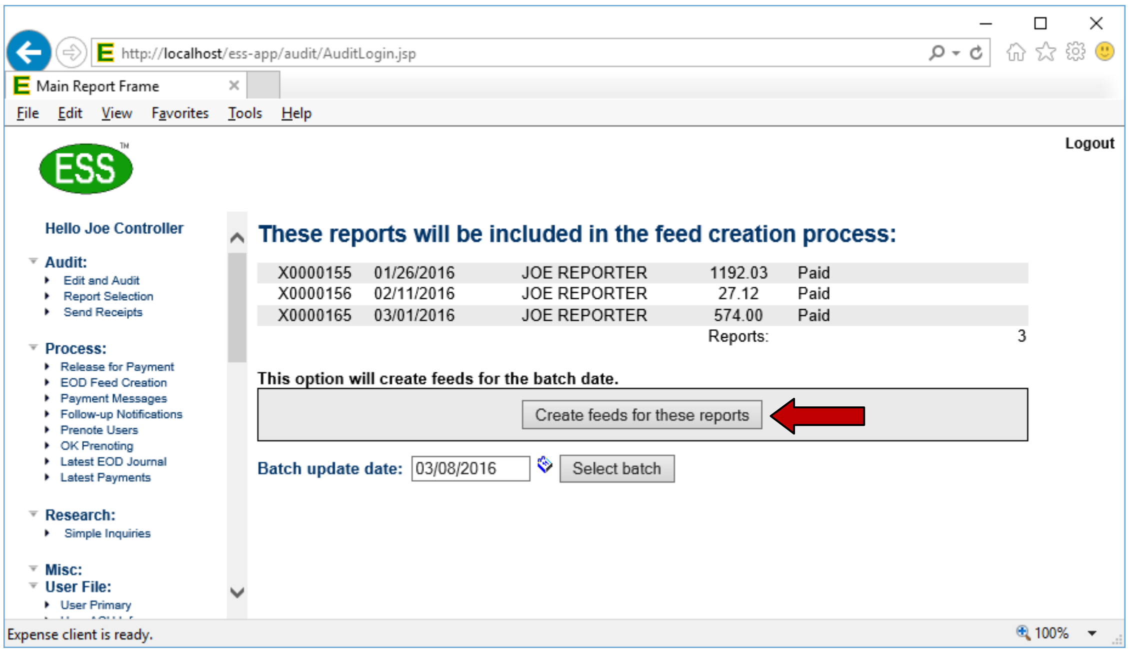


Checked reports will be released for processing:

	Report	User's #	Last Submit	Reporter	Amount	Cost center	Status	Payment Method
✓	X0000155	A0000156	01/26/2016	Joe Reporter	1192.03	OPERATIONS	Audited - ready to pay	A/P
✓	X0000156	A0000157	02/11/2016	Joe Reporter	27.12	OPERATIONS	Audited - ready to pay	A/P
✓	X0000165	A0000164	03/01/2016	Joe Reporter	574.00	OPERATIONS	Audited - ready to pay	A/P
							Reports:	3

Expense client is ready.

Reports that are eligible to be batched on the EOD feed are presented with a check box. If you do not want to include the report in the EOD batch, uncheck it. To batch the checked reports, click on the "Release the check reports for processing" button.



These reports will be included in the feed creation process:

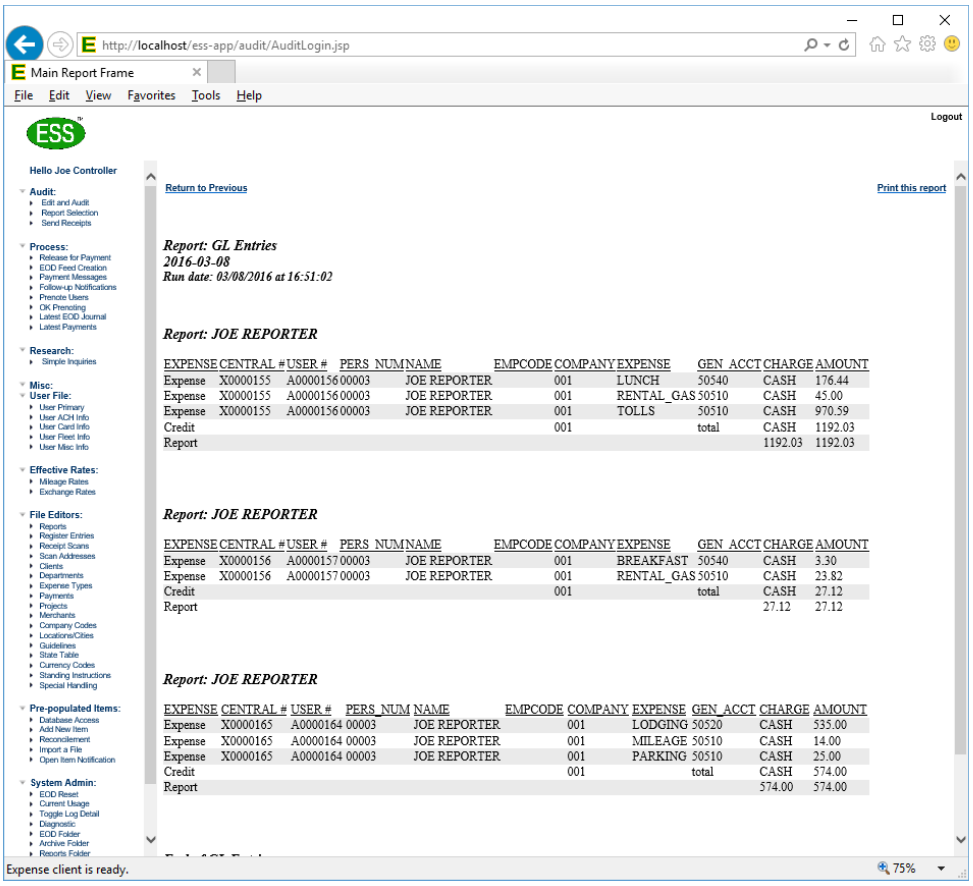
X0000155	01/26/2016	JOE REPORTER	1192.03	Paid
X0000156	02/11/2016	JOE REPORTER	27.12	Paid
X0000165	03/01/2016	JOE REPORTER	574.00	Paid
			Reports:	3

This option will create feeds for the batch date.

Batch update date: 03/08/2016

After the reports have been batched, the process will chain to the feed creation screen. To create feeds, click on the "Create feeds for these reports" button. If for some reason you wish to create a feed from a previous day's batch, you can use the "Select batch" button at this point.

At this point ESS will run a SQL select, against each report, to produce a report that can be used to manually enter information to a receiving application, such as your A/P system.



Report: GL Entries
2016-03-08
Run date: 03/08/2016 at 16:51:02

Report: JOE REPORTER

EXPENSE	CENTRAL #	USER #	PERS_NUM	NAME	EMPCODE	COMPANY	EXPENSE	GEN_ACCT	CHARGE	AMOUNT
Expense	X0000155	A0000156	00003	JOE REPORTER		001	LUNCH	50540	CASH	176.44
Expense	X0000155	A0000156	00003	JOE REPORTER		001	RENTAL_GAS	50510	CASH	45.00
Expense	X0000155	A0000156	00003	JOE REPORTER		001	TOLLS	50510	CASH	970.59
Credit						001		total	CASH	1192.03
Report									1192.03	1192.03

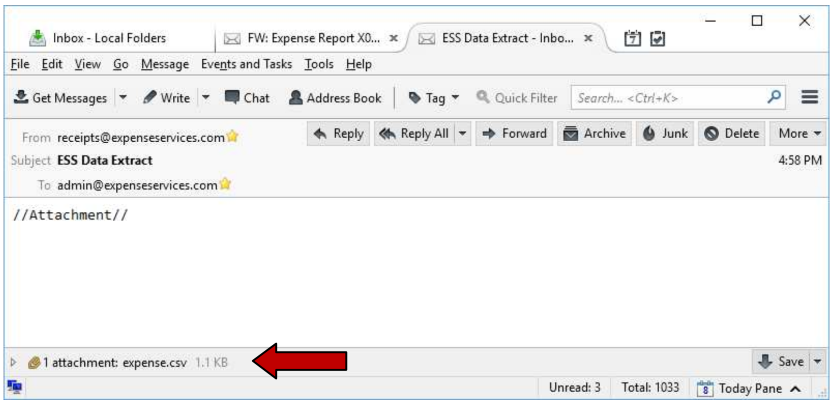
Report: JOE REPORTER

EXPENSE	CENTRAL #	USER #	PERS_NUM	NAME	EMPCODE	COMPANY	EXPENSE	GEN_ACCT	CHARGE	AMOUNT
Expense	X0000156	A0000157	00003	JOE REPORTER		001	BREAKFAST	50540	CASH	3.30
Expense	X0000156	A0000157	00003	JOE REPORTER		001	RENTAL_GAS	50510	CASH	23.82
Credit						001		total	CASH	27.12
Report									27.12	27.12

Report: JOE REPORTER

EXPENSE	CENTRAL #	USER #	PERS_NUM	NAME	EMPCODE	COMPANY	EXPENSE	GEN_ACCT	CHARGE	AMOUNT
Expense	X0000165	A0000164	00003	JOE REPORTER		001	LODGING	50520	CASH	535.00
Expense	X0000165	A0000164	00003	JOE REPORTER		001	MILEAGE	50510	CASH	14.00
Expense	X0000165	A0000164	00003	JOE REPORTER		001	PARKING	50510	CASH	25.00
Credit						001		total	CASH	574.00
Report									574.00	574.00

The report can be printed by using the link at the top right corner of the report display. Normally the feed will also produce a CSV of the information on the post. It is emailed to the email address of the logged in account. It can also be found in the /var/ess/reports folder under the personnel number of the person who ran it (e.g., 99999.csv).



From receipts@expenseservices.com
Subject **ESS Data Extract**
To admin@expenseservices.com
4:58 PM

```
//Attachment//
```

1 attachment: expense.csv 1.1 KB

Defining the process

In order to use a Simple Inquiry you need to do several things. These are:

- Let ESS know you will use the Report Generator file creation process.
- Let ESS know what Simply Inquiry to use.
- Let ESS know which reports to include.
- Define the Simple Inquiry to use.

This is a very straightforward process.

Set up the *system.xml*

To use the RGF, you will need to change two sub-elements and add two sub-elements to the *endofday* element in the *system.xml* file.

To let ESS know to run the Report Generator file creation process instead of a customized feed programmed in Java, change these two sub-elements:

- **feedjsp** - Change the text to *eod/EODReportGenerator.jsp.*
- **feedoption** - Change the text to *parent.contents.PersWithDBase(parent.contents.defaultApps + 'eod/EODList.jsp?downlevel=','approvallevel','1')*

Additionally, the following two sub-elements need to be added to the *endofday* element:

- **postingreport** - This sub-element specifies the Simple Inquiry to run in the *reporttype* attribute and whether to create a CSV file in the *createfile* attribute.
- **postthesereports** - The text contains the SQL to execute to produce a list of reports to have the Simple Inquiry run against. The $date$ macro will be replaced with the date specified in the feed creation screen.

Below is an example of how the sub-elements look:

```
<endofday>
....
  <feedjsp>eod/EODReportGenerator.jsp</feedjsp>
  <feedoption>parent.contents.PersWithDBase(parent.contents.defaultApps +
eod/EODList.jsp?downlevel=','approvallevel','1')</feedoption>
  <postingreport reporttype="glentry" createfile="yes" />
  <postthesereports>SELECT VOUCHER FROM REPORT WHERE UP_DATE = '$date$' AND
RP_STAT =H4';</postthesereports>
</endofday>
```

This example will create a list of reports with paid start (i.e.,) that where released for payment on the date that generated the list on the create feed screen.

Simple Inquiry for RGF

The RGF uses a Simple Inquiry, defined in the report.xml file, to create the detail entries for each report that is in the feed. Here is an example of an actual feed that summarizes entries based on a departmnet, expense type and sales code, It also generates an offsetting payment entry:

```
<glentry>
  <title>AP Entries</title>
  <pulldown>AP Entries for a report</pulldown>
  <sql>
select 'Expense',max(ex.VOUCHER) 'Central #', max(rp.PERS_NUM) 'AP #', max(rp.COMPANY)
'Company #',dp.GL_TERM + ac.GEN_ACCT + us.EMPCODE 'ACCOUNT', sum(ex.AMOUNT)
'AMOUNT' from EXPENSE as ex join REPORT as rp on ex.VOUCHER = rp.VOUCHER join
RECEIPT as rc on ex.VOUCHER = rc.VOUCHER and ex.RECEIPT = rc.RECEIPT join DEPART as
dp on rp.DEPART = dp.DEPART join COMPANY as co on rp.COMPANY = co.COMPANY join
ACCOUNT as ac on ex.EXPENSE = ac.EXPENSE and co.EXPENSE = ac.COMPANY join
EXPUSER as us on rp.PERS_NUM = us.PERS_NUM where ex.VOUCHER = '$field1$' group by
dp.GL_TERM, ac.GEN_ACCT, us.EMPCODE union select 'Payment','','', '', '020001',
sum(ex.AMOUNT) from EXPENSE as ex join REPORT as rp on ex.VOUCHER = rp.VOUCHER join
RECEIPT as rc on ex.VOUCHER = rc.VOUCHER and ex.RECEIPT = rc.RECEIPT join COMPANY
as co on rp.COMPANY = co.COMPANY join ACCOUNT as ac on ex.EXPENSE = ac.EXPENSE and
co.EXPENSE = ac.COMPANY join EXPUSER as us on rp.PERS_NUM = us.PERS_NUM where
ex.VOUCHER = '$field1$' group by rc.CHARGE;
  </sql>
  .....
  <ignoredaterange>Yes</ignoredaterange>
  <datetypeoverride></datetypeoverride>
  <users>;AUDITOR;</users>
  <myworldcheck>no</myworldcheck>
  <field1name>Voucher</field1name>
  <usedates>no</usedates>
    <startatcolumn>1</startatcolumn>
</glentry>
```

The above Simple Inquiry will produce a report that looks like the one below:

Report: AP Entries
2016-02-15
Run date: 03/08/2016 at 19:10:38

Report: Joe Controller

	CENTRAL #	AP #	COMPANY #	ACCOUNT	AMOUNT
Expense	X0000170	99999	002	00408031	103.87
Expense	X0000170	99999	002	00408032	15.00
Expense	X0000170	99999	002	00408150	15.00
Payment				020001	133.87

Report: Tom Jones

	CENTRAL #	AP #	COMPANY #	ACCOUNT	AMOUNT
Expense	X0000171	00001	002	004380312	84.54
Expense	X0000171	00001	002	004380322	998.66
Expense	X0000171	00001	002	004380332	588.00
Expense	X0000171	00001	002	004381602	45.00
Payment				020001	1716.20

End of AP Entries

In addition, if the *createfile* attribute is set to "Yes" in the *postingreport* sub-element, a file with the contents like the one below will be produced and email to the person running the process:

expense.csv

```
Expense,X0000170,99999,002,00408031,103.87
Expense,X0000170,99999,002,00408032,15.00
Expense,X0000170,99999,002,00408150,15.00
Payment,,,,,020001,133.87
Expense,X0000171,00001,002,004380312,84.54
Expense,X0000171,00001,002,004380322,998.66
Expense,X0000171,00001,002,004380332,588.00
Expense,X0000171,00001,002,004381602,45.00
Payment,,,,,020001,1716.20
```

If you need to quote strings, you can use the SQL *concat* operator, or its equivalent for those fields.

The *$field1$* macro

The *$field1$* macro is used by the Report Generator process to passing the report's reference number (i.e., VOUCHER) in to the Simple Inquiry being used. You notice in the above example a portion of the SQL:

```
...where ex.VOUCHER = '$field1$'...
```

In the first report of the above example, this gets expanded to:

```
...where ex.VOUCHER = 'X0000171'...
```

This is how the Report Generator know which report to select.

Getting Support

Please contact us at *service@expenseservices.com* with any questions.

###