Outsourcer 5.1.2
Administrator Guide

http://www.PivotalGuru.com

Document Date 2016-04-27

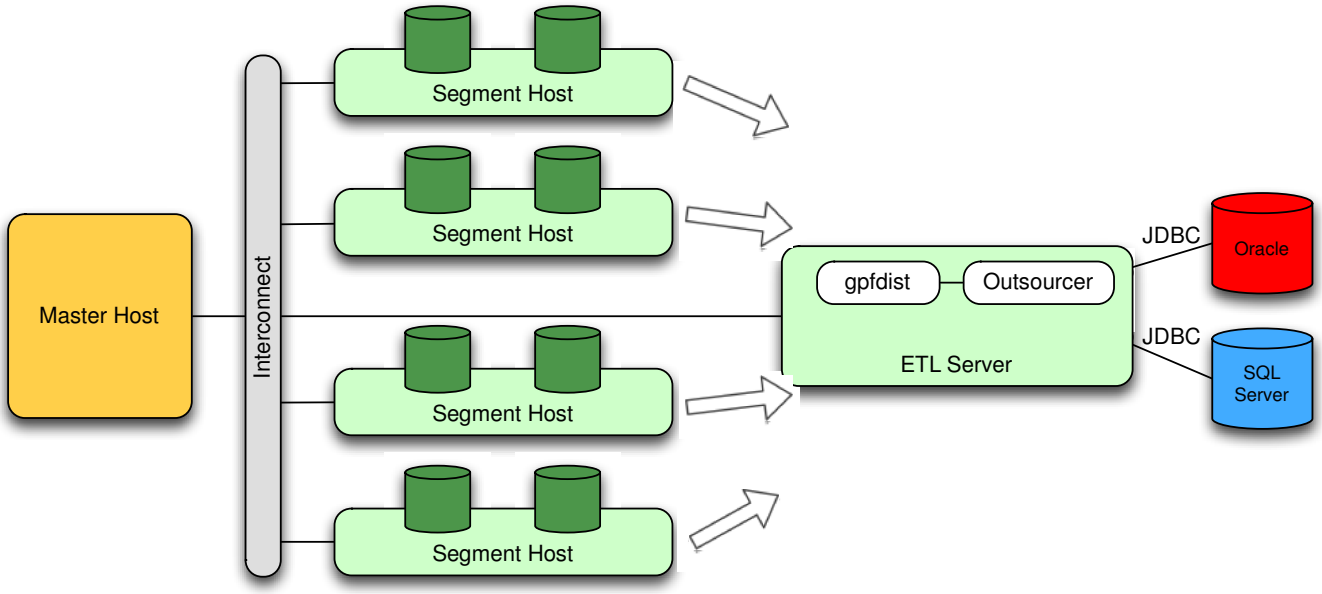
Overview of Outsourcer

Outsourcer is an open source Java program that automates all of the tasks needed to load data into Greenplum from SQL Server and Oracle by leveraging Java and Greenplum's External Tables. It automates everything for you. You simply define the source to target mapping in a "Job", and then submit the Job into a "Queue" to be processed. It can be done completely from a web browser so no coding is needed.

How it Works

Typical data loading in Greenplum and HAWQ uses files server with a gpfdist process through gpload or through a user defined External Table. Outsourcer also uses this process but instead of creating files, gpfdist executes an Outsourcer Java program to connect and get data from Oracle and SQL Server.

How it Works Illustration



The image above depicts Outsourcer installed on a dedicated ETL server but Outsourcer can be installed on any server in which the Segment Hosts / Data Nodes have network connectivity and name resolution to in addition to network connectivity and name resolution to the Oracle / SQL Server source databases. You can install Outsourcer on the Master or Standby-Master host as well.

Main Components

Jobs

Outsourcer defines a Source to Target mapping as a "Job". It contains all of the information about the Source Oracle or SQL Server database as well as the target table in Greenplum or Hawq. A job is defined in a table in Greenplum or Hawq and can inserted directly or through the User Interface.

Queue

Jobs are processed in a Queue by the Queue Date column. Jobs can be submitted to the Queue through a function call (os.fn_queue(?)), through the User Interface, or by a recurring schedule defined in Outsourcer. The number of concurrent jobs is determined by the os.variables value of “max_jobs”. The default is 4 but you can increase or decrease this value to optimize the throughput of lading data.

Custom Tables

Outsourcer allows you to create a persistent External Table to SQL Server or Oracle with your own custom SQL statement. The User Interface limits you to 50 columns but you can insert into os.ao_custom_sql and specify more columns if you like. If you do this, be sure to restart Outsourcer so that you get a gpfdist assigned to your table.

Daemons

Queue Daemon

Execute the “osstart” command to start the background process that watches the Queue. Submit jobs to the queue with os.fn_queue(?), os_fn_queue_all(), or by recurring schedule. The ossstart command will also start the gpfdist processes needed for the Custom SQL tables you may have defined.

The Queue Daemon must be running for Jobs to be processed in the Queue and this process also starts the gpfdist processes needed by any Custom SQL tables you create.

“osstop” stops the Queue as well as Custom SQL tables’ gpfdist processes.

Scheduler Daemon

Execute the “agentstart” command to start the Scheduler. This process will automatically insert jobs into the queue based on the schedule you have defined. Schedules are defined in os.schedule and the schedule description field in the Job. This process is optional and only needed if you are scheduling recurring jobs with Outsourcer. “agentstop” stops the scheduler.

UI Daemon

Execute the “uistart” command to start the User Interface. This will also start the gpfdist process that is needed for the User Interface. The User Interface is optional and not required for jobs to be processing in the Queue. “uistop” stops the UI.

Starting All

start_all and stop_all start all Daemons listed above.

Installation

Prerequisites

1. Outsourcer host must have network connectivity to all Segment Hosts (Greenplum) / Data Nodes (HAWQ).
2. Name resolution from the Segment Hosts / Data Nodes to the Outsourcer host.

3. Greenplum / HAWQ account with Super User rights (gpadmin).
4. SQL Server and/or Oracle account that uses password authentication.
5. Client connectivity from Outsourcer host to the Master host.
6. pg_hba.conf configuration to allow connections from Outsourcer host with some form of password authentication (LDAP, MD5, password, etc).
7. Greenplum / HAWQ loaders client with gpfdist and psql.
8. Oracle JDBC driver. Note: requires downloading manually from Oracle's site in order to accept their license agreement.
9. Microsoft JDBC driver. Note: automatically downloaded if there is an Internet connection available.
10. Date and time must be in sync between the Outsource host and Greenplum / HAWQ.

Upgrading from 4.x or 3.x

1. Stop all Outsourcer jobs and background processes.
2. Installation is recommended on the Standby-Master or dedicated ETL server.
3. If installing 5.x on the Master, edit the .bashrc file owned by gpadmin and REMOVE the following lines:

```
source /usr/local/os/os_path.sh
export AUTHSERVER=x.x.x.x
export UIPORT=xxxx
```

4. Exit the shell and connect again.

Installation Process

1. Download the `x_x_x.zip` file from http://www.pivotalguru.com, transfer it to any host that is accessible to all Segment Hosts in Greenplum or all Data Nodes in HAWQ.
2. As user gpadmin (or any valid Unix account that meets the prerequisites), unzip the file to the installation directory (example: /usr/local/os)
3. Download the Oracle JDBC driver from http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010-090769.html and place the file in the jar/ directory. This is a manual step because Oracle requires you accept their license agreement.
4. If Internet access is not available on the Outsourcer host, download the Microsoft SQL Server JDBC driver from http://download.microsoft.com/download/0/2/A/02AAE597-3865-456C-AE7F-613F99F850A8/sqljdbc_4.0.2206.100_enu.tar.gz and extract the file. Then move sqljdbc_4.0/enu/sqljdbc4.jar to the jar/ directory.
5. As user gpadmin, execute:

```
./os_install.sh
```

6. Follow the installer prompts to complete the installation.
7. After the installation is complete, source your .bashrc file.

```
source ~/.bashrc
```

Configuration Files

config.properties

Connection information used by Outsourcer to connect to Greenplum / HAWQ used by Java programs. This is build during the installation.

.pgpass

Connection information used by Outsourcer to connect to Greenplum / HAWQ used by psql during the installation. The gpmon entry is retained and there is a backup file created.

os_path.sh

Contains the environment variables used by Outsourcer. You can edit this file to make environmental changes but please use caution when doing so. Some variables require re-installation and are noted in the file.

Changing any values in the os_path.sh file should be done with Outsourcer fully stopped. Some variables require a re-installation and others should not be altered.

.bashrc

This is updated during the installation to add an entry for sourcing the os_path.sh file. A backup is also created during the installation.

Jobs

Refresh Job Type

A Refresh Job refreshes data in Greenplum / HAWQ from the source.

Refresh Process

Outsourcer Refresh Jobs do the following:

1. Connects to the Source and checks that it is valid
2. Creates Target schema if not found
3. Creates Target table if not found based on DDL from Source
4. Starts gpfdist process
5. Creates External Table
6. Truncates Target table
7. Insert all data from Source to Target
8. Stops gpfdist process
9. Executes any SQL found in the SQL Text field. *Note: See Transform job for more details.*

Append Job Type

An Append Job is similar to a Refresh Job but it also has a feature of only getting new data. This is only useful for tables that never have DELETE or UPDATE statements executed on in the Source. A table that contains Web Logs is a good example where an Append Job would be useful.

It is recommended to use a numeric value for the column to determine if you need to append new data or not. Typically, an ordered sequence / ordered identity is used to insert new integer or big integer values in the source which is ideal for determining what data needs to be appended into Greenplum/HAWQ. Alternatively, a timestamp can be used for this column. Outsourcer only supports integer, big integer, and timestamp columns and all should be ordered in the source.

Append Process

Outsourcer Append Jobs do the following:

1. Connects to the Source and checks that it is valid
2. Creates Target schema if not found
3. Creates Target table if not found based on DDL from Source
4. Gets the MAX value from Greenplum
5. Gets the MAX value from the Source
6. Start gpfdist process
7. If this is the initial load
    a. Create External Table
8. If not the initial load, compare the two MAX values and if don't match, create External Table with added filter of greater than Greenplum MAX value
9. Insert data
10. Stop gpfdist process
11. Executes any SQL found in the SQL Text field. *Note: See Transform job for more details.*

Replication Job Type (Only Valid for Greenplum and not HAWQ)

This type of Job does Change Data Capture using Triggers in the Source. It is a complete solution that does everything for you. You will not have to manually create database triggers, use files, etc. to replicate data from Oracle and SQL Server to Greenplum.

Replication requires that the Source table has a Primary Key. If a key is not available, the Job will fail. The data type for the column should be integer or big integer.

SQL Server Objects

Three triggers are created in the Source database to track INSERTS, UPDATES, and DELETES. These changes are inserted into a new table in the Source that Outsourcer creates dynamically.

Oracle Objects

One trigger is create in the Source database to track INSERTS, UPDATES, and DELETES. These changes are inserted into a new table in the Source that Outsourcer creates dynamically.

Replication Process

Outsourcer Replication Jobs do the following:

1. Connects to the Source and checks that it is valid
2. Creates Target schema if not found
3. Creates Target table if not found based on DDL from Source
4. Creates stage and archive tables in Greenplum if not found

5. Creates triggers and log table in Source if not found
6. If Job is configured to snapshot or if any of the required, dynamically created objects that aren't present, force a snapshot.
7. If snapshot
    a. Refresh the table
    b. Recreate Archive and Stage tables plus the triggers and log table in the Source
8. If not a Snapshot
    a. Get the MAX ID from the log table in the Source
    b. Get the MAX ID from the Archive table in Greenplum
    c. If the two don't match
        i. Load new changes to the stage table
        ii. Apply changes with Outsourcer Replication function in Greenplum
        iii. Archive changes from stage table to archive table in Greenplum
11. Executes any SQL found in the SQL Text field. *Note: See Transform job for more details.*

As you can see, the Replication Job is very powerful and complete. There are other Change Data Capture solutions available from commercial vendors that read database log files directly rather than using triggers. Unfortunately, those tools do not provide a complete solution like Outsourcer. You will have to manually create all of the logic that is built into Outsourcer.

Note: Be sure VACUUM your replicated tables from time to time to prevent table bloat. You can also use the sql_text value to VACUUM the table.

Transform Job Type

Transform Jobs are useful to manipulate data in Greenplum. This allows you to create an ELT or Extract Load and Transform solution for Greenplum.

You can have one or many SQL statements separated by a semicolon (;). Each statement is executed in a new transaction. Note: Please refrain from using comment blocks or embedded semicolons in your SQL statement.

DDL Job Type

A DDL Job only creates the table in Greenplum based on the DDL from the source. It is similar to a refresh Job but it doesn't load any data.

DDL Process

Outsourcer Refresh Jobs do the following:

1. Connects to the Source and checks that it is valid
2. Creates Target schema if not found
3. Creates Target table if not found based on DDL from Source
4. Executes any SQL found in the SQL Text field. *Note: See Transform job for more details.*

Append-Optimized/Append-Only

All job types, except for Transform, now have the option to make tables Append-Only/Append-Optimized, Compressed, and Row or Column oriented. There are some exceptions to this.

1. HAWQ installations have the Append-Only/Append-Optimized option defaulted to on and the ability to change this has been disabled. This is because HAWQ only supports Append-Only tables.

2. Greenplum Database 4.2 and earlier is defined as “Append-Only”.
3. Greenplum Database 4.3 and later is defined as “Append-Optimized” as DELETE and UPDATE statements are allowed.
4. Replication to Greenplum Database 4.2 has the Append-Only option removed.
5. Replication to HAWQ has been disabled and will be introduced in a future release.

In Greenplum, it is highly recommended to use Append-Only or Append-Optimized tables as they use less space, allow for compression, allow for column orientation, and generally speed up query performance. HAWQ only has Append-Only tables.

Compression

Greenplum database and HAWQ both support quicklz and zlib compression algorithms. Quicklz is the most common and provides a good balance of performance of compressing and decompressing data while saving disk space. Zlib has the ability to specify a range of 1 through 10 for the level of compression desired. Outsourcer simply uses the quicklz algorithm if you define a table to be compressed. This was done to keep the user interface as simple as possible and because quicklz is the most commonly used compression algorithm.

HAWQ has been enhanced to use Snappy compression with Parquet tables (column orientation) but for row oriented tables, Quicklz is used.

Note: Compression requires using Append-Only/Append-Optimized tables.

Row/Column Orientation

Greenplum database and HAWQ both support row or column oriented tables. This is useful for reducing IO and improving compression for very wide and large tables. The default is for tables to be row oriented.

HAWQ uses Parquet files for column orientation.

Note: Column orientation requires using Append-Only/Append-Optimized tables.

Dependent Jobs

Dependent jobs can not be achieved by updating the sql_text field of a Job and adding “select os.fn_queue(12)” which will put job.id = 12 in the Queue after the main Job completes. You can also add multiple Jobs to kick off by using multiple SQL statements that are separated with a semicolon.

Queue

This is the heart of the multi-threaded processing that is done in Outsourcer. Each Job in the Queue is processed in a separate thread and records that date it was put in the Queue, when it actually started, and when it completed. It also shows the duration, number of rows, the target table, and any error message that might have happened.

Statuses

A Job goes through several statuses in the Queue.

1. Queued
2. Processing
3. Successful or Failed

The Queue limits the number of concurrent jobs with the os.variables value of “max_jobs”. You can update this value to increase or decrease the load dynamically.

Custom SQL Tables

Instead of defining a Job based on the table definition found in the Source, you can create an External Table based on a SQL statement that will execute in the Source database. You will have to provide the External Table Name, the SQL statement to execute, and the columns with data types.

The User Interface automates all of the steps needed for Custom SQL tables but you can manually create the table by inserting into os.ao_custom_sql and then restarting Outsourcer.

User Interface

All Web Pages

All pages have a Search bar, Limit for the number of records to show, sorting of every column, and pagination (Previous and Next).

Login

Enter a valid username and password from the Greenplum database. The username you use must be a “superuser” in the database. The password must the the database password (or LDAP if configured in the pg_hba.conf file) and not the operating system password.

Sources

Clicking on the Sources tab, you see the list of configured Sources (Oracle and SQL Server connections). You can Define New Source or Update, Delete, and Create Jobs with an existing Source configuration. Creating jobs from this page automatically creates jobs for all tables in a Source schema to a Target schema in Greenplum / HAWQ.

Jobs

A Job is a source to target mapping. You can Define New Job or Update, Delete, or Queue an existing Job. Placing a Job in the Queue means it will be executed immediately in the Queue.

There are two additional links on this page. Queue All Jobs which puts all Jobs into the Queue to execute immediately and Delete All Jobs which removes all Jobs from the Job page.

Queue

Rerun Link

A Successful or Failed Job can be Rerun.

Delete Link
A Job in the Queue that hasn't started to process yet can be removed and there will be a link for this.

Cancel Link
Jobs currently processing will have the ability to be cancelled.

Schedules
A Schedule defines how often Outsourcer will automatically put a Job into the Queue to be processed. Several predefined schedules are included with Outsourcer and can be updated, deleted, or added.

Assign to Jobs Link
This is a quick way to assign a schedule to a group of Jobs based on the target schema.

Custom Tables

Define New Custom External Table Link
Create a new Custom Table by defining the table name, columns, SQL statement, and Source Connection.

Update Link
Update an existing Custom Table.

Delete Link
Delete an existing Custom Table.

Environment
View and update Environment settings.

Scheduler Daemon Link
This will Start and Stop the daemon that schedules Jobs.

Queue Daemon Link
This will Start and Stop the daemon that processes Jobs in the Queue.

Max Jobs Link
This is a dynamic setting which controls how many concurrent Jobs (threads) Outsourcer will execute. Try adjusting this up or down based on the demand you can place on the source databases and how much load you want in Greenplum for loading data.

oFetchSize Link
This is an Oracle specific setting for how many records are retrieved per fetch. The larger this value, the less fetches are needed to get all of the rows but this requires more memory per fetch. There is also a point of diminishing returns on making this value too large.

Appendix

Email Alerting

Job failures utilize the SMTP alerting feature in HAWQ and Greenplum to notify administrators of a failed job. The email will contain the queue id as well as the error message.

Steps to Enable

1. Edit $MASTER_DATA_DIRECTORY/postgresql.conf on your MASTER host.
2. Uncomment the gp_email_smtp_server line and add a valid SMTP server.
3. Uncomment gp_email_from and add a valid value.
4. Uncomment gp_email_to and add a valid value.
5. Execute “gpstop -u” for the changes to take affect.

Refer to the Systems Administrator Guide for more assistance on configuring SMTP alerting.

Once in place, every failed job will send an email alert.

Data Cleansing

Data sometimes contains special characters that break typical loading utilities and then require manually editing files to correct the problem. This makes the entire load process much slower because you will have a single threaded process be the bottleneck for your data loads. The other problem encountered are the various code pages/character sets involved with the source database, source file server, target file system, and target database (Greenplum). Using UTF-8 solves this problem.

UTF-8

Outsourcer uses Microsoft’s, Oracle’s, and Greenplum’s JDBC drivers to connect to the source and target databases. “High-ASCII” values that are often times problematic are no problem for Outsourcer. A common example of this is a smart quote from Microsoft. This loads easily to Greenplum with Outsourcer.

Special Characters

Some characters are just special and need to be addressed beyond just the character set.

\

This is an escape character for the Greenplum loading utility. To fix this, Outsourcer escapes the escape character.

|

The pipe is the delimiter and if this is found in your data, it will fail to load unless it is escaped. To fix this, Outsourcer escapes the pipe.

\r and \n

This is carriage return and new line characters. These characters are replaced with a space so that data will load and you will still be able to analyze the text.

\0

This is a “null” character. This is sometimes used to insert “nothing” into a NOT NULL column. This character is replaced with an empty string.

Outsourcer Installation Files

/LICENSE.txt	Software license
/README.txt	Read me file with basic information
/config.properties	Database connection information for Java program
~/.bashrc	Bashrc file for the gpadmin user
~/.pgpass	Pgpass file for the gpadmin user
/bin/agentstart	Stops the Scheduler Agent
/bin/agentstatus	Retuns the status (up or down) of the Scheduler Agent
/bin/agentstop	Starts the Scheduler Agent
/bin/customstart	Starts gpfdist process for custom tables
/bin/customstop	Stops a gpfdist process
/bin/getdata	Used by external tables
/bin/jobstart	Starts a gpfdist process for a job in the queue
/bin/jobstop	Stops a gpfdist process
/bin/jobstopall	Stops all gpfdist processes for jobs
/bin/osstart	Starts Queue daemon
/bin/osstatus	Returns the status (up or down) of the Queue Daemon
/bin/osstop	Stops Queue daemon
/bin/start_all	Starts all processes
/bin/stop_all	Stops all processes
/bin/uistart	Starts the UI
/bin/uistop	Stops the UI
/jar/Outsourcer.jar	Outsourcer Jar file
/jar/OutsourcerScheduler.jar	Outsourcer Scheduler Jar file
/jar/OutsourcerUI.jar	Outsourcer UI Jar file
/jar/gpdb.jar	Greenplum JDBC driver
/jar/nanohttpd.jar	NanoHttpd Jar file
/log/	Log files are placed here for debugging
/os_install.sh	Install script for the database components of Outsourcer
/os_path.sh	Sets the Outsourcer path and default environment variables
/sql/*	Used by os_install.sh to install database components

Datatypes

A key feature of Outsourcer is the automatic conversion of Oracle and SQL Server datatypes to Greenplum datatypes. The following charts describe the conversion done by Outsourcer. Please note that the chart does not include data types that have the same datatype name between the source and Greenplum. Also, when a column is defined in the source with one of the unsupported datatypes, the column is omitted from the table in Greenplum.

Oracle		Greenplum	
Datatype	**Description**	**Datatype**	**Description**

BFILE			Not supported by Outsourcer
BINARY_DOUBLE		float8	
BINARY_FLOAT		float8	
BLOB			Not supported by Outsourcer
CHAR	Single byte characters	character(length)	Single and Multi-byte characters
CLOB	Large text	text	Single and multi-byte large text
DATE		timestamp	
DECIMAL	Sub-type of NUMER that has 38 decimal digits	numeric	decimal is equivalent to numeric in Greenplum so Outsourcer uses numeric
INT	Sub-type of NUMBER that has 38 digits	numeric	Far larger than a Greenplum integer or bigint
LONG	Only supported by Oracle for backward compatibility	text	Single and multi-byte large text
LONG RAW	De-supported by Oracle		Not supported by Outsourcer
MLSLABEL	De-supported by Oracle		Not supported by Outsourcer
NCHAR	Single and Multi-byte characters	character(length)	Single and Multi-byte characters
NCLOB	Single and multi-byte large text	text	Single and multi-byte large text
NUMBER		numeric	Precision and scale does not reduce overhead of storing data
NVARCHAR2	Single and Multi-byte characters	character varying(length)	Single and Multi-byte characters
RAW	De-supported by Oracle		Not supported by Outsourcer
ROWID	Base 64 encoding of the location	character varying(18)	Convert this value to text
TIMESTAMP		timestamp	
TIMESTAMP WITH LOCAL TIME ZONE		timestamptz	
TIMESTAMP WITH TIME ZONE		timestamptz	
UROWID	Universal Row ID	character varying(length)	Convert this value to text
VARCHAR	Single byte characters	character varying(length)	Single and Multi-byte characters
VARCHAR2	Single byte	character	Single and Multi-byte

	characters	varying(length)	characters
XMLTYPE			Not supported by Outsourcer

SQL Server		Greenplum	
Datatype	**Description**	**Datatype**	**Description**
bigint		bigint	
binary			Not supported by Outsourcer
bit		boolean	
char	Single byte characters	character(length)	
date		date	
datetime		timestamp	
datetime2		timestamp	
datetimeoffset		timestamptz	Includes timezone offset
decimal		numeric	Numeric is equivalent to decimal
float		float8	
hierarchyid			Not supported by Outsourcer
image	De-supported datatype by Microsoft		Not supported by Outsourcer
int		integer	
money		numeric	
nchar	Single and multi-byte characters	character(length)	Single and multi-byte characters
ntext	De-supported datatype by Microsoft		Not supported by Outsourcer
nvarchar		character varying(length)	Single and multi-byte characters
real		float8	
smalldatetime		timestamp	
smallint		smallint	
smallmoney		numeric	
sql_variant			Not supported by Outsourcer
sysname		character varying(length)	Single and multi-byte characters
text	De-supported datatype by Microsoft		Not supported by Outsourcer
time		time	
timestamp	Not the ANSI standard for date but a binary field.		Not supported by Outsourcer
tinyint		smallint	
uniqueidentifier		character varying(36)	

varbinary			Not supported by Outsourcer
xml			Not supported by Outsourcer

Insert Only Model

The Outsourcer catalog is stored in the `os` schema and contains tables prefixed with `ao_` and views with the same name but without the prefix. Example: table: `os.ao_job` and view: `os.job`. HAWQ does not support UPDATE or DELETE commands from a table so Outsourcer has been designed to only INSERT data into the `ao_` tables. These tables have 2 additional columns, which is `insert_id` and `deleted`.

When a row needs to be UPDATED, a new row is inserted into the `ao_` table and then the view only shows the greatest `insert_id` for the given logical primary key.

When a row needs to be DELETED, a new row is inserted into the `ao_` table but with the `deleted` column set to true.

The user interface and functions take care of this insert only model but if you need to manually edit tables, be aware of how this works.

Release Notes

Version 5.1.2

Made minor changes to exception handling to make sure all connections are closed and not left open.

Version 5.1.1

Append jobs now can key off big integer in addition to integer columns. Also, timestamp support was added for Append jobs too.

Version 5.0.9

Added support for HAWQ 2.0.

Version 5.0.8

Updated DDL creation for HAWQ to use Parquet instead of Column orientation. When Parquet and Compression are chosen, then Snappy compression is used and Quicklz is used for Row orientation.

Version 5.0.7

Fix

1. A customer found an issue with an Oracle TIMESTAMP where the year was 0013. Oracle's JDBC driver doesn't automatically format this to be 0013 and drops the year to just 13. This isn't a valid year so loading this data failed with 5.0.6. A similar fix was already in place for DATE but now I format TIMESTAMP too.

Version 5.0.6

Enhancements

2. Changed session management to be in UTC rather than the local time zone. This handles the situation where the time zone of the database is set differently than the time zone of the host where Outsourcer is running.

Version 5.0.5

Enhancements

1. Handles gpfdist failing to start.
2. If port being requested by gpfdist is in use, Outsourcer will skip it and go to the next without erroring out.
3. If all ports are exhausted within the range of OSPORT_LOWER and OSPORT_UPPER (defined in os_path.sh), then the job will fail and log the error that gpfdist was unable to start.
4. Retain gpmon entry in the .pgass file if found.
5. osstart can work independently of the user interface. It will start/stop gpfdist processes for jobs as needed.

Version 5.0.3

Enhancements

1. Added parameter for gpfdist max row size to os_path.sh.
2. Spawn a unique gpfdist process for every Job being processed.
3. Spawn a unique gpfdist process for every Custom SQL table.

Fixes

1. A single gpfdist process was used in prior 5.x releases which caused performance problems. Now Outsourcer will create a unique gpfdist process for each job and also each Custom SQL table.
2. Replication jobs weren't using the trigger table properly which caused jobs to fail.

Version 5.0.1

Enhancements

1. Added email alerting for failed jobs. This is utilizing the SMTP alerting feature in Greenplum and HAWQ. Edit your postgresql.conf file to enable.

Fixes

3. Installations failed when the path had an underscore.

Version 5.0.0

Enhancements

1. External Web Tables have been replaced with External Tables utilizing gpfdist.

2. Installation can be done on any host accessible by the Segment Hosts / Data Nodes.
3. Automatic download of Microsoft SQL Server JDBC driver.
4. Root is no longer needed for installation.
5. New “start_all” and “stop_all” scripts.
6. Custom External Tables are accessible in User Interface.

Version 4.1.6

Bugs Fixed

1. Small bug fix for cancel feature on transform jobs. Multiple statement transform jobs were not being cancelled properly.
2. Added missing documentation on new External Tables.

Version 4.1.5

Bugs Fixed

1. Error messages raised from Oracle and SQL Server could include special characters which would cause the thread in Outsourcer to die and the status remain labeled as “processing”. Error messages no longer have escape and quote characters to prevent this problem.
2. On new installations, the permissions for the Oracle and SQL Server Jar files would remain owned by root. This has now been corrected to be the admin user (gpadmin).
3. On the “Sources” screen, the label for Append-Only now is correct for Greenplum Database 4.3 and reflects that these are really “Append-Optimized” tables.

Enhancements

1. Added External Tables to enable starting and stopping the UI rather than having to ssh to the Master host.
2. “Queue” screen now provides the ability to cancel processing jobs.
3. Stopping Outsourcer Queue Daemon (Environment Screen), now will cancel currently running Jobs in the Queue.
4. Starting Outsourcer Queue Daemon (Environment Screen), now will make sure orphaned jobs in the Queue will be marked as Failed and if any are currently running, it will cancel these jobs too.

Version 4.1.4

- Corrected Jar file installation instructions in os_install.sh
- Added support for new SQL Server 2014 data types including bit, date, datetime2, datetimeoffset, real, time, and tinyint
- Removed ANALYZE from table loading because it is handled automatically by gp_autostats_mode=on_no_stats (default) or on_change. If you are using none, you can still ANALYZE the table using the sql_text value.

Version 4.1.3

- Added ability to specify the User Name and UI Port during the install.

- Huge UI performance increase by using PGPoolingDataSource and by managing sessions with in a file rather than in a table.
- Fixed a bug in the installer where the AUTHSERVER environment variable wasn't being set in the .bashrc file correctly.

Version 4.1.1

- Fixed a bug in the Create External Table function used to create static external tables. This was introduced when refactoring code and using SourcePort rather than port for the column names.
- Fixed a bug in the Java code to execute static External Tables.

Version 4.1.0

- Added support for HAWQ for Append, Refresh, Transform, and DDL refresh types. Replication to be added in the future.
- Added ability to make target tables Append-Only (<= GPDB v4.2) or Append-Optimized (>= GPDB v4.3).
- Added ability to make target tables Compressed (quicklz) for Append-Only/Append-Optimized tables for GPDB and HAWQ.
- Added ability to pick row or column orientation for Append-Only/Append-Optimized tables for GPDB and HAWQ.
- Append-Optimized tables allow for DELETE statements (>= GPDB v4.3) while Append-Only tables (<= GPDB v4.2) do not. Replication jobs in GPDB v4.2 will not give the ability to pick Append-Optimized tables.
- All Outsourcer tables are now Append-Optimized/Append-Only.
- New installer makes installation much simpler with automatic backups of the os schema, automatic upgrades from previous versions, automatic editing of configuration files, and automatic starting of services.
- Removed the automatic creation of primary keys on all tables.
- Fixed bug where an extra database session was kept open while tables were being loaded.

Version 4.0.2

- Corrected Open Source license file. It is now using a BSD license and the NanoHTTPD license (web server).
- Corrected install.sql file that incorrectly had $BODY$ for one function in the declaration.
- Corrected cookie conflict with Command Center.
- Reduced the number of database calls when using the Web interface.
- Removed updates from the os.sessions table for the Web interface.

Version 4.0.1

- Changed the default Oracle Fetch Size from 40,000 to 2000. 2000 seems to be the point of diminishing returns for any values greater. The Oracle default is only 10, which makes exporting data very slow. The higher the value, the more memory is needed and the data exporting goes faster. But a fetch size of more than 2000 doesn't improve performance but it does consume more memory.

- Separated the Greenplum JDBC driver into a separate JAR file so that it is possible to upgrade just this driver.
- Separated the User Interface classes into a separate JAR file so it needs less memory.
- Separated the Scheduler classes into a separate JAR file so it needs less memory.
- Separated the NanoHttpd classes into a separate JAR file so it can be upgraded independently of Outsourcer.
- Fixed a bug in the UI where the SQL Text Box wasn't visible when creating a Transform Job.
- Fixed a bug in the UI where quotes weren't escaped properly for SQL entered into the SQL Text Box.

Version 4.0

Open Source

- Outsourcer is now open source with the Boost Software License and the NanoHTTPD license (web server).

User Interface

- An all-new User Interface has been built allowing you to manage Sources, Jobs, the Queue, Schedules, and the Environment.
- No separate web server is needed. Everything you need for the User Interface is included.
- Dynamic Job creation based on a source schema.
- Start/Stop the Queue and Scheduler.
- Update the dynamic variables in the Environment.
- Security is handled by the database authentication method so it leverages the `pg_hba.conf` file.

Scheduling

- Jobs can be scheduled to execute on a recurring basis with many predefined schedules included.
- Separate daemon process for the scheduler agent.

Transform Jobs

- In previous versions, a transform Job would not execute unless there weren't any other Job types Processing or Queued. This was done to force all data to be loaded before it is transformed. In other words, Extract, Load, and then Transform. This restriction has been removed and will execute based on the queue_date just like the other Job types.

SQL_Text

- The sql_text column in the Job table was ignored for all Jobs types except transform for previous versions. This is now available for all Job types and it is execute as the last step.

Dependent Jobs

- Dependent jobs can not be achieved by updating the sql_text field of a Job and adding "select os.fn_queue(12)" which will put job.id = 12 in the Queue after the main Job completes. You can also add multiple Jobs to kick off by using multiple SQL statements that are separated with a semicolon.

Dynamic Environment Variables

- The Oracle fetch size and the maximum number of concurrent Jobs are now dynamically set and no longer set in the `.bashrc` file.

Version 3.1

Oracle

- FetchSize is now configurable. To minimize round trips to Oracle, make this setting rather large (greater than the default of 10) which increases exporting speed but at the expense of needing more RAM. Adjust this setting up or down based on your environment. Default is 2000. There is a point of diminishing returns when setting this higher than 2000 and the larger this value, the more RAM will be needed by Java.

SQL Server

- Fix to better handle non-default schemas

DDL Refresh Type

- Several customers have requested for Outsourcer to just create the tables in Greenplum based on the DDL in the source without loading the tables. This new Refresh Type does just that.

Version 3.0

Security

- The External Tables created by Outsourcer no long contain the username and password for the source connection.
- Oracle sources now use ALL_% objects rather than DBA_% objects. This means you don't need to grant DBA or SELECT on the DBA Views in Oracle to use Outsourcer.
- TRUST authentication is used to start Outsourcer so no password is stored.

Enhancements

- Oracle connections are now faster and use less memory than with 2.x.
- New External Table feature for defining a SQL statement to execute in Oracle or SQL Server to be the source of an External Table.
- Better examples of the Job types for Oracle and SQL Server

Fixes

- When a Job fails and under certain conditions, Outsourcer could leave a database connection open to the source. Outsourcer now properly closes the connection on Failed Jobs.

- If multiple Jobs executed at the same time, it was possible for more than 1 Job to attempt to create the target schema at the same time. This no longer is a problem.