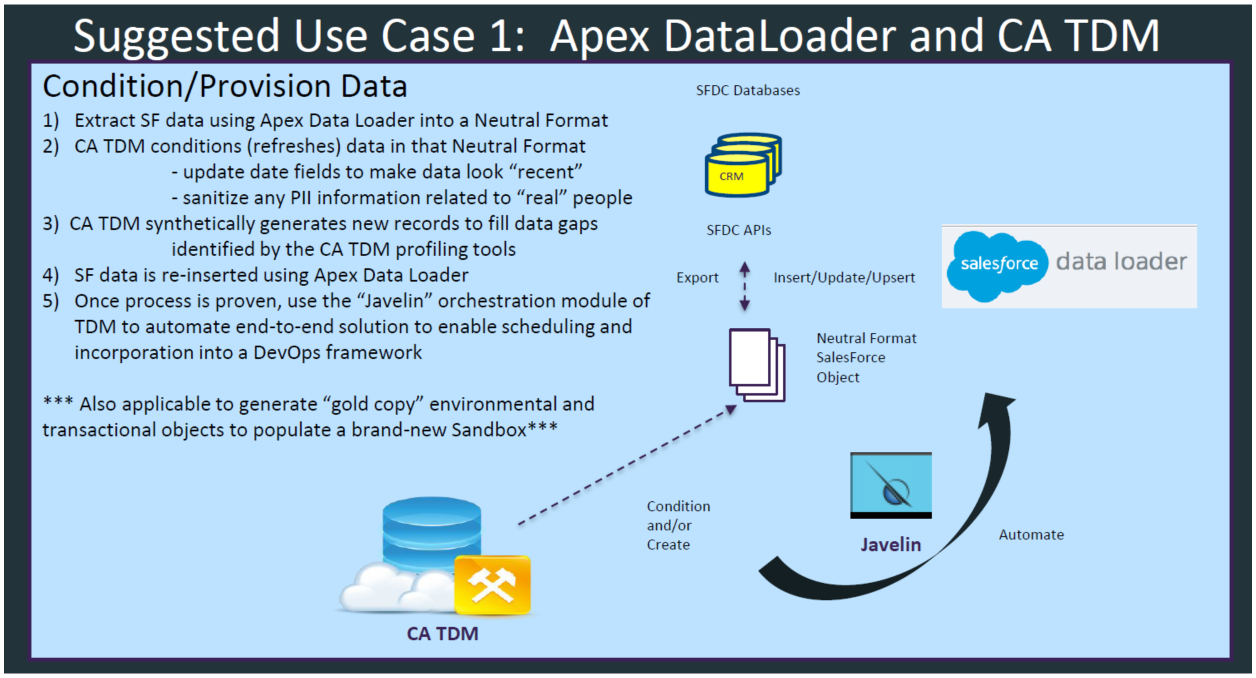
Setting up a Broadcom CA TDM demo with SalesForce.com

Version 0.3 August 20, 2020

Broadcom CA TDM can work with Salesforce via the "Data Loader". It is designed to work with 50K-5M records: https://developer.salesforce.com/page/Data_Loader

The Data Loader provides the capability to insert, update, delete or extract Records from the Salesforce database.



Suggested Use Case 1: Apex DataLoader and CA TDM

Condition/Provision Data

1) Extract SF data using Apex Data Loader into a Neutral Format
2) CA TDM conditions (refreshes) data in that Neutral Format
   - update date fields to make data look "recent"
   - sanitize any PII information related to "real" people
3) CA TDM synthetically generates new records to fill data gaps identified by the CA TDM profiling tools
4) SF data is re-inserted using Apex Data Loader
5) Once process is proven, use the "Javelin" orchestration module of TDM to automate end-to-end solution to enable scheduling and incorporation into a DevOps framework

*** Also applicable to generate "gold copy" environmental and transactional objects to populate a brand-new Sandbox***

To start: Register for a free Salesforce developer account

https://developer.salesforce.com/signup

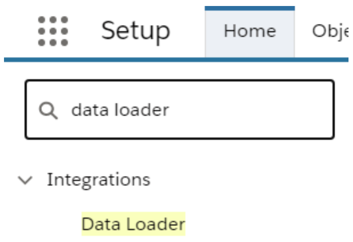
Once you are registered & validated, you can access your Developer account via

https://login.salesforce.com/

Download the Salesforce Apex Data Loader – from the Salesforce docs:

System administrators can download the Data Loader from the following locations
> **In Salesforce Classic:** Setup | Administration Setup | Data Management | Data Loader
> **In Lightning Experience:** Gear icon | Setup | Integrations | Data Loader

Or just search for "data loader"


Setup
Home
Obj
data loader
Integrations
Data Loader

Setup / Synthetic Data Generation

We will use the dataloader command line interface / batch execution to perform operations. The following references are used:

Pre-requisites for use of command line/batch mode:

https://help.salesforce.com/articleView?id=command_line_intro.htm&type=5

How to run in batch mode:

https://help.salesforce.com/articleView?id=using_the_command_line_interface.htm&type=5

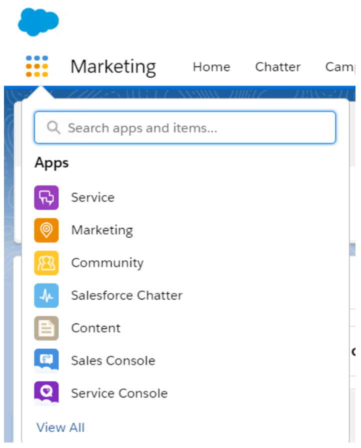
(1) Install the dataloader by double-clicking and installing. The default location will be:

C:\Users\<your username>\dataloader\v48.0.0

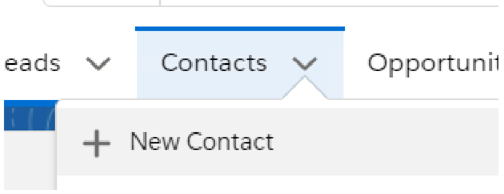
Pre-requisite Note:

Install Zulu OpenJDK version 11 for Windows using the .MSI file.

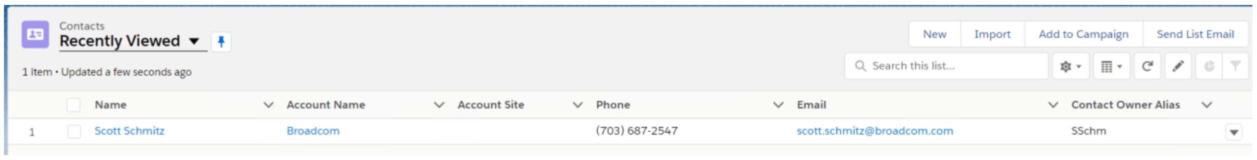
(2) In Salesforce, decide which sample application you'll use. My choice is Marketing. After login, click the 3x3 button top left, select Marketing



(3) For this example, we'll work with the Salesforce "Contacts" Entity. Select the Contacts pulldown and select New Contact



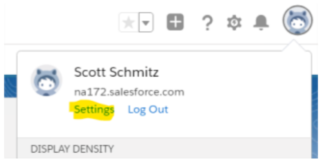
(4) Enter information into the fields and Save. You'll now see your contact listed:



Contacts
Recently Viewed
1 item • Updated a few seconds ago

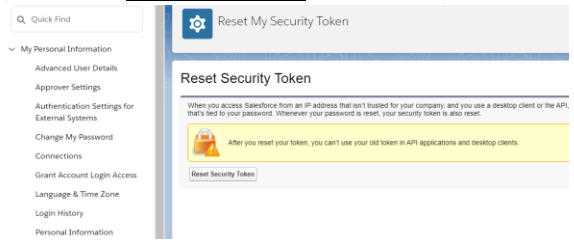
	Name	Account Name	Account Site	Phone	Email	Contact Owner Alias
1	Scott Schmitz	Broadcom		(703) 687-2547	scott.schmitz@broadcom.com	SSchm

(5) In order to use Data Loader, you'll need not only your userid/password combination, but you'll also need a Security Token to access via the api.   In Salesforce, go to Settings:



Scott Schmitz
na172.salesforce.com

Then go to My Personal Information>Reset My Security Token. Click Reset Security Token.
You'll get it sent to your email.   Copy it, and when you are prompted for the password, enter your password <u>appended with</u> the security token.



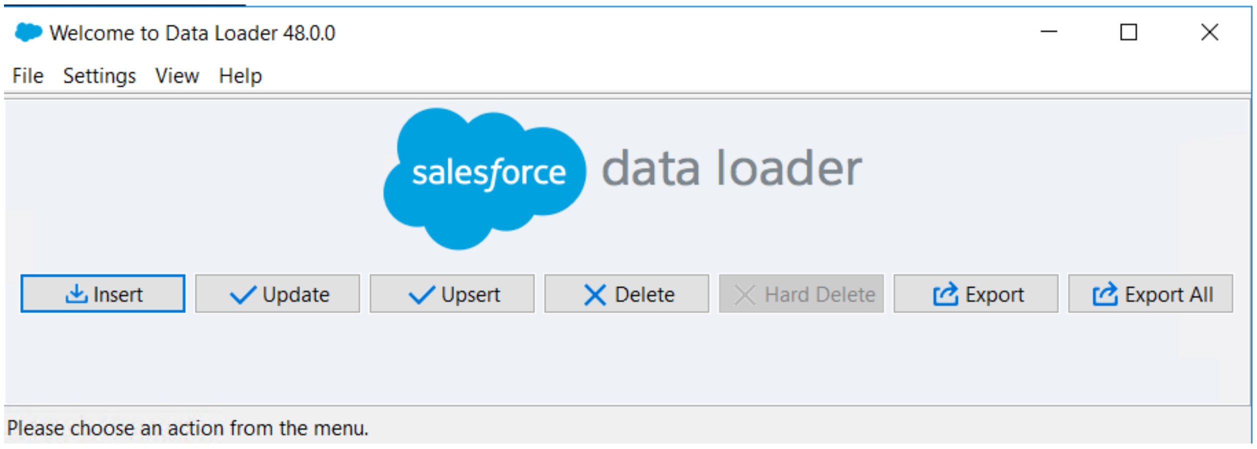
Reset My Security Token

Reset Security Token

When you access Salesforce from an IP address that isn't trusted for your company, and you use a desktop client or the API, that's tied to your password. Whenever your password is reset, your security token is also reset.

After you reset your token, you can't use your old token in API applications and desktop clients.

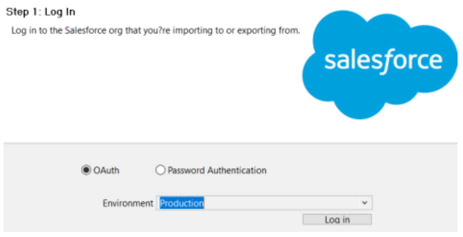
(6) In order to create a generator, we need to know the format of the Contact Entity that the DataLoader knows (without requiring filling in field mapping information).   Launch the UI for the Data Loader:



Welcome to Data Loader 48.0.0

Please choose an action from the menu.

(7) Select Export. You'll want to select Production from the pulldown as the Developer area is Prod.



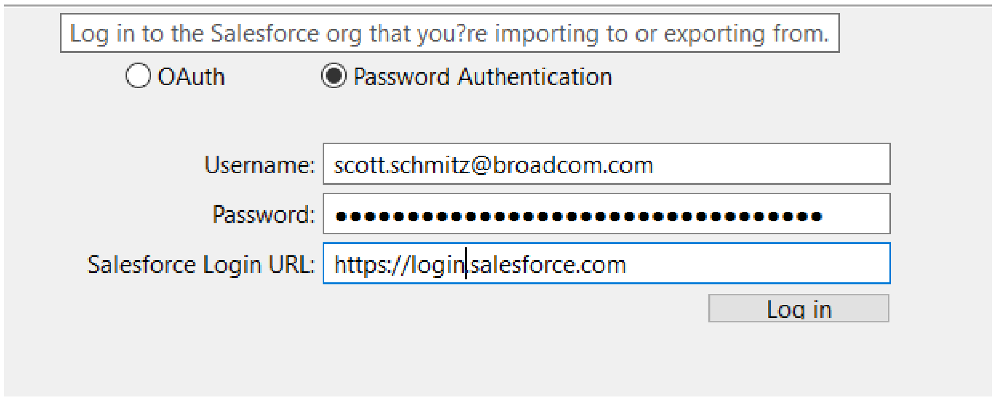
Step 1: Log In

Log in to the Salesforce org that you?re importing to or exporting from.

[✓ OAuth] Password Authentication

Environment: Production

(8) Select Password Authentication radio, enter your Salesforce credentials (with password appended with security token), and click Log in.


Log in to the Salesforce org that you?re importing to or exporting from.

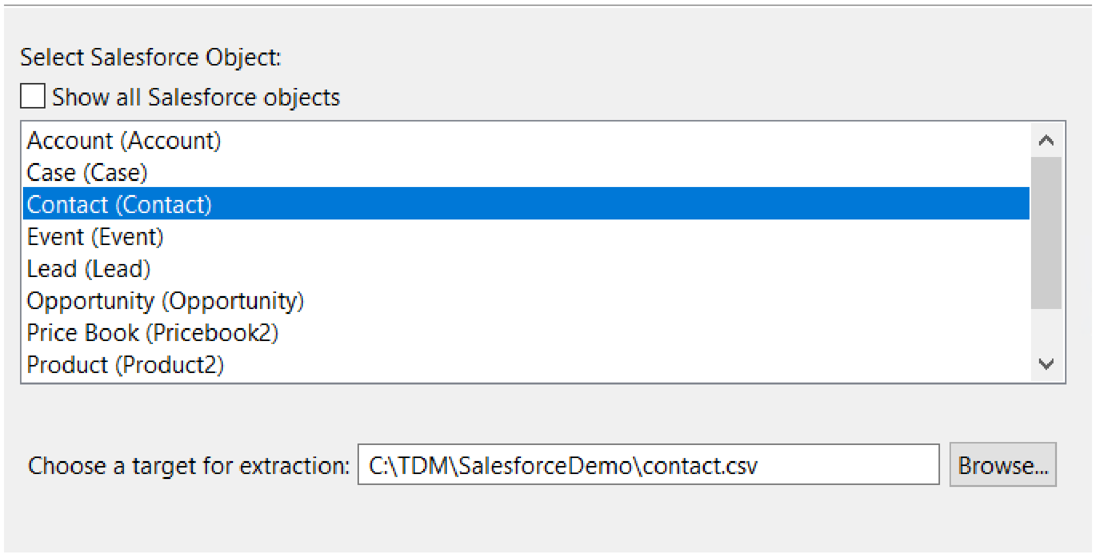
OAuth [✓ Password Authentication]

Username: scott.schmitz@broadcom.com

Password: ●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

Salesforce Login URL: https://login.salesforce.com

(9) After you see “Login Successful” next to the Log in button, select Next. Select Contact entity, and provide the name & location of the output .csv file

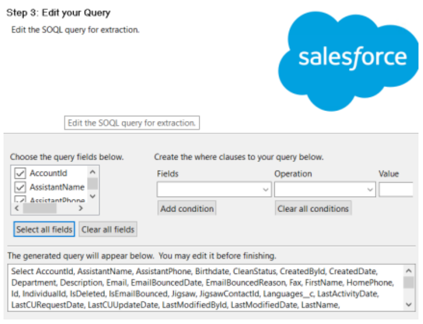

Select Salesforce Object:

☐ Show all Salesforce objects

Account (Account)
Case (Case)
[✓ Contact (Contact)]
Event (Event)
Lead (Lead)
Opportunity (Opportunity)
Price Book (Pricebook2)
Product (Product2)

Choose a target for extraction: C:\TDM\SalesforceDemo\contact.csv

(10)Select your desired fields, or select all fields and click Finish.


Step 3: Edit your Query

Edit the SOQL query for extraction.

Edit the SOQL query for extraction.

Choose the query fields below.

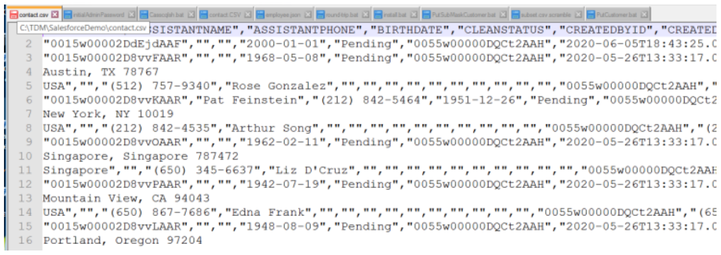
✓ AccountId
✓ AssistantName
✓ AssistantPhone

Create the where clauses to your query below.

The generated query will appear below. You may edit it before finishing.

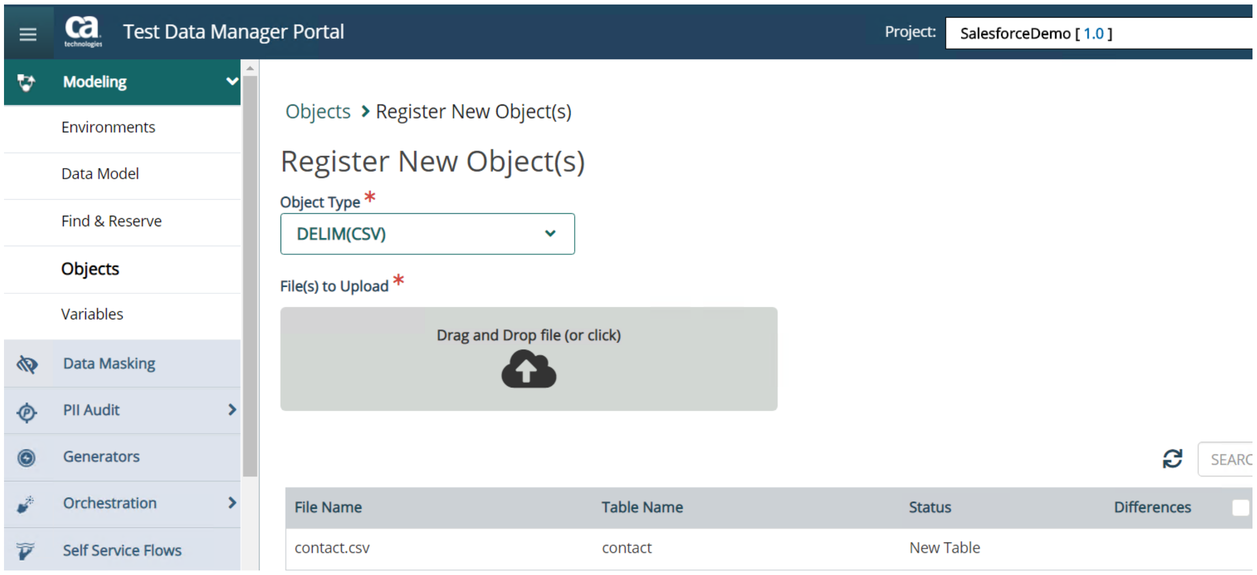
```
Select AccountId, AssistantName, AssistantPhone, Birthdate, CleanStatus, CreatedById, CreatedDate,
Department, Description, Email, EmailBouncedDate, EmailBouncedReason, Fax, FirstName, HomePhone,
Id, IndividualId, IsDeleted, IsEmailBounced, Jigsaw, JigsawContactId, Languages__c, LastActivityDate,
LastCURequestDate, LastCUUpdateDate, LastModifiedById, LastModifiedDate, LastName,
```

(11)The .csv file is exported.



```
C:\TDM\SalesforceDemo\contact.csv SISTANTNAME","ASSISTANTPHONE","BIRTHDATE","CLEANSTATUS","CREATEDBYID","CREATED
"0015w00002DdEjdAAF","","","2000-01-01","Pending","0055w00000DQCt2AAH","2020-06-05T18:43:25.0
"0015w00002D8vvFAAR","","","1968-05-08","Pending","0055w00000DQCt2AAH","2020-05-26T13:33:17.0
Austin, TX 78767
USA","","(512) 757-9340","Rose Gonzalez","","","","","","","","","","","0055w00000DQCt2AAH","
"0015w00002D8vvKAAR","Pat Feinstein","(212) 842-5464","1951-12-26","Pending","0055w00000DQCt2
New York, NY 10019
USA","","(212) 842-4535","Arthur Song","","","","","","","","","","","0055w00000DQCt2AAH","(2
"0015w00002D8vvOAAR","","","1962-02-11","Pending","0055w00000DQCt2AAH","2020-05-26T13:33:17.0
Singapore, Singapore 787472
Singapore","","(650) 345-6637","Liz D'Cruz","","","","","","","","","","","0055w00000DQCt2AAH
"0015w00002D8vvPAAR","","","1942-07-19","Pending","0055w00000DQCt2AAH","2020-05-26T13:33:17.0
Mountain View, CA 94043
USA","","(650) 867-7686","Edna Frank","","","","","","","","","","","0055w00000DQCt2AAH","(65
"0015w00002D8vvLAAR","","","1948-08-09","Pending","0055w00000DQCt2AAH","2020-05-26T13:33:17.0
Portland, Oregon 97204
```

(12) In TDM Portal, create a Project. Go to Model/Objects and register that delimited (.csv) file



Project: SalesforceDemo [1.0]

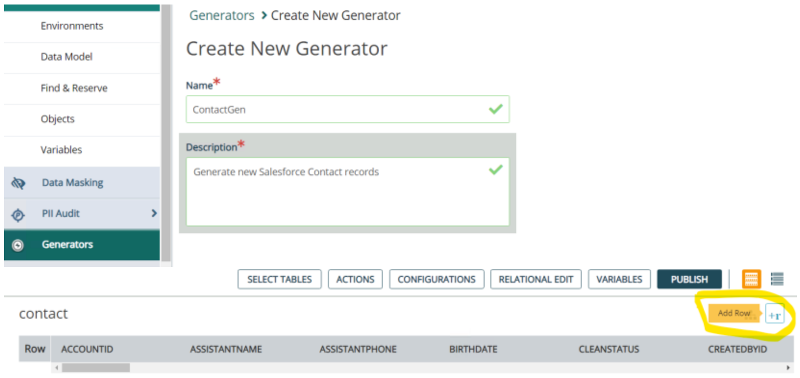
Register New Object(s)

Object Type * DELIM(CSV)

File(s) to Upload *

File Name	Table Name	Status	Differences
contact.csv	contact	New Table	

(13) Create a Generator "ContactGen". Double click to open and select the "contact" table. Click the +r button to add a line.



Create New Generator

Name* ContactGen

Description* Generate new Salesforce Contact records

contact

Row	ACCOUNTID	ASSISTANTNAME	ASSISTANTPHONE	BIRTHDATE	CLEANSTATUS	CREATEDBYID

(14) Enter formulas into the key/mandatory fields. Some examples below:

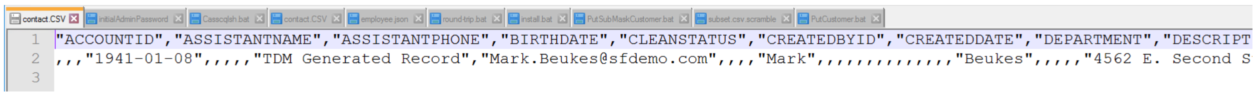
Table Name	Column Name	Definition
contact	BIRTHDATE	@dob(18,100,~CDATE~)@

contact	EMAIL	@collapse(^FIRSTNAME^.^LASTNAME^@atsign(1)@sfdemo.com)@
contact	FIRSTNAME	@randlov(0,@seedlist(FirstNameTitleGender)@,1)@
contact	LASTNAME	@randlov(0,@seedlist(LastName)@)@
contact	MAILINGADDRESS	

@randrange(1,9999)@ @percval(10%N.,5%North,10%E.,5%East,10%S.,5%South,10%W.,5%West,40%)@ @percval(10%Second St.,10%Main St.,10%Park Ave.,10%Oak St.,10%Pine St.,10%Maple Ln.,10%Washington St.,10%Lake Dr.,10%Hill Ave.,10%Ninth St.)@

contact	MAILINGCITY	@randlov(0,@seedlist(US Zip-Codes)@,3)@
contact	MAILINGPOSTALCODE	@randlov(0,@seedlist(US Zip-Codes)@,1)@
contact	MAILINGSTATE	@randlov(0,@seedlist(US Zip-Codes)@,2)@
contact	NAME	^FIRSTNAME^ ^LASTNAME^
contact	PHONE	@randlov(0,@seedlist(US Phone no)@)@
contact	TITLE	@randlov(0,@seedlist(FirstNameTitleGender)@,2)@
contact	DESCRIPTION	TDM Generated Record

(15) Publish one record to csv.   Download the zip file from the results and extract the contact.csv file.

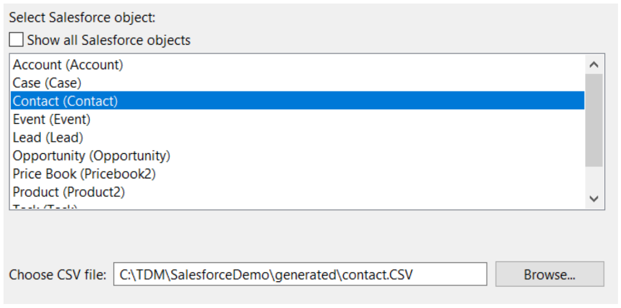


```
"ACCOUNTID","ASSISTANTNAME","ASSISTANTPHONE","BIRTHDATE","CLEANSTATUS","CREATEDBYID","CREATEDDATE","DEPARTMENT","DESCRIPT
,,,"1941-01-08",,,,,"TDM Generated Record","Mark.Beukes@sfdemo.com",,,,"Mark",,,,,,,,,,,,,,"Beukes",,,,,"4562 E. Second S

```

(16) By default, TDM produces a UTF-8-BOM formatted .csv file.   <u>Data Loader will not tolerate this</u>. Use Notepad++ to open the file, then select Encoding/UTF-8 and save the file.

(17)Launch Data Loader.

(18) Select Insert. Select Contact. Browse to where you've saved the contact.csv file. Click Next.



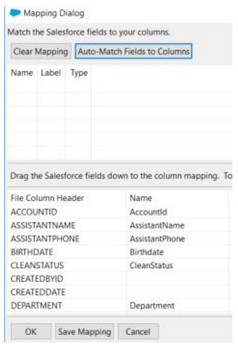
Select Salesforce object:

☐ Show all Salesforce objects

Account (Account)
Case (Case)
Contact (Contact)
Event (Event)
Lead (Lead)
Opportunity (Opportunity)
Price Book (Pricebook2)
Product (Product2)

Choose CSV file: C:\TDM\SalesforceDemo\generated\contact.CSV

Use the option to auto-match fields to columns.


Mapping Dialog

Match the Salesforce fields to your columns.

Clear Mapping | Auto-Match Fields to Columns

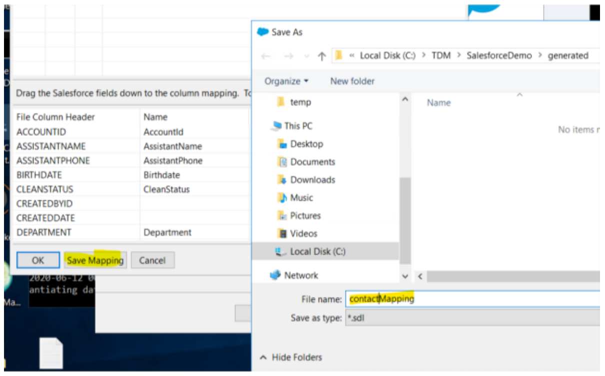
Name	Label	Type

Drag the Salesforce fields down to the column mapping. To

File Column Header	Name
ACCOUNTID	AccountId
ASSISTANTNAME	AssistantName
ASSISTANTPHONE	AssistantPhone
BIRTHDATE	Birthdate
CLEANSTATUS	CleanStatus
CREATEDBYID	
CREATEDDATE	
DEPARTMENT	Department

OK | Save Mapping | Cancel

Save the mapping for later use:


Drag the Salesforce fields down to the column mapping. To

File Column Header	Name
ACCOUNTID	AccountId
ASSISTANTNAME	AssistantName
ASSISTANTPHONE	AssistantPhone
BIRTHDATE	Birthdate
CLEANSTATUS	CleanStatus
CREATEDBYID	
CREATEDDATE	
DEPARTMENT	Department

OK | Save Mapping | Cancel

Save As

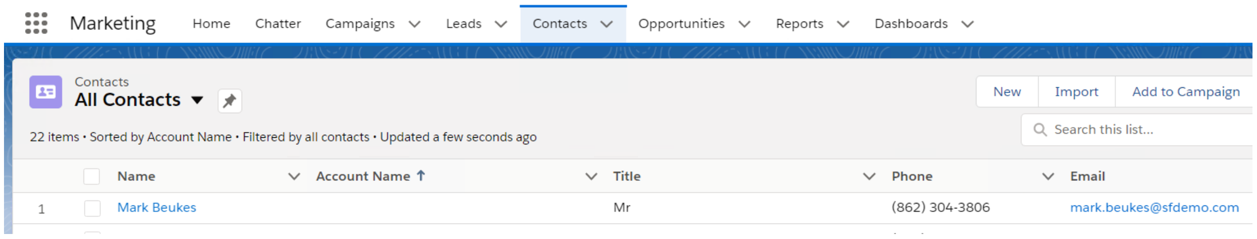
Local Disk (C:) › TDM › SalesforceDemo › generated

File name: contactMapping

Save as type: *.sdl

Define the Success/Error log location.   Click Finish.

(19) Login to Salesforce, and in the Contacts tab, select All Contacts.   You should see the new contact:



Contacts

All Contacts

22 items • Sorted by Account Name • Filtered by all contacts • Updated a few seconds ago

	Name	Account Name	Title	Phone	Email
1	Mark Beukes		Mr	(862) 304-3806	mark.beukes@sfdemo.com

(20) Now that we've confirmed that we can do a round-trip, let's automate the insert.

We'll start by encrypting the password used in the command-line operations. Change directory to C:\Users\<your_user>\dataloader\v48.0.0\bin.   Unfortunately, the OpenJDK installed by TDM at JAVA_HOME is not recent enough to use. Modify the encrypt.bat and process.bat to hard-code the path to the Zulu JDK install (C:\Program Files\Zulu\Zulu-11).

In addition, update the .bat files to provide a full path to the dataloader-48.0.0-uber.jar file in the classpath so we can execute this batch from any starting path.   E.g.

"C:\Program Files\Zulu\zulu-11\bin\java" -cp C:\Users\<yourusername>\dataloader\v48.0.0\dataloader-48.0.0-uber.jar -Dsalesforce.config.dir=%1 com.salesforce.dataloader.process.ProcessRunner %PROCESS_OPTION%

Now we'll create the encrypted password for use by the automation.   See: https://help.salesforce.com/articleView?id=command_line_create_encryption_key.htm&type=5

Open a command window and cd to C:\Users\<yourusername>\dataloader\v48.0.0\bin
From the command line, execute: *encrypt.bat –k ..\samples\conf\mysfdc.key*
Then execute: *encrypt.bat –e YourSDFCPassword-and-yoursecuritytoken ..\samples\conf\mysfdc.key*
Copy the output and store it safely – this is your encrypted password. We'll use it next.

The Data Loader\samples\conf directory already has a "process-conf.xml" file. We'll use it as a template. Make a backup copy and modify to incorporate your Salesforce ID and the encrypted password that we generated above. We'll copy one of the bean specifications to work with our Contact entity and create our Contacts insert bean. Highlighted areas need to be customized for your implementation:

```
<bean id="contactInsertProcess"
      class="com.salesforce.dataloader.process.ProcessRunner"
      singleton="false">
    <description>Contact Insert job gets the synthetically generated Contact record from a CSV file and uploads them to salesforce using 'insert'.</description>
    <property name="name" value="contactInsertProcess"/>
    <property name="configOverrideMap">
      <map>
        <entry key="sfdc.endpoint" value="https://login.salesforce.com"/>
        <entry key="sfdc.username" value="yoursalesforceid"/>
        <!-- password below has been encrypted using key file, therefore it will not work without the key setting: process.encryptionKeyFile
        the password is not a valid encrypted value, please generate the real value using encrypt.bat utility -->
        <entry key="sfdc.password" value="555555555555"/>
        <entry key="process.encryptionKeyFile" value="C:\Users\<youruser>\dataloader\v48.0.0\samples\conf\mysfdc.key"/>
        <entry key="sfdc.timeoutSecs" value="600"/>
        <entry key="sfdc.loadBatchSize" value="200"/>
        <entry key="sfdc.externalIdField" value="Id"/>
        <entry key="sfdc.entity" value="Contact"/>
        <entry key="process.operation" value="insert"/>
        <entry key="process.mappingFile" value="C:\TDM\SalesforceDemo\generated\contactMapping.sdl"/>
        <entry key="dataAccess.name" value="C:\TDM\SalesforceDemo\generated\contact.csv"/>
        <entry key="dataAccess.type" value="csvRead"/>
        <entry key="process.initialLastRunDate" value="2006-12-01T00:00:00.000-0800"/>
      </map>
    </property>
  </bean>
```

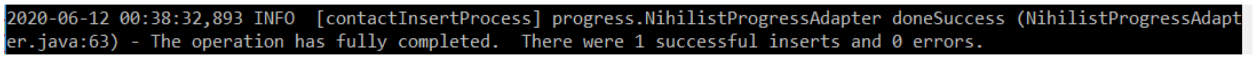
Let's run this once to confirm it works. Generate another contact.csv, get rid of the BOM using Notepad++ and save it to the c:\TDM\SalesforceDemo\generated\contact.csv location (or your custom-configured location).

Open a command window and execute:

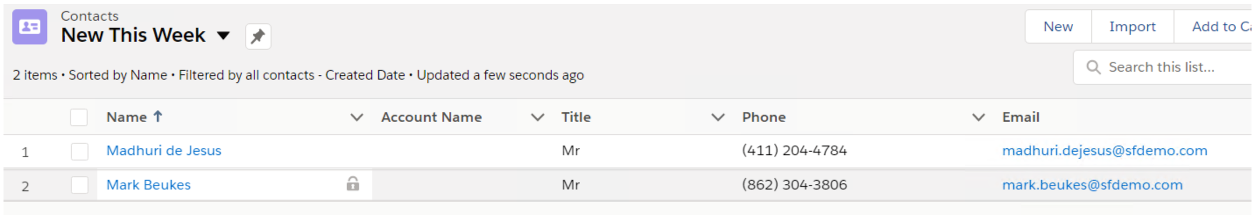
```
C:\Users\<your_user>\dataloader\v48.0.0\bin\process.bat C:\Users\<your_user>\dataloader\v48.0.0\ contactInsertProcess
```

(spaces after the .bat file and before the contactInsertProcess)

You should see success. If you see failure, double-check your bean settings…



```
2020-06-12 00:38:32,893 INFO  [contactInsertProcess] progress.NihilistProgressAdapter doneSuccess (NihilistProgressAdapter.java:63) - The operation has fully completed.  There were 1 successful inserts and 0 errors.
```



2 items • Sorted by Name • Filtered by all contacts - Created Date • Updated a few seconds ago

	Name	Account Name	Title	Phone	Email
1	Madhuri de Jesus		Mr	(411) 204-4784	madhuri.dejesus@sfdemo.com
2	Mark Beukes		Mr	(862) 304-3806	mark.beukes@sfdemo.com

Now we'll configure all the steps we'll need and put them in a batch file to execute as a TDM post-publish process.

After publishing, we'll need to grab the contact.csv file from the Job directory, strip off the BOM from the start of the file, and write it to the "generated" directory as seen above. Then we'll call the Data Loader process.bat to do the import.

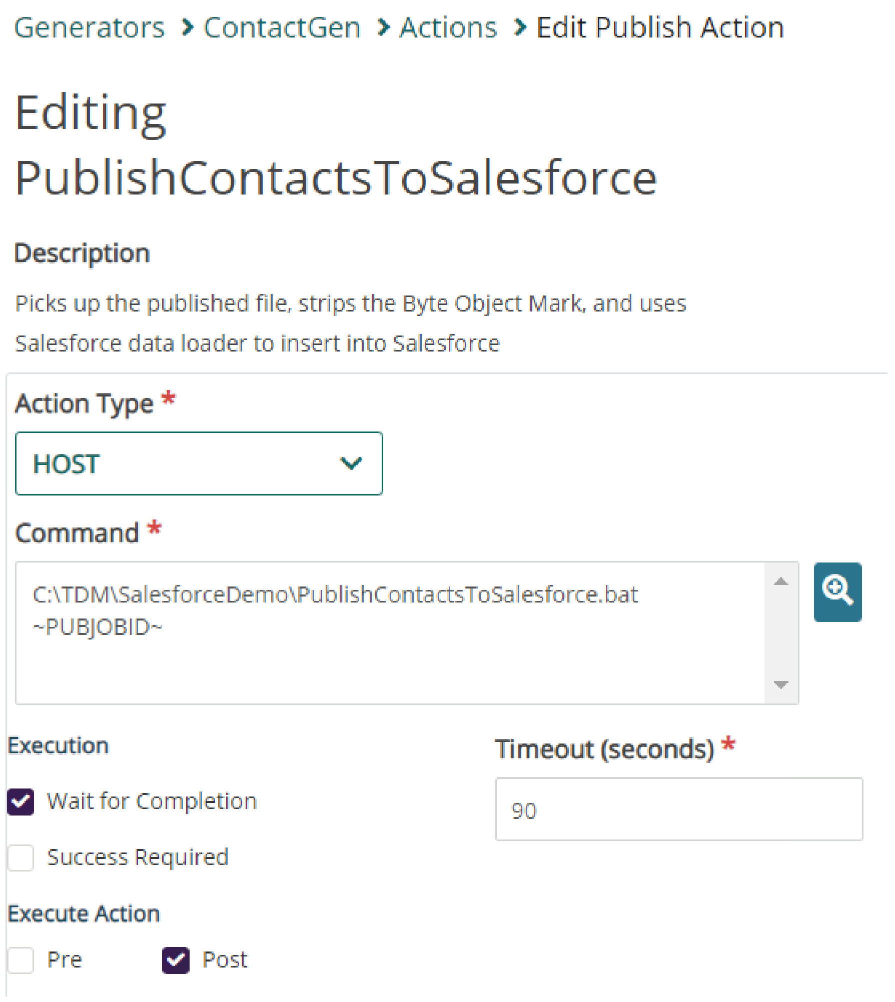
PublishContactsToSalesforce.bat

```
REM
REM PublishContactsToSalesforce <JOBID>
REM
REM We need to pickup the published contact.csv file from the JOB folder.  The JOBID should be passed to this
REM bat file.   The full path to the folder will be C:\ProgramData\CA\CA Test Data Manager Portal\Jobs\Job_<JOBID>\
REM
REM We need to strip the leading Byte Object Mark (BOM) from the .csv file as Data Loader will not tolerate it.
REM We will use a Powershell utility program "stripbom" to accomplish that.
REM Pass two parameters to stripbom - the first is the full path to the source file, the second is the full path to the target.

C:\Windows\System32\WindowsPowerShell\v1.0\powershell -executionpolicy bypass -File C:\TDM\SalesforceDemo\stripbom.ps1
"C:\ProgramData\CA\CA Test Data Manager Portal\Jobs\Job_%1\contact.CSV" "C:\TDM\SalesforceDemo\generated\contact.csv"

REM The file is now in the proper location and format (UTF-8)
REM Call Data Loader to Insert
REM As we are calling a .bat from a .bat, we must use the Call statement

Call C:\Users\<your_user>\dataloader\v48.0.0\bin\process.bat C:\Users\<your_user>\dataloader\v48.0.0\samples\conf\ contactInsertProcess
```

stripbom.ps1:

```
Param
  (
      [String] $SourceFilePath,
      [String] $TargetFilePath
  )
  Try
  {
    $utf8NoBomEncoding = New-Object System.Text.UTF8Encoding($False)
    [System.IO.File]::WriteAllLines($TargetFilePath, (Get-Content $SourceFilePath), $utf8NoBomEncoding)
    Write-Host "Removed UTF-8 BOM successfully"
  }
  Catch [Exception]
  {
    Write-Error $_.Exception.ToString()
  }
```

Create the Post-Publish Action for the Generator, referencing the .bat file and using the ~PUBJOBID~ variable to inform the .bat file where to pickup the .csv file:


Generators > ContactGen > Actions > Edit Publish Action

Editing PublishContactsToSalesforce

Description

Picks up the published file, strips the Byte Object Mark, and uses Salesforce data loader to insert into Salesforce

Action Type *

HOST

Command *

```
C:\TDM\SalesforceDemo\PublishContactsToSalesforce.bat
~PUBJOBID~
```

Execution

[✓] Wait for Completion

[ ] Success Required

Timeout (seconds) *

90

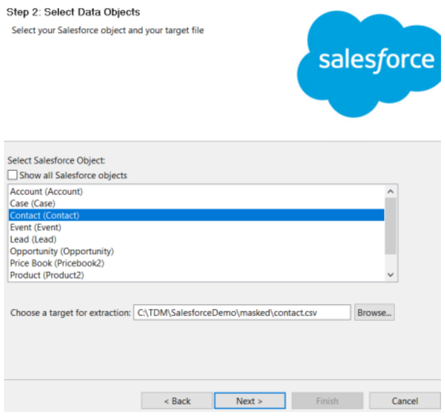
Execute Action

[ ] Pre [✓] Post

Setup / Masking

Masking is the process where existing records are exported, key Personally Identifiable Information (PII) is overwritten, then the record is imported in upsert mode to overwrite the PII in Salesforce.

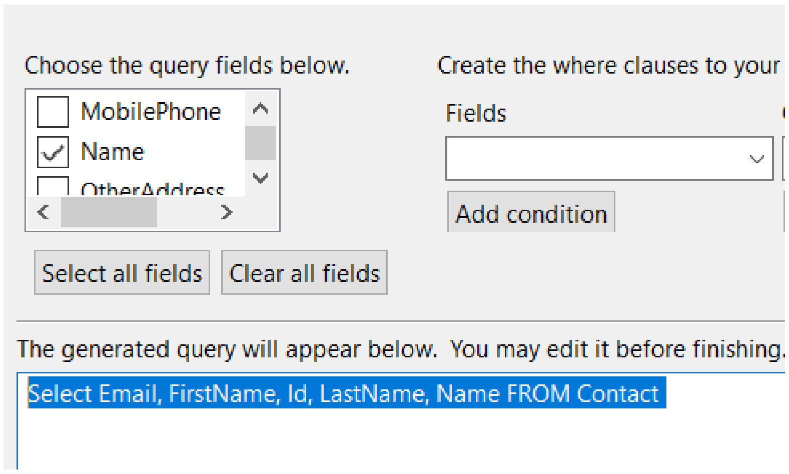
We will continue to use Contacts as the example for this User Story. Let's say we have reviewed the Contact fields, and decided that we wish to obscure a Contact by modifying the FIRSTNAME. But we note that the Contact's firstname is also used in the NAME (fullname) and the EMAIL fields. So we will want to prototype generating the exported .csv file using Data Loader to select the appropriate fields.   Launch Data Loader, and click Extract.


Step 2: Select Data Objects
Select your Salesforce object and your target file

Select Salesforce Object:
☐ Show all Salesforce objects
Account (Account)
Case (Case)
Contact (Contact)
Event (Event)
Lead (Lead)
Opportunity (Opportunity)
Price Book (Pricebook2)
Product (Product2)

Choose a target for extraction: C:\TDM\SalesforceDemo\masked\contact.csv

Select the fields we'll need for the masking.   NOTE: we need to bring down the LastName to be able to create the email address & the fullname.

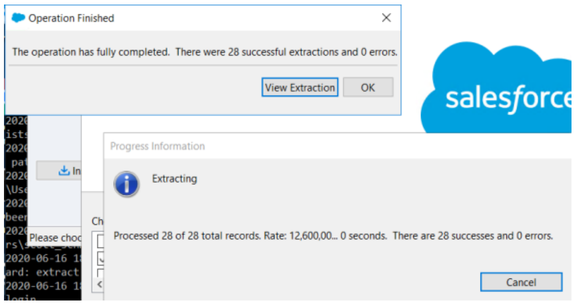

Choose the query fields below.
☐ MobilePhone
✓ Name
☐ OtherAddress

Create the where clauses to your
Fields

The generated query will appear below. You may edit it before finishing.

```
Select Email, FirstName, Id, LastName, Name FROM Contact
```

Select Email, FirstName, Id, LastName, Name FROM Contact


Operation Finished

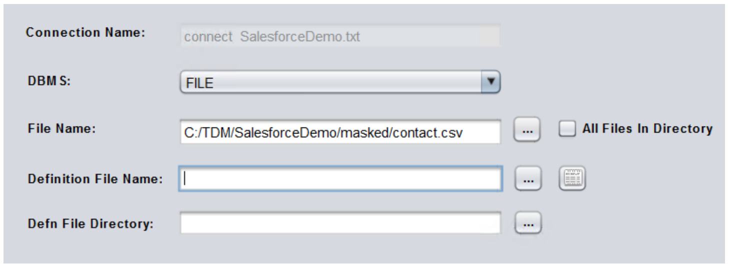
The operation has fully completed. There were 28 successful extractions and 0 errors.

Progress Information

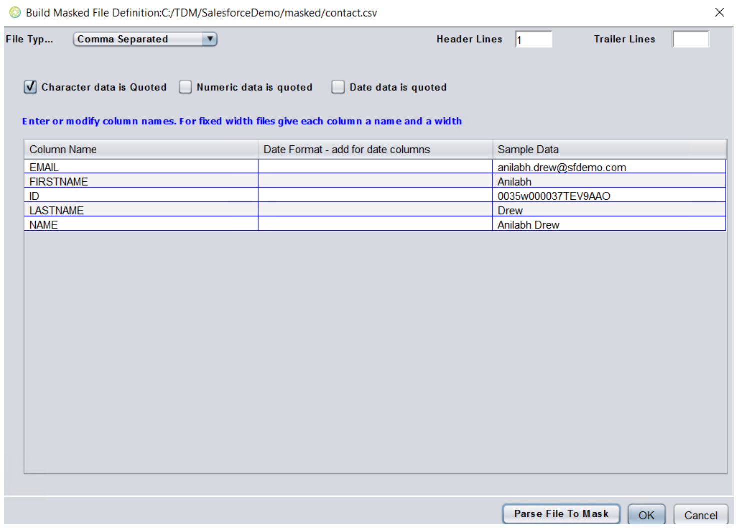
Extracting

Processed 28 of 28 total records. Rate: 12,600,00... 0 seconds. There are 28 successes and 0 errors.

The contact.csv is in the Masking directory “C:\TDM\SalesforceDemo\masked”. Start by launching the Fast Data Masker component of Broadcom Test Data Manager. When the Connection screen appears, select FILE as the DBMS, and select the contact.csv file as the File Name.


Connection Name:	connect SalesforceDemo.txt
DBMS:	FILE
File Name:	C:/TDM/SalesforceDemo/masked/contact.csv
Definition File Name:	
Defn File Directory:	

Select the right-most button next to the Definition File Name.
Leave the default File Type (Comma Separated), and enter “1” in the Header Lines field. Check any the “is quoted” boxes that apply (only Character in our case). Click on “Parse File to Mask” button. Enter the Date Format for Date Columns if you plan to mask them.


Build Masked File Definition:C:/TDM/SalesforceDemo/masked/contact.csv

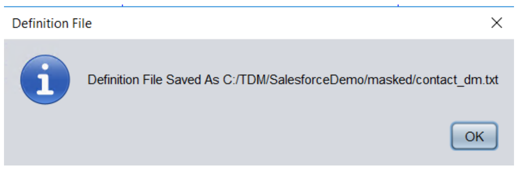
File Typ... Comma Separated   Header Lines 1   Trailer Lines

[✓] Character data is Quoted   [ ] Numeric data is quoted   [ ] Date data is quoted

Enter or modify column names. For fixed width files give each column a name and a width

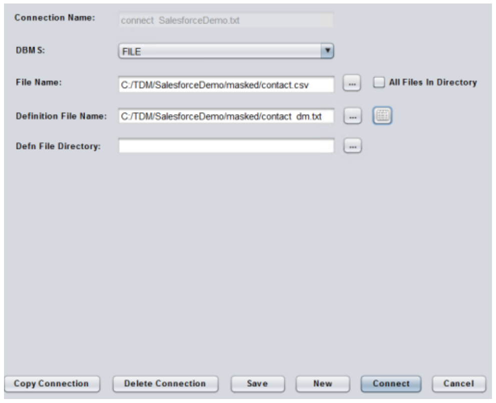
Column Name	Date Format - add for date columns	Sample Data
EMAIL		anilabh.drew@sfdemo.com
FIRSTNAME		Anilabh
ID		0035w000037TEV9AAO
LASTNAME		Drew
NAME		Anilabh Drew

Click OK. Note the name & location of the definition file:


Definition File

Definition File Saved As C:/TDM/SalesforceDemo/masked/contact_dm.txt

Now that the Definition File Name is filled in, click Connect:


Connection Name: connect SalesforceDemo.txt

DBMS: FILE

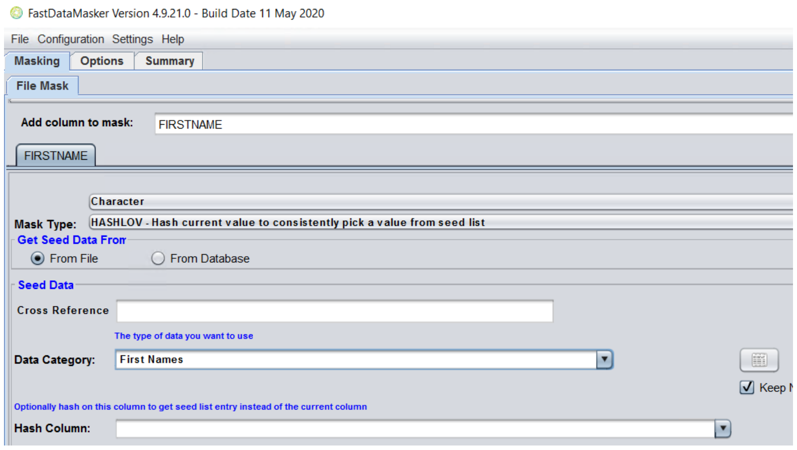
File Name: C:/TDM/SalesforceDemo/masked/contact.csv

All Files In Directory

Definition File Name: C:/TDM/SalesforceDemo/masked/contact_dm.txt

Defn File Directory:

You will be able to select the columns to mask. Pull down the list, and choose FIRSTNAME, and click the Add button. Set the top pulldown to Character, the Mask Type to HASHLOV, and the Data Category to First Names.


FastDataMasker Version 4.9.21.0 - Build Date 11 May 2020

Add column to mask: FIRSTNAME

Character

Mask Type: HASHLOV - Hash current value to consistently pick a value from seed list

Get Seed Data From

From File [✓] From Database

Seed Data

Cross Reference

The type of data you want to use

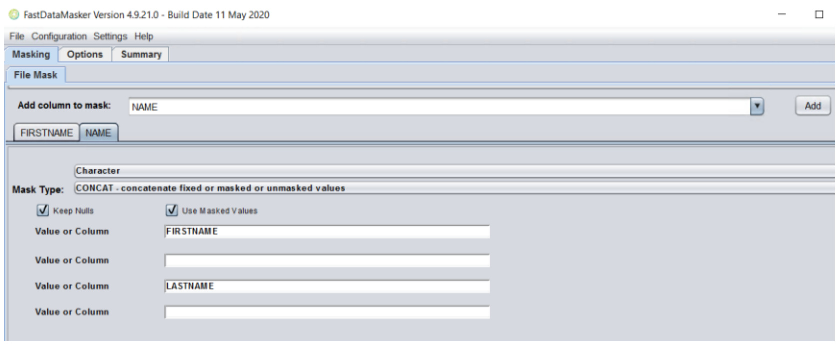
Data Category: First Names

Keep

Optionally hash on this column to get seed list entry instead of the current column

Hash Column:

Pull down the field list and select NAME, and click Add. (NAME is the person's fullname. As we are changing their firstname, we'll need to update both the fullname and email addresses).
Use the CONCAT function to mask the NAME field. Two key items: click the checkbox "Use Masked Values", and put a space in the 2nd Value or Column field.


FastDataMasker Version 4.9.21.0 - Build Date 11 May 2020
File Configuration Settings Help
Masking
Options
Summary
File Mask
Add
FIRSTNAME
NAME

Add column to mask: NAME

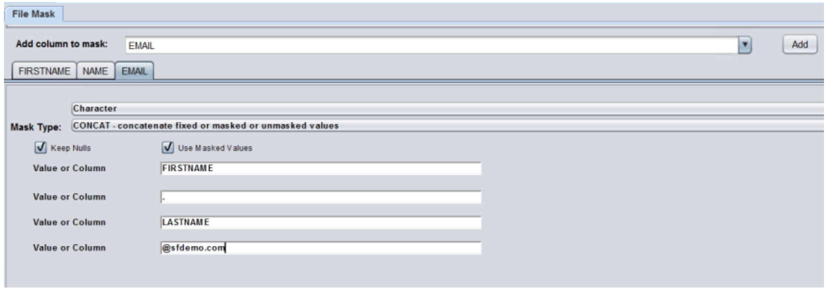
Character

Mask Type: CONCAT - concatenate fixed or masked or unmasked values

[✓] Keep Nulls [✓] Use Masked Values

Value or Column	FIRSTNAME
Value or Column	
Value or Column	LASTNAME
Value or Column	

Similarly, for EMAIL, click the Use Masked Values, put a period/dot in the 2nd Value or Column, and put @sfdemo.com in the 4th Value or Column field.



Add column to mask: EMAIL

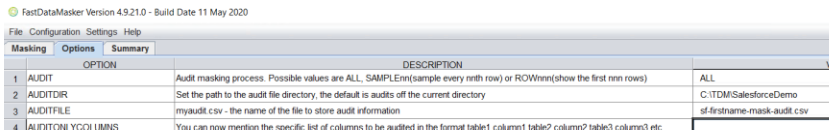
Character

Mask Type: CONCAT - concatenate fixed or masked or unmasked values

[✓] Keep Nulls [✓] Use Masked Values

Value or Column	FIRSTNAME
Value or Column	.
Value or Column	LASTNAME
Value or Column	@sfdemo.com

Switch to the Options tab and fill in the audit values to match your environment:


FastDataMasker Version 4.9.21.0 - Build Date 11 May 2020
File Configuration Settings Help
Masking
Options
Summary

	OPTION	DESCRIPTION	
1	AUDIT	Audit masking process. Possible values are ALL, SAMPLEnn(sample every nnth row) or ROWnnn(show the first nnn rows)	ALL
2	AUDITDIR	Set the path to the audit file directory, the default is audits off the current directory	C:\TDM\SalesforceDemo
3	AUDITFILE	myaudit.csv - the name of the file to store audit information	sf-firstname-mask-audit.csv
4	AUDITONLYCOLUMNS	You can now mention the specific list of columns to be audited in the format table1.column1,table2.column2,table3.column3 etc	

Switch to the Summary tab and click the “Save and Run Mask” button.

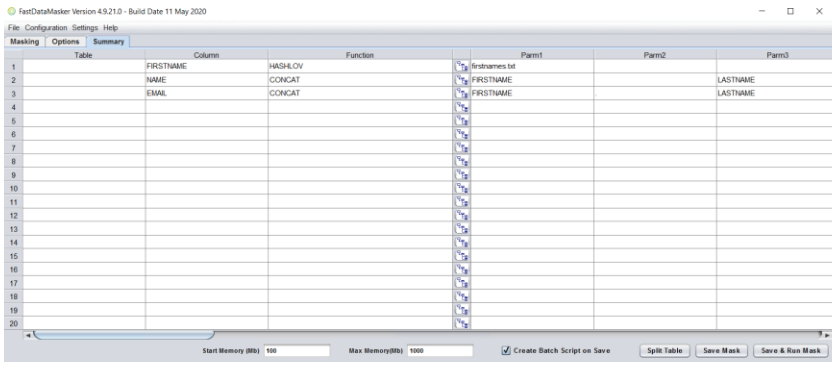
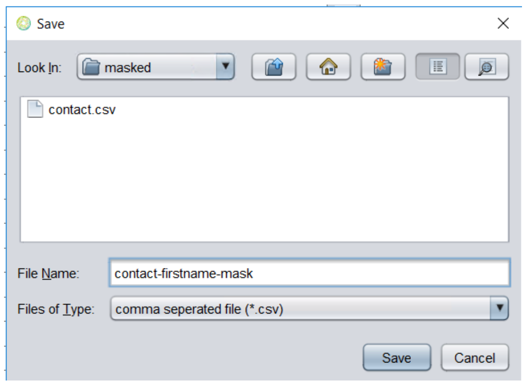

FastDataMasker Version 4.9.21.0 - Build Date 11 May 2020
File Configuration Settings Help
Masking
Options
Summary
Split Table
Save Mask
Save & Run Mask

	Table	Column	Function	Parm1	Parm2	Parm3
1		FIRSTNAME	HASHLOV	firstnames.txt		
2		NAME	CONCAT	FIRSTNAME		LASTNAME
3		EMAIL	CONCAT	FIRSTNAME	.	LASTNAME
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Start Memory (Mb) 100 Max Memory(Mb) 1000 [✓] Create Batch Script on Save

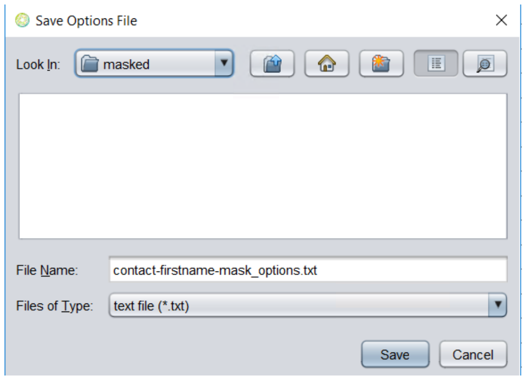
You will be prompted for a name to save the masking configuration:



File Name: contact-firstname-mask

Files of Type: comma seperated file (*.csv)

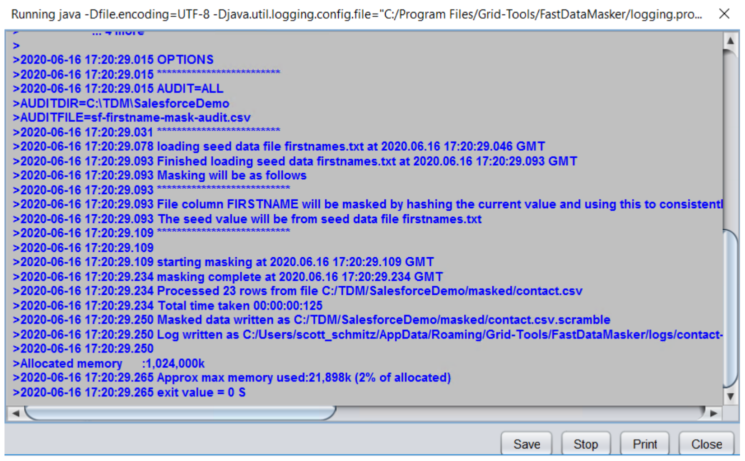
And then the Options file:



File Name: contact-firstname-mask_options.txt

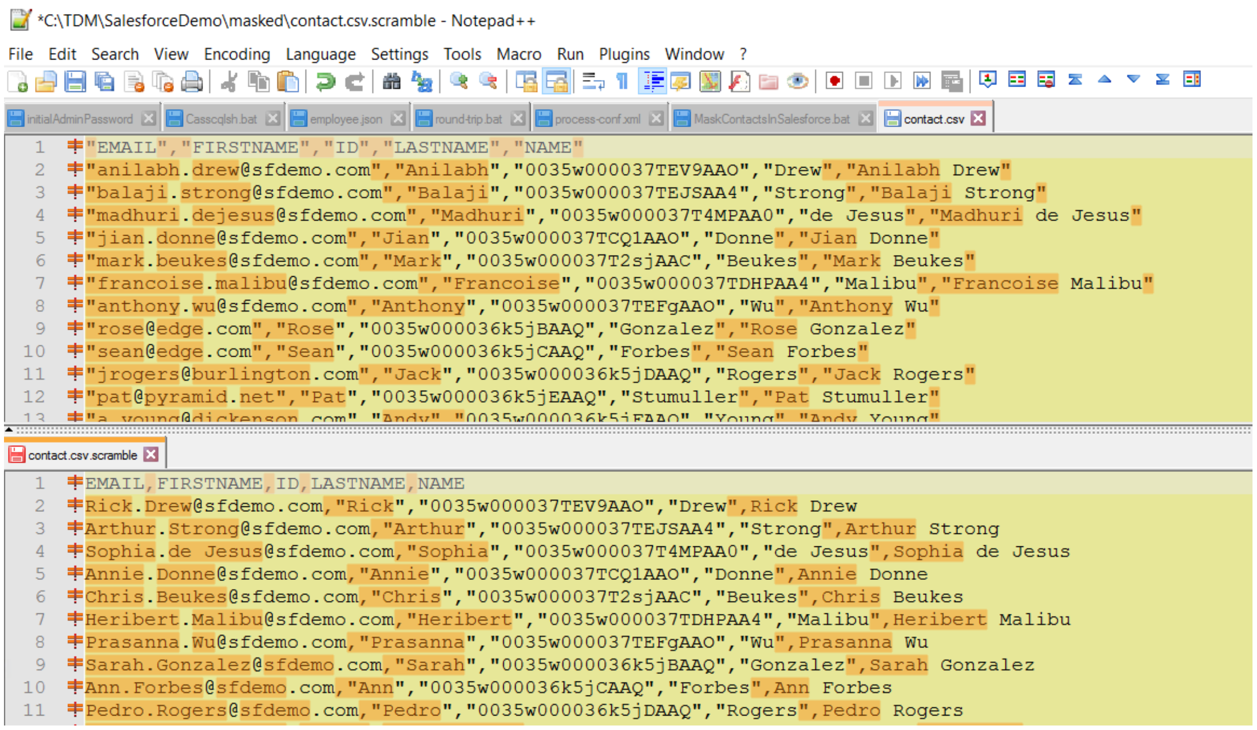
Files of Type: text file (*.txt)

You'll see the run window show success in masking the rows.



```
Running java -Dfile.encoding=UTF-8 -Djava.util.logging.config.file="C:/Program Files/Grid-Tools/FastDataMasker/logging.pro...
>
>2020-06-16 17:20:29.015 OPTIONS
>2020-06-16 17:20:29.015 ************************
>2020-06-16 17:20:29.015 AUDIT=ALL
>AUDITDIR=C:\TDM\SalesforceDemo
>AUDITFILE=sf-firstname-mask-audit.csv
>2020-06-16 17:20:29.031 ************************
>2020-06-16 17:20:29.078 loading seed data file firstnames.txt at 2020.06.16 17:20:29.046 GMT
>2020-06-16 17:20:29.093 Finished loading seed data firstnames.txt at 2020.06.16 17:20:29.093 GMT
>2020-06-16 17:20:29.093 Masking will be as follows
>2020-06-16 17:20:29.093 **************************
>2020-06-16 17:20:29.093 File column FIRSTNAME will be masked by hashing the current value and using this to consistentl
>2020-06-16 17:20:29.093 The seed value will be from seed data file firstnames.txt
>2020-06-16 17:20:29.109 **************************
>2020-06-16 17:20:29.109
>2020-06-16 17:20:29.109 starting masking at 2020.06.16 17:20:29.109 GMT
>2020-06-16 17:20:29.234 masking complete at 2020.06.16 17:20:29.234 GMT
>2020-06-16 17:20:29.234 Processed 23 rows from file C:/TDM/SalesforceDemo/masked/contact.csv
>2020-06-16 17:20:29.234 Total time taken 00:00:00:125
>2020-06-16 17:20:29.250 Masked data written as C:/TDM/SalesforceDemo/masked/contact.csv.scramble
>2020-06-16 17:20:29.250 Log written as C:/Users/scott_schmitz/AppData/Roaming/Grid-Tools/FastDataMasker/logs/contact-
>2020-06-16 17:20:29.250
>Allocated memory     :1,024,000k
>2020-06-16 17:20:29.265 Approx max memory used:21,898k (2% of allocated)
>2020-06-16 17:20:29.265 exit value = 0 S
```

The masked output is now in the contacts.csv.scramble file. If you have something like Notepad++ available with the Compare plugin, it is easy to see the deltas:

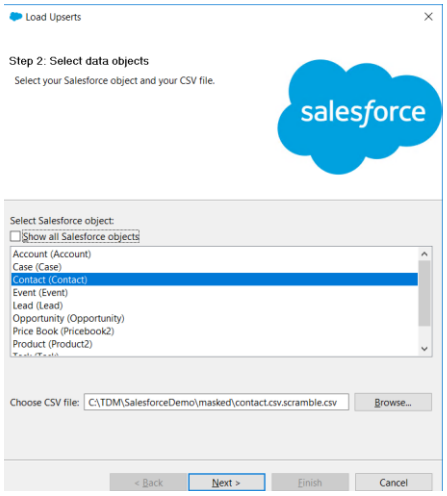

*C:\TDM\SalesforceDemo\masked\contact.csv.scramble - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
initialAdminPassword
Cassqlsh.bat
employee.json
round-trip.bat
process-conf.xml
MaskContactsInSalesforce.bat
contact.csv

```
"EMAIL","FIRSTNAME","ID","LASTNAME","NAME"
"anilabh.drew@sfdemo.com","Anilabh","0035w000037TEV9AAO","Drew","Anilabh Drew"
"balaji.strong@sfdemo.com","Balaji","0035w000037TEJSAA4","Strong","Balaji Strong"
"madhuri.dejesus@sfdemo.com","Madhuri","0035w000037T4MPAA0","de Jesus","Madhuri de Jesus"
"jian.donne@sfdemo.com","Jian","0035w000037TCQ1AAO","Donne","Jian Donne"
"mark.beukes@sfdemo.com","Mark","0035w000037T2sjAAC","Beukes","Mark Beukes"
"francoise.malibu@sfdemo.com","Francoise","0035w000037TDHPAA4","Malibu","Francoise Malibu"
"anthony.wu@sfdemo.com","Anthony","0035w000037TEFgAAO","Wu","Anthony Wu"
"rose@edge.com","Rose","0035w000036k5jBAAQ","Gonzalez","Rose Gonzalez"
"sean@edge.com","Sean","0035w000036k5jCAAQ","Forbes","Sean Forbes"
"jrogers@burlington.com","Jack","0035w000036k5jDAAQ","Rogers","Jack Rogers"
"pat@pyramid.net","Pat","0035w000036k5jEAAQ","Stumuller","Pat Stumuller"
"a.young@dickenson.com","Andy","0035w000036k5jFAAQ","Young","Andy Young"
```

```
EMAIL,FIRSTNAME,ID,LASTNAME,NAME
Rick.Drew@sfdemo.com,"Rick","0035w000037TEV9AAO","Drew",Rick Drew
Arthur.Strong@sfdemo.com,"Arthur","0035w000037TEJSAA4","Strong",Arthur Strong
Sophia.de Jesus@sfdemo.com,"Sophia","0035w000037T4MPAA0","de Jesus",Sophia de Jesus
Annie.Donne@sfdemo.com,"Annie","0035w000037TCQ1AAO","Donne",Annie Donne
Chris.Beukes@sfdemo.com,"Chris","0035w000037T2sjAAC","Beukes",Chris Beukes
Heribert.Malibu@sfdemo.com,"Heribert","0035w000037TDHPAA4","Malibu",Heribert Malibu
Prasanna.Wu@sfdemo.com,"Prasanna","0035w000037TEFgAAO","Wu",Prasanna Wu
Sarah.Gonzalez@sfdemo.com,"Sarah","0035w000036k5jBAAQ","Gonzalez",Sarah Gonzalez
Ann.Forbes@sfdemo.com,"Ann","0035w000036k5jCAAQ","Forbes",Ann Forbes
Pedro.Rogers@sfdemo.com,"Pedro","0035w000036k5jDAAQ","Rogers",Pedro Rogers
```

Email anilabh.drew@sfdemo.com became rick.drew@sfdemo.com
FIRSTNAME Anilabh became Rick
NAME Anilabh Drew became Rick Drew

We'll use the Data Loader to upsert the data back into Salesforce. Because Data Loader will filter for only .csv files, we'll rename contact.csv.scramble to contact.csv.scramble.csv
Select Upsert on Data Loader, select Contact, and select the scramble file.

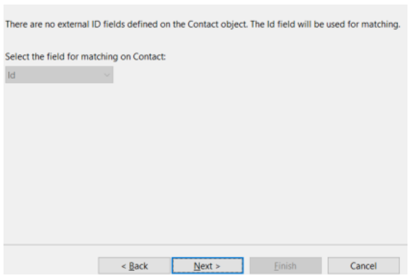


Step 2: Select data objects
Select your Salesforce object and your CSV file.

Select Salesforce object:
[ ] Show all Salesforce objects
Account (Account)
Case (Case)
Contact (Contact) ✓
Event (Event)
Lead (Lead)
Opportunity (Opportunity)
Price Book (Pricebook2)
Product (Product2)

Choose CSV file: C:\TDM\SalesforceDemo\masked\contact.csv.scramble.csv

As there are no external ID fields, it will use ID as the matching field.

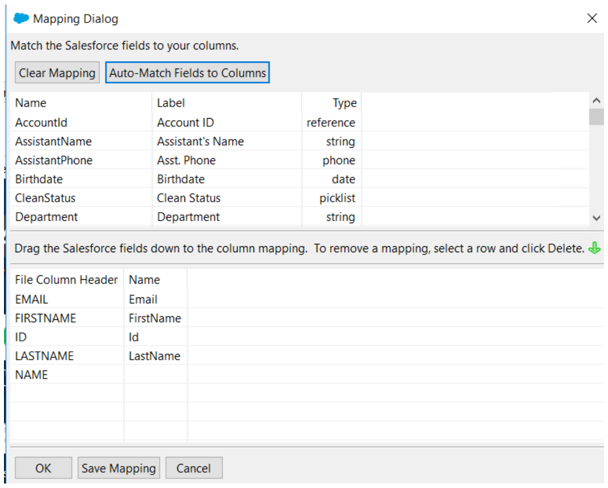


There are no external ID fields defined on the Contact object. The Id field will be used for matching.

Select the field for matching on Contact:

Id

Auto-match the fields; It appears that NAME is a generated field in Salesforce, so it is ignored for Upsert.



Mapping Dialog

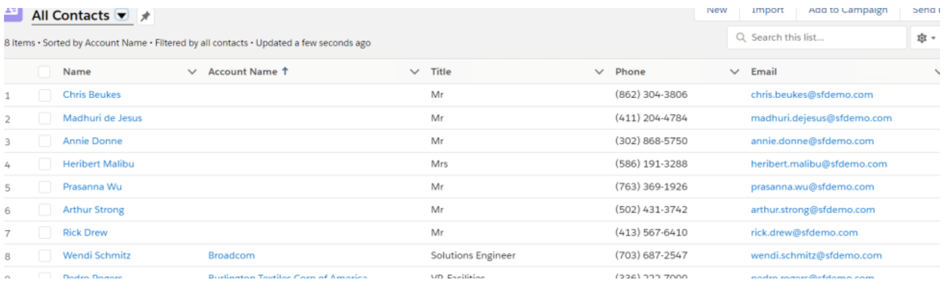
Match the Salesforce fields to your columns.

Name	Label	Type
AccountId	Account ID	reference
AssistantName	Assistant's Name	string
AssistantPhone	Asst. Phone	phone
Birthdate	Birthdate	date
CleanStatus	Clean Status	picklist
Department	Department	string

Drag the Salesforce fields down to the column mapping. To remove a mapping, select a row and click Delete.

File Column Header	Name
EMAIL	Email
FIRSTNAME	FirstName
ID	Id
LASTNAME	LastName
NAME	

Finish;   Once the import is complete, refresh Salesforce and you'll see the masked names & email addresses:



8 items • Sorted by Account Name • Filtered by all contacts • Updated a few seconds ago

	Name	Account Name	Title	Phone	Email
1	Chris Beukes		Mr	(862) 304-3806	chris.beukes@sfdemo.com
2	Madhuri de Jesus		Mr	(411) 204-4784	madhuri.dejesus@sfdemo.com
3	Annie Donne		Mr	(302) 868-5750	annie.donne@sfdemo.com
4	Heribert Malibu		Mrs	(586) 191-3288	heribert.malibu@sfdemo.com
5	Prasanna Wu		Mr	(763) 369-1926	prasanna.wu@sfdemo.com
6	Arthur Strong		Mr	(502) 431-3742	arthur.strong@sfdemo.com
7	Rick Drew		Mr	(413) 567-6410	rick.drew@sfdemo.com
8	Wendi Schmitz	Broadcom	Solutions Engineer	(703) 687-2547	wendi.schmitz@sfdemo.com

Masking Programmatically:

Now it's time to automate the process.   We'll start by creating two beans – the first to export the Contacts out of Salesforce so we can process the resulting .csv file, and the second to upsert the Contacts with the new "scramble" file results.

Backup the Data Loader\samples\conf "process-conf.xml" again to preserve the work we did with the import. The first bean should be configured something like this (highlighted areas for you to configure to your environment):

```
<bean id="csvContactExtractProcess"
        class="com.salesforce.dataloader.process.ProcessRunner"
        singleton="false">
     <description>csvContactExtract job gets contact records from salesforce and saves into a CSV file."</description>
       <property name="name" value="csvContactExtract"/>
       <property name="configOverrideMap">
          <map>
             <entry key="sfdc.debugMessages" value="false"/>
             <entry key="sfdc.debugMessagesFile" value="c:\TDM\SalesforceDemo\masked\csvContactExtract.log"/>
             <entry key="sfdc.endpoint" value="https://login.salesforce.com"/>
             <entry key="sfdc.username" value="yoursalesforceid"/>
             <!-- password specified below is invalid, please generate one using the encrypt.bat utility -->
             <entry key="sfdc.password" value="1111111111111111"/>
             <entry key="process.encryptionKeyFile" value="C:\Users\<youruser>\dataloader\v48.0.0\samples\conf\mysfdc.key"/>
             <entry key="sfdc.timeoutSecs" value="600"/>
             <entry key="sfdc.loadBatchSize" value="200"/>
             <entry key="sfdc.entity" value="Contact"/>
             <entry key="sfdc.extractionRequestSize" value="500"/>
             <entry key="sfdc.extractionSOQL" value="Select Email, FirstName, Id, LastName, Name FROM Contact"/>
             <entry key="process.operation" value="extract"/>
             <entry key="process.mappingFile" value="C:\TDM\SalesforceDemo\generated\contactMapping.sdl"/>
             <entry key="dataAccess.name" value="C:\TDM\SalesforceDemo\masked\contact.csv"/>
             <entry key="dataAccess.type" value="csvWrite"/>
          </map>
       </property>
   </bean>
```

The 2nd bean, for re-loading the masked data, should look something like this:

```
<bean id="contactUpdateProcess"
        class="com.salesforce.dataloader.process.ProcessRunner"
        singleton="false">
       <description>Contact Update job gets the masked Contact records from a CSV file and uploads them to salesforce using
'upsert'.</description>
       <property name="name" value="contactUpdateProcess"/>
       <property name="configOverrideMap">
          <map>
             <entry key="sfdc.debugMessages" value="false"/>
             <entry key="sfdc.debugMessagesFile" value="c:\TDM\SalesforceDemo\masked\csvContactUpdate.log"/>
             <entry key="sfdc.endpoint" value="https://login.salesforce.com"/>
             <entry key="sfdc.username" value="yoursalesforceid"/>
             <!-- password below has been encrypted using key file, therefore it will not work without the key setting:
process.encryptionKeyFile
             the password is not a valid encrypted value, please generate the real value using encrypt.bat utility -->
             <entry key="sfdc.password" value="555555555555"/>
             <entry key="process.encryptionKeyFile" value="C:\Users\<youruser>\dataloader\v48.0.0\samples\conf\mysfdc.key"/>
```

```
                <entry key="sfdc.timeoutSecs" value="600"/>
                <entry key="sfdc.loadBatchSize" value="200"/>
                <entry key="sfdc.externalIdField" value="Id"/>
                <entry key="sfdc.entity" value="Contact"/>
                <entry key="process.operation" value="upsert"/>
                <entry key="process.mappingFile" value="C:\TDM\SalesforceDemo\generated\contactMapping.sdl"/>
                <entry key="dataAccess.name" value="C:\TDM\SalesforceDemo\masked\contact.csv.scramble"/>
                <entry key="dataAccess.type" value="csvRead"/>
                <entry key="process.initialLastRunDate" value="2006-12-01T00:00:00.000-0800"/>
            </map>
        </property>
    </bean>
```

Since we are already experts at creating .bat files for Salesforce Data Loader operations, let's create & run this one:

MaskContactsInSalesforce.bat

```
REM

REM MaskContactsInSalesforce

REM

REM This script will extract Contact records from Salesforce into a local .csv file

REM It will then call Broadcom TDM's "Fast Data Masker" utility to mask the .csv file, producing a .scramble file as output

REM Last, it will Update/Upsert the Contact records in Salesforce from the masked records in the .scramble file.

REM We will first execute the Data Loader process to extract the Contact records to a local .csv file

REM As we are calling a .bat from a .bat, we must use the Call statement


Call C:\Users\<userid>\dataloader\v48.0.0\bin\process.bat C:\Users\<userid>\dataloader\v48.0.0\samples\conf\ csvContactExtractProcess


REM The file is now in the proper location and format (UTF-8)

REM We will call the FDM .bat file that was generated during the configuration process:


Call C:\TDM\SalesforceDemo\masked\contact-firstname-mask.bat


REM The .scramble file is now generated (still UTF-8)

REM We will execute the Data Loader process to Upsert the Salesforce Records


Call C:\Users\<userid>\dataloader\v48.0.0\bin\process.bat C:\Users\<userid>\dataloader\v48.0.0\samples\conf\ contactUpdateProcess
```