Module 1: System configuration

This toolkit is designed for Adobe Campaign Classic Developer Professional Exam Aspirants. There are six modules. Study each module per week to stick to schedule. Technical parts of applications are depicted in videos, about which you can learn more from Experience League. You can visit Get prep page to understand the contents and anticipate the learning journey.

This is Developer Professional Exam, toolkit Module 1. This module contains six sections.

1.1 Data schemas

Principles

To edit, create and configure the schemas, click the Administration > Configuration > Data schemas node of the Adobe Campaign client console.

NOTE

Built-in data schemas can only be deleted by an administrator of your Adobe Campaign Classic console.

The edit field shows the XML content of the source schema:
NOTE

The “Name” edit control lets you enter the schema key made up of the name and namespace. The “name” and “namespace” attributes of the root element of the schema are automatically updated in the XML editing zone of the schema.

The preview automatically generates the extended schema:
NOTE

When the source schema is saved, generation of the extended schema is automatically launched.

If you need to check the complete structure of a schema, you can use the preview tab. If the schema has been extended, you will then be able to visualize all its extensions. As a complement, the Documentation tab displays all the schema attributes and elements, and their properties (SQL Field, type/length, label, description). The Documentation tab only applies to generated schemas. For more on this, refer to the Regenerating schemas section.

Example: creating a contract table

In the following example, we want to create a new table for contracts in the database model of the Adobe Campaign database. This table lets you store first and last names and email addresses of holders and co-holders, for each contract.

To do this, you need to create the schema of the table and update the database structure to generate the corresponding table. Apply the following stages:

1. Edit the Administration > Configuration > Data schemas node of the Adobe Campaign tree and click New.
2. Choose the Create a new table in the data model option and click Next.
3. Specify a name for the table and a namespace.
NOTE

By default, schemas created by users are stored in the ‘cus’ namespace. For more on this, refer to Identification of a schema.

4. Create the content of the table. We recommend using the entry wizard to make sure no settings are missing. To do this, click the **Insert** button and choose the type of setting to be added.
5. Define the settings for the contract table:

6. `<srcSchema desc="Active contracts" img="ncm:channels.png" label="Contracts" labelSingular="Contract" mappingType="sql" name="Contracts" namespace="cus" xtkschema="xtk:srcSchema">

7.   <element desc="Active contracts" img="ncm:channels.png" label="Contracts" labelSingular="Contract"

8.     name="Contracts" autopk="true">

9.     <attribute name="holderName" label="Holder last name" type="string"/>

10.    <attribute name="holderFirstName" label="Holder first name" type="string"/>

11.    <attribute name="holderEmail" label="Holder email" type="string"/>

12.    <attribute name="co-holderName" label="Co-holder last name" type="string"/>

13.    <attribute name="co-holderFirstName" label="Co-holder first name" type="string"/>

14.    <attribute name="co-holderEmail" label="Co-holder email" type="string"/>

15.    <attribute name="date" label="Subscription date" type="date"/>

16.    <attribute name="noContract" label="Contract number" type="long"/>

17. </element>

18. </srcSchema>`
Add the type of contract and place an index on the contract number.

```
<srcSchema _cs="Contracts (cus)" desc="Active contracts" entitySchema="xtk:srcSchema" img="ncm:channels.png"
   label="Contracts" labelSingular="Contract" name="Contracts" namespace="cus" xtkschema="xtk:srcSchema">
  <enumeration basetype="byte" name="typeContract">
    <value label="Home" name="home" value="0"/>
    <value label="Car" name="car" value="1"/>
    <value label="Health" name="health" value="2"/>
    <value label="Pension fund" name="pension fund" value="2"/>
  </enumeration>
  <element autopk="true" desc="Active contracts" img="ncm:channels.png"
    label="Contracts">
    <attribute label="Holder last name" name="holderName" type="string"/>
    <attribute label="Holder first name" name="holderFirstName" type="string"/>
    <attribute label="Holder email" name="holderEmail" type="string"/>
    <attribute label="Co-holder last name" name="co-holderName" type="string"/>
    <attribute label="Co-holder first name" name="co-holderFirstName" type="string"/>
    <attribute label="Co-holder email" name="co-holderEmail" type="string"/>
    <attribute label="Subscription date" name="date" type="date"/>
    <attribute desc="Type of contract" enum="cus:Contracts:typeContract"
      label="Type of contract"
      name="type" type="byte"/>
    <attribute label="Contract number" name="noContract" type="long"/>
    <dbindex name="noContract" unique="true">
      <keyfield xpath="@noContract"/>
    </dbindex>
  </element>
</srcSchema>
```

19. Save the schema to generate the structure:
20. Update the database structure to create the table which the schema will be linked to. For more on this, refer to Updating the database structure.

Business.Adobe.com resources

1.2 About schema edition

Adobe Campaign employs Data Schemas to:

- Define how data objects within the application are tied to underlying database tables.
- Define links between the different data objects within the Campaign application.
- Define and describe the individual fields included in each object.

For a better understanding of Campaign built-in tables and their interaction, refer to this section.

Extending or creating schemas

To add a field or index or other element to one of the core data schemas in Campaign, such as the recipient table (nms:recipient), you have to extend that schema. For more on this, refer to the Extending a schema section.
To add an entirely new type of data that does not exist out-of-the-box in Adobe Campaign (a table of contracts for example) you can create a custom schema directly. For more on this, refer to the Data schemas section.

Once you have extended or created a schema to work in, the best practice is to define its XML content elements in the same order they appear in below.

**Enumerations**

Enumerations are defined first, before the main element of the schema. They allow you to display values in a list in order to restrict the choices that the user has for a given field.

Example:

```xml
<enumeration basetype="byte" name="exTransactionTypeEnum" default="store">
  <value label="Website" name="web" value="0"/>
  <value label="Call Center" name="phone" value="1"/>
  <value label="In Store" name="store" value="2"/>
</enumeration>
```

Copy

Toggle Text Wrapping
When defining fields, you can then use this enumeration like so:

```xml
<attribute desc="Type of Transaction" label="Transaction Type" name="transactionType" type="string" enum="exTransactionTypeEnum"/>
```

You can also employ user-managed enumerations (usually under Administration > Platform) to specify the values for a given field. These are effectively global enumerations, and a better choice if your enumeration may be used outside of the specific schema you are working in.

To find out more about enumerations, refer to the Enumerations and <enumeration> element sections.

**Index**

Indexes are the first elements declared in the main element of the schema.

They can be unique or not, and reference one or more fields.

Examples:

```xml
<dbindex name="email" unique="true">
  <keyfield xpath="@email"/>
</dbindex>
```

```xml
<dbindex name="lastNameAndZip">
  <keyfield xpath="@lastName"/>
  <keyfield xpath="location/@zipCode"/>
</dbindex>
```

The xpath attribute points to the field in your schema that you wish to index.

**IMPORTANT**

It is important to remember that the SQL query read performance gains provided by indexes also come with a performance hit on writing records. The indexes should therefore be used with precaution.

For more on indexes, refer to the Indexed fields section.
Keys

Every table must have at least one key, and often it is automatically established in the main element of the schema by using the `@autopk=true` attribute set to “true”.

The primary key can also be defined using the `internal` attribute.

Example:

```xml
<key name="householdId" internal="true">
    <keyfield xpath="@householdId"/>
</key>
```

In this example, instead of letting the `@autopk` attribute create a default primary key named “id” we are specifying our own “householdId” primary key.

**IMPORTANT**

When creating a new schema or during a schema extension, you need to keep the same primary key sequence value (@pkSequence) for the whole schema.

To find out more about keys, refer to the Management of keys section.

Attributes (Fields)

Attributes allow you to define the fields which make up your data object. You can use the Insert button in the schema edition toolbar to drop empty attribute templates into your XML where your cursor is. For more on this, refer to the Data schemas section.
The full list of attributes is available in the `<attribute>` element section. Here are some of the more commonly used attributes:

- @advanced
- @dataPolicy
- @default
- @desc
- @enum
- @expr
- @label
- @length
- @name
- @notNull
- @required
- @ref
- @xml
- @type

To view a table listing the mappings for the data types generated by Adobe Campaign for the different database management systems, refer to the Mapping the types of Adobe Campaign/DBMS data section.

For more information on each attribute, refer to the Attribute description section.

Examples

Example of defining a default value:
Example of using a common attribute as a template for a field also marked as mandatory:

```xml
<attribute name="mobile" label="Mobile" template="nms:common:phone" required="true" />
```

Example of a computed field that is hidden using the `@advanced` attribute:

```xml
<attribute name="domain" label="Email domain" desc="Domain of recipient email address" expr="GetEmailDomain([@email])" advanced="true" />
```

Example of an XML field also stored in an SQL field and which has an `@dataPolicy` attribute.

```xml
<attribute name="secondaryEmail" label="Secondary email address" length="100" xml="true" sql="true" dataPolicy="email" />
```

**IMPORTANT**

Although most attributes are linked according to a 1-1 cardinality to a physical field of the database, this is not the case for the XML fields or the computed fields. An XML field is stored in a memo field ("mData") of the table. A computed field however is created dynamically each time a query is started, it therefore only exists in the applicative layer.

**Links**

Links are some of the last elements in the main element of your schema. They define how all of the different schemas in your instance relate to one another.

Links are declared in the schema that contains the `foreign` key of the table to which it is linked.

There are three types of cardinality: 1-1, 1-N, and N-N. It is the 1-N type that is used by default.
Examples

An example of a 1-N link between the recipient table (out-of-the-box schema) and a table of custom transactions:

```xml
<element label="Recipient" name="lnkRecipient" revLink="lnkTransactions" target="nms:recipient" type="link"/>
```

An example of a 1-1 link between a custom schema “Car” (in the “cus” namespace) and the recipient table:

```xml
<element label="Car" name="lnkCar" revCardinality="single" revLink="recipient" target="cus:car" type="link"/>
```

Example of an external join between the recipient table and a table of addresses based on the email address and not a primary key:

```xml
<element name="emailInfo" label="Email Info" revLink="recipient" target="nms:address" type="link" externalJoin="true">
    <join xpath-dst="@address" xpath-src="@email"/>
</element>
```

Here “xpath-dst” corresponds to the primary key in the target schema and “xpath-src” corresponds to the foreign key in the source schema.

Audit trail

One useful element you may want to include at the bottom of your schema is a tracking element (Audit trail).

Use the example below to include fields relating to the creation date, the user that created the data, the date, and the author of the last modification for all data in your table:

```xml
<element aggregate="xtk:common:auditTrail" name="auditTrail"/>
```

Updating the database structure

Once your changes are completed and saved, any changes that may impact the SQL structure need to be applied to the database. To do this, use the database update wizard.
For more on this, refer to the Updating the database structure section.

NOTE

When modifications do not impact the database structure, you just need to regenerate schemas. To do this, select the schema(s) to update, right click and choose Actions > Regenerate selected schemas… . For more on this, refer to the Regenerating schemas section.

1.3 Schema structure

The basic structure of an <srcschema> is as follows:

```xml
<srcSchema>
  <enumeration>
    ...  //definition of enumerations
  </enumeration>

  <element>  //definition of the root <element> (mandatory)
```
<compute-string/> //definition of a compute-string
<dbindex>
  ...
  //definition of indexes
</dbindex>

.GetKey
  ...
  //definition of keys
</Key>
<sysFilter>
  ...
  //definition of filters
</sysFilter>
<attribute>
  ...
  //definition of fields
</attribute>

<element> //definition of sub-<element>
  <attribute> //(collection, links or XML)
    ...
    //and additional fields
  </attribute>
  ...
</element>

</element>

<methods> //definition of SOAP methods
  <method>
    ...
  </method>
  ...
</methods>
</srcSchema>
The XML document of a data schema must contain the `<srcSchema>` root element with the name and namespace attributes to populate the schema name and its namespace.

```
<srcSchema name="schema_name" namespace="namespace">
 ...
</srcSchema>
```

Let us use the following XML content to illustrate the structure of a data schema:

```
<recipient email="John.doe@aol.com" created="2009/03/12" gender="1">
  <location city="London"/>
</recipient>
```

With its corresponding data schema:

```
<srcSchema name="recipient" namespace="cus">
  <element name="recipient">
    <attribute name="email"/>
    <attribute name="created"/>
    <attribute name="gender"/>
    <element name="location">
      <attribute name="city"/>
    </element>
  </element>
</srcSchema>
```

**Description**

The point of entry of the schema is its main element. It is easy to identify because it has the same name as the schema, and it should be the child of the root element. The description of the content begins with this element.

In our example, the main element is represented by the following line:

```
<element name="recipient">
```
The elements `<attribute>` and `<element>` that follow the main element enable you to define the locations and names of the data items in the XML structure.

In our sample schema, these are:

```xml
<attribute name="email"/>
<attribute name="created"/>
<attribute name="gender"/>
<element name="location">
  <attribute name="city"/>
</element>
```

The following rules must be adhered to:

- Each `<element>` and `<attribute>` must be identified by name via the `name` attribute.

  **IMPORTANT**

  The name of the element should be concise, preferably in English, and include only authorized characters in accordance with XML naming rules.

- Only `<element>` elements can contain `<attribute>` elements and `<element>` elements in the XML structure.
- An `<attribute>` element must have a unique name within an `<element>`.
- The use of `<elements>` in multi-line data strings is recommended.

**Data types**

The data type is entered via the `type` attribute in the `<attribute>` and `<element>` elements.

A detailed list is available in the description of the `<attribute>` `element` and the `<element>` `element`.

When this attribute is not populated, `string` is the default data type unless the element contains child elements. If it does, it is used only to structure the elements hierarchically (`<location>` `element` in our example).

The following data types are supported in schemas:

- **string**: character string. Examples: a first name, a town, etc.
The size can be specified via the \texttt{length} attribute (optional, default value “255”).

- \texttt{boolean}: Boolean field. Example of possible values: true/false, 0/1, yes/no, etc.
- \texttt{byte, short, long}: integers (1 byte, 2 bytes, 4 bytes). Examples: an age, an account number, a number of points, etc.
- \texttt{double}: double-precision floating point number. Examples: a price, a rate, etc.
- \texttt{date, datetime}: dates and dates + times. Examples: a birth date, a purchase date, etc.
- \texttt{datetimenotz}: date + time without time zone data.
- \texttt{timespan}: durations. Example: seniority.
- \texttt{memo}: long text fields (multiple lines). Examples: a description, a comment, etc.
- \texttt{uuid}: “uniqueidentifier” fields to support a GUID (supported in Microsoft SQL Server only).

\textbf{NOTE}

To contain a \texttt{uuid} field in engines other than Microsoft SQL Server, the “newuuid()” function must be added and completed with its default value.

Here is our example schema with the types entered:

```xml
<srcSchema name="recipient" namespace="cus">
  <element name="recipient">
    <attribute name="email" type="string" length="80"/>
    <attribute name="created" type="datetime"/>
    <attribute name="gender" type="byte"/>
    <element name="location">
      <attribute name="city" type="string" length="50"/>
    </element>
  </element>
</srcSchema>
```

Mapping the types of Adobe Campaign/DBMS data

The table below lists the mappings for the types of data generated by Adobe Campaign for the different database management systems.
Properties

The `<elements>` and `<attributes>` elements of the data schema can be enriched with various properties. You can populate a label in order to describe the current element.

Labels and descriptions

- The **label** property lets you enter a brief description.

**NOTE**

The label is associated with the current language of the instance.

Example:

```xml
<attribute name="email" type="string" length="80" label="Email"/>
```

The label can be seen from the Adobe Campaign client console input form:
The `desc` property lets you enter a long description.

The description can be seen from the input form in the status bar of the Adobe Campaign client console main window.

**NOTE**

The description is associated with the current language of the instance.

**Example:**

```xml
<attribute name="email" type="string" length="80" label="Email" desc="Email of recipient"/>
```

**Default values**

The `default` property lets you define an expression returning a default value on content creation.

The value must be an expression compliant with XPath language. For more on this, refer to [Referencing with XPath](#).

**Example:**

- Current date: `default="GetDate()"`
- Counter: `default="FRM'+CounterValue('myCounter')"`

In this example, the default value is constructed using the concatenation of a string and calling the `CounterValue` function with a free counter name. The number returned is incremented by one at each insertion.

**NOTE**

In the Adobe Campaign client console, the *Administration>Counters* node is used to manage counters.

To link a default value to a field, you can use the `<default>` or `<sqldefault>` field. `</sqldefault> </default>`

`<default>`: allows you to pre-fill the field with a default value when creating entities. The value will not be a default SQL value.
allows you have an added value when creating a field. This value appears as an SQL result. During a schema update, only the new records will be impacted by this value.

Enumerations

Free enumeration

The userEnum property lets you define a free enumeration to memorize and display the values entered via this field. The syntax is as follows:

userEnum="name of enumeration"

The name given to the enumeration can be chosen freely and shared with other fields.

These values are shown in a drop-down list from the input form:

NOTE

In the Adobe Campaign client console, the Administration > Enumerations node is used to manage enumerations.

Set enumeration

The enum property lets you define a fixed enumeration used when the list of possible values is known in advance.

The enum attribute refers to the definition of an enumeration class populated in the schema outside the main element.

Enumerations allow the user to select a value from a drop-down list instead of entering the value in a regular input field:
Example of an enumeration declaration in the data schema:

```
<enumeration name="gender" basetype="byte" default="0">
    <value name="unknown" label="Not specified" value="0"/>
    <value name="male" label="male" value="1"/>
    <value name="female" label="female" value="2"/>
</enumeration>
```

An enumeration is declared outside the main element via the `<enumeration>` element.

The enumeration properties are as follows:

- **baseType**: type of data associated with the values,
- **label**: description of the enumeration,
- **name**: name of the enumeration,
- **default**: default value of the enumeration.

The enumeration values are declared in the `<value>` element with the following attributes:

- **name**: name of the value stored internally,
- **label**: label displayed via the graphical interface.

**dbenum enumeration**

- The `dbenum` property lets you define an enumeration whose properties are similar to those of the `enum` property.

  However, the **name** attribute does not store the value internally, it stores a code which lets you extend the concerned tables without modifying their schema.

  The values are defined via the **Administration>Enumerations** node.

  This enumeration is used for specifying the nature of campaigns, for example.
Example

Here is our example schema with the properties filled in:

```xml
<srcSchema name="recipient" namespace="cus">
  <enumeration name="gender" basetype="byte">
    <value name="unknown" label="Not specified" value="0"/>
    <value name="male" label="male" value="1"/>
    <value name="female" label="female" value="2"/>
  </enumeration>

  <element name="recipient">
    <attribute name="email" type="string" length="80" label="Email" desc="Email of recipient"/>
    <attribute name="created" type="datetime" label="Date of creation" default="GetDate()"/>
    <attribute name="gender" type="byte" label="gender" enum="gender"/>
  </element>

</srcSchema>
```
Collections

A collection is a list of elements with the same name and the same hierarchical level.

The **unbound** attribute with the value “true” lets you populate a collection element.

**Example**: definition of the `<group>` collection element in the schema.

```xml
<element name="group" unbound="true" label="List of groups">
  <attribute name="label" type="string" label="Label"/>
</element>
```

With projection of the XML content:

```xml
<group label="Group1"/>
<group label="Group2"/>
```

Referencing with XPath

The XPath language is used in Adobe Campaign to reference an element or attribute belonging to a data schema.

XPath is a syntax that lets you locate a node in the tree of an XML document.

Elements are designated by their name, and attributes are designated by the name preceded by the character “@”.

**Example**:

- `@email`: selects the email,
- `location/@city`: selects the “city” attribute under the `<location>` element
- `.../@email`: selects the email address from the parent element of the current element
- `group[1]/@label`: selects the “label” attribute that is the child of the first `<group>` collection element
• **group[@label='test1']**: selects the “label” attribute that is the child of the `<group>` element and contains the value “test1”

**NOTE**

An additional constraint is added when the path crosses a sub-element. In this case, the following expression must be placed between brackets:

- `location/@city` is not valid; please use `[location/@city]`
- `[@email]` and `@email` are equivalent

It is also possible to define complex expressions, such as the following arithmetic operations:

- `@gender+1`: adds 1 to the content of the `gender` attribute,
- `@email + '(+@created+)')`: constructs a string by taking the value of the email address added to the creation date between parentheses (for the string type, put the constant in quotes).

High-level functions have been added to the expressions in order to enrich the potential of this language.

You can access the list of available functions via any expression editor in the Adobe Campaign client console:

![List of available functions](image)

**Example:**

- `GetDate()`: returns the current date
- `Year(@created)`: returns the year of the date contained in the “created” attribute.
- `GetEmailDomain(@email)`: returns the domain of the email address.
Building a string via the compute string

A Compute string is an XPath expression used to construct a string representing a record in a table associated with the schema. Compute string is mainly used in the graphical interface to display the label of a selected record.

The Compute string is defined via the `<compute-string>` element under the main element of the data schema. An expr attribute contains an XPath expression to calculate the display.

Example: compute string of the recipient table.

```xml
<srcSchema name="recipient" namespace="nms">
  <element name="recipient">
    <compute-string expr="@lastName + ' ' + @firstName +' (' + @email + ')"/>
    ...
  </element>
</srcSchema>
```

Result of the computed string for a recipient: Doe John (john.doe@aol.com)

**NOTE**

If the schema does not contain a Compute string, a Compute string is populated by default with the values of the primary key of the schema.

### 1.4 List of Campaign Classic options

The Administration / Platform / Options node allows you to configure Adobe Campaign options. Some of them are built-in when installing Campaign, and others can be added manually when needed. Available options vary according to the packages installed with your instance.

**CAUTION**

- Options not listed in this page are internal only and must not be modified.
- Modifying or updating Adobe Campaign options can be performed by experts users only.

**Delivery**
<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverability_LastBroadLogMsgDate</td>
<td>Date of the last broadLogMsg retrieved from the deliverability instance.</td>
</tr>
<tr>
<td>Deliverability_LastBroadLogMsgSent</td>
<td>Date of the last broadLogMsg sent to the deliverability instance.</td>
</tr>
<tr>
<td>DmrRendering_cuid</td>
<td>Delivery reports identifier. Please contact support to obtain your identifier.</td>
</tr>
<tr>
<td>DmrRendering_SeedTargets</td>
<td>List of schemas for which you want to use test addresses for inbox Rendering. (element names are separated with commas) E.g.: custom_nms_recipient,</td>
</tr>
<tr>
<td>EMTA_BCC ADDRESS</td>
<td>BCC email address to which the Enhanced MTA will send a raw copy of the sent emails.</td>
</tr>
<tr>
<td>NMS_ActivateOwnerConfirmation</td>
<td>Lets you allow the operator in charge of the delivery to confirm the send, if a specific operator or group of operators is designated for starting a delivery in the delivery's properties.</td>
</tr>
<tr>
<td></td>
<td>To do this, activate the option by entering &quot;1&quot; as the value. To deactivate this option, enter &quot;0&quot;.</td>
</tr>
<tr>
<td></td>
<td>The send confirmation process will then function as default. only the operator or group of operators designated for the send in the delivery properties (or an administrator) will be able to confirm and carry out the send. See this section.</td>
</tr>
<tr>
<td>Nms_DefaultRcpSchema</td>
<td>Adobe Campaign uses a 'Nms_DefaultRcpSchema' global variable to dialog with the default recipient database (nms-recipient).</td>
</tr>
<tr>
<td></td>
<td>The option value must correspond to the name of the schema which matches the external recipient table.</td>
</tr>
<tr>
<td>NmsBilling_MainActionThreshold</td>
<td>Minimum number of recipients in order for a delivery to be considered as the main one in the billing report.</td>
</tr>
<tr>
<td>NmsBroadcast_DefaultProvider</td>
<td>Default routing service provider for the new templates.</td>
</tr>
<tr>
<td>NmsBroadcast_LogsPerTransac</td>
<td>Minimal batch size (number of rows) for the insertion of broadLogs during a delivery preparation.</td>
</tr>
<tr>
<td>NmsBroadcast_MaxDelayPerTransac</td>
<td>Batch duration threshold (number of milliseconds) under which the batch size for the insertion of broadLogs is doubled during a delivery preparation.</td>
</tr>
<tr>
<td>NmsBroadcast_MidAnalyzeBatchSize</td>
<td>Grouping size of delivery parts when analyzing mid-sourcing deliveries.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>NmsBroadcast_MsgValidityDuration</td>
<td>Default delivery period for a delivery (in seconds).</td>
</tr>
<tr>
<td>NmsBroadcast_RegexpRules</td>
<td>Regular expressions for normalizing delivery messages.</td>
</tr>
<tr>
<td>NmsBroadcast_RemoveBlacklist</td>
<td>Entering &quot;1&quot; as the value lets you exclude recipients who no longer wish to be contacted.</td>
</tr>
<tr>
<td>NmsBroadcast_RemoveDuplicatesRecipients</td>
<td>Entering &quot;1&quot; as the value lets you automatically ignore doubles.</td>
</tr>
<tr>
<td>NmsDelivery_ErrorAddressMasks</td>
<td>Lets you define the syntax of the Error address used when replying to a message.</td>
</tr>
<tr>
<td>NmsDelivery_FromAddressMasks</td>
<td>Lets you define the syntax of the From address used when sending a message.</td>
</tr>
<tr>
<td>NmsDelivery_ImagesserverTimeout</td>
<td>Lets you define a timeout limit (in seconds) for getting a response from the server when retrieving an image downloaded from a personalized URL and attached to an email. If this value is exceeded, the message cannot be sent. The default value is 60 seconds.</td>
</tr>
<tr>
<td>NmsDelivery_MaxDownloadedImageSize</td>
<td>Lets you define the maximum size (in bytes) allowed for an image downloaded from a personalized URL and attached to an email. The default value is 100,000 bytes (100 KB). When sending a proof and downloading the image(s) to process the email, if the size of an image exceeds this value or if there is a downloading issue, an error will be displayed in the Delivery logs and the proof delivery will fail.</td>
</tr>
<tr>
<td>NmsDelivery_MaxRecommendedAttachments</td>
<td>Lets you set a maximum number of attachments in an email or transactional email template. If this value is exceeded, a warning will be displayed in the delivery analysis logs or when publishing the transactional email template. The default value is 1 attachment.</td>
</tr>
<tr>
<td>NmsDelivery_MaxRetry</td>
<td>Maximum number of retries during analysis.</td>
</tr>
<tr>
<td>NmsDelivery_PublishingScript</td>
<td>Publication script.</td>
</tr>
<tr>
<td>NmsDelivery_NoCountBroadcastMsgPush</td>
<td>Disable the broadcast/mg count for push messages.</td>
</tr>
<tr>
<td>NmsDeliveryWizard_ShowDeliveryWeight</td>
<td>Display the message weight in the delivery wizard.</td>
</tr>
<tr>
<td>NmsEmail_DefaultFromAddr</td>
<td>Default 'From' email address at instance's level used for email delivery if left empty by user.</td>
</tr>
<tr>
<td>NmsEmail_DefaultReplyToAddr</td>
<td>Default 'Reply-To' email address at instance's level used for email delivery if left empty by user.</td>
</tr>
<tr>
<td>NmsEmail_DefaultToAddr</td>
<td>Default 'To' email address at instance's level used for email delivery if left empty by user.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>NmsEmail_ExpOrganization</td>
<td>Common name of the customer. Used in some warning messages displayed to the recipients. “You are receiving this message because you have been in contact with ‘Organization’ or an affiliated company. To no longer receive messages from ‘Organization’”</td>
</tr>
<tr>
<td>NmsEmail_FromName</td>
<td>Default ‘from’ email label at instance’s level used for email delivery if left empty by user.</td>
</tr>
<tr>
<td>NmsEmail_ReplyToName</td>
<td>Default ‘reply-to’ email label at instance’s level used for email delivery if left empty by user.</td>
</tr>
<tr>
<td>NmsEmail_RetryCount</td>
<td>Period between two retries of an email message (in seconds).</td>
</tr>
<tr>
<td>NmsEmail_RetryPeriod</td>
<td>Period of retries for email messages.</td>
</tr>
<tr>
<td>NmsForecast_MegaWeightFormula</td>
<td>Formula used to calculate the weighting of a message for a provisional delivery.</td>
</tr>
<tr>
<td>NmsInMail_AllowlistEmails</td>
<td>List of authorized forwarding email addresses (from the inbound mail processing module). The addresses have to be separated by commas (or * to allow all). E.g. <a href="mailto:xyz@abc.com">xyz@abc.com</a>, <a href="mailto:pqr@abc.com">pqr@abc.com</a>.</td>
</tr>
<tr>
<td>NmsLine_AESKey</td>
<td>AES key used in the 'listimage' servlet to encode the URLs (LINE channel).</td>
</tr>
<tr>
<td>NmsNPai_EmailMaxError</td>
<td>On channel ‘email’ (use as default): Maximal number of errors that is accepted, for SOFT errors during sending before putting the recipient into quarantine.</td>
</tr>
<tr>
<td>NmsNPai_EmailSignificantErrorDelay</td>
<td>On channel ‘email’ (use as default): Minimal period to spent since the previous referenced SOFT error, before taking into account a new SOFT error.</td>
</tr>
<tr>
<td>NmsNPai_MobileMaxError</td>
<td>On channel ‘mobile’*: Maximal number of errors that is accepted, for SOFT errors during sending before putting the recipient into quarantine.</td>
</tr>
<tr>
<td>NmsNPai_MobileSignificantErrorDelay</td>
<td>On channel ‘mobile’*: Minimal period to spent since the previous referenced SOFT error, before taking into account a new SOFT error.</td>
</tr>
<tr>
<td>NmsMidSourcing_LogsPeriodHour</td>
<td>Allows a maximum period (expressed in hours) to be specified as to limit the number of backlogs recovered every time the synchronization workflow is executed.</td>
</tr>
<tr>
<td>NmsMidSourcing_PrepareFlow</td>
<td>Maximum number of calls in MidSourcing session, which can be run in parallel (3 by default).</td>
</tr>
<tr>
<td>NmsMTA_Agent_Delay</td>
<td>Custom delay (in minutes) after which a delivery is considered as ‘delayed’, default being 10 minutes.</td>
</tr>
<tr>
<td>Resource Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NimsOperation_DeliveryPreparationWindow</td>
<td>This option is used by the operationMgt technical workflow when counting the number of running deliveries.</td>
</tr>
<tr>
<td></td>
<td>It allows you to define the number of days above which deliveries with inconsistent status will be excluded from the count of running deliveries.</td>
</tr>
<tr>
<td></td>
<td>By default, the value is set to &quot;7&quot;, meaning that inconsistent deliveries older than 7 days will be excluded.</td>
</tr>
<tr>
<td>NimsPaper_SenderLine1</td>
<td>Line 1 of the sender's address.</td>
</tr>
<tr>
<td>NimsPaper_SenderLine3</td>
<td>Line 3 of the sender's address.</td>
</tr>
<tr>
<td>NimsPaper_SenderLine4</td>
<td>Line 4 of the sender's address.</td>
</tr>
<tr>
<td>NimsPaper_SenderLine6</td>
<td>Line 6 of the sender's address.</td>
</tr>
<tr>
<td>NimsPaper_SenderLine7</td>
<td>Line 7 of the sender's address.</td>
</tr>
<tr>
<td>NimsServer_MirrorPageURL</td>
<td>URL of the mirror page server (by default, should be identical to NimsTracking_ServerURL). It is the default value of email deliveries when the URL is not specified in the routing definition.</td>
</tr>
<tr>
<td>NimsSMS_Priority</td>
<td>Parameters of sent SMS messages: information transmitted to the SMS gateway to indicate the message priority.</td>
</tr>
<tr>
<td>NimsSMS_RetryCount</td>
<td>Number of retries when sending SMS messages.</td>
</tr>
<tr>
<td>NimsSMS_RetryPeriod</td>
<td>Period during which retries of SMS messages will be performed.</td>
</tr>
<tr>
<td>NimsUserAgentStats_LastConsolidation</td>
<td>Last consolidation date for NimsUserAgent statistics.</td>
</tr>
<tr>
<td>NimsWebSegments_LastStates</td>
<td>Name of the option which contains the web segments and their states.</td>
</tr>
<tr>
<td>XtkBarcode_SpecialChar</td>
<td>Enable/disable support for special characters for Code120.</td>
</tr>
<tr>
<td>XtkEmail_Characters</td>
<td>Valid characters for an email address.</td>
</tr>
<tr>
<td>XtkSecurity_Restrict_EditXML</td>
<td>Add this option with the &quot;0&quot; value to disable the edition of deliveries' XML code (right-click / Edit XML source or CTRL + F4 shortcut).</td>
</tr>
</tbody>
</table>

**Resources**
<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NmsResourcesDir</td>
<td>Location of resources for publication in the Adobe Campaign client console. See this section.</td>
</tr>
<tr>
<td>NmsResourcesDirPreview</td>
<td>Location of resources for previewing in the Adobe Campaign client console. See this section.</td>
</tr>
<tr>
<td>NmsDelivery_DefaultIgnoredImage</td>
<td>List of URL masks for the images skipped during upload.</td>
</tr>
<tr>
<td>NmsDelivery_ImagePublishing</td>
<td>Configuration of image uploading. The values can be none / tracking / script / list (can be overridden by operator’s optional settings).</td>
</tr>
<tr>
<td>NmsDelivery_ImageSubDirectory</td>
<td>Folder in which the images on the server are to be stored.</td>
</tr>
<tr>
<td>NmsServer_LogoPath</td>
<td>Space to display logos.</td>
</tr>
<tr>
<td>NmsPublishingDir</td>
<td>Root folder for publications. For more on HTML and Text contents generation, refer to this section.</td>
</tr>
<tr>
<td>XtkImageUrl</td>
<td>Lets you define the server on which the images used in the deliveries are stored to enable the browser to get them. For build versions &lt;= 5098, we use the URL of the images that were uploaded onto the instance. For build versions &gt; 5098, we use instead the delivery’s public URL or the XKFFileRes_Public_URL option’s URL.</td>
</tr>
<tr>
<td>NmsDelivery_MediaInstanceId</td>
<td>Lets you configure the instance name for image uploading.</td>
</tr>
<tr>
<td>NmsDelivery_MediaPassword</td>
<td>Lets you configure the password for image uploading.</td>
</tr>
<tr>
<td>NmsDelivery_MediaServers</td>
<td>Lets you configure the media URL(s) for image uploading.</td>
</tr>
<tr>
<td>NmsBroadcast_MsgWebValidityDuration</td>
<td>Default validity duration for online resources of a delivery (in seconds).</td>
</tr>
<tr>
<td>XKFFileRes_Public_URL</td>
<td>New URL for public resource files.</td>
</tr>
</tbody>
</table>

**Campaign & workflow management**

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CrmMarketingActivityWindow</td>
<td>Marketing history shown for this number of months.</td>
</tr>
<tr>
<td>NmsOperation_Duration</td>
<td>Default validity period of a campaign (in seconds).</td>
</tr>
<tr>
<td>NmsOperation_LimitConcurrency</td>
<td>Maximum number of delivery/workflow/hypothesis/simulation jobs that can be processed at a time, started by operationMgt workflow.</td>
</tr>
<tr>
<td>NmsOperation_OperationMgtDebug</td>
<td>Allows you to monitor the operationMgt technical workflow execution. When activated (value &quot;1&quot;), the execution information are logged in the workflow audit logs.</td>
</tr>
<tr>
<td>NmsOperation_TimeRange</td>
<td>Time period for targeting and extraction conditions in scheduled mode.</td>
</tr>
<tr>
<td>Workflow_AnalysisThreshold</td>
<td>Number of affected records after which table statistics are automatically recomputed.</td>
</tr>
<tr>
<td>XtkReport_Logo</td>
<td>Logo to be displayed in the top right-hand corner of the reports exported.</td>
</tr>
<tr>
<td>NmsServer_PausedWorkflowPeriod</td>
<td>Number of days to wait between checks for paused workflows.</td>
</tr>
<tr>
<td>NmsCampaign_Activate_OwnerConfirmation</td>
<td>Activate the deliveries validation by the owner of the operation by entering “Y” as the value. To deactivate this option, enter “N”.</td>
</tr>
<tr>
<td>NmsAsset_JavascriptExt</td>
<td>Additional JS library to load in workflow’s activity “Marketing resource notifications”.</td>
</tr>
</tbody>
</table>

**Security**
### Message Center

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MC_EnrichmentCustomJs</strong></td>
<td>JavaScript library to be personalized for enriching events. Must contain the implementation of these two functions:</td>
</tr>
<tr>
<td></td>
<td>• enrichREvent(allEventId) : enriches and saves events in the database (where allEventId corresponds to the table of real time events processed).</td>
</tr>
<tr>
<td></td>
<td>• enrichBatchREvent(allEventId) : enriches and saves events in the database (where allEventId corresponds to the ID of batch events processed).</td>
</tr>
<tr>
<td><strong>MC_LastUpdateFromRL</strong></td>
<td>Date of the last event status update via delivery logs.</td>
</tr>
<tr>
<td><strong>MC_RoutingCustomJs</strong></td>
<td>JavaScript library to be personalized for routing events. Must contain the implementation of these two functions:</td>
</tr>
<tr>
<td></td>
<td>• dispatchREvent(allEventId) : returns the internal name of the transactional message selected to process the real time event (where allEventId corresponds to the ID of the real time event processed).</td>
</tr>
<tr>
<td></td>
<td>• dispatchBatchREvent(allEventId) : returns the internal name of the transactional message selected to process the batch event (where allEventId corresponds to the ID of the batch event processed).</td>
</tr>
<tr>
<td><strong>MC_REventAvgDeliveryTimeAlert</strong></td>
<td>Alert threshold for the average sending time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgDeliveryTimeWarning</strong></td>
<td>Warning threshold for the average sending time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgProcessTimeAlert</strong></td>
<td>Alert threshold for the average processing time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgProcessTimeWarning</strong></td>
<td>Warning threshold for the average processing time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgQueueAlert</strong></td>
<td>Alert threshold for the average number of queued real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgQueueTimeAlert</strong></td>
<td>Alert threshold for average queuing time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgQueueTimeWarning</strong></td>
<td>Warning threshold for average queuing time of real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventAvgQueueWarning</strong></td>
<td>Warning threshold for the average number of queued real-time events.</td>
</tr>
<tr>
<td><strong>MC_REventErrorAlert</strong></td>
<td>Alert threshold for processing errors of real-time events.</td>
</tr>
<tr>
<td>Option name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NmsCleanup_LastCleanup</td>
<td>Defines the last time the cleanup process was run.</td>
</tr>
<tr>
<td>NmsCleanup_BroadLogPurgeDelay</td>
<td>Lets you define the delay after which broadlog are erased from the database.</td>
</tr>
<tr>
<td></td>
<td>This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_EventHistoryPurgeDelay</td>
<td>Lets you define the delay after which the event history is erased from the database.</td>
</tr>
<tr>
<td></td>
<td>This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_EventPurgeDelay</td>
<td>Lets you define the delay after which events are erased from the database.</td>
</tr>
<tr>
<td></td>
<td>This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_EventStatPurgeDelay</td>
<td>Lets you define the delay after which the event statistics are erased from the database.</td>
</tr>
<tr>
<td></td>
<td>This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_PropositionPurgeDelay</td>
<td>Lets you define the delay after which propositions are erased from the database.</td>
</tr>
<tr>
<td></td>
<td>This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>NmsCleanup_QuarantineMailboxFull</td>
<td>Lets you define the delay after which the quarantines are erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_RecycledDeliveryPurgeDelay</td>
<td>Lets you define the delay after which recycled deliveries are erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_RejectsPurgeDelay</td>
<td>Lets you define the delay after which rejects are erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_TrackingLogPurgeDelay</td>
<td>Lets you define the delay after which tracking logs are erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_TrackingStatPurgeDelay</td>
<td>Lets you define the delay after which tracking statistics is erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>NmsCleanup_VisitorPurgeDelay</td>
<td>Lets you define the delay after which visitors are erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>Setting</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NimsCleanup_WorkflowResultPurgeDelay</td>
<td>Lets you define the delay after which workflow results is erased from the database. This option is automatically created once the delay is modified within the interface. If you modify the value from the options list, it should be expressed in seconds.</td>
</tr>
<tr>
<td>WdbcCapabilities_AzureDw</td>
<td>Azure SQL Datawarehouse connector options.</td>
</tr>
</tbody>
</table>
| WdbcKillSessionPolicy                        | Lets you affect Unconditional Stop behavior on all the workflows and PostgreSQL database queries according to the following potential values:
|                                             | • 0 – default: stops workflow process, no impact on the database
|                                             | • 1 - pg. cancel, backend: stops workflow process and cancels query in the database
|                                             | • 2 – pg. terminate, backend: stops workflow process and terminates query in the database.                                               |
| WdbcOptions_TableSpaceUser                  | Name of the tablespace intended to contain the data of the Adobe Campaign cost tables. See Creating and configuring the database.         |
| WdbcOptions_TableSpaceIndex                 | Name of the tablespace intended to contain the indexes of the Adobe Campaign cost tables. See Creating and configuring the database.        |
| WdbcOptions_TableSpaceWork                  | Name of the tablespace intended to contain the data of the Adobe Campaign work tables. See Creating and configuring the database.          |
| WdbcOptions_TableSpaceWorkIndex             | Name of the tablespace intended to contain the indexes of the Adobe Campaign work tables. See Creating and configuring the database.       |
| WdbcOptions_TempDbName                      | Allows you to configure a separate database for working tables on Microsoft SQL Server, in order to optimize backups and replication. The option corresponds to the name of the temporary database. Work tables will be written in this database if specified. Example: "tempdb.dbo" (note that the name must end with a dot). Read more |
| WdbcTimeZone                                | Time zone of the Adobe Campaigns instance. See Configuration.                                                                           |
| WdbcUseNChar                                | Are the database's string fields defined with NChar?                                                                                     |
| WdbcUseTimeStmpWithTZ                       | Do the database's 'datetime' fields store timezone info?                                                                                   |
| XlKDatabaseId                               | ID of the database. Begins by 'U' for Unicode Database.                                                                                   |
| XlKInstancePrefix                           | Prefix added to internal names generated automatically.                                                                                   |
| XlKQuery_Schema_LincCount                   | Maximum number of results returned by a query on xlk:schema and xlk:srcSchema.                                                           |
| XlKSequence_AutoGeneration                  | All customized schemas, created after this time, with autodef="true" and without the attribute "xlkSequence" will get an auto-generated sequence 'auto_<scheme namespace>_<scheme name>_seq' |
| NilMigration_KeepFolderStructure            | During migration, the tree structure is automatically reorganized based on the new version standards. This option allows you to disable the automatic migration of the navigation tree. If you use it, after migration you will have to delete obsolete folders, add the new folders and run all necessary checks.
|                                             | • Data type: Integer.
|                                             | • Value (text): 1
|                                             | This option should only be used if the out-of-the-box navigation tree has undergone too many changes.                                       |
| NimsLastErrorStatCoalesce                   | Last processing date of the NimsEmailErrorStat table cleanup.                                                                             |
| PostUpgradeLastError                        | Information concerning the error that occurred in the Postupgrade, following the syntax below: (Build number) (mode: pre/post...)[The 'lessThan'/'greaterOrEqualThan' where error occurred + sub-step] |
| XlKCleanup_NoStats                          | Enter the 'I' value so that the update of statistics is not performed through the cleanup workflow.                                         |

**Integration**
<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEIMResourceTypeFilter</td>
<td>Types of AEM resources that can be used in Adobe Campaign. Values must be</td>
</tr>
<tr>
<td></td>
<td>separated by commas.</td>
</tr>
<tr>
<td>nmsPipeline_config</td>
<td>Lets you configure Experience Cloud Triggers. Data type is &quot;long text&quot; and</td>
</tr>
<tr>
<td></td>
<td>must be in JSON format. See How to use Experience Cloud Triggers with</td>
</tr>
<tr>
<td></td>
<td>Adobe Campaign Classic.</td>
</tr>
<tr>
<td>LASTIMPORT_&lt;%=instance.internalName%&gt;_&lt;%=activityName%&gt;</td>
<td>This option is used when importing data from a third-party system through a</td>
</tr>
<tr>
<td></td>
<td>CRM connector. Enabling the option lets you collect only objects modified</td>
</tr>
<tr>
<td></td>
<td>since the last import. This option has to be manually created and populated</td>
</tr>
<tr>
<td></td>
<td>as below:</td>
</tr>
<tr>
<td></td>
<td>- Internal name :</td>
</tr>
<tr>
<td></td>
<td>LASTIMPORT_&lt;%=instance.internalName%&gt;_&lt;%=activityName%&gt;</td>
</tr>
<tr>
<td></td>
<td>- Value (field) : date of the last import, with the yyyy/MM/dd hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>format.</td>
</tr>
<tr>
<td>TNT_EdgeServer</td>
<td>Adobe Target server used for the integration. This option is already</td>
</tr>
<tr>
<td></td>
<td>selected by default. This value corresponds to the Adobe Target Domains</td>
</tr>
<tr>
<td></td>
<td>Server, followed by the value /m2. For example: ttdomtrdc.net/m2.</td>
</tr>
<tr>
<td></td>
<td>See this section.</td>
</tr>
<tr>
<td>TNT_TenantName</td>
<td>Adobe Target Organization name. This value corresponds to the name of the</td>
</tr>
<tr>
<td></td>
<td>Adobe Target Client.</td>
</tr>
<tr>
<td></td>
<td>See this section.</td>
</tr>
<tr>
<td>AAM_DataSourceId</td>
<td>Option used for the integration with Adobe Audience Manager.</td>
</tr>
<tr>
<td>AAM_DestinationId</td>
<td>Option used for the integration with Adobe Audience Manager.</td>
</tr>
<tr>
<td>WebCapabilities_Teradata</td>
<td>Teradata connector options.</td>
</tr>
<tr>
<td>WebCapabilities_Hive</td>
<td>Hive connector options.</td>
</tr>
</tbody>
</table>

**Offers**

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NmsCoupoleMaxPerTransact</td>
<td>Number of coupons that are updated per SQL transaction.</td>
</tr>
<tr>
<td>NmsInteraction_LastPropositionSynchControl_&lt; proposition's schema &gt; + &quot;&quot; + extAccountSource @executionInstanceid + [proposition's schema] + &quot;&quot; + var&lt;extensioninstanceidFilter&gt;</td>
<td>Synchronization workflow tracking.</td>
</tr>
<tr>
<td>NmsInteraction_LastPropositionSynchExec_&lt; proposition's schema &gt; + &quot;&quot; + extAccountSource @executionInstanceid + [&quot;&quot;] + extAccountTarget @executionInstanceid</td>
<td>Enable/disable asynchronous proposition writing (&quot;0&quot; to disable, &quot;1&quot; to enable).</td>
</tr>
<tr>
<td>NmsModule_CouponsEnabled</td>
<td>Lets you enable coupons.</td>
</tr>
</tbody>
</table>
### Tracking

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NmsExecutionInstanceID</td>
<td>Execution instance identifier.</td>
</tr>
<tr>
<td>NmsServer_CustomerID</td>
<td>Customer identifier used when sending the billing report.</td>
</tr>
<tr>
<td>NmsServer_IntranetURL</td>
<td>Internal base URL to access the application server.</td>
</tr>
<tr>
<td>NmsServer_LastPostUpgrade</td>
<td>Build number of the AC instance before the last upgrade.</td>
</tr>
<tr>
<td>NmsServer_URL</td>
<td>Base URL of the public web application server.</td>
</tr>
<tr>
<td>XiiPassUnknownSQLFunctionsToRDBMS</td>
<td>Lets you continue using old undeclared SQL functions after migrating. We strongly recommend against using this option due to the security risks it introduces.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NmsTracking_Available</td>
<td>Option that lets you activate tracking.</td>
</tr>
<tr>
<td>NmsTracking_ClickFormula</td>
<td>Tracked-URL calculation script.</td>
</tr>
<tr>
<td>NmsTracking_ExtAccount</td>
<td>Lets you define the tracking server's external account.</td>
</tr>
<tr>
<td>NmsTracking_Instance</td>
<td>Lets you define the instance name on the tracking server.</td>
</tr>
<tr>
<td>NmsTracking_LastConsolidation</td>
<td>Last time the tracking information has been consolidated with new data.</td>
</tr>
<tr>
<td>NmsTracking_OpenFormula</td>
<td>Open URL computation script.</td>
</tr>
<tr>
<td>NmsTracking_Password</td>
<td>Password for the tracking server.</td>
</tr>
<tr>
<td>NmsTracking_Pointer</td>
<td>The pointer keeps track of the last message events that were processed through their IDs and date.</td>
</tr>
<tr>
<td>NmsTracking_SecureServerURL</td>
<td>Secure URL of the frontal tracking server.</td>
</tr>
<tr>
<td>NmsTracking_ServerURL</td>
<td>URL of the frontal tracking server.</td>
</tr>
<tr>
<td>NmsTracking_ServerURLList</td>
<td>List of the URLs used to contact the tracking servers.</td>
</tr>
<tr>
<td>NmsTracking_UserAgentRules</td>
<td>Browser-identification rule set.</td>
</tr>
<tr>
<td>NmsTracking_WebFormula</td>
<td>Script used to compute Web tracking tags.</td>
</tr>
<tr>
<td>NmsTracking_WebTrackingDelivery</td>
<td>Name of the virtual delivery designed for web tracking management.</td>
</tr>
<tr>
<td>NmsTracking_WebTrackingMode</td>
<td>Lets you define the web tracking mode.</td>
</tr>
</tbody>
</table>

### Privacy

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy_Request_ConfirmDeletePending</td>
<td>If option 1 is selected, you have to confirm manually the deletion in the interface in a second step. Otherwise, data will be deleted without confirmation.</td>
</tr>
<tr>
<td>Privacy_Request_ConfirmDeletePendingDelay</td>
<td>Delay between request waits for deleting confirmation and request is cancelled.</td>
</tr>
<tr>
<td>Privacy_Request_MaxErrorAllowed</td>
<td>The maximum number of error allowed when processing/deleting a privacy request.</td>
</tr>
<tr>
<td>Privacy_Request_PurgeDelay</td>
<td>Delay between request is created on the queue and request data is deleted.</td>
</tr>
</tbody>
</table>
1.5 Configure your external accounts

Adobe Campaign comes with a set of pre-defined external accounts. In order to set up connections with external systems, you can create new external accounts.

External accounts are used by technical processes such as technical workflows or campaign workflows. For example, when setting up a file transfer in a workflow or a data exchange...
with any other application (Adobe Target, Experience Manager, etc.), you need to select an external account.

You can access external accounts from Adobe Campaign Explorer: browse to Administration > Platform > External accounts.

CAUTION

- As a Managed Cloud Services user, external accounts are configured for your instance by Adobe and must not be modified.
- In the context of an Enterprise (FFDA) deployment, a specific Full FDA (ffda) external account manages connection between Campaign local database and Cloud database (Snowflake).

Campaign-specific external accounts

The following technical accounts are used by Adobe Campaign to enable and execute specific processes.

Bounce mails

NOTE

The Microsoft Exchange Online OAuth 2.0 authentication for POP3 capability is available starting Campaign v8.3. To check your version, refer to this section.
The **Bounce mails** external account specifies the external POP3 account to be used to connect to the email service. All servers configured for POP3 access can be used to receive return mail.

Learn more about inbound emails in [this page](#).

To configure the **Bounce mails (defaultPopAccount)** external account:

- **Server** - URL of the POP3 server.
- **Port** - POP3 connection port number. The default port is 110.
- **Account** - Name of the user.
- **Password** - User account password.
- **Encryption** - Type of chosen encryption between **By default, POP3 + STARTTLS, POP3 or POP3S.**

The **Bounce mails** external account specifies the external POP3 account to be used to connect to the email service. All servers configured for POP3 access can be used to receive return mail.

- **Function** - Inbound email or SOAP router
CAUTION

Before configuring your POP3 external account using Microsoft OAuth 2.0, you first need to register your application in the Azure portal. For more on this, refer to this page.

To configure a POP3 external using Microsoft OAuth 2.0, check the Microsoft OAuth 2.0 option and fill in the following fields:

- **Azure tenant** - Azure ID (or Directory (tenant) ID) can be found in the Essentials drop-down of your application overview in the Azure portal.
- **Azure Client ID** - Client ID (or Application (client) ID) can be found in the Essentials drop-down of your application overview in the Azure portal.
- **Azure Client secret** - Client secret ID can be found in the Client secrets column from the Certificates & secrets menu of your application in the Azure portal.
- **Azure Redirect URL** - Redirect URL can be found in the Authentication menu of your application in the Azure portal. It should end with the following syntax `nl/jsp/oauth.jsp`, e.g. `https://redirect.adobe.net/nl/jsp/oauth.jsp`.

After entering your different credentials, you can click Setup the connection to finish your external account configuration.

Routing

The Routing external account allows you to configure each channel available in Adobe Campaign depending on the packages installed.

Execution instance

In the context of transactional messaging, the execution instances is linked to the control instance and connect them. Transactional message templates are deployed to the execution instance. Learn more about Message Center architecture in this page.

Access to External Systems external accounts

- **External database (FDA)** - The External database type external account is used to connect to external an database via Federated Data Access (FDA). Learn more about Federated Data Access (FDA) option in this section.

  External databases compatible with Adobe Campaign v8 are listed in the Compatibility matrix.

- **Twitter** - The Twitter type external account is used to connect Campaign to your twitter account, to post messages on your behalf. Learn more about Twitter integration in this section.

Adobe Solution Integration external accounts
- **Adobe Experience Cloud** - The **Adobe Experience Cloud** external account is used to implement Adobe Identity Management Service (IMS) to connect to the Adobe Campaign. Learn more about Adobe Identity Management Service (IMS) in this section.

- **Web Analytics** - The **Web Analytics (Adobe Analytics)** external account is used to configure data transfer from Adobe Analytics to Adobe Campaign. Learn more about Adobe Campaign - Adobe Analytics integration in this page.

- **Adobe Experience Manager** - The **AEM** external account allows you to manage the content of your email deliveries as well as your forms directly in Adobe Experience Manager. Learn more about Adobe Campaign - Adobe Analytics integration in this page.

**CRM Connector External accounts**

- **Microsoft Dynamics CRM** - The **Microsoft Dynamics CRM** external account allows you to import and export Microsoft Dynamics data into Adobe Campaign. Learn more about Adobe Campaign - Microsoft Dynamics CRM integration in this page.

- **Salesforce.com** - The **Salesforce CRM** external account allows you to import and export Salesforce data into Adobe Campaign. Learn more about Adobe Campaign - Salesforce.com CRM integration in this page.

**Transfer Data external accounts**

These external accounts can be used to import or export data to Adobe Campaign using a Transfer file workflow activity. Learn more about File transfer in workflows in this page.

- **FTP and SFTP** - The **FTP** external account lets you configure and test access to a server outside of Adobe Campaign. To set up connections with external systems such as SFTP or FTP servers 898 used for file transfers, you can create your own external accounts.

  To do so, specify in this external account the address and credentials used to establish the connection to the SFTP or FTP server.

**NOTE**

Starting from release 8.5, you can now securely authenticate using a private key when configuring your SFTP external account. Learn more on key management.

- **Amazon Simple Storage Service (S3)** - The **AWS S3** connector can be used to import or export data to Adobe Campaign using a Transfer file workflow activity. As you are setting up this new external account, you need to provide the following details:
  - **AWS S3 Account Server**: URL of your server, filled as follows: `<S3bucket name>.s3.amazonaws.com/<s3object path>`
  - **AWS access key ID**: Learn how to find your AWS access key ID in Amazon documentation.
• **Secret access key to AWS**: Learn how to find your secret access key to AWS in Amazon documentation.

• **AWS Region**: Learn more on AWS regions in Amazon documentation.

• The **Use server side encryption** checkbox allows you to store your file in S3 encrypted mode. Learn how to find the access key ID and secret access key in Amazon documentation.

• **Azure Blob Storage** - The **Azure** external account can be used to import or export data to Adobe Campaign using a **Transfer file** workflow activity. To configure the **Azure** external account to work with Adobe Campaign, you need to provide the following details:
  - **Server**: URL of your Azure Blob storage server.
  - **Encryption**: Type of encryption between **None** or **SSL**.
  - **Access key**: Learn how to find your **Access key** in Microsoft documentation.

### 1.6 About campaign typologies

Campaign Optimization is the Adobe Campaign module which lets you control, filter and monitor the sending of deliveries. To avoid conflicts between campaigns, Adobe Campaign can test various combinations by applying specific constraint rules. This guarantees that the messages sent meet the needs and expectations of customers and company communication policies.

Discover this feature in video

**NOTE**

Depending on your offering, Campaign Optimization can be included or an add-on. Please check your license agreement.

**Typology rules**

With Adobe Campaign you can design and apply four types of typology rules:

• **Filtering** rules which let you exclude part of the target based on criteria. For more on this, refer to Filtering rules.

• **Pressure** rules which let you control marketing fatigue. For more on this, refer to Pressure rules.

• **Capacity** rules which let you limit loads to guarantee optimal processing conditions. For more on this, refer to Controlling capacity.

• **Control** rules which let you check the validity of messages before they are sent. For more on this, refer to Control rules.
Once they have been created, typology rules are grouped in campaign typologies which are referenced in deliveries. See Applying typologies.

**Typologies**

A campaign typology can contain several typology rules, but a delivery can only reference one typology.

The Rules tab lets you add, delete or view the typology rules to apply.

---

**Applying typologies**

Steps to create and apply a typology to your deliveries are listed below:

1. Create typology rules.

   Typology rules are found in the Administration > Campaign management > Typology management > Typology rules node.

   Different rules available in Campaign are described in the following sections: sales pressure rules, capacity rules, control rules and filtering rules.

2. Create a typology and reference the rules you created into it.

   Typologies are accessed via the Administration > Campaign Management > Typology management > Typologies node.
3. Configure your delivery to use the typology you created. For more on this, refer to this section.
4. Test and control the behavior through campaign simulations. For more on campaign simulations, refer to this section.

During delivery preparation, recipients are excluded when criterion is met. You can check logs to monitor exclusions. Sample use cases on pressure typology rules are available in this page.

**Tutorial videos**

How to set up fatigue management using typology rules

This video explains how to implement fatigue management in Adobe Campaign by leveraging typology rules.

How to set up fatigue management using predefined filters

Fatigue management controls frequency and quantity of messaging to avoid over-solicitation of recipients. If you do not have the campaign optimization module in your campaign instance, you may configure a predefined filter that will filter the target population by the number of messages received. This video explains how to implement fatigue management in Adobe Campaign Classic by using filters.

Additional Campaign how-to videos are available here.

**Related topic**

- Get started with typologies and fatigue management