

# 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.

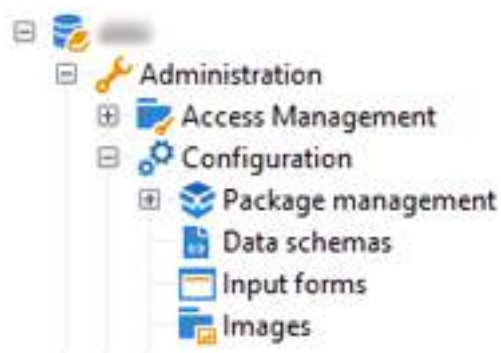
## 1.1 Data schemas

### 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:

Source schema 'Invitation (xac)'

Name: xac:invitation Label: Invitation

Label (singular): Newsletter Description: Monthly News

```
<srcSchema _cs="Invitation (xac)" created="2011-09-19 09:56:19.1772" createdBy-id="0"
  desc="Monthly News" entitySchema="xtk:srcSchema" img="xtk:schema.png"
  label="Invitation" labelSingular="Newsletter" lastModified="2011-09-19 10:02:24.5762"
  mappingType="sql" md5="A29208E57A86B1E10E7B8BC9CDB4D634" modifiedBy-id="0"
  name="invitation" namespace="xac" xtk:schema="xtk:srcSchema">
  <createdBy _cs="Administrator (admin)"/>
  <modifiedBy _cs="Administrator (admin)"/>
  <element label="Invitation" name="invitation" template="ncm:content" xmlChildren="true">
    <compute-string expr="@name"/>
    <attribute label="Title" length="40" name="title" type="string"/>
    <element label="Presentation" name="presentation" type="html"/>
    <attribute label="Date" name="date" type="date"/>
    <attribute label="Name" length="10" name="name" type="string"/>
    <attribute label="URL" name="url" type="string"/>
    <element label="Author" name="author" type="memo"/>
    <element label="Image" name="image" target="xtk:fileRes" type="link"/>
  </element>
</srcSchema>
```

Edit Preview Structure Documentation Data

Ln 9, Col 17 | INS

## 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:

Source schema 'client (cus)'

Name: cus:client Label: client

Label (singular): Description:

Generated schema preview

```
<?xml version='1.0'?>
<schema implements="" label="client" mappingType="sql" name="client" namespace="cus"
  xtk:schema="xtk:schema">
  <element label="client" name="client" sqltable="CusClient">
    <element advanced="true" integrity="neutral" label="Current operator" name="currentOperator"
      revLink="_NONE_" target="xtk:operator" type="link">
      <join xpath-dst="@id" xpath-src="{loginId}"/>
    </element>
  </element>
  <enumeration basetype="byte" name="gender">
    <value label="Non specified" name="unknown" value="0"/>
    <value label="Male" name="male" value="1"/>
    <value label="Female" name="female" value="2"/>
  </enumeration>
  <attribute desc="Email of the client" label="Email" length="80" name="email" type="string"/>
  <element label="Localization" length="50" name="location" type="string">
    <attribute label="City" length="50" name="city" type="string"/>
  </element>
</schema>
```

Edit Preview Structure Documentation Data

Ln 1, Col 1 | INS

## NOTE

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

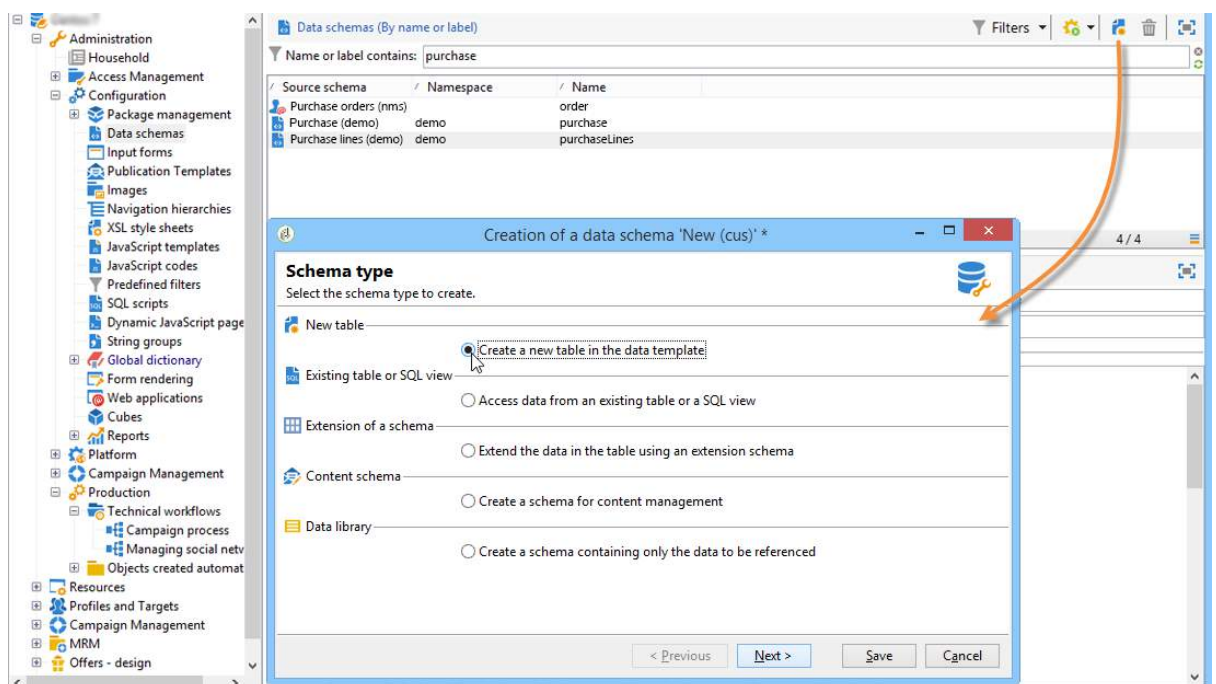
If you need to check the complete structure of a schema, you can use the preview tab. If the schema has been extended, you will then be able to visualize all its extensions. As a complement, the Documentation tab displays all the schema attributes and elements, and their properties (SQL Field, type/length, 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.

Creation of a data schema 'Contracts (cus)' \*

### Schema parameters

Please specify the different parameters of the data schema.

General parameters

Namespace: cus

Name: Contracts

Label: Contracts

Label (singular): Contract

Description: Active contracts

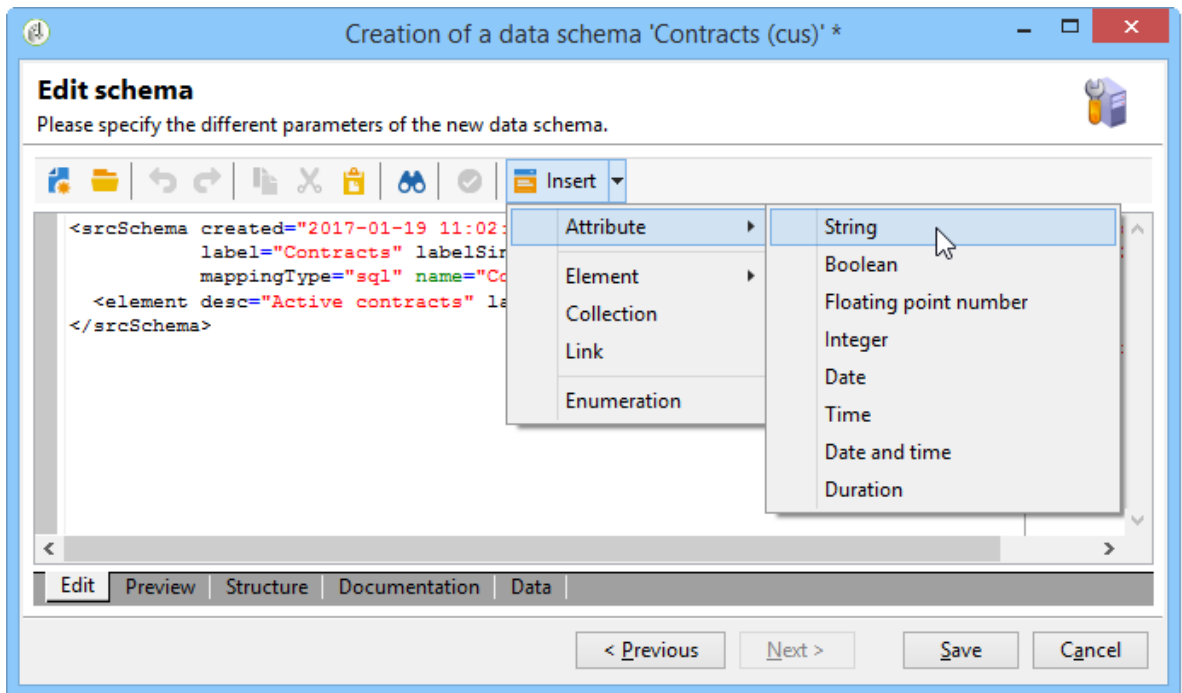
Image: schema.png (xkt)

< Previous   Next >   Save   Cancel

#### 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>`

Copy

Toggle Text Wrapping

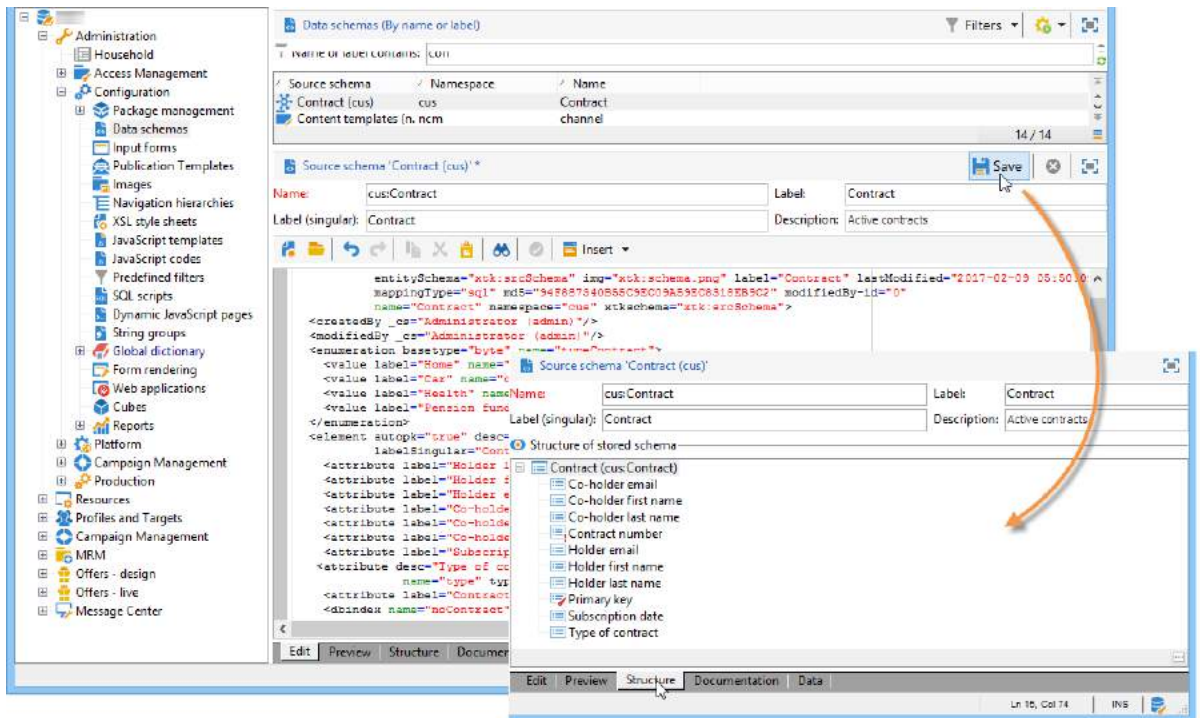
Add the type of contract and place an index on the contract number.

```
<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"
  labelSingular="Contract" name="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>
```

Copy

Toggle Text Wrapping

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:

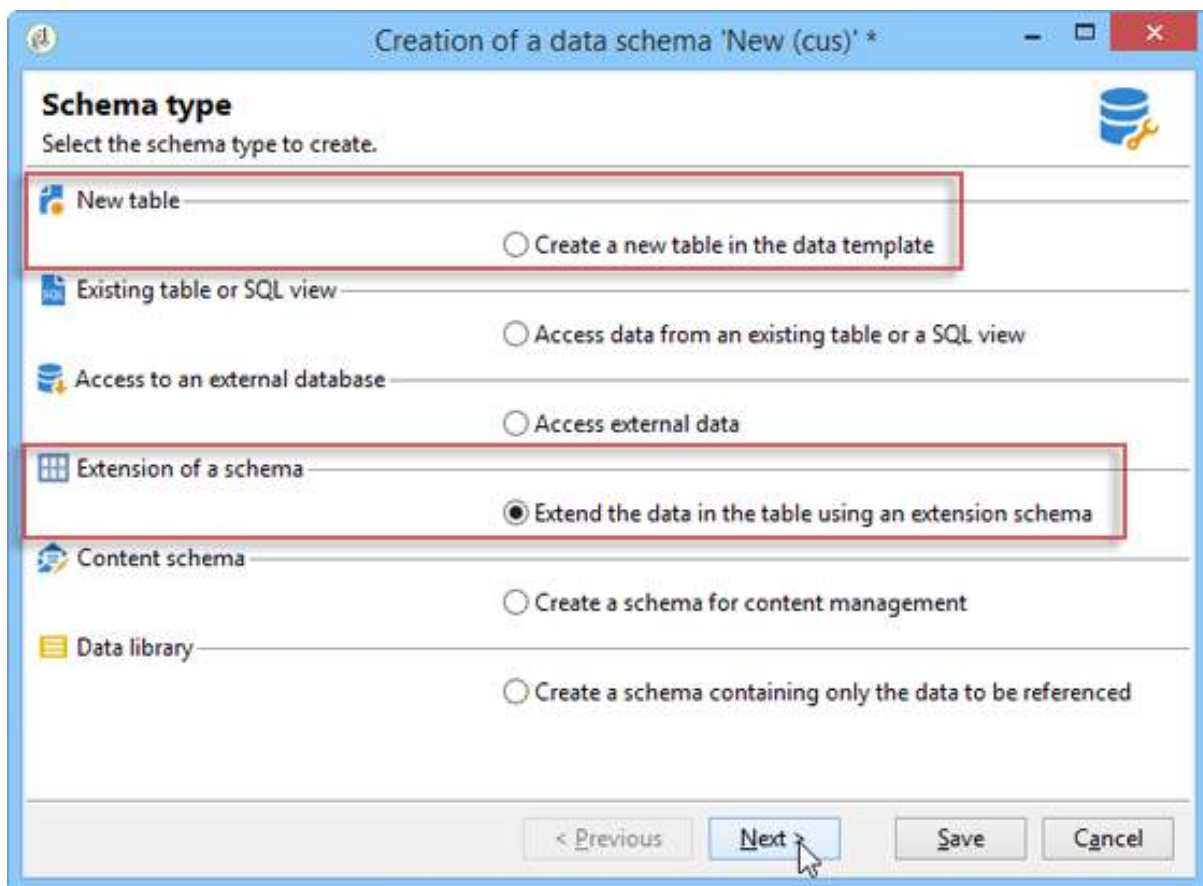
- Define how data objects within the application are tied to underlying database tables.
- Define links between the different data objects within the Campaign application.
- Define and describe the individual fields included in each object.

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:

```
<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:

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

Copy

Toggle Text Wrapping

#### NOTE

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:

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

Copy

Toggle Text Wrapping

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

Copy

Toggle Text Wrapping

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:

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

Copy

Toggle Text Wrapping

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

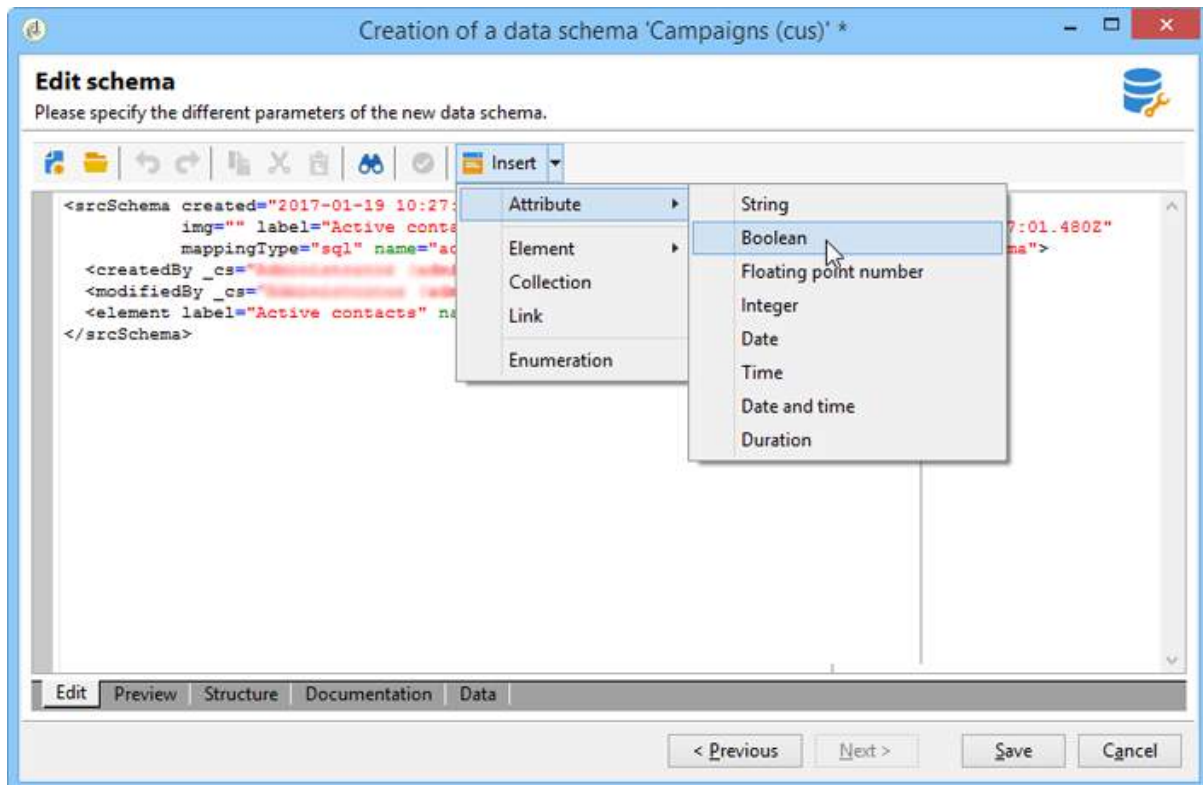
### IMPORTANT

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:

```
<attribute name="transactionDate" label="Transaction Date" type="datetime"
default="GetDate()"/>
```

Copy

Toggle Text Wrapping

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

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

Copy

Toggle Text Wrapping

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

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

Copy

Toggle Text Wrapping

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

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

Copy

Toggle Text Wrapping

### 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:

```
<element label="Recipient" name="InkRecipient" revLink="InkTransactions"
target="nms:recipient" type="link"/>
```

Copy

Toggle Text Wrapping

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

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

Copy

Toggle Text Wrapping

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

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

Copy

Toggle Text Wrapping

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:

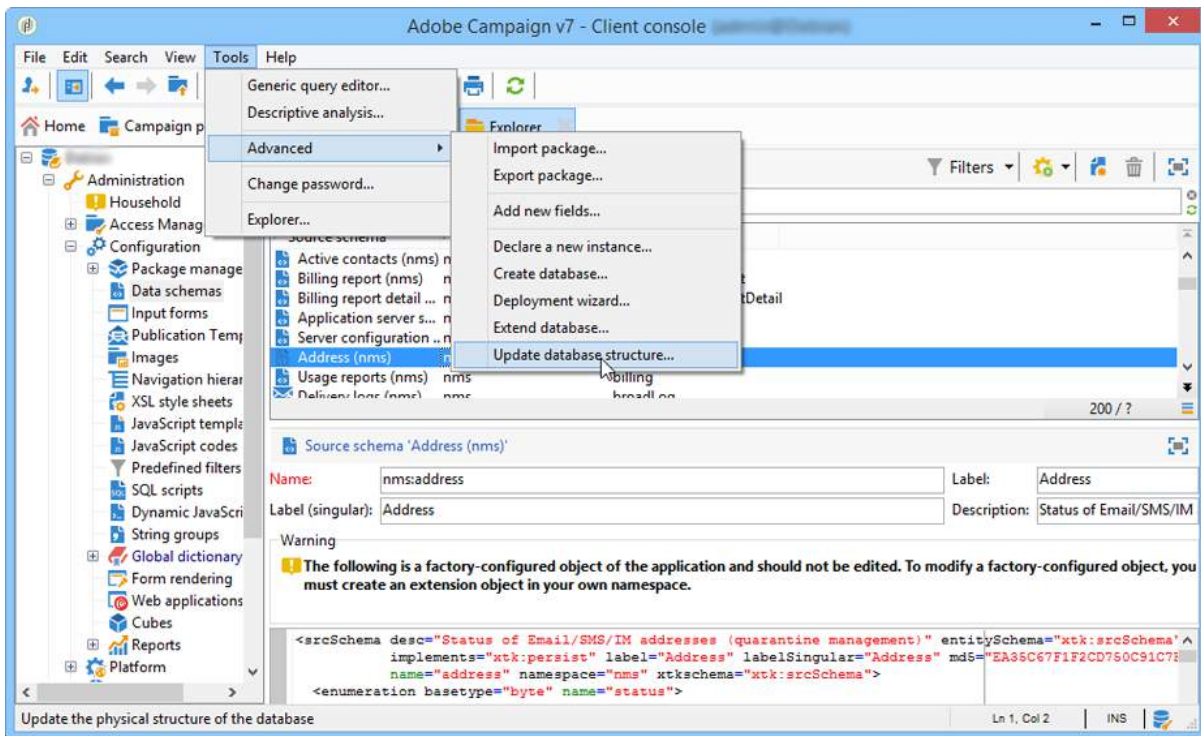
```
<element aggregate="xtk:common:auditTrail" name="auditTrail"/>
```

Copy

Toggle Text Wrapping

## 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:

```
<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>
<key>
  ... //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>
```

Copy

Toggle Text Wrapping

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>
```

Copy

Toggle Text Wrapping

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>
```

Copy

Toggle Text Wrapping

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>
```

Copy

Toggle Text Wrapping

## 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">
```



Copy

Toggle Text Wrapping

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:

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

Copy

Toggle Text Wrapping

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 **length** attribute (optional, default value “255”).

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

#### NOTE

To contain a **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:

```
<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>
```

Copy

Toggle Text Wrapping

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.

Adobe Campaign	PosgreSQL	Oracle	MS SQL
String	VARCHAR(255)	VARCHAR2 (NVARCHAR2 if unicode)	VARCHAR (NVARCHAR if unicode)
Boolean	SMALLINT	NUMBER(3)	TINYINT
Byte	SMALLINT	NUMBER(3)	TINYINT
Short	SMALLINT	NUMBER(5)	SMALLINT
Double	DOUBLE PRECISION	FLOAT	FLOAT
Long	INTEGER	NUMBER(10)	INT
Int64	BIGINT	NUMBER(20)	BIGINT
Date	DATE	DATE	DATETIME
Time	TIME	FLOAT	FLOAT
Datetime	TIMESTAMPZ	DATE	MS SQL < 2008: DATETIME MS SQL >= 2012: DATETIMEOFFSET
Datetimenotz	TIMESTAMPZ	DATE	MS SQL < 2008: DATETIME MS SQL >= 2012: DATETIME2
Timespan	DOUBLE PRECISION	FLOAT	FLOAT
Memo	TEXT	CLOB (NCLOB if Unicode)	TEXT (NTEXT if Unicode)
Blob	BLOB	BLOB	IMAGE

## 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:

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

Copy

Toggle Text Wrapping

The label can be seen from the Adobe Campaign client console input form:

Email:

 john.druiers@graph-museum.org

- 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:

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

Copy

Toggle Text Wrapping

#### 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.

<sqldefault> : 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

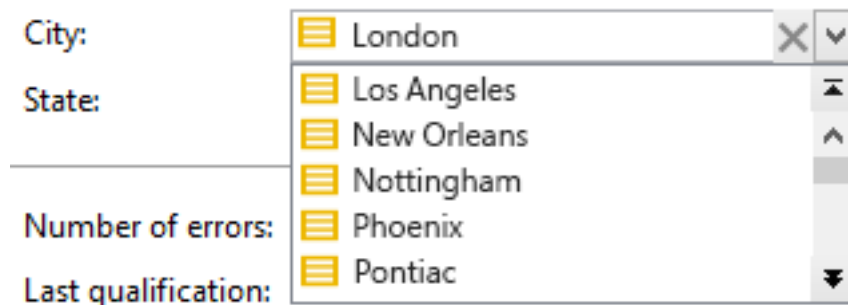
### Free enumeration

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:



The image shows a form with four fields: "City:", "State:", "Number of errors:", and "Last qualification:". A drop-down menu is open over the "City:" field, displaying a list of cities: London, Los Angeles, New Orleans, Nottingham, Phoenix, and Pontiac. Each city name is preceded by a small yellow icon with three horizontal lines. The drop-down menu has a close button (X) and a downward arrow on the right side.

### NOTE

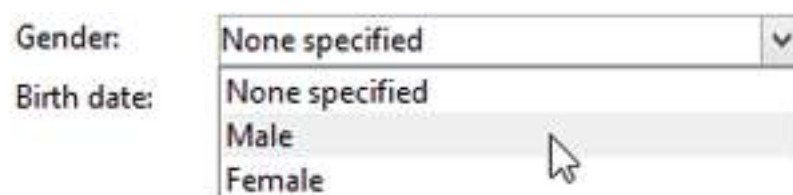
In the Adobe Campaign client console, the **Administration > Enumerations** node is used to manage enumerations.

### 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:



The image shows a form with two fields: "Gender:" and "Birth date:". A drop-down menu is open over the "Gender:" field, displaying a list of options: "None specified", "Male", and "Female". The "Male" option is highlighted with a mouse cursor. The drop-down menu has a downward arrow on the right side.

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>
```

Copy

Toggle Text Wrapping

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:

```

<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 name="location" label="Location">
      <attribute name="city" type="string" length="50" label="City" userEnum="city"/>
    </element>
  </element>
</srcSchema>

```

Copy

Toggle Text Wrapping

## 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.

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

Copy

Toggle Text Wrapping

With projection of the XML content:

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

Copy

Toggle Text Wrapping

## 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:

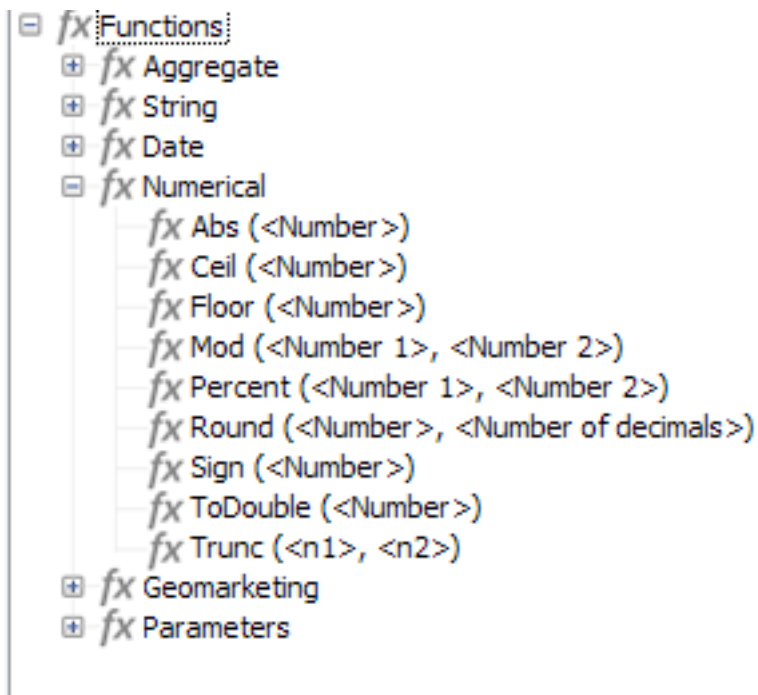
- **location/@city** is not valid; please use [location/@city]
- [email] and @email are equivalent

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:



#### 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.

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

Copy

Toggle Text Wrapping

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

Option name	Description
Deliverability_LastBroadLogMsgDate	Date of the last broadLogMsg retrieved from the deliverability instance.
Deliverability_LastBroadLogMsgSent	Date of the last broadLogMsg sent to the deliverability instance.
DmRendering_cuid	Delivery reports identifier. Please contact support to obtain your identifier.
DmRendering_SeedTargets	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.
EMTA_BCC_ADDRESS	BCC email address to which the Enhanced MTA will send a raw copy of the sent emails.
NMS_ActivateOwnerConfirmation	<p>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.</p> <p>To do this, activate the option by entering "1" as the value. To deactivate this option, enter "0".</p> <p>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 <a href="#">this section</a>.</p>
Nms_DefaultRcpSchema	<p>Adobe Campaign uses a "Nms_DefaultRcpSchema" global variable to dialog with the default recipient database (nms:recipient).</p> <p>The option value must correspond to the name of the schema which matches the external recipient table.</p>
NmsBilling_MainActionThreshold	Minimum number of recipients in order for a delivery to be considered as the main one in the billing report.
NmsBroadcast_DefaultProvider	Default routing service provider for the new templates.
NmsBroadcast_LogsPerTransac	Minimal batch size (number of rows) for the insertion of broadLogs during a delivery preparation.
NmsBroadcast_MaxDelayPerTransac	Batch duration threshold (number of milliseconds) under which the batch size for the insertion of broadLogs is doubled during a delivery preparation.
NmsBroadcast_MidAnalyzeBatchSize	Grouping size of delivery parts when analyzing mid-sourcing deliveries.

NmsBroadcast_MsgValidityDuration	Default delivery period for a delivery (in seconds).
NmsBroadcast_RegexRules	Regular expressions for normalizing delivery messages.
NmsBroadcast_RemoveBlackList	Entering "1" as the value lets you exclude recipients who no longer wish to be contacted.
NmsBroadcast_RemoveDuplicatesRecipients	Entering "1" as the value lets you automatically ignore doubles.
NmsDelivery_ErrorAddressMasks	Lets you define the syntax of the Error address used when replying to a message.
NmsDelivery_FromAddressMasks	Lets you define the syntax of the From address used when sending a message.
NmsDelivery_ImageServerTimeout	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.
NmsDelivery_MaxDownloadedImageSize	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.
NmsDelivery_MaxRecommendedAttachments	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.
NmsDelivery_MaxRetry	Maximum number of retries during analysis.
NmsDelivery_PublishingScript	Publication script.
NmsDelivery_NoCountBroadLogMsgPush	Disable the broadLogMsg count for push messages.
NmsDeliveryWizard_ShowDeliveryWeight	Display the message weight in the delivery wizard.
NmsEmail_DefaultErrorAddr	Default 'error' email address at instance's level used for email delivery if left empty by user.
NmsEmail_DefaultFromAddr	Default 'from' email address at instance's level used for email delivery if left empty by user.
NmsEmail_DefaultReplyToAddr	Default 'reply-to' email address at instance's level used for email delivery if left empty by user.

NmsEmail_ExpOrganization	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`"
NmsEmail_FromName	Default 'from' email label at instance's level used for email delivery if left empty by user.
NmsEmail_ReplyToName	Default 'reply-to' email label at instance's level used for email delivery if left empty by user.
NmsEmail_RetryCount	Period between two retries of an email message (in seconds).
NmsEmail_RetryPeriod	Period of retries for email messages.
NmsForecast_MsgWeightFormula	Formula used to calculate the weighting of a message for a provisional delivery.
NmsInmail_AllowlistEmails	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. xyz@abc.com,pqr@abc.com.
NmsLine_AESKey	AES key used in the 'lineImage' servlet to encode the URLs (LINE channel).
NmsNPAl_EmailMaxError	On channel "email"(use as default) : Maximal number of errors that is accepted, for SOFT errors during sending before putting the recipient into quarantine.
NmsNPAl_EmailSignificantErrorDelay	On channel "email"(use as default) : Minimal period to spent since the previous referenced SOFT error, before taking into account a new SOFT error.
NmsNPAl_MobileMaxError	On channel "mobile" : Maximal number of errors that is accepted, for SOFT errors during sending before putting the recipient into quarantine.
NmsNPAl_MobileSignificantErrorDelay	On channel "mobile" : Minimal period to spent since the previous referenced SOFT error, before taking into account a new SOFT error.
NmsMidSourcing_LogsPeriodHour	Allows a maximum period (expressed in hours) to be specified as to limit the number of broadlogs recovered every time the synchronization workflow is executed.
NmsMidSourcing_PrepareFlow	Maximum number of calls in MidSourcing session, which can be run in parallel (3 by default).
NmsMTA_Alert_Delay	Custom delay (in minutes) after which a delivery is considered as 'delayed', default being 30 minutes.

NmsOperation_DeliveryPreparationWindow	<p>This option is used by the <a href="#">operationMgt</a> technical workflow when counting the number of running deliveries.</p> <p>It allows you to define the number of days above which deliveries with inconsistent status will be excluded from the count of running deliveries.</p> <p>By default, the value is set to "7", meaning that inconsistent deliveries older than 7 days will be excluded.</p>
NmsPaper_SenderLine1	Line 1 of the sender's address.
NmsPaper_SenderLine3	Line 3 of the sender's address.
NmsPaper_SenderLine4	Line 4 of the sender's address.
NmsPaper_SenderLine6	Line 6 of the sender's address.
NmsPaper_SenderLine7	Line 7 of the sender's address.
NmsServer_MirrorPageUrl	<p>URL of the mirror page server (by default, should be identical to NmsTracking_ServerUrl).</p> <p>It is the default value of email deliveries when the URL is not specified in the routing definition.</p>
NmsSMS_Priority	Parameters of sent SMS messages: information transmitted to the SMS gateway to indicate the message priority.
NmsSMS_RetryCount	Number of retries when sending SMS messages.
NmsSMS_RetryPeriod	Period during which retries of SMS messages will be performed.
NmsUserAgentStats_LastConsolidation	Last consolidation date for NmsUserAgent statistics.
NmsWebSegments_LastStates	Name of the option which contains the web segments and their states.
XtkBarcode_SpecialChar	Enable/disable support for special characters for Code128.
XtkEmail_Characters	Valid characters for an email address.
XtkSecurity_Restrict_EditXML	Add this option with the "0" value to disable the edition of deliveries' XML code (right-click / Edit XML source or CTRL + F4 shortcut).

## Resources

Option name	Description
NcmRessourcesDir	Location of resources for publication in the Adobe Campaign client console. See <a href="#">this section</a> .
NcmRessourcesDirPreview	Location of resources for previewing in the Adobe Campaign client console. See <a href="#">this section</a> .
NmsDelivery_DefaultIgnoredImage	List of URL masks for the images skipped during upload.
NmsDelivery_ImagePublishing	Configuration of image uploading. The values can be none / tracking / script / list (can be overridden by operator's optional settings).
NmsDelivery_ImageSubDirectory	Folder in which the images on the server are to be stored.
NmsServer_LogoPath	Space to display logos.
NcmPublishingDir	Root folder for publications. For more on HTML and Text contents generation, refer to <a href="#">this section</a> .
XtkImageUrl	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 <= 5098, we use the URL of the images that were uploaded onto the instance. For build versions > 5098, we use instead the delivery's public URL or the XtkFileRes_Public_URL option's URL.
NmsDelivery_MediaInstance	Lets you configure the instance name for image uploading.
NmsDelivery_MediaPassword	Lets you configure the password for image uploading.
NmsDelivery_MediaServers	Lets you configure the media URL(s) for image uploading.
NmsBroadcast_MsgWebValidityDuration	Default validity duration for online resources of a delivery (in seconds).
XtkFileRes_Public_URL	New URL for public resource files.

## Campaign & workflow management

Option name	Description
CrmMarketingActivityWindow	Marketing history shown for this number of months.
NmsOperation_Duration	Default validity period of a campaign (in seconds).
NmsOperation_LimitConcurrency	Maximum number of delivery/workflow/hypothesis/simulation jobs that can be processed at a time, started by operationMgt workflow.
NmsOperation_OperationMgtDebug	Allows you to monitor the <code>operationMgt</code> technical workflow execution. When activated (value "1"), the execution information are logged in the workflow audit logs.
NmsOperation_TimeRange	Time period for targeting and extraction conditions in scheduled mode.
Workflow_AnalysisThreshold	Number of affected records after which table statistics are automatically recomputed.
XtkReport_Logo	Logo to be displayed in the top right-hand corner of the reports exported.
NmsServer_PausedWorkflowPeriod	Number of days to wait between checks for paused workflows.
NmsCampaign_Activate_OwnerConfirmation	Activate the deliveries validation by the owner of the operation by entering "1" as the value. To deactivate this option, enter "0".
NmsAsset_JavascriptExt	Additional JS library to load in workflow's activity "Marketing resource notifications".

## Security

Option name	Description
RestrictEditingOOTBSchema	(starting 21.1.3 release) If 1 is selected (default value), this option disables edition of build-in schemas.
RestrictEditingOOTBJavascript	(starting 21.1.3 release) If 1 is selected (default value), this option disables edition of build-in JavaScript codes.
XtkAcceptOldPasswords	(Install compatibility mode: build>6000) When activated (value "1"), this option allows the use of old passwords stored in the database for the connection to external accounts or to the instance.
XtkKey	This key is used to encrypt most passwords in the database. (external accounts, LDAP password...).
XtkSecurity_Allow_PrivilegeEscalation	If 1 is selected, this option to allow privilegeEscalation in JavaScript.
XtkSecurity_Disable_ControlsOnFileDownload	If 1 is selected, this option disable ACL controls during a file download (via fileDownload.jsp).
XtkSecurity_Disable_JSFileSandboxing	If 1 is selected, this option disable the file sandboxing within Javascript.
XtkSecurity_SaveOptions_AllowNonAdmin	If set to 'true', authorized non-admin operator to update the xtkOption values through the deployment wizard.
XtkSecurity_Unsafe_DecryptString	If 1 is selected, this option allow to use decryptString to decrypt some passwords.
XtkTraceDeleteLogin	Enter the "1" value to trace the deletion of elements with Audit trail information in the mData, through the modification of its "modified by" field before the deletion of the record.

## Message Center

Option name	Description
MC_EnrichmentCustomJs	JavaScript library to be personalized for enriching events. Must contain the implementation of these two functions: <ul style="list-style-type: none"> <li>enrichRtEvents(aiEventId); : enriches and saves events in the database (where aiEventId corresponds to the table of real time events processed).</li> <li>enrichBatchEvents(aiEventId); : enriches and saves events in the database (where aiEventId corresponds to the ID table of batch events processed).</li> </ul>
MC_LastUpdateFromBL	Date of the last event status update via delivery logs.
MC_RoutingCustomJs	JavaScript library to be personalized for routing events. Must contain the implementation of these two functions: <ul style="list-style-type: none"> <li>dispatchRtEvent(iEventId); : returns the internal name of the transactional message selected to process the real time event (where iEventId corresponds to the ID of the real time event processed).</li> <li>dispatchBatchEvent(iEventId); : returns the internal name of the transactional message selected to process the batch event (where iEventId corresponds to the ID of the batch event processed).</li> </ul>
MC_RtEventAvgDeliveryTimeAlert	Alert threshold of average sending time of real-time events.
MC_RtEventAvgDeliveryTimeWarning	Warning threshold for average sending time of real-time events.
MC_RtEventAvgProcessTimeAlert	Alert threshold for the average processing time of real-time events.
MC_RtEventAvgProcessTimeWarning	Warning threshold for the average processing time of real-time events.
MC_RtEventAvgQueueAlert	Alert threshold for the average number of queued real-time events.
MC_RtEventAvgQueueTimeAlert	Alert threshold for average queuing time of real-time events.
MC_RtEventAvgQueueTimeWarning	Warning threshold for average queuing time of real-time events.
MC_RtEventAvgQueueWarning	Warning threshold for the average number of queued real-time events.
MC_RtEventErrorAlert	Alert threshold for processing errors of real-time events.



MC_RtEventErrorWarning	Warning threshold for processing errors of real-time events.
MC_RtEventMaxQueueAlert	Alert threshold for maximum number of queued real-time events.
MC_RtEventMaxQueueWarning	Warning threshold for maximum number of queued real-time events.
MC_RtEventMinQueueAlert	Alert threshold for minimum number of queued real-time events.
MC_RtEventMinQueueWarning	Warning threshold for minimum number of queued real-time events.
MC_RtEventQueueAlert	Threshold before critical condition for the queue of pending real time events.
MC_RtEventQueueWarning	Threshold before warning for the queue of pending real time events.
MC_RtEventThroughputAlert	Alert threshold for real-time event throughput.
MC_RtEventThroughputWarning	Warning threshold for real-time event throughput.
NmsMessageCenter_RoutingBatchSize	Size of the regrouping for the event routing.
MC_LastRtEventStat	Update pointer of RtEvent status (last date until when the data was retrieved).
NmsLine_MessageCenterURL	Message Center server URL used to send welcome messages (LINE channel).

## Database

Option name	Description
NmsCleanup_LastCleanup	Defines the last time the cleanup process was run.
NmsCleanup_BroadLogPurgeDelay	<p>Lets you define the delay after which broadlog are erased from the database.</p> <p>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.</p>
NmsCleanup_EventHistoPurgeDelay	<p>Lets you define the delay after which the event history is erased from the database.</p> <p>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.</p>
NmsCleanup_EventPurgeDelay	<p>Lets you define the delay after which events are erased from the database.</p> <p>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.</p>
NmsCleanup_EventStatPurgeDelay	<p>Lets you define the delay after which the event statistics are erased from the database.</p> <p>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.</p>
NmsCleanup_PropositionPurgeDelay	<p>Lets you define the delay after which propositions are erased from the database.</p> <p>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.</p>

NmsCleanup_QuarantineMailboxFull	<p>Lets you define the delay after which the quarantines are erased from the database.</p> <p>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.</p>
NmsCleanup_RecycledDeliveryPurgeDelay	<p>Lets you define the delay after which recycled deliveries are erased from the database.</p> <p>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.</p>
NmsCleanup_RejectsPurgeDelay	<p>Lets you define the delay after which rejects are erased from the database.</p> <p>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.</p>
NmsCleanup_TrackingLogPurgeDelay	<p>Lets you define the delay after which tracking logs are erased from the database.</p> <p>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.</p>
NmsCleanup_TrackingStatPurgeDelay	<p>Lets you define the delay after which tracking statistics is erased from the database.</p> <p>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.</p>
NmsCleanup_VisitorPurgeDelay	<p>Lets you define the delay after which visitors are erased from the database.</p> <p>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.</p>

NmsCleanup_WorkflowResultPurgeDelay	<p>Lets you define the delay after which workflow results is erased from the database.</p> <p>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.</p>
WdbcCapabilities_AzureDw	Azure SQL Datawarehouse connector options.
WdbcKillSessionPolicy	<p>Lets you affect Unconditional Stop behavior on all the workflows and PostgreSQL database queries according to the following potential values:</p> <ul style="list-style-type: none"> <li>• 0 – default: stops workflow process, no impact on the database</li> <li>• 1 - pg_cancel_backend: stops workflow process and cancels query in the database</li> <li>• 2 – pg_terminate_backend: stops workflow process and terminates query in the database</li> </ul>
WdbcOptions_TableSpaceUser	<p>Name of the tablespace intended to contain the data of the Adobe Campaign ootb tables. See <a href="#">Creating and configuring the database</a>.</p>
WdbcOptions_TableSpaceIndex	<p>Name of the tablespace intended to contain the indexes of the Adobe Campaign ootb tables. See <a href="#">Creating and configuring the database</a>.</p>
WdbcOptions_TableSpaceWork	<p>Name of the tablespace intended to contain the data of the Adobe Campaign work tables. See <a href="#">Creating and configuring the database</a>.</p>
WdbcOptions_TableSpaceWorkIndex	<p>Name of the tablespace intended to contain the indexes of the Adobe Campaign work tables. See <a href="#">Creating and configuring the database</a>.</p>
WdbcOptions_TempDbName	<p>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). <a href="#">Read more</a></p>
WdbcTimeZone	Time zone of the Adobe Campaign's instance. See <a href="#">Configuration</a> .
WdbcUseNChar	Are the database's string fields defined with NChar?
WdbcUseTimeStampWithTZ	Do the database's 'datetime' fields store timezone info?
XtkDatabaseId	ID of the database. Begins by 'u' for Unicode DataBase.
XtkInstancePrefix	Prefix added to internal names generated automatically.
XtkQuery_Schema_LineCount	Maximum number of results returned by a query on xtk:schema and xtk:srcSchema.
XtkSequence_AutoGeneration	All customized schemas, created after this time, with autopk="true" and without the attribute "pkSequence" will get an auto-generated sequence "auto_<schemanamespace> <schemaname> _seq"
NIMigration_KeepFolderStructure	<p>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.</p> <ul style="list-style-type: none"> <li>• Data type: Integer</li> <li>• Value (text) : 1</li> </ul> <p>This option should only be used if the out-of-the-box navigation tree has undergone too many changes.</p>
NmsLastErrorStatCoalesce	Last processing date of the NmsEmailErrorStat table cleanup.
PostUpgradeLastError	Information concerning the error that occurred in the Postupgrade, following the syntax below: <b>{Build number}:-{mode: pre/post/...}:{The 'lessThan'/'greaterOrEqualThan' where error occurred + sub-step}</b>
XtkCleanup_NoStats	Enter the "1" value so that the update of statistics is not performed through the cleanup workflow.

## Integration

Option name	Description
AEMResourceTypeFilter	Types of AEM resources that can be used in Adobe Campaign. Values must be separated by commas.
nmsPipeline_config	Lets you configure Experience Cloud Triggers. Data type is "long text" and must be in JSON format. See <a href="#">How to use Experience Cloud Triggers with Adobe Campaign Classic</a> .
LASTIMPORT_<%=instance.internalName%>_<%=activityName%>	<p>This option is used when importing data from a third-party system through a CRM connector. Enabling the option lets you collect only objects modified since the last import. This option has to be manually created and populated as below:</p> <ul style="list-style-type: none"> <li>Internal name : LASTIMPORT_&lt;%=instance.internalName%&gt;_&lt;%=activityName%&gt;</li> <li>Value (field) : date of the last import, with the yyyy/MM/dd hh:mm:ss format.</li> </ul>
TNT_EdgeServer	Adobe Target server used for the integration. This option is already selected by default. This value corresponds to the Adobe Target Domain Server, followed by the value /m2. For example: tt.omtrdc.net/m2. See <a href="#">this section</a> .
TNT_TenantName	Adobe Target Organization name. This value corresponds to the name of the Adobe Target Client. See <a href="#">this section</a> .
AAM_DataSourceId	Option used for the integration with Adobe Audience Manager.
AAM_DestinationId	Option used for the integration with Adobe Audience Manager.
WdbcCapabilities_Teradata	Teradata connector options.
WdbcCapabilities_Hive	Hive connector options.

## Offers

Option name	Description
NmsCoupons_MaxPerTransac	Number of coupons that are updated per SQL transaction.
NmsInteraction_LastPropositionSynchControl_	'+ [proposition's schema] + "_" + extAccountSource.@executionInstanceId + [proposition's schema] + "_" + vars.executionInstanceIdFilter
NmsInteraction_LastPropositionSynchExec_	'+ [proposition's schema] + "_" + extAccountSource.@executionInstanceId + "_" + extAccountTarget.@executionInstanceId
NmsInteraction_SynchWorkflowids	Synchronization workflow tracking.
NmsInteraction_UseDaemon	Enable/disable asynchronous proposition writing ("0" to disable, "1" to enable).
NmsModule_CouponsEnabled	Lets you enable coupons.

## Server

Option name	Description
NmsExecutionInstanceCeld	Execution instance identifier.
NmsServer_CustomerId	Customer identifier used when sending the billing report.
NmsServer_IntranetURL	Internal base URL to access the application server.
NmsServer_LastPostUpgrade	Build number of the AC instance before the last upgrade.
NmsServer_URL	Base URL of the public web application server.
XtkPassUnknownSQLFunctionsToRDBMS	Lets you continue using old undeclared SQL functions after migrating. We strongly recommend against using this option due to the security risks it introduces.

## Tracking

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

## Privacy

Option name	Description
Privacy_Request_ConfirmDeletePending	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.
Privacy_Request_ConfirmDeletePendingDelay	Delay between request waits for deleting confirmation and request is cancelled.
Privacy_Request_MaxErrorAllowed	The maximum number of error allowed when processing/deleting a privacy request.
Privacy_Request_PurgeDelay	Delay between request is created on the queue and request data is deleted.

## LDAP

Option name	Description
XtkLdap_Active	Enable LDAP server to be used to authenticate users and provide authorizations to users.
XtkLdap_AppLogin	Application login to contact the server for various searches.
XtkLdap_AppPassword	Encrypted password for the application login.
XtkLdap_AutoOperator	Enable automatic creation of operators and rights in Adobe Campaign.
XtkLdap_DN	Computation formula for LDAP DN based on login.
XtkLdap_DNSearch	Enable DN search in directory.
XtkLdap_DNSearchBase	Search base.
XtkLdap_DNSearchFilter	DN search filter.
XtkLdap_DNSearchScope	Search scope.
XtkLdap_Mechanism	Authentication type used to contact the LDAP server (plain, md5, lds, ntlm, dpa).
XtkLdap_Rights	Enable synchronization of authorizations and groups from LDAP directory to named rights in Adobe Campaign.
XtkLdap_RightsAttr	LDAP attribute containing the authorization name.
XtkLdap_RightsBase	Search base.
XtkLdap_RightsFilter	Search filter for user authorizations.
XtkLdap_RightsMask	Expression to extract the names of the Adobe Campaign rights from the LDAP authorizations.
XtkLdap_RightsScope	Search scope.
XtkLdap_Server	LDAP server address (it is possible to specify a port by specifying ':' as the separator).

## Web forms

Option name	Description
XtkUseScrollBar	Value set to 1 will allow Scroll bar addition to detail forms.
XtkWebForm_Instance	Instance to be used for web form invalidation in 'other server(s)' mode.
XtkWebForm_Password	Password of the instance to be used for web form invalidation in 'other server(s)' mode.
XtkWebForm_ServersMode	Option that lets you specify the invalidation mode of web forms: local by default, uses tracking servers if the option is 'tracking' and uses a personalized list with the 'other server(s)' option.
XtkWebForm_ServersURLs	Personalized address list of servers to be contacted for web form invalidation ('other server(s)' mode).

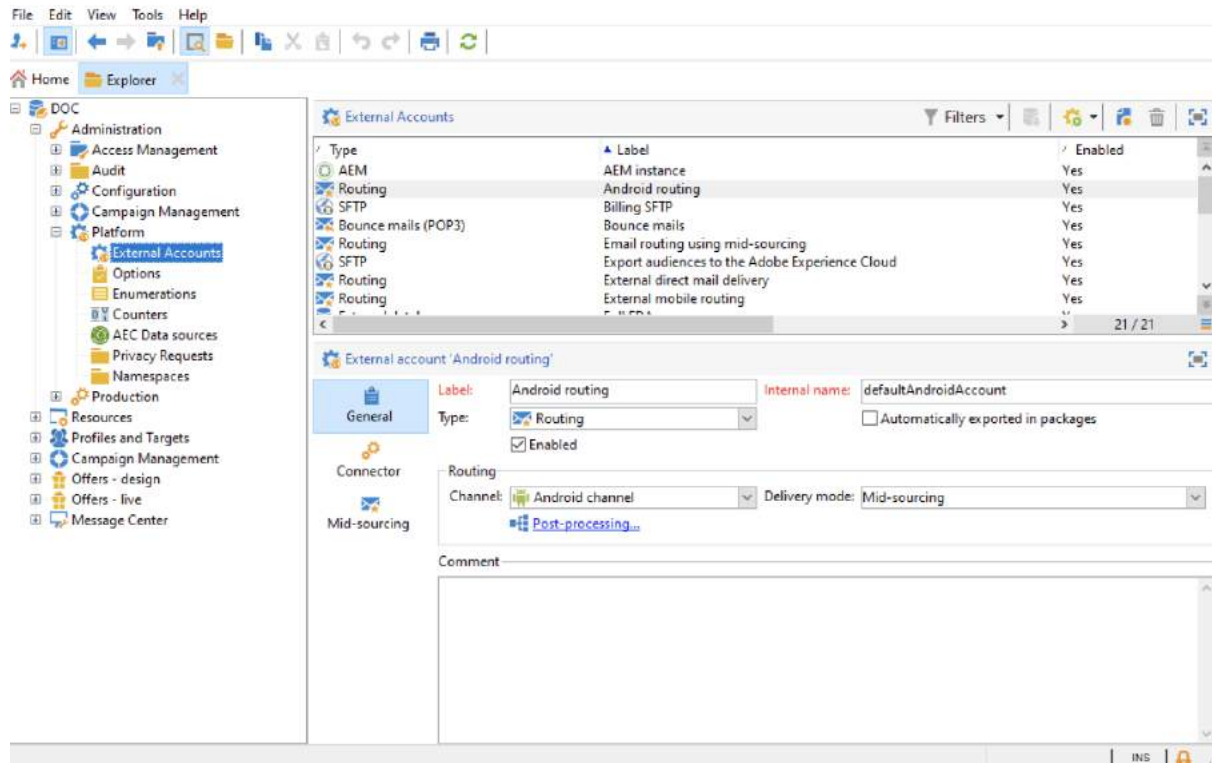
# 1.5 Configure your external accounts

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 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:

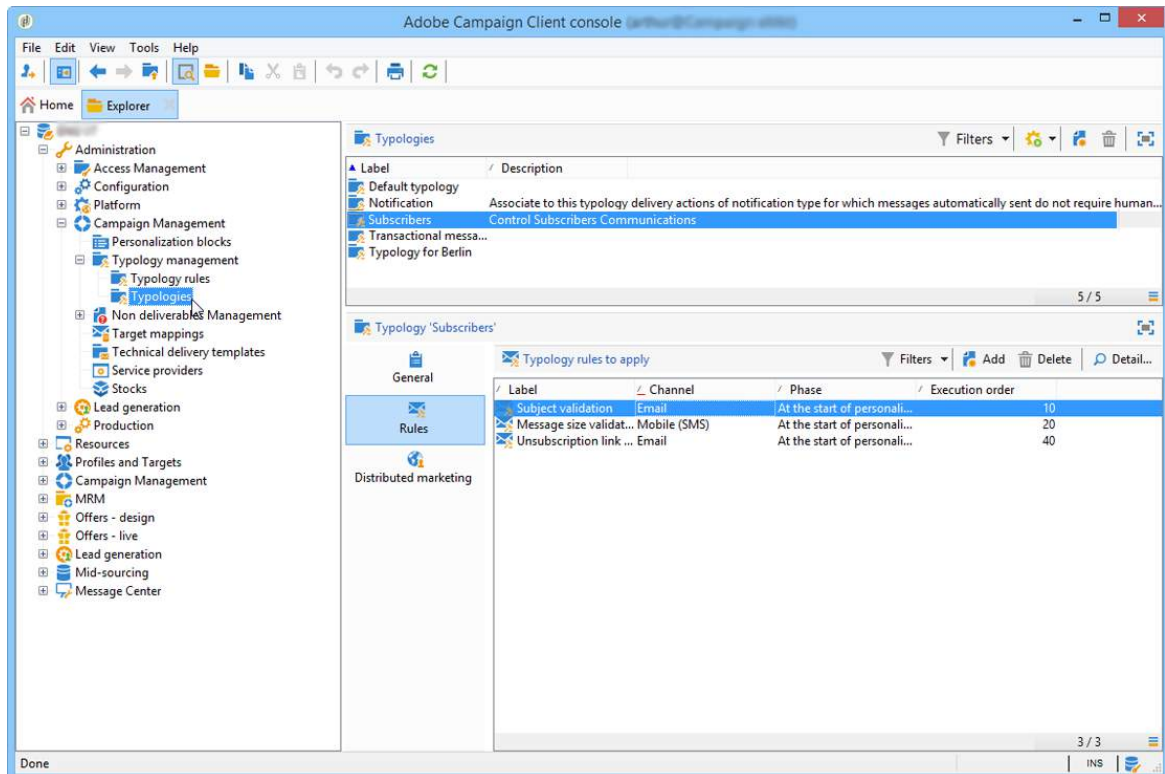
- **Filtering** rules which let you exclude part of the target based on criteria. For more on this, refer to [Filtering rules](#).
- **Pressure** rules which let you control marketing fatigue. For more on this, refer to [Pressure rules](#).
- **Capacity** rules which let you limit loads to guarantee 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](#)