# DBC XML Format Technical Reference

## Schema Document Properties

Properties

| | |
|---|---|
| **Target Namespace** | http://www.w3.org/namespace/ |
| **Element and Attribute Namespaces** | • Global element and attribute declarations belong to t<br>• By default, local element declarations have no name<br>• By default, local attribute declarations have no name |

Documentation

This document defines each element that can be used on DBC files, their attributes, and how to properly use them.

For a description of what DBC files are and how the Update DB proccess works, refer to Database Configuration Scripts.

Schema Component Representation

```
<schema targetNamespace="http://www.w3.org/namespace/">
...
</schema>
```

---

## Global Schema Components

### Element: script

Properties

| | |
|---|---|
| **Name** | script |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

This is the main element of the script that is required on all DBC files. It is good practice to correctly provide the `author` and `scriptname` attributes so that anyone can determine the purpose of the script and identify the creator of the script.

## Attributes:

- **author:** The name of the author of the file.
- **scriptname:** The name of the script.
- **for_demo_only:** If this attribute is present and defined as *true*, the script is executed only against `MaxDemo` databases and ignored on other cases. The default value is *false*.
- **for_install_only:** If this attribute is present and defined as *true*, the script is executed only in install mode and ignored on other cases. The default value is *false*
- **context:** The context where this script runs in a Multi-Tenant environment.
- **tenantcode:** The code of the tenant where this script runs. This attribute is only considered if **context** is defined as `tenants`.

The following example is the beginning of a regular DBC file:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE script SYSTEM "script.dtd">

<script author="John Doe" scriptname="Add Tenant License Use action menu to the Tenan
ts app">

  ...
```

XML Instance Representation

```
<script

 author="string" [1]

 scriptname="string" [1]

 for_demo_only="string (value comes from list: {'true'|'false'})" [0..1]

 for_install_only="string (value comes from list: {'true'|'false'})" [0..1]

 context="string (value comes from list: {'master'|'landlord'|'tenants'|'all'})" [0..
1]

 tenantcode="string" [0..1]

>

   <description> ... </description> [0..1]

   <check> ... </check> [0..*]

   <statements> ... </statements> [1]

</script>
```

Schema Component Representation

```xml
<element name="script">
   <complexType>
      <sequence>
         <element ref="description" minOccurs="0" maxOccurs="1"/>
         <element ref="check" minOccurs="0" maxOccurs="unbounded"/>
         <element ref="statements"/>
      </sequence>
      <attribute name="author" type="string" use="required"/>
      <attribute name="scriptname" type="string" use="required"/>
      <attribute name="for_demo_only" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="for_install_only" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="context" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="master"/>
               <enumeration value="landlord"/>
               <enumeration value="tenants"/>
               <enumeration value="all"/>
            </restriction>
```

```
          </simpleType>
       </attribute>
       <attribute name="tenantcode" type="string" use="optional"/>
    </complexType>
</element>
```

## Element: description

### Properties

| | |
|---|---|
| **Name** | description |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

This element provides a meaningful description of the actions taken by the DBC file.

Some examples of valid descriptions of different DBC files are:

- `<description>Inserting authorization for Data Sheet report application</description>`
- `<description>update relationship JOBPLANSPECCLASS and add org/site to unique index jobplanspec_ndx1. See Java File.</description>`
- `<description>Restores Assign and WMAssign class names to core values, and the same for Scheduler-related fields</description>`
- `<description>Add the new field OPTCLEANUP to the MAPMANAGER table and corntask for delete old SKDOriginDestMatrix</description>`
- `<description>Create a system property to define the maximum time since the last LBS update after which a labor/crew marker should not be displayed using the Nearby Resources Tool</description>`
- `<description>Creating sigoption for enabling the visualization of appointments made flag in the Create WO dialog for GAB.</description>`

### XML Instance Representation

```
<description/>
```

### Schema Component Representation

```
<element name="description">
   <complexType mixed="true"/>
</element>
```

## Element: check

### Properties

| | |
|---|---|
| **Name** | check |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

This section defines the checks to make before running the script. It can have one or more check_query children elements and each one performs an SQL query against the database. If any query returns a row, the information is logged and usually the script is skipped. The first check that has a query return a row takes precedence. If no query returns rows, the script will be run.

Generally used to skip running the script if the script contents have already been applied.

When a `check_query` needs to register a log message, an internal `MXException` is created and passed to the Maximo logging mechanism. It is possible to pass information to that object using the attributes `group`, `key` and `default`, overriding the default values, as described below:

## Attributes:

- **tag:** If this attribute is set to `WARNMODIFICATION`, the logged message generated by execution of the check is tagged as a warning, instead of the default `info` tag.
- **group:** Defines the Error Group used by the `MXException`.
- **key:** Defines the Error Key used by the `MXException`.
- **default:** Defines the Error Message used by the `MXException`.
- **skip_script:** If `true`, the script runs even if any `check_query` returns any rows.

### XML Instance Representation

```
<check
 tag="string" [0..1]
 group="string" [0..1]
 key="string" [0..1]
 default="string" [0..1]
 skip_script="string (value comes from list: {'true'|'false'})" [0..1]
>
   <check_query> ... </check_query> [1..*]
</check>
```

Schema Component Representation

```
<element name="check">
   <complexType>
      <sequence>
         <element ref="check_query" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="tag" type="string" use="default" value="INFO"/>
      <attribute name="group" type="string" use="default" value="scriptrun"/>
      <attribute name="key" type="string" use="default" value="ScriptNotNeeded"/>
      <attribute name="default" type="string" use="default" value="This script was in
tentionally skipped."/>
      <attribute name="skip_script" use="default" value="true">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: check_query

Properties

| | |
|---|---|
| **Name** | check_query |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

## Documentation

It works in conjunction with check, please read about it first.

Before running a ascript, it is a good practice to always check if the changes of the script were already applied. An example of how to use `check_query` is if a new attribute, called `RELATIONSHIP`, should be added to the object `WFASSIGNMENT`. A safe script should test the existence of the attribute. The `check` and `check_query` elements can be used to validate this pre-condition:

```
<check>

    <check_query>select 1 from maxattribute where objectname = 'WFASSIGNMENT' and att
ributename = 'RELATIONSHIP'</check_query>

</check>
```

Another example is before a new object is created, the script should include a check to validate if the object already exists:

```
<check>

    <check_query>select 1 from maxintobject where intobjectname ='MXSCRIPT'</check_qu
ery>

</check>
```

**Note:** Don't include a terminator at the end of the SQL command - `;` or `go`.

## XML Instance Representation

```
<check_query

 query="string" [1]

/>
```

## Schema Component Representation

```
<element name="check_query">

   <complexType>

      <attribute name="query" type="string" use="required"/>

   </complexType>

</element>
```

# Element: statements

## Properties

| | |
|---|---|
| **Name** | statements |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

## Documentation

This element encloses all the elements that represents runnable statements that will perform the necessary changes on database schema or it's data. Each element runs following the order of appearence inside this tag and, if any error is captured, the update process is aborted and the error is logged.

## XML Instance Representation

```
<statements>
   Start Choice [0..*]
      <create_relationship> ... </create_relationship> [1]
      <modify_relationship> ... </modify_relationship> [1]
      <drop_relationship> ... </drop_relationship> [1]
      <freeform> ... </freeform> [1]
      <add_attributes> ... </add_attributes> [1]
      <modify_attribute> ... </modify_attribute> [1]
      <drop_attributes> ... </drop_attributes> [1]
      <define_table> ... </define_table> [1]
      <modify_table> ... </modify_table> [1]
      <drop_table> ... </drop_table> [1]
      <specify_index> ... </specify_index> [1]
      <drop_index> ... </drop_index> [1]
      <specify_synonym_domain> ... </specify_synonym_domain> [1]
      <add_synonyms> ... </add_synonyms> [1]
      <specify_aln_domain> ... </specify_aln_domain> [1]
      <specify_numeric_domain> ... </specify_numeric_domain> [1]
      <specify_crossover_domain> ... </specify_crossover_domain> [1]
      <drop domain> ... </drop domain> [1]
      <specify table domain> ... </specify table domain> [1]
      <add sigoption> ... </add sigoption> [1]
      <drop sigoption> ... </drop sigoption> [1]
      <create maxvar> ... </create maxvar> [1]
      <modify maxvar> ... </modify maxvar> [1]
      <drop maxvar> ... </drop maxvar> [1]
      <modify domain type> ... </modify domain type> [1]
      <add service> ... </add service> [1]
      <modify service> ... </modify service> [1]
      <drop service> ... </drop service> [1]
      <create app> ... </create app> [1]
      <modify app> ... </modify app> [1]
      <drop app> ... </drop app> [1]
      <create module> ... </create module> [1]
      <modify module> ... </modify module> [1]
      <drop module> ... </drop module> [1]
      <create app menu> ... </create app menu> [1]
      <additional app menu> ... </additional app menu> [1]
      <define view> ... </define view> [1]
      <modify view> ... </modify view> [1]
      <drop view> ... </drop view> [1]
      <drop view attribute> ... </drop view attribute> [1]
      <add view attribute> ... </add view attribute> [1]
      <modify view attributes> ... </modify view attributes> [1]
      <add property> ... </add property> [1]
      <set property> ... </set property> [1]
      <drop property> ... </drop property> [1]
      <module app> ... </module app> [1]
      <insert> ... </insert> [1]
      <logical relationship> ... </logical relationship> [1]
   End Choice
</statements>
```

Schema Component Representation

```
<element name="statements">
   <complexType>
      <choice minOccurs="0" maxOccurs="unbounded">
         <element ref="create_relationship"/>
         <element ref="modify_relationship"/>
         <element ref="drop_relationship"/>
         <element ref="freeform"/>
         <element ref="add_attributes"/>
         <element ref="modify_attribute"/>
         <element ref="drop_attributes"/>
         <element ref="define_table"/>
         <element ref="modify_table"/>
         <element ref="drop_table"/>
         <element ref="specify_index"/>
         <element ref="drop_index"/>
         <element ref="specify_synonym_domain"/>
         <element ref="add_synonyms"/>
         <element ref="specify_aln_domain"/>
         <element ref="specify_numeric_domain"/>
         <element ref="specify_crossover_domain"/>
         <element ref="drop_domain"/>
         <element ref="specify_table_domain"/>
         <element ref="add_sigoption"/>
         <element ref="drop_sigoption"/>
         <element ref="create_maxvar"/>
         <element ref="modify_maxvar"/>
         <element ref="drop_maxvar"/>
         <element ref="modify_domain_type"/>
         <element ref="add_service"/>
         <element ref="modify_service"/>
         <element ref="drop_service"/>
         <element ref="create_app"/>
         <element ref="modify_app"/>
```

```
          <element ref="drop_app"/>
          <element ref="create_module"/>
          <element ref="modify_module"/>
          <element ref="drop_module"/>
          <element ref="create_app_menu"/>
          <element ref="additional_app_menu"/>
          <element ref="define_view"/>
          <element ref="modify_view"/>
          <element ref="drop_view"/>
          <element ref="drop_view_attribute"/>
          <element ref="add_view_attribute"/>
          <element ref="modify_view_attributes"/>
          <element ref="add_property"/>
          <element ref="set_property"/>
          <element ref="drop_property"/>
          <element ref="module_app"/>
          <element ref="insert"/>
          <element ref="logical_relationship"/>
      </choice>
   </complexType>
</element>
```

## Element: modify_domain_type

Properties

| | |
|---|---|
| **Name** | modify_domain_type |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

This statement allows you to change a Domain's maxtype definition. This definition is applied to all columns that are defined as using the domain.

## Attributes:

- **domain:** Targeted domain.
- **maxtype:** The new Domain's maxtype.
- **length:** Set the length of the maxtype, when appropriate. Must be used only with numeric domains.
- **scale:** Set the scale of the maxtype, when appropriate. Must be used only with decimal domains.

## Use Example:

```
<modify_domain_type domain="DOMAINNAME" length="25" maxtype="ALN" scale="0" />
```

XML Instance Representation

```
<modify_domain_type

 domain="string" [1]

 maxtype="string (value comes from list: {'ALN'|'LONGALN'|'LOWER'|'UPPER'|'AMOUNT'|'D
ECIMAL'|'DURATION'|'FLOAT'|'INTEGER'|'SMALLINT'})" [0..1]

 length="NMTOKEN" [0..1]

 scale="NMTOKEN" [0..1]

/>
```

Schema Component Representation

```
<element name="modify_domain_type">

   <complexType>

      <attribute name="domain" type="string" use="required"/>

      <attribute name="maxtype" use="optional">

         <simpleType>

            <restriction base="string">

               <enumeration value="ALN"/>

               <enumeration value="LONGALN"/>

               <enumeration value="LOWER"/>

               <enumeration value="UPPER"/>

               <enumeration value="AMOUNT"/>
```

```
                <enumeration value="DECIMAL"/>
                <enumeration value="DURATION"/>
                <enumeration value="FLOAT"/>
                <enumeration value="INTEGER"/>
                <enumeration value="SMALLINT"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="length" type="NMTOKEN" use="optional"/>
       <attribute name="scale" type="NMTOKEN" use="optional"/>
    </complexType>
</element>
```

## Element: define_view

### Properties

| | |
|---|---|
| **Name** | define_view |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Creates a database view based on the query defined using table, view_select, view_from and view_where (optional).

The view is associated with a Maximo Business Object (MBO), that must already exists on Maximo Asset Management.

### Attributes:

- **name:** The name of the new view.
- **description:** A meaninful description about the view and why its is necessary.
- **service:** The Maximo service associated with the MBO.
- **classname:** The fully qualified name of the MBOSet class.

- **extends:** If this MBO extends another existing MBO, the name of this MBO can be assigned here.
- **type:** Access level of the view.
- **mainobject:** Defines if the MBO should be a main object or not.
- **internal:** Indicates that the object is reserved for internal use by the platform and cannot be altered by using the Database Configuration application.

## Use Example:

For example, in order to create a view that combines `SKDWORKPLANEXTCAP` and `CRAFTSKILL`, the element should looks like:

```
<define_view name="SKDExtraCapCraftView" classname="com.ibm.tivoli.maximo.skd.app.SKDExtraCapCraftViewSet" service="SCHEDULER" extends="SKDWORKPLANEXTCAP" type="system" description="View that combines SKDWORKPLANEXTCAP and CRAFTSKILL.">

    <autoselect/>

    <table name="skdworkplanextcap"/>

    <table name="craftskill"/>

    <view_where>skdworkplanextcap.resourcename=craftskill.craft and skdworkplanextcap.orgid=craftskill.orgid  and (skdworkplanextcap.skilllevel=craftskill.skilllevel or (craftskill.skilllevel is null and skdworkplanextcap.skilllevel is null)) </view_where>

</define_view>
```

XML Instance Representation

```
<define_view

 name="string" [1]

 description="string" [1]

 service="string" [1]

 classname="string" [1]

 extends="string" [0..1]

 type="string (value comes from list: {'system'|'site'|'companyset'|'itemset'|'org'|'orgappfilter'|'orgsite'|'siteappfilter'|'systemappfilter'|'systemorg'|'systemorgsite'|'systemsite'})" [1]

 mainobject="string (value comes from list: {'true'|'false'})" [0..1]

 internal="string (value comes from list: {'true'|'false'})" [0..1]

>

   Start Choice [1]

      <autoselect> ... </autoselect> [1]

      <table> ... </table> [1..*]
      <view_column> ... </view_column> [0..*]
      <view_select> ... </view_select> [1]
      <view_from> ... </view_from> [1]
   End Choice
   <view_where> ... </view_where> [1]
   <longdescription> ... </longdescription> [0..1]
</define_view>
```

Schema Component Representation

```
<element name="define_view">
   <complexType>
      <sequence>
         <choice>
            <sequence>
               <element ref="autoselect"/>
               <element ref="table" maxOccurs="unbounded"/>
               <element ref="view_column" minOccurs="0" maxOccurs="unbounded"/>
               <element ref="view_select"/>
               <element ref="view_from"/>
            </sequence>
         </choice>
         <element ref="view_where"/>
         <element ref="longdescription" minOccurs="0" maxOccurs="1"/>
      </sequence>
      <attribute name="name" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="service" type="string" use="required"/>
      <attribute name="classname" type="string" use="required"/>
      <attribute name="extends" type="string" use="optional"/>
      <attribute name="type" use="required">
         <simpleType>
            <restriction base="string">
               <enumeration value="system"/>
               <enumeration value="site"/>
```

```
                <enumeration value="companyset"/>
                <enumeration value="itemset"/>
                <enumeration value="org"/>
                <enumeration value="orgappfilter"/>
                <enumeration value="orgsite"/>
                <enumeration value="siteappfilter"/>
                <enumeration value="systemappfilter"/>
                <enumeration value="systemorg"/>
                <enumeration value="systemorgsite"/>
                <enumeration value="systemsite"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="mainobject" use="default" value="false">
          <simpleType>
             <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="internal" use="default" value="false">
          <simpleType>
             <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: autoselect

### Properties

| | |
|---|---|
| **Name** | autoselect |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

AutoSelect views must reference all columns of the table that the view extends.

### XML Instance Representation

```
<autoselect/>
```

### Schema Component Representation

```
<element name="autoselect">
   <complexType/>
</element>
```

## Element: table

### Properties

| | |
|---|---|
| **Name** | table |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Defines a table that is part of the query present in view_select, view_from and view_where. There must be one table element for each table present in the query.

### Attributes:

- **name:** Table's name.

### Use Example:

The example for define_view uses two tables in the query with two table elements:

```
<table name="skdworkplanextcap"/>

<table name="craftskill"/>
```

XML Instance Representation

```
<table
 name="string" [1]
/>
```

Schema Component Representation

```
<element name="table">
   <complexType>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: view_column

Properties

|  |  |
|---|---|
| **Name** | view_column |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines a view's column. If the element is not present, the columns are infered from the query.

### Attributes:

- **view:** Name of the target view.

- **view_column:** Name of the attribute in the view.
- **table:** Name of the table that contains the column.
- **column:** Name of the column.
- **same_storage_as:** Defines this attribute as a alias of another column.

## Use Example:

The columns `sets.setid`, `groupuser.userid` and `applicationauth.app` are defined in the following example:

```
<define_view service="CONFIGURE" name="companysetfilter"

  <view_column column="setid" table="sets" view_column="setid"/>

  <view_column column="userid" table="groupuser" view_column="userid"/>

  <view_column column="app" table="applicationauth" view_column="app"/>

  <view_select>select distinct sets.setid, groupuser.userid, applicationauth.app </vi
ew_select>

  <view_from>sets, organization, groupuser, applicationauth </view_from>

  <view_where>sets.setid = organization.companysetid and settype in (select value fro
m synonymdomain where domainid='SETTYPE' and maxvalue = 'COMPANY') and organization.o
rgid in (select orgid from orgfilter where userid = groupuser.userid and app = applic
ationauth.app)</view_where>

</define_view>
```

### XML Instance Representation

```
<view_column
 table="string" [1]
 column="string" [1]
 view_column="string" [1]
 same_storage_as="string" [0..1]
/>
```

### Schema Component Representation

```
<element name="view_column">
   <complexType>
      <attribute name="table" type="string" use="required"/>
      <attribute name="column" type="string" use="required"/>
      <attribute name="view_column" type="string" use="required"/>
      <attribute name="same_storage_as" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: view_select

### Properties

| | |
|---|---|
| **Name** | view_select |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Defines which columns are returned in the result set and is equivalent to the *select* part of a SQL query. An example is shown in the view_column element and is Used to get only three columns: `sets.setid`, `groupuser.userid` and `applicationauth.app`:

## Use Example:

```
<view_select>select distinct sets.setid, groupuser.userid, applicationauth.app </view
_select>
```

### XML Instance Representation

```
<view_select/>
```

### Schema Component Representation

```
<element name="view_select">
   <complexType mixed="true"/>
</element>
```

## Element: view_from

### Properties

| | |
|---|---|
| **Name** | view_from |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines which tables is part of the query and is equivalent to the *from* part of a SQL query. For examples on how this element is used, see the define_view and view_column elements:

## Use Example:

```
<view_where>sets.setid = organization.companysetid and settype in (select value from synonymdomain where domainid='SETTYPE' and maxvalue = 'COMPANY') and organization.orgid in (select orgid from orgfilter where userid = groupuser.userid and app = applicationauth.app)</view_where>

<view_where>skdworkplanextcap.resourcename=craftskill.craft and skdworkplanextcap.orgid=craftskill.orgid  and (skdworkplanextcap.skilllevel=craftskill.skilllevel or (craftskill.skilllevel is null and skdworkplanextcap.skilllevel is null)) </view_where>
```

XML Instance Representation

```
<view_from/>
```

Schema Component Representation

```
<element name="view_from">
   <complexType mixed="true"/>
</element>
```

## Element: view_where

Properties

| | |
|---|---|
| **Name** | view_where |
| **Type** | Locally-defined complex type |
| **Nillable** | no |

| | |
|---|---|
| **Abstract** | no |

Documentation

Defines which data is present on the result set based on SQL-like conditions and is equivalent to the *where* part of a SQL query. For examples on how this element is used, see the define_view and view_column elements:

## Use Example:

```
<view_where>sets.setid = organization.companysetid and settype in (select value from
synonymdomain where domainid='SETTYPE' and maxvalue = 'COMPANY') and organization.org
id in (select orgid from orgfilter where userid = groupuser.userid and app = applicat
ionauth.app)</view_where>


<view_where>skdworkplanextcap.resourcename=craftskill.craft and skdworkplanextcap.org
id=craftskill.orgid  and (skdworkplanextcap.skilllevel=craftskill.skilllevel or (craf
tskill.skilllevel is null and skdworkplanextcap.skilllevel is null)) </view_where>
```

XML Instance Representation

```
<view_where/>
```

Schema Component Representation

```
<element name="view_where">

   <complexType mixed="true"/>

</element>
```

## Element: modify_view

Properties

| | |
|---|---|
| **Name** | modify_view |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies an existing view if the name matches the value of the **name** attribute. This element accepts the same attributes and child elements of define_view.

## Attributes:

- **name:** View's name.
- **classname:** Full Qualified Name (FQN) of the View's object class name.
- **description:** Description of the responsibilities of this particular view.
- **mainobject:** Object that refers to the the view's based table.
- **internal:** Indicates that the object is reserved for internal use by the platform and cannot be altered by using the Database Configuration application.
- **service:** Service name if exists one only for this particular app.
- **type:** View's visibility level

## Use Example:

It performs an update on the selected view and affects only the attributes and elements that are present in the `modify_view` tag, with the exception of the attribute **name**.

The following example changes the view_select value of the `UNASSIGNEDWORKVIEW` to the value defined below:

```
<modify_view name="UNASSIGNEDWORKVIEW">

  <view_select> wo.workorderid,wo.wonum ,wo.siteid,wo.orgid,wo.parent,wo.status,wo.de
scription,wo.schedstart ,wo.schedfinish, wo.amcrew,wo.assetnum,wo.location,wo.crewwor
kgroup,wo.hasld,wo.worktype,wo.wopriority,ws.latitudey,ws.longitudex, wo.targstartdat
e, wo.targcompdate</view_select>

</modify_view>
```

**Note:** autoselect views can not be modified.

XML Instance Representation

```
<modify_view

 name="string" [1]

 classname="string" [0..1]

 description="string" [0..1]

 mainobject="string (value comes from list: {'true'|'false'})" [0..1]

 internal="string (value comes from list: {'true'|'false'})" [0..1]

 service="string" [0..1]

 type="string (value comes from list: {'system'|'site'|'companyset'|'itemset'|'org'|'
orgappfilter'|'orgsite'|'siteappfilter'|'systemappfilter'|'systemorg'|'systemorgsite'
|'systemsite'})" [0..1]

>

   <view_select> ... </view_select> [0..1]

   <view_from> ... </view_from> [0..1]
   <view_where> ... </view_where> [0..1]
   <longdescription> ... </longdescription> [0..1]
</modify_view>
```

Schema Component Representation

```
<element name="modify_view">
   <complexType>
      <sequence>
         <element ref="view_select" minOccurs="0" maxOccurs="1"/>
         <element ref="view_from" minOccurs="0" maxOccurs="1"/>
         <element ref="view_where" minOccurs="0" maxOccurs="1"/>
         <element ref="longdescription" minOccurs="0" maxOccurs="1"/>
      </sequence>
      <attribute name="name" type="string" use="required"/>
      <attribute name="classname" type="string" use="optional"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="mainobject" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="internal" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="service" type="string" use="optional"/>
      <attribute name="type" use="optional">
```

```
          <simpleType>
             <restriction base="string">
                <enumeration value="system"/>
                <enumeration value="site"/>
                <enumeration value="companyset"/>
                <enumeration value="itemset"/>
                <enumeration value="org"/>
                <enumeration value="orgappfilter"/>
                <enumeration value="orgsite"/>
                <enumeration value="siteappfilter"/>
                <enumeration value="systemappfilter"/>
                <enumeration value="systemorg"/>
                <enumeration value="systemorgsite"/>
                <enumeration value="systemsite"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: drop_view

### Properties

| | |
|---|---|
| **Name** | drop_view |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Drops an existing view if the name matches the value of the **name** attribute.

### Attributes:

- **name:** View's name that should be deleted.

### Use Example:

The following markup drops the `DBINFOOBJECTVIEW` view:

```
<drop_view name="DBINFOOBJECTVIEW"/>
```

**Note:** autoselect views can not be dropped.

#### XML Instance Representation

```
<drop_view
 name="string" [1]
/>
```

#### Schema Component Representation

```
<element name="drop_view">
   <complexType>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: drop_view_attribute

#### Properties

| | |
|---|---|
| **Name** | drop_view_attribute |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

#### Documentation

Drops a single view's attribute if the name matches the value of the **attribute** attribute.

A modify_view element must be used to remove the dropped attribute from the view's *select*, *from* and *where*.

**Note:** autoselect views can not be modified.

## Attributes:

- **view:** Target view's name.
- **attribute:** Attribute that should be dropped.

## Use Example:

```
<drop_view_attribute view="VIEWNAME" attribute="ATTRIBUTENAME" />
```

XML Instance Representation

```
<drop_view_attribute
 view="string" [1]
 attribute="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_view_attribute">
   <complexType>
      <attribute name="view" type="string" use="required"/>
      <attribute name="attribute" type="string" use="required"/>
   </complexType>
</element>
```

## Element: add_view_attribute

Properties

| | |
|---|---|
| **Name** | add_view_attribute |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Adds a new attribute to an existing view. This attribute must be a column from one of the tables present in the view's query. This element has the same attributes of view_column element.

A modify_view element must be used to add the new attribute to the view's *select*, *from* and *where*, if necessary.

**Note:** autoselect views can not be modified.

## Attributes:

- **view:** View name.
- **view_column:** Attribute name to be added.
- **table:** Table name of the based table.
- **column:** Attribute name from based table;
- **same_storage_as:** Representes the storage definitions of another existent storage.

## Use Example:

```
<add_view_attribute view="VIEWNAME" view_column="VIEWATTRIBUTENAME" table="TABLENAME"
column="TABLEATTRIBUTENAME" />
```

XML Instance Representation

```
<add_view_attribute
 view="string" [1]
 view_column="string" [1]
 table="string" [1]
 column="string" [1]
 same_storage_as="string" [0..1]
/>
```

Schema Component Representation

```
<element name="add_view_attribute">
   <complexType>
      <attribute name="view" type="string" use="required"/>
      <attribute name="view_column" type="string" use="required"/>
      <attribute name="table" type="string" use="required"/>
      <attribute name="column" type="string" use="required"/>
      <attribute name="same_storage_as" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: modify_view_attributes

### Properties

| | |
|---|---|
| **Name** | modify_view_attributes |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Modifies one or more attributes of an existing view. It holds a sequence of modify_view_data where each one acts on a single attribute.

**Note:** autoselect views can not be modified.

### Attributes:

Refer to modify_view_data to futher information about the element.

- **view:** View that should have the attributes modified.

## Use Example:

```
<modify_view_attributes view="VIEWNAME" >

    <modify_view_data view_column="VIEWATTRIBUTE" column="TABLEATTRIBUTE" new_name="V
IEWNEWNAME" table="TABLENAME" same_storage_as="STORAGE"/>

</modify_view_attributes >
```

### XML Instance Representation

```
<modify_view_attributes

 view="string" [1]

>

   <modify_view_data> ... </modify_view_data> [1..*]

</modify_view_attributes>
```

### Schema Component Representation

```
<element name="modify_view_attributes">

   <complexType>
```

```
      <sequence>
         <element ref="modify_view_data" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="view" type="string" use="required"/>
   </complexType>
</element>
```

## Element: modify_view_data

Properties

| | |
|---|---|
| **Name** | modify_view_data |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies a single attribute of an existing view. This element has the same attributes of view_column element. It must be used in conjunction with modify_view_attributes.

### Attributes:

- **new_name:** New name for the particular view.
- **view_column:** Attribute name to be added.
- **table:** Table name of the based table.
- **column:** Attribute name from based table.
- **same_storage_as:** Representes the storage definitions of another existent storage.

### Use Example:

You can find the same sample from the modify_view_attributes element sample.

```
<modify_view_attributes view="VIEWNAME" >
    <modify_view_data view_column="VIEWATTRIBUTE" column="TABLEATTRIBUTE" new_name="V
IEWNEWNAME" table="TABLENAME" same_storage_as="STORAGE"/>
</modify_view_attributes >
```

XML Instance Representation

```
<modify_view_data
 view_column="string" [1]
 new_name="string" [0..1]
 table="string" [0..1]
 column="string" [0..1]
 same_storage_as="string" [0..1]
/>
```

Schema Component Representation

```
<element name="modify_view_data">
   <complexType>
      <attribute name="view_column" type="string" use="required"/>
      <attribute name="new_name" type="string" use="optional"/>
      <attribute name="table" type="string" use="optional"/>
      <attribute name="column" type="string" use="optional"/>
      <attribute name="same_storage_as" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: create_maxvar

Properties

| | |
|---|---|
| **Name** | create_maxvar |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation
Creates a new Maxvar with the given attributes.

### Attributes:

- **name:** Name of the Maxvar.
- **description:** A meaningful description about the Maxvar and why the Maxvar is necessary.
- **default:** The default value of the Maxvar, if any.
- **type:** Access level of the Maxvar.

## Use Example:

The following example create the `USECLIENTTIMEZONE` Maxvar:

```
<create_maxvar name="USECLIENTTIMEZONE" description="Use the client timezone for labt
rans" default="0" type="system" />
```

XML Instance Representation

```
<create_maxvar

 name="string" [1]

 description="string" [1]

 default="string" [0..1]

 type="string (value comes from list: {'system'|'site'|'organization'|'system_tenant'
})" [1]

/>
```

Schema Component Representation

```
<element name="create_maxvar">

   <complexType>

      <attribute name="name" type="string" use="required"/>

      <attribute name="description" type="string" use="required"/>

      <attribute name="default" type="string" use="optional"/>

      <attribute name="type" use="required">

         <simpleType>

            <restriction base="string">

               <enumeration value="system"/>

               <enumeration value="site"/>

               <enumeration value="organization"/>

               <enumeration value="system_tenant"/>

            </restriction>

         </simpleType>

      </attribute>
   </complexType>
</element>
```

## Element: modify_maxvar

### Properties

| | |
|---|---|
| **Name** | modify_maxvar |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Modifies an existing Maxvar where the name matches the value of the **name** attribute. This element accepts the same attributes of create_maxvar.

The element performs an update on the selected Maxvar and affects only the attributes that are present in the `modify_maxvar` tag, with the exception of the attribute **name**.

### Attributes:

- **name:** Variable's name
- **description:** Description about the variable Use.
- **default:** Variable's default value.
- **type:** Variable type represents the level this variable should be applied [System|Site|Organization].

## Use Example:

The following example changes the attributes **description**, **default** and **type** of `USECALFORSLAESCPT`:

```
<modify_maxvar name="USECALFORSLAESCPT" description="Use Calendars" default="1" type=
"site" />
```

### XML Instance Representation

```
<modify_maxvar
 name="string" [1]
 description="string" [0..1]
 default="string" [0..1]
 type="string (value comes from list: {'system'|'site'|'organization'|'system_tenant'
})" [0..1]
/>
```

Schema Component Representation

```
<element name="modify_maxvar">
   <complexType>
      <attribute name="name" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="default" type="string" use="optional"/>
      <attribute name="type" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="system"/>
               <enumeration value="site"/>
               <enumeration value="organization"/>
               <enumeration value="system_tenant"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: drop_maxvar

Properties

| | |
|---|---|
| **Name** | drop_maxvar |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drops a existing Maxvar.

### Attributes:

- **name:** Name of the Maxvar.

### Use Example:

The following example drops a Maxvar named `PLUSCMOBREADONLY`:

```
<drop_maxvar name="PLUSCMOBREADONLY" />
```

XML Instance Representation

```
<drop_maxvar
 name="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_maxvar">
   <complexType>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: create_relationship

Properties

| | |
|---|---|
| **Name** | create_relationship |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Creates a Maximo relationship between the **parent** and **child** objects that are ruled by the **whereclause** value, a SQL-like that will join **parent** and **child**.

In the where clause, fields that are defined with ":" are the attributes of parent object. The other fields are from child object.

## Attributes:

- **parent:** Name of the Maximo object indicating *from* where is the relationship.
- **name:** Relationship's name, to be used everytime the relationship needs to be referred, as in the presentation xml.
- **child:** Name of Maximo object indicating the *to* of the relationship, the target.
- **whereclause:** SQL-like where clause of the relationship.
- **remarks:** A meaningful description about the relationship.

## Use Example:

The following example creates a relationship named `SKDODMECONSTRAINTINFO` between `SKDODMECONFLICTMESSAGE` and `SKDODMECONSTRAINTINFO` objects and returns all `SKDODMECONSTRAINTINFO` records that has the `constraintname` field equal to the parent `SKDODMECONFLICTMESSAGE.constraintname`. The field `:constraintname` refers to the value of `constraintname` column from `SKDODMECONFLICTMESSAGE` table:

```
<create_relationship name="SKDODMECONSTRAINTINFO" parent="SKDODMECONFLICTMESSAGE" child="SKDODMECONSTRAINTINFO" whereclause="constraintname=:constraintname" remarks="Relationship to list constraint description for the given conflict message" />
```

### XML Instance Representation

```
<create_relationship
 parent="string" [1]
 name="string" [1]
 child="string" [1]
 whereclause="string" [1]
 remarks="string" [1]
/>
```

### Schema Component Representation

```
<element name="create_relationship">
   <complexType>
      <attribute name="parent" type="string" use="required"/>
      <attribute name="name" type="string" use="required"/>
      <attribute name="child" type="string" use="required"/>
      <attribute name="whereclause" type="string" use="required"/>
      <attribute name="remarks" type="string" use="required"/>
   </complexType>
</element>
```

## Element: modify_relationship

Properties

| | |
|---|---|
| **Name** | modify_relationship |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies an existing Maximo relationship where the name matches the value of the **name** attribute. This element accepts the same attributes of create_relationship.

It performs an update on the selected relationship and affects only the attributes present inside `modify_relationship` tag, with the exception of the attribute **name**.

### Attributes:

- **parent:** Name of the Maximo object indicating *from* where is the relationship.
- **name:** Relationship's name, to be used everytime the relationship needs to be referred, as in the presentation xml.
- **child:** Name of Maximo object indicating the *to* of the relationship, the target.
- **whereclause:** SQL-like where clause of the relationship.
- **remarks:** A meaninful description about the relationship.

### Use Example:

The following example changes the attributes **description**, **default** and **type** of `USECALFORSLAESCPT`:

```
<modify_maxvar name="USECALFORSLAESCPT" description="Use Calendars" default="1" type=
"site" />
```

XML Instance Representation

```
<modify_relationship
```

```
 parent="string" [1]
 name="string" [1]
 child="string" [0..1]
 whereclause="string" [0..1]
 remarks="string" [0..1]
/>
```

Schema Component Representation

```
<element name="modify_relationship">
   <complexType>
      <attribute name="parent" type="string" use="required"/>
      <attribute name="name" type="string" use="required"/>
      <attribute name="child" type="string" use="optional"/>
      <attribute name="whereclause" type="string" use="optional"/>
      <attribute name="remarks" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: drop_relationship

Properties

| | |
|---|---|
| **Name** | drop_relationship |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drops a existing Maximo relationship.

### Attributes:

- **parent:** Name of the parent object.
- **name:** Name of the relationship.

## Use Example:

The following example drops a relationship named `BUDGETANALYSISAXIS` from `BUDGET`:

```
<drop_relationship name="BUDGETANALYSISAXIS" parent="BUDGET"/>
```

### XML Instance Representation

```
<drop_relationship
 parent="string" [1]
 name="string" [1]
/>
```

### Schema Component Representation

```
<element name="drop_relationship">
   <complexType>
      <attribute name="parent" type="string" use="required"/>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: freeform

### Properties

| | |
|---|---|
| **Name** | freeform |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

A freeform statement is made of up SQL sections. Each SQL section can have one or more SQL statements, delimited with semicolons. A SQL section can be targeted for specific server types, or all servers.

Since the statement is executed directly on a database server, proceed with extreme caution. The difference in syntax must also be considered when writting the statements. If you cannot create a statement that is compatible with all supported servers, use a sql element for each database product.

## Attributes:

- **description:** A meaningful description about the changes been made by the SQL statements.

## Use Example:

The following example shows a sequence of sql elements performing updates on different objects:

```
<freeform description="Updates MaxObject and Maxattribute">

  <sql target="all">

    update maxobject set classname = 'psdi.skd.app.workorder.SKDWOSet' where objectna
me = 'WORKORDER';

    update maxobjectcfg set classname = 'psdi.skd.app.workorder.SKDWOSet' where objec
tname = 'WORKORDER';

  </sql>

  <sql target="all">

    update maxobject set classname = 'psdi.skd.app.workorder.SKDWOActivitySet' where 
objectname = 'WOACTIVITY';

    update maxobjectcfg set classname = 'psdi.skd.app.workorder.SKDWOActivitySet' whe
re objectname = 'WOACTIVITY';

  </sql>

  <sql target="all">

    update maxattribute set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' wher
e objectname = 'WORKORDER' and attributename = 'ISTASK';

    update maxattributecfg set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' w
here objectname = 'WORKORDER' and attributename = 'ISTASK';

  </sql>

  <sql target="all">

    update maxattribute set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' wher
e objectname = 'WOACTIVITY' and attributename = 'ISTASK';

    update maxattributecfg set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' w
here objectname = 'WOACTIVITY' and attributename = 'ISTASK';

  </sql>

</freeform>
```

The previous example puts one statement per sql element but they also can be grouped on one sigle sql element, as the next example shows:

```
<freeform description="Updates MaxObject and Maxattribute">

  <sql target="all">
```

```
    update maxobject set classname = 'psdi.skd.app.workorder.SKDWOSet' where objectna
me = 'WORKORDER';

    update maxobjectcfg set classname = 'psdi.skd.app.workorder.SKDWOSet' where objec
tname = 'WORKORDER';

    update maxobject set classname = 'psdi.skd.app.workorder.SKDWOActivitySet' where
objectname = 'WOACTIVITY';

    update maxobjectcfg set classname = 'psdi.skd.app.workorder.SKDWOActivitySet' whe
re objectname = 'WOACTIVITY';

    update maxattribute set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' wher
e objectname = 'WORKORDER' and attributename = 'ISTASK';

    update maxattributecfg set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' w
here objectname = 'WORKORDER' and attributename = 'ISTASK';

    update maxattribute set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' wher
e objectname = 'WOACTIVITY' and attributename = 'ISTASK';

    update maxattributecfg set  classname = 'psdi.skd.app.workorder.SKDFldWOIsTask' w
here objectname = 'WOACTIVITY' and attributename = 'ISTASK';

  </sql>

</freeform>
```

XML Instance Representation

```
<freeform

 description="string" [1]

>

   <sql> ... </sql> [1..*]

</freeform>
```

Schema Component Representation

```
<element name="freeform">

   <complexType>

      <sequence>

         <element ref="sql" maxOccurs="unbounded"/>

      </sequence>

      <attribute name="description" type="string" use="required"/>

   </complexType>

</element>
```

## Element: sql

Properties

| | |
|---|---|
| **Name** | sql |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Works in conjunction with the freeform element and encloses the SQL statement that is executed against the database. Since Maximo Asset Management supports different database products that has different SQL syntaxes, one SQL statement can run successfully on one platform but not on another. In this case, use the **target**attribute to define the scope of this statement.

## Attributes:

- **target:** Defines when the statements must be executed, based on the underlying database configuration. It's possible to have product-specific statements, such as `oracle` for Oracle statements, statements that run on all products except one, like `not_db2` and statements that run on all products (`all`).

## Use Example:

The following example shows different statements running against different databases. There are statements that runs in all platforms(`target="all"`), and statements that, due to differences in syntax, has versions to Oracle, SQL Server and DB2 databases:

```
<freeform description="add seed">

  <sql target="all">delete from maxintobjdetail where intobjectname in ('MXBUDGET') a
nd objectname in ('BUDGET');</sql>

  <sql target="all">delete from maxintobjdetail where intobjectname in ('MXBUDGET') a
nd objectname in ('BUDGETLINE');</sql>

  <sql target="all">delete from maxintobject where intobjectname in ('MXBUDGET') and
authapp in ('BUDGET');</sql>

  <sql target="all">delete from maxifacein where ifacename in ('MXBUDGETInterface') a
nd intobjectname in ('MXBUDGET');</sql>

  <sql target="all">delete from maxifaceout where ifacename in ('MXBUDGETInterface')
and intobjectname in ('MXBUDGET');</sql>



  <sql target="oracle">insert into maxintobjdetail( intobjectname, objectname, relati
on, objectorder, processorder, userdefined, changeby, changedate, maxintobjdetailid,
objectid, parentobjid, hierarchypath, description, altkey, excludeparentkey, deleteon
create, propagateevent) values ('MXBUDGET', 'BUDGET', null, 1, 1, 0, 'MAXIMO', to_dat
```

```
e('2012-03-01 23:23:23','yyyy-mm-dd hh24:mi:ss'), maxintobjdetailseq.nextval, 1, null
, 'BUDGET', null, null, 0, 0, 0);</sql>

  <sql target="oracle">insert into maxintobjdetail( intobjectname, objectname, relati
on, objectorder, processorder, userdefined, changeby, changedate, maxintobjdetailid,
objectid, parentobjid, hierarchypath, description, altkey, excludeparentkey, deleteon
create, propagateevent) values ('MXBUDGET', 'BUDGETLINE', 'BUDGETLINES', 1, 2, 0, 'MA
XIMO', to_date('2012-03-01 23:23:30','yyyy-mm-dd hh24:mi:ss'), maxintobjdetailseq.nex
tval, 2, 1, 'BUDGET/BUDGETLINE', null, null, 1, 1, 1);</sql>

  <sql target="oracle">insert into maxintobject( intobjectname, description, selfrefe
rencing, userdefined, changeby, changedate, maxintobjectid, langcode, hasld, defclass
, procclass, queryonly, configurable, flatsupported, usewith, aliasconflict, app, aut
happ) values ('MXBUDGET', 'Budget definition for excel export/import', 0, 0, 'MAXIMO'
, to_date('2012-03-01 23:26:26','yyyy-mm-dd hh24:mi:ss'), maxintobjectseq.nextval, 'E
N', 0, null, 'psdi.iface.mic.StatefulMicSetIn', 0, 1, 0, 'INTEGRATION', 0, null, 'BUD
GET');</sql>

  <sql target="sqlserver">insert into maxintobjdetail( intobjectname, objectname, rel
ation, objectorder, processorder, userdefined, changeby, changedate, maxintobjdetaili
d, objectid, parentobjid, hierarchypath, description, altkey, excludeparentkey, delet
eoncreate, propagateevent) values ('MXBUDGET', 'BUDGET', null, 1, 1, 0, 'MAXIMO', '20
120301 23:23:23', maxintobjdetailseq.nextval, 1, null, 'BUDGET', null, null, 0, 0, 0)
;</sql>

  <sql target="sqlserver">insert into maxintobjdetail( intobjectname, objectname, rel
ation, objectorder, processorder, userdefined, changeby, changedate, maxintobjdetaili
d, objectid, parentobjid, hierarchypath, description, altkey, excludeparentkey, delet
eoncreate, propagateevent) values ('MXBUDGET', 'BUDGETLINE', 'BUDGETLINES', 1, 2, 0,
'MAXIMO', '20120301 23:23:30', maxintobjdetailseq.nextval, 2, 1, 'BUDGET/BUDGETLINE',
null, null, 1, 1, 1);</sql>

  <sql target="sqlserver">insert into maxintobject( intobjectname, description, selfr
eferencing, userdefined, changeby, changedate, maxintobjectid, langcode, hasld, defcl
ass, procclass, queryonly, configurable, flatsupported, usewith, aliasconflict, app,
authapp) values ('MXBUDGET', 'Budget definition for excel export/import', 0, 0, 'MAXI
MO', '20120301 23:26:26', maxintobjectseq.nextval, 'EN', 0, null, 'psdi.iface.mic.Sta
tefulMicSetIn', 0, 1, 0, 'INTEGRATION', 0, null, 'BUDGET');</sql>

  <sql target="db2">insert into maxintobjdetail( intobjectname, objectname, relation,
objectorder, processorder, userdefined, changeby, changedate, maxintobjdetailid, obje
ctid, parentobjid, hierarchypath, description, altkey, excludeparentkey, deleteoncrea
te, propagateevent) values ('MXBUDGET', 'BUDGET', null, 1, 1, 0, 'MAXIMO', '2012-03-0
1 23:23:23', nextval for maxintobjdetailseq, 1, null, 'BUDGET', null, null, 0, 0, 0);
</sql>

  <sql target="db2">insert into maxintobjdetail( intobjectname, objectname, relation,
objectorder, processorder, userdefined, changeby, changedate, maxintobjdetailid, obje
ctid, parentobjid, hierarchypath, description, altkey, excludeparentkey, deleteoncrea
te, propagateevent) values ('MXBUDGET', 'BUDGETLINE', 'BUDGETLINES', 1, 2, 0, 'MAXIMO
', '2012-03-01 23:23:30', nextval for maxintobjdetailseq, 2, 1, 'BUDGET/BUDGETLINE',
null, null, 1, 1, 1);</sql>

  <sql target="db2">insert into maxintobject( intobjectname, description, selfreferen
cing, userdefined, changeby, changedate, maxintobjectid, langcode, hasld, defclass, p
rocclass, queryonly, configurable, flatsupported, usewith, aliasconflict, app, authap
p) values ('MXBUDGET', 'Budget definition for excel export/import', 0, 0, 'MAXIMO', '
```

```
2012-03-01 23:26:26', nextval for maxintobjectseq, 'EN', 0, null, 'psdi.iface.mic.Sta
tefulMicSetIn', 0, 1, 0, 'INTEGRATION', 0, null, 'BUDGET');</sql>

  <sql target="oracle">insert into maxifacein ( changeby, changedate, userdefined, de
scription, ifacename, ifacetype, ifaceexitclass, ifaceuserexitclass, ifacemapname, in
tobjectname, ifacecontrol, maxifaceinid, ifacetbname, interpreterclass, replyexitclas
s, replyuserexitclass, replymapname, elementname, replyschemaloc, replyelementname, r
eplyrequired, langcode, messagetype, schemalocation, hasld, splittag, usetxntracking,
useexternalschema, storemsg, extmsgidfield, searchfield)values ('MAXIMO', to_date('20
12-03-01 23:23:30','yyyy-mm-dd hh24:mi:ss'), 0, 'BUDGET', 'MXBUDGETInterface', 'MAXIM
O', null, null, null, 'MXBUDGET', null, maxifaceinseq.nextval, null, null, null, null
, null, null, null, null, 0, 'EN', 'Sync', null, 0, null, 0, 0, 0, null, null);</sql>

  <sql target="oracle">insert into maxifaceout( changeby, changedate, userdefined, de
scription, ifacename, ifacetype, intobjectname, ifaceexitclass, ifaceuserexitclass, i
facemapname, maxifaceoutid, ifacetbname, eventfilterclass, retainmbos, messagetype, l
istener, hasld, langcode, storemsg, usetxntracking, extmsgidfield, searchfield) value
s('MAXIMO', to_date('2012-03-01 23:23:30','yyyy-mm-dd hh24:mi:ss'), 0, 'BUDGET', 'MXB
UDGETInterface', 'MAXIMO', 'MXBUDGET', null, null, null, maxifaceoutseq.nextval, null
, null, 1, 'Publish', 0, 0, 'EN', 0, 0, null, null);</sql>

  <sql target="sqlserver">insert into maxifacein ( changeby, changedate, userdefined,
description, ifacename, ifacetype, ifaceexitclass, ifaceuserexitclass, ifacemapname,
intobjectname, ifacecontrol, maxifaceinid, ifacetbname, interpreterclass, replyexitcl
ass, replyuserexitclass, replymapname, elementname, replyschemaloc, replyelementname,
replyrequired, langcode, messagetype, schemalocation, hasld, splittag, usetxntracking
, useexternalschema, storemsg, extmsgidfield, searchfield)values ('MAXIMO', '20120301
23:23:30', 0, 'BUDGET', 'MXBUDGETInterface', 'MAXIMO', null, null, null, 'MXBUDGET',
null, maxifaceinseq.nextval, null, null, null, null, null, null, null, null, 0, 'EN',
'Sync', null, 0, null, 0, 0, 0, null, null);</sql>

  <sql target="sqlserver">insert into maxifaceout( changeby, changedate, userdefined,
description, ifacename, ifacetype, intobjectname, ifaceexitclass, ifaceuserexitclass,
ifacemapname, maxifaceoutid, ifacetbname, eventfilterclass, retainmbos, messagetype,
listener, hasld, langcode, storemsg, usetxntracking, extmsgidfield, searchfield) valu
es('MAXIMO', '20120301 23:23:30', 0, 'BUDGET', 'MXBUDGETInterface', 'MAXIMO', 'MXBUDG
ET', null, null, null, maxifaceoutseq.nextval, null, null, 1, 'Publish', 0, 0, 'EN',
0, 0, null, null);</sql>

  <sql target="db2">insert into maxifacein ( changeby, changedate, userdefined, descr
iption, ifacename, ifacetype, ifaceexitclass, ifaceuserexitclass, ifacemapname, intob
jectname, ifacecontrol, maxifaceinid, ifacetbname, interpreterclass, replyexitclass,
replyuserexitclass, replymapname, elementname, replyschemaloc, replyelementname, repl
yrequired, langcode, messagetype, schemalocation, hasld, splittag, usetxntracking, us
eexternalschema, storemsg, extmsgidfield, searchfield)values ('MAXIMO', '2012-03-01 2
3:23:30', 0, 'BUDGET', 'MXBUDGETInterface', 'MAXIMO', null, null, null, 'MXBUDGET', n
ull, nextval for maxifaceinseq, null, null, null, null, null, null, null, null, 0, 'E
N', 'Sync', null, 0, null, 0, 0, 0, null, null);</sql>

  <sql target="db2">insert into maxifaceout( changeby, changedate, userdefined, descr
iption, ifacename, ifacetype, intobjectname, ifaceexitclass, ifaceuserexitclass, ifac
emapname, maxifaceoutid, ifacetbname, eventfilterclass, retainmbos, messagetype, list
ener, hasld, langcode, storemsg, usetxntracking, extmsgidfield, searchfield) values('
MAXIMO', '2012-03-01 23:23:30', 0, 'BUDGET', 'MXBUDGETInterface', 'MAXIMO', 'MXBUDGET
', null, null, null, nextval for maxifaceoutseq, null, null, 1, 'Publish', 0, 0, 'EN'
, 0, 0, null, null);</sql>
```

```
</freeform>
```

XML Instance Representation

```
<sql

 target="string (value comes from list: {'oracle'|'sqlserver'|'db2'|'all'|'not_oracle
'|'not_sqlserver'|'not_db2'})" [0..1]

/>
```

Schema Component Representation

```
<element name="sql">
   <complexType mixed="true">
      <attribute name="target" use="default" value="all">
         <simpleType>
            <restriction base="string">
               <enumeration value="oracle"/>
               <enumeration value="sqlserver"/>
               <enumeration value="db2"/>
               <enumeration value="all"/>
               <enumeration value="not_oracle"/>
               <enumeration value="not_sqlserver"/>
               <enumeration value="not_db2"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: define_table

### Properties

| | |
|---|---|
| **Name** | define_table |
| **Type** | Locally-defined complex type |

|  |  |
|---|---|
| **Nillable** | no |
| **Abstract** | no |

Documentation

Creates a database table.

The table is associated to a Maximo Business Object (MBO) that must already exists on Maximo Asset Management.

## Attributes:

- **object:** Table's name that is used everytime the table needs to be referred.
- **description:** A meaningful description about the view and why the view is necessary.
- **service:** The Maximo service that is associated with the MBO.
- **classname:** The fully qualified name of the MBOSet class.
- **persistent:** Defines if the MBO is persistent or not.
- **type:** Access level of the table.
- **primarykey:** A comma-separeted list of columns names that represents the table's primary key.
- **mainobject:** Defines if the MBO should be a main object or not.
- **internal:** Sets the internal field of MBO.
- **trigroot:** Each Maximo table includes a column named ROWSTAMP, for this columns, the trigeroot must be set in order to autaticaly fill it.
- **storagetype:** Defines the Storage Types for Multitenancy used by the DBChange code.
- **resource_type:** Only used for tables's representing the WEATHERAPI resources.

## Use Example:

The following example creates a table that is named `INSIGHTFACTOR`:

```
<define_table object="INSIGHTFACTOR" classname="psdi.app.insight.InsightFactorSet" service="CUSTAPP" type="system" description="Insight Factor" persistent="true" mainobject="false" internal="false" primarykey="INSIGHTNAME,FACTORNAME" trigroot="INSIGHTFACTOR">

  <attrdef attribute="DESCRIPTION" maxtype="ALN" length="50" scale="0" haslongdesc="true" title="Description" remarks="Description" />

  <attrdef attribute="FACTORNAME" maxtype="UPPER" length="50" scale="0" title="Factor Name" remarks="Name of the Insight Factor" />

  <attrdef attribute="INSIGHTNAME" sameasobject="INSIGHT" sameasattribute="NAME" title="Insight Name" remarks="Insight this factor is associated with" />

  <attrdef attribute="AGGRFUNCTION" maxtype="ALN" length="50" scale="0" title="Aggregate Function" remarks="Aggregate Function" />
  <attrdef attribute="AGGRATTRIBUTE" sameasobject="MAXATTRIBUTE" sameasattribute="ATT
RIBUTENAME" title="Aggregate Attribute" remarks="Aggregate Attribute" />

  <attrdef attribute="FILTER" maxtype="ALN" length="200" scale="0" title="Filter" rem
arks="Filter" />

</define_table>
```

## XML Instance Representation

```
<define_table

 object="string" [1]

 description="string" [1]

 service="string" [1]

 classname="string" [1]

 persistent="string (value comes from list: {'true'|'false'})" [0..1]

 type="string (value comes from list: {'system'|'site'|'organization'|'orgwithsite'|'
companyset'|'itemset'|'org'|'orgappfilter'|'orgsite'|'siteappfilter'|'systemappfilter
'|'systemorg'|'systemorgsite'|'systemsite'})" [1]

 primarykey="string" [0..1]

 mainobject="string (value comes from list: {'true'|'false'})" [0..1]

 internal="string (value comes from list: {'true'|'false'})" [0..1]

 trigroot="string" [0..1]

 storagetype="string (value comes from list: {'tenant'|'master'|'system'|'template'|'
system_resource'|'master_with_setup'|'template_with_setup'|'tenant_monitor'})" [0..1]

 resource_type="string" [0..1]

>

   <attrdef> ... </attrdef> [1..*]

   <longdescription> ... </longdescription> [0..1]

</define_table>
```

## Schema Component Representation

```
<element name="define_table">

   <complexType>

      <sequence>

         <element ref="attrdef" maxOccurs="unbounded"/>

         <element ref="longdescription" minOccurs="0" maxOccurs="1"/>

      </sequence>

      <attribute name="object" type="string" use="required"/>

      <attribute name="description" type="string" use="required"/>
```

```
    <attribute name="service" type="string" use="required"/>
    <attribute name="classname" type="string" use="required"/>
    <attribute name="persistent" use="default" value="true">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="type" use="required">
        <simpleType>
            <restriction base="string">
                <enumeration value="system"/>
                <enumeration value="site"/>
                <enumeration value="organization"/>
                <enumeration value="orgwithsite"/>
                <enumeration value="companyset"/>
                <enumeration value="itemset"/>
                <enumeration value="org"/>
                <enumeration value="orgappfilter"/>
                <enumeration value="orgsite"/>
                <enumeration value="siteappfilter"/>
                <enumeration value="systemappfilter"/>
                <enumeration value="systemorg"/>
                <enumeration value="systemorgsite"/>
                <enumeration value="systemsite"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="primarykey" type="string" use="optional"/>
    <attribute name="mainobject" use="default" value="false">
        <simpleType>
            <restriction base="string">
```

```
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="internal" use="default" value="false">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="trigroot" type="string" use="optional"/>
    <attribute name="storagetype" use="default" value="tenant">
        <simpleType>
            <restriction base="string">
                <enumeration value="tenant"/>
                <enumeration value="master"/>
                <enumeration value="system"/>
                <enumeration value="template"/>
                <enumeration value="system_resource"/>
                <enumeration value="master_with_setup"/>
                <enumeration value="template_with_setup"/>
                <enumeration value="tenant_monitor"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="resource_type" type="string" use="optional"/>
  </complexType>
</element>
```

## Element: modify_table

### Properties

| | |
|---|---|
| **Name** | modify_table |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Above, in the list for type, organization and orgwithsite are deprecated and are kept only so existing scripts can be used. In the statement run method, organization will be converted to org and orgwithsite converted to orgsite.

```
The list of storagetype comes from the MTStorageType class. That list must correspond
to this list.
```

### Attributes:

- **name:** Table's name.
- **object:** New table's name .
- **description:** Brief description that should replace the original one.
- **service:** The Maximo service associated with the MBO.
- **classname:** The fully qualified name of the MBOSet class.
- **type:** Access level of the table.

### Use Example:

```
<modify_table name="TABLENAME" object="OBJECTNAME" >
```

<longdescription>Some comments goes here</longdescription> </modify_table>

### XML Instance Representation

```
<modify_table
 name="string" [1]
 object="string" [0..1]
 description="string" [0..1]
 service="string" [0..1]
 classname="string" [0..1]
 type="string (value comes from list: {'system'|'site'|'organization'|'orgwithsite'|'
companyset'|'itemset'|'org'|'orgappfilter'|'orgsite'|'siteappfilter'|'systemappfilter
'|'systemorg'|'systemorgsite'|'systemsite'})" [0..1]

 primarykey="string" [0..1]

 mainobject="string (value comes from list: {'true'|'false'})" [0..1]

 internal="string (value comes from list: {'true'|'false'})" [0..1]

 trigroot="string" [0..1]

 unique_column="string" [0..1]

 storagetype="string (value comes from list: {'tenant'|'master'|'system'|'template'|'
system_resource'|'master_with_setup'|'template_with_setup'|'tenant_monitor'})" [0..1]

>

   Start Sequence [0..1]

      <longdescription> ... </longdescription> [1]

   End Sequence

</modify_table>
```

Schema Component Representation

```
<element name="modify_table">

   <complexType>

      <sequence minOccurs="0" maxOccurs="1">

         <element ref="longdescription"/>

      </sequence>

      <attribute name="name" type="string" use="required"/>

      <attribute name="object" type="string" use="optional"/>

      <attribute name="description" type="string" use="optional"/>

      <attribute name="service" type="string" use="optional"/>

      <attribute name="classname" type="string" use="optional"/>

      <attribute name="type" use="optional">

         <simpleType>

            <restriction base="string">

               <enumeration value="system"/>

               <enumeration value="site"/>

               <enumeration value="organization"/>

               <enumeration value="orgwithsite"/>

               <enumeration value="companyset"/>

               <enumeration value="itemset"/>
```

```
                <enumeration value="org"/>
                <enumeration value="orgappfilter"/>
                <enumeration value="orgsite"/>
                <enumeration value="siteappfilter"/>
                <enumeration value="systemappfilter"/>
                <enumeration value="systemorg"/>
                <enumeration value="systemorgsite"/>
                <enumeration value="systemsite"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="primarykey" type="string" use="optional"/>
    <attribute name="mainobject" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="internal" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="trigroot" type="string" use="optional"/>
    <attribute name="unique_column" type="string" use="optional"/>
    <attribute name="storagetype" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="tenant"/>
                <enumeration value="master"/>
                <enumeration value="system"/>
                <enumeration value="template"/>
                <enumeration value="system_resource"/>
                <enumeration value="master_with_setup"/>
                <enumeration value="template_with_setup"/>
                <enumeration value="tenant_monitor"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: drop_table

Properties

| | |
|---|---|
| **Name** | drop_table |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop an object/table (created previously by a define_table statement)

### Attributes:

- **object:** Table's name, to be used everytime the table needs to be referred.

## Use Example:

```
<drop_table object="INSIGHTFACTOR"  type="system"  </drop_table>
```

XML Instance Representation

```
<drop_table
```

```
 object="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_table">
   <complexType>
      <attribute name="object" type="string" use="required"/>
   </complexType>
</element>
```

## Element: add_attributes

Properties

| | |
|---|---|
| **Name** | add_attributes |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Adds new attributes to an existing object/table (created previously by a define_table statement).

### Attributes:

- Check attrmod fot a further information about nested attributes.

### Use Example:

```
<add_attributes object="LOCATIONS">
  <attrdef attribute="MODELID" maxtype="ALN" title="Import ID"
    remarks="Import ID Typically from a BIM model or COBie data set" length="45" loca
lizable="false" />
  <attrdef attribute="BIMIMPORTSRC" maxtype="ALN" title="Import source" domain="COBIE
SHEETTYPE"
```

```
    remarks="The external source for the data in this record such as a COBie sheet or
an IFC class name" length="25" />

  <attrdef attribute="BIMUSAGE" maxtype="ALN" title="BIM Location type" domain="BIMLO
CATIONUSE"

    remarks="Describes how a location is used and controls COBie export behavior" len
gth="20" />

  <attrdef attribute="BIMROOMNAME" maxtype="ALN" title="Room Name/Number"

    remarks="The name or number of the room or space" length="40" />


</add_attributes>
```

## XML Instance Representation

```
<add_attributes
 object="string" [1]
>
   <attrdef> ... </attrdef> [1..*]
</add_attributes>
```

## Schema Component Representation

```
<element name="add_attributes">
   <complexType>
      <sequence>
         <element ref="attrdef" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="object" type="string" use="required"/>
   </complexType>
</element>
```

## Element: attrmod

### Properties

| | |
|---|---|
| **Name** | attrmod |
| **Type** | Locally-defined complex type |

| | |
|---|---|
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines each attribute should be modified/added and should be configured as a nested element into the add_attribute.

## Attributes:

- **attribute:** Attrinute's name.
- **maxtype:** Maximo data type
- **length:** The number of characters that can be entered into this attribute.
- **persistent:** Defines if the attribute is persistent or not.
- **haslongdesc:** If the attribute has a log description, this should be set.
- **required:** When set to **TRUE**, indicates that a value is required in this field.
- **userdefined:** Inform to Maximo that this attribute was defined/added by user.
- **domain:** Associate this attribute with a domain through the domain's name
- **classname:** Associate to this attribute with a field validation class trhough the class's FQN (Full Qualified Name).
- **defaultvalue:** Defines a default value for the attribute.
- **title:** Define a clear, short column identifier to be substituted in messages, screen labels, etc.
- **remarks:** A brief description or a remark about the attribute's purpose.
- **sameasobject:** SameAsObject indicate the master object that controls the maxtype, length, and scale of this attribute.
- **sameasattribute:** SameAsAttribute indicate the master attribute that controls the maxtype, length, and scale of this attribute.
- **mustbe:** When checked, indicates that Maxtype, Length, and Scale of the attribute can NOT be changed. When cleared, indicates that they can be changed.
- **ispositive:** For a numeric field, when checked, only positive values (and zero) are allowed; when cleared, both positive and negative values are allowed.
- **autokey:** Defines the autokey for this field.
- **canautonum:** Defines if this attribute can use a trigger.
- **searchtype:** Indicates to Maximo what kind of search type is allowed for this attribute.
- **localizable:** Defines if this attribute is localized or not.
- **domainlink:** Defines if the value should be norrow down by another domain.
- **restriction:** Type of restriction being configured. Valid types are Hidden, Required, Read-only and Qualified. Qualified means that only data meeting the condition is 'qualified' to be

fetched from the database and can only be applied to top-level objects (not child objects or attributes).

## Use Example:

The follow example add attributes `EXTLOCID`, `EXTSITENAME`, `EXTSITEUSE` and `EXTROOMNAME` against the `LOCATIONS` object/table.

```
<add_attributes object="LOCATIONS">

    <attrdef attribute="EXTLOCID" maxtype="ALN" title="Import ID"

      remarks="External Location ID" length="45" localizable="false" />

    <attrdef attribute="EXTSITENAME" maxtype="ALN" title="Import source" domain="COBI
ESHEETTYPE"

      remarks="External site name" length="25" />

    <attrdef attribute="EXTSITEUSE" maxtype="ALN" title="BIM Location type" domain="B
IMLOCATIONUSE"

      remarks="External site use purpose" length="20" />

    <attrdef attribute="EXTROOMNAME" maxtype="ALN" title="Room Name/Number"

      remarks="Exteranal room name" length="40" />

</add_attributes>
```

XML Instance Representation

```
<attrmod

 attribute="string" [1]

 maxtype="string (value comes from list: {'ALN'|'AMOUNT'|'BIGINT'|'BLOB'|'CLOB'|'CRYP
TO'|'CRYPTOX'|'DATE'|'DATETIME'|'DECIMAL'|'DURATION'|'FLOAT'|'GL'|'INTEGER'|'LONGALN'
|'LOWER'|'SMALLINT'|'TIME'|'UPPER'|'YORN'})" [0..1]

 length="NMTOKEN" [0..1]

 persistent="string (value comes from list: {'true'|'false'})" [0..1]

 haslongdesc="string (value comes from list: {'true'|'false'})" [0..1]

 required="string (value comes from list: {'true'|'false'})" [0..1]

 userdefined="string (value comes from list: {'true'|'false'})" [0..1]

 domain="string" [0..1]

 classname="string" [0..1]

 defaultvalue="string" [0..1]

 title="string" [0..1]

 remarks="string" [0..1]

 sameasobject="string" [0..1]
 sameasattribute="string" [0..1]
 mustbe="string (value comes from list: {'true'|'false'})" [0..1]
 ispositive="string (value comes from list: {'true'|'false'})" [0..1]
 scale="NMTOKEN" [0..1]
 autokey="string" [0..1]
 canautonum="string (value comes from list: {'true'|'false'})" [0..1]
 searchtype="string (value comes from list: {'WILDCARD'|'EXACT'|'NONE'|'TEXT'})" [0..
1]
 localizable="string (value comes from list: {'true'|'false'})" [0..1]
 domainlink="string" [0..1]
 restricted="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

Schema Component Representation

```
<element name="attrmod">
   <complexType>
      <attribute name="attribute" type="string" use="required"/>
      <attribute name="maxtype" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
               <enumeration value="AMOUNT"/>
               <enumeration value="BIGINT"/>
               <enumeration value="BLOB"/>
               <enumeration value="CLOB"/>
               <enumeration value="CRYPTO"/>
               <enumeration value="CRYPTOX"/>
               <enumeration value="DATE"/>
               <enumeration value="DATETIME"/>
               <enumeration value="DECIMAL"/>
               <enumeration value="DURATION"/>
               <enumeration value="FLOAT"/>
               <enumeration value="GL"/>
               <enumeration value="INTEGER"/>
               <enumeration value="LONGALN"/>
               <enumeration value="LOWER"/>
               <enumeration value="SMALLINT"/>
               <enumeration value="TIME"/>
               <enumeration value="UPPER"/>
               <enumeration value="YORN"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="length" type="NMTOKEN" use="optional"/>
      <attribute name="persistent" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="haslongdesc" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="required" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
```

```
<attribute name="userdefined" use="optional">
   <simpleType>
      <restriction base="string">
         <enumeration value="true"/>
         <enumeration value="false"/>
      </restriction>
   </simpleType>
</attribute>
<attribute name="domain" type="string" use="optional"/>
<attribute name="classname" type="string" use="optional"/>
<attribute name="defaultvalue" type="string" use="optional"/>
<attribute name="title" type="string" use="optional"/>
<attribute name="remarks" type="string" use="optional"/>
<attribute name="sameasobject" type="string" use="optional"/>
<attribute name="sameasattribute" type="string" use="optional"/>
<attribute name="mustbe" use="optional">
   <simpleType>
      <restriction base="string">
         <enumeration value="true"/>
         <enumeration value="false"/>
      </restriction>
   </simpleType>
</attribute>
<attribute name="ispositive" use="optional">
   <simpleType>
      <restriction base="string">
         <enumeration value="true"/>
         <enumeration value="false"/>
      </restriction>
   </simpleType>
</attribute>
<attribute name="scale" type="NMTOKEN" use="optional"/>
<attribute name="autokey" type="string" use="optional"/>
<attribute name="canautonum" use="optional">
```

```
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="searchtype" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="WILDCARD"/>
            <enumeration value="EXACT"/>
            <enumeration value="NONE"/>
            <enumeration value="TEXT"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="localizable" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="domainlink" type="string" use="optional"/>
   <attribute name="restricted" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   </complexType>
</element>
```

## Element: modify_attribute

Properties

| | |
|---|---|
| **Name** | modify_attribute |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies an attribute from an existing object/table (created previously by a define_table; add_attributes statements

### Attributes:

- **attribute:** Attrinute's name.
- **maxtype:** Maximo data type.
- **length:** The number of characters that can be entered into this attribute.
- **persistent:** Defines if the attribute is persistent or not.
- **haslongdesc:** If the attribute has a log description, this should be set.
- **required:** When set to **TRUE**, indicates that a value is required in this field.
- **userdefined:** Inform to Maximo that this attribute was defined/added by user.
- **domain:** Associate this attribute with a domain through the domain's name.
- **classname:** Associate to this attribute with a field validation class trhough the class's FQN (Full Qualified Name).
- **defaultvalue:** Defines a default value for the attribute.
- **title:** Define a clear, short column identifier to be substituted in messages, screen labels, etc.
- **remarks:** A brief description or a remark about the attribute's purpose.
- **sameasobject:** SameAsObject indicate the master object that controls the maxtype, length, and scale of this attribute.
- **sameasattribute:** SameAsAttribute indicate the master attribute that controls the maxtype, length, and scale of this attribute.

- **mustbe:** When checked, indicates that Maxtype, Length, and Scale of the attribute can NOT be changed. When cleared, indicates that they can be changed.
- **ispositive:** For a numeric field, when checked, only positive values (and zero) are allowed; when cleared, both positive and negative values are allowed.
- **autokey:** Defines the autokey for this field.
- **canautonum:** Defines if this attribute can use a trigger.
- **searchtype:** Indicates to Maximo what kind of search type is allowed for this attribute.
- **localizable:** Defines if this attribute is localized or not.
- **domainlink:** Defines if the value should be norrow down by another domain.
- **restriction:** Type of restriction being configured. Valid types are Hidden, Required, Read-only and Qualified. Qualified means that only data meeting the condition is 'qualified' to be fetched from the database and can only be applied to top-level objects (not child objects or attributes).

## Use Example:

The follow example change attributes `EXTLOCID`, `EXTSITENAME`, `EXTSITEUSE` and `EXTROOMNAME` from `LOCATIONS` object/table. The attributes were add in add_attributeselement.

```
<modify_attribute object="LOCATIONS">

    <attrdef attribute="EXTLOCID" maxtype="ALN" title="Import ID"

      remarks="External Location ID" length="45" localizable="false" />

    <attrdef attribute="EXTSITENAME" maxtype="ALN" title="Import source" domain="COBI
ESHEETTYPE"

      remarks="External site name" length="25" />

    <attrdef attribute="EXTSITEUSE" maxtype="ALN" title="BIM Location type" domain="B
IMLOCATIONUSE"

      remarks="External site use purpose" length="20" />

    <attrdef attribute="EXTROOMNAME" maxtype="ALN" title="Room Name/Number"

      remarks="Exteranal room name" length="40" />

</modify_attribute>
```

### XML Instance Representation

```
<modify_attribute

 object="string" [1]

 attribute="string" [1]

 maxtype="string (value comes from list: {'ALN'|'AMOUNT'|'BIGINT'|'BLOB'|'CLOB'|'CRYP
TO'|'CRYPTOX'|'DATE'|'DATETIME'|'DECIMAL'|'DURATION'|'FLOAT'|'GL'|'INTEGER'|'LONGALN'
|'LOWER'|'SMALLINT'|'TIME'|'UPPER'|'YORN'})" [0..1]
```

```
 length="NMTOKEN" [0..1]
 persistent="string (value comes from list: {'true'|'false'})" [0..1]
 haslongdesc="string (value comes from list: {'true'|'false'})" [0..1]
 required="string (value comes from list: {'true'|'false'})" [0..1]
 userdefined="string (value comes from list: {'true'|'false'})" [0..1]
 domain="string" [0..1]
 classname="string" [0..1]
 defaultvalue="string" [0..1]
 title="string" [0..1]
 remarks="string" [0..1]
 sameasobject="string" [0..1]
 sameasattribute="string" [0..1]
 mustbe="string (value comes from list: {'true'|'false'})" [0..1]
 ispositive="string (value comes from list: {'true'|'false'})" [0..1]
 scale="NMTOKEN" [0..1]
 autokey="string" [0..1]
 canautonum="string (value comes from list: {'true'|'false'})" [0..1]
 searchtype="string (value comes from list: {'WILDCARD'|'EXACT'|'NONE'|'TEXT'})" [0..
1]
 localizable="string (value comes from list: {'true'|'false'})" [0..1]
 domainlink="string" [0..1]
 restricted="string (value comes from list: {'true'|'false'})" [0..1]
 excludetenants="string" [0..1]
/>
```

Schema Component Representation

```
<element name="modify_attribute">
   <complexType>
      <attribute name="object" type="string" use="required"/>
      <attribute name="attribute" type="string" use="required"/>
      <attribute name="maxtype" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
```

```
                <enumeration value="AMOUNT"/>
                <enumeration value="BIGINT"/>
                <enumeration value="BLOB"/>
                <enumeration value="CLOB"/>
                <enumeration value="CRYPTO"/>
                <enumeration value="CRYPTOX"/>
                <enumeration value="DATE"/>
                <enumeration value="DATETIME"/>
                <enumeration value="DECIMAL"/>
                <enumeration value="DURATION"/>
                <enumeration value="FLOAT"/>
                <enumeration value="GL"/>
                <enumeration value="INTEGER"/>
                <enumeration value="LONGALN"/>
                <enumeration value="LOWER"/>
                <enumeration value="SMALLINT"/>
                <enumeration value="TIME"/>
                <enumeration value="UPPER"/>
                <enumeration value="YORN"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="length" type="NMTOKEN" use="optional"/>
    <attribute name="persistent" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="haslongdesc" use="optional">
        <simpleType>
            <restriction base="string">
```

```
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="required" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="userdefined" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="domain" type="string" use="optional"/>
    <attribute name="classname" type="string" use="optional"/>
    <attribute name="defaultvalue" type="string" use="optional"/>
    <attribute name="title" type="string" use="optional"/>
    <attribute name="remarks" type="string" use="optional"/>
    <attribute name="sameasobject" type="string" use="optional"/>
    <attribute name="sameasattribute" type="string" use="optional"/>
    <attribute name="mustbe" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
```

```
      </simpleType>
   </attribute>
   <attribute name="ispositive" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="scale" type="NMTOKEN" use="optional"/>
   <attribute name="autokey" type="string" use="optional"/>
   <attribute name="canautonum" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="searchtype" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="WILDCARD"/>
            <enumeration value="EXACT"/>
            <enumeration value="NONE"/>
            <enumeration value="TEXT"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="localizable" use="optional">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
```

```
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="domainlink" type="string" use="optional"/>
       <attribute name="restricted" use="optional">
          <simpleType>
             <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="excludetenants" type="string" use="optional"/>
    </complexType>
</element>
```

## Element: drop_attributes

Properties

| | |
|---|---|
| **Name** | drop_attributes |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop an attribute according with its name.

### Attributes:

- **objecname:** Object's name that owns this attribute.
- **attrname:** Attribute's name. For a further information about the attribute, refers to attrname element.

## Use Example:

Following a simple sample how to drop an attribute from a Table/Object.

```
<drop_attributes object="OBJECTNAME">
    <attrname name="ATTRIBUTENAME"/>
</drop_attributes>
```

### XML Instance Representation

```
<drop_attributes
 object="string" [1]
>
   <attrname> ... </attrname> [1..*]
</drop_attributes>
```

### Schema Component Representation

```
<element name="drop_attributes">
   <complexType>
      <sequence>
         <element ref="attrname" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="object" type="string" use="required"/>
   </complexType>
</element>
```

## Element: attrname

### Properties

| | |
|---|---|
| **Name** | attrname |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Used nested to the drop_attribute element.

### Attributes:

- **attrname:** Attribute's name. For a further information about the attribute, refers to attrname element.

## Use Example:

To a complete example about this element, please refer to drop_attribute element's Use examples section.

XML Instance Representation

```
<attrname
 name="string" [1]
/>
```

Schema Component Representation

```
<element name="attrname">
   <complexType>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: attrdef

Properties

| | |
|---|---|
| **Name** | attrdef |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

The **attrdef** element is nested to add_attributes element and serves to specify each attribute that should be added in a particular table, view or object.

## Attributes:

- **attribute:** Attrinute's name.
- **maxtype:** Maximo data type
- **length:** The number of characters that can be entered into this attribute.
- **persistent:** Defines if the attribute is persistent or not.
- **haslongdesc:** If the attribute has a log description, this should be set.
- **required:** When set to **TRUE**, indicates that a value is required in this field.
- **userdefined:** Inform to Maximo that this attribute was defined/added by user.
- **domain:** Associate this attribute with a domain through the domain's name
- **classname:** Associate to this attribute with a field validation class trhough the class's FQN (Full Qualified Name).
- **defaultvalue:** Defines a default value for the attribute.
- **title:** Define a clear, short column identifier to be substituted in messages, screen labels, etc.
- **remarks:** A brief description or a remark about the attribute's purpose.
- **sameasobject:** SameAsObject indicate the master object that controls the maxtype, length, and scale of this attribute.
- **sameasattribute:** SameAsAttribute indicate the master attribute that controls the maxtype, length, and scale of this attribute.
- **mustbe:** When checked, indicates that Maxtype, Length, and Scale of the attribute can NOT be changed. When cleared, indicates that they can be changed.
- **ispositive:** For a numeric field, when checked, only positive values (and zero) are allowed; when cleared, both positive and negative values are allowed.
- **autokey:** Defines the autokey for this field.
- **canautonum:** Defines if this attribute can use a trigger.
- **searchtype:** Indicates to Maximo what kind of search type is allowed for this attribute.
- **localizable:** Defines if this attribute is localized or not.
- **domainlink:** Defines if the value should be norrow down by another domain.
- **restriction:** Type of restriction being configured. Valid types are Hidden, Required, Read-only and Qualified. Qualified means that only data meeting the condition is 'qualified' to be fetched from the database and can only be applied to top-level objects (not child objects or attributes).

## Use Example:

For a further information about complete samples, please refer
to add_attributes and modify_attribute element's Use example's section.

XML Instance Representation

```
<attrdef
 attribute="string" [1]
 maxtype="string (value comes from list: {'ALN'|'AMOUNT'|'BIGINT'|'BLOB'|'CLOB'|'CRYP
TO'|'CRYPTOX'|'DATE'|'DATETIME'|'DECIMAL'|'DURATION'|'FLOAT'|'GL'|'INTEGER'|'LONGALN'
|'LOWER'|'SMALLINT'|'TIME'|'UPPER'|'YORN'})" [0..1]
 length="NMTOKEN" [0..1]
 persistent="string (value comes from list: {'true'|'false'})" [0..1]
 haslongdesc="string (value comes from list: {'true'|'false'})" [0..1]
 required="string (value comes from list: {'true'|'false'})" [0..1]
 userdefined="string (value comes from list: {'true'|'false'})" [0..1]
 domain="string" [0..1]
 classname="string" [0..1]
 defaultvalue="string" [0..1]
 title="string" [1]
 remarks="string" [1]
 sameasobject="string" [0..1]
 sameasattribute="string" [0..1]
 mustbe="string (value comes from list: {'true'|'false'})" [0..1]
 ispositive="string (value comes from list: {'true'|'false'})" [0..1]
 scale="NMTOKEN" [0..1]
 autokey="string" [0..1]
 canautonum="string (value comes from list: {'true'|'false'})" [0..1]
 searchtype="string (value comes from list: {'WILDCARD'|'EXACT'|'NONE'|'TEXT'})" [0..
1]
 localizable="string (value comes from list: {'true'|'false'})" [0..1]
 domainlink="string" [0..1]
 restricted="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

Schema Component Representation

```
<element name="attrdef">
   <complexType>
      <attribute name="attribute" type="string" use="required"/>
      <attribute name="maxtype" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
               <enumeration value="AMOUNT"/>
               <enumeration value="BIGINT"/>
               <enumeration value="BLOB"/>
               <enumeration value="CLOB"/>
               <enumeration value="CRYPTO"/>
               <enumeration value="CRYPTOX"/>
               <enumeration value="DATE"/>
               <enumeration value="DATETIME"/>
               <enumeration value="DECIMAL"/>
               <enumeration value="DURATION"/>
               <enumeration value="FLOAT"/>
               <enumeration value="GL"/>
               <enumeration value="INTEGER"/>
               <enumeration value="LONGALN"/>
               <enumeration value="LOWER"/>
               <enumeration value="SMALLINT"/>
               <enumeration value="TIME"/>
               <enumeration value="UPPER"/>
               <enumeration value="YORN"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="length" type="NMTOKEN" use="optional"/>
      <attribute name="persistent" use="default" value="true">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
```

```
        <attribute name="haslongdesc" use="default" value="false">
           <simpleType>
              <restriction base="string">
                 <enumeration value="true"/>
                 <enumeration value="false"/>
              </restriction>
           </simpleType>
        </attribute>
        <attribute name="required" use="default" value="false">
           <simpleType>
              <restriction base="string">
                 <enumeration value="true"/>
                 <enumeration value="false"/>
              </restriction>
           </simpleType>
        </attribute>
        <attribute name="userdefined" use="default" value="false">
           <simpleType>
              <restriction base="string">
                 <enumeration value="true"/>
                 <enumeration value="false"/>
              </restriction>
           </simpleType>
        </attribute>
        <attribute name="domain" type="string" use="optional"/>
        <attribute name="classname" type="string" use="optional"/>
        <attribute name="defaultvalue" type="string" use="optional"/>
        <attribute name="title" type="string" use="required"/>
        <attribute name="remarks" type="string" use="required"/>
        <attribute name="sameasobject" type="string" use="optional"/>
        <attribute name="sameasattribute" type="string" use="optional"/>
        <attribute name="mustbe" use="default" value="false">
           <simpleType>
              <restriction base="string">
```

```
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="ispositive" use="default" value="false">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="scale" type="NMTOKEN" use="optional"/>
    <attribute name="autokey" type="string" use="optional"/>
    <attribute name="canautonum" use="default" value="false">
        <simpleType>
            <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="searchtype" use="optional">
        <simpleType>
            <restriction base="string">
                <enumeration value="WILDCARD"/>
                <enumeration value="EXACT"/>
                <enumeration value="NONE"/>
                <enumeration value="TEXT"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="localizable" use="optional">
```

```
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="domainlink" type="string" use="optional"/>
      <attribute name="restricted" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: specify_index

Properties

| | |
|---|---|
| **Name** | specify_index |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Use *specify* to modify an existing index or bring new a one into existence

### Attributes:

- **name:** Index name.

- **object:** Object's or Table's name wich should have the index specified.
- **primary:** Indicates if this index contains or is a primary key.
- **unique:** *TRUE* or **FALSE** values indicating that this is a unique index.
- **clustered:** Indicates if this index is clustered or not
- **required:** Indicates if the index field is required or not.
- **addtenantid:** Associate this index with the table's tenant id.

## Use Example:

The follow example demostrate the **specify_index** element. For a further information about the nested attribute "indexkey", please refer to indexkey documentation.

```
<specify_index object="OBJECTNAME" name="INDEX_NDX1" primary="false" unique="false" c
lustered="false" required="false" >

    <indexkey column="ATTRIBUTE" ascending="true" />

</specify_index>
```

### XML Instance Representation

```
<specify_index

 name="string" [0..1]

 object="string" [1]

 primary="string (value comes from list: {'true'|'false'})" [0..1]

 unique="string (value comes from list: {'true'|'false'})" [0..1]

 clustered="string (value comes from list: {'true'|'false'})" [0..1]

 required="string (value comes from list: {'true'|'false'})" [0..1]

 addtenantid="string (value comes from list: {'true'|'false'})" [0..1]

>

   <indexkey> ... </indexkey> [1..*]

</specify_index>
```

### Schema Component Representation

```
<element name="specify_index">

   <complexType>

      <sequence>

         <element ref="indexkey" maxOccurs="unbounded"/>

      </sequence>

      <attribute name="name" type="string" use="optional"/>
      <attribute name="object" type="string" use="required"/>
      <attribute name="primary" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="unique" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="clustered" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="required" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="addtenantid" use="default" value="true">
```

```
          <simpleType>
             <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: drop_index

Properties

| | |
|---|---|
| **Name** | drop_index |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop index by name or definition.

### Attributes:

- **object:** Object's name of the target index.
- **name:** Index name.

## Use Example:

The follow example would drop the index created through the specify_index command.

```
<drop_index object="OBJECTNAME" name="INDEX_NDX1" />
```

XML Instance Representation

```
<drop_index
 name="string" [0..1]
 object="string" [1]
>
   <indexkey> ... </indexkey> [0..*]
</drop_index>
```

Schema Component Representation

```
<element name="drop_index">
   <complexType>
      <sequence>
         <element ref="indexkey" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="name" type="string" use="optional"/>
      <attribute name="object" type="string" use="required"/>
   </complexType>
</element>
```

## Element: indexkey

Properties

| | |
|---|---|
| **Name** | indexkey |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

The indexkey element is a nested part of the specify_index and drop_index elements.

### Attributes:

- **column:** Attribute name associated with this index.
- **asceding:** When checked, indicates that this is an ascending key.

## Use Example:

Refer to specify_index for a complete example of this element.

XML Instance Representation

```
<indexkey
 column="string" [1]
 ascending="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

Schema Component Representation

```
<element name="indexkey">
   <complexType>
      <attribute name="column" type="string" use="required"/>
      <attribute name="ascending" use="default" value="true">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: specify_synonym_domain

Properties

| | |
|---|---|
| **Name** | specify_synonym_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Specifies a synonym domain.

## Attributes:

- **domainid:** Domain's name or unique identifier.
- **description:** Description of the domain's value.
- **maxtype:** Internal Maximo data type.
- **length:** Defines the number of characters allowed to be inputed by the user.
- **overwrite:** Override an existent domain.
- **internal:** Flag to indicate if the domain is internal and/or localizable. 0: localizable and not internal; 1: internal and not localizable; 2: not localizable but not internal.

# Use Examples

```
<specify_synonym_domain domainid="TESTING" >
    <synonymvalueinfo maxvalue="ONE" value="ONE" defaults="true"/>
    <synonymvalueinfo maxvalue="TWO" value="TWO" defaults="true"/>
</specify_synonym_domain>
```

XML Instance Representation

```
<specify_synonym_domain
 domainid="string" [1]
 description="string" [0..1]
 maxtype="string (value comes from list: {'ALN'|'LONGALN'|'LOWER'|'UPPER'})" [0..1]
 length="NMTOKEN" [0..1]
 overwrite="string (value comes from list: {'true'|'false'})" [0..1]
 internal="string (value comes from list: {'true'|'false'})" [0..1]
>
   <synonymvalueinfo> ... </synonymvalueinfo> [1..*]
</specify_synonym_domain>
```

Schema Component Representation

```
<element name="specify_synonym_domain">
   <complexType>
      <sequence>
         <element ref="synonymvalueinfo" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="domainid" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="maxtype" use="default" value="UPPER">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
               <enumeration value="LONGALN"/>
               <enumeration value="LOWER"/>
               <enumeration value="UPPER"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="length" type="NMTOKEN" use="default" value="8"/>
      <attribute name="overwrite" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="internal" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: add_synonyms

## Properties

| | |
|---|---|
| **Name** | add_synonyms |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

## Documentation

Add a new synonym to an existent synonyms domain ( created (created previously by a specify_synonym_domain;

## Attributes:

- **domainid:** Target domain.

## Use Example:

The add_synonyms element is combined with the synonymvalueinfo element.

```
<add_synonyms domainid="TESTING" >

    <synonymvalueinfo value="THREE" maxvalue="THREE"  defaults="true" description="Imported from building model" />

</add_synonyms>
```

## XML Instance Representation

```
<add_synonyms

 domainid="string" [1]

>

   <synonymvalueinfo> ... </synonymvalueinfo> [1..*]

</add_synonyms>
```

## Schema Component Representation

```
<element name="add_synonyms">

   <complexType>

      <sequence>

         <element ref="synonymvalueinfo" maxOccurs="unbounded"/>

      </sequence>

      <attribute name="domainid" type="string" use="required"/>

   </complexType>
```

```
</element>
```

## Element: synonymvalueinfo

Properties

| | |
|---|---|
| **Name** | synonymvalueinfo |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines the synonym domain's value. This element is used nested to specify_synonym_domain and add_synonyms elements.

### Attributes:

- **value:** Domain's value.
- **maxvalue:** Maximo internal value.
- **defaults:** Default value to this domain.
- **description:** Brief description about the value itself.

## Use Example:

Refers to specify_synonym_domain and add_synonyms elements to a complete example.

XML Instance Representation

```
<synonymvalueinfo
 value="string" [1]
 maxvalue="string" [1]
 defaults="string (value comes from list: {'true'|'false'})" [1]
 description="string" [0..1]
/>
```

Schema Component Representation

```
<element name="synonymvalueinfo">
```

```
   <complexType>
      <attribute name="value" type="string" use="required"/>
      <attribute name="maxvalue" type="string" use="required"/>
      <attribute name="defaults" use="required">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="description" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: specify_aln_domain

Properties

| | |
|---|---|
| **Name** | specify_aln_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Specifies a **ALN** domain, a domain that handle directly with maxtypes.

### Attributes:

- **domainid:** Domain's name or unique identifier.
- **description:** Description of the domain's value.
- **maxtype:** Internal Maximo data type.
- **length:** Defines the number of characters allowed to be inputed by the user.
- **overwrite:** Override an existent domain.

- **internal:** Flag to indicate if the domain is internal and/or localizable. 0: localizable and not internal; 1: internal and not localizable; 2: not localizable but not internal.

## Use Example:

This domain represents the characters that can be selected through a domain at the Maximo UI.

```
<specify_aln_domain maxtype="ALN" length="3" domainid="ALPHABETIC" description="Simpl
e domain with aphabetic's characters" overwrite="false">

    <alnvalueinfo value="A" description="Letter A" />

    <alnvalueinfo value="B" description="Letter B" />

    <alnvalueinfo value="C" description="Letter C" />

    <alnvalueinfo value="D" description="Letter D" />

    <alnvalueinfo value="E" description="Letter E" />

    <alnvalueinfo value="F" description="Letter F" />

    <alnvalueinfo value="G" description="Letter G" />

    <alnvalueinfo value="H" description="Letter H" />

    <alnvalueinfo value="I" description="Letter I" />

    <alnvalueinfo value="J" description="Letter J" />

    <alnvalueinfo value="K" description="Letter K" />

    <alnvalueinfo value="L" description="Letter L" />

    <alnvalueinfo value="M" description="Letter M" />

    <alnvalueinfo value="N" description="Letter N" />

    <alnvalueinfo value="O" description="Letter O" />

    <alnvalueinfo value="P" description="Letter P" />

    <alnvalueinfo value="Q" description="Letter Q" />

    <alnvalueinfo value="R" description="Letter R" />

    <alnvalueinfo value="S" description="Letter S" />

    <alnvalueinfo value="T" description="Letter T" />

    <alnvalueinfo value="U" description="Letter U" />

    <alnvalueinfo value="V" description="Letter V" />

    <alnvalueinfo value="W" description="Letter W" />

    <alnvalueinfo value="X" description="Letter X" />

    <alnvalueinfo value="Y" description="Letter Y" />

    <alnvalueinfo value="Z" description="Letter Z" />

</specify_aln_domain>
```

XML Instance Representation

```
<specify_aln_domain
 domainid="string" [1]
 description="string" [0..1]
 maxtype="string (value comes from list: {'ALN'|'LONGALN'|'LOWER'|'UPPER'})" [0..1]
 length="NMTOKEN" [0..1]
 overwrite="string (value comes from list: {'true'|'false'})" [0..1]
 internal="string (value comes from list: {'true'|'false'})" [0..1]
>
   <alnvalueinfo> ... </alnvalueinfo> [1..*]
</specify_aln_domain>
```

Schema Component Representation

```
<element name="specify_aln_domain">
   <complexType>
      <sequence>
         <element ref="alnvalueinfo" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="domainid" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="maxtype" use="default" value="UPPER">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
               <enumeration value="LONGALN"/>
               <enumeration value="LOWER"/>
               <enumeration value="UPPER"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="length" type="NMTOKEN" use="default" value="8"/>
      <attribute name="overwrite" use="default" value="false">
         <simpleType>
            <restriction base="string">
```

```
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
       <attribute name="internal" use="default" value="false">
          <simpleType>
             <restriction base="string">
                <enumeration value="true"/>
                <enumeration value="false"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: specify_numeric_domain

Properties

| | |
|---|---|
| **Name** | specify_numeric_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Specifies a **numeric** domain. A numeric domain works with **AMOUNT**,**DECIMAL**,**DURATION**,**FLOAT**,**INTEGER** or a **SMALLINT** values, those values should be set through the **maxtype** attribute.

## Attributes:

- **domainid:** Domain's name or unique identifier.
- **description:** Description of the domain's value.
- **maxtype:** Internal Maximo data type.

- **length:** Defines the number of characters allowed to be inputed by the user.
- **scale:** Scale of the numeric domain value for numeric domains.
- **overwrite:** Override an existent domain.
- **internal:** Flag to indicate if the domain is internal and/or localizable. 0: localizable and not internal; 1: internal and not localizable; 2: not localizable but not internal.

## Use Example:

```
<specify_numeric_domain length="2" domainid="NUMBERS" description="A domain with 5 it
ems, for scroll/trunctation testing" overwrite="false">

      <numericvalueinfo value="0"  description="0" />

      <numericvalueinfo value="1"  description="1" />

      <numericvalueinfo value="2"  description="2" />

      <numericvalueinfo value="3"  description="3" />

      <numericvalueinfo value="4"  description="4" />

      <numericvalueinfo value="5"  description="5" />

<specify_numeric_domain/>
```

### XML Instance Representation

```
<specify_numeric_domain

 domainid="string" [1]

 description="string" [0..1]

 maxtype="string (value comes from list: {'AMOUNT'|'DECIMAL'|'DURATION'|'FLOAT'|'INTE
GER'|'SMALLINT'})" [0..1]

 length="NMTOKEN" [0..1]

 scale="NMTOKEN" [0..1]

 overwrite="string (value comes from list: {'true'|'false'})" [0..1]

 internal="string (value comes from list: {'true'|'false'})" [0..1]

>

   <numericvalueinfo> ... </numericvalueinfo> [1..*]

</specify_numeric_domain>
```

### Schema Component Representation

```
<element name="specify_numeric_domain">

   <complexType>

      <sequence>

         <element ref="numericvalueinfo" maxOccurs="unbounded"/>
```

```xml
   </sequence>
   <attribute name="domainid" type="string" use="required"/>
   <attribute name="description" type="string" use="optional"/>
   <attribute name="maxtype" use="default" value="INTEGER">
      <simpleType>
         <restriction base="string">
            <enumeration value="AMOUNT"/>
            <enumeration value="DECIMAL"/>
            <enumeration value="DURATION"/>
            <enumeration value="FLOAT"/>
            <enumeration value="INTEGER"/>
            <enumeration value="SMALLINT"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="length" type="NMTOKEN" use="default" value="4"/>
   <attribute name="scale" type="NMTOKEN" use="default" value="0"/>
   <attribute name="overwrite" use="default" value="false">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="internal" use="default" value="false">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
</complexType>
```

```
</element>
```

## Element: specify_crossover_domain

Properties

| | |
|---|---|
| **Name** | specify_crossover_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

A crossover domain will allow additional values to be carried or "crossed" over from a child table when a field is populated in a parent table, with data that connects the two tables based on a where clause that creates a one to one relationship between the two objects.

### Attributes:

- **domainid:** Domain's name or unique identifier.
- **description:** Description of the domain's value.
- **overwrite:** Override an existent domain.
- **validationwhereclause:** The Validation Where Clause is critical to the crossover functioning and defines the one to one relationship between the two objects that will trigger the crossover when that criteria is met
- **listwhereclause:** Where clause used to get list members
- **errorbundle:** Resource bundle of message to be returned if value fails to match domain
- **errorkey:** Key to access error message (displayed if value not in domain)
- **objectname:** The Object value should be the child table and is the object the value will be brought over from.
- **internal:** Flag to indicate if the domain is internal and/or localizable. 0: localizable and not internal; 1: internal and not localizable; 2: not localizable but not internal.

**NOTE:** For `listwhereclause` and `validationwhereclause`, Maximo system does not validate your entry for syntax or any other errors. Be sure that you have typed a correct WHERE clause. If you make errors, errors do not become apparent until you configure the database.

### Use Example:

In the following example we will set up a crossover from the `ASSET` table to the `WORKORDER` table. When the `ASSETNUM` field of Work Order Tracking is populated while inserting a work order record, the crossover will populate the corresponding `SERIALNUM` value from the ASSET record in the `WORKORDER.SERIALNUM` field. Because the WORKORDER.ASSETNUM is the attribute used in the where clause from the destination table, we need to associate the `ASSET2WORK` domain to `DOMAINID` column for the ASSETNUM attribute. In the Source Field, we are going to enter SERIALNUM. Note that the lookup can be used to select the ASSET.SERIALNUM value as the object for the child table has been defined. The Destination field is a free form field due to the fact that the domain needs to be created before the domain can be associated to the destination object. It is critical to enter the proper value for the crossover to work. We have created a SERIALNUM attribute on the WORKORDER object so we will enter that value.It is possible to have more than one source and destination pairing for one crossover domain. Please refer to crossovervalueinfo element for a further information abour sources and destinations <;specify_crossover_domain objectname="ASSET" validationwhereclause="ASSETNUM = :ASSETNUM and SITEID = :SITEID" domainid="ASSET2WORK" description="Crossover from Asset to Work Order Tracking" > <crossovervalueinfo sourcefield="SERIALNUM"/> </specify_crossover_domain>

## XML Instance Representation

```
<specify_crossover_domain
 domainid="string" [1]
 description="string" [0..1]
 overwrite="string (value comes from list: {'true'|'false'})" [0..1]
 validationwhereclause="string" [1]
 listwhereclause="string" [0..1]
 errorbundle="string" [0..1]
 errorkey="string" [0..1]
 objectname="string" [1]
 internal="string (value comes from list: {'true'|'false'})" [0..1]
>
   <crossovervalueinfo> ... </crossovervalueinfo> [1..*]
</specify_crossover_domain>
```

## Schema Component Representation

```
<element name="specify_crossover_domain">
   <complexType>
      <sequence>
         <element ref="crossovervalueinfo" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="domainid" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="overwrite" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="validationwhereclause" type="string" use="required"/>
      <attribute name="listwhereclause" type="string" use="optional"/>
      <attribute name="errorbundle" type="string" use="optional"/>
      <attribute name="errorkey" type="string" use="optional"/>
      <attribute name="objectname" type="string" use="required"/>
      <attribute name="internal" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: specify_table_domain

Properties

| | |
|---|---|
| **Name** | specify_table_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |

| | |
|---|---|
| **Abstract** | no |

Documentation

Defines a table domain. You do add a table domain when you want to add a domain that draws its values directly from a column in the database. This process creates a dynamic value list because the values it draws from the database might change.

## Attributes:

- **domainid:** Domain's name or unique identifier.
- **description:** Description of the domain's value.
- **overwrite:** Override an existent domain.
- **validationwhereclause:** The Validation Where Clause is critical to the crossover functioning and defines the one to one relationship between the two objects that will trigger the crossover when that criteria is met
- **listwhereclause:** Type the part of the clause that specifies the values that you want to select based on the validation WHERE clause.
- **errorbundle:** Resource bundle of message to be returned if value fails to match domain
- **errorkey:** Key to access error message (displayed if value not in domain)
- **objectname:** The Object value should be the child table and is the object the value will be brought over from.
- **internal:** Flag to indicate if the domain is internal and/or localizable. 0: localizable and not internal; 1: internal and not localizable; 2: not localizable but not internal.

**NOTE:** For `listwhereclause` and `validationwhereclause`, Maximo system does not validate your entry for syntax or any other errors. Be sure that you have typed a correct WHERE clause. If you make errors, errors do not become apparent until you configure the database.

## Use Example:

The following example defines a dynamic asset list.

```
  <specify_table_domain domainid="ASSETLIST" objectname="ASSET" validationwhereclause
="1=1" description="Table domain with a list of assets" listwhereclause="1=1" />
```

XML Instance Representation

```
<specify_table_domain
 domainid="string" [1]
 description="string" [0..1]
 overwrite="string (value comes from list: {'true'|'false'})" [0..1]
 validationwhereclause="string" [1]
 listwhereclause="string" [0..1]
 errorbundle="string" [0..1]
 errorkey="string" [0..1]
 objectname="string" [1]
 internal="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

Schema Component Representation

```
<element name="specify_table_domain">
   <complexType>
      <attribute name="domainid" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="overwrite" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="validationwhereclause" type="string" use="required"/>
      <attribute name="listwhereclause" type="string" use="optional"/>
      <attribute name="errorbundle" type="string" use="optional"/>
      <attribute name="errorkey" type="string" use="optional"/>
      <attribute name="objectname" type="string" use="required"/>
      <attribute name="internal" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
```

```
</element>
```

## Element: drop_domain

Properties

| | |
|---|---|
| **Name** | drop_domain |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop an existent domain.

### Attributes:

- **domaid:** Unique domain identification.

## Use Example:

The sample bellow delete an existent domain from the system.

```
    <drop_domain domainid="ASSETLIST"/>
```

XML Instance Representation

```
<drop_domain
 domainid="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_domain">
   <complexType>
      <attribute name="domainid" type="string" use="required"/>
   </complexType>
</element>
```

## Element: alnvalueinfo

### Properties

| | |
|---|---|
| **Name** | alnvalueinfo |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Represents the `ALN` domain values.

### Attributes:

- **value:** ALN domain value.
- **description:** Description about the ALN domain value.

## Use Example:

To a complete example about this element, please refers to specify_aln_domain element.

### XML Instance Representation

```
<alnvalueinfo
 value="string" [1]
 description="string" [0..1]
/>
```

### Schema Component Representation

```
<element name="alnvalueinfo">
   <complexType>
      <attribute name="value" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: numericvalueinfo

### Properties

| | |
|---|---|
| **Name** | numericvalueinfo |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Defines the numeric domain's value. This element is used nested to specify_numeric_domain elements.

### Attributes:

- **value:** Domain's value.
- **description:** Brief description about the value itself.

## Use Example:

Refers to the specify_numeric_domain element for a complete example of this element.

### XML Instance Representation

```
<numericvalueinfo
 value="NMTOKEN" [1]
 description="string" [0..1]
/>
```

### Schema Component Representation

```
<element name="numericvalueinfo">
   <complexType>
      <attribute name="value" type="NMTOKEN" use="required"/>
      <attribute name="description" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: crossovervalueinfo

### Properties

| | |
|---|---|
| **Name** | crossovervalueinfo |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

It is critical to enter the proper value for the crossover to work. the `crossovervalueinfo` defines more than one source and destination pairing for one crossover domain.

### Attributes:

- **sourcefield:** Field from which a user will copy the database field value
- **destfield:** Field to which a user will copy the database field value
- **copyofnull:** Boolean value, when it is `TRUE`, the value of the source or destination attributes will only be set to the opposite direction attribute when its value is `NULL` or empty.
- **copyevenifsrcnull:** Boolean value, if true, the destination attribute will also be set to NULL when the source attribute is `NULL` or empty. If false, the destination attribute will be left as what it is when the source attribute is NULL or empty.
- **copyonlyifdestnull:** Boolean value, when it is `TRUE`, the value of the source attribute will only be set to destination attribute when the destination attribute's value is `NULL` or empty.

## Use Example:

Refer to specify_crossover_domain for a further information about this sample.

### XML Instance Representation

```
<crossovervalueinfo
 sourcefield="string" [1]
 destfield="string" [0..1]
 copyifnull="string (value comes from list: {'true'|'false'})" [0..1]
 copyevenifsrcnull="string (value comes from list: {'true'|'false'})" [0..1]
 copyonlyifdestnull="string (value comes from list: {'true'|'false'})" [0..1]
```

```
/>
```

Schema Component Representation

```
<element name="crossovervalueinfo">
   <complexType>
      <attribute name="sourcefield" type="string" use="required"/>
      <attribute name="destfield" type="string" use="optional"/>
      <attribute name="copyifnull" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="copyevenifsrcnull" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="copyonlyifdestnull" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: add_sigoption

### Properties

| | |
|---|---|
| **Name** | add_sigoption |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

`grantapp` and `grantoption` determine what groups should be granted application authorization for the new `sigoption`. If the grant is based on `STARTCNTR.READ`, then groups that have READ access to the Start Center will be authorized for the new option when the script is run.
`granteveryone` will grant the sigoption to the all users group.
`grantcondition` will apply the condition to all ApplicationAuth records created.
`langcode` is not longer required, and actually this setting will be ignored. The code for the base language of the database will be used. The attribute is only kept to allow existing scripts to be valid.
Attributes:

- **app:** Signature Option Application Name.
- **optionname:** Signature Option Name.
- **decription:** Signature Option Description.
- **esigenabled:** Indicates Electronic Signature (ESIG) is enabled for this function.
- **visible:** Indicates whether this sigoption can be granted via the Security Groups application. It will be visible and grantable if the value is 1 and not visible if the value is 0. If it is not visible, it should be granted and revoked along with another application by listing it in the `ALSOGRANTS` and `ALSOREVOKES` columns.
- **alsogrants:** List of other sigoptions that will be automatically granted when this sigoption is selected.
- **alsorevokes:** List of other sigoptions that will be automatically revoked when this sigoption is deselected.
- **prerequisite:** List of other sigoptions that must be selected before this sigoption can be selected.
- **langcode:** Language Column.
- **grantapp:** List of applications that will be automatically granted when this sigoption is selected.
- **grantoption:** Signature Option that will be automatically granted when this sigoption is selected.

- **granteveryone:** Automatically granted access for everyone when this sigoption is selected.
- **grantcondition:** Non-persistent field used by the Security Groups application to populate restrictions assigned to a Security Group that are eventually stored in the `APPLICATIONAUTH` table.

## Use Example:

The sample below explain how the sigoption command can be used to grant access for the common actions present in `APPBASIC` application in Maximo. Be ware about the link between some options made through the `prerequisite`, `alsogrants` and `alsorevokes` attributes.

```
<add_sigoption visible="true" app="APPBASIC" optionname="DELETE"     description="Delete Record"           grantapp="APPBASIC"  grantoption="SAVE"   esigenabled="false" langcode="EN" prerequisite="SAVE" />

<add_sigoption visible="true" app="APPBASIC" optionname="BOOKMARK"   description="Add to Bookmarks"        grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHMORE" description="More Search Fields"      grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="READ"       description="Read Access to Ref Main" grantapp="STARTCNTR" grantoption="READ"   esigenabled="false" langcode="EN" alsogrants="CLEAR,BOOKMARK,NEXT,PREVIOUS" alsorevokes="ALL" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHTIPS" description="View Search Tips"        grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHBOOK" description="Bookmarks"               grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHSQRY" description="Save Current Query"      grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHVMQR" description="View/Manage Queries"     grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SEARCHWHER" description="Where Clause"            grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="SAVE"       description="Save Record"             grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" alsorevokes="INSERT,DUPLICATE,DELETE" />

<add_sigoption visible="true" app="APPBASIC" optionname="INSERT"     description="New Record"              grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" langcode="EN" alsogrants="SAVE" />
```

```
<add_sigoption visible="true" app="APPBASIC" optionname="CLEAR"      description="Cle
ar Changes"           grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="PREVIOUS"   description="Pre
vious Record"         grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="NEXT"       description="Nex
t Record"             grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="DUPLICATE"  description="Dup
licate Record"        grantapp="APPBASIC"  grantoption="INSERT" esigenabled="false" l
angcode="EN" prerequisite="INSERT" />

<add_sigoption visible="true" app="APPBASIC" optionname="ITEMIMAGE"  description="Add
/Modify Image"        grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="ASSOCFOLD"  description="Ass
ociate Folders"       grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="MANAGEFOLD" description="Man
age Folders"          grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />

<add_sigoption visible="true" app="APPBASIC" optionname="MANAGELIB"  description="Man
age Library"          grantapp="APPBASIC"  grantoption="READ"   esigenabled="false" l
angcode="EN" />
```

XML Instance Representation

```
<add_sigoption
 app="string" [1]
 optionname="string" [1]
 description="string" [1]
 esigenabled="string (value comes from list: {'true'|'false'})" [0..1]
 visible="string (value comes from list: {'true'|'false'})" [0..1]
 alsogrants="string" [0..1]
 alsorevokes="string" [0..1]
 prerequisite="string" [0..1]
 langcode="string" [0..1]
 grantapp="string" [0..1]
 grantoption="string" [0..1]
 granteveryone="string (value comes from list: {'true'|'false'})" [0..1]
 grantcondition="string" [0..1]
>
```

```
   Start Sequence [0..1]
      <longdescription> ... </longdescription> [1]
   End Sequence
</add_sigoption>
```

Schema Component Representation

```
<element name="add_sigoption">
   <complexType>
      <sequence minOccurs="0" maxOccurs="1">
         <element ref="longdescription"/>
      </sequence>
      <attribute name="app" type="string" use="required"/>
      <attribute name="optionname" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="esigenabled" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="visible" use="default" value="true">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="alsogrants" type="string" use="optional"/>
      <attribute name="alsorevokes" type="string" use="optional"/>
      <attribute name="prerequisite" type="string" use="optional"/>
      <attribute name="langcode" type="string" use="optional"/>
      <attribute name="grantapp" type="string" use="optional"/>
      <attribute name="grantoption" type="string" use="optional"/>
      <attribute name="granteveryone" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="grantcondition" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: drop_sigoption

Properties

| | |
|---|---|
| **Name** | drop_sigoption |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop a sigoption

### Attributes:

- **app:** Signature Option Application name.
- **optioname:** Signature Option name.

## Use Example:

The follow example delete the option `NEXT` associated with `APPBASIC` application.
<drop_sigoption app="APPBASIC" optionname="NEXT" />

XML Instance Representation

```
<drop_sigoption
 app="string" [1]
 optionname="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_sigoption">
   <complexType>
      <attribute name="app" type="string" use="required"/>
      <attribute name="optionname" type="string" use="required"/>
   </complexType>
</element>
```

## Element: create_module

Properties

| | |
|---|---|
| **Name** | create_module |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Create a module application that incorporetes a group of applications menus.

### Attributes:

- **module:** Maximo Module Name
- **description:** Maximo Module Description
- **menu_position:** Maximo Module's position at the module menu.
- **menu_param:** If exists, a menu param that enables some app to work with a data start point.
- **image:** Reference for a menu image that represents the module fuctions.

## Use Example:

The sample bellow refers to create an simple module application with some applications inside that represents a group of Maximo applications grouped by purpose. Refer to the module_menu_app element, for a furhter information about module's menu.

```
<create_module module="APPBASIC" description="Basic Applications">

    <module_menu_app app="APPBASIC" image="appimg_basic.gif"/>

    <module_menu_app app="APPMOBILE" image="appimg_basic.gif"/>

    <module_menu_app app="APPNEG" image="appimg_basic.gif"/>

    <module_menu_app app="APPCOMM" image="appimg_basic.gif"/>

</create_module>
```

This statement inserts into to the module menu

XML Instance Representation

```
<create_module
 module="string" [1]
 description="string" [1]
 menu_position="string (value comes from list: {'first'|'last'|'before'|'after'})" [0
..1]
 menu_param="string" [0..1]
 image="string" [0..1]
>
   Start Choice [1..*]
      <module_menu_app> ... </module_menu_app> [1]
      <module_menu_header> ... </module_menu_header> [1]
   End Choice
</create_module>
```

Schema Component Representation

```
<element name="create_module">
   <complexType>
      <choice maxOccurs="unbounded">
         <element ref="module_menu_app"/>
         <element ref="module_menu_header"/>
      </choice>
      <attribute name="module" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="menu_position" use="default" value="last">
         <simpleType>
            <restriction base="string">
               <enumeration value="first"/>
               <enumeration value="last"/>
               <enumeration value="before"/>
               <enumeration value="after"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="menu_param" type="string" use="optional"/>
      <attribute name="image" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: modify_module

Properties

| | |
|---|---|
| **Name** | modify_module |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies a module existent module application.

###Attributes:

- **module:** Maximo Module Name
- **description:** Maximo Module Description
- **menu_position:** Maximo Module's position at the menu.

- **menu_param:** If exists, a menu param that enables some app to work with a data start point.
- **image:** Reference for a menu image that represents the module fuctions.

## Use Example:

The example below modifies the module `APPBASIC` created through the example create_module previously, by adding a new module menu `APPGISSERVICE`.

```
<modify_module module="APPBASIC" description="Basic Applications">

    <module_menu_app app="APPGISSERVICE" image="appimg_basic.gif"/>

</modify_module>
```

Note that image refers to the module's image, not for the applications

### XML Instance Representation

```
<modify_module

 module="string" [1]

 description="string" [0..1]

 menu_position="string (value comes from list: {'first'|'last'|'before'|'after'})" [0..1]

 menu_pos_param="string" [0..1]

 image="string" [0..1]

/>
```

### Schema Component Representation

```
<element name="modify_module">

   <complexType>

      <attribute name="module" type="string" use="required"/>

      <attribute name="description" type="string" use="optional"/>

      <attribute name="menu_position" use="optional">

         <simpleType>

            <restriction base="string">

               <enumeration value="first"/>

               <enumeration value="last"/>

               <enumeration value="before"/>
```

```
                <enumeration value="after"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="menu_pos_param" type="string" use="optional"/>
    <attribute name="image" type="string" use="optional"/>
  </complexType>
</element>
```

## Element: module_app

Properties

| | |
|---|---|
| **Name** | module_app |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Create a module for an specific application.

### Attributes:

- **module:** Maximo Module Name
- **app:** Application name.
- **description:** Maximo Module Description
- **menu_position:** Maximo Module's position at the menu.
- **menu_pos_param:** Name of the app or optionname. Will be null if this is a header.
- **image:** Reference for a menu image that represents the module fuctions.

## Use Example:

The sample bellow refers to create an simple module to an specific application named `FINANCIAL`.

```
<module_app module="FINANCIAL" app="BUDGET" menu_position="last" />
```

XML Instance Representation

```
<module_app
 module="string" [1]
 app="string" [1]
 menu_position="string (value comes from list: {'first'|'last'|'before'|'after'})" [0
..1]
 menu_pos_param="string" [0..1]
 image="string" [0..1]
/>
```

Schema Component Representation

```
<element name="module_app">
   <complexType>
      <attribute name="module" type="string" use="required"/>
      <attribute name="app" type="string" use="required"/>
      <attribute name="menu_position" use="default" value="last">
         <simpleType>
            <restriction base="string">
               <enumeration value="first"/>
               <enumeration value="last"/>
               <enumeration value="before"/>
               <enumeration value="after"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="menu_pos_param" type="string" use="optional"/>
      <attribute name="image" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: drop_module

Properties

| | |
|---|---|
| **Name** | drop_module |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop a specific module. Dropping an app will also drop related sigoption and maxmenu entries

## Attributes:

- **module:** Module's name.

## Use Example:

The example bellow will drop the `APPBASIC` module.

```
<drop_module module="APPBASIC" />
```

XML Instance Representation

```
<drop_module
 module="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_module">
   <complexType>
      <attribute name="module" type="string" use="required"/>
   </complexType>
</element>
```

## Element: create_app

Properties

|   |   |
|---|---|
| **Name** | create_app |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Creates an Maximo application.

## Attributes:

- **app:** Application Name (Name of the .RUN file)
- **description:** Brief description about the application functionality.
- **restrictions:** Application Restrictions.
- **orderby:** Application Default Order By.
- **maintbname:** Main Object name (table or view) for APP used to get JAVA MBO set.

## Use Example:

The following example creates an app for the Maximo application.

```
<create_app app="APPBASIC" description="Basic Application" maintbname="APPBASIC" / >
```

XML Instance Representation

```
<create_app
 app="string" [1]
 description="string" [1]
 restrictions="string" [0..1]
 orderby="string" [0..1]
 maintbname="string" [0..1]
/>
```

Schema Component Representation

```
<element name="create_app">
   <complexType>
      <attribute name="app" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="restrictions" type="string" use="optional"/>
      <attribute name="orderby" type="string" use="optional"/>
      <attribute name="maintbname" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: modify_app

Properties

| | |
|---|---|
| **Name** | modify_app |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modify an existent app created by create_app element.

### Attributes:

- **app:** Application Name (Name of the .RUN file)
- **description:** Brief description about the application functionality.
- **restrictions:** Application Restrictions.
- **orderby:** Application Default Order By.
- **maintbname:** Main Object name (table or view) for APP used to get JAVA MBO set.

### Use Example:

The foolow example handle with the main table change from the original app created through the example in create_app element description.

```
<modify_app app="APPBASIC" maintbname="PLUSAPPBASIC" >
  <longdescription>Some description about the main table's new functions</longdescrip
tion>
</modify_app>
```

XML Instance Representation

```
<modify_app
 app="string" [1]
 description="string" [0..1]
 restrictions="string" [0..1]
 orderby="string" [0..1]
 maintbname="string" [0..1]
/>
```

Schema Component Representation

```
<element name="modify_app">
   <complexType>
      <attribute name="app" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="restrictions" type="string" use="optional"/>
      <attribute name="orderby" type="string" use="optional"/>
      <attribute name="maintbname" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: drop_app

Properties

| | |
|---|---|
| **Name** | drop_app |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop a specific application. Dropping an app will also drop related `sigoption` and `maxmenu` entries

### Attributes:

- **app:** Applications unique name.

## Use Example:

The example bellow deletes the application created by the create_app element use examples section.

```
<drop_app app="APPBASIC"/ >
```

### XML Instance Representation

```
<drop_app
 app="string" [1]
/>
```

### Schema Component Representation

```
<element name="drop_app">
   <complexType>
      <attribute name="app" type="string" use="required"/>
   </complexType>
</element>
```

## Element: module_menu_app

### Properties

| | |
|---|---|
| **Name** | module_menu_app |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Nested element from module_app and create_module, where the menu for applications will be defined as well as images will be associated wiht defined menus.

## Attributes:

- **app:** Applications unique name.

- **image:** Associated image that will represents the application among the module hyperlinks.

## Use Example:

```
For a further information about the example of this element, please refers to [create
_module](#element_create_module) element usage samples.
```

XML Instance Representation

```
<module_menu_app
 app="string" [1]
 image="string" [0..1]
/>
```

Schema Component Representation

```
<element name="module_menu_app">
   <complexType>
      <attribute name="app" type="string" use="required"/>
      <attribute name="image" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: module_menu_header

Properties

| | |
|---|---|
| **Name** | module_menu_header |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines a menu header for a module/submodule application. This is an nested element, present on create_module root element.

## Attributes:

- **headerdescription:** Header description. Will be null if this is not a header.

## Use Example:

This example can also be used or refered by the create_module element, however, the complete sample is represented by the code's snap bellow.

```
<create_module module="APPBASIC" description="Basic Application Description">
    <module_menu_app app="APPBASIC"/>
    <module_menu_header headerdescription="Sub menu for mobile basic apps">
        <module_menu_app app="APPMOBILE"/>
        <module_menu_app app="APPGIS"/>
    </module_menu_header>
 </create_module>
```

### XML Instance Representation

```
<module_menu_header
 headerdescription="string" [1]
>
   Start Sequence [1..*]
      <module_menu_app> ... </module_menu_app> [1]
   End Sequence
</module_menu_header>
```

### Schema Component Representation

```
<element name="module_menu_header">
   <complexType>
      <sequence maxOccurs="unbounded">
         <element ref="module_menu_app"/>
      </sequence>
      <attribute name="headerdescription" type="string" use="required"/>
   </complexType>
</element>
```

## Element: create_app_menu

## Properties

| | |
|---|---|
| **Name** | create_app_menu |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

## Documentation

This statement creates a new menu for an specific application in Maximo.

## Attributes:

- **app:** Application's unique name.
- **type:*Application Type (.RUN .EXE etc.)

## Use Example:

The bellow example creates a menu structure that will allow de user to delete, add or edit an specific element from the list tab of the 'MP3 Basic Application'. Once the MP3 file were selected, the user are ablle to upload or download lirics from the main tab (The element at this time were selected and is presenting several details) lyrics associated with this particular mp3 file. For a further information about the nested elements of this example please refere to the app_menu_option,app_menu_header and menu_separator elements.

```
<create_app_menu app="MP3BASICAPP">
    <app_menu_header headerdescription="">
        <app_menu_option tabdisplay="LIST" option="DELETE"/>
        <app_menu_option tabdisplay="LIST" option="ADD"/>
        <app_menu_option tabdisplay="LIST" option="EDIT"/>
    </app_menu_header>
    <app_menu_option tabdisplay="MAIN" option="DOWNLOADLYRICS"/>
    <menu_separator/>
    <app_menu_option tabdisplay="MAIN" option="UPLOADLYRICS"/>
 </create_app_menu>
```

## XML Instance Representation

```
<create_app_menu
 app="string" [1]
 type="string (value comes from list: {'action'|'tool'|'search'})" [0..1]
>
```

```
   Start Choice [1..*]
      <app_menu_option> ... </app_menu_option> [1]
      <menu_separator> ... </menu_separator> [1]
      <app_menu_header> ... </app_menu_header> [1]
   End Choice
</create_app_menu>
```

Schema Component Representation

```
<element name="create_app_menu">
   <complexType>
      <choice maxOccurs="unbounded">
         <element ref="app_menu_option"/>
         <element ref="menu_separator"/>
         <element ref="app_menu_header"/>
      </choice>
      <attribute name="app" type="string" use="required"/>
      <attribute name="type" use="default" value="action">
         <simpleType>
            <restriction base="string">
               <enumeration value="action"/>
               <enumeration value="tool"/>
               <enumeration value="search"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: app_menu_option

Properties

| | |
|---|---|
| **Name** | app_menu_option |

| | |
|---|---|
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines a menu into a application menu structure.

## Attributes:

- **option:** Option's name associated to a Maximo Application.
- **image:** Application associated image/icon that will be shown at the UI.
- **accesskey:** Used for Access Keys on `Actions`.
- **tabdisplay:** Indicates the tabs on which an option can be displayed (i.e. `LIST`|`ALL`|`MAIN`).

## Use Example:

Please refer to create_app_menu element at the use example section, for a complete menu example.

XML Instance Representation

```
<app_menu_option
 option="string" [1]
 image="string" [0..1]
 accesskey="string" [0..1]
 tabdisplay="string (value comes from list: {'LIST'|'MAIN'|'ALL'})" [1]
/>
```

Schema Component Representation

```
<element name="app_menu_option">
   <complexType>
      <attribute name="option" type="string" use="required"/>
      <attribute name="image" type="string" use="optional"/>
      <attribute name="accesskey" type="string" use="optional"/>
      <attribute name="tabdisplay" use="required">
         <simpleType>
            <restriction base="string">
               <enumeration value="LIST"/>
```

```
                <enumeration value="MAIN"/>
                <enumeration value="ALL"/>
             </restriction>
          </simpleType>
       </attribute>
    </complexType>
</element>
```

## Element: menu_separator

Properties

| | |
|---|---|
| **Name** | menu_separator |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Add a menu separator to a existent menu in the structure.

### Attributes:

- **tabdisplay:** Indicates the tabs on which an option can be displayed (i.e. `LIST` | `ALL` | `MAIN` ).

### Use Example:

Please refer to create_app_menu element at the use example section, for a complete menu example.

XML Instance Representation

```
<menu_separator
 tabdisplay="string (value comes from list: {'LIST'|'MAIN'|'ALL'})" [0..1]
/>
```

Schema Component Representation

```
<element name="menu_separator">
```

```
   <complexType>
      <attribute name="tabdisplay" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="LIST"/>
               <enumeration value="MAIN"/>
               <enumeration value="ALL"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: app_menu_header

Properties

| | |
|---|---|
| **Name** | app_menu_header |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines a header/Submenu for a menu application.

### Attributes:

- **headerdescription:** Description of the header of a sub menu that represents some group of apps.
- **image:** Application associated image/icon that will be shown at the UI.
- **tabdisplay:** Indicates the tabs on which an option can be displayed (i.e. `LIST` | `ALL` | `MAIN`).

### Use Example:

Please refer to create_app_menu element at the use example section, for a complete menu example.

## XML Instance Representation

```
<app_menu_header
 headerdescription="string" [1]
 image="string" [0..1]
 tabdisplay="string (value comes from list: {'LIST'|'MAIN'|'ALL'})" [0..1]
>
   Start Choice [1..*]
      <app_menu_option> ... </app_menu_option> [1]
      <menu_separator> ... </menu_separator> [1]
   End Choice
</app_menu_header>
```

## Schema Component Representation

```
<element name="app_menu_header">
   <complexType>
      <choice maxOccurs="unbounded">
         <element ref="app_menu_option"/>
         <element ref="menu_separator"/>
      </choice>
      <attribute name="headerdescription" type="string" use="required"/>
      <attribute name="image" type="string" use="optional"/>
      <attribute name="tabdisplay" use="default" value="ALL">
         <simpleType>
            <restriction base="string">
               <enumeration value="LIST"/>
               <enumeration value="MAIN"/>
               <enumeration value="ALL"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: additional_app_menu

### Properties

| | |
|---|---|
| **Name** | additional_app_menu |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

This statement adds more menu stuff to an existing application menu.

### Attributes:

- **app:** Application's unique name.
- **type:** Menu type (i.e. `Action`, `Tool` or `Search`).
- **menu_position:** Menu application position at the menu list.
- **pos_param:** Name of the app or optionname. Will be null if this is a header.

## Use Example:

For this example, a new menu type of an `Action` will be added for `NEWWRKPKG` application to the `WOTRACK` application.
<additional_app_menu app="WOTRACK" menu_position="after" pos_param="NEWWRKPKG" type="action" >

### XML Instance Representation

```
<additional_app_menu
 app="string" [1]
 type="string (value comes from list: {'action'|'tool'|'search'})" [0..1]
 menu_position="string (value comes from list: {'first'|'last'|'before'|'after'})" [0
..1]
 pos_param="string" [0..1]
>
   Start Choice [1..*]
      <app_menu_option> ... </app_menu_option> [1]
      <menu_separator> ... </menu_separator> [1]
      <app_menu_header> ... </app_menu_header> [1]
   End Choice
</additional_app_menu>
```

## Schema Component Representation

```
<element name="additional_app_menu">
   <complexType>
      <choice maxOccurs="unbounded">
         <element ref="app_menu_option"/>
         <element ref="menu_separator"/>
         <element ref="app_menu_header"/>
      </choice>
      <attribute name="app" type="string" use="required"/>
      <attribute name="type" use="default" value="action">
         <simpleType>
            <restriction base="string">
               <enumeration value="action"/>
               <enumeration value="tool"/>
               <enumeration value="search"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="menu_position" use="default" value="first">
         <simpleType>
            <restriction base="string">
               <enumeration value="first"/>
               <enumeration value="last"/>
               <enumeration value="before"/>
               <enumeration value="after"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="pos_param" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: add_service

Properties

| | |
|---|---|
| **Name** | add_service |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Add a service wich will bind through a RMI service remote and local requisitions.

### Attributes:

- **servicename:** Service's unique name.
- **description:** Description of the service purposes.
- **classname:** Full Qualifiend Name for the java class that implements the service.
- **singleton:** Specifies whether the session is singleton or can be different for tenant.

### Use Example:

The sample bellow indicates that is necessary add a service named `BASICAPPSERVICE` to bound the remote requisitions for the Basic Applications clients.

```
<add_service description="Service for Bacic Applications" classname="ibm.com.BasicApp
licationServer" servicename="BASICAPPSERVICE"/>
```

XML Instance Representation

```
<add_service
 servicename="string" [1]
 description="string" [1]
 classname="string" [1]
 singleton="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

Schema Component Representation

```
<element name="add_service">
   <complexType>
      <attribute name="servicename" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="classname" type="string" use="required"/>
      <attribute name="singleton" use="default" value="true">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: modify_service

Properties

| | |
|---|---|
| **Name** | modify_service |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Modifies a existent service.

### Attributes:

- **servicename:** Service's unique name.
- **description:** Description of the service purposes.

- **classname:** Full Qualifiend Name for the java class that implements the service.
- **singleton:** Specifies whether the session is singleton or can be different for tenant.

## Use Example:

The sample bellow modify the `BASICAPPSERVICE` service description.

```
<modify_service description="Bacic Applications Service" servicename="BASICAPPSERVICE
"/>
```

### XML Instance Representation

```
<modify_service
 servicename="string" [1]
 description="string" [0..1]
 classname="string" [0..1]
 singleton="string (value comes from list: {'true'|'false'})" [0..1]
/>
```

### Schema Component Representation

```
<element name="modify_service">
   <complexType>
      <attribute name="servicename" type="string" use="required"/>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="classname" type="string" use="optional"/>
      <attribute name="singleton" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: drop_service

Properties

| | |
|---|---|
| **Name** | drop_service |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Drop an existent service.

### Attributes:

- **servicename:** Service's unique name.

### Use Example:

The example bellow will drop the service created at add_service element named `BASICAPPSERVICE`.

```
<drop_service servicename="BASICAPPSERVICE" />
```

XML Instance Representation

```
<drop_service
 servicename="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_service">
   <complexType>
      <attribute name="servicename" type="string" use="required"/>
   </complexType>
</element>
```

## Element: add_property

## Properties

| | |
|---|---|
| **Name** | add_property |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

## Documentation

Add a system property to be configured through the UI.

## Attributes:

- **name:** Property's unique name.
- **description:** Brief description of this property.
- **maxtype:** Maximo type of this property (i.e. `ALN` | `INTEGER` | `YORN`).
- **domainid:** Associated domain to this property.
- **scope:** Represents the use scope of the property (i.e. `global`, `instance` or `open`).
- **secure_level:** Indicates the level of access permitted for this property (i.e. `private`, `public`, `secure` or `mtsecure`)
- **live_refresh:** Indicates whether live refresh of cache is supported for this property.
- **required:** When checked, indicates that a value is required for this property.
- **online_changes:** Indicates whether the user is allowed to override the value via the Property application.
- **user_defined:** Indicates whether this property was delivered with Maximo or was defined by a user.
- **default_value:** Property's default value.
- **encrypted:** Indicates whether the value of the property should be encrypted when stored on the MaxPropValue table.
- **masked:** Identified whether this value is masked in the System Properties application
- **value:** Property's value.
- **valuerules:** Value rules.
- **accesstype:** Access Type (i.e. master = **0**, landlord = **1**, delta = **2**, tenant = **3** )

## Use Example:

This sample defines a property that controls or 'allows' the user to see the logs from the server output.

```
  <add_property name="TESTON" maxtype="ALN" secure_level="private" description="Enabl
es test outputs for BasicApplication"/>
```

XML Instance Representation

```
<add_property
 name="string" [1]
 description="string" [1]
 maxtype="string (value comes from list: {'ALN'|'INTEGER'|'YORN'})" [1]
 domainid="string" [0..1]
 scope="string (value comes from list: {'global'|'instance'|'open'})" [0..1]
 secure_level="string (value comes from list: {'private'|'public'|'secure'|'mtsecure'
})" [1]
 live_refresh="string (value comes from list: {'true'|'false'})" [0..1]
 required="string (value comes from list: {'true'|'false'})" [0..1]
 online_changes="string (value comes from list: {'true'|'false'})" [0..1]
 user_defined="string (value comes from list: {'true'|'false'})" [0..1]
 default_value="string" [0..1]
 encrypted="string (value comes from list: {'true'|'false'})" [0..1]
 masked="string (value comes from list: {'true'|'false'})" [0..1]
 value="string" [0..1]
 valuerules="string (value comes from list: {'ONONLY'|'OFFONLY'|'INCONLY'|'DECONLY'|'
NONE'})" [0..1]
 accesstype="string (value comes from list: {'0'|'1'|'2'|'3'|'master'|'landlord'|'del
ta'|'tenant'})" [0..1]
/>
```

Schema Component Representation

```
<element name="add_property">
   <complexType>
      <attribute name="name" type="string" use="required"/>
      <attribute name="description" type="string" use="required"/>
      <attribute name="maxtype" use="required">
         <simpleType>
            <restriction base="string">
               <enumeration value="ALN"/>
```

```xml
            <enumeration value="INTEGER"/>
            <enumeration value="YORN"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="domainid" type="string" use="optional"/>
   <attribute name="scope" use="default" value="open">
      <simpleType>
         <restriction base="string">
            <enumeration value="global"/>
            <enumeration value="instance"/>
            <enumeration value="open"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="secure_level" use="required">
      <simpleType>
         <restriction base="string">
            <enumeration value="private"/>
            <enumeration value="public"/>
            <enumeration value="secure"/>
            <enumeration value="mtsecure"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="live_refresh" use="default" value="true">
      <simpleType>
         <restriction base="string">
            <enumeration value="true"/>
            <enumeration value="false"/>
         </restriction>
      </simpleType>
   </attribute>
   <attribute name="required" use="default" value="false">
```

```
        <simpleType>
           <restriction base="string">
              <enumeration value="true"/>
              <enumeration value="false"/>
           </restriction>
        </simpleType>
     </attribute>
     <attribute name="online_changes" use="default" value="true">
        <simpleType>
           <restriction base="string">
              <enumeration value="true"/>
              <enumeration value="false"/>
           </restriction>
        </simpleType>
     </attribute>
     <attribute name="user_defined" use="default" value="false">
        <simpleType>
           <restriction base="string">
              <enumeration value="true"/>
              <enumeration value="false"/>
           </restriction>
        </simpleType>
     </attribute>
     <attribute name="default_value" type="string" use="optional"/>
     <attribute name="encrypted" use="default" value="false">
        <simpleType>
           <restriction base="string">
              <enumeration value="true"/>
              <enumeration value="false"/>
           </restriction>
        </simpleType>
     </attribute>
     <attribute name="masked" use="default" value="false">
        <simpleType>
```

```
                <restriction base="string">
                    <enumeration value="true"/>
                    <enumeration value="false"/>
                </restriction>
            </simpleType>
        </attribute>
        <attribute name="value" type="string" use="optional"/>
        <attribute name="valuerules" use="optional">
            <simpleType>
                <restriction base="string">
                    <enumeration value="ONONLY"/>
                    <enumeration value="OFFONLY"/>
                    <enumeration value="INCONLY"/>
                    <enumeration value="DECONLY"/>
                    <enumeration value="NONE"/>
                </restriction>
            </simpleType>
        </attribute>
        <attribute name="accesstype" use="default" value="master">
            <simpleType>
                <restriction base="string">
                    <enumeration value="0"/>
                    <enumeration value="1"/>
                    <enumeration value="2"/>
                    <enumeration value="3"/>
                    <enumeration value="master"/>
                    <enumeration value="landlord"/>
                    <enumeration value="delta"/>
                    <enumeration value="tenant"/>
                </restriction>
            </simpleType>
        </attribute>
    </complexType>
```

```
</element>
```

## Element: set_property

### Properties

| | |
|---|---|
| **Name** | set_property |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Set a value to an existent property.

### Attributes:

- **name:** Property's unique name.
- **value:** New value for this property.

### Use Example:

The follow example set to **'ON'** the `TESTON` property added in add_property element's use example section.

```
<set_property name="TESTON" value="ON"/>
```

### XML Instance Representation

```
<set_property
 name="string" [1]
 value="string" [1]
/>
```

### Schema Component Representation

```
<element name="set_property">
   <complexType>
      <attribute name="name" type="string" use="required"/>
      <attribute name="value" type="string" use="required"/>
   </complexType>
</element>
```

## Element: drop_property

Properties

| | |
|---|---|
| **Name** | drop_property |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation
Drop an existent property

### Attributes:

- **name:** Property's unique name.

### Use Example:

The follow example will drop the `TESTON` created on add_property element usa example section.
<drop_property name="TESTON"/>

XML Instance Representation

```
<drop_property
 name="string" [1]
/>
```

Schema Component Representation

```
<element name="drop_property">
   <complexType>
      <attribute name="name" type="string" use="required"/>
   </complexType>
</element>
```

## Element: longdescription

Properties

| | |
|---|---|
| **Name** | longdescription |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Enable or disable a long description for an aln field.

### Use Example:

The modify_table element implements a sample with the long description field.

```
...
    >longdescription>The log description of some element goes here. </longdescription
>
...
```

XML Instance Representation

```
<longdescription/>
```

Schema Component Representation

```
<element name="longdescription">
   <complexType mixed="true"/>
</element>
```

## Element: insert

Properties

| | |
|---|---|
| **Name** | insert |

| | |
|---|---|
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Will insert a new row to a Table in Maximo.

## Attributes:

- **table:** Target table's name.
- **selectfrom:** Source data table's name.
- **selectwhere:** Where clause to list the data from source table.
- **ignore_duplicates:** If checked `TRUE` will allow the user to insert duplicate data.

## Use Example:

The follow example will insert a row into the SKDACTION table by change the attribute values `usewith` and `frame` and will copy the rest of columns based ont he 'selectwhere' attribute relationship.

```
<insert table="skdaction" ignore_duplicates="false" selectfrom="skdaction"

    selectwhere="where usewith='SCHEDULER'">

    <insertrow>

      <columnvalue column="skdobjectname" fromcolumn="skdobjectname"/>

      <columnvalue column="objectname" fromcolumn="objectname"/>

      <columnvalue column="appletactclass" fromcolumn="appletactclass"/>

      <columnvalue column="skdactclass" fromcolumn="skdactclass"/>

      <columnvalue column="title" fromcolumn="title"/>

      <columnvalue column="remark" fromcolumn="remark"/>

      <columnvalue column="actionname" fromcolumn="actionname"/>

      <columnvalue column="ismenubased" fromcolumn="ismenubased"/>

      <columnvalue column="multirec" fromcolumn="multirec"/>

      <columnvalue column="dlgname" fromcolumn="dlgname"/>

      <columnvalue column="menuorder" fromcolumn="menuorder"/>

      <columnvalue column="usewith" string="SCHEDACM"/>

      <columnvalue column="frame" string="A"/>

    </insertrow>
```

```
  </insert>
```

XML Instance Representation

```
<insert
 table="string" [1]
 selectfrom="string" [0..1]
 selectwhere="string" [0..1]
 ignore_duplicates="string (value comes from list: {'true'|'false'})" [0..1]
>
   <insertrow> ... </insertrow> [1..*]
</insert>
```

Schema Component Representation

```
<element name="insert">
   <complexType>
      <sequence>
         <element ref="insertrow" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="table" type="string" use="required"/>
      <attribute name="selectfrom" type="string" use="optional"/>
      <attribute name="selectwhere" type="string" use="optional"/>
      <attribute name="ignore_duplicates" use="default" value="false">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
   </complexType>
</element>
```

## Element: insertrow

### Properties

| | |
|---|---|
| **Name** | insertrow |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

### Documentation

Intert a row in an specif table through the insert element.

## Use Example:

For a complete example with the nested element columnvalue and other parameters, please refer to insert element.

### XML Instance Representation

```
<insertrow>
   <columnvalue> ... </columnvalue> [1..*]
</insertrow>
```

### Schema Component Representation

```
<element name="insertrow">
   <complexType>
      <sequence>
         <element ref="columnvalue" maxOccurs="unbounded"/>
      </sequence>
   </complexType>
</element>
```

## Element: columnvalue

### Properties

| | |
|---|---|
| **Name** | columnvalue |
| **Type** | Locally-defined complex type |

|  |  |
|---|---|
| **Nillable** | no |
| **Abstract** | no |

Documentation

Defines the parameter to insert a new row to into a table through the inser and insertrow elements.

## Attributes:

- **column:** Destination column.
- **string:** String value that will be set to the new row.
- **fromcolumn:** Source column name.
- **boolean:** Boolean value that will be set to the new row.
- **number:** Numeric value that will be set to the new row.
- **date:** Date value that will be set to the new row.
- **defaultsynonym:** Default domain that will be set to the new row.

## Use Example:

For a complete example with the nested element columnvalue and other parameters, please refer to insert element.

XML Instance Representation

```
<columnvalue
 column="string" [1]
 string="string" [0..1]
 fromcolumn="string" [0..1]
 boolean="string (value comes from list: {'true'|'false'})" [0..1]
 number="string" [0..1]
 date="string (value comes from list: {'sysdate'})" [0..1]
 defaultsynonym="string" [0..1]
/>
```

Schema Component Representation

```
<element name="columnvalue">
   <complexType>
      <attribute name="column" type="string" use="required"/>
      <attribute name="string" type="string" use="optional"/>
      <attribute name="fromcolumn" type="string" use="optional"/>
      <attribute name="boolean" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="true"/>
               <enumeration value="false"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="number" type="string" use="optional"/>
      <attribute name="date" use="optional">
         <simpleType>
            <restriction base="string">
               <enumeration value="sysdate"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="defaultsynonym" type="string" use="optional"/>
   </complexType>
</element>
```

## Element: logical_relationship

Properties

| | |
|---|---|
| **Name** | logical_relationship |
| **Type** | Locally-defined complex type |
| **Nillable** | no |
| **Abstract** | no |

Documentation

Persist an existent relationship in a logical relationship when necessary in a relational database. (Frequently used when huges ammount of data are deleted and the database clains for reorg and reindex commands from the database)

## Attributes:

- **object:** Table's name, to be used everytime the table needs to be referred.
- **keys:** Primary keys of the referred table.
- **targetobj:** Table's name of the target table in this relationship.
- **targetkeys:** Primary keys for the target table or foren keys in this relationship.
- **status:** The status quem be |unverified|verified|invalidated
- **description:** A brief description of what the relationship is capable of
- **number:** Specifies the cardinality of the relationship

## Use Example:

```
<logical_relationship status="verified" object="MAXVARS" keys="VARNAME" targetobj="MA
XVARTYPE" targetkeys="VARNAME"

        description="Type information for MaxVars" number="many to 1">
```

### XML Instance Representation

```
<logical_relationship
 object="string" [1]
 keys="string" [1]
 targetobj="string" [1]
 targetkeys="string" [1]
 status="string (value comes from list: {'unverified'|'verified'|'invalidated'})" [1]
 description="string" [0..1]
 number="string" [0..1]
>
   Start Sequence [0..1]
      <longdescription> ... </longdescription> [1]
   End Sequence
</logical_relationship>
```

### Schema Component Representation

```
<element name="logical_relationship">
   <complexType>
      <sequence minOccurs="0" maxOccurs="1">
         <element ref="longdescription"/>
      </sequence>
      <attribute name="object" type="string" use="required"/>
      <attribute name="keys" type="string" use="required"/>
      <attribute name="targetobj" type="string" use="required"/>
      <attribute name="targetkeys" type="string" use="required"/>
      <attribute name="status" use="required">
         <simpleType>
            <restriction base="string">
               <enumeration value="unverified"/>
               <enumeration value="verified"/>
               <enumeration value="invalidated"/>
            </restriction>
         </simpleType>
      </attribute>
      <attribute name="description" type="string" use="optional"/>
      <attribute name="number" type="string" use="optional"/>
   </complexType>
</element>
```