Custom Metadata Types Implementation Guide

Salesforce, Winter '17

@salesforcedocs

Last updated: December 9, 2016

CONTENTS

CUSTOM METADATA TYPES

You can create your own declarative developer frameworks for internal teams, partners, and customers. Rather than building apps from data, you can build apps that are defined and driven by their own types of metadata. Metadata is the information that describes the configuration of each customer's organization.

Custom metadata is customizable, deployable, packageable, and upgradeable application metadata. First, you create a *custom metadata type*, which defines the form of the application metadata. Then you build reusable functionality that determines the behavior based on metadata of that type. Similar to a custom object or custom setting, a custom metadata type has a list of custom fields that represent aspects of the metadata. After you create a public custom metadata type, you or others can declaratively create *custom metadata records* that are defined by that type. When you package a public custom metadata type, customers who install the package can add their own records to the metadata type. Your reusable functionality reads your custom metadata and uses it to produce customized application behavior.

Custom metadata rows resemble custom object rows in structure. You create, edit, and delete custom metadata rows in the Metadata API or in Setup. Because the records are metadata, you can migrate them using packages or Metadata API tools. Custom metadata records are read-only in Apex and in the Enterprise and Partner APIs.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

With custom metadata types, you can issue unlimited Salesforce Object Query Language (SOQL) queries for each Apex transaction.

Custom metadata types support the following custom field types.

- Metadata Relationship
- Checkbox
- Date
- Date and Time
- Email
- Number
- Percent
- Phone
- Picklist
- Text
- Text Area
- URL

A subscriber to a managed package containing a custom metadata type can't add their own fields to that type. Only the org that develops the type can add custom fields to it.

Custom metadata fields are *manageable*, which means that the developer of a type can decide who can change field values after they are deployed to a subscriber organization.

- Locked after release—For any record of the type, the value of the field is immutable after deployment, even on the developer organization where the record was created.

- Subscriber editable—Anyone with the correct permissions can change the value of the field at will. Any changes the developer deploys do not overwrite values in the subscriber's organization.
- Upgradable—The developer of a record can change the value of the field by releasing a new version of the custom metadata package. The subscriber can't change the value of the field.

Custom metadata types and records have names and labels. Type names must be unique within their namespace. Record names must be unique within their custom metadata type and namespace.

Custom metadata records can be protected. If a developer releases protected records in a managed package, access to them is limited in specific ways.

- Code that's in the same managed package as custom metadata records can read the records.
- Code that's in the same managed package as custom metadata types can read the records that belong to that type.
- Code that's in a managed package that doesn't contain either the type or the protected record can't read the protected records.
- Code that the subscriber creates and code that's in an unmanaged package can't read the protected records.
- The developer can modify protected records only with a package upgrade. The subscriber can't read or modify protected records. The developer name of a protected record can't be changed after release.

If you create a protected custom metadata record in your organization, then it's accessible only by your code, code from unmanaged packages, and code from the managed package that defines its type.

Custom metadata types can also be protected, providing the same access protection as protected records. If you change a type from protected to public, its protected records remain protected and all other records become public. If you use Setup to create a record on a protected type, the Protected Component checkbox is checked by default. Once a type is public, you can't convert it to protected. The subscriber can't create records of a protected type.

CUSTOM METADATA TYPES LIMITATIONS

When using custom metadata types, be aware of these special behaviors and limitations.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

Updating Types and Records

You can't update protected types and records in an installed managed package programmatically. You can modify protected types and records only by performing a package upgrade.

You can't update public types and records by using Apex directly. To modify records from Apex, you must make calls to the Metadata API.

Application lifecycle management tools

Custom metadata types don't support these application lifecycle management tools:

- Version control
- Tooling API
- Developer Console

Licenses

Licenses that are defined for an extension package aren't enforced on custom metadata records in that package unless the types are also in the package.

SOQL

Custom metadata types support the following SOQL query syntax.

```
SELECT fieldList [...]
FROM objectType
    [USING SCOPE filterScope]
[WHERE conditionExpression]
[ORDER BY field {ASC|DESC} [NULLS {FIRST|LAST}] ]
```

- You can use metadata relationship fields in the *fieldList* and *conditionExpression*.
- FROM can include only 1 object.
- You can use the following operators.
  - IN and NOT IN
  - =, >, >=, <, <=, and !=
  - LIKE, including wild cards
  - AND
- You can use ORDER BY only with non-relationship fields.
- You can use ORDER BY, ASC, and DESC with multiple (non-relationship) fields.
- You can only use ORDER BY when the ordered field is a selected field.
- Metadata relationship fields support all standard relationship queries.

Protected custom metadata types

Subscribers can't add custom metadata records to installed custom metadata types that are protected. To allow subscribers to create custom metadata records, the custom metadata type must be public.

Metadata API returns protected custom entity definitions (but not custom metadata records) in subscriber orgs.

Caching

Custom metadata records are cached at the type level after the first read request. Caching enhances performance on subsequent requests. Requests that are in flight when metadata is updated don't get the most recent metadata.

Global Picklists

Global picklists aren't supported on custom metadata types. You can only use sObject picklists.

Picklists and Managed Packages

- You can add a custom metadata type that has a picklist field with inactive values to a managed package, but you can't upload the package. To upload the package, delete or reactivate the picklist values.
- Subscribers to a released managed package that contains a custom metadata type with a picklist field can't add, delete, or deactivate values from that picklist.
- Developers who release a managed packaged that contains a custom metadata type with a picklist field can add picklist values but not delete or deactivate them.

CUSTOM METADATA LIMITS

Be aware of these requirements for custom metadata types and records.

Description	Maximum amount
SOQL queries per Apex transaction	Unlimited
Custom metadata per organization *	10 MB
Custom metadata per certified managed package *	10 MB Note: Custom metadata records in certified managed packages that you've installed don't count toward your organization's limit. However, custom metadata records that you create do count toward the limit. This rule applies regardless of whether you create records in your own custom metadata type or in a type from a certified managed package.
Fields per custom metadata type or record	100
Custom metadata types per organization	100. This number includes all types developed in the organization and installed from managed and unmanaged packages.
Characters per description field	1,000
Records returned per transaction	50,000

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

* Record size is based on the maximum field size of each field type, not the actual storage that's used in each field. When adding fields to a custom metadata record, use the appropriate type and specify a length that doesn't exceed what's needed for your data. This action helps you avoid reaching the cached data limit. For example, if you create a US social security number (SSN) field, select the `Text` data type and specify a length of 9. If instead you selected `Text Area`, the field would add 255 characters to the usage count for each record, regardless of the number of characters entered.

CREATE, EDIT, AND DELETE CUSTOM METADATA TYPES AND RECORDS

You can use Setup to create, update, and delete custom metadata types and records declaratively. Use the Metadata API to perform these tasks programmatically.

For more information about creating and managing custom metadata types programmatically, see "Custom Metadata Types (CustomObject)" in the *Metadata API Developer Guide*

Define a Custom Metadata Type Declaratively
Use the Salesforce UI to create and update custom metadata types declaratively.

Add or Edit Custom Metadata Records Declaratively
You can add, modify, or delete a custom metadata record declaratively from Setup.

Custom Metadata Relationships
Custom metadata relationships provide additional metadata about objects and let you make direct comparisons between different custom metadata types.

Load or Update Records with the Custom Metadata Loader
Use the custom metadata loader to bulk load or update records of your custom metadata types from a `.csv` file.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

Define a Custom Metadata Type Declaratively

Use the Salesforce UI to create and update custom metadata types declaratively.

1. Search Setup for **Custom Metadata Types**.
2. On the All Custom Metadata Types page, click **New Custom Metadata Type**, or click the Label name to modify an existing custom metadata type.
3. Complete these fields.

Field	Description
`Label`	This name refers to the type in a user interface page.
`Plural Label`	The plural name of the type. If you create a tab for this type, this name is used for the tab.
`Starts with a vowel sound`	If it is appropriate for your org's default language, indicate whether the label is preceded by "an" instead of "a."
`Object Name`	A unique name used to refer to the object when using the API. In managed packages, this name prevents naming conflicts with package installations. Use only alphanumeric characters and underscores. The name must begin with a letter and have no spaces. It cannot end with an underscore nor have 2 consecutive underscores.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

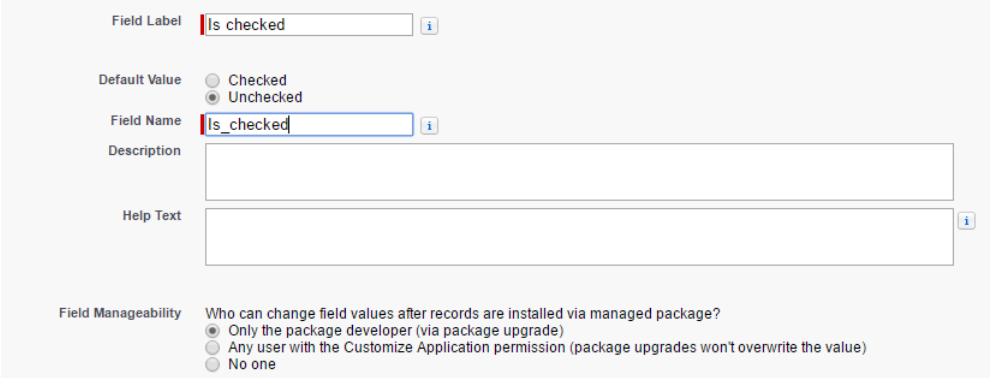
Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

USER PERMISSIONS

To create or edit custom metadata types:
- "Author Apex"

Field	Description
`Description`	An optional description of the object. A meaningful description helps you remember the differences between your custom objects when you are viewing them in a list.
`Visibility`	Who can see the type: • Public—Anyone can see it. • Protected—If the type is installed as part of a managed package, only Apex code in that managed package can use it.

4. Click **Save**.
5. Under Custom Fields, click **New** to start adding fields to the custom metadata type. As with other custom fields, you must specify the type of information that the field contains, such as picklist or metadata relationship. For each field, remember to choose a **Field Manageability** value to determine who can change the field later.



Note: Custom metadata types that were created before the Winter '15 release don't automatically get layouts. Before adding, updating, or viewing records of this custom metadata type using the UI, you must add a layout that contains all the fields that you want to make editable. In the All Custom Metadata Types page, click the custom metadata type. Then click **New** under Page Layouts. If you plan to release a custom metadata type as a managed package, make sure that you add all the fields you want to add first. After a customer downloads the managed package, you must change the layout manually because you can't add fields to a layout via an upgrade.

Add or Edit Custom Metadata Records Declaratively

You can add, modify, or delete a custom metadata record declaratively from Setup.

1. Search Setup for **Custom Metadata Types**.
2. On the All Custom Metadata Types page, click **Manage Records** next to the custom metadata type for which you want to add or modify records..
3. On the list of custom metadata records, click **New**, or click **Edit** to modify an existing custom metadata record.
4. Fill out the fields. The **Protected Component** checkbox determines whether the record is *protected*. A protected record is only accessible to code in the same namespace as either the record or its associated custom metadata type: code you create, code in an unmanaged package, and code in the same managed package as either the protected record or its custom metadata type.
5. Click **Save**.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

USER PERMISSIONS

To create or modify custom metadata records:

- "CustomizeApplication"

Custom Metadata Relationships

Custom metadata relationships provide additional metadata about objects and let you make direct comparisons between different custom metadata types.

Like other relationships in Salesforce, custom metadata relationships have a particular domain. When you create a metadata relationship field on a type, you can relate it to another custom metadata type or the EntityDefinition object.

When you create a record on a custom metadata type that has a relationship field, you pick the specific object or custom metadata type that record relates to.

The value of a relationship field with the EntityDefinition domain is custom or standard object that:

- Supports custom fields
- Supports Apex triggers
- Supports custom layouts
- Is not a type of activity, such as a Task or Event
- Is not the User object
- Is not a Trialforce object, such as a SignupRequest

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

To set the EntityDefinition object as a new value on a relationship field, your org must have access to the object. If, however, your org doesn't have access to relationship field objects for existing records, you can still edit the record and change other field values. Your org can lack access if, for example, the record is part of a package for which you don't have an active license.

Create Custom Metadata Relationship Fields

Creating relationships between custom metadata types or entity definitions is just like creating any other custom field on a custom metadata type.

1. From the detail page of your custom metadata type, click **New** under Custom Fields.
2. For the field type, select **Metadata Relationship**.
3. Select the custom metadata type that you want the active type to relate to, or pick Entity Definition to relate it to a supported standard or custom object.

You pick the specific custom metadata record or object when you create a record on your custom metadata type. After you save the record, the relationship field shows up as a link to the Setup page for the related custom metadata type or object. If you do not have permission to view the object, the field shows up as unlinked text.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

USER PERMISSIONS

To create custom metadata relationships:

- "Customize Application"

Custom Metadata Relationship Considerations

Before you start using custom metadata relationships, keep these considerations in mind.

- You can query custom metadata relationships the same way you query normal relationships.
- Public custom metadata types can't be related to protected custom metadata types. Protected custom metadata types *can* be related to public custom metadata types.
- Public custom metadata records can't be related to protected custom metadata records. Protected custom metadata records *can* be related to public custom metadata records.
- If you use SOQL to query a custom metadata type, the results include only those records that reference objects you have permission to access. However, a similar query using Setup or the Metadata API results in all relevant records, including records that reference objects you cannot access.

  For example, imagine a custom metadata type that has a relationship field that relates to the EntityDefinition object. In some of the custom metadata records, the relationship field references a Lead object. You have permission to access the custom metadata type and its records, but not the Lead object. Using Setup or the Metadata API to query the custom metadata type, the results include records that reference a Lead object. However, if you use a similar SOQL query in your Apex code, the results do not include those records.

  The difference in behavior is to allow you to manage the custom metadata type record without requiring explicit access to the object it references.
- You can't install a package that contains custom metadata type records whose relationship fields reference objects that your org can't access. The installation error message includes the list of objects to which you need access.
- You can install a package that contains custom objects for which you don't have an active license. However, those records do not appear in SOQL queries for any users until you acquire the license to the objects.
- If you don't have permission to view an object in Setup, relationship field values that reference that object appear as plain text rather than links.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

View Filtering on Metadata Relationship Fields

When you create a view and filter on a relationship field of a custom metadata type, use these guidelines for entering the filter values.

Setup doesn't provide a lookup window because the list of values is potentially long and unwieldy. Use the following guidelines to determine which values to enter when specifying the filter criteria in the Filter By Additional Fields section.

Filter by an EntityDefinition relationship field to find records that reference a particular object

1. Select the child's metadata relationship field.
2. Select the operator.
3. For the filter value, enter the object name of the referenced object. To find the object name of a custom object, navigate to its Setup management page. For a standard object, use its API name.

Filter by a relationship field to find records that reference a record of another custom metadata type

1. Select the child's metadata relationship field.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

2. Select the operator.
3. For the filter value, enter the name of the custom metadata type of the parent's record. To find the name of a custom metadata record, navigate to its detail page.

Load or Update Records with the Custom Metadata Loader

Use the custom metadata loader to bulk load or update records of your custom metadata types from a `.csv` file.

The custom metadata loader lets you load or update up to 200 records with a single call.

1. Download the tool from GitHub and deploy the package to your org via Workbench. Create the `.zip` file from the contents of the `custom_md_loader` directory instead of zipping up the directory itself.
2. Create a `.csv` file with a header that contains the custom metadata type's field API names. Either the Label or the Developer Name field is required. See `sample.csv` in your download for an example. If your org is namespaced, include the namespace prefix in your header. To update an existing custom metadata record, use the Label or Developer Name field to identify it.
3. From Setup, assign the Custom Metadata Loader permission set to the appropriate users, including yourself.
4. From the App Picker, select **Custom Metadata Loader**.
5. Click the **Custom Metadata Loader** tab. If you haven't already done so, the app prompts you to configure your Remote Site Settings.
6. Upload your `.csv` file and select the corresponding custom metadata type.
7. Click **Create/Update custom metadata** to bulk-load the records from the `.csv` file or update existing records. If the file has duplicate Label or Developer Name entries, the last entry becomes the new or updated record.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

USER PERMISSIONS

To create custom metadata records:

- "Customize Application"

To use the custom metadata loader:

- "Custom Metadata Loader"

ACCESS CUSTOM METADATA RECORDS PROGRAMMATICALLY

Use SOQL to access your custom metadata types and to retrieve the API names of the records of those types. DML operations aren't allowed on custom metadata in Apex, the Partner APIs, and Enterprise APIs.

For information about the *Custom Metadata Type*__mdt sObject, see *Custom Metadata Type*__mdt in the *Object Reference for Salesforce and Force.com*.

The following example declares the Apex variable *custMeta* of the custom metadata type `MyCustomMetadataType__mdt`, which is in your namespace.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

```
MyCustomMetadataType__mdt custMeta;
```

Declare the *custMeta* variable of the custom metadata type `TheirCustomMetadataType__mdt`, which isn't in your namespace but is in the `their_ns` namespace.

```
their_ns__TheirCustomMetadataType__mdt custMeta;
```

The following example is a simple query that returns standard and custom fields for all records of the `Threat_Tier_Mapping` custom metadata type and accesses some of their fields.

```
Threat_Tier_Mapping__mdt[] threatMappings = [SELECT MasterLabel, QualifiedApiName,
Field_Mapping__c ,Minimum_Support_Level__c FROM Threat_Tier_Mapping__mdt];

for (Threat_Tier_Mapping__mdt threatMapping : threatMappings) {
    System.debug(threatMapping.MasterLabel + ': ' +
                 threatMapping.Field_Mapping__c + ' from ' +
                 threatMapping.Team_Building_to_SFA_Field_Mapping__c + ' to '
                 threatMapping.Minimum_Support_Level__c);
}
```

To provide an entity that looks more like a `Schema.SObjectDescribeResult` than SOQL, make the Apex class `vacations.ThreatTierMappingDescribeResult` encapsulate the information queried from `vacations__ThreatTierMappingDescribeResult__mdt`. Then create the class `vacations.Vacations` with methods such as:

```
vacations.ThreatTierMappingDescribeResult describeThreatTierMappings(String qualifiedApiName)
 {
   Threat_Tier_Mapping__mdt threatMapping = [SELECT <fields> FROM Threat_Tier_Mapping__mdt
 WHERE QualifiedApiName = :qualifiedApiName];
```

```
    return new ThreatTierMappingDescribeResult(<fieldValues>);
}
```

In the preceding example, `<fields>` refers to the fields you want to include in the *describe* and `<fieldValues>` refers to the values of those fields.

The next example uses a metadata relationship that references another custom metadata type, `Team_Building_to_SFA_Field_Mapping__mdt`, to do a simple right outer join.

```
ThreatTierMapping threatMapping =
    [SELECT MasterLabel, Team_Building_to_SFA_Field_Mapping__r.MasterLabel FROM
Threat_Tier_Mapping__mdt WHERE QualifiedApiName='Easy_Vacations'];

System.debug(threatMapping.MasterLabel + ' is part of ' +
Team_Building_to_SFA_Field_Mapping__r.MasterLabel);
```

The following example shows a left outer join starting from `EntityDefinition`. This query uses a relationship field called `Team_Building_Object__c` on `Team_Building_to_SFA_Field_Mapping__mdt`. The child relationship name of this relationship field is `Field_Mappings_From`.

```
for (EntityDefinition entity : allObjects) {
    System.debug('Processing mappings for: ' + entity.QualifiedApiName);
   for (Team_Building_to_SFA_Field_Mapping__mdt fieldMapping : entity.FieldMappingsFrom__r)
 {
    System.debug('  Field ' + fieldMapping.Team_Building_Field__c +
        ' has mapping ' + fieldMapping.QualifiedApiName);
    }
}
```

PACKAGE CUSTOM METADATA TYPES AND RECORDS

You can package custom metadata types and records in unmanaged packages, managed packages, or managed package extensions. Your packages can then be installed in Professional, Developer, Enterprise, Performance, Unlimited, and Database.com Edition organizations.

You can add custom metadata types and records to packages using the Force.com user interface. From Setup, enter *Packages* in the `Quick Find` box, then select **Packages**, click your package name, and then click **Add**.

To add custom metadata types:

1. Select the **Custom Metadata Type** component type.
2. Select the custom metadata type you want to add to your package.
3. Click **Add to Package**.

To add custom metadata records:

1. Select the custom metadata type's label from the available component types—for example, `Threat Tier`, or if the type is from a package that you're extending, `Threat Tier [vacations]`.
2. Select the records to add.
3. Click **Add to Package**.

If you add a record to your package, its corresponding type is automatically added.

For information on packaging and installing, see the *ISVforce Guide*.

Note: You can't uninstall a package with a custom metadata type if you've created your own records of that custom metadata type.

As with all packageable metadata components, you can also add custom metadata types and records to a package by specifying the package's full name in `package.xml`. For example, we specify the package in this fragment from Relaxation Gauntlet's `package.xml` file.

```
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
   <fullName>Relaxation Gauntlet</fullName>
...
```

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

Access Rules When Packaging Custom Metadata Types and Records

When you develop a managed package that contains or reads custom metadata types and records, be aware of the access rules.

Considerations for Custom Metadata Type Packages

Be aware of the behaviors for packages that contain custom metadata types.

Access Rules When Packaging Custom Metadata Types and Records

When you develop a managed package that contains or reads custom metadata types and records, be aware of the access rules.

The packaging access rules govern which orgs can read, update, or delete custom metadata records. It is assumed that the custom metadata types are public.

- An admin in the org developing the package can create a custom metadata record in their own package regardless of the location of its corresponding type. If they add the new record to the package, it's deployed to the subscriber org.
- No one can create a custom metadata record in an installed managed package using the Metadata API. You can, however, create an unpackaged record using a Metadata API callout, even from managed code. Managed installed code needs a remote site setting configured to execute all callouts.
- If a field of a custom metadata type is *upgradeable*, the record creator can change the field value for that record in the creator's own org and upload a new version of the package, even if a different org created the type. If the record is in a managed package, these changes are propagated to the subscriber org when they upgrade to a new version.
- If a field is *subscriber controlled*, both the record creator and a subscriber can change the value in their own org. If the record is in a managed package, the new field value is propagated only to new package subscribers. Existing subscribers that upgrade to the latest version of the package do not get the new field value.
- You can delete *protected* managed released records in the org in which they were created, even if the corresponding type was created in a different org. When subscribers upgrade, the records are deleted from the subscriber org. You can never delete *public* managed released records.
- SOQL queries in your Apex code can view custom metadata records only if exactly one of the following conditions is true.
  - The records are public.
  - Your Apex code is in the same package as the custom metadata type.
  - Your Apex code is in the same package as the record.
- Metadata API callouts behave as if they're being executed by subscriber org code. As a result, a callout can view or change all records created by the subscriber org, although it can only view or change public records of installed managed packages. Configure a remote site setting to the subscriber's Metadata API endpoint to use the Metadata API in the subscriber's org.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

Considerations for Custom Metadata Type Packages

Be aware of the behaviors for packages that contain custom metadata types.

After you upload a Managed - Released package that contains a custom metadata type, you can't:

- Add required fields to the custom metadata type.
- Set non-required fields to required.
- Delete custom fields that are in the uploaded version of the package. If you add a custom field after the upload, you can still delete it until you upload a new Managed - Released version.
- Delete public custom metadata records that are included in the package.
- Change a public custom metadata record or type in the package to protected. You can, however, change protected records and types to public.
- Change the manageability of any custom field in the package.
- Query protected custom metadata types or records in an installed package with Apex code that you have written. You can, however, run packaged Apex code that queries protected custom metadata types or records in the same package. These queries return the protected records.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.

DEPLOY CUSTOM METADATA TYPES AND RECORDS TO PRODUCTION ORGS USING CHANGE SETS

Use change sets to deploy custom metadata types and records from a sandbox to another org. Typically you deploy the change set to a production org.

You can add custom metadata types and records to change sets using the Force.com user interface. From Setup, enter *Outbound Change Sets* in the Quick Find box, then select **Outbound Change Sets**, click your change set name, and then click **Add**.

To add custom metadata types:

1. Select the **Custom Metadata Type** component type.
2. Select the custom metadata type you want to add to your outbound change set.
3. Click **Add to Change Set**.
4. To view the dependent components, such as a custom field or a page layout, click **View/Add Dependencies**.
5. Select the dependent components you want to add.
6. Click **Add to Change Set**.

To add custom metadata records:

1. Select the custom metadata type's label from the available component types, for example, `Threat Tier`. If the type is from a package that you're extending, use `Threat Tier [vacations]`.
2. Select the records to add.
3. Click **Add to Change Set**.

If you add a record to a change set, its corresponding type is included in the list of dependent components. The record itself is not added to the change set.

For more information on deploying change sets, see the *Development Lifecycle Guide*.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Professional Edition orgs can create, edit, and delete custom metadata records only from types in installed packages.