Custom Metadata Types Implementation Guide

Salesforce, Spring '24

@salesforcedocs

Last updated: December 29, 2023

CONTENTS

Custom Metadata Types 1

Custom Metadata Types Limitations 2

Custom Metadata Allocations and Usage Calculations 5

Create, Edit, and Delete Custom Metadata Types and Records 7

Add or Edit a Custom Metadata Type Declaratively 7
Add or Edit Custom Metadata Records Declaratively 9
Custom Metadata Relationships 10
- Create a Relationship to a Custom Metadata Type or Entity Definition 10
- Create a Relationship to a Field Definition or Entity Particle 11
- Custom Metadata Relationship Considerations 11
- View Filtering on Metadata Relationship Fields 13

Create and Manage Custom Metadata Types Using CLI Commands 14

Access Custom Metadata Records Programmatically 17

Control Read Access to Custom Metadata Types 19

Package Custom Metadata Types and Records 20

Add Custom Metadata Types 20
Add Custom Metadata Records 21
Access Rules When Packaging Custom Metadata Types and Records 21
Considerations for Custom Metadata Type Packages 23

Deploy Custom Metadata Types and Records to Production Orgs Using Change Sets 24

Add Custom Metadata Types 24
Add Custom Metadata Records 24

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.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

CUSTOM METADATA TYPES LIMITATIONS

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

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

Updating Types and Records

If your Apex code is in the same namespace as either protected metadata types or their records, you can update the types and records in an installed managed package programmatically.

To modify records from Apex, you must use the Metadata package in Apex.

If you delete a protected custom metadata type record that was part of a released package, you can't create another record with that name again.

Application Lifecycle Management Tools

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

- 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

  - `OR` when on the same column with `LIKE` and `=` operations

    Note: `OR` can't be used as a compound filter when child filters are on two different columns.

- You can use `ORDER BY` only with non-relationship fields.
- You can use `ORDER BY`, `ASC`, and `DESC` with multiple (non-relationship) fields.
- 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 package that contains a custom metadata type with a picklist field can add picklist values but not delete or deactivate them.

Custom Metadata Type Records and Managed Packages

- After you remove the records from a package, if those records are visible to the subscriber org, they're marked as deprecated. If the records aren't visible to the subscriber org, they're deleted.
- Deprecated records in a subscriber org count towards that org's custom metadata limits. When a subscriber org reaches its custom metadata record limit, future package installs or upgrades could be blocked until deprecated records are deleted by the subscriber.

Shield Platform Encryption

Custom Metadata Types don't support Shield Platform Encryption for fields.

Validation Rules

- You can reference up to 15 unique custom metadata types in all validation rules per entity. For example, from all validation rule formulas combined for a specified object, you can reference up to 15 different custom metadata types. However, you can include more than 15 references to the same custom metadata type in your validation rules.
- During deployment, custom metadata records that are included in the package are locked in order to maintain referential integrity. Other processes must wait until the deployment completes and the records are no longer locked before the same records can be updated. Validation rules that read custom metadata types can also create locks. For example, validation rule formulas that reference a custom metadata type record can create a read lock on the record. Because of the locks created during deployment, row lock errors can occur. If you encounter row locks, Salesforce recommends deploying during off-peak hours.

Formula Fields

Spanning custom metadata type relationships isn't supported in formula fields. Use Apex to avoid spanning relationships.

Permission Set Muting

Custom Metadata Types don't support permission set muting.

Workflow Rules

Custom metadata types that contain fields that are associated with workflow field updates aren't supported.

CUSTOM METADATA ALLOCATIONS AND USAGE CALCULATIONS

Understand requirements for custom metadata types and records and how your custom metadata type usage is calculated.

Tip: View custom metadata type use in System Overview. From Setup, in the Quick Find box, enter *System Overview*, and then select **System Overview**. You can get information about the number of custom metadata types and the size of the custom metadata type records used.

Description	Maximum amount
SOQL queries per Apex transaction	Unlimited
SOQL queries for custom metadata type records in flows	Count toward Apex governor limits. Per-transaction limits, which Apex enforces, govern flows.
SOQL queries containing long text area fields	Count toward Apex governor limits.
Custom metadata per organization *	10 million characters
Custom metadata per certified managed package *	10 million characters Custom metadata records in certified managed packages that you installed don't count toward your org's allotment. However, if the package developer removes the records from the managed package, the deprecated records remain in your org and count against your org's allotment until you delete them. Custom metadata records that you create also count toward your org's allotment. 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	200. This number includes all types developed in the org and installed from managed and unmanaged packages.
Characters per description field	1,000
Records returned per transaction	50,000
Long Text Area Fields	Long text area fields count toward the custom metadata 10 million characters allowed. 255

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

Description	Maximum amount
	characters per long text area field count for a given type.
Custom Metadata Types formula references allowed in a process (Process Builder)	15

*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 maximum. For example, if you create a US social security number (SSN) field, select the **Text** data type, and specify a length of 9. If you select **Text Area**, the field adds 255 characters to the usage count for each record, regardless of the number of characters entered.

Usage Calculation

- Usage is calculated in *characters*. You can store up to 10 million characters.
- Standard fields such as Label, Name, and Namespace are included in your usage calculation, but Description and Qualified API Name aren't.
- Long text area fields, up to 255 characters per long text area field for a given type, are included in the usage calculation.
- Metadata relationship fields count as 15 characters in the usage calculation if their target is another custom metadata type, or 10 characters if the target is Entity Definition or Field Definition.
- Picklists and checkboxes count as 10 characters.

Retrieving Custom Metadata Type Information

When you retrieve custom metadata type records or a count of custom metadata types, the number returned by SOQL, Apex, formulas, flows, or APIs can differ from the number in the user interface. This difference is because of the custom metadata type visibility settings and the access a user has to the type based on profiles and permissions.

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

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

Add or Edit a Custom Metadata Type Declaratively
You can declaratively create and update custom metadata types.

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

Custom Metadata Relationships
Create relationships between custom metadata types, entity definitions, field definitions, or entity particles. Use relationships rather than text fields to directly reference objects, simplify your Apex code, and enforce referential integrity, for example, when packaging custom metadata types.

Create and Manage Custom Metadata Types Using CLI Commands
You can use the Salesforce command-line interface (CLI) to create custom metadata types, generate fields, create records, create records from a CSV file, and generate custom metadata types from an sObject.

Add or Edit a Custom Metadata Type Declaratively

You can declaratively create and update custom metadata types.

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.

1. From Setup, in the Quick Find box, enter `Custom Metadata Types` and then select **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	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, Plural Label is used.
Starts with a vowel sound	Indicates whether "an" precedes the label rather than "a" (only when applicable for your org's default language).

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

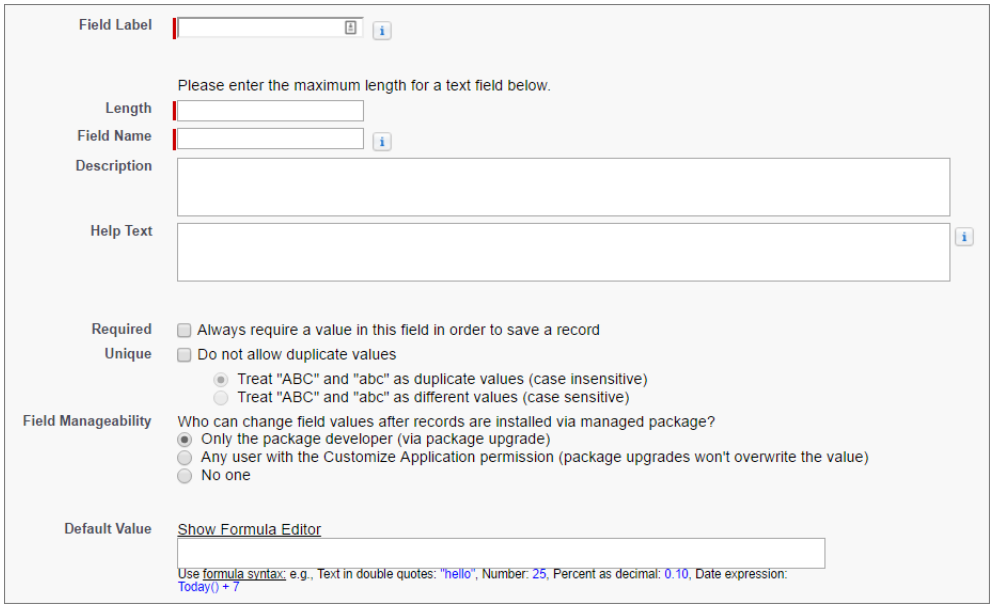
You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To create or edit custom metadata types:
- Customize Application

Field	Description
Object Name	A unique name 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 can't end with an underscore or contain two consecutive underscores.
Description	An optional description of the object. A meaningful description helps you understand the differences between your custom objects when you view them in a list.
Visibility	Who can see the type. • (Public) Regardless of the type of package (managed or unmanaged), these have access: – Apex – Formulas – Flows – API for users with Customize Application permission or permissions granted through profiles or permission sets. The custom metadata type, fields, and unprotected records are visible in Setup. • (Protected) Only Apex code in the same namespace can see the type. The name of the type and the record are visible if they're referenced in a formula. • (PackageProtected) When in a second-generation managed package, only Apex code in the same managed package can see the type. The name of the type and the record are visible if they're referenced in a formula.

4. Click **Save**.
5. Under Custom Fields, click **New** to start adding fields to the custom metadata type. Specify the type of information that the field contains, such as a picklist or metadata relationship. For each field, choose a Field Manageability value to determine who can change the field later.
   - If FieldType is `MetadataRelationship` and the manageability of the entity definition field is subscriber-controlled, the Field Definition field must be subscriber-controlled.
   - If the manageability of the entity definition field is `upgradeable`, the Field Definition field must be either upgradeable or subscriber-controlled.



Custom metadata types 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.

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.
5. The **Protected Component** checkbox determines whether the record is *protected*.

   When a custom metadata type is released in a managed package, access 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 with a package upgrade or by using the Metadata Apex classes (if the Apex code is in the same namespace as either the records or their type).
   - The subscriber can't read or modify protected records. The developer name of a protected record can't be changed after release.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To create or modify custom metadata records:

- Customize Application

- The subscriber can't create records of a protected type.

Records that are hidden by these access rules are also unavailable to REST, SOAP, SOQL, and Setup.

6. Click **Save**.

Custom Metadata Relationships

Create relationships between custom metadata types, entity definitions, field definitions, or entity particles. Use relationships rather than text fields to directly reference objects, simplify your Apex code, and enforce referential integrity, for example, when packaging 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, an entity definition, a field definition, or an entity particle.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

Create a Relationship to a Custom Metadata Type or Entity Definition
Create direct relationships to a custom metadata type or entity definition.

Create a Relationship to a Field Definition or Entity Particle
Create direct relationships to field definitions or entity particles.

Custom Metadata Relationship Considerations
Before you start using custom metadata relationships, keep these considerations in mind.

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.

Create a Relationship to a Custom Metadata Type or Entity Definition

Create direct relationships to a custom metadata type or entity definition.

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 either:
   - Another custom metadata type that you want to be the child of the active custom metadata type.
   - **Entity Definition** (Represents the metadata of a standard or custom object)

   Public custom metadata types can't be related to protected custom metadata types.
4. Create a record for storing the metadata relationship.
   - If the **Required** box is checked, you can't create a record for the parent types unless the child type has at least one record. A link to the records of the child custom metadata types is available from the parent custom metadata types record.
   - For entity definitions, select the standard or custom object that represents the entity definition.

You can now query your custom metadata type or entity definition using Apex.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To create custom metadata relationships:
- Customize Application

Create a Relationship to a Field Definition or Entity Particle

Create direct relationships to field definitions or entity particles.

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 **Entity Definition**.

   Entity definition represents the metadata of the standard or custom object that the field definition or entity particle are components of.

   Note: Public custom metadata types can't be related to protected custom metadata types.
4. From the detail page of your custom metadata type, create another custom field.
5. For the data type, select **Metadata Relationship**.
6. Select either:
   - **Field Definition**—A standard or custom field from the entity definition object.
   - **Entity Particle**—A compound value of a standard field or a geolocation field from the entity definition object.
7. On the detail page of the custom field, select a controlling field. The controlling field is the entity definition that controls the field definition or entity particle.
8. Create a record for storing the metadata relationship and select the field definitions or entity particles that you want to include in the record.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To create custom metadata relationships:
- Customize Application

You can now query your custom metadata type or entity definition using Apex.

You can access the field in the CustomField object in the Metadata API. For example, if you create a field definition named SFA_Field, you can access it when viewing CustomField details in a tool such as the Salesforce CLI.

Tip: To access relationship fields with Apex, you can use the `QualifiedApiName` field in the EntityDefinition tooling API object.

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.
- You need the Customize Application permission to view custom metadata type records.
- You can't relate public custom metadata types to protected custom metadata types. Protected custom metadata types *can* be related to public custom metadata types.
- 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.
- 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.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

- 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.
- To set the EntityDefinition object as a new value on a relationship field, your org must be able to access the object. However, if your org can't 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.
- A relationship field with the EntityDefinition domain can be a custom or standard object. The following rules apply to the metadata relationship field type. The entity:
  - Must be publicly exposed
  - Can be queried using the API
  - Supports Apex triggers
  - Is customizable
  - Supports layouts
  - Is not part of a union, such as a task, activity, event, holiday
  - Is not a setup entity, such as a permission set or a user
- Unsupported values for a relationship field with the EntityDefinition domain are:
  - A type of activity, such as a Task or Event
  - A Trialforce object, such as a SignupRequest
  - sObjects:
    - FieldPermissions
    - Group
    - GroupMember
    - ObjectPermissions
    - PermissionSet
    - PermissionSetAssignment
    - QueueSObject
    - ObjectTerritory2AssignmentRule
    - ObjectTerritory2AssignmentRuleItem
    - RuleTerritory2Association
    - SetupEntityAccess
    - Territory2
    - Territory2Model
    - UserTerritory2Association
    - User
    - UserRole
    - UserTerritory
    - Territory

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.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

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 FieldDefinition Relationship Field to Find Records that Reference a Particular Field

1. Select the child's metadata relationship field.
2. Select the operator.
3. For the filter value, enter the field name of the referenced field. To find the field name of a custom field, navigate to its Setup management page.
4. Specify a separate, additional filter criteria for the controlling entity definition. Both filters are required when filtering on a field definition relationship field.

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.
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.

Create and Manage Custom Metadata Types Using CLI Commands

You can use the Salesforce command-line interface (CLI) to create custom metadata types, generate fields, create records, create records from a CSV file, and generate custom metadata types from an sObject.

Support for custom metadata types is available in the Salesforce CLI plugin version 49.0. See the *Salesforce CLI Setup Guide* for information about how to set up CLI, set up a developer hub, create a project, and create a scratch org.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To run custom metadata types CLI commands:
- Customize Application

Commands

The following commands are available to create and manage custom metadata types. For flags and usage information, use the `--help` flag. For example, `sf cmdt generate records --help`.

- Create a custom metadata type.

  ```
  sf cmdt generate object
  ```

- Generate a custom metadata field based on the specified field type. You can create fields within the metadata object folder or passed in the directory of the object folder.

  ```
  sf cmdt generate field
  ```

- Generate a custom metadata type and all its records for an sObject. Use this command to migrate existing custom objects or custom settings to custom metadata types. The default directory is `force-app/main/default/customMetadata`.

  ```
  sf cmdt generate fromorg
  ```

  Note: Custom Settings of type hierarchy are not supported.

- Create a record for a specified custom metadata type.

  ```
  sf cmdt generate record
  ```

- Insert new custom metadata type records from a CSV file.

  ```
  sf cmdt generate records
  ```

Considerations

- Specify the object folder when creating custom metadata types or fields. For example, `--output-directory force-app/main/dirObjects/Mycmdt`.
- Specify unique names when creating custom metadata types.
- There are no restrictions on the number of records that can be inserted. When inserting a large number of records, be aware that the `project deploy start` command defaults to 33 minutes. The default is the number of minutes the command waits to complete and display results to the terminal window.
- When using the `cmdt generate records` command, the `DeveloperName` identifier defaults to the column `Name` and is a required column. However, any column name can be specified by using the `--name-column` flag. `Label` is not supported as an alternative identifier.

- The `cmdt generate records` command can be used to create new custom metadata types records or update existing custom metadata records.

Example: Create a custom metadata type that is protected along with the field types percent and checkbox. The metadata XML is created in the local directory for your SDFX project.

```
sf cmdt generate object --type-name Mycmdt --visibility Protected --output-directory
force-app/main/dirObjects
sf cmdt generate field --name Checkbox --type Checkbox --output-directory
force-app/main/dirObjects/Mycmdt
sf cmdt generate field --name Percent --type Percent --output-directory
force-app/main/dirObjects/Mycmdt
```

Example: Generate a custom metadata type from a custom object. Use this command to migrate an existing custom object to a new custom metadata type.

```
sf cmdt generate fromorg --dev-name FromCustomObject --sobject MyCustomObject__c
```

Example: Insert records into an existing custom metadata type from a CSV file.

Create a CSV file and provide the API name of the custom metadata type in the insert command. For example,

Name	CountryCode__c	CountryName__c
Australia	AU	Australia
Brazil	BZ	Brazil
Canada	CA	Canada

```
sf cmdt generate records --csv ~/Downloads/CMT_CSV_country.csv --type-name CmdtCountry
```

Migrating Custom Settings and Custom Objects to Custom Metadata Types

When converting sObjects to a custom metadata type, unsupported object types are converted to a string format.

Table 1: Mapping

sObject Type	Conversion Type
Auto-Number	Text field
Formula	Converted based on formula return type. If it is of text type converting it into a long text area with default length of 32768
Lookup	Text field
Roll-Up summary	Text field
External lookup	Text field
Master detail	Text field
Encrypted text	Unreadable text string.

sObject Type	Conversion Type
Geolocation	Two separate text fields representing the latitude and longitude
Multi-select picklist	Text field
Time	Text field
Currency	Text field

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.

Apex code can create, read, and update (but not delete) custom metadata records, as long as the metadata is subscriber-controlled and visible from within the code's namespace. DML operations aren't allowed on custom metadata in the Partner or Enterprise APIs. With unpackaged metadata, both developer-controlled and subscriber-controlled access behave the same: like subscriber-controlled access. For information about the *Custom Metadata Type*__mdt sObject, see *Custom Metadata Type*__mdt in the *Object Reference for Salesforce*. Refer to Trust, but Verify: Apex Metadata API and Security to learn more about package access in developer-controlled and subscriber-controlled orgs.

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

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

USER PERMISSIONS

To create or edit custom metadata types:

- Author Apex

```
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);
    }
}
```

CONTROL READ ACCESS TO CUSTOM METADATA TYPES

Admins with the Customize Application permission can grant Read access to specific custom metadata types through profiles and permission sets.

1. From Setup, enter *Schema Settings* in the Quick Find box, and make sure that the Restrict access to custom metadata types org permission is enabled.
2. Enter *User Management Settings* in the Quick Find box, and enable **Enhanced Profile User Interface**.

   This setting provides a uniform and streamlined interface, but isn't a requirement for granting permissions.
3. Enter *Profiles* or *Permission Sets* in the Quick Find box.
4. Click the name of the profile or permission set that you want to edit.
5. Click **Custom Metadata Types**.
6. Click **Edit**.
7. Select the custom metadata types that you want to grant access to, and add them to the Enabled Custom Metadata Types list.
8. Click **Save**.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Protected custom metadata types in managed packages are available in: **Developer** Edition and scratch orgs

Package uploads and installs are available in **Group**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Create, edit, and delete custom metadata type records from installed packages **Group** and **Professional** Editions

USER PERMISSIONS

To grant access to custom metadata types:

- Customize Application

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 Lightning Platform user interface. From Setup, enter *Packages* in the Quick Find box, then select **Packages**, click your package name, and then click **Add**.

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.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Protected custom metadata types in managed packages are available in: **Developer** Edition and scratch orgs

Package uploads and installs are available in **Group**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Create, edit, and delete custom metadata type records from installed packages **Group** and **Professional** Editions

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

Access Rules When Packaging Custom Metadata Types and Records

Understand the access rules when you develop a managed package that contains or reads custom metadata types and records.

Considerations for Custom Metadata Type Packages

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

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**.

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 Components Available in Managed Packages.

Access Rules When Packaging Custom Metadata Types and Records

Understand the access rules when you develop a managed package that contains or reads custom metadata types and records.

When you create a custom metadata type, the package type and the Visibility field determine whether the custom metadata type is public or private. You can only create protected custom metadata types in a developer or scratch org that are then deployed in a managed package.

When a custom metadata type is package-level protected using 2GP, records are only accessible from code within that managed package. Also the subscriber, and other packages, even within the same namespace, can't access the custom metadata type or its records. A 2GP can only be created through the Salesforce DX command-line interface (SFDX CLI).

To enable package-level protection for a custom metadata type, set the `Visibility` field to `PackageProtected` declaratively, or using metadata API.

When a custom metadata type is namespace protected, code that's in the same namespace as the custom metadata types can read the records. Code that's in a namespace that doesn't contain either the type or the protected record can't read the protected records. To set the accessibility of a package as namespace protected, set the `Visibility` field to `Protected` declaratively, or using metadata API.

Warning: Protected custom metadata types behave like public custom metadata types when they're outside of a managed package. Public custom metadata types are readable for all profiles, including the guest user. Don't store secrets, personally identifying information, or any private data in these records. Use protected custom metadata types only in managed packages. Outside of a managed package, use named credentials or encrypted custom fields to store secrets like OAuth tokens, passwords, and other confidential material.

When a type is public, you can't convert it to protected. The subscriber can't create records of a protected type.

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, **Protected Component** is selected by default.

After a managed package is released, subsequent versions of the package can be changed to a less restrictive protection level. For example, a package protected custom metadata type can be re-released as namespace protected. However, you can't change the protection level to be more restrictive after it has been released in a managed package.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Protected custom metadata types in managed packages are available in: **Developer** Edition and scratch orgs

Package uploads and installs are available in **Group**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Create, edit, and delete custom metadata type records from installed packages **Group** and **Professional** Editions

Entity	Accessibility
Package Creator Org	• Admins in the org developing the package can create a custom metadata record in their own package, regardless of the location of the record's corresponding type. If an admin adds the record to the package, the record is deployed to the subscriber org. • Package creator orgs 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.
Metadata API Callout	Metadata API callouts behave as if they're executed by the subscriber org code. As a result, someone can use a callout to view or change all records created by the subscriber org. However, the callout is used only to view or change the 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.
Metadata in Apex	Metadata in Apex callouts behave as if they're executed by subscriber org code. As a result, someone can use a callout to view or change all records created by the subscriber org. The callout can be used to view or change the public and protected records of installed managed packages.
Record Creator	• When you create a protected custom metadata record in your org, only your code, code from unmanaged packages, and code from the same namespace can access the record. • Record creators can 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. However, creators and subscribers can't create a custom metadata record in an installed managed package using the Metadata API. • If a field of a custom metadata type is upgradeable, the record creator can change the record's field value in the creator's own org. Then, the record creator can 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 the org upgrades to a new version. • If a field is subscriber controlled, subscribers can also 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 don't get the new field value.
Subscriber Org	If a field is subscriber controlled, subscribers can also change the value in their own org. If the record is in a managed package, the new field value is propagated only to Subscriber Org new package subscribers. Existing subscribers that upgrade to the latest version of the package don't get the new field value.
SQL Queries in Apex	You can use SOQL queries in your Apex code to view a custom metadata record only if at least one of the following conditions is true. • The record is 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.

Considerations for Custom Metadata Type Packages

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

Note: We recommend that you keep sensitive custom metadata types such as secrets, personally identifying information, or any private data in their own managed package and set their Visibility to package protected for security.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Protected custom metadata types in managed packages are available in: **Developer** Edition and scratch orgs

Package uploads and installs are available in **Group**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

Create, edit, and delete custom metadata type records from installed packages **Group** and **Professional** Editions

Types and Records

After you upload a Managed - Released package that contains a custom metadata type, you can't change a public custom metadata record or type in the package to protected. But you can change protected records and types to public, or change package protected types to namespace protected.

Fields

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.
- Change the manageability of any custom field in the package.

Visibility

- Custom metadata types with Visibility set to protected aren't visible to flows in a subscriber org.
- If you increase the visibility of a custom metadata type, you can't later decrease the visibility. For example, if you change the visibility from **Only Apex code in the same managed package can see the type** to **All Apex code and APIs can use the type, and it's visible in Setup**, you can't change it later.

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 Lightning Platform 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**.

EDITIONS

Available in: Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

You can create, edit, and delete custom metadata type records from installed packages in: **Group** and **Professional** Editions

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**.

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. If you add a type to a change set, its records are not automatically included in the list of dependent components.

For more information on using change sets, see the Change Set Development Model module on Trailhead.