**Data Security Topics:**

* Control Access to the Org
* Control Access to Objects
* Control Access to Fields
* Control Access to Records
* Create a Role Hierarchy
* Define Sharing Rules

**Overview of Data Security**

**Learning Objectives**

After completing this unit, you'll be able to:

* Explain the importance of giving the right people access to the right data.
* List the four levels at which you can control data access.
* Describe a typical scenario for limiting data access at each of the four levels.

**Introduction**

Choosing the data set each user or group of users can see is one of the key decisions that affects the security of your Salesforce org or app. Once you’ve designed and implemented your data model, give some thought to the kinds of things your users are doing and the data they need to do it.

Let’s say you’re building a recruiting app to help manage open positions, candidates, and job applications. You’ll have to store confidential data, such as social security numbers, salary amounts, and applicant reviews, that only some types of users should see. You’ll want to secure the sensitive data without making life harder for recruiters, hiring managers, and interviewers.

With the Salesforce platform’s flexible, layered sharing model, it’s easy to assign different data sets to different sets of users. You can balance security and convenience, reduce the risk of stolen or misused data, and still make sure all users can easily get the data they need.

The platform makes it easy to specify which users can view, create, edit, or delete any record or field in the app. You can control access to your whole org, a specific object, a specific field, or even an individual record. By combining security controls at different levels, you can provide just the right level of data access to thousands of users without having to specify permissions for each user individually.

Note

**Note**

Although you can configure the security and sharing model entirely using the user interface, the model works at the API level. That means any permissions you specify apply even if you query or update the data via API calls. The security of your data is protected, regardless of how users get to it.

**Levels of Data Access**

You can control which users have access to which data in your whole org, a specific object, a specific field, or an individual record.

**Organization**

For your whole org, you can maintain a list of authorized users, set password policies, and limit logins to certain hours and locations.

**Objects**

Access to object-level data is the simplest thing to control. By setting permissions on a particular type of object, you can prevent a group of users from creating, viewing, editing, or deleting any records of that object. For example, you can use object permissions to ensure that interviewers can view positions and job applications but not edit or delete them.

**Fields**

You can restrict access to certain fields, even if a user has access to the object. For example, you can make the salary field in a position object invisible to interviewers but visible to hiring managers and recruiters.

**Records**

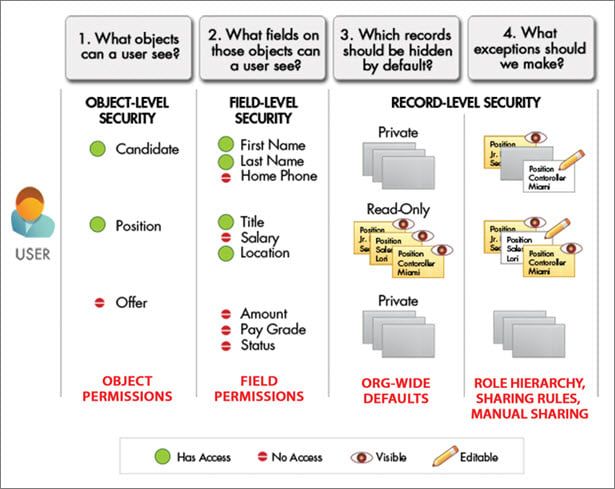
You can allow particular users to view an object, but then restrict the individual object records they're allowed to see. For example, an interviewer can see and edit her own reviews, but not the reviews of other interviewers. You can manage record-level access in these four ways.

* ***Organization-wide defaults*** specify the default level of access users have to each others’ records. You use org-wide sharing settings to lock down your data to the most restrictive level, and then use the other record-level security and sharing tools to selectively give access to other users.
* ***Role hierarchies*** give access for users higher in the hierarchy to all records owned by users below them in the hierarchy. Role hierarchies don’t have to match your organization chart exactly. Instead, each role in the hierarchy should represent a level of data access that a user or group of users needs.
* ***Sharing rules*** are automatic exceptions to organization-wide defaults for particular groups of users, so they can get to records they don’t own or can’t normally see. Sharing rules, like role hierarchies, are only used to give additional users access to records. They can’t be stricter than your organization-wide default settings.
* ***Manual sharing*** allows owners of particular records to share them with other users. Although manual sharing isn’t automated like org-wide sharing settings, role hierarchies, or sharing rules, it can be useful in some situations, such as when a recruiter going on vacation needs to temporarily assign ownership of a job application to someone else.

Tip

**Tip**

Make a table of the various types of users in your organization. In the table, specify the level of access to data that each type of user for each object and for fields and records within the object. You can then refer to this table as you set up your security model.

Controlling Data Access with the Salesforce Platform

**Audit System Use**

Auditing provides important information for diagnosing potential security issues or dealing with real ones. Someone in your organization should audit regularly to detect potential abuse. Look for unexpected changes or patterns of use.

**Record Modification Fields**

All objects include fields to store the name of the user who created the record and who last modified the record. This provides some basic auditing information.

**Login History**

You can review a list of successful and failed login attempts for the past six months. For more information, see Monitor Login History.

**Field History Tracking**

You can turn on auditing to automatically track changes in the values of individual fields. Although field-level auditing is available for all custom objects, only some standard objects allow it. For more information, see Field History Tracking.

**Setup Audit Trail**

The Setup Audit Trail logs when modifications are made to your organization’s configuration. For more information, see Monitor Setup Changes.

**Control Access to the Org**

**Learning Objectives**

After completing this unit, you'll be able to:

* Create, view, and manage users.
* Set password policies.
* Limit the IP addresses from which users can log in.
* Limit the times at which users can log in.

**Control Access to the Organization**

When you ensure that only employees who meet certain criteria can log in to Salesforce, you're protecting your data at the broadest level. You do this by managing authorized users, setting password policies, and limiting when and where users can log in.

**Manage Users**

Every Salesforce user is identified by a username, a password, and a single profile. Together with other settings, the profile determines what tasks users can perform, what data they see, and what they can do with the data.

To view and manage the users in your org, use the Quick Find box in Setup to find **Users**. The user list shows all the users in your org.

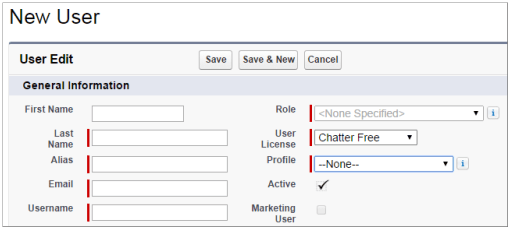
**Create a User**

You can create users—even multiple users—in just a few clicks. It’s as simple as entering a username, alias, and email, and selecting a role, license, and profile. Many more options are available, of course, but that’s all you need to get started.

Salesforce auto-generates a password and notifies new users immediately. Users can change or add to their own personal information after they log in.

1. Use the Quick Find box to find **Users** | **Users** in Setup.
2. Click **New User**.

Or you can click **Add Multiple Users** to add up to ten users at a time.



1. Enter the user’s name, email address, and a unique username in the form of an email address. By default, the username is the same as the email address.
2. Select the user license this user will have.

The license determines which profiles are available for each user.

1. Select a profile, which specifies the user’s minimum permissions and access settings.
2. Select the option to generate a new password and notify the user, then save.

**Deactivate a User**

You can't delete a user, but you can deactivate an account so a user can’t log in. Deactivated users lose access to all records. (That includes records that are shared with them individually and records shared with them as team members.) However, you can still transfer this data to other users and view the names on the Users page.

1. In Setup, use the Quick Find box to go to **Users**.
2. Click **Edit** next to the name of the user you want to deactivate.
3. Clear the **Active** checkbox and click **Save**.

If you can’t immediately deactivate an account (for example, when the user is selected in a custom hierarchy field), you can freeze their account. That prevents the user from logging in to your organization while you’re working on deactivating them.

* 1. On the Users page in Setup, click the username of the user whose account you want to freeze.
  2. Click **Freeze**.

**Set Password Policy**

You can configure several settings to ensure that your users’ passwords are strong and secure.

**Password policies**

Set password and login policies, such as specifying an amount of time before all users’ passwords expire and the level of complexity required for passwords.

**User password expiration**

Expire the passwords for all the users in your org, except for users with “Password Never Expires” permission.

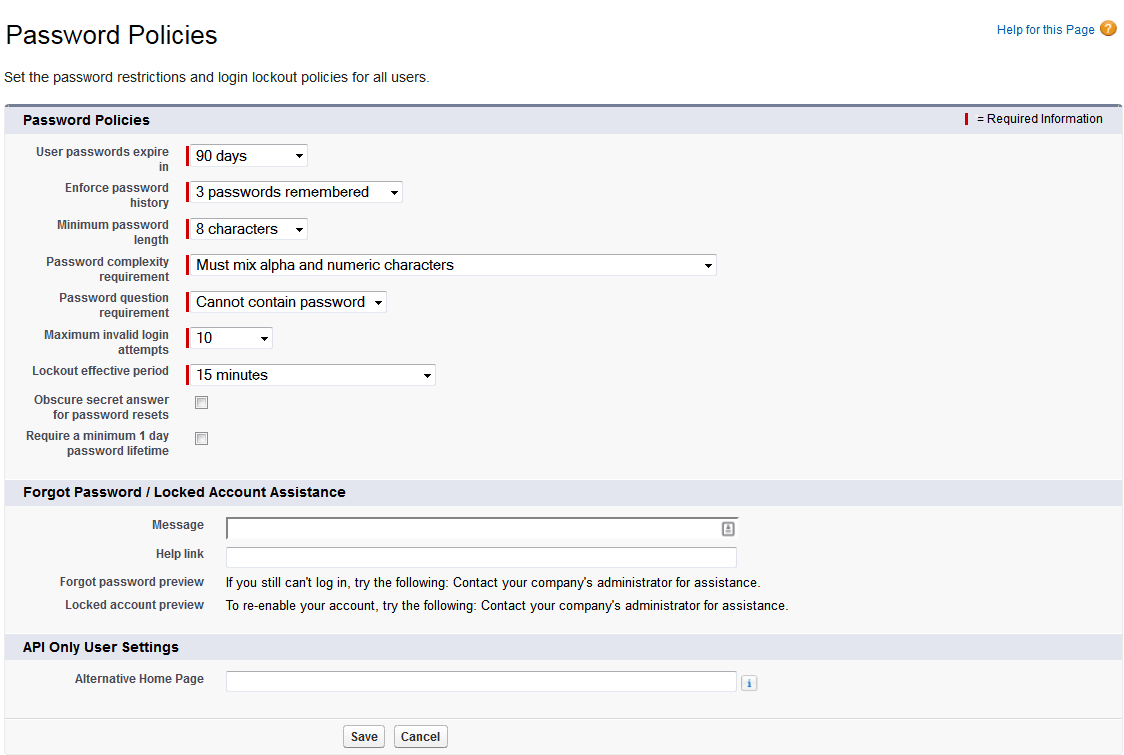
**User password resets**

Reset the password for specified users.

**Login attempts and lockout periods**

If a user is locked out due to too many failed login attempts, you can unlock the person’s access.

1. Use the Quick Find box to find **Password Policies** in Setup.



1. Customize the password settings.
   1. How long should passwords be?

Longer is usually better, within reason.

* 1. How complex do you want your passwords?

You can require alphabetical, numeric, uppercase, lowercase, or special characters.

* 1. How many days is a password valid?
  2. How many times can someone try to log in with invalid credentials before being locked out?

1. Choose what to do about forgotten passwords and locked accounts.
2. Click **Save**.

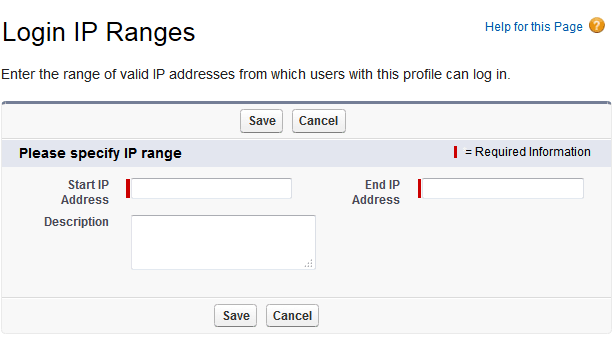
**Restrict Login Access by IP Address**

You can control where your users can log in from. For example, maybe some users shouldn’t be able to log in if they’re using an IP address that’s outside your corporate firewall. The IP range you choose is called your “trusted” IP range.

* If you set your trusted IP range for your whole org, users with addresses outside that range aren't completely excluded. They can log in if they complete a challenge question, typically by entering an activation code sent to their phone or email.
* If you set your trusted IP range only for a given user profile, all users with that profile who are outside the trusted range are locked out.

By default, Salesforce doesn't restrict locations for login access. If you do nothing, users can log in from any IP address.

1. Go to your Setup panel.
   * If you're doing this for your whole org, use the Quick Find box to find **Network Access**.
   * If you're doing this for a profile, find **Profiles** instead, then click the name of the profile you want to edit.
2. Click **New** in the Login IP Range related list.



1. Enter the start and end point of the range of trusted IP addresses, and save.

**Restrict Login Access by Time**

For each profile, you can specify the hours when users can log in. For example, if you decide your call center employees really only need to look at customer data while they're taking phone calls nine to five, you can make it so they can't log in during evenings and weekends.

1. In Setup, use the Quick Find box to find **Profiles**.
2. Click the profile you want to change.
3. Under **Login Hours**, click **Edit**.
4. Set the days and hours when users with this profile can log in to the organization.
   * To allow users to log in at any time, click **Clear all times**.
   * To prohibit users from using the system on a specific day, set the start and end times to the same value.

# Control Access to Objects

## Learning Objectives

After completing this unit, you'll be able to:

* View existing profiles and create new ones.
* Modify access to objects using profiles.
* View all assigned users in a profile.
* Create new permission sets.
* Assign permission sets to single and multiple users.

## Manage Object Permissions

The simplest way to control data access is to set permissions on a particular type of object. (An object is a collection of records, like leads or contacts.) You can control whether a group of users can create, view, edit, or delete any records of that object.

You can set object permissions with profiles or permission sets. A user can have one profile and many permission sets.

* A user’s profile determines the objects they can access and the things they can do with any object record (such as create, read, edit, or delete).
* Permission sets grant additional permissions and access settings to a user.

Use profiles to grant the minimum permissions and settings that all users of a particular type need. Then use permission sets to grant more permissions as needed. The combination of profiles and permission sets gives you a great deal of flexibility in specifying object-level access.

### **Object Permissions for the Recruiting App**

As an example, let’s explore how you might configure object-level access in the Recruiting app. The app has four main types of users: hiring managers, recruiters, interviewers, and standard employees. What kinds of access to objects does each type of user need?

**Hiring Managers**

Ben, a hiring manager, should be able to access the recruiting records related to his open positions, but shouldn't have access to other recruiting records (unless they're owned by other hiring managers who report to him). Also, there are certain sensitive fields that he has no need to see, like the social security number field. Let’s consider the permissions Ben needs for each of the key custom objects in the app.

* **Position**—Ben should be able to post new positions, as well as update and view all fields for positions for which he's the hiring manager, but he should only be able to view other managers' positions.
* **Candidate**—Ben should only be able to view those candidates who have applied for a position on which he's the hiring manager. Also, since Ben has no reason to see a candidate's social security number, this field should be restricted from his view.
* **Job Application**— Ben needs to be able to update the status of job applications to specify which candidates should be selected or rejected. However, he should not be able to change the candidate listed on the job application, nor the position to which the candidate is applying, so we'll have to find a way of preventing Ben from updating the lookup fields on job applications.
* **Review**—To make a decision about the candidates who are applying, Ben needs to see the reviews posted by the interviewers, as well as make comments on them if he thinks the interviewer was being too biased in his or her review. Likewise, Ben needs to be able to create reviews so that he can remember his own impressions of the candidates he interviews.

**Recruiters**

Mario, a recruiter, needs to be able to create, view, and modify any position, candidate, job application, or review that's in the system. He also needs to view and modify the recruiting records that all of the other recruiters own, since all the recruiters work together to fill each position, regardless of who created it.

We need to make sure a recruiter will never accidentally delete a record with information about a candidate. That’s because state and federal laws require recruitment-related records be saved for a number of years, so that if a hiring decision is questioned, it can be defended in court.

**Interviewers**

Melissa is an engineer who interviews candidates for highly technical positions. She should be able to view only the candidates and job applications to which she's assigned as an interviewer. Also, she shouldn't be able to view the minimum and maximum salary values for any of the positions or the social security number of any candidate, as that’s sensitive information that has nothing to do with her job.

**Standard Employees**

Employees, such as Harry, are often the best resources for recruiting new hires, even if they are not active hiring managers or interviewers. For this reason, we need to make sure that employees can view open positions, but that they can't see the values for the positions' minimum and maximum salary fields—otherwise they might tip off friends to negotiate for a position's maximum salary value! Harry also shouldn't be able to view any other records in the Recruiting app.

Here are the required permissions for each of the four types of users.

| **Custom Object** | **Recruiter** | **Hiring Manager** | **Interviewer** | **Standard Employee** |
| --- | --- | --- | --- | --- |
| Position | Read Create Edit | Read Create Edit\* | Read (No min/max pay) | Read (No min/max pay) |
| Candidate | Read Create Edit | Read\* (No SSN) | Read \* (No SSN) |  |
| Job Application | Read Create Edit | Read Edit (No lookup fields) | Read \* |  |
| Review | Read Create Edit | Read Create Edit | Read \*\* Create Edit \*\* |  |

\* Only for those records that are associated with a position to which the hiring manager or interviewer has been assigned.

\*\* Only for those records that the interviewer owns.

In the rest of this module, you'll learn how you can use the platform to implement these rules in the Recruiting app. As you’ll see, this will require configuring security controls at all three levels: objects, fields, and records.

## Use Profiles to Restrict Access

Each user has a single profile that controls which data and features that user has access to. A profile is a collection of settings and permissions. Profile settings determine which data the user can see, and permissions determine what the user can do with that data.

* The settings in a user’s profile determine whether she can see a particular app, tab, field, or record type.
* The permissions in a user’s profile determine whether she can create or edit records of a given type, run reports, and customize the app.

Profiles usually match up with a user's job function (for example, system administrator, recruiter, or hiring manager), but you can have profiles for anything that makes sense for your Salesforce org. A profile can be assigned to many users, but a user can have only one profile at a time.

### **Standard Profiles**

The platform includes a set of standard profiles. Some examples are:

* Read Only
* Standard User
* Marketing User
* Contract Manager
* System Administrator

Each standard profile includes a default set of permissions for all standard objects available on the platform. For example, a Standard User can create and edit records while a Read Only user can view records, but not create or edit them. The System Administrator profile has the widest access to data and the greatest ability to configure and customize Salesforce. The System Administrator profile also includes two special permissions:

* View All Data
* Modify All Data

These permissions override all other sharing settings, so use caution when assigning them to any profile other than System Administrator. You can view a list of all standard and custom profiles in Setup.

You can’t edit the object permissions on a standard profile. However, you can clone any existing profile, and use that as the basis for a new profile, adjusting the apps and system settings as needed. For example, in the Recruiting app, you might create three new profiles, one each for recruiters, interviewers, and hiring managers. Each profile can then be configured to provide the specific type of data access required for a particular role. You can then use permission sets to grant additional permissions, as required.

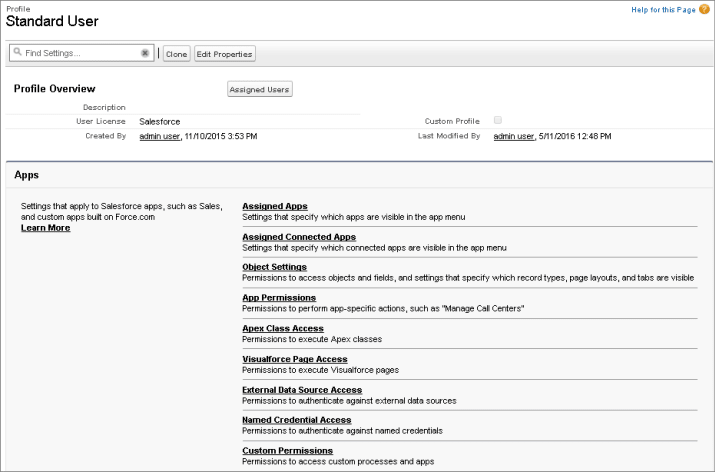
Note

#### Note

The profiles functionality in an org depends on the user license type.

### **Managing Profiles**

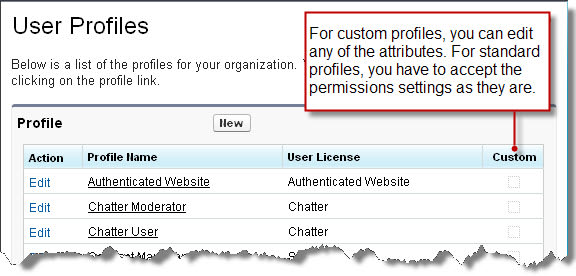
The profile overview page provides an entry point for all of the settings and permissions for a single profile. In Setup, use the Quick Find box to find **Profiles** and click the profile you want to view.



## Create a Profile

The easiest way to create a profile is to clone an existing profile that’s similar to the one you want to create, and then modify it.

Salesforce has an enhanced profile user interface that makes it easy to find and modify profile settings. That’s what we’ll use for this exercise. To do that, find User Management Settings in the **Quick Find** box in Setup, then enable **Enhanced Profile User Interface** and click **Save**.



1. In Setup, use the Quick Find box to find **Profiles**.
2. Click **Clone** next to a profile similar to the one you want to create.
3. Give your new profile a name, and save.

## Assign a Profile

Once you've created a profile, you'll want to customize it to match the needs of a set of users, and then assign the profile to those users.

1. Find Profiles in Setup.
2. Select a profile and then click **Object Settings**. Click **Edit** to see its settings.
3. Set the most restrictive settings and permissions you can for this user type, and save.

(Don’t worry about blocking the user from doing things they need to do. We'll open up more possibilities for them later, when we give them permission sets.)

1. Find Users in Setup and click **Edit** next to one of them.
2. From the **Profile** dropdown select the profile you just set up, and save.

## Use Permission Sets to Grant Access

A permission set is a collection of settings and permissions that give users access to various tools and functions. The settings and permissions in permission sets are also found in profiles, but permission sets extend users’ functional access without changing their profiles.

Permission sets make it easy to grant access to the various apps and custom objects in your org, and to take away access when it’s no longer needed.

Users can have only one profile, but they can have multiple permission sets.

You'll be using permission sets for two general purposes: to grant access to custom objects or apps, and to grant permissions—temporarily or long term—to specific fields.

**Grant access to custom objects or apps.**

Let’s say you have many users in your org with the same fundamental job functions. You can assign them all one profile that grants them all the access they need to do their job. But a few of those users are working on a special project and they need access to an app no one else uses. And a few other users need access to that app, as well as another app that the first group doesn’t need. If we only had profiles, you’d have to create more profiles customized to those few users’ needs, or take your chances and add more access to the original profile, making the apps available to users that don’t need them. Neither of these options is ideal, especially if your org is growing and your users’ needs change regularly.

**Grant permissions to specific fields.**

Let’s say you have a user, Tom, who needs temporary edit access to a field while his co-worker is on vacation. You can create a permission set that grants access to the field and assign the permission set to Tom. When Tom’s co-worker returns from vacation and Tom no longer needs access to the field, you just remove the permission set assignment from Tom’s user record.

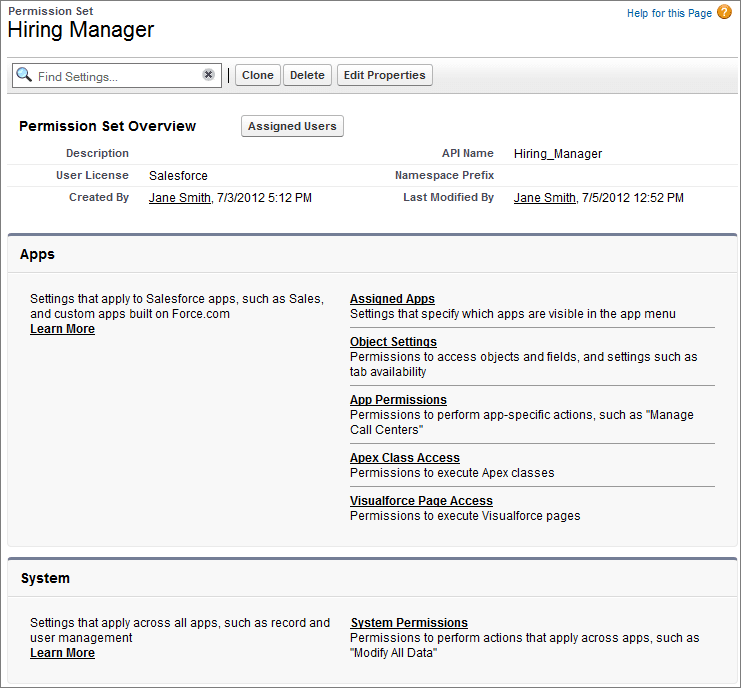
Note

#### Note

If a user has a permission in their base profile, you can’t remove it by assigning a permission set to that user. A permission can only add permissions. To take away a permission, you have to remove it from the user's base profile and from any permission sets the user may have.

### **Managing Permission Sets**

A permission set’s overview page is the entry point for all of the permissions in a permission set. To open a permission set overview page, find Permission Sets in Setup, then select the permission set you want to view. In each permission set, permissions and settings are organized into app settings, system settings, object permissions, and field permissions.



## Create a Permission Set

Create a permission set to grant additional permissions to specific users, on top of their existing profile permissions, without having to modify existing profiles, create new profiles, or grant an administrator profile.

1. Use the Quick Find box to find **Permission Sets** in Setup.
2. Click **Clone** next to the set you want to copy.

Note

#### Note

A cloned permission set has the same user license as the original. To create a set with a different license, click **New**instead.

1. Enter a label and a description.

The API name is a unique name used by the API and managed packages. It automatically replicates the label, but you can modify it.

1. If this is a new permission set, select a user license option.
   * If you plan to assign this permission set to multiple users with different licenses, select --None--.
   * If only users with one type of license will use this permission set, select that user license.
2. Click **Save** to go back to the permission set overview page.
3. In the permission set toolbar, click **Manage Assignments**, then click **Add Assignments**.
4. Select the users to assign to this permission set and click **Assign**.

Review the messages on the Assignment Summary page. If any users weren’t assigned, the Message column tells you why.

1. Click **Done** to return to a list of the users assigned to the permission set.

## Profiles and Permission Sets for the Recruiting App

Now that you've seen how to create and modify profiles and permission sets, let’s set up the appropriate object-level access for our Recruiting app. The app has four main types of users: recruiters, hiring managers, interviewers, and standard employees.

Here are the key considerations for deciding whether to create a profile or permission set for each type of user.

**Recruiters**

These represent a clearly defined job function, and they need access to different types of data than other users. Hence, it makes sense to create a profile for recruiters.

**Hiring Managers**

For most orgs, a hiring manager in Sales will need access to a different type of data than a hiring manager in Engineering. However, all hiring managers still need the same types of access to recruiting data—reviews, candidates, positions, and job applications. Hence, it’s convenient to create a hiring manager permission set that can be assigned to various types of users.

**Interviewers**

An employee from any department and in any job function might be called upon to perform an interview and requires access to recruiting information only for a limited amount of time. It makes sense to define a permission set for interviewers, since permissions can be easily assigned and revoked as needed.

**Standard Employees**

This is a generic group that doesn’t reflect a particular job function. For most employees, you can create a base profile that provides access to a small set of data, and then depending on what their specialties are, create and assign permission sets to give them more access as needed.

So from what we've seen, the optimal way to configure object permissions for the Recruiting app is like this:

1. Create two profiles: Recruiters and Standard Employees.
2. Create two permission sets: Hiring Managers and Interviewers.
3. Assign the Standard Employee profile to hiring managers and interviewers, and then grant the appropriate permission set for their function.

# Control Access to Fields

## Learning Objectives

After completing this unit, you'll be able to:

* List reasons to limit access to specific fields.
* View and edit field-level security settings.

## Modify Field-Level Security

Defining field-level security for sensitive fields is the second piece of the security and sharing puzzle, after controlling object-level access.

In some cases, you want users to have access to an object, but limit their access to individual fields in that object. Field-level security settings—or field permissions—control whether a user can see, edit, and delete the value for a particular field on an object. These are the settings that allow us to protect sensitive fields such as a candidate's social security number without having to hide the candidate object.

Unlike page layouts, which only control the visibility of fields on detail and edit pages, field-level security controls the visibility of fields in any part of the app, including related lists, list views, reports, and search results. In fact, to make absolutely sure that a user can't access a particular field, it's important to use the field-level security page for a given object to restrict access to the field. There are simply no other shortcuts that provide the same level of protection for a particular field.

For example, here are some field-level security settings you can set for the Recruiting app.

* Position object—hide minimum and maximum pay from standard employees and interviewers.
* Candidate object—hide social security numbers from hiring managers and interviewers.
* Job Application object—make the Position and Candidate lookup fields read-only for hiring managers.

Field settings can be applied either by modifying profiles or permission sets or from the Field Accessibility menu in Setup.

After setting field-level security for users, you can:

* Create page layouts to organize the fields on detail and edit pages.
* Verify users’ access to fields by checking the field accessibility.
* Customize search layouts to set the fields that display in search results, in lookup dialog search results, and in the key lists on tab home pages.

## Restrict Field Access with a Profile

You apply field settings by modifying profiles or permission sets. Let’s try restricting a user's general access with a profile. Then we'll expand it as needed with a permission set.

Tip

#### Tip

If you haven’t already done it, before you start, enable the enhanced profile user interface. Type User Management Settings in the **Quick Find** box in Setup, then enable **Enhanced Profile User Interface** and click **Save**.

1. Use the Quick Find box to find **Profiles** in Setup.
2. Select the profile you want to change. "Standard User" will do nicely.
3. Click **Object Settings** and select the object for which you want to update the field settings.
4. Click **Edit**.
5. For each field, specify the kind of access you want for users with this profile, and save your settings.

Now that you've set field-level security for sensitive data, you can create page layouts to organize the fields for users' convenience, and customize how the fields display in search results and lists. For the final piece of the puzzle, specify the individual records to which each user needs access. By combining security controls at all three levels, you can set up a highly secure data access model which is flexible enough to meet the needs of many different types of users.

## Add Field Access with a Permission Set

Let’s look at how field settings can be applied by modifying permission sets. Remember, a permission set is for expanding a user’s access to fields that are restricted in their profile.

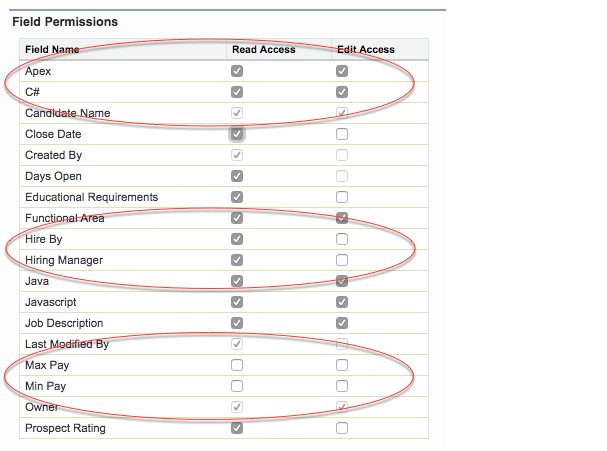
Let's set up our interviewers to update the candidate record when they've interviewed a candidate. We’ll assume our interviewers have the Standard User profile.

We worked with Permission Sets when we set up our custom objects. Now we'll go back to that Setup page to make sure the right fields in one or our objects are available to the users who need them.

1. In Setup, use the Quick Find box to find **Permission Sets**.
2. Select a permission set and click **Object Settings**.
3. Click the object you're working with, then click **Edit**.

In this example, we're modifying the Candidate object.

1. Under **Field Permissions**, specify the kinds of access your interviewers need, then save this permission set.

See how we've enabled our interviewers to both read and change the values of the Apex and C# checkboxes? Now they can check or uncheck those boxes when they’ve determined the candidate’s command of those skills. We’ve prevented them from changing the Hire By date or the name of the hiring manager, but they can see that information. And they don’t need to know the pay rate for the position, so we’ve removed both their Read and Edit access for those fields.

1. Click **Manage Assignments** and select the users who you expect to need the permissions you’ve just specified. Click **Add Assignments** and **Done**, and you're done!

Now you’ve defined field-level security for sensitive data. For the final piece of the puzzle, specify the individual records each user needs access to. By combining security controls at all three levels, you can set up a highly secure data access model that’s flexible enough to meet the needs of many different types of users.

**Control Access to Records**

**Learning Objectives**

After completing this unit, you'll be able to:

* List the four ways to control access to records.
* Describe situations in which to use each of the four record-level security controls.
* Explain how the different record controls interact with each other.
* Set org-wide sharing defaults to control access to records.

**Record-Level Security**

To control data access precisely, you can allow particular users to view specific fields in a specific object, but then restrict the individual records they're allowed to see.

Record access determines which individual records users can view and edit in each object they have access to in their profile. First ask yourself these questions:

* Should your users have open access to every record, or just a subset?
* If it’s a subset, what rules should determine whether the user can access them?

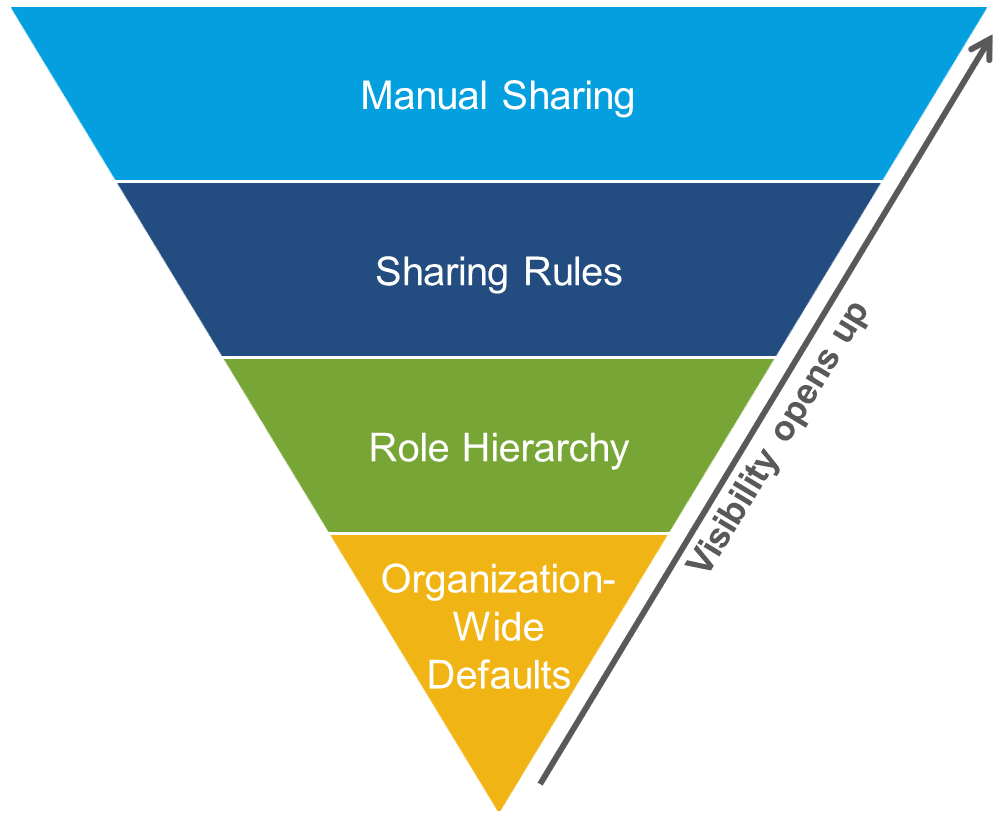
Let’s say you create a new profile called Recruiter to give recruiters the object-level permissions they need. You restrict the power to delete recruiting-related objects, so recruiters will never be able to delete these objects. However, granting recruiters permission to create, read, or edit recruiting objects does not necessarily mean recruiters can read or edit every record in the recruiting object. This is a consequence of two important concepts:

* The permissions on a record are always evaluated according to a combination of object-level, field-level, and record-level permissions.
* When object-level permissions conflict with record-level permissions, the most restrictive settings win.

That means even if you grant a profile create, read, and edit permissions on the recruiting objects, if the record-level permissions for an individual recruiting record are more restrictive, those are the rules that define what a recruiter can access.

You control record-level access in four ways. They’re listed in order of increasing access. You use org-wide defaults to lock down your data to the most restrictive level, and then use the other record-level security tools to grant access to selected users, as required.

* **Org-wide defaults**specify the default level of access users have to each other’s records.
* **Role hierarchies** ensure managers have access to the same records as their subordinates. Each role in the hierarchy represents a level of data access that a user or group of users needs.
* **Sharing rules** are automatic exceptions to org-wide defaults for particular groups of users, to give them access to records they don’t own or can’t normally see.
* **Manual sharing** lets record owners give read and edit permissions to users who might not have access to the record any other way.



The visibility and access for any type of data is determined by the interaction of the above security controls, based on these key principles.

* A user’s baseline permissions on any object are determined by their profile.
* If the user has any permission sets assigned, these also set the baseline permissions in conjunction with the profile.
* Access to records a user does not own are set first by the org-wide defaults.
* If the org-wide defaults are anything less than **Public Read/Write**, you can open access back up for certain roles using the role hierarchy.
* You can use sharing rules to expand access to additional groups of users.
* Each record owner can manually share individual records with other users by using the Share button on the record.

You’ve already seen how to configure object-level and field-level access using profiles and permission sets. Now we’ll look at details of the various record-level security controls.

**Org-Wide Sharing**

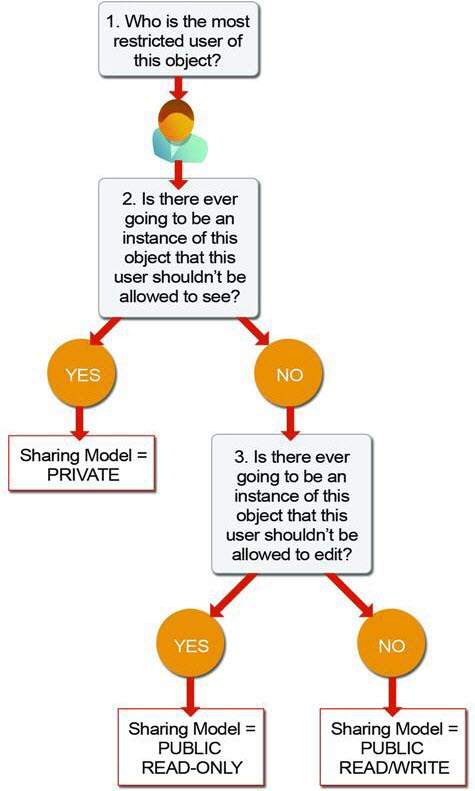
Org-wide defaults specify the baseline level of access that the most restricted user should have. Use org-wide defaults to lock down your data, and then use the other record-level security and sharing tools (role hierarchies, sharing rules, and manual sharing) to open up the data to users who need it.

Object permissions determine the baseline level of access for all the records in an object. Org-wide defaults modify those permissions for records a users doesn't own. Org-wide sharing settings can be set separately for each type of object.

Org-wide defaults can never grant users more access than they have through their object permission.

To determine the org-wide defaults you need for your app, ask yourself these questions about each object:

1. Who is the most restricted user of this object?
2. Is there ever going to be an instance of this object that this user shouldn't be allowed to see?
3. Is there ever going to be an instance of this object that this user shouldn't be allowed to edit?



Based on your answers, you can set the sharing model for that object to one of these settings.

**Private**

Only the record owner, and users above that role in the hierarchy, can view, edit, and report on those records.

**Public Read Only**

All users can view and report on records, but only the owner, and users above that role in the hierarchy, can edit them.

**Public Read/Write**

All users can view, edit, and report on all records.

**Controlled by Parent**

A user can view, edit, or delete a record if she can perform that same action on the record it belongs to.

When the org-wide sharing setting for an object is **Private**or **Public Read Only**, an admin can grant users additional access to records by setting up a role hierarchy or defining sharing rules. Sharing rules can only be used to grant additional access. They cannot be used to restrict access to records beyond what was originally specified with the org-wide sharing defaults.

As an example, let’s go through and answer the above list of questions for the Position object in the Recruiting app.

**Who is the most restricted user of this object?**

A member of the Standard Employee profile. All that they're allowed to do is view a position.

**Is there ever going to be an instance of this object that this user shouldn't be allowed to see?**

No. Although the values for the minimum and maximum pay fields are hidden from standard employees, they're still allowed to view all position records.

**Is there ever going to be an instance of this object that this user shouldn't be allowed to edit?**

Yes. Standard employees aren’t allowed to edit any position record.

Since we answered “Yes” to the third question, the sharing model for the Position object should be set to Public Read Only. By repeating the same exercise with the other recruiting objects, you can easily figure out the appropriate org-wide default settings for them. The Standard Employee profile is the most restricted user for each object, and there are going to be candidate, job application, and review records that particular employees won't be able to view. Consequently, the sharing model for the Candidate, Job Application, and Review objects should all be set to **Private**.

Note

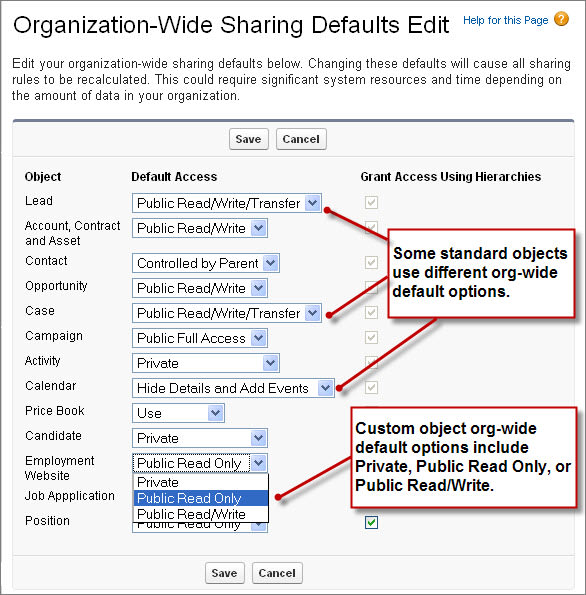
**Note**

You can't set the org-wide defaults for the Review object, because that object is on the detail side of a master-detail relationship, and a detail record automatically inherits the sharing setting of its parent. So in our app, the Review object is automatically set to **Private**.

**Set Your Org-Wide Sharing Defaults**

Use org-wide defaults to specify the baseline level of access that the most restricted user should have.

1. In Setup, use the Quick Find box to find **Sharing Settings**.
2. Click **Edit** in the Organization-Wide Defaults area.



1. For each object, select the default access you want to give everyone.
2. To disable automatic access using your hierarchies, deselect **Grant Access Using Hierarchies** for any custom object that does not have a default access of Controlled by Parent.

By default, a role hierarchy automatically grants access to records for users above the record owner in the hierarchy. Setting an object to **Private** makes those records visible *only* to record owners and those above them in the role hierarchy. Use the **Grant Access Using Hierarchies** checkbox to disable access to records to users above the record owner in the hierarchy for custom objects. If you deselect this checkbox for a custom object, only the record owner and users granted access by the org-wide defaults receive access to the records.

Even if **Grant Access Using Hierarchies** is deselected, some users—such as those with the “View All” and “Modify All” object permissions and the “View All Data” and “Modify All Data” system permissions—can still access records they don’t own.

Note

**Note**

Updating the org-wide defaults automatically runs sharing recalculation to apply any access changes to your records. You receive a notification email when the recalculation completes and you can refresh the Sharing Settings page to see your changes. To view the update status, from Setup, enter View Setup Audit Trailin the Quick Find box, then select **View Setup Audit Trail**.

Once you’ve locked down your data with org-wide defaults, the resulting settings might be too restrictive for some users. You can then use the remaining record-level security controls (role hierarchies, sharing rules, and manual sharing) to open up record access selectively to specific employees who need it.

Bottom of Form

**Create a Role Hierarchy**

**Learning Objectives**

After completing this unit, you'll be able to:

* Explain how a role hierarchy is different from an org chart.
* View and modify a role hierarchy.
* Create and assign roles to simplify access to records.

**Create and Edit Roles**

A role hierarchy works together with sharing settings to determine the levels of access users have to your Salesforce data. Users can access the data of all the users directly below them in the hierarchy.

Users who need to see a lot of data (such as the CEO, executives, or other management) often appear near the top of the hierarchy. But role hierarchies don't have to match your org chart. Each role in the hierarchy just represents a level of data access that a user or group of users needs.

* A manager always has access to the same data as his or her employees, regardless of the org-wide default settings.
* Users who tend to need access to the same types of records can be grouped together. We'll use these groups later when we talk about sharing rules.

Depending on your sharing settings, roles can control the level of visibility that users have into your Salesforce data. Users at any given role level can view, edit, and report on all data owned by or shared with users below them in the role hierarchy, unless your sharing model for an object specifies otherwise. Specifically, in the Organization-Wide Defaults related list, if the **Grant Access Using Hierarchies** option is disabled for a custom object, only the record owner and users granted access by the org-wide defaults receive access to the object's records.

Beyond setting the org-wide sharing defaults for each object, you can specify whether users have access to the data owned by or shared with their subordinates in the hierarchy. For example, the role hierarchy automatically grants record access to users above the record owner in the hierarchy. By default, the **Grant Access Using Hierarchies** option is enabled for all objects. It can only be changed for custom objects.

To control sharing access using hierarchies for any custom object, enter Sharing Settings in the Quick Find box and select **Sharing Settings**. In the Organization Wide Defaults section, click **Edit**. Deselect **Grant Access Using Hierarchies** if you want to prevent users from gaining automatic access to data owned by or shared with their subordinates in the hierarchies.

**Define a Role Hierarchy**

Implementing a role hierarchy in the platform is easy once you have an idea of what the hierarchy should look like. It’s best to start with your company’s org chart and then consolidate different job titles into single roles wherever possible.

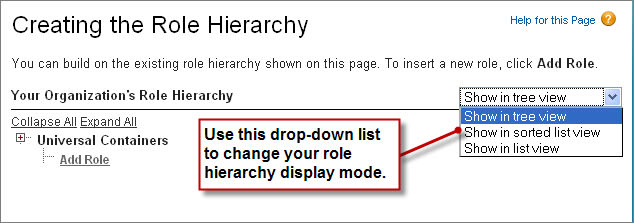
For example, if a software development group has a staff software engineer and a junior software engineer, these positions can be consolidated into a single Software Engineer role in the hierarchy. Once that’s done, you can get started defining the role hierarchy itself.

1. From Setup, use the Quick Find box to find **Roles**.

Tip

**Tip**

If you see an introductory splash page called Understanding Roles, click **Set Up Roles** at the bottom of the page to skip to the actual tool.



The default view for this page is the tree view, as indicated in the drop-down list on the far right side of the Role Hierarchy title bar. When creating a role hierarchy, it's probably easiest to stick with this or the list view, because they both make it easy to see how the roles all fit together in the hierarchy. The sorted list view is best if you know the name of a role that you want to find but aren't sure where it fits in the hierarchy, or if you don't want to click open all the tree nodes. For our purposes, we'll stick with the tree view.

When you first start defining a role hierarchy, the tree view displays a single placeholder node with the name of your org. From this point, we need to add the name of the role that is highest up in the hierarchy—in our case, the CEO.

Note

**Note**

If you’re building your app with a free Developer Edition or Trailhead Playground org, you may have a role hierarchy predefined as a sample. That's all right. You can still follow along and create some more roles.

1. Just under the company name, click **Add Role**.

Note

**Note**

If the CEO role already exists, click **Edit**.

1. In the **Label** text box, enter CEO.

The **Role Name** text box autopopulates with CEO.

1. In the **This role reports to** text box, click the lookup icon Lookup icon and click **Select** next to the name of your org.

By choosing the name of the org in the **This role reports to** text box, we’re indicating that the CEO role is a top-level position in our role hierarchy and doesn't report to anyone.

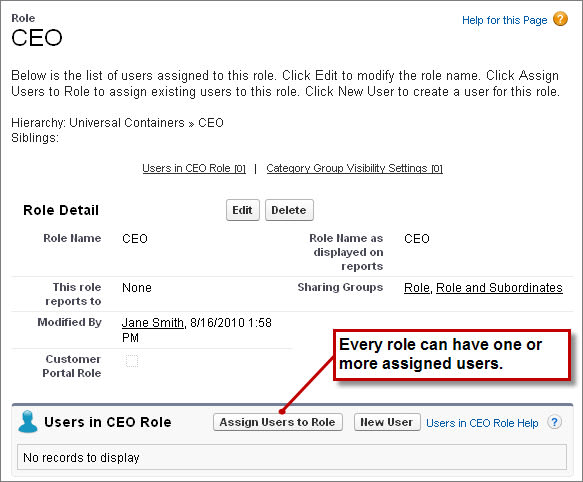
1. In the **Role Name as displayed on reports** text box, enter CEO.

This text is used in reports to indicate the name of a role. Since a long role name, like Vice President of Product Development, takes extra space in your report columns, we recommend using a short but readable abbreviation.

1. Leave any other options, such as **Opportunity Access**, set to their defaults, and save.

These access options don't have anything to do with our Recruiting app, and only appear if you have the org-wide defaults for a standard object set to a level more restrictive than **Public Read/Write**.

1. Now that you’ve created your first role, you can assign the appropriate user to it. Click **CEO**, and on the CEO role detail page, click **Assign Users to Role**.



1. In the **Available Users** drop-down list, select All Unassigned.
2. Choose a user from the list, and click **Add** to move her to the **Selected Users for CEO** list, then save.

If you return to the main Roles page from Setup, you can now see the new CEO role in the hierarchy. You can define the rest of the roles according to your role hierarchy diagram. There's no need to assign users to every role right away—you can do that later as you create the rest of your users and test out your app.

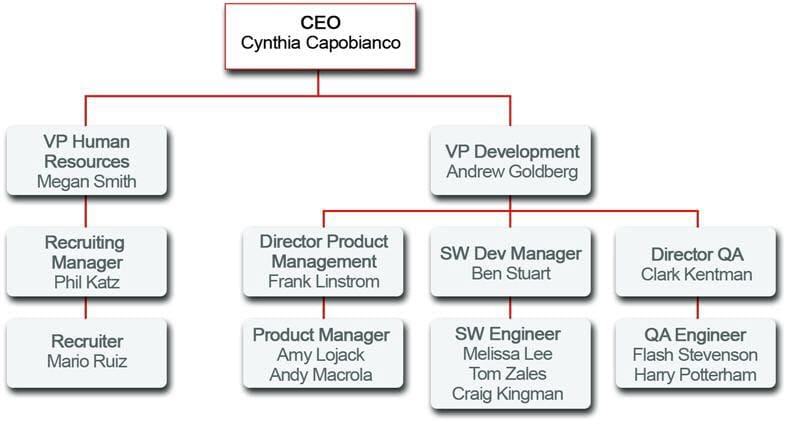
Tip

**Tip**

To speed up the process of adding a new role, click **Add Role** directly under the name of the role to which the new role should report. When you do this, the **This role reports to** text box is automatically filled in with the name of the appropriate role.

**Role Hierarchy for the Recruiting App**

Let's take a look at a branch of the role hierarchy for a fictional company that’s using our Recruiting app. Remember, with the org-wide defaults we defined, hiring managers are allowed to view (but not create or update) all position, job posting, and employment website records, and to view and update other recruiting records they own. That doesn't make our app all that useful. However, once our role hierarchy is in place, our users can get to the data they need, and our app is off and running.



This role hierarchy automatically grants these kinds of record-level permissions:

* The CEO, Cynthia, can view and update every record that anyone else in the organization can view and update.
* The VP of Development, Andrew, can view and update any record that his managers or his managers' employees can view or update.
* The VP of Human Resources, Megan, can view and update any record that Phil, her recruiting manager, or Mario, Phil's recruiter, can view and update.
* The Recruiting Manager, Phil, can view and update any record that is owned by Mario, his recruiter.
* The Software Development manager, Ben, can view and update any record that is owned by Melissa, Tom, or Craig, his software engineers.
* The director of QA, Clark, can view and update any record that is owned by Flash or Harry, his QA engineers.

# Define Sharing Rules

## Learning Objectives

After completing this unit, you'll be able to:

* Use sharing rules to extend access beyond the role hierarchy structure.
* Create a public group that includes users with different profiles and roles.

## Extend Access with Sharing Rules

Your org-wide default sharing settings give you a (relatively restrictive) baseline level of access for each object. If you have org-wide sharing defaults of Public Read Only or Private, you can open access back up for some users with sharing rules. This enables you to make automatic exceptions to your org-wide sharing settings for selected sets of users.

Sharing rules can be based on who owns the record or on the values of fields in the record. For example, use sharing rules to extend sharing access to users in public groups or roles. As with role hierarchies, sharing rules can never be stricter than your org-wide default settings. They just allow greater access for particular users.

Each sharing rule has three components.

**Share which records?**

You can share records owned by certain users or meeting certain criteria. Criteria-based sharing rules determine what records to share based on field values other than ownership.

**With which users?**

You can define groups of users by role or by defining a public group. A public group is an admin-defined grouping of users that can be used to simplify the creation of sharing rules. Each public group can be a combination of:

* individual users
* roles
* roles and subordinates
* other public groups

**What kind of access?**

You can assign either Read-Only or Read/Write access.

Sharing rules work best when they're defined for a particular group of users that you can determine or predict in advance, rather than a set of users that frequently changes. For example, in the Recruiting app, it’s important to share every position, candidate, job application, and review with every recruiter. Since recruiters all belong to either the Recruiting Manager or Recruiter roles in the role hierarchy, we can easily use a sharing rule to share those objects with the Recruiting Manager role and its subordinates.

Alternatively, consider another use case from the Recruiting app: interviewers need read access on the candidates and job applications for people they're interviewing. In this case, the set of interviewers is a lot harder to predict in advance—hiring managers might use different sets of interviewers depending on the position for which they're hiring, and the interviewers might come from different groups in the role hierarchy. So, this use case probably shouldn't be handled with sharing rules—the team of interviewers for any given manager is just too hard to predict.

### **Should we use a sharing rule?**

Let's go through the set of required permissions we still want to implement and pick out the ones that work best with sharing rules.

**Recruiters need read and update access on every position, candidate, job application, and review record that exists in the app.**

Yes. As we discussed previously, it's easy to pick out the group of recruiters in our role hierarchy.

**Hiring managers need read and update access on position and job posting records on which they're the hiring manager.**

No. It's too hard to predict which positions will be assigned to which hiring manager. We'll need to handle this use case some other way.

**Hiring managers need read access on candidate records on which they're the hiring manager.**

No. Again, it's too hard to predict which positions will be assigned to which hiring manager.

**Hiring managers need read and update access on every job application and review record.**

Yes. Since we're not restricting which job applications and reviews a hiring manager gets to read and update, we can easily pick out all of the hiring managers from our role hierarchy and define a sharing rule for them.

**Interviewers need read access on the candidate and job application records for people they're interviewing.**

No. As we discussed previously, it's hard to predict who will be a member of an interview team for a particular position.

To summarize, here are the key sharing rules we define for the Recruiting app.

| **Object** | **Rule Label** | **Owned by...** | **Share with...** | **Access Level** |
| --- | --- | --- | --- | --- |
| Review | Edit Reviews | Entire Organization | Recruiters and Hiring Managers | Read/Write |
| Candidate | Edit Candidates | Entire Organization | The role and subordinates of the Recruiting Manager | Read/Write |
| Position | Edit Positions | The role and subordinates of the Recruiting Manager | The role and subordinates of the Recruiting Manager | Read/Write |

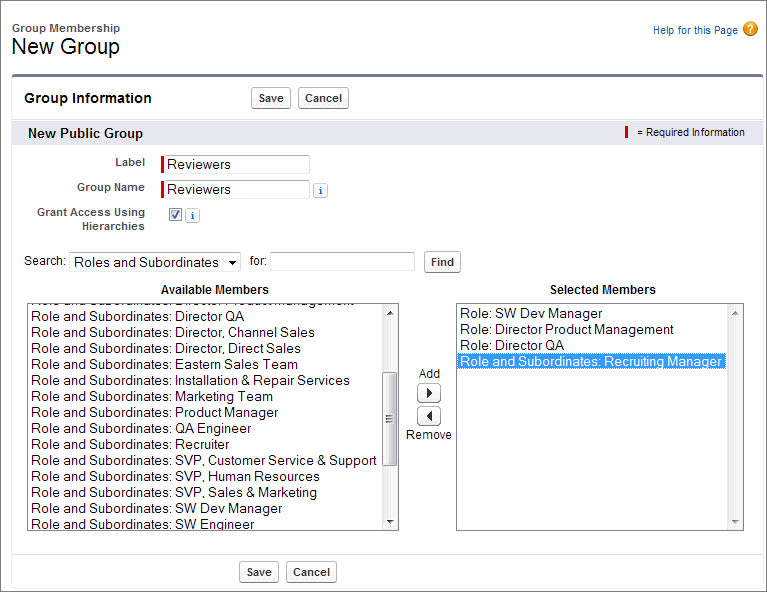
## Define a Public Group

Before creating a sharing rule, it’s important to set up the appropriate public group. A public group is a collection of individual users, other groups, individual roles, and/or roles with their subordinates that all have a function in common. For example, users with the Recruiter profile as well as users in the SW Dev Manager role both review job applications.

Using a public group when defining a sharing rule makes the rule easier to create and, more important, easier to understand later, especially if it's one of many sharing rules that you're trying to maintain in a large organization. Create a public group if you want to define a sharing rule that encompasses more than one or two groups or roles, or any individual.

Looking at the required permissions that we want to implement, there are just two objects that need a public group for their sharing rules: Job Application and Review. The good news is that we can cover these objects in a single group because the Review object is on the detail side of a master-detail relationship, so it inherits the sharing settings we apply to the Job Application object. Since both recruiters and hiring managers need read and update access to job applications and reviews, we can define a public group called Reviewers that includes both recruiters and hiring managers.

1. In Setup, use the Quick Find box to find **Public Groups**.
2. Click **New**.



The New Public Group page allows you to choose other public groups, individual roles, individual roles including the roles’ subordinates, or individual users.

1. Give your rule the label Reviewers.

The **Group Name** text box populates automatically when you click it. This is the unique name used by the API and managed packages.

1. In the **Search** drop-down list, choose Roles.
2. In the **Available Members** list, select SW Dev Manager, Director Product Management, and Director QA, then click**Add**.
3. Go back up to the **Search** drop-down list, and this time choose Role and Subordinates.
4. In the **Available Members** list, select Recruiting Manager, click **Add**, and save.

Once you’ve defined your group, you can use it to define sharing rules.

## Define a Sharing Rule

Let's use the public group we've created to define a sharing rule for review records. You can define a sharing rule for a single public group, role, or role plus subordinates.

There is already one default public group that encompasses every user in your organization.

1. In Setup, use the Quick Find box to find **Sharing Settings**. This is the same page used to define org-wide defaults.
2. In the **Manage sharing settings for** drop-down list, choose Job Application.

Choosing an object in this drop-down list allows you to focus in on the org-wide defaults and sharing rules for a single object at a time rather than looking at all of them in a long page—a useful thing if you've got a large org with multiple custom objects. Let’s use this Sharing Rules related list to create a sharing rule that applies to both the Job Application and the Review objects.

1. In the Job Application Sharing Rules area, click **New** and give your rule the label Review Records.

The **Rule Name** text box populates automatically when you click it.

1. For the rule type, make sure **Based on record owner** is selected.
2. For **Select which records to be shared**, select Public Groups, then choose Entire Organization.
3. For **Select users to share with**, select Public Groups, then choose Reviewers.
4. For **Select the level of access for the users**,, select Read/Write, then save.

We just created a rule that shares reviews written and owned by any member of the org with all recruiters and hiring managers. Since reviewers and hiring managers all need to read and update reviews, we handled everyone with a single sharing rule and a public group.