**Data Security**

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.

# **Control Access to Org**

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.

We cannot delete a User in SFDC but we can deactivate a user. Once the user is deactivated then that user loses the access to all records including all the records shared to it individually.

Salesforce lets you deactivate users, but not delete them outright.

The reason is because a user can own accounts, leads, and groups, and can be on multiple teams. Removing a user from Salesforce affects many processes in the org. After departure from the org, we don’t want the user to retain access to their account. But merely deleting a user can result in orphaned records and the loss of critical business information.

## **Mass Transfer Records**

Use the Mass Transfer tool to transfer multiple accounts, leads, service contracts, and custom objects from one user to another.

To transfer any records that you don’t own, you need the required user permissions and read sharing access on the records.

**To see the login button at the user page -> Go to Login Access Policies in quick find and then enable Admin can Log in with any user.**

# **Control Access to Objects**

The simplest way to control data access is to set permissions on a particular type of object.

You can set object permissions with profiles or permission sets. Each user is assigned one profile. Users can be assigned one or more 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.

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

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.

## **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 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.

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 set 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. Or, if the user is assigned to a permission set group, you can use a muting permission set to mute selected permissions.

## **Control Access to Fields**

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.

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.

# **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.

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.

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.

# **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 user 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.**

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.

# **Apex Sharing**

Apex managed sharing allows developers to programmatically share records associated with custom objects. When you use Apex managed sharing for any custom object, only users with the “Modify All Data” permission can add or change the sharing on that custom object's records, and the sharing access stays the same even if the record owner changes.

Salesforce displays Apex sharing reasons in the Reason column when viewing the sharing for a custom object record in the user interface. This allows users and administrators to understand the purpose of the sharing.

When working with Apex sharing reasons, note the following:

* Only users with the “Modify All Data” permission can add, edit, or delete sharing that uses an Apex sharing reason.
* Deleting an Apex sharing reason will delete all sharing on the object that uses the reason.
* You can create up to 10 Apex sharing reasons per custom object.
* You can create Apex sharing reasons using the Metadata API.

# **Create a Role Hierarchy**

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.

# **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, by territory, 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. Depending on the group member types available in your org, public groups can be a combination of:

* individual users
* roles
* roles and subordinates
* territories
* territories 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. ok