

## Development: Design Apps for Multiple Users

### Lesson 11: Controlling Access to Records





## Lesson Objectives

By the end of this lesson, you will be able to:

- List the features that affect access to data at the record level
- List the organization-wide default (OWD) settings
- List and define the sharing levels
- Set organization-wide defaults
- Create a role
- Create a public group
- Create a sharing rule
- Manually share records

## Overview of Record Access



## Record Access

The sharing model determines access to specific records

- Who has access?
- What level of access?
- Why they have access?

Access to records is dependent on object CRUD



# Levels of Record Access

Read-only privileges:

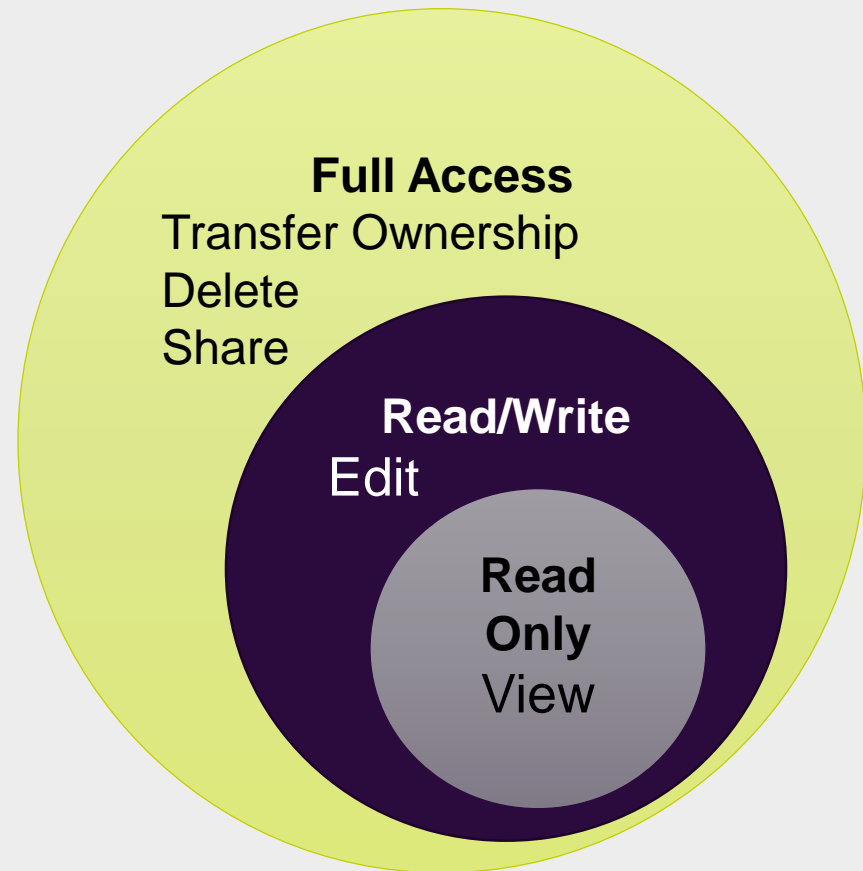
- View

Read/Write privileges:

- View
- Edit

Full Access privileges:

- View
- Edit
- Transfer ownership
- Delete
- Share





## Levels of Record Access

### Full Access:

- Owner field
  - User
  - Queue member
- Above user (has ownership) in role hierarchy
- Profile permissions: "Modify All Data"
- Object permissions: "Modify All"

### Read/Write or Read-only Access:

- Organization-wide default
- Above user (who has read/write or read only access) in role hierarchy
- Manually sharing
- Sharing rules
- Apex sharing
- Profile permissions: "View All Data"
- Object permissions: "View All"



## Compare: Profiles and the Sharing Model

Profiles	Sharing Model
<ul style="list-style-type: none"><li>•Controls access to objects (Candidates, Positions, etc)</li><li>•Controls access to fields (Candidate Name, Min Pay, skill Required, etc)</li></ul>	<ul style="list-style-type: none"><li>•Controls access to records (e.g one candidate, Joe Schmoe, one position, Black Box Tester)</li></ul>

So, a user's profile might specify that a user can see candidates, but the sharing model determines which candidates that user can see.


The sharing model might determine that a user can see Joe Schmoe, but the profile specifies which fields that user can view and edit

Record Ownership





# Record Ownership

 **Position**  
**Black Box Tester**

[Customize Page](#) | [Edit Layout](#) | [Printable View](#)

[« Back to List: Permission Sets](#)

[Position History \[4\]](#) | [Open Activities \[0\]](#) | [Activity History \[0\]](#) | [Notes & Attachments \[0\]](#) | [Job Applications \[1\]](#) | [Job Sites \[0\]](#) | [Approval History \[0\]](#)


**Position Detail**

[Edit](#) [Delete](#) [Clone](#) [Sharing](#)

Title

Black Box Tester

Owner

 [vaishali kunchur](#) [\[Change\]](#)

Most Records have an access Owner

- Exception: Child records in a master-detail relationship inherit access right from parent record

Types of Owner:

- User
- Queues

Record owners have Full Access



## Custom Object Queues

Queues allow groups of users to manage a shared workload more effectively

A queue is a location where records can be routed to await processing by a group member

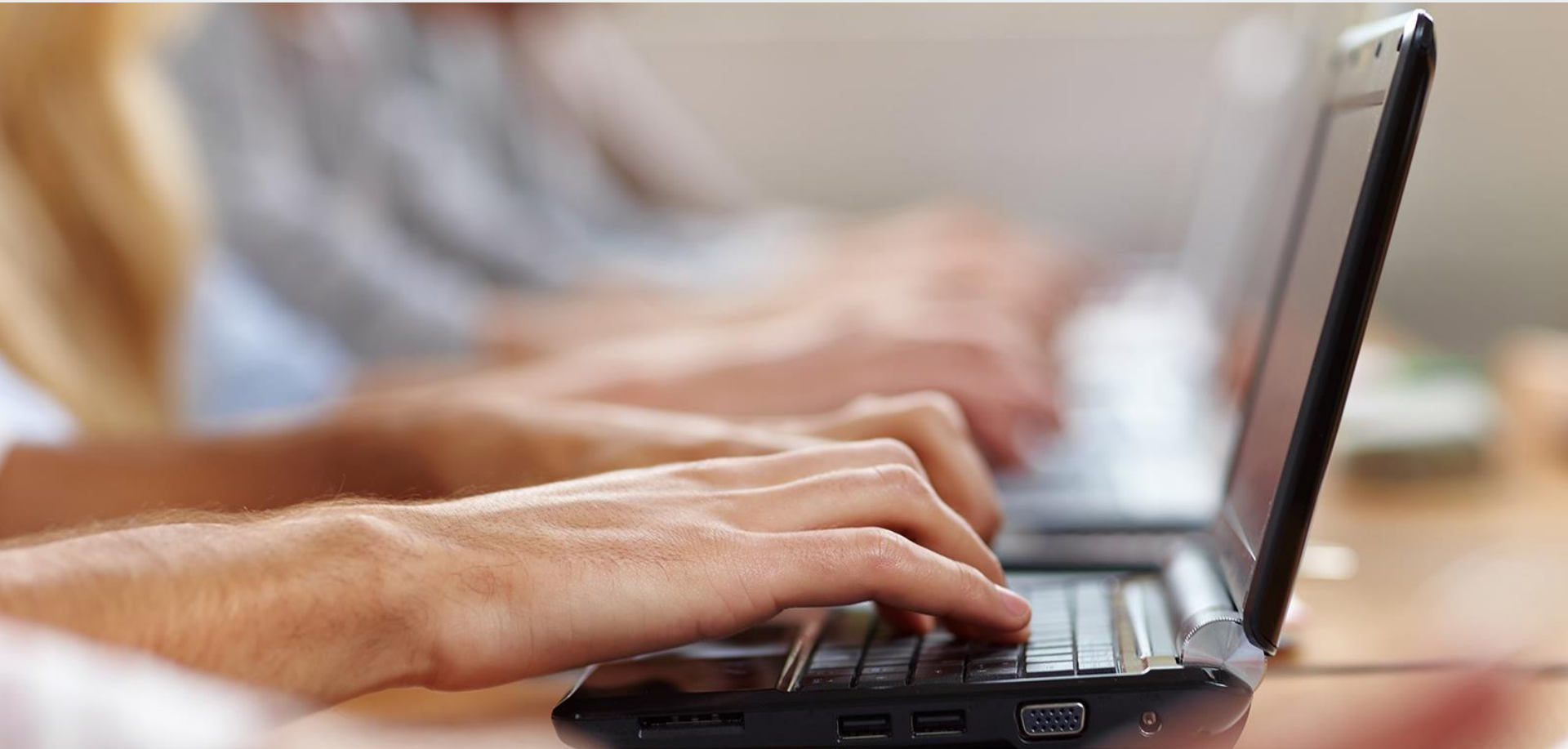
Records remain in the queue until a user accepts them for processing or they are transferred to another queue

Developers can specify the set of objects that are supported by each queue, as well as the set of users that are allowed to retrieve records from the queue

Any member of a queue has the same access to all records in the queue that an owner would have



## Creating Custom Object Queues



## Organization wide Defaults



# What are Organization wide Defaults (OWD)?

Organization wide Defaults are a security setting that defines the baseline level of access to data records that you do not own

They are the only way to restrict access to data in the sharing model

They can be defined for the custom as well as several standard objects

Access levels:

- Public Read/Write (all users can see and edit every record)
- Public Read only (all user can see every record)
- Private (users can only see the records they own)



# Determine how to set OWD for an 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?

YES

Sharing Model=  
PRIVATE

NO

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

YES

Sharing Model=  
PUBLIC READ ONLY

NO

Sharing Model= PUBLIC  
READ WRITE

Questions to ask:

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



## Organization Wide Defaults Considerations

Child records in master-detail relationships inherit their organization-wide defaults from their parents

Child records in lookup relationships have independent organization-wide defaults from their parents

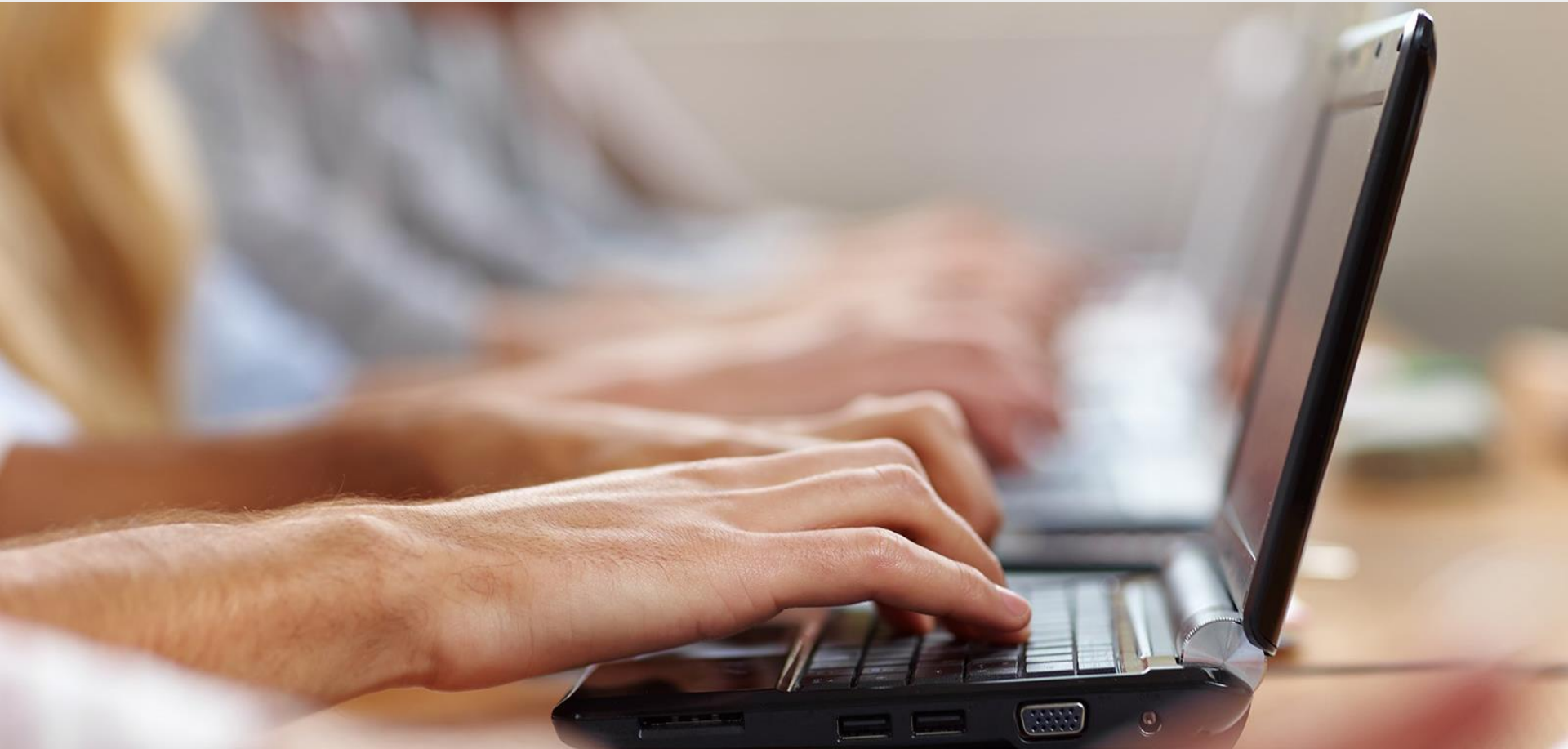
Changing organization-wide defaults can produce unintended consequences; consider your business requirements carefully before setting your organization-wide defaults

Changing organization-wide defaults can potentially delete manual sharing if that sharing is no longer needed

- For example, changing from Private to Public



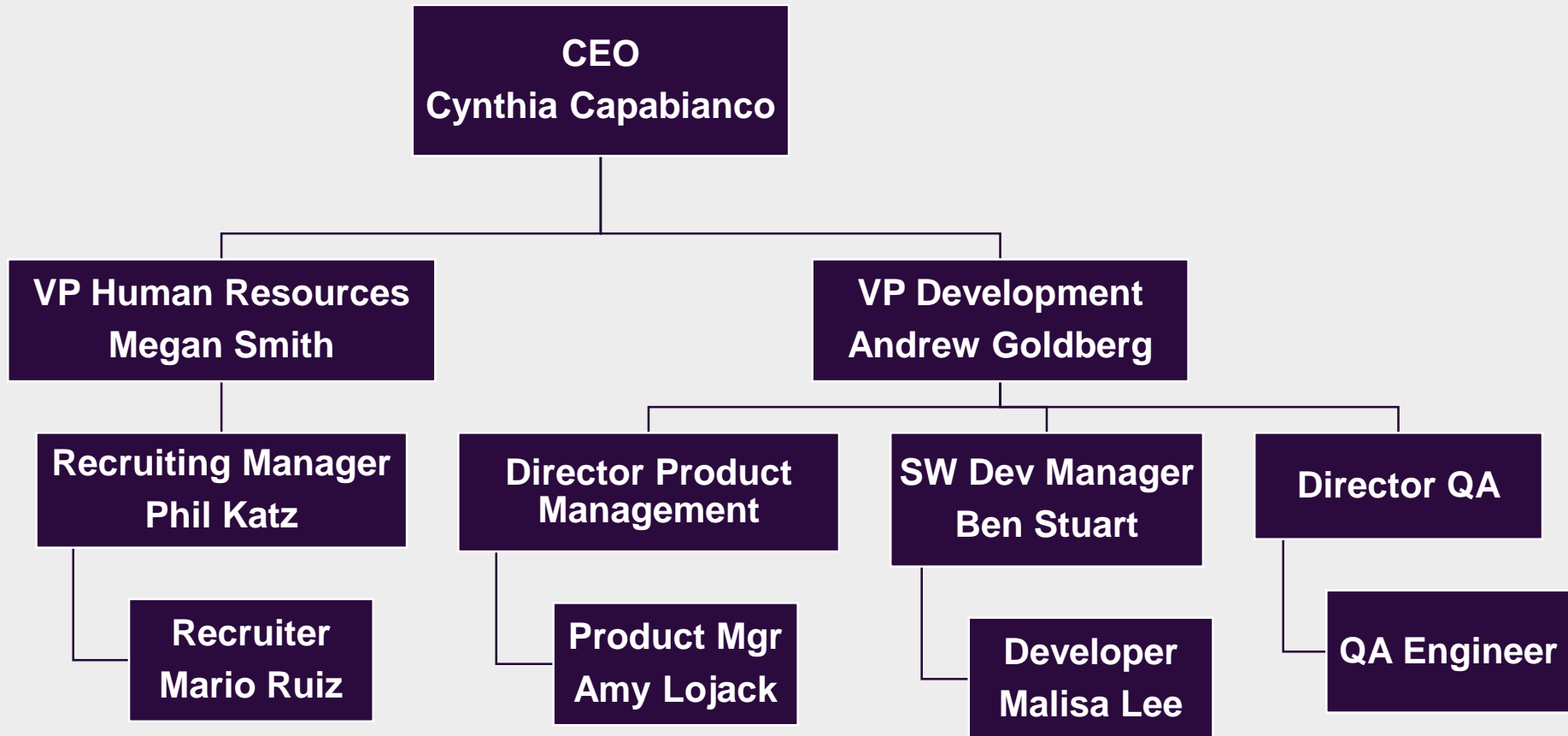
- Setting Organization-Wide Defaults





## Roles and Groups of Users

# Universal Containers Scenario





# What are Roles and Role Hierarchy?

## A Role

- Controls the level of visibility that users have to an organization's data
- A user may be associated

## The Role Hierarchy

- Controls data visibility
- Controls record roll up for reporting
- Users **usually** inherit the special privileges of data owned by or shared with users below them in the hierarchy
- Not necessarily the company's organization chart



## Role Hierarchy Considerations

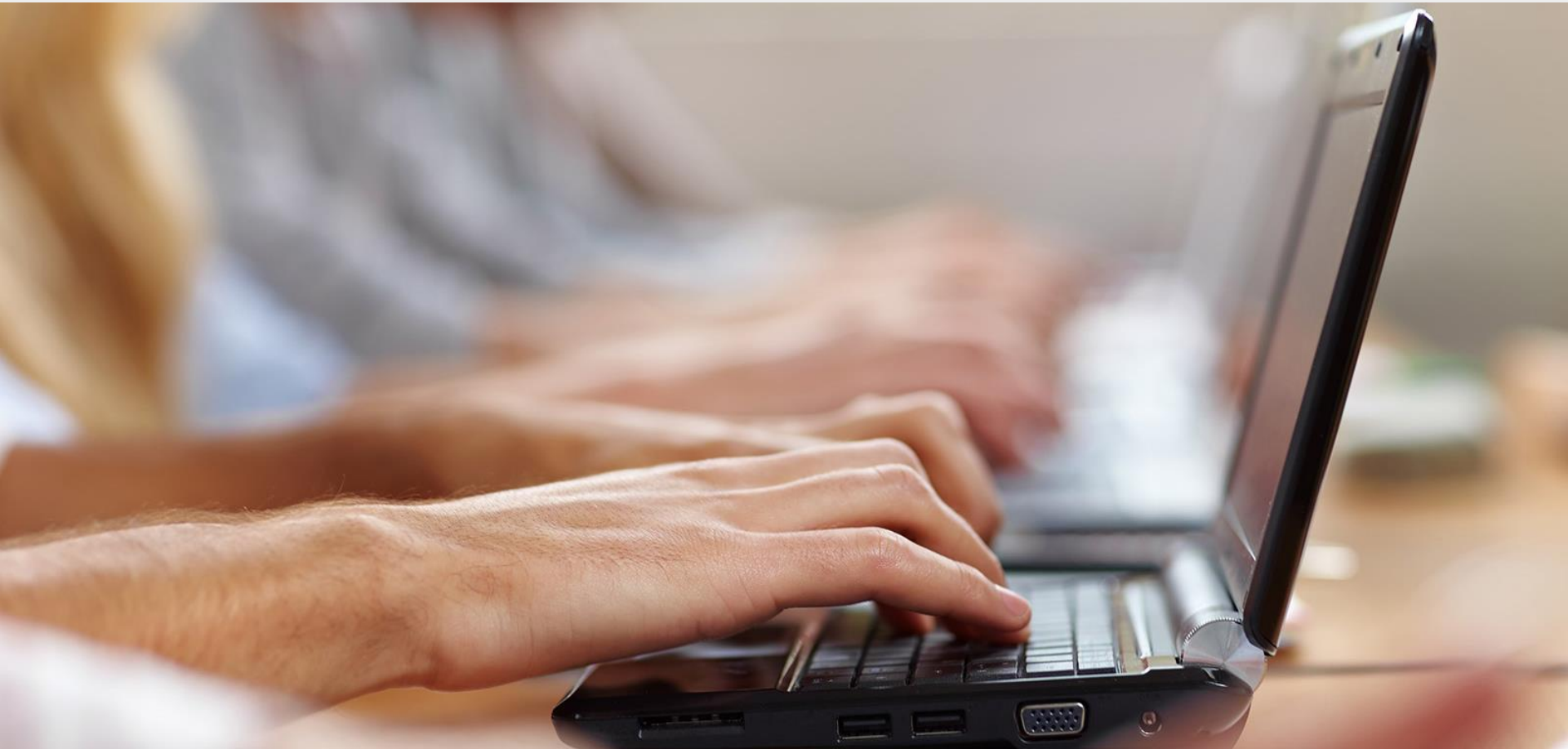
With Standard Objects, access to records rolls up through the Role Hierarchy

With Custom Objects, developers choose whether or not access should roll up through the role hierarchy

- Determined by the Grant Access Using Hierarchies setting on organization-wide defaults



- Implementing a Role Hierarchy





# Public Groups

Public groups are a way of grouping together users for access

- Can be used in a sharing rule
- Can be used to give access to folders

Every organization has a default public group that includes all users, or if portals or sites are used, includes all internal users

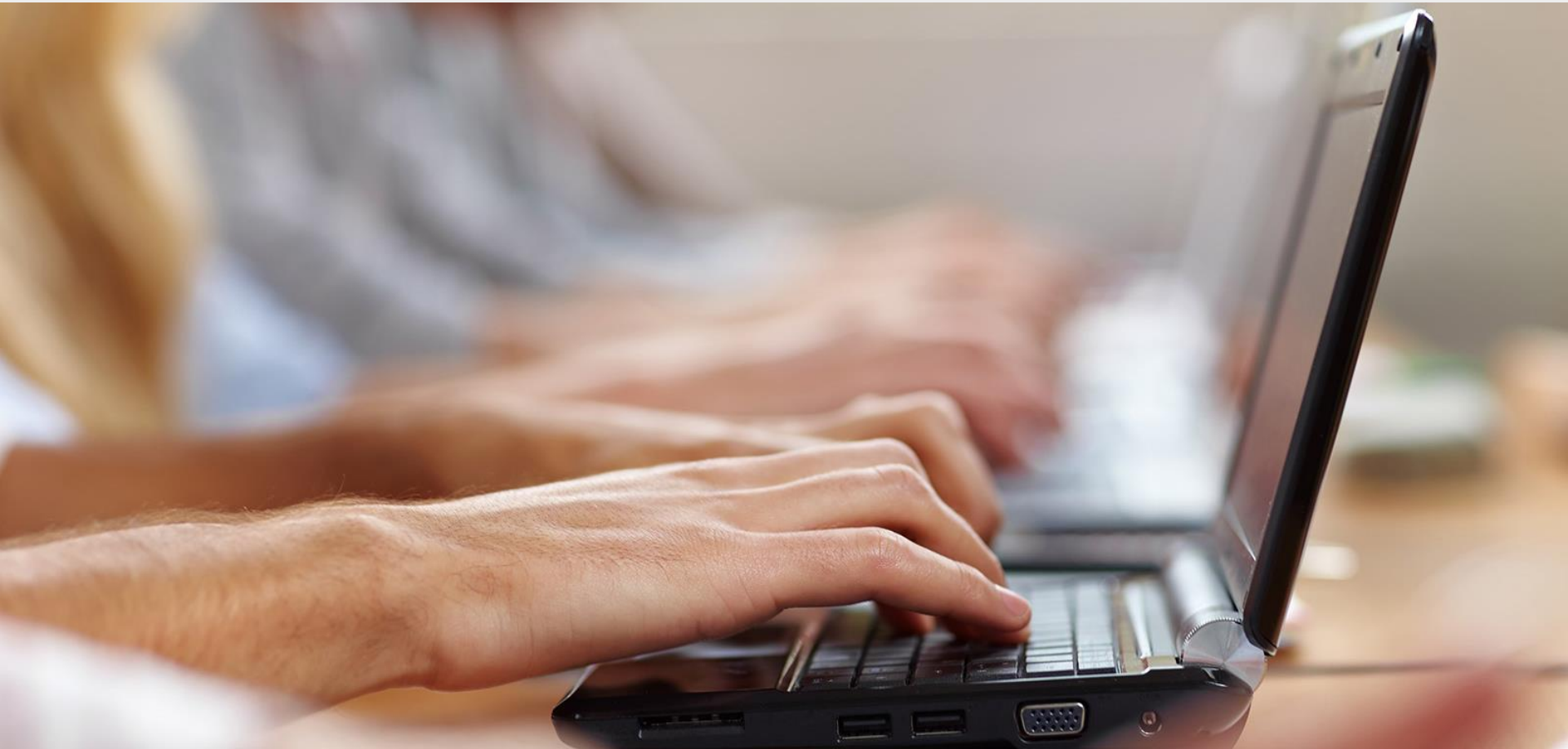
Public groups can be made up of any combination of:

- Users
- Roles
- Roles and Subordinates
- Public Groups

Public group membership is updated when public groups are made up of roles and subordinates, or when a user is added or removed from the role.



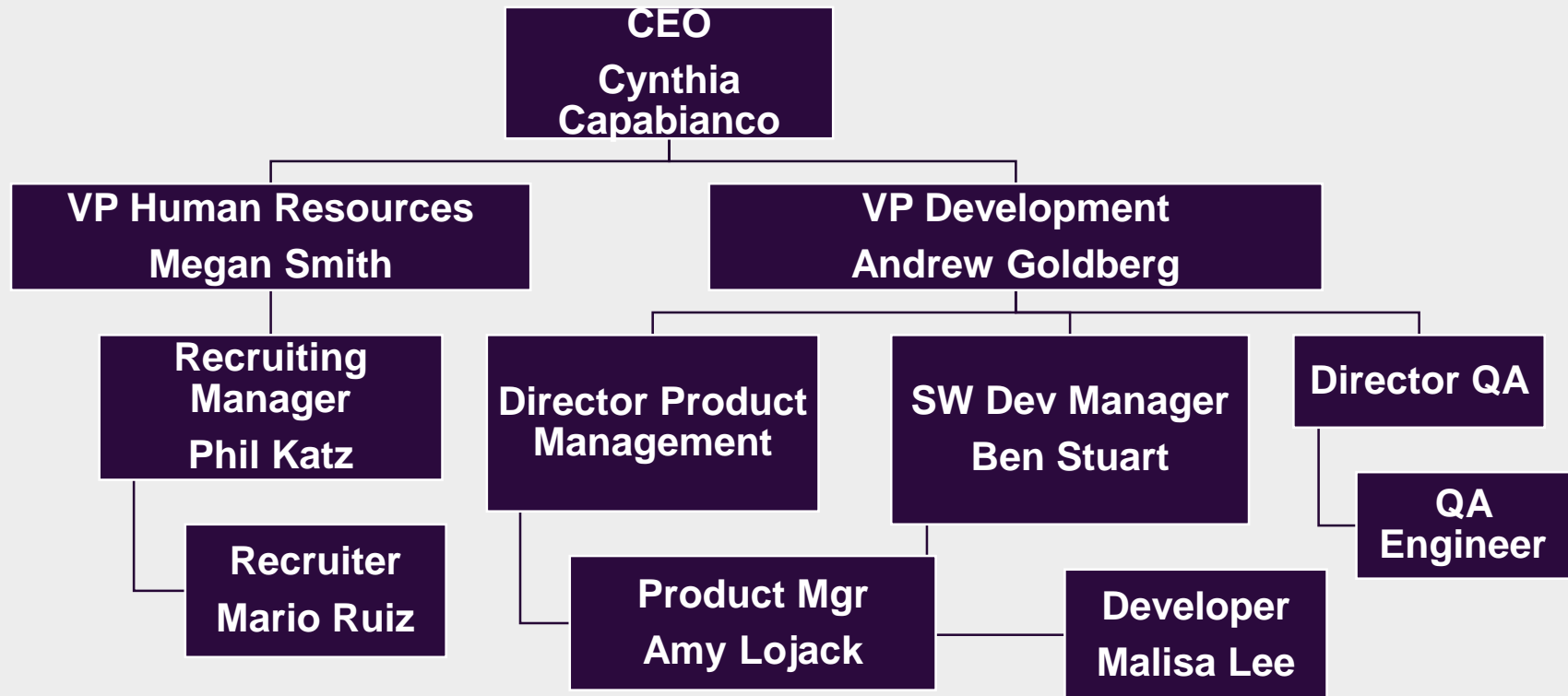
- Creating a Public Group



Sharing



# Universal Containers Scenario



- Megan Smith's team cannot see any reviews owned by Andrew Goldberg's team
- Ben Stuart cannot see reviews written by QA or Product Management
- Melissa Lee cannot see records for candidates she needs to interview



## Manual Sharing

- Used to open up access to a specific record
- Granted by owners, anyone above owners in the role hierarchy, and System Administrators



# Sharing Rules

**Step 1: Rule Name**

Label

Rule Name

i

**Step 2: Select your rule type**

Rule Type

☐ Based on record owner

☒ Based on criteria

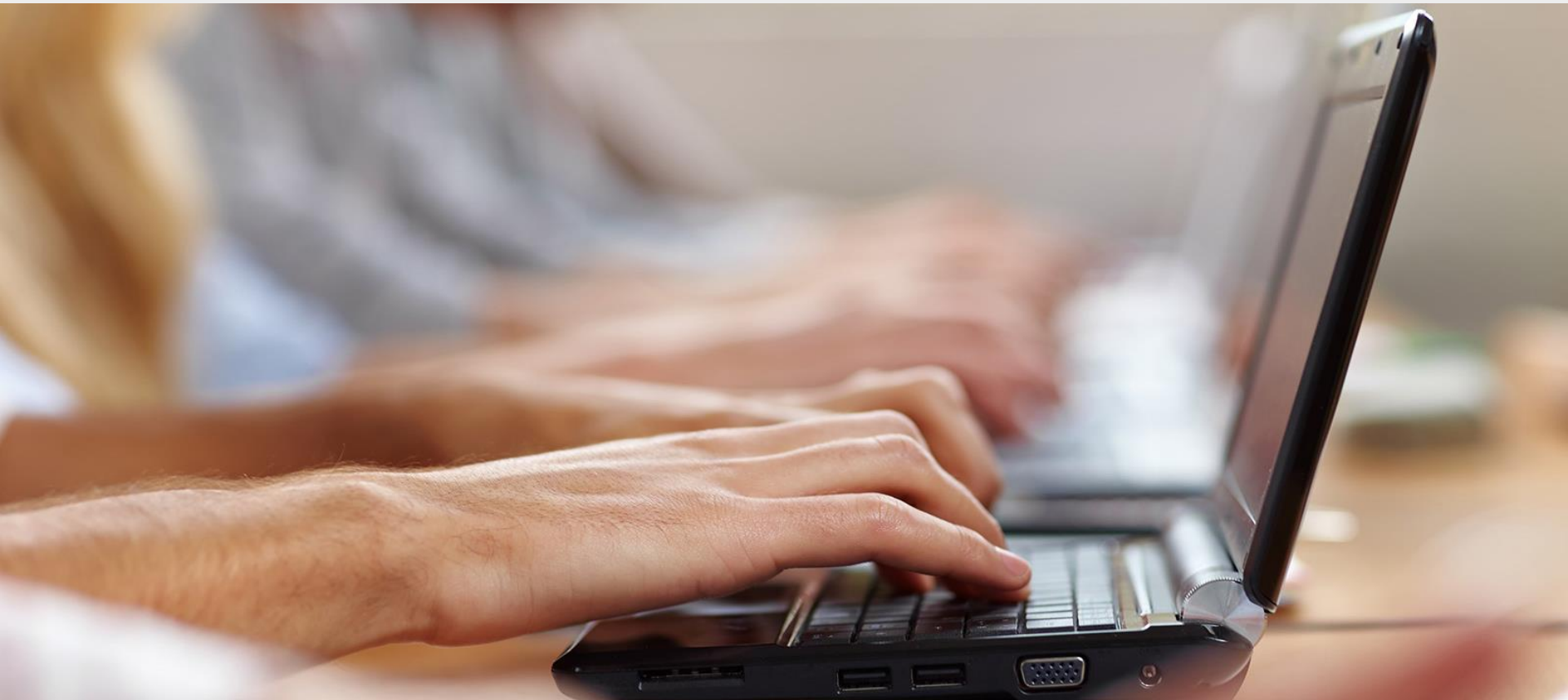
**Step 3: Select which records to share**

Criteria	Field	Operator	Value	
	--None--	--None--		AND

- Can be based on the owner of a record or criteria on that record
- Create exceptions to organization-wide defaults for particular groups of users
- Are used to open access to records
- Cannot be more strict than organization-wide default settings
- Access granted through a sharing rule rolls up through the hierarchy



- Implementing Manual Sharing
- Implementing sharing rules





## Apex Sharing Reasons

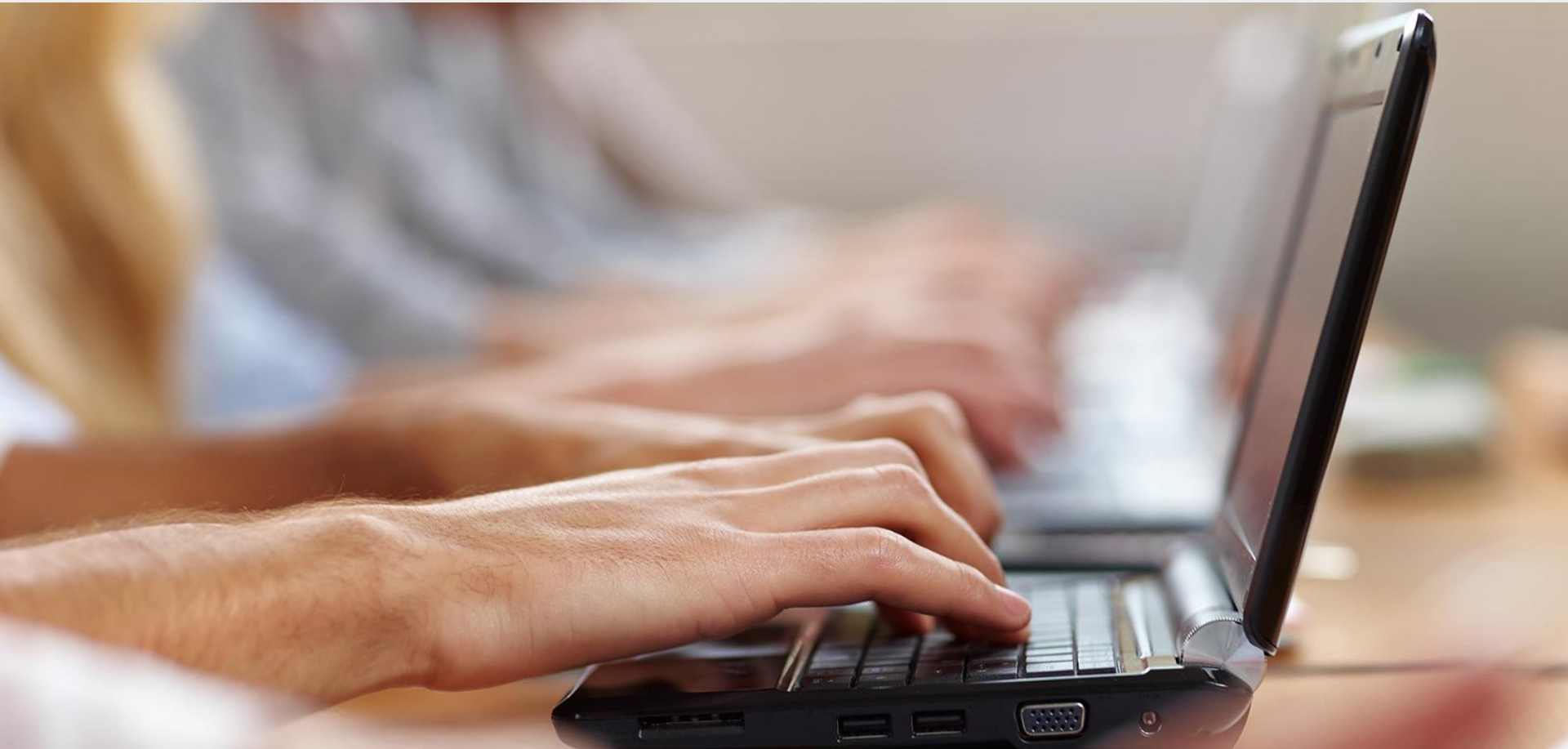
- Clicking the Sharing button on a record displays the various reasons that a user might have access to a record.

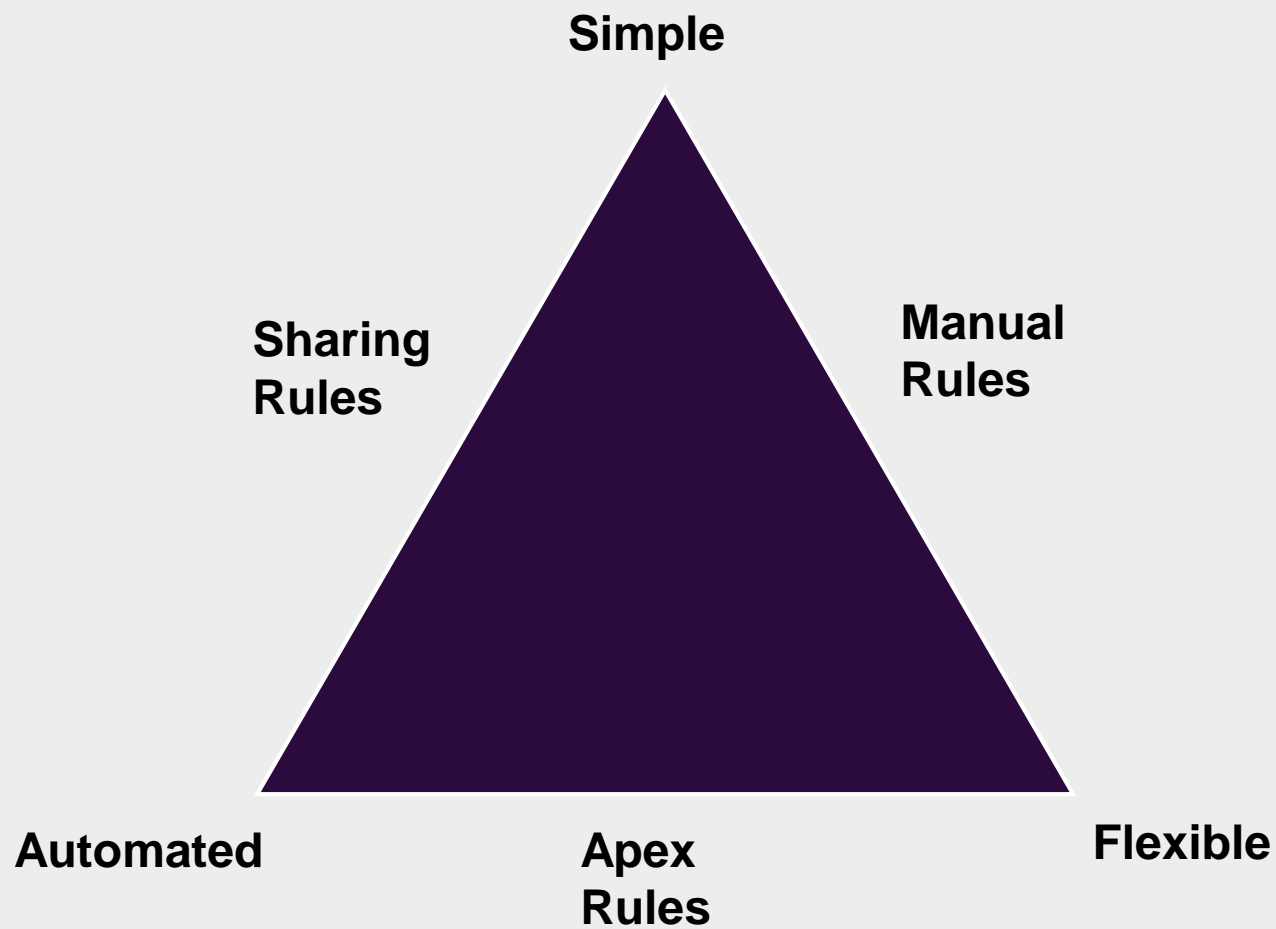
Examples of sharing reasons include:

- Administrator
- Owner
- Custom Object Sharing Rule
- Establishing Apex sharing reasons allows developers to define the reason that a user or group of users might have access to a record
- The Apex sharing reasons describe why users can access a record. Examples might be:
  - Open Position (users have access to a position record because that position is currently open)
  - Hiring Manager (users might have access to a position because they are the hiring manager on that position)
- Apex sharing reasons are defined for custom objects only
- Apex sharing reasons are defined object by object
  - So, positions might have different reasons than candidates



- Creating Apex Sharing Reasons





# Controlling Data Review



## Permissions:

- View All Data

## Record Ownership

## Organization-Wide Defaults

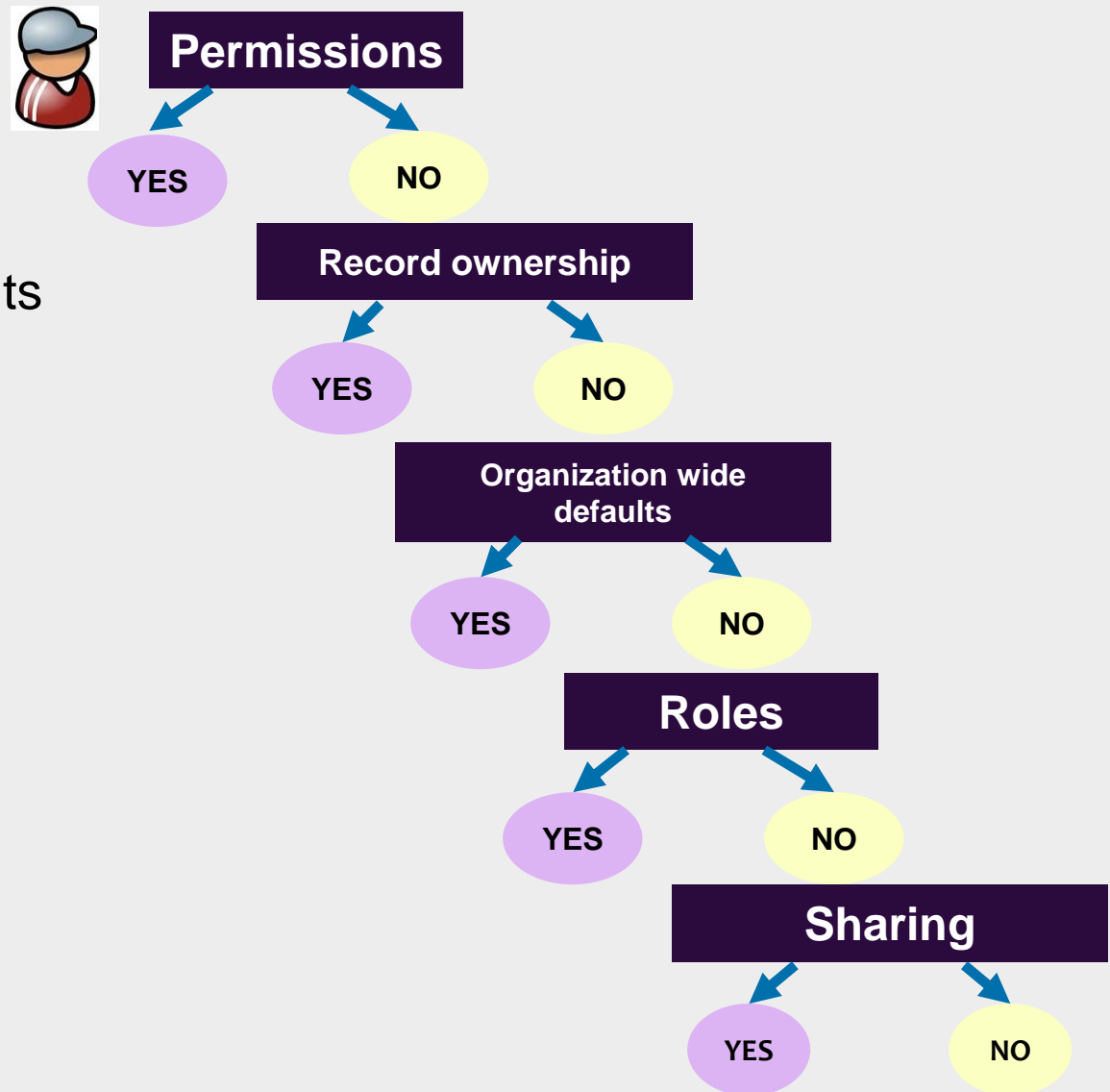
- Read-Only
- Read-Write

## Roles

- Above the Owner?

## Sharing

- Sharing Rules
- Manual Sharing





# Summary



Record Access  
Record Ownership – Users and Queues





## Module Review

- List all the reasons that a user might be able to see a particular record
- What is the difference between a role and a profile?
- Sharing rules are used to restrict access to records
  - True or False
- If a developer wanted to set up his organization so that managers always see records owned by their subordinates, what features should that manager use?
- If a user needs to give access to just one record, what features should that user choose?