

How scope works in practice

Four behaviours explain almost every "why can this person see that?" question.

Roles are additive

An employee with `hub_manager` + `report_viewer` can operate the hub **and** view reports. Neither role restricts the other, and the order you assign them in makes no difference.

Scope filters data automatically

Having access to a module does not mean seeing all of its data. The system filters everything to the employee's scope.

A `report_viewer` whose scope is limited to the brand "Pizza Express" opens the same reports screen as everyone else, but sees only that brand's figures. Nothing is hidden or greyed out — the data simply isn't in their results.

“ This is worth explaining to people directly. Two colleagues comparing the same report and seeing different totals is expected behaviour, not a fault.

Scope applies equally to every role

Unlike older systems, scope is **not** configured per role. If an employee can see stores 1, 2 and 3, that applies to everything they do — hub, reports, metrics, store configuration, all of it.

This is a deliberate simplification. It means you can reason about one employee's visibility by looking at one setting, rather than reconciling several.

Individual exceptions override roles

You can grant or deny one specific permission to one specific employee, optionally with an expiry date.

A deny always wins, whatever their roles grant. See [Individual permission exceptions](#).

Putting it together

When someone reports they can't do something, check in this order:

1. **Do they have a role that grants the action?** → [Access by section](#)
2. **Is there a deny exception on them?** → [Individual permission exceptions](#)
3. **Is the data simply outside their scope?** → [Setting an employee's scope](#)

Most reports resolve at step 3, and look like a missing permission until you check.

Created 2026-09-17 16:28:31 UTC by Fran Sanchis
Updated 2026-09-18 07:48:22 UTC by Fran Sanchis