

# Importing and securing data in ServiceNow

**Team ID :** LTVIP2025TMID28721

**Team Size :** 4

**Team Leader :** Ravanam Teja Lakshman

**Team member :** Reddy Dinesh Babu

**Team member :** Rudrakshula Veera Venkata Satish

**Team member :** Sadhanala Likhith Varma

To solve the problem of \*importing and securing data in ServiceNow\* while \*linking records to employees\* and \*pulling employee details\* (like department) for easier reporting, here's a step-by-step guide that breaks down the process:

You want to:

1. Import records into ServiceNow.
2. Link each imported record to an employee.
3. Automatically populate employee details like \*department\* into the record.
4. Ensure data is \*secured\* (only the right users can view/edit it).
5. Make reporting easier using this linked data.

## Step 1: Prepare the Data Source

Ensure your imported data (CSV, Excel, or external DB) includes a unique employee identifier (e.g., User ID, Email, or Employee Number) to match ServiceNow users.

<C:\Users\likhi\OneDrive\Documents\Book.xlsx>

## Step 2: Create the Target Table or Use an Existing One

If you're importing into a custom table (e.g., u\_custom\_record):

- Navigate to System Definition > Tables
- Create a custom table if needed.
- Add a reference field: Employee (Reference to sys\_user table)
- Add other fields, like Department (String or Reference to cmn\_department)

The screenshot shows the ServiceNow 'Load Data' interface. The left sidebar contains a navigation menu with 'Import Sets' expanded. The main area is titled 'Load Data' and contains a form for creating a new import set table. The 'Import set table' section has 'Create table' selected, with a 'Label' of 'Employee Training' and a 'Name' of 'u\_employee\_training'. The 'Source of the import' section has 'File' selected, with a 'File' field containing 'Choose File | Employee Training.xlsx', a 'Sheet number' of '1', and a 'Header row' of '1'. A 'Submit' button is at the bottom.

---

### Step 3: Use Transform Maps to Import and Link Employee

1. Go to System Import Sets > Load Data
2. Load your data source (CSV or external)
3. Create a \*Transform Map\*:

Target Table: your custom table

Map fields like:

Source field employee\_email → Target field Employee (use a reference lookup to sys\_user)

Auto-populate Department using a \*Scripted Field Map\*:

javascript

```
// Scripted field map for Department field var user = new  
GlideRecord('sys_user'); user.addQuery('email', source.employee_email);  
user.query(); if (user.next()) { target.department = user.department.name; // or  
user.department for reference  
}
```

The screenshot shows the ServiceNow 'Table Transform Map' configuration interface. The left sidebar contains a navigation menu with options like 'Load Data', 'Create Transform Map', 'Run Transform', 'Administration', 'Data Sources', 'ETL Definitions', 'Transform Maps', 'Scheduled Imports', 'Execution Contexts', 'Advanced', 'Import Sets', 'Concurrent Import Sets', 'Concurrent Import Set Jobs', 'Multi Import Sets', 'Progress', 'Transform History', and 'Transform Errors'. The main panel is titled 'Table Transform Map - ...' and shows a 'New record' form. The form includes fields for 'Name' (Employee Transform), 'Source table' (Employee Training [u\_employ...]), 'Application' (Global), 'Created' (empty), 'Target table' (Employee Training Records [u...]), 'Order' (100), and 'Run script' (checkbox). There are also checkboxes for 'Active', 'Run business rules', 'Enforce mandatory fields' (set to 'No'), 'Copy empty fields', and 'Create new record on empty coalesce fields'. A 'Submit' button is at the bottom left of the form.

4. Run Transform.

---

#### Step 4: Auto-Populate Fields with Business Rules (Optional for Future Updates)

If you want the department to stay in sync when the Employee changes:

- Create a \*Business Rule\* on your custom table
- Trigger on \*Insert/Update\*
- Script:

javascript

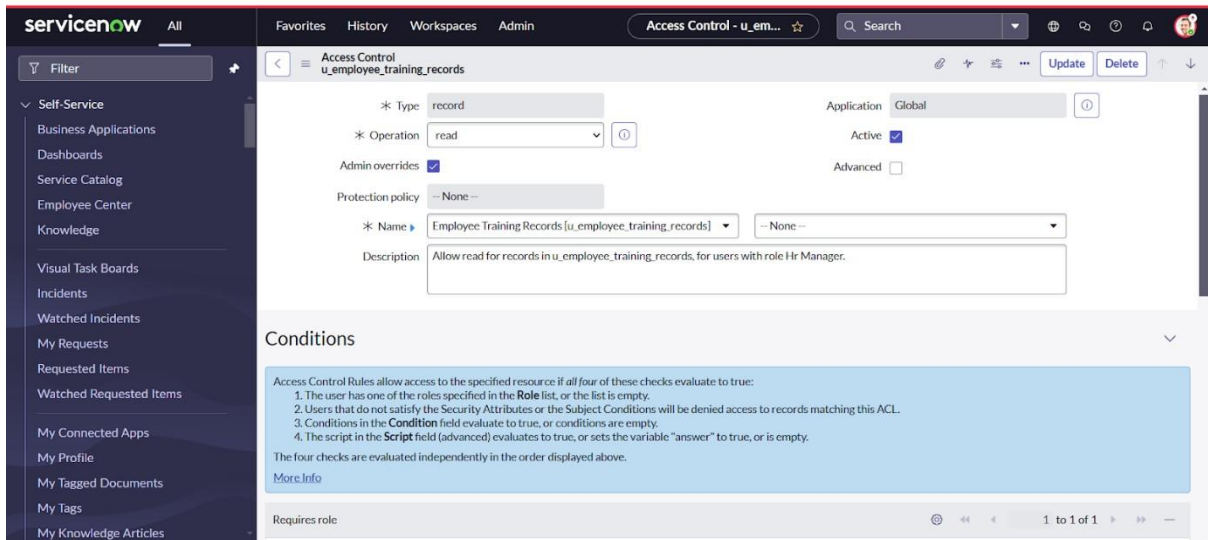
```
(function executeRule(current, previous /*null when async*/) {  
  if (current.employee) {    var user = new  
    GlideRecord('sys_user');    if (user.get(current.employee)) {  
      current.department = user.department.name; // or user.department for reference  
field  
    }  
  }  
})(current, previous);
```

---

### **Step 5: Secure the Data**

Use Access Control Rules (ACLs):

- Navigate to System Security > Access Control (ACL)
- Create ACLs on the table (e.g., u\_custom\_record):



-Record ACLs (read, write, delete)

-Field-level ACLs (hide department if needed)

-Conditions based on roles or ownership:

```
javascript    gs.hasRole('admin') || current.employee ==
gs.getUserID();
```

---

## Step 6: Use Reporting/Performance Analytics

Now that department is stored in the record:

-Use Reports to group by Department

-Use Dashboards to visualize employee-linked metrics

✓ Example Use Case Summary:

- Imported Record: Helpdesk survey, training completion, asset assignment
- Employee Linked By: Email or Employee ID
- Pulled Info: Department (from sys\_user.department)
- Secured By: ACLs (only employee or manager can view)