[Creating Foundation Object Records in Employee Central](https://learning.sap.com/learning-journeys/configure-sap-successfactors-employee-central-core/creating-foundation-object-records-during-implementation_af193388-9df9-4517-a5ce-09273fa4f21f)

## 🔹 What are Foundation Objects (FOs)?

1. Think of **Foundation Objects** as the **building blocks** of your company’s structure inside SuccessFactors.
2. Before you can add employees, you must define your company’s **legal entities, business units, departments, locations, cost centers, jobs, pay structures, etc.**
3. They are like the **master data** that employees "inherit" when you create their profiles.

👉 Example: If you hire someone as “Engineer” in the “ACE Germany Legal Entity” at “Berlin Office,” the details (job level, pay grade, country laws, etc.) come from **Foundation Objects** already set up.

## 🔹 Two Types of Foundation Objects

1. **XML-based (Old / Legacy) FOs**
2. Managed in **Corporate Data Model (XML file)**.
3. Records created/edited in **Manage Organization, Pay, and Job Structures**.
4. Imported/exported with **Import Foundation Data** tool.
5. **MDF-based (Modern / Generic) FOs**
6. Managed inside the application (no XML required).
7. Records created/edited in **Manage Data**.
8. Configured in **Configure Object Definitions**.
9. Imported/exported with **Import and Export Data** tool.

👉 Just remember:

1. XML-based = older, less flexible.
2. MDF-based = newer, customizable, preferred.

## 🔹 Effective Dating

1. Like employee data, FO records are **time-sensitive**.
2. Example:
3. Berlin Office record is effective from **01/01/1990**.
4. If later you move office, a new record starts, and the old one ends automatically one day before.

This allows tracking changes over time without losing history.

## 🔹 Propagation

1. **Propagation = auto-fill.**
2. When you assign an employee to a Job Classification (e.g., “Engineer”), the system **automatically fills related fields** like pay grade, job level, etc.
3. Saves time and ensures consistency.
4. Controlled by **rules** in the system.

## 🔹 Fields in Foundation Objects

1. **Standard fields** → pre-delivered by SAP (like Job Title, Legal Entity Name).
2. **Custom fields** → company can add (like “Shift Type”).
3. **Country-Specific Fields (CSF)** → fields that appear only for a certain country (like Social Security for US, PAN for India).

## 🔹 Associations

1. FO records can be **linked together**.
2. Example:
3. A **Location** (Berlin Office) is associated with a **Legal Entity** (ACE Germany).
4. When you select ACE Germany, only its locations appear in the dropdown.

👉 This keeps data clean and avoids errors.

## 🔹 Picklists

1. Dropdown lists used across objects (like Job Level, Employment Type).
2. Managed in **Picklist Center**.
3. You need permissions like **Picklist Management**.

## 🔹 Types of Foundation Objects

### 1. ****Organization-Related****

1. **Legal Entity** → actual registered company in a country.
2. **Business Unit** → big division of company (e.g., Sales, Storage).
3. **Division** → smaller subdivision inside business unit.
4. **Department** → specific teams or units under divisions.
5. **Cost Center** → accounting/finance unit for tracking costs.
6. **Location** → physical office address (Berlin, San Mateo).
7. **Location Group** → grouping of locations (West Coast offices).
8. **Geo Zone** → regional grouping (North America, EMEA) – used for pay adjustments.

### 2. ****Pay-Related****

1. **Pay Components, Pay Ranges, Pay Groups** → salary structure details.
2. Example: Pay Range = min/max salary for a grade.

### 3. ****Job-Related****

1. **Job Classification** → defines roles (Engineer, Manager) with details like job level, pay grade.
2. **Job Function** → broader grouping of jobs (Engineering, HR, Finance).

## 🔹 Record Administration

1. For XML-based → use **Manage Organization, Pay, and Job Structures** or **Import Foundation Data**.
2. For MDF-based → use **Manage Data** or **Import/Export Data**.
3. **Best practice**: Always download the CSV template before bulk import, otherwise format mismatches cause errors.

## 🔹 Example Exercise Recap

Company expanding into Berlin + new Storage business:

1. Create **Business Unit: Storage**.
2. Create **Location: Berlin Office** (linked to Legal Entity = ACE Germany).
3. Import new **Cost Centers**.
4. Create **Job Classification: Storage Hardware Engineer**.
5. Create **Pay Range** for US and Germany.

## 🔑 Key Takeaways

1. Foundation Objects = company master data (org, pay, job).
2. They are **prerequisite** for adding employees.
3. Two types: **XML-based (old)** vs **MDF-based (new)**.
4. Propagation = auto-fill from FO to employee records.
5. Associations = link objects (Legal Entity ↔ Location).
6. Effective Dating = track changes over time.
7. Manage Data (MDF) is the modern tool; Manage Org/Pay/Job Structures (XML) is for legacy objects.

👉 Quick analogy: Think of **Employee Central** as a house.

1. **Foundation Objects** = bricks and layout (legal entities, departments, jobs).
2. **Employees** = people who move in. You can’t move people into a house without first building its foundation.

## Summary: Foundation Objects in Employee Central

1. **Foundation Objects (FOs)** are the **master data** that define the company’s structure (organization, pay, jobs). They must be created **before adding employees**.
2. **Types of Foundation Objects**:
3. **Organization-related** → Legal Entity, Business Unit, Division, Department, Cost Center, Location, Geo Zone.
4. **Pay-related** → Pay Components, Pay Groups, Pay Ranges.
5. **Job-related** → Job Classification, Job Function.
6. **Two Formats of FOs**:
7. **XML-based (Legacy)** → Managed in Manage Organization, Pay, and Job Structures. Import with Import Foundation Data.
8. **MDF-based (Modern/Generic)** → Managed in Manage Data. Import/export with Import and Export Data.
9. **Effective Dating** → Records are time-based; new records start where the old ones end (keeps history).
10. **Propagation** → Auto-fills employee data (e.g., selecting “Engineer” job classification auto-fills job level, pay grade, etc.).
11. **Associations** → Objects can be linked (e.g., Location ↔ Legal Entity). Keeps dropdowns clean and relevant.
12. **Fields**:
13. Standard (delivered by SAP)
14. Custom (defined by company)
15. Country-Specific Fields (e.g., PAN for India, SSN for USA).
16. **Picklists** → Dropdowns for consistent data entry (e.g., Job Level options). Managed in Picklist Center.
17. **Administration**:
18. Use the right tool based on XML vs MDF object.
19. For bulk updates, always download/import using the system’s CSV template.

✅ **Most Important Points to Remember (for interviews & practice):**

1. Foundation Objects = **base data** of company, required before employee data.
2. Two types: **XML-based** (old) vs **MDF-based** (modern).
3. Managed in **Manage Org/Pay/Job Structures (XML)** or **Manage Data (MDF)**.
4. **Effective dating** ensures historical tracking.
5. **Propagation** auto-fills employee records from FOs.
6. **Associations** link objects together (Legal Entity → Location, Business Unit → Division, etc.).

# **Configuring Legacy and MDF Foundation Object Definitions within SAP SuccessFactors Employee Central Core**

Let’s break down what you pasted on **Configuring Legacy and MDF Foundation Object Definitions**:

### 1. ****Corporate Data Model****

1. Think of the **Corporate Data Model** as a blueprint that defines **company-wide structures** in SAP SuccessFactors.
2. It is **XML-based** → meaning the data is stored and managed in XML files.
3. What you can do in this model:
4. Define **labels** (like field names you want displayed).
5. Control which fields are **visible** and which are **mandatory**.
6. Add **custom fields** if your company needs something extra.
7. Define **associations** (how one object connects with another, e.g., an employee belongs to a department).
8. Attach **business rules** (like validations, defaults).

👉 **Where it’s managed?**

1. Admins can **import/export** this data model using Provisioning (backend system) or Admin Center → Import/Export Corporate Data Model.

👉 **Why it’s useful?** Let’s say in your company, you want the “Standard Hours” field on Location to be renamed to “Standard Weekly Hours” (to match how HR speaks). You edit the Corporate Data Model XML, re-import, and boom—it shows up everywhere with the new label.

### 2. ****Metadata Framework (MDF) Objects****

1. MDF is a newer, **flexible way to manage objects** in SuccessFactors (not XML-based).
2. You use **Configure Object Definition** in Admin Center to manage these.
3. Each object has:
4. **Structure** (fields, relationships).
5. **Details** (labels, rules, visibility).

👉 **Example**: Legal Entity object.

1. You can see its structure in Configure Object Definition.
2. If you go to **Manage Data**, you’ll see how the fields you configured appear when entering data.

👉 **Main advantage**: More user-friendly than editing XML and allows you to build **custom objects**.

### 3. ****Custom Generic Objects****

1. SAP delivers many standard objects (like Company, Location, Division).
2. But sometimes a company needs **extra info** beyond what SAP provides.
3. That’s when you create **Custom Objects** (called Generic Objects) using MDF.

👉 **Example**:

1. SAP standard organization hierarchy: Legal Entity → Business Unit → Department.
2. But suppose your company wants an additional “Region” level in between.
3. You can create a **custom object: Region** using MDF.

👉 **Characteristics of custom objects**:

1. Can have **effective dating** (records active from a certain date).
2. Can be **secured** (only certain roles can access).
3. Can be **associated** with other objects.
4. Can be available for **self-service** (end-users can update themselves).

### 4. ****Terminology Review****

1. **Foundation Objects**: Core company structures (like Legal Entity, Location, Job Classification).
2. **Legacy Foundation Objects**: XML-based (found in Corporate Data Model).
3. **MDF-Based Foundation Objects**: Same concept but built using MDF (newer method).
4. **Generic Objects**: Any MDF-based object (standard or custom).

### 5. ****Hands-On Tasks****

These are exercises you’d do as an admin:

1. **Task 1: Customize Location Object**
2. Export Corporate Data Model XML → Edit “Standard Hours” label → Change to “Standard Weekly Hours” → Re-import → Verify in system.
3. **Task 2: Create Picklists**
4. Picklists = dropdown values.
5. Example: Scope (Global, Local), Region (AMER, EMEA, APAC).
6. Create them in Picklist Center.
7. **Task 3: Customize Job Classification Object**
8. Go to Configure Object Definition → Update fields/labels → Save.
9. **Task 4: Customize Legal Entity Object**
10. Same process as Job Classification → Add new fields if required → Save.
11. **Task 5: Verify Changes**
12. Use Manage Data → Create mock records (Job Classification, Company, Business Unit).
13. Check if new labels and fields show up properly.

✅ **Summary (Key Points to Remember)**

1. **Corporate Data Model** = XML-based → For configuring legacy Foundation Objects.
2. **MDF (Metadata Framework)** = New flexible method → Configure via Configure Object Definition.
3. **Foundation Objects** = Core company structures (Legal Entity, Location, Job Classification).
4. **Legacy vs MDF** = Legacy = XML; MDF = Metadata Framework.
5. **Generic Objects** = Any MDF object (standard/custom).
6. **Picklists** = Dropdown lists for values like Region, Scope.
7. Hands-on = Change labels, create picklists, add fields, verify in Manage Data.

Do you want me to also **give you real-world analogies** (like relating SAP objects to a real company setup) so that you don’t just memorize but actually visualize it better?

# **Configuring Foundation Object Associations in Employee Central**

Perfect 👍 You pasted the **Foundation Object Associations** lesson. Since you said you want **detailed explanation first, then summary at the end**, let’s go step by step:

## 🔹 What are Foundation Object Associations?

1. **Foundation Objects (FOs)** are the building blocks of company structure (Legal Entity, Location, Division, etc.).
2. **Associations** = Relationships between these FOs.
3. They tell the system how different objects connect and which values should be available to employees based on hierarchy.

👉 Example:

1. ACE Corp has **30 Locations**, but only **13 belong to the U.S.**
2. When you select **Legal Entity = ACE USA**, the system should show **only the 13 U.S. Locations** (not all 30).
3. This is possible because of the **association** between **Legal Entity** and **Location**.

So, **associations = filtering + linking objects together**.

## 🔹 Multiplicity (How many records can be associated?)

There are **two main types** of multiplicity:

1. **One-to-One**
2. One record is linked to **exactly one other record**.
3. Example: Seattle Location → belongs to **only one Location Group** (NA\_West).
4. Displays as a **picklist** (since you can only choose one).
5. ⚠️ Note: SAP recommends avoiding custom one-to-one unless necessary.
6. **One-to-Many** (most common)
7. One record can be linked to **multiple records**.
8. Example: Job Classification (parent) → Multiple Country-Specific Job Classifications (child).
9. Displays as a **separate section** at the bottom, where multiple records can be added.

👉 Important: One-to-many doesn’t mean you must choose many. It just allows multiple links.

## 🔹 Association Types

There are 4 categories, based on whether the objects are **XML** (legacy) or **MDF** (modern):

1. **MDF → MDF**
2. Both objects are MDF-based.
3. Example: Business Unit filters Divisions.
4. Steps:
5. Configure Object Definition → Add association in Division object → Multiplicity One-to-Many → Destination = Business Unit → Type = Valid When.
6. Align in Employee File (via **BCUI** or Succession Data Model).
7. (If Position Management enabled) → Align in Position Object.
8. **XML → MDF**
9. Higher-level object is **MDF**, lower-level is **XML**.
10. Example: Legal Entity (MDF) filters Location (XML).
11. Steps:
12. Add association in **Corporate Data Model XML**.
13. Set multiplicity = ONE\_TO\_MANY.
14. Destination = LegalEntity (MDF).
15. Upload updated XML back.
16. Align in Employee File + Position Management.
17. **XML → XML**
18. Both objects are XML-based.
19. Example: Location Group filters Location.
20. Steps:
21. Add <association> line inside <hris-associations> of the Location object in Corporate Data Model.
22. Align in Employee File by adding field criteria.
23. Align in Position Management if enabled.
24. **MDF → XML**
25. Higher-level is MDF, lower-level is XML.
26. Direct linking is **not allowed** → you must use a **wrapper object** (MDF object that represents the legacy XML FO).
27. Example: Garage (custom MDF) associated with Location (XML).
28. Use Location Wrapper as destination object.
29. Association Type = Composite.
30. Create field criteria in Employee File (like custom-string4).

## 🔹 Composite vs Valid-When Associations

1. **Composite** = Parent-child relationship (child can’t exist without parent).
2. Example: Garage → Location.
3. **Valid-When** = Defines hierarchy and filtering.
4. Example: Division → filtered by Business Unit.

## 🔹 Hands-on Example (Garage Object Exercise)

ACE Corp wants to track **garages** linked to locations.

1. Create **Garage object** in Configure Object Definition:
2. Fields: Garage Name, Car Capacity, etc.
3. Association: Garage → Location Wrapper (One-to-Many, Composite).
4. Create garage records in Manage Data.
5. Gala Garage (125 cars, linked to San Mateo).
6. Amidi Garage (60 cars, linked to Arlington).

## ✅ Summary (Key Points)

1. **Associations = relationships** between objects → used for filtering + hierarchy.
2. **Multiplicity**:
3. One-to-One → Only one allowed.
4. One-to-Many → Multiple allowed (most common).
5. **Association Types**:
6. MDF → MDF (configure in Object Definition).
7. XML → MDF (configure in Corporate Data Model).
8. XML → XML (add association in XML).
9. MDF → XML (use Wrapper object).
10. **Composite = Parent-Child**, **Valid-When = Filtering/Hierarchy**.
11. Employee records & Position Management must always be **aligned** with FO associations.

Do you want me to also **draw a simple diagram** (objects + arrows showing associations) so you can visually connect these relationships?

# **Configuring New Country/Region Fields for MDF Foundation Objects**

Perfect — thanks for pasting 🙌 Let’s break this down **step by step in detail first** (since you’re new to SAP SuccessFactors and MDF), and then I’ll give you a **short summary at the end** with the exam/real-life key points.

# 🔹 Configuring Country/Region Fields for MDF Foundation Objects

### 1. ****Country-Specific Fields (CSF)****

1. SAP allows you to configure different **fields based on country**.
2. Example: A **Business Address** in the US may need State, ZIP Code, Federal Reserve Bank, while in China it may need Province, Region, Postal Code.
3. This is handled in the **CSF Corporate Data Model** (for legacy objects) OR in MDF (for modern objects).

👉 In short: Different countries = different required fields.

### 2. ****CSF Corporate Data Model (Legacy Foundation Objects)****

1. Used for **old XML-based Foundation Objects**.
2. It defines **which country-specific fields appear** for objects like Location, Business Address, etc.

Example:

1. In CSF Corporate Data Model → Corporate Address element → configure fields separately for US and China.

### 3. ****CSF for MDF Foundation Objects****

1. For **MDF-based Foundation Objects** (modern ones, like **Legal Entity**).
2. Example:
3. When creating a new Legal Entity record in Manage Data → you must assign a country.
4. The **assigned country** decides what **extra fields appear** (country-specific ones).

👉 Pre-delivered standard fields exist for some countries:

1. **United States** → Federal Reserve Bank, Fed Reserve Bank District, EEO Company Code.
2. **Germany** → Tax Unit, Social Accident Insurance, etc.

If a country doesn’t have pre-delivered fields (like the Philippines), you can **create them manually**.

### 4. ****How to Add Country-Specific Fields (High-Level Steps)****

1. **Create a child object** for country-specific fields (e.g., Legal Entity Philippines).
2. **Associate it with the parent object** (Legal Entity).
3. Add **conditions** so the fields only appear for the right country (e.g., countryOfRegistration = PHL).

👉 This ensures only **Philippines fields appear for Philippine Legal Entities**.

### 5. ****Exercise: France Example****

Business case: ACE is expanding to France, wants a custom address format.

**Steps:**

#### Task 1: Create Picklist for France (Address Type)

1. Go to **Picklist Center** → Create New → Add values: Physical, Mailing, Deliveries, Other.

#### Task 2: Customize France’s Corporate Address

1. Export **Country-Specific Corporate Data Model** (CSF).
2. Open XML → find country="FRA".
3. Add/replace corporate address fields.
4. Upload modified CSF back.
5. Test by creating a new Location with Country = France → check if fields appear correctly.

#### Task 3: Customize Legal Entity for France

1. Go to **Configure Object Definition → Legal Entity France**.
2. Edit (Make Correction) → add new fields required.
3. Test in **Manage Data → Create New Legal Entity → Country = France** → check fields.

### 6. ****Foundation Data Translation****

Why? → Because SuccessFactors is multi-language, and foundation data (like Division names, Location names) should be shown in the user’s preferred language.

#### How it works:

1. **For Legacy Foundation Objects** →
2. Add/change translations directly in **UI** (good for small fixes).
3. OR Import/export CSV for **bulk translations**.
4. Process:
5. Export FoTranslation object → CSV contains multiple language columns.
6. Translator fills translations offline.
7. Import back → updates translations.
8. **For MDF Foundation Objects** →
9. Translations are included directly in the import template of the object.

👉 Rules:

1. No incremental upload → Every new upload replaces old translations → must consolidate into a single CSV.
2. Display logic:
3. Show in user’s logon language (if available).
4. Else → default company language.
5. Else → fallback to US English.

# ✅ Summary (Key Points to Remember)

1. **Country-Specific Fields (CSF):** Allow different data formats for different countries.
2. **CSF Corporate Data Model:** Used for legacy foundation objects.
3. **MDF Objects (like Legal Entity):** Country assigned at creation decides which fields show.
4. **Adding new fields for a country:** Create child object + composite association + condition (ISO code).
5. **France Example:** Create picklist, update CSF XML for address, customize Legal Entity fields.
6. **FoTranslation object:** Used to translate foundation data into multiple languages.
7. **Translation logic:** Logon language → default language → US English.

👉 Do you want me to also **map this into a step-by-step flow diagram** (like “User creates Legal Entity → Country chosen → country-specific fields appear”) so you can visualize the process better?