Development: Data Management

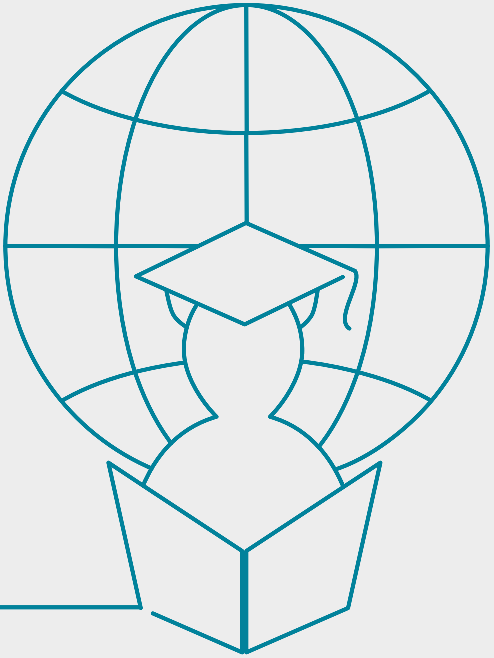
Lesson 20: Introduction to Upsert

salesforce global strategic consulting partner

Lesson Objectives

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

- Define Upsert
- Define external ID
- List typical use case where upsert is useful
- List typical use case where upsert relationship is useful
- Explain how external IDs are used in combination with upsert
- Explain some advantages of making use of an external ID field in a Force.com application



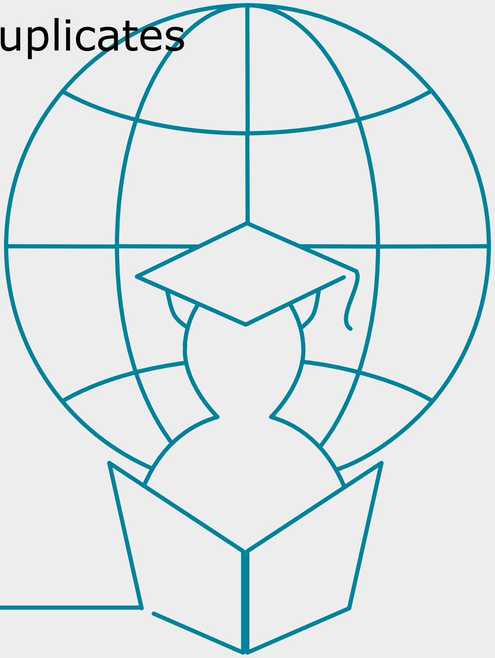
20.1: Upsert
Upsert

Upsert is an API function that combines insert and update into a single call

Upsert uses an ID or external ID to determine whether to create a new record or update an existing record

- If the ID is not matched, then a new record is created
- If the ID is matched once, then the existing record is updated
- If the ID is matched multiple times, then an error is reported

Use Upsert when importing data to prevent the creation of duplicates



20.2: External IDs
External IDs

External ID is a flag that can be added to a custom field to indicate that it should be indexed and treated as an ID

Custom index on any custom field of type **Text**, **Number** or **Email**

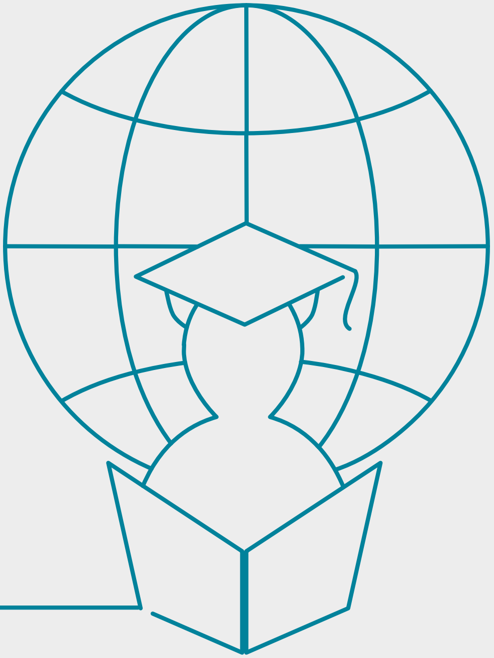
Available on all objects that support custom fields

User-defined cross-reference field

Why is it important?

- Increase report and API SOQL performance
- Used with upsert to easily integrate apps with other systems

An object can have three (3) **External ID** fields

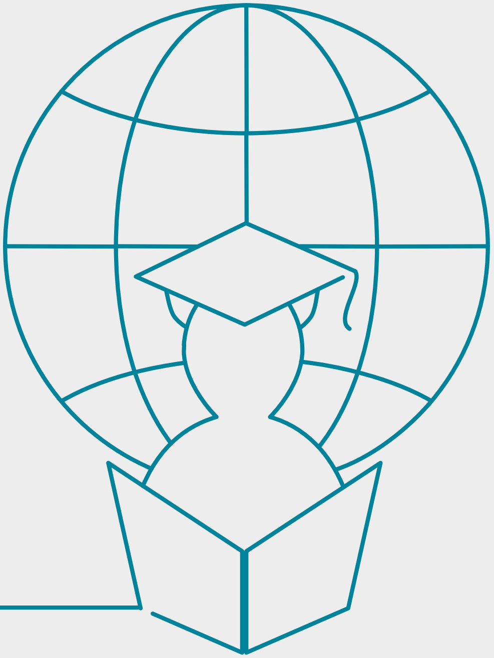


20.3: Typical Use Cases Where Upsert is Useful
Why Use Upsert and External IDs

Typically used for migrations and integrations between Salesforce and other systems

Useful to have external foreign key to link data in both systems

Helps in auditing flow of information across systems

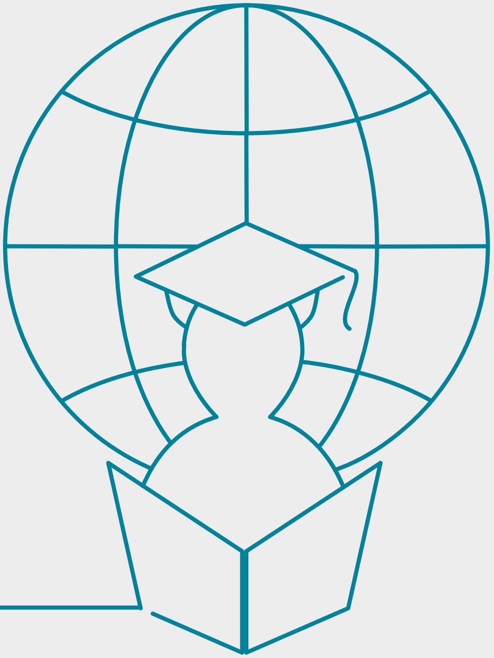


20.3: Typical Use Cases Where Upsert is Useful
Upsert and External ID Typical Use Case

Legacy Position records in an old Recruiting Application have a Legacy Position Number.

Create a custom text field called Legacy_Position_Number on the Salesforce Position object. Flag the new field as an External ID.

When importing Position data from Legacy system to Salesforce, use the Upsert function in combination with the Legacy_Position_Number field instead of using the Create or Update API calls.



The information contained in this document is proprietary and confidential.

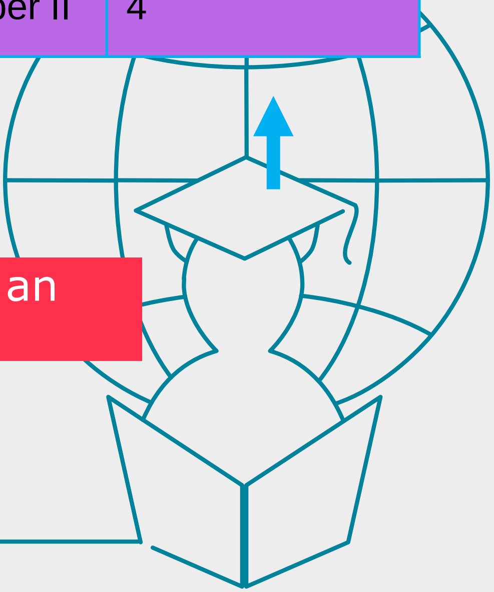
20.3: Typical Use Cases Where Upsert is Useful
Upsert and External ID

ID	NAME
1	Chief Architect
2	SW Developer I
3	QA Engineer
4	SW Developer II

Salesforce ID	NAME	LegacyID__c
a0530000005qe8c	Chief Architect	1
a0530000005qe8N	SW Developer I	2
a0530000005oYCa	QA Engineer	3
a0530000005oYCP	SW Developer II	4

Upsert (LegacyId__c)

Note that ID corresponds with LegacyID__c, an External ID



20.4: Upsert with Relationships
Upsert with Relationships

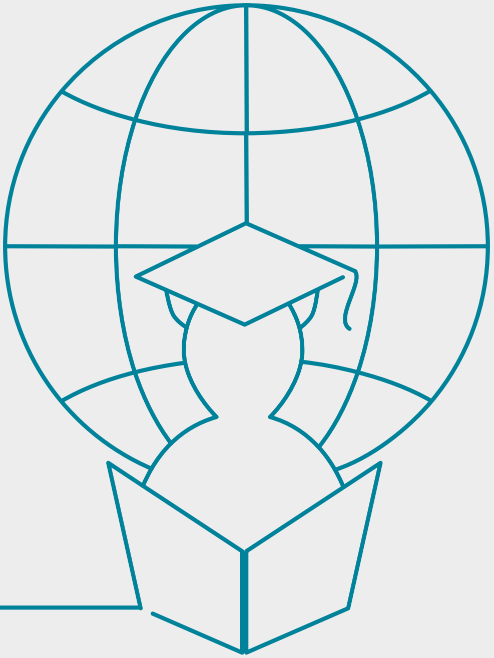
Allows use of relationships defined in legacy systems when importing data into Force.com apps

Configure upsert action to traverse object relationships defined in your Force.com app, but use external IDs from legacy system to discover Force.com record IDs

No need to know Force.com record IDs to load data!

Very convenient for integrations and migrations

- Shifts more of the work to Salesforce



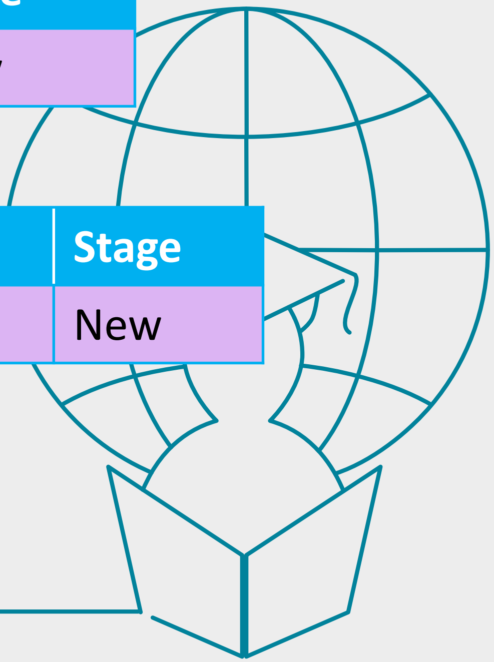
20.4: Upsert with Relationships
Example: Upsert with Relationships

Example Job Application CSV using Upsert w/Relationships: Specify the External ID field of the related object so there is no need to know the Salesforce ID of the related record

Do This…

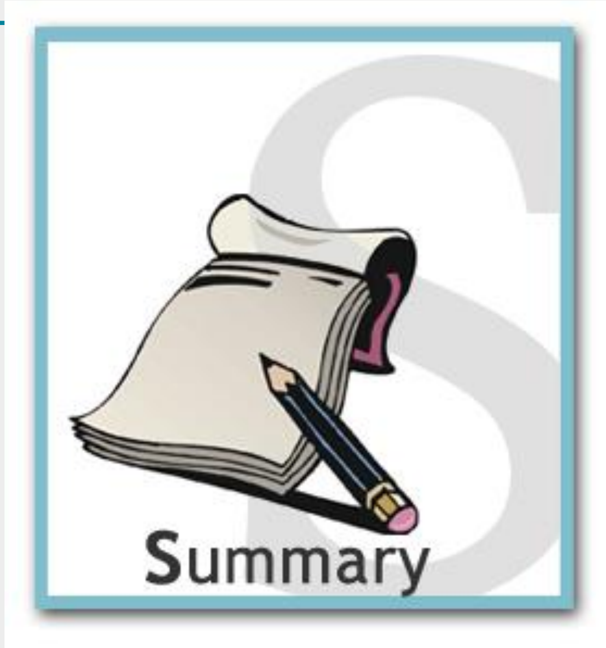
OLD_Position	OLD_Candidate	OLD_JobApp	Status	Stage
p222	C444	j1	Open	New

Position_ID	Candidate_ID	OLD_JobApp	Status	Stage
a00x00000004WBo	a02x0000000iu8	j1	Open	New



Summary

Upset
External IDs
Upsert and External ID
Upsert with Relationships



Module Review

What is the advantage of performing upserts rather than creates on large data sets

An External ID field is always unique

True or False