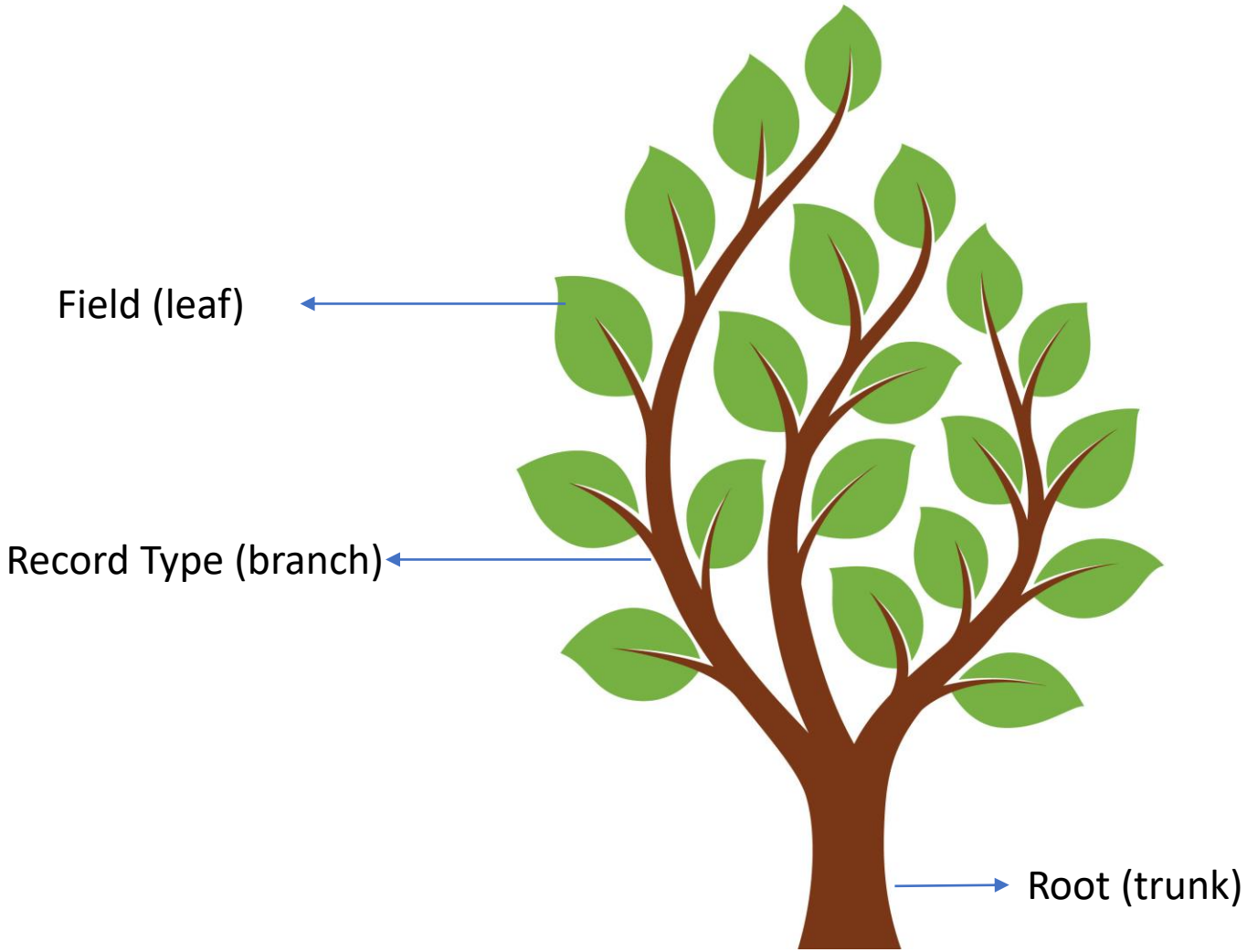
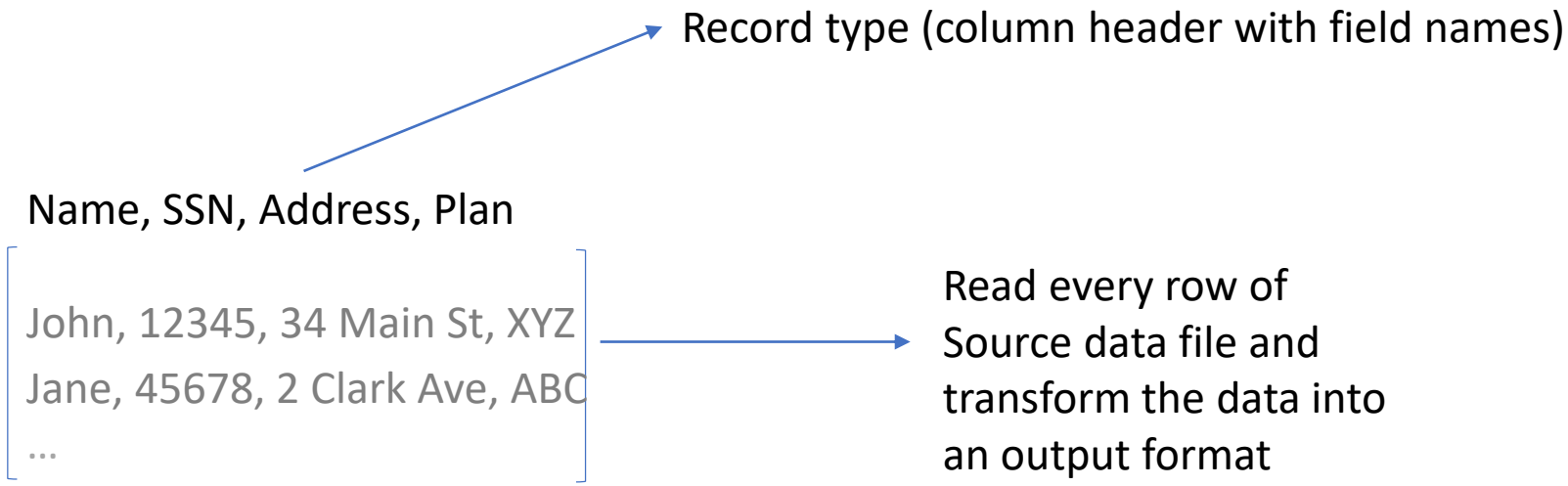
Data Mapping – Basic concepts on For-each condition

Conceptualizing the data

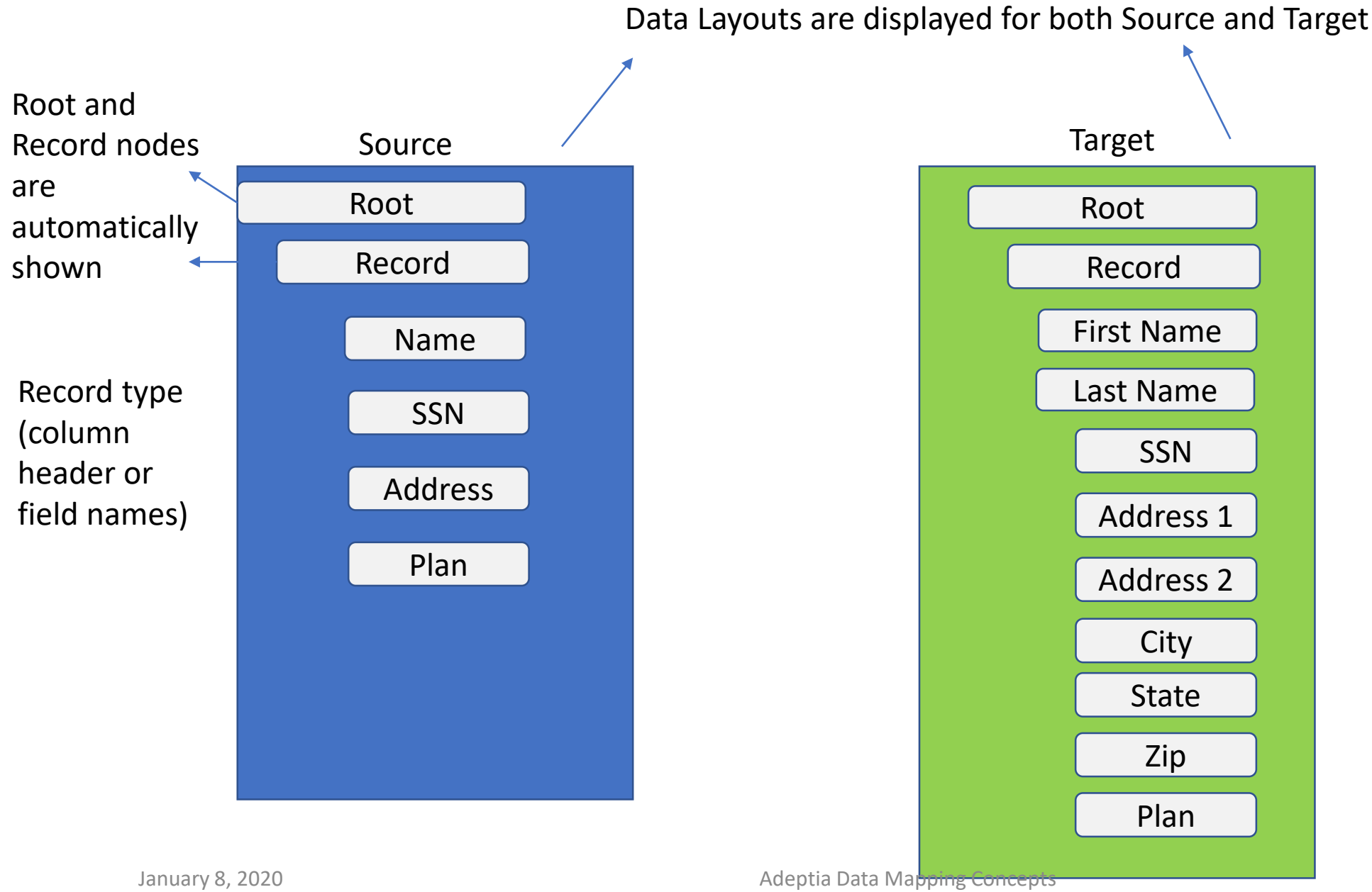


How Mapper reads the data file?

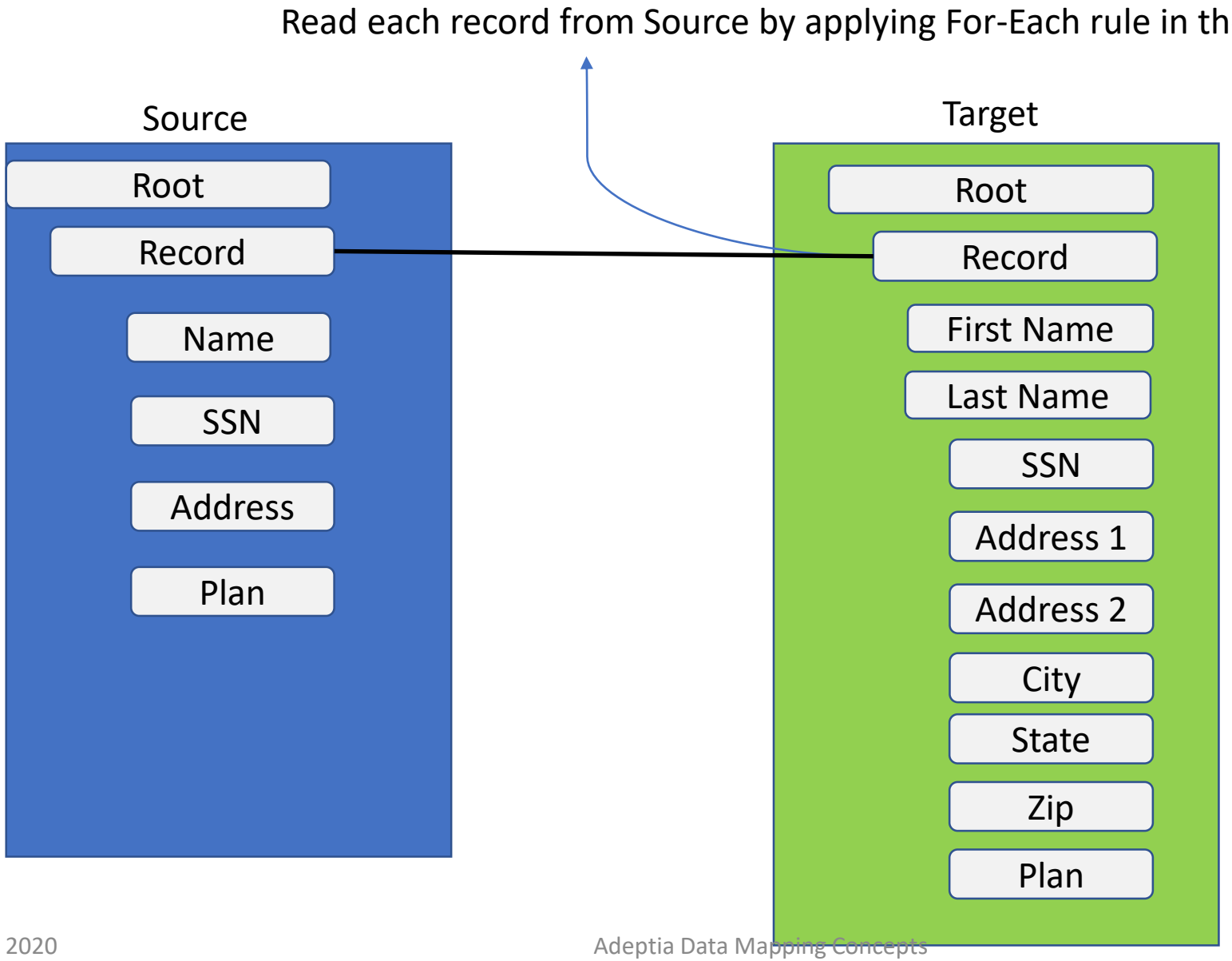


Note: File can have multiple Record types (in other words, file can have many branches)

Loading the Layouts in Data Mapper



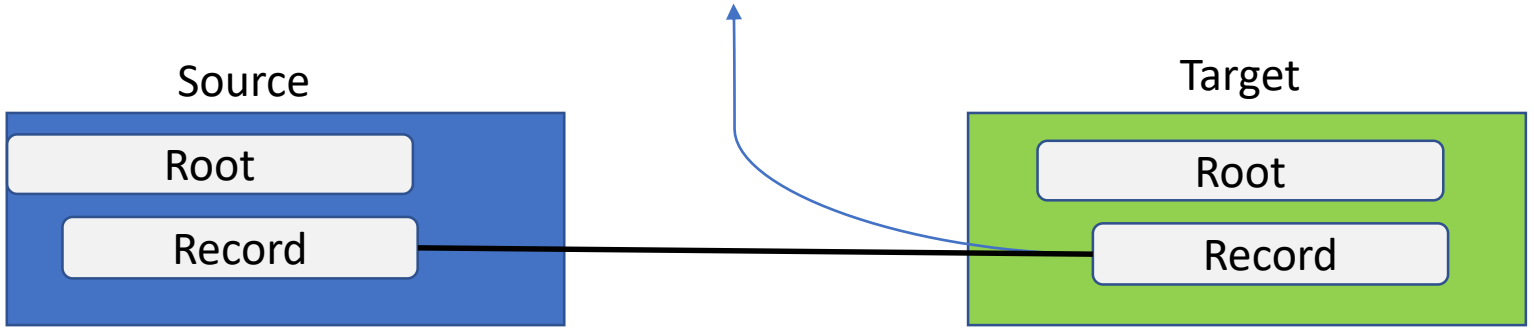
Rule #1 –> For-each



Join the Source Record branch to Target Record branch. This means that Mapper is going to pull and read each record from the source file for that branch.

Rule #1 –> For-each
How it works?

For-each rule runs as a cursor and traverses through the source file and reads one record at a time



1. Get the first record

John, 12345, 34 Main St, XYZ

2. Get the second record

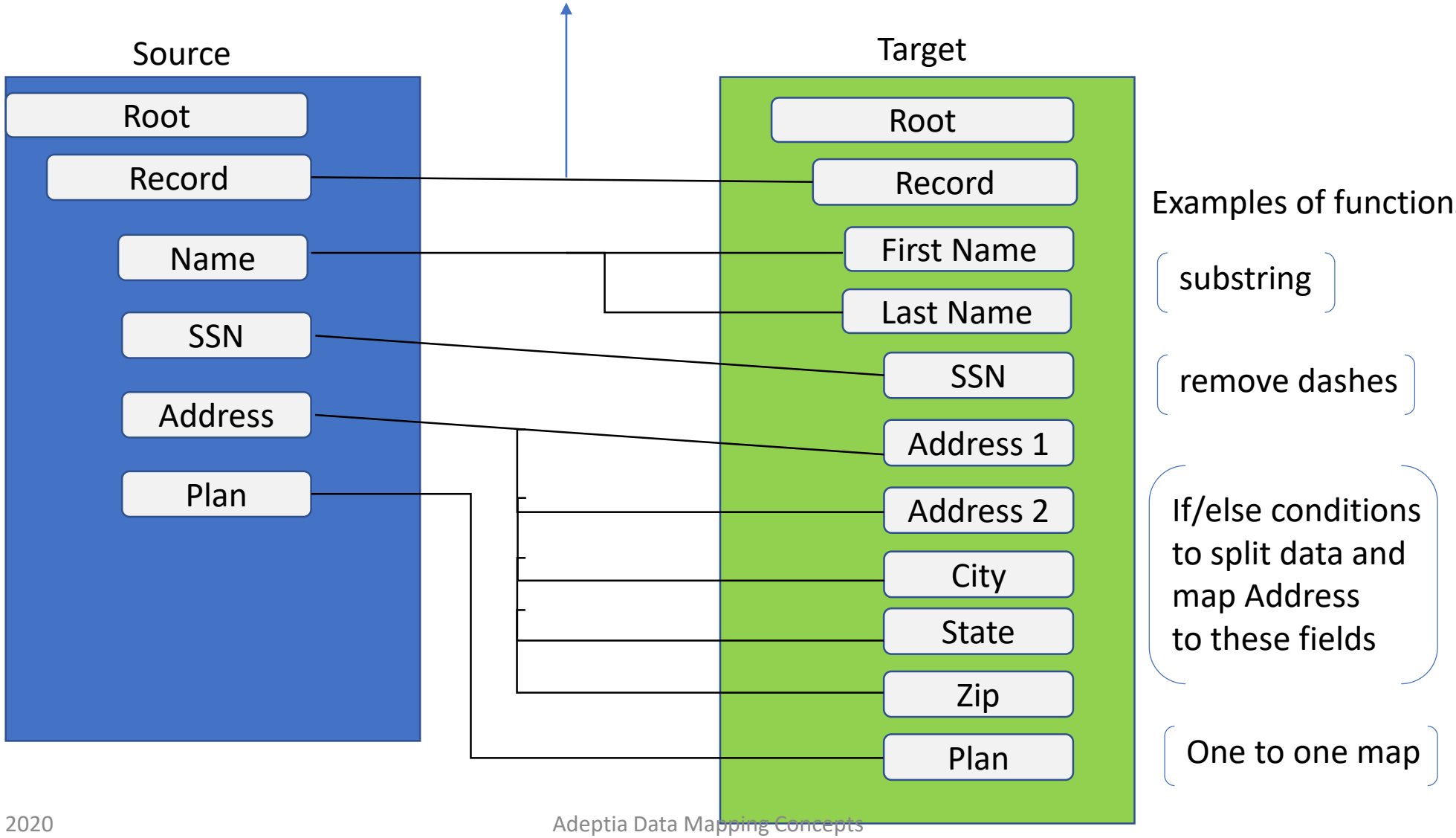
Jane, 45678, 2 Clark Ave, ABC

Name, SSN, Address, Plan

John, 12345, 34 Main St, XYZ
Jane, 45678, 2 Clark Ave, ABC
…

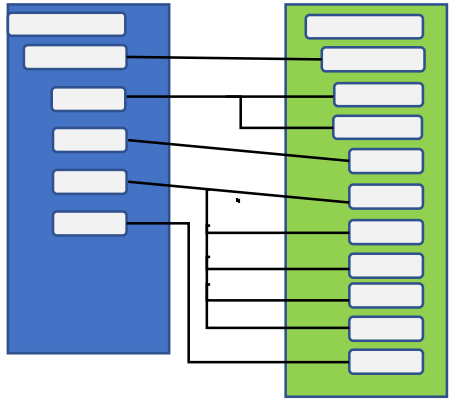
Rule #2 –> Apply field level functions such as substring, concat, if/else conditions etc.

Each record is read from source and all the field level rules are applied on that record.



Mapping behavior as it is traversing through the source file

Each record is read from source and then all the field level rules are applied to that record and the related output record is generated.



Apply these field level rules

1. Get the first record and produce an output record.

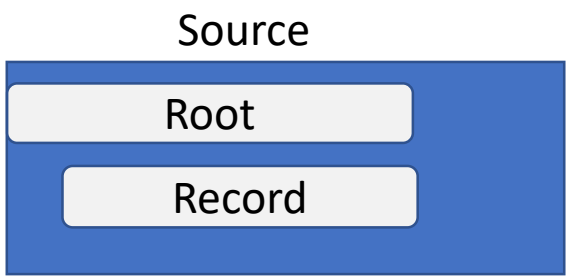
Source record: John, 12345, 34 Main St, XYZ

2. Get the second record and produce an output record

Source record: Jane, 45678, 2 Clark Ave, ABC

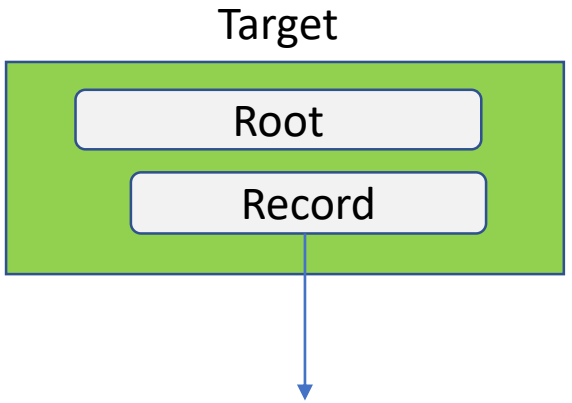
How to apply filters and check for certain conditions before loading the data into the output?

Use Case 1: Filtering from single source file
Filters require a qualified For-each condition that has a condition (also called a predicate).
For-each condition is available under the Properties tab in the Mapping Expression panel.



Name, SSN, Address, Plan

John, 12345, 34 Main St, XYZ
Jane, 45678, 2 Clark Ave, ABC
…



Jane, 45678, 2 Clark Ave, ABC

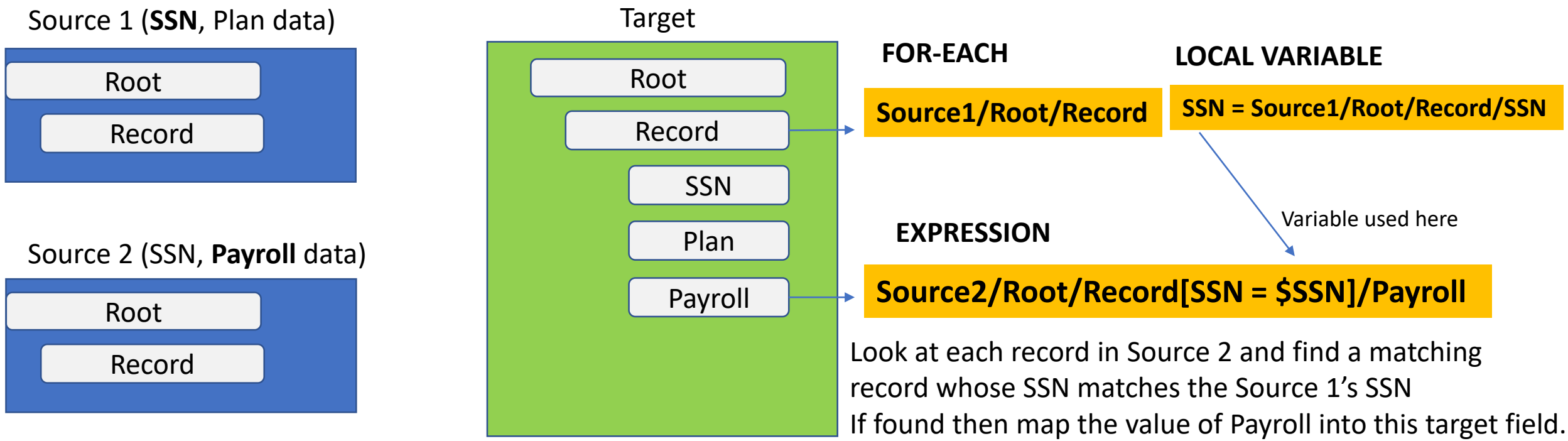
Root/Record[Plan = 'ABC']

From the source, only select records whose Plan = 'ABC'. This rule will pull only those records from source that meets this condition. You can add multiple conditions separated by and/or.

Note: Xpath (Root/Record) shown in above rule is shortened to show you the main syntax. Actual syntax would be like $Input_LayoutName/Root/Record[Plan = 'ABC']

Use Case 2: Joining data from two source files
We want to select the matching Payroll data from Source 2 by using Source 1's SSN (common key).

We will traverse through Source 1, get SSN for each record (in a variable) and go and lookup into Source 2 and find the matching data. Loop through the above logic for each record in Source 1.

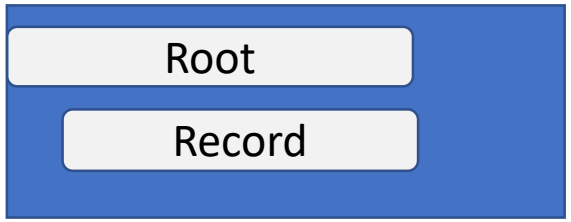


Look at each record in Source 2 and find a matching record whose SSN matches the Source 1's SSN
If found then map the value of Payroll into this target field.

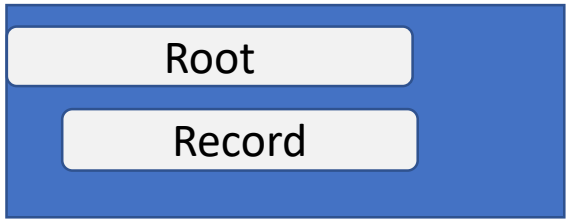
Use Case 3: Finding Delta between two files
We want to find delta between two Master data files and select only the Updated and New records.

First create two clones of the Target Record node as shown here.
Updated record is based on changed address.

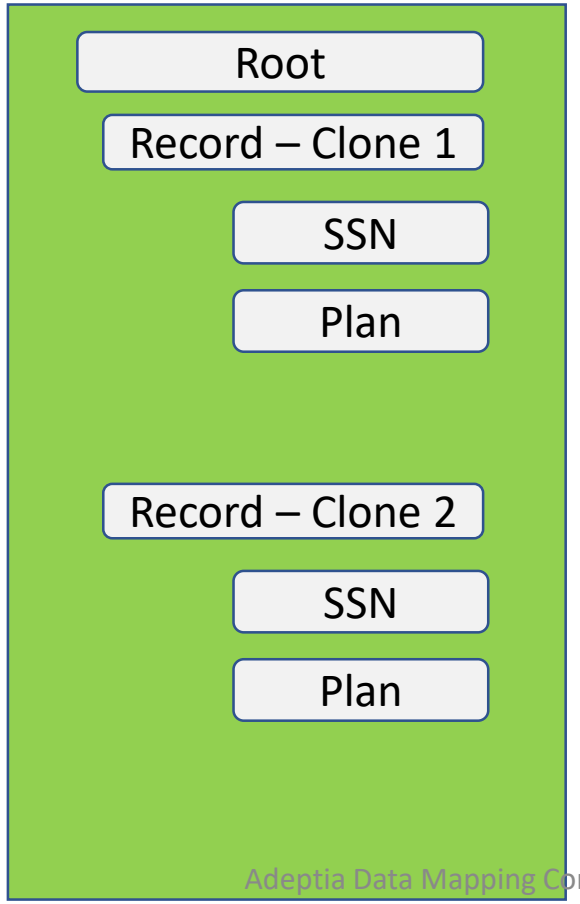
Source 1 (New Master Data)



Source 2 (Old Master Data)



Target



CLONE

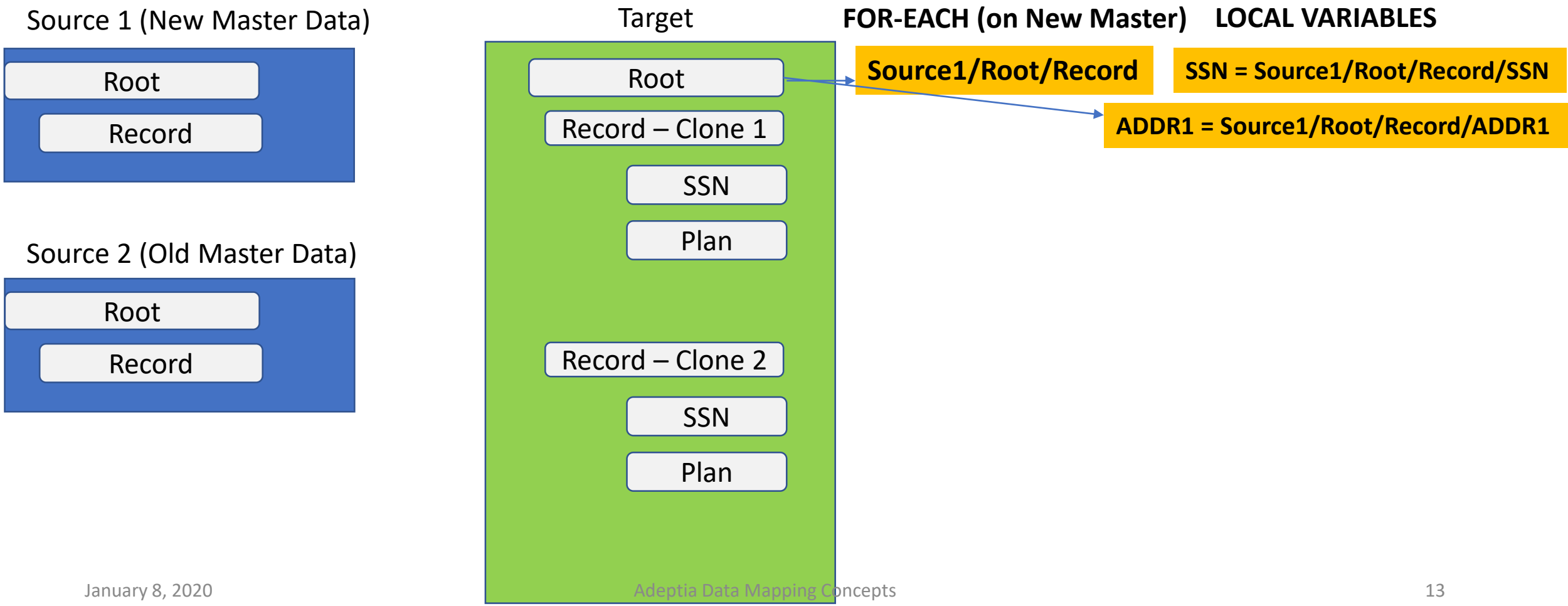
Used to select only New records

CLONE

Used to select only Updated records

Use Case 3 A: Finding Delta between two files
We want to find delta between two files and select only the Updated/New records.

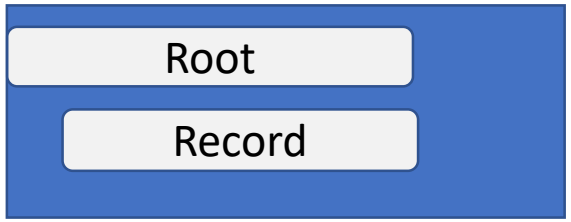
At the Target's Root node create For-each and local variable as shown here.



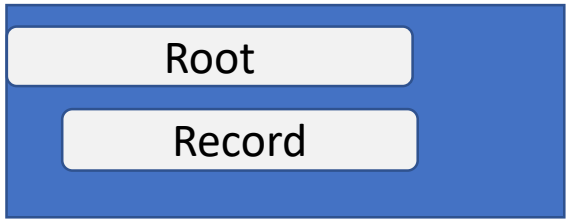
Use Case 3 B: Finding Delta between two files
We want to find delta between two files and select only the Updated/New records.

Now apply the rule on Record node – Clone 1

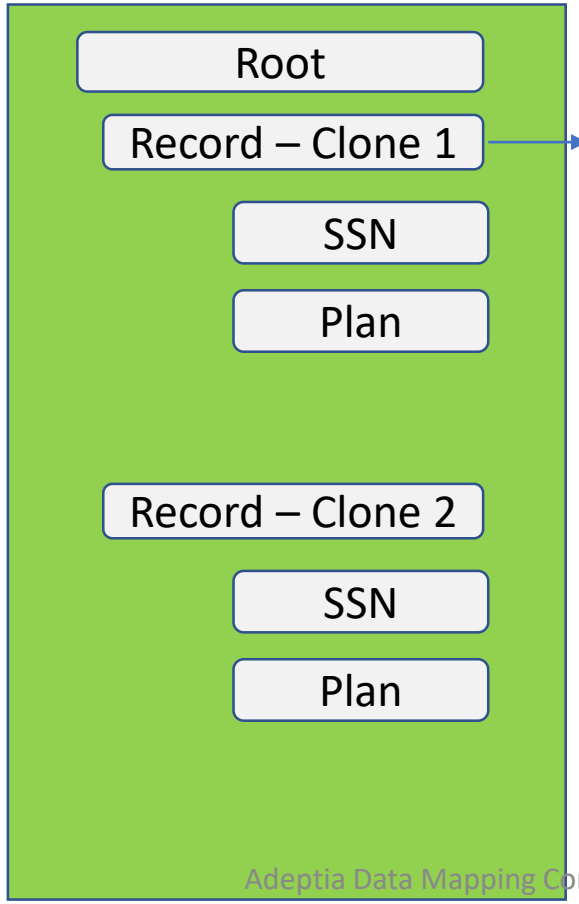
Source 1 (New Master Data)



Source 2 (Old Master Data)



Target



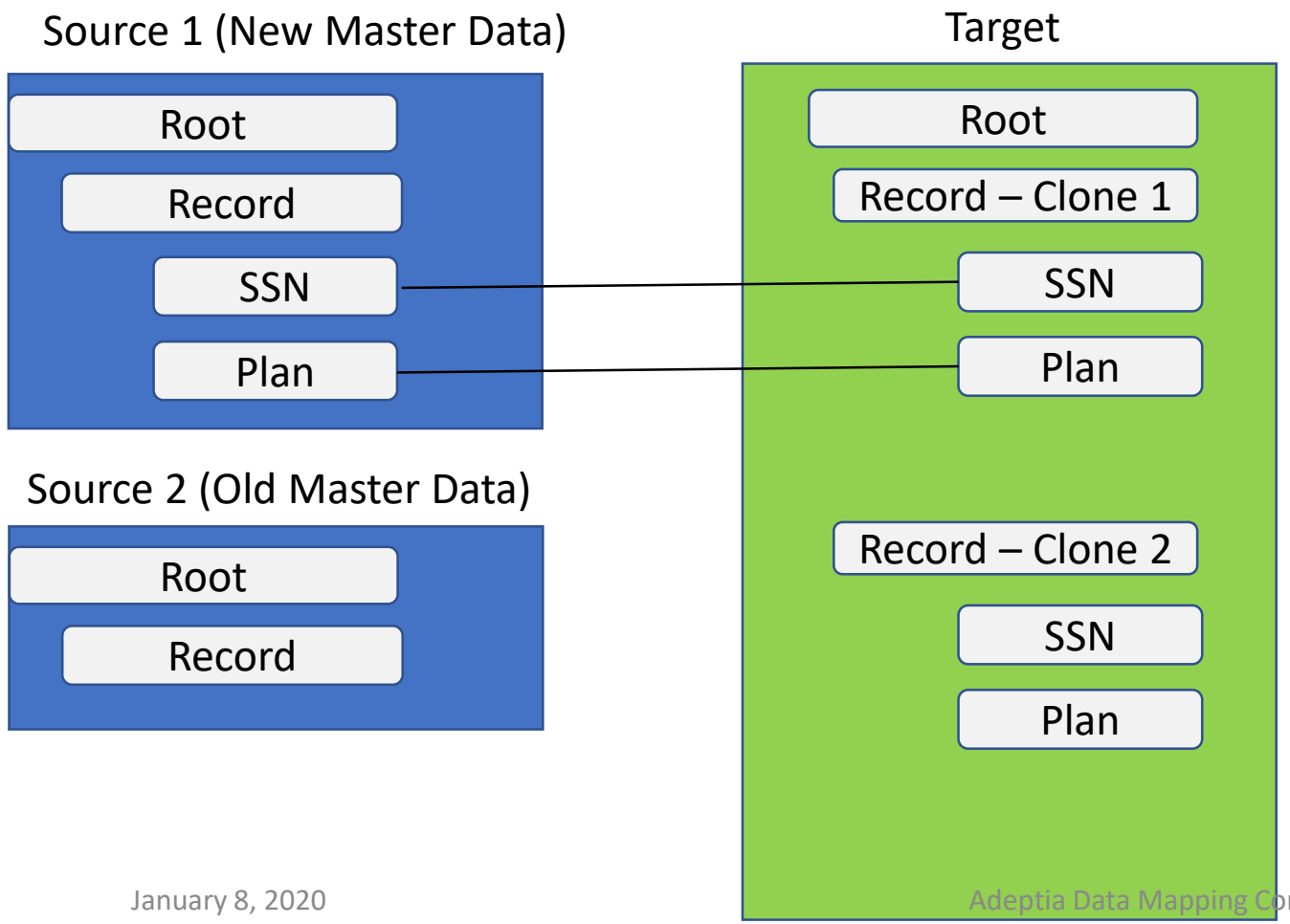
Filter IF (IFF) Condition

IFF count(Source2/Root/Record[SSN != $SSN]) = 0

Look at Source 2 and check if the SSN of the New Master matches with SSN of the Old Master. If the count is 0 then there's no match and this record is a new record.

Use Case 3 C: Finding Delta between two files
We want to find delta between two files and select only the Updated/New records.

Now apply the rule on each fields for Record node – Clone 1

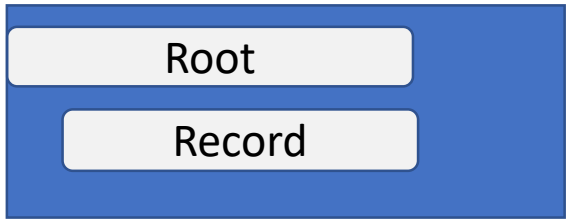


Fields from Source 1 are applied as one-to-one to the matching fields in the target

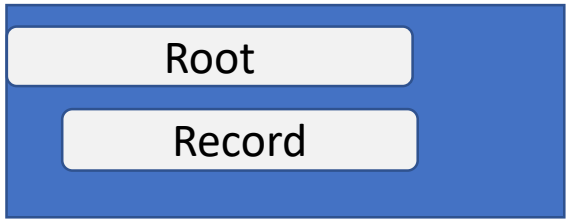
Use Case 3 D: Finding Delta between two files
We want to find delta between two files and select only the Updated/New records.

Now apply the rule on Record node – Clone 2

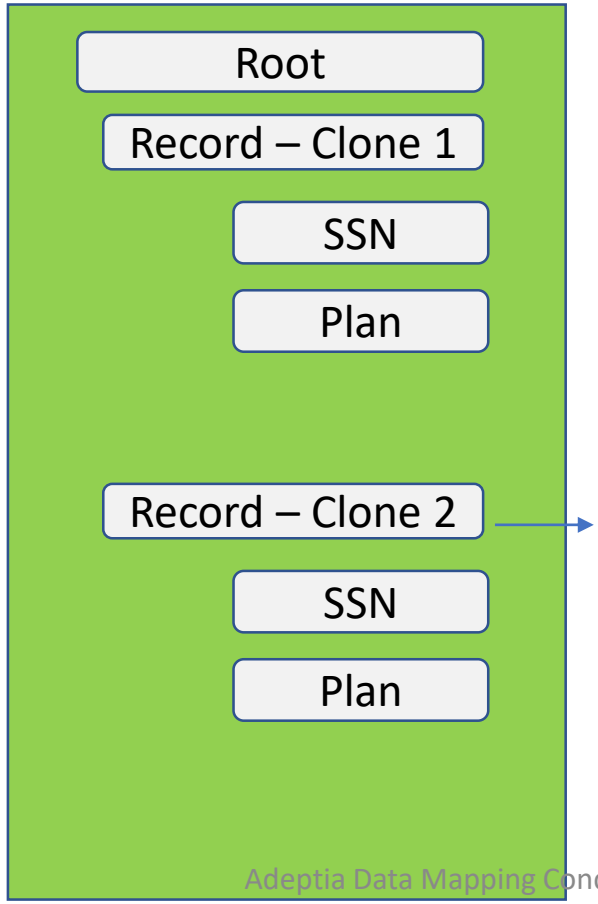
Source 1 (New Master Data)



Source 2 (Old Master Data)



Target



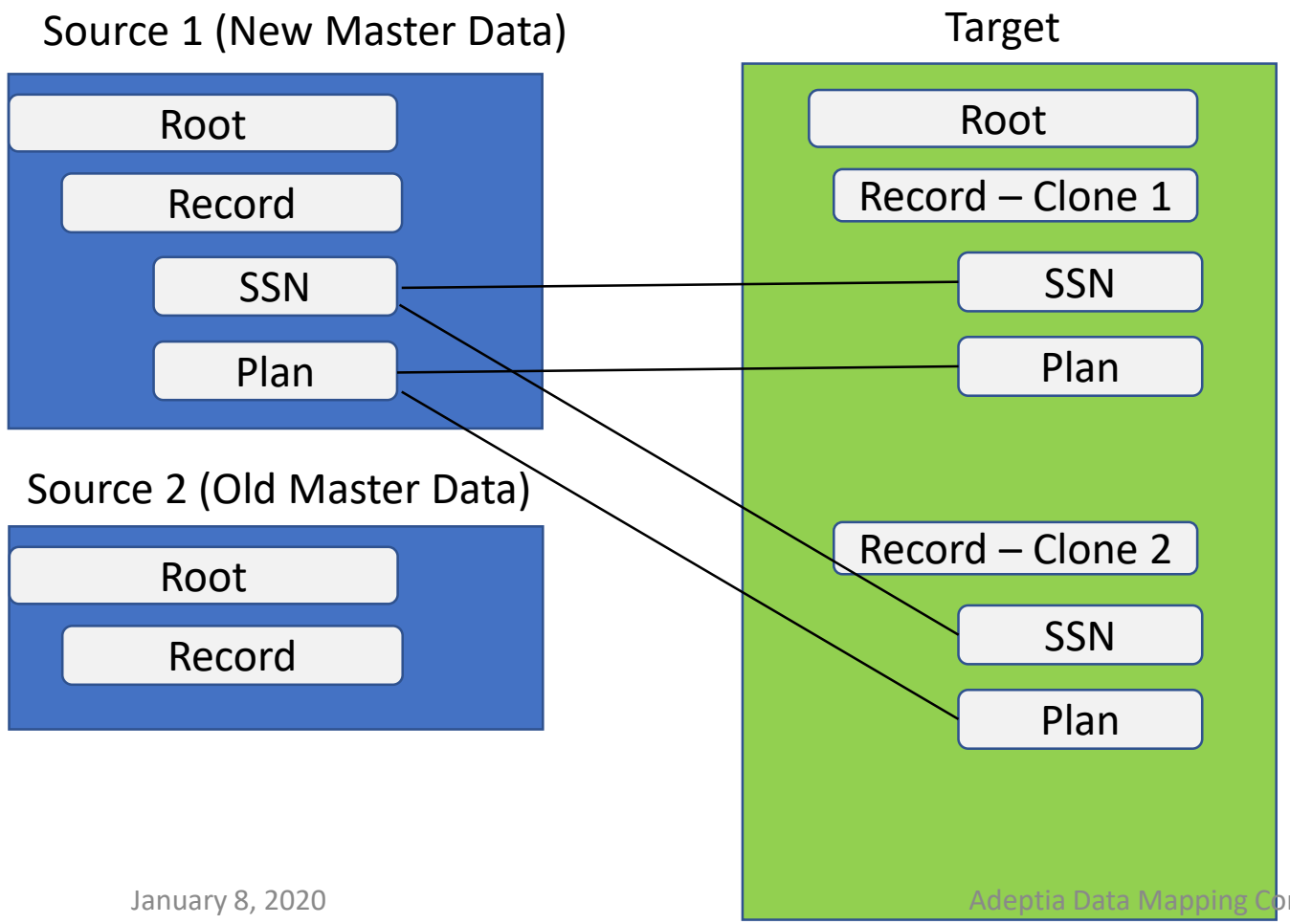
Filter IF (IFF) Condition

IFF count(Source2/Root/Record[SSN = $SSN]) != 0
and count(Source2/Root/Record[ADDR1 = ADDR1]) = 0

Look at Source 2 and check if the SSN of the New Master matches with SSN and if its Address has changed (not equal). If the Address returns 0 that means that the address has changed for an existing SSN. Thus it is a changed record. Apply one-to-one rules on the fields from New Master source fields.

Use Case 3 E: Finding Delta between two files
We want to find delta between two files and select only the Updated/New records.

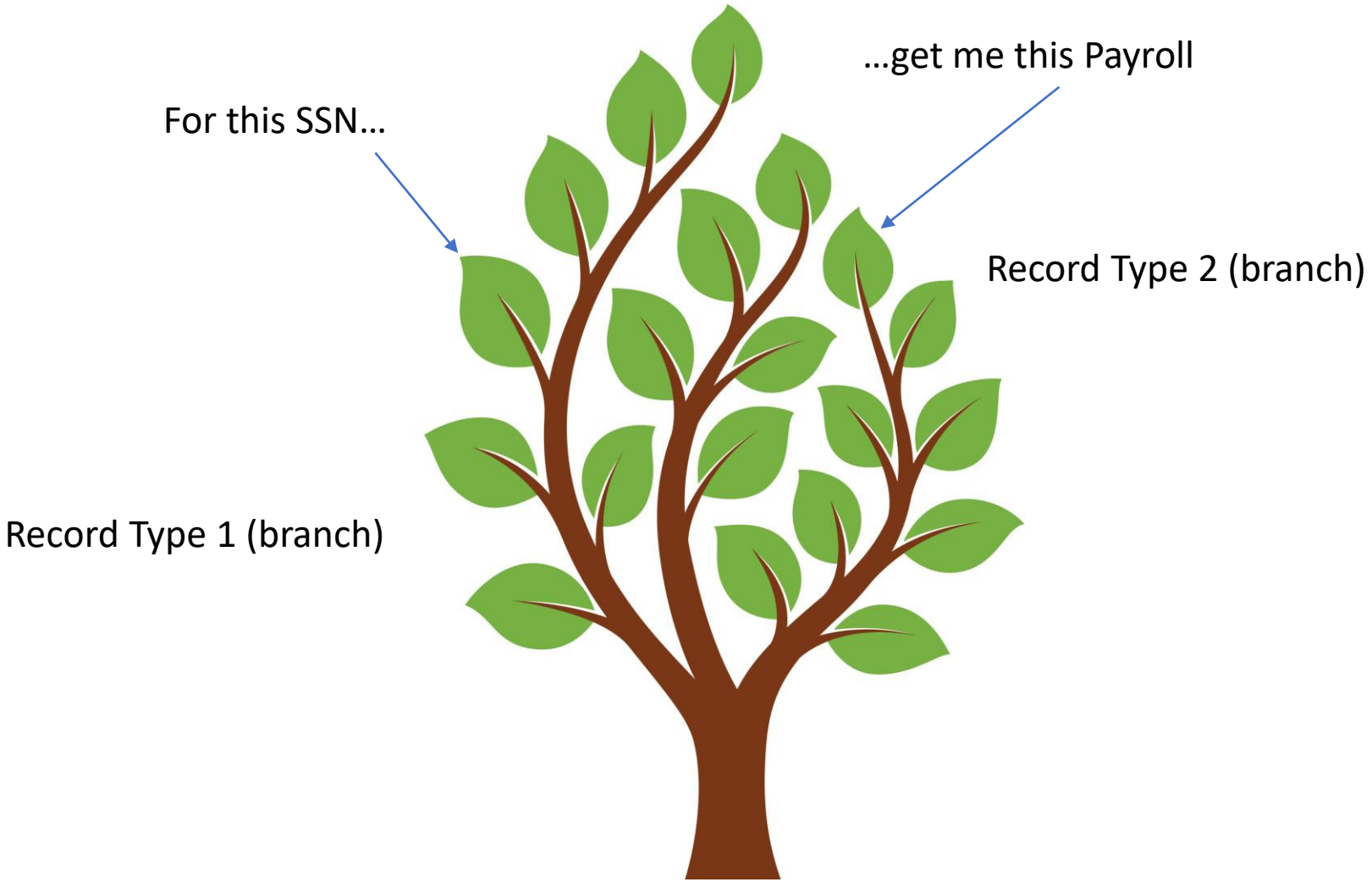
Now apply the rule one-to-one rule on each fields for Record node – Clone 2.



The output will give us a combined data of new and updated records.

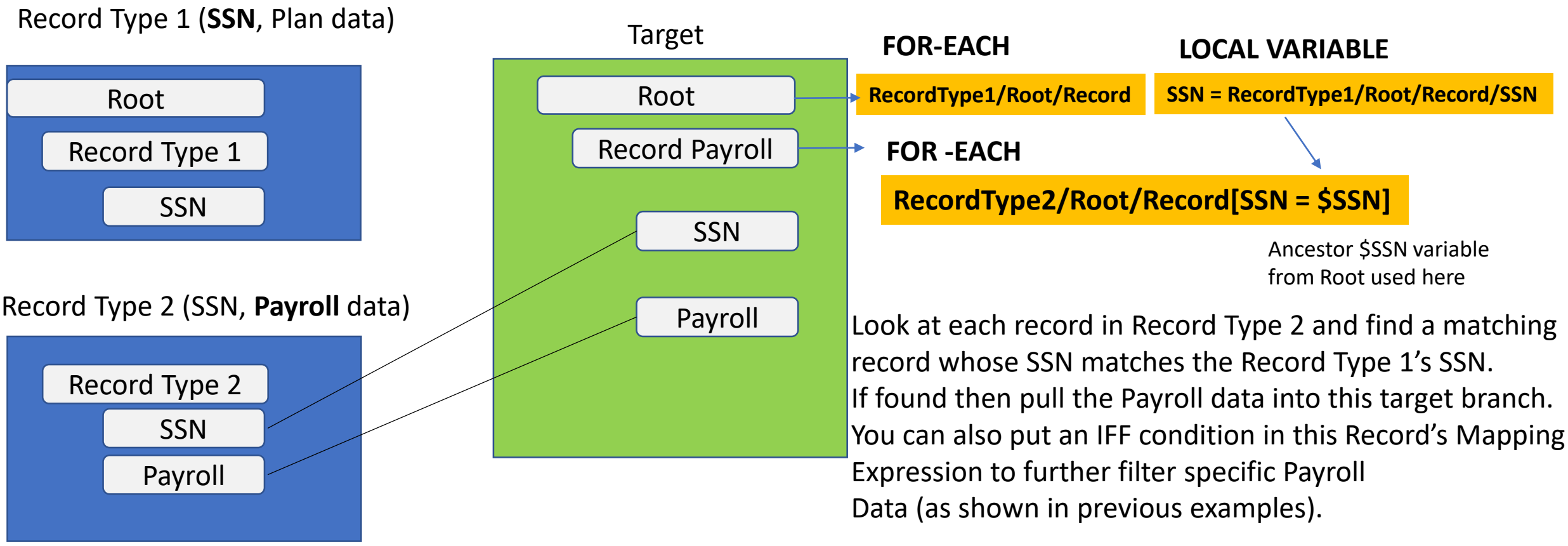
Filtering between multiple Record Types (branches) within a single file

Use Case 4: Get data from other branches within a single file



Use Case 4: Joining data from two branches within a single file
We want to select the matching Payroll data from Record Type 2 by using Record Type 1's SSN (common key).
Suppose the Target's Record branch needs Payroll data.

We will traverse through Record Type 1, get SSN for each record (in a variable) and go and lookup into Record Type 2 and find the matching data. Traverse through each record in Record Type 1 and apply this rule.



Look at each record in Record Type 2 and find a matching record whose SSN matches the Record Type 1's SSN.
If found then pull the Payroll data into this target branch.
You can also put an IFF condition in this Record's Mapping Expression to further filter specific Payroll Data (as shown in previous examples).

Map other fields from Record Type 2 to the matching fields in this target branch.

Tutorial on using common Mapping Functions
https://adeptia.com/data-mapping-use-cases

Mapping Functions Help
https://docs.adeptia.com/display/AC2/Using+Mapping+Functions