Request Source

JavaScript validation Of picklist In browser

BackEnd

1

Original record load
Or initialized **if upsert**

2

NEW overwrites OLD record

If **request source = Standard UI edit page**
SF runs system validation to check record for:
*Compliance with layout-specific rules
*Required values at the layout level and field-definition level
*Valid field formats
*Maximum field length

If **request source = Apex application or a SOAP API** call
Then SF validates only the foreign keys.

Meaning It does not check other fields but only the foreign keys.

Prior to executing a trigger, Salesforce verifies that any custom foreign keys do not refer to the object itself

Validation rules apply to new and updated records for an object, even if the fields referenced in the validation rule are not included in a page layout or an API call.
Validation rules don't apply if you create new records for an object with Quick Create.

User defined validation always execute after "before triggers".
System validation rules are the ones who runs before "before triggers".

Layout specific rules are the ones where you could define if the field is required or not. It can be found by clicking on the edit layout link in the record detail page

Validation rules on the other hand are more configurable set of rules where you can throw out an error upon certain rules. (Ex. Validate only if the Opportunity Stage is "Closed Won")

Salesforce **runs user-defined validation rules** if multiline items were created, such as quote line items and opportunity line items.

3

Executes all **before triggers**.

A trigger is Apex code that executes before or after the following types of operations:
insert
update
delete
merge
upsert
undelete

There are **two types of triggers**:
Before triggers are used to **update or validate record values before they're saved to the database.**
After triggers are used to **access field values that are set by the system** (such as a record's Id or LastModifiedDate field), and to effect changes in other records, such as logging into an audit table or firing asynchronous events with a queue. **The records that fire the after trigger are read-only.**

If you update or delete a record in its before trigger, or delete a record in its after trigger, you will receive a runtime error.

A trigger invoked by an insert, delete, or update of a recurring event or recurring task results in a runtime error when the trigger is called in bulk from the Force.com API.

4

Runs most system validation steps again, such as verifying that all **required fields have a non-null value**, and runs any **user-defined validation rules**.
The only system validation that Salesforce doesn't run a second time (when the request comes from a standard UI edit page) is the enforcement of layout-specific rules.

5

Executes **duplicate rules**.
If the duplicate rule identifies the record as a duplicate and uses the block action, the record is not saved and no further steps, such as after triggers and workflow rules, are taken.

When a user attempts to save a new record, the record is first compared with existing Salesforce records to identify possible duplicates (1).
The criteria used to compare records and identify the possible duplicates are defined by a matching rule. Next, a list of possible duplicates is returned (2).
What happens when the record being saved is identified as a possible duplicate depends on what's defined in the duplicate rule (3).
For example, the duplicate rule could block users from saving the possible duplicate record or allow them to save it anyway.
When a user attempts to save an edited record, the record is first checked to see if the user has changed the value of a matching rule field. If so, the duplicate management process works as described for new records. If not, no further action is taken and duplicates are not detected.

If the first duplicate rule finds a match for a particular record, that record will not be evaluated by subsequent duplicate rules. Therefore, you should order your duplicate rule so that rules with the Block action are run before rules with the Allow action.
Duplicate rules are available for accounts, contacts, leads, and custom objects. All other objects, including Opportunities and Person Accounts, are not currently supported.
Duplicate rules don't run when:
Records are created using Quick Create.
Leads are converted to accounts or contacts* and your *organization doesn't have the "Use Apex Lead Convert" permission.
Records are restored with the Undelete button.
Records are added using Exchange Sync.
Records are manually merged.
A *Self-Service user creates records and the rules include conditions based on the User object.*
Duplicate rule conditions are set for lookup relationship fields and records with no value for these fields are saved.

6

Saves the record to the database, but doesn't commit yet.

7

Executes all after triggers.

8

Executes assignment rules.

Lead Assignment Rules—Specify how leads are assigned to users or queues as they are created manually, captured from the web, or imported via the Data Import Wizard.

Case Assignment Rules—Determine how cases are assigned to users or put into queues as they are created manually, using Web-to-Case, Email-to-Case, On-Demand Email-to-Case, the Self-Service portal, the Customer Portal, Outlook, or Lotus Notes

For each rule type, only one rule can be in effect at any time.

Each rule consists of multiple rule entries that specify exactly how the leads or cases are assigned. For example, your standard case assignment rule may have two entries: cases with "Type equals Gold" are assigned to "Gold Service" queue, and cases with "Type equals Silver" are assigned to "Silver Service" queue.

9

Executes <u>auto-response rules</u>.

Send automatic email responses to lead or case submissions based on the record's attributes. Set up **auto-response** rules to send quick replies to customers to let them know someone at your company received their inquiry or details about their issue.

10

Executes <u>workflow rules</u>.

Workflow allows you to **automate standard internal procedures and processes** to save time across your organization. A **workflow rule is the main container for a set of workflow instructions.**

Workflow email alerts **= Notifications to interested parties**
<u>WHEN</u> **A case or lead is created or edited**
<u>TO</u> **Anyone you choose**.

Auto-response rules **= Initial response to the contact who created a case or the person who submitted the lead on the Web.**
<u>WHEN</u> **A case or lead is created.**
<u>TO</u> **Contact on a case or the person who submitted the lead on the Web.**

11

If there are workflow field updates, <u>updates the record again</u>.

12

If the record was updated with workflow field updates, <u>fires before update triggers and after update triggers one more time</u> <u>*(and only one more time)*</u>, in <u>addition to standard validations.</u> *Custom validation rules, duplicate rules, and escalation rules are not run again.*

13

Executes processes.
If there are workflow flow triggers, executes the flows.

The Process Builder has **superseded flow trigger workflow actions, formerly available in a pilot program**. *Organizations that are using flow trigger workflow actions can continue to create and edit them, but flow trigger workflow actions aren't available for new organizations.*

	Process Builder	Visual Workflow	Workflow	Approvals
Complexity	Multiple IF	Complex	Single IF	Single IF
Visual designer	X	X		
Starts when:	Record is changed	*User click *User access *Process start *Apex called	Record is changed	*User click *Process or flow start and includes „Submit for approval" *Apex called
Time-based action support	X(multiple shedules per criteria)	X	X	
User interaction support		X		
Call Apex code	X	X		
Create records	X	X	Tasks only	Tasks only
Delete records		X		
Launch flow	X	X	X(Pilot only)	
Post to chatter	X	X		
Send email	X(ONLY Email alerts)	X	X(ONLY Email alerts)	X(ONLY Email alerts)
Send outbound messages without code			X	
Submit for approval	X	X		
Update fields	Any related record	Any record	The record or its parent	The record or its parent

14

Executes escalation rules.

Each rule **defines a condition that determines how cases are processed.**
Each time you save a case or change the case owner, your **escalation rules re-evaluate that case**. Once the **case matches an escalation rule entry, calculates when the case should be escalated and stops checking other escalation rule entries**. For example, if you have two escalation rule entries that specify:
Escalate three hours after creation date if Case Reason equals Crash
Escalate four hours after creation date if Case Reason equals Bug
A case created with Case Reason of Bug will be scheduled for escalation four hours after it was created. Later, a user changes the case, which causes the escalation rules to re-evaluate the case. If escalation rules find that the Case Reason is now Crash, it schedules the case to be escalated three hours after creation date. If the case was created more than three hours ago, the case is escalated as soon as possible.
Escalation rules are not evaluated when transferring multiple cases at one time from a case list view. Also note that **if you use assignment rules to change case ownership, the escalation rules are evaluated before any assignment rules.**

Salesforce processes rules in the following order:
1 Validation rules
2 Assignment rules
3 Auto-response rules
4 Workflow rules (with immediate actions)
5 Escalation rules

15

Executes entitlement rules

A timeline that includes all the steps (milestones) that support agents must complete to resolve a case. Each process includes the logic needed to determine how to enforce the correct service level for your customers. Not all entitlements need processes. For example, a simple entitlement might just state that a customer is eligible for phone support 24/7. If you need to add time-dependent steps or service levels to that definition—for example, if you want a supervisor to be notified by email when a customer's case goes unresolved for two hours—you need an entitlement process

1 **A support agent linked a case to an entitlement that has an entitlement process**. This can be done in several ways:
* The support agent creates the case from the Cases related list on the entitlement.
* The support agent creates the case, then uses the Entitlement lookup field on the case to select the proper entitlement.

2 **The case enters the process based on its creation date or a custom date/time field**. A custom date/time field lets users edit a date on the case to trigger when it enters the process.

3 **Salesforce assigns milestones with matching criteria to the case.** For example, if a milestone's criteria is Priority equals High, and a case has a Priority of High, Salesforce assigns it to the Priority equals High milestone. A case associates with one milestone at a time. It can associate with many milestones as it moves through the process.

4 **Milestone actions determine when and if warning, violation, or success workflow actions fire for the case**.

5 **A support agent updates the case to complete a milestone action.**

6 **After a case is updated, it cycles through the entitlement process and initiates any milestones that match its criteria.**

7 **The case exits the process based on custom criteria or when it's closed.**

16

If the record contains a roll-up summary field or is part of a cross-object workflow, performs calculations and updates the roll-up summary field in the parent record. <u>Parent record goes through save procedure.</u>

17

If the parent record is updated, and a grandparent record contains a roll-up summary field or is part of a cross-object workflow, performs calculations and updates the roll-up summary field in the grandparent record. <u>Grandparent record goes through save procedure.</u>

18

Executes Criteria Based Sharing evaluation.

Criteria-based sharing rules determine whom to share records with based on field values in records. For example, let's say you use a custom object for job applications, with a custom picklist field named "Department." You can create a criteria-based sharing rule that shares all job applications in which the Department field is set to "IT" with all IT managers in your organization

Text and Text Area are case-sensitive. For example, a criteria-based sharing rule that specifies "Manager" in a text field won't share records with "manager" in the field.

If in the trigger you have written a code to send email then after the complete trigger finishes and the updates are committed to database **then salesforce sends the email**. This action of sending email can be called a post-commit logic

19

Commits all DML operations to the database.

20

Executes post-commit logic, such as sending email.

A save writes your changes to the database, however at this point **these changes are only visible to you within your transaction scope. <u>The database has also generated undo information which contains the old values of your transaction which can be used to rollback your modifications.</u>**

A commit ends the current transaction and makes permanent all changes performed in the transaction. The transaction is a sequence of SQL statements that the database treats as a single unit. A commit also erases all savepoints in the transaction and releases transaction locks. <u>After your data is committed, it is visible to other users of the system.</u>

What happens when a record is saved?
Your data is stored and can still be rolled back to its previous values.
Is a Salesforce ID and autonumber (if applicable) assigned?
Yes
At which step are formula fields calculated?
Formulas are calculated when the data is read, not when it is written.

During a recursive save, Salesforce skips <u>steps 8</u> (assignment rules) <u>through 17</u> (roll-up summary field in the grandparent record).

!

Aditional Considerations

<u>The order of execution isn't guaranteed</u> when having multiple triggers for the same object due to the same event. For example, if you have two before insert triggers for Case, and a new Case record is inserted that fires the two triggers, the order in which these triggers fire isn't guaranteed.

When a DML call is made with partial success allowed, more than one attempt can be made to save the successful records if the initial attempt results in errors for some records.DML calls allow partial success when you set the allOrNone parameter of a Database DML method to false or when you call the SOAP API with default settings.

Inserts, updates, and deletes on person accounts fire Account triggers, not Contact triggers.

Trigger.old contains a version of the objects before the specific update that fired the trigger. However, there is an exception. When a record is updated and subsequently triggers a workflow rule field update, Trigger.old in the last update trigger won't contain the version of the object immediately prior to the workflow update, but the object before the initial update was made. <u>For example, suppose an existing record has a number field with an initial value of 1. A user updates this field to 10, and a workflow rule field update fires and increments it to 11. In the update trigger that fires after the workflow field update, the field value of the object obtained from Trigger.old is the original value of 1, rather than 10, as would typically be the case.</u>

If errors occur on an opportunity product, you must return to the opportunity and fix the errors before cloning.
If any opportunity products contain unique custom fields, you must null them out before cloning the opportunity.

<u>The following actions don't trigger workflow rules:</u>
Mass replacing picklist values
Mass updating address fields
Mass updating divisions
Changing the territory assignments of accounts and opportunities
Converting leads to person accounts
Deactivating Self-Service Portal, Customer Portal, or Partner Portal users
Converting state and country data using the Converttool in Setup | Data Management | State and Country Picklists
Making changes to state and country picklists using AddressSettings in the Metadata API

<u>Workflow Rule Limitations</u>
You can't create email alerts for workflow rules on activities.
You can't package workflow rules with time triggers.
You can't create outbound messages for workflow rules on junction objects.

<u>Order of Execution-WORKFLOW RULES</u>
Workflow is triggered Correct order is
Field Updates > Task/Actions > Emails Alerts > Outbound message

<u>Order of Execution-RULES</u>
Validation Rules
Assignment rules
Auto-response rules
Workflow rules (with immediate actions)
Escalation rules

<u>Flow Trigger Limits</u>
A flow trigger is a workflow action that launches a flow.
Flow triggers are available only for workflow rules. You can't use them as actions elsewhere, for example, in approval processes.
Flow triggers are available on most—but not all—objects that are supported by workflow rules.
Flow triggers aren't available as time-dependent workflow actions. You can add flow triggers to workflow rules only as immediate workflow actions.
Only active, trigger-ready flows can be launched by flow triggers. However, if a flow trigger is in test mode, administrators run the latest flow version while other users run the active flow version.
A flow trigger can set the values of up to 25 variables and sObject variables in the flow, with the following limitations.
Flow triggers can't use multi-select picklist fields to set flow variables or sObject variables.
When a flow trigger uses a currency field to set a flow variable, only the amount is passed into the flow. Any currency ISO code or locale information is ignored. If your organization uses multiple currencies, the flow trigger uses the amount in the currency of the record that contains the specified currency field.
Flow triggers can't pass values into sObject collection variables in flows.
Flow triggers aren't available in change sets.
Flow triggers aren't packageable

1 **ENTRY POINT** : Standard UI
-Javascript browser validation
-Layout specific rules |
-Required values |-------NOT RAN WITH QUICK CREATE, APEX or SOAP API
-Valid field formats |
-Field length |

Layout specific rules = field required or not
Validation rules = more complex, check value, throw error

If ENTRY POINT = APEX or SOAP API it only checks foreign keys

3 **Before triggers**
!!! Multi-line item = 1 Opportunity that contains multiple products!!!
!!! If you create multi-line items that activate USER DEFINED VALIDATION RULES =>
These rules will ran BEFORE the "before triggers"!!!

*Update/Delete record in its own Before trigger => RUNTIME ERROR (like an infinite loop)
*Delete a record in its own After trigger =>RUNTIME ERROR
*API bulk Insert/Update/Delete of a recurring event or task => RUNTIME ERROR

4 **System validation rules** WITHOUT LAYOUT SPECIFIC RULES and User defined rules

5 **Duplicate rules** = looks for duplicate records
-Possible duplicates are identified by a matching rule
-Can Block or Allow save of a duplicate
-Can STOP the process and throw an error

6 **Save the record but NOT commit**

7 **After triggers**

8 **Assignment rules** = Lead assignment rules, Case assignment rules
Lead = prospect contact, contact that can become a client
Lead assignment rules = how leads are assigned to users or queues. (ex: a prospect asks more sales related info , we assign him to the sales dept pool or in a queue of prospects that asked the same thing)
Case = support case (for ex. Issue reported by a user)

9 **Auto response rules** = automatic email responses

10 **Workflow rules** = automate internal procedures and processes
If the workflow updates another record => before/after update triggers fired again(only one more time), standard validations.
NO custom validation, duplicate and escalation rules are run.

13 **Execute processes** = > flows designed with different tool (Process builder, Visual Workflow, Workflow, Approvals)

14 **Escalation rules** = > condition that determines how a case is escalated. (for example: if a case is opened you can define a certain amount of time required for the case to be solved and closed before it escalates and require attention)

15 **Entitlement rules** => timeline that includes all the steps(milestones) that support agents must complete to resolve a case.

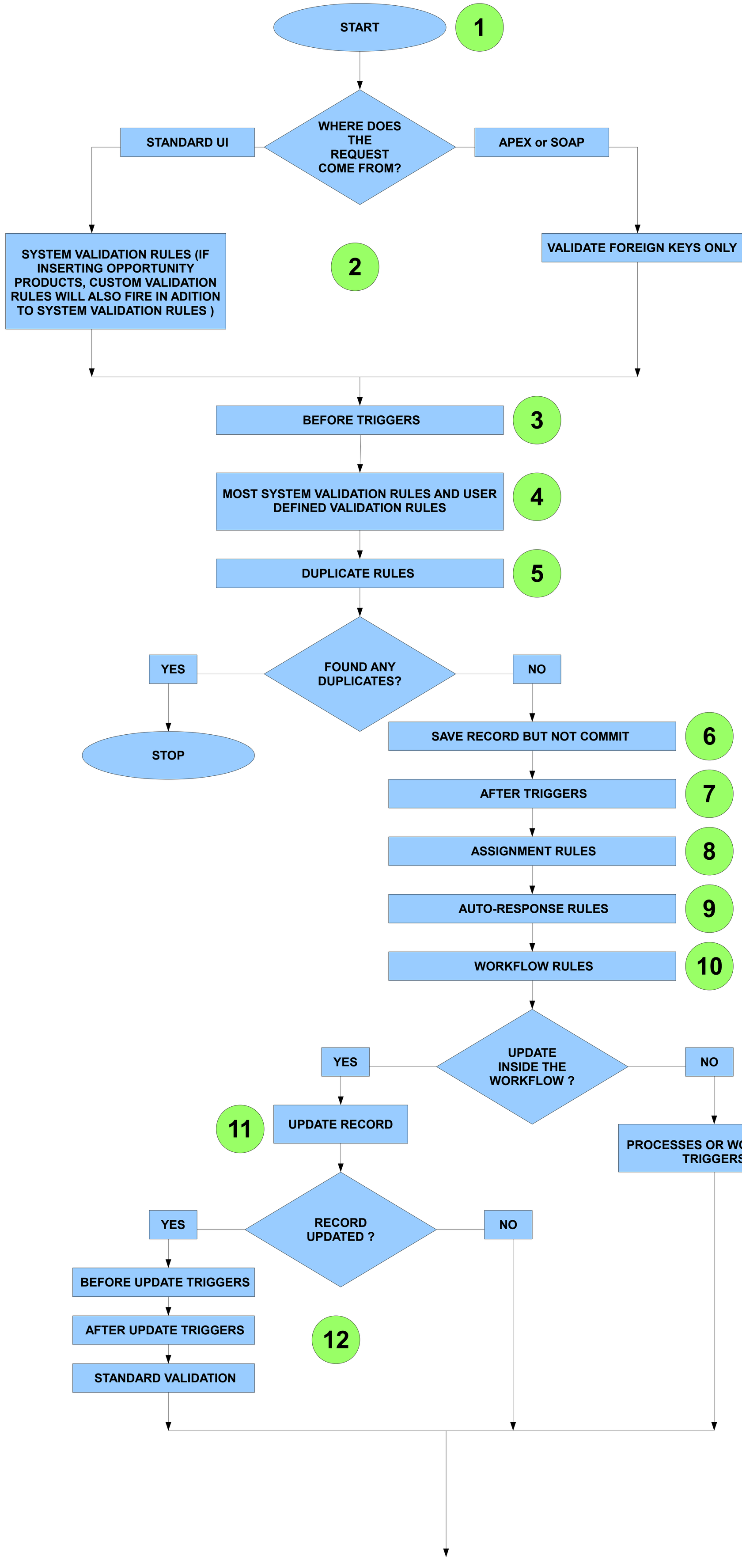
16 **Roll-up summary fields** => While formula fields calculate values using fields within a single record, roll-up summaryfields calculate values from a set of related records, such as those in a related list.
Cross object workflow =>Workflow Rule based on Case would be allowed to have a Workflow Field Update change a Contact field for the Contact record associated with the Case.

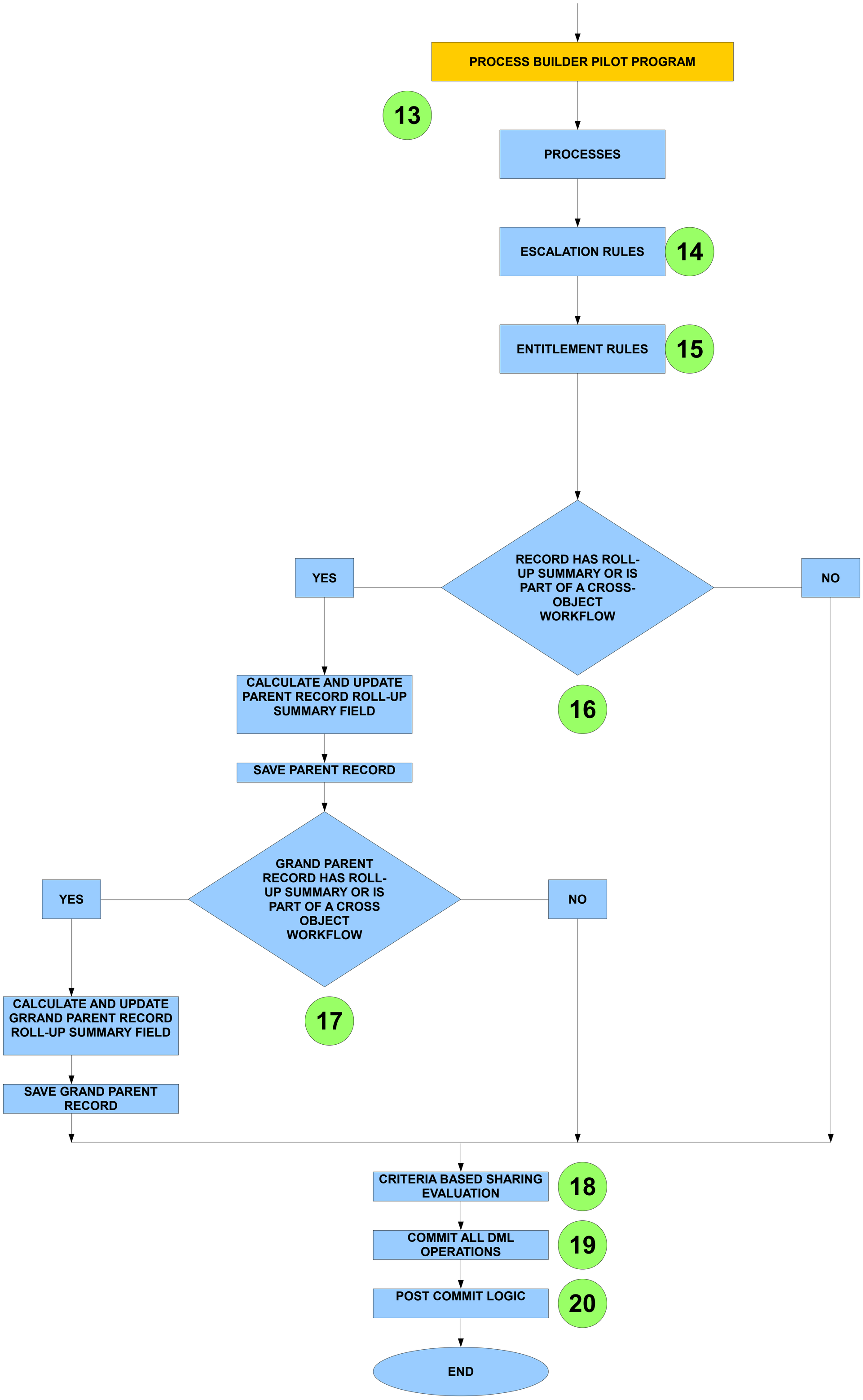
17 If roll-up summary or cross object workflow updates a PARENT and that PARENT HAS GRAND PARENT then the rpocess is repeated for the GRAND PARENT record

18 **Criteria based sharing** => share record based on a key word that it contains. IT IS CASE SENSITIVE.

19 **Commit all DML operations** => NO MORE ROLLBACK

20 **Post-commit logic** => send email after inserting/modifying record


START
1
WHERE DOES THE REQUEST COME FROM?
STANDARD UI
APEX or SOAP
SYSTEM VALIDATION RULES (IF INSERTING OPPORTUNITY PRODUCTS, CUSTOM VALIDATION RULES WILL ALSO FIRE IN ADITION TO SYSTEM VALIDATION RULES)
2
VALIDATE FOREIGN KEYS ONLY
BEFORE TRIGGERS
3
MOST SYSTEM VALIDATION RULES AND USER DEFINED VALIDATION RULES
4
DUPLICATE RULES
5
FOUND ANY DUPLICATES?
YES
NO
STOP
SAVE RECORD BUT NOT COMMIT
6
AFTER TRIGGERS
7
ASSIGNMENT RULES
8
AUTO-RESPONSE RULES
9
WORKFLOW RULES
10
UPDATE INSIDE THE WORKFLOW ?
YES
NO
11
UPDATE RECORD
PROCESSES OR WORKFLOW TRIGGERS
RECORD UPDATED ?
YES
NO
BEFORE UPDATE TRIGGERS
AFTER UPDATE TRIGGERS
STANDARD VALIDATION
12


PROCESS BUILDER PILOT PROGRAM
13
PROCESSES
ESCALATION RULES
14
ENTITLEMENT RULES
15
RECORD HAS ROLL-UP SUMMARY OR IS PART OF A CROSS-OBJECT WORKFLOW
YES
NO
16
CALCULATE AND UPDATE PARENT RECORD ROLL-UP SUMMARY FIELD
SAVE PARENT RECORD
GRAND PARENT RECORD HAS ROLL-UP SUMMARY OR IS PART OF A CROSS OBJECT WORKFLOW
YES
NO
17
CALCULATE AND UPDATE GRRAND PARENT RECORD ROLL-UP SUMMARY FIELD
SAVE GRAND PARENT RECORD
CRITERIA BASED SHARING EVALUATION
18
COMMIT ALL DML OPERATIONS
19
POST COMMIT LOGIC
20
END