VALIDATION RULES

Validation Rules are used to verify the proper entry of data in the fields or not.

By using Validation Rules, we can maintain the quality and accuracy of the data inside the application.

Salesforce provides 2 Types of Validation Rules.

1. Standard/System Validation Rules: These are the readymade validation rules provided by Salesforce by default.

Ex: Making the field required.

2. Custom Validation Rules: The Developer/Administrator can create their own Validation Rules based on the business requirement to validate the data.

Validation Rules fire "Before Insert and Before Update" the record inside the object.

Note: An object can have maximum of 100 Active Validation Rules.

Salesforce provides a set of readymade functions to be used to validate the data.

1. **Boolean IsBlank(<FieldName>):** This method is used to verify the specified field is blank/ not. It returns TRUE, if the field is blank, else it returns FALSE.

Ex: IsBlank(Phone)

IsBlank(Email)

IsBlank(City)

IsBlank(HR_Email_ID__c)

Use Case: Create a Validation Rule on the account object, to make the Account Phone field is Required.

Validation Condition: IsBlank(Phone)

Navigation:

Click on "Setup" menu.

1. Go to the "Object Manager"
2. Click on the Required Object Name (Ex: Account)
3. Click on "Validation Rules" link from the Left Panel.
4. Click on "New Rule" button.
5. Enter the Validation Rule Name, and Description.
6. Select the Checkbox "Active".
7. Enter the "Validation Condition" inside the "Formula Editor".
8. Click on "Check Syntax" button, to validate the Syntax.
9. Enter the Error Message in the text box to be visible, if the validation is failed.

10. Select the Error Location (i.e., Top of the Page/Field Level)
11. Select the "Field Name" from the Picklist, to represent the error message.
12. Click on "Save" button.

Use Case: Create a Validation Rule on the contact object, to make the Email Field required.

Validation Condition: IsBlank(Email)

Use Case: Create a Validation Rule on the Position object, to make sure always "Minimum Pay" should be less than "Maximum Budget".

Validation Condition: Minimum_Pay__c > Maximum_Budget__c

Use Case: Create a Validation Rule on the Position object to make sure the Milestone Date should be greater than Open Date.

Validation Condition: Open_Date__C > Milestone_Date__C

Boolean IsNull(<FieldName>): This function is used to check whether the specified field value is empty or not. If the field is empty it returns TRUE, else it returns FALSE.

This function can be applicable only on "Numerical Fields" (Ex: Number, Percent, Currency).

Use Case: Create a Validation Rule on the lead object, to make sure the Lead Annual Revenue is required.

Validation Condition: IsNull(AnnualRevenue)

Boolean IsChanged(<FieldName>): This function returns TRUE if the specified field value has been changed, else it returns FALSE.

Ex: IsChanged(Phone)

IsChanged(City)

Use Case: Create a Validation Rule on the Hiring Manager object to prevent the changes to be made on Hiring Manager Name.

Validation Condition: IsChanged(Name)

Logical Functions: While validating the data, in few cases we have to prepare multiple conditions using logical functions.

We have the below 3 Logical Functions.

1. AND():

Syntax: AND(<Condition1>, <Condition2>, <Condition3>,.,<Condition n>)

It returns TRUE if all the Conditions are satisfied, else it returns FALSE.

2. OR():

Syntax: OR(<Condition1>, <Condition2>, <Condition3>,.....,<Condition n>)

It returns TRUE if any of condition is satisfied, else it returns FALSE.

3. NOT():

Syntax: NOT (<Condition>)

By using this function, we can change the result of the condition from TRUE to FALSE and vice-versa.

Use Case: Create a Validation Rule on the lead object, to make sure the "Mobile Phone" field is required, if the phone number has been entered.

```
AND (
        Not IsBlank(Phone),
        ISBlank(MobilePhone)
)
```

Boolean IsNew(): This is function is used to identify whether the user is creating a new record or updating an existing record.

It returns TRUE, if the user is creating a new record else it returns FALSE.

It will identify the record status based on the "Record ID" inside the URL as below.

Use Case: Create a Validation Rule on the Contact Object, to make sure the Phone and Fax should be required for New Contact Records.

```
AND (
    IsNew(),
        OR (
            IsBlank(Phone),
            IsBlank(Fax)
            )
    )
```

Boolean IsPickVal(<Picklist FieldName>,<Text Literal>): This Function returns TRUE, if the currently selected value in the Picklist field is equals to the text literal, else it returns FALSE.

Ex:

```
IsPickVal(LeadSource, 'Web')
IsPickVal(Rating,'Hot')
IsPickVal(Location__C,'Hyderabad')
```

Use Case: Create a Validation Rule on the Account Object to make sure the New Account Status should be always "Yes".

```
And (
        IsNew(),
        Not IsPickVal(Active__C,'Yes')
```

)

Use Case: Create a Validation Rule on the Account Object to make sure the Rating should be "Hot" for banking customers.

```
AND (
        IsPickVal(Industry, 'Banking'),
        Not IsPickVal(Rating,'Hot')
    )
```

Text(<FieldName>): This function is used to get the currently selected value in the Picklist field.

Ex: Text(Rating)

Text(Location__C)

Use Case: Create a Validation Rule on the lead object to make sure the Industry Field is Required.

ISBLANK(Text(Industry))

PriorValue(<FieldName>): This function is used to fetch the Previous/old value of the field.

Ex: PriorValue(Rating)

PriorValue(Phone)

PriorValue(Location__C)

Use Case: Create a Validation Rule on the Account object to prevent modifying the Rating value from "Cold To Hot".

```
And(
        IsPickVal ( PriorValue(Rating), 'Cold'),
        IsPickVal(Rating,'Hot')
    )
```

Regex(): By using this function, we can compare a field value with the specified expression.

Expression should be prepared with the help of "Wild Card Characters". Salesforce supports the below wild Card Characters.

[ ] ---> Represents a Group of Characters.

[A-P] --> Represents the word, where each character in the word should be between the range A-P

[0-9] --> Represents a number, where each digit in the number should be between the range 0 - 9.

[A R P] --> Represents a word, where each character in the word should be either A / R / P.

{Number} --> Represents the Specified Number of Characters to be exist in the word.

[A-Z, a-z] --> Represents a word, where each character inside the word should be between A-Z or a-z.

Ex: PAN Number : ALLPP4567E ---> [A-Z,a-z]{5}-[0-9]{4}-[A-Z,a-z]{1}

Credit Card Number : [0-9]{4}-[0-9]{4}-[0-9]{4}-[0-9]{4}

Use Case: Create a Validation Rule on the Candidate object, to Validate the PAN Number format.

Not Regex(PAN_Number__C, '[A-Z]{5}[0-9]{4}[A-Z,a-z]{1}')