

When to run **Actions** Advanced

Specify whether the business rule should run on **Insert** or **Update**. Use **Filter Conditions** to specify under which condition

When: before

Order: 100

Insert ☒ Update ☒ Delete ☐ Query ☐

Filter Conditions: Add Filter Condition Add "OR" Clause

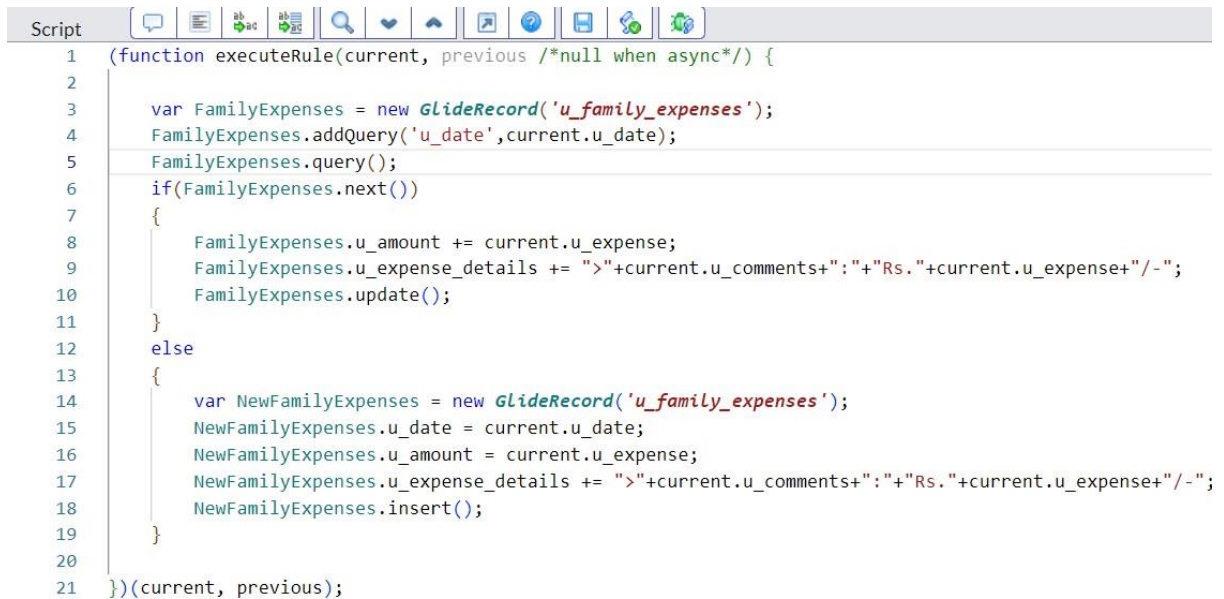
-- choose field -- -- oper -- -- value --

Role conditions

5. In Advance (we write the code): Write the below code >>
 

```
(function executeRule(current, previous /*null when async*/) {
  var FamilyExpenses = new GlideRecord('u_family_expenses');
  FamilyExpenses.addQuery('u_date', current.u_date);
  FamilyExpenses.query(); if(FamilyExpenses.next())
  {
    FamilyExpenses.u_amount += current.u_expense;
    FamilyExpenses.u_expense_details +=
    ">" + current.u_comments + ":" + "Rs." + current.u_expense + "/-";
    FamilyExpenses.update();
  }
  else
  {
    var NewFamilyExpenses = new GlideRecord('u_family_expenses');
    NewFamilyExpenses.u_date = current.u_date;
    NewFamilyExpenses.u_amount = current.u_expense;
    NewFamilyExpenses.u_expense_details +=
    ">" + current.u_comments + ":" + "Rs." + current.u_expense + "/-";
    NewFamilyExpenses.insert();
  }
}
```

})(current, previous);



```
1 (function executeRule(current, previous /*null when async*/) {
2
3     var FamilyExpenses = new GlideRecord('u_family_expenses');
4     FamilyExpenses.addQuery('u_date',current.u_date);
5     FamilyExpenses.query();
6     if(FamilyExpenses.next())
7     {
8         FamilyExpenses.u_amount += current.u_expense;
9         FamilyExpenses.u_expense_details += ">" + current.u_comments + ":" + "Rs." + current.u_expense + "/-";
10        FamilyExpenses.update();
11    }
12    else
13    {
14        var NewFamilyExpenses = new GlideRecord('u_family_expenses');
15        NewFamilyExpenses.u_date = current.u_date;
16        NewFamilyExpenses.u_amount = current.u_expense;
17        NewFamilyExpenses.u_expense_details += ">" + current.u_comments + ":" + "Rs." + current.u_expense + "/-";
18        NewFamilyExpenses.insert();
19    }
20
21 })(current, previous);
```

6. Go to the Header and right click there>> click on Save

## Configure the Relationship

1. Go to All >> In the filter search for Relationships >> Open Relationships.
2. In that, open Daily Expenses Relationship.
3. For Applies to table : Select Family Expenses.
4. In Query with : write the below Query.

```
(function refineQuery(current, parent) {
```

```
    // Add your code here, such as current.addQuery(field, value);
    current.addQuery('u_date',parent.u_date);
```

```
    current.query();
```

```
})(current, parent);
```

5. Click on Update.

Relationship  
Daily Expenses

Name: Daily Expenses

Application: Global

Advanced: ☐

Applies to table: Family Expenses [u\_family\_expenses]

Queries from table: Daily Expenses [u\_daily\_expenses]

This script refines the query in current that will populate the related list. For more information about it, its parameters and control variables, see the documentation. See also the article about the recommended form of the script.

Query with

```
1 (function refineQuery(current, parent) {  
2  
3     // Add your code here, such as current.addQuery(field, value);  
4     current.addQuery('u_date', parent.u_date);  
5     current.query();  
6  
7 })(current, parent);
```

Update Delete

## Conclusion:

Implementing a **Family Expenses Tracking System** using **ServiceNow** demonstrates the flexibility and power of the platform beyond its traditional ITSM use cases. By leveraging ServiceNow's capabilities—such as custom applications, form design, workflow automation, and reporting—you can create a structured and userfriendly solution to manage and monitor household spending.