

# Calculating Family Expenses Using ServiceNow

## Problem statement:

Manually managing family expenses often makes it challenging to track daily spending, categorize expenses, and maintain precise financial records. Conventional approaches like paper notes or spreadsheets lack automation, transparency, and real-time insights. As a result, families may face issues such as overspending, ineffective budgeting, and limited understanding of monthly financial patterns. To overcome these problems, a centralized and automated solution within ServiceNow is essential to efficiently record, organize, and analyze family expenses.

## Execution:

### Creation of New Update Set:

1. Go to All >> In the filter search for Local Update set > click on New.
2. Enter the Details as:  
Name: Family Expenses
3. Then click on Submit and Make current.

The screenshot shows the ServiceNow interface for creating a new update set. The form is titled 'Update Set - Family Expenses'. It includes fields for Name (Family Expenses), State (In progress), Application (Global), Created (2025-10-25 04:31:09), Created by (admin), and Merged to. There are also fields for Parent, Release date, Install date, Installed from, and Description. Below the form, there are 'Update' and 'Related Links' buttons. The 'Related Links' section includes 'Merge With Another Update Set' and 'Scan Update Set'. At the bottom, there is a table with columns for 'Created', 'Time', 'Version', 'Target name', 'Installed by', 'Parent update set', and 'Action'. The table is currently empty, and a message at the bottom states 'No templates are available. Create A New One?'.

### Creation of Family Expenses Table:

1. Go to All > In the filter search for Tables > click on New.
2. Enter the Details:  
Label: Family Expenses

Name: Auto-Populated

New menu name: Family Expenditure

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

The screenshot shows the 'Table' configuration page for 'Family Expenses'. The header includes a back arrow, a hamburger menu, the title 'Table Family Expenses', and action buttons: 'Delete', 'Update', 'Delete All Records', and sort arrows. A blue informational banner states: 'A table is a collection of records in the database. Each record corresponds to a row in a table, and each field on a record corresponds to a column on that table. Applications use tables and records to manage data and processes. [More Info](#)'. Below this, the configuration fields are: '\* Label' (Family Expenses), '\* Name' (u\_family\_expenses), 'Application' (Global), and 'Remote Table' (empty).

## Creation of Daily Expenses Table:

1. Go to All > In the filter search for Tables > click on New.
2. Enter the Details:
  - Label: Daily Expenses
  - Name: Auto-Populated
  - Add Module to menu: Family Expenditure
3. Go to the Header and right click there>> click on Save.

The screenshot shows the 'Table' configuration page for 'Daily Expenses'. The header includes a back arrow, a hamburger menu, the title 'Table Daily Expenses', and action buttons: 'Delete', 'Update', 'Delete All Records', and sort arrows. A blue informational banner states: 'A table is a collection of records in the database. Each record corresponds to a row in a table, and each field on a record corresponds to a column on that table. Applications use tables and records to manage data and processes. [More Info](#)'. Below this, the configuration fields are: '\* Label' (Daily Expenses), '\* Name' (u\_daily\_expenses), 'Application' (Global), and 'Remote Table' (empty).

The screenshot shows the 'Table Columns' configuration page for 'Daily Expenses'. The header includes a back arrow, a hamburger menu, the title 'Table Columns', and action buttons: 'Delete', 'Update', 'Delete All Records', and sort arrows. A blue informational banner states: 'A table is a collection of records in the database. Each record corresponds to a row in a table, and each field on a record corresponds to a column on that table. Applications use tables and records to manage data and processes. [More Info](#)'. Below this, the configuration fields are: '\* Label' (Daily Expenses), '\* Name' (u\_daily\_expenses), 'Application' (Global), and 'Remote Table' (empty). The 'Table Columns' section shows a list of columns with the following details:

Column label	Type	Reference	Max length	Default value	Display
Number	String	(empty)	40	javascript:getNextObjNumberPadded();	false
Expense	Integer	(empty)	40		false
Updates	Integer	(empty)	40		false
Created by	String	(empty)	40		false
Updated by	String	(empty)	40		false
Comments	String	(empty)	800		false
Date	Date	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Updated	Date/Time	(empty)	40		false
Created	Date/Time	(empty)	40		false
Family Member Name	Reference	User	32		false

## Creation of Relationship between Family Expenses and Daily Expenses tables:

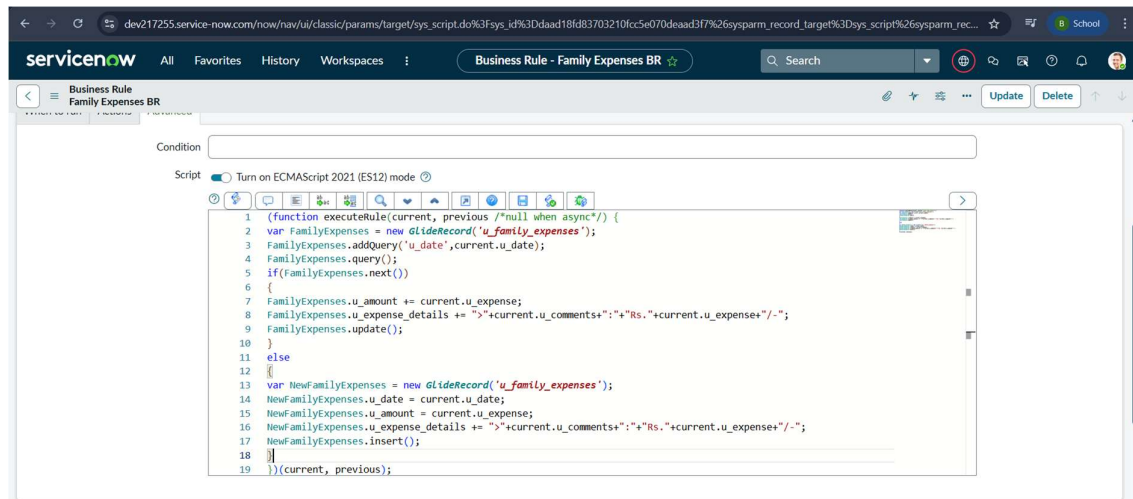
1. Go to All >> In the filter search for Relationships >> Open Relationships
2. Click on New.
3. Enter the details:  
Name: Daily Expenses  
Applies to table: Select Family Expenses  
Daily Expenses: Select Daily Expenses
4. Click Save.

The screenshot shows the 'Relationship - Daily Expenses' form in ServiceNow. The 'Name' field is set to 'Daily Expenses'. The 'Application' is 'Global'. The 'Applies to table' dropdown is set to 'Family Expenses [u\_family\_expenses]'. The 'Queries from table' dropdown is set to 'Daily Expenses [u\_daily\_expenses]'. There is a blue informational banner at the bottom stating: '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.'

## Creation of Business Rules:

1. Go to All >> In the filter search for Business Rules.
2. Under System Definition Select Business Rules then click on New.
3. Enter the Details:  
Name: Family Expenses BR  
Table: Select Daily Expenses  
Check Advanced

The screenshot shows the 'Business Rule - Family Expenses BR' form in ServiceNow. The 'Name' field is set to 'Family Expenses BR'. The 'Table' dropdown is set to 'Daily Expenses [u\_daily\_expenses]'. The 'Application' is 'Global'. The 'Active' checkbox is checked. The 'Advanced' checkbox is checked. A blue informational banner at the top states: 'A business rule is a server-side script that runs when a record is displayed, inserted, deleted, or when a table is queried. Use business rules to automatically change values in form fields when the specified conditions are met. More Info'.



4. In when to run Check Insert and Update

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);

```

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) {  
    current.addQuery('u\_date,parent.u\_date);  
    current.query();  
    }) (current, parent);
5. Click on Update.

Relationship - Daily Expenses

Name:

Advanced: ☐

Application: Global

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: ☒ Turn on ECMAScript 2021 (ES12) mode

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

Run Query Diagnostics Update Delete

## Conclusion:

The Family Expense Calculation System using ServiceNow provides an efficient and automated solution for managing household financial activities. By leveraging ServiceNow's capabilities like custom tables, business rules, UI policies, and reports, the system ensures accurate tracking, budget control, and real-time insights. This project demonstrates how ServiceNow can be utilized to streamline everyday financial management, reduce manual effort, and promote smarter budgeting decisions for families.