ANJALAI AMMAL MAHALINGAM ENGINEERING COLLEGE DEPARTMENT OF INFORMATION TECHNOLOGY

NM-SERVICE NOW ADMINISTRATOR

TITLE: CALCULATING FAMILY EXPENSES USING SERVICE NOW

Team Members:

Team ID: NM2025TMID05267521

NAME	REGISTER NUMBER
MOHANAVEL M	820422205044
MOHAMED MUSARAF V N	820422205303
RAMESHKANNA S	820422205304
CHERAN K	820422205012

CALCULATING FAMILY EXPENSES USING SERVICE NOW

Abstract

This project utilized the ServiceNow App Engine to design and implement a custom Family Expense Management System, aimed at modernizing and simplifying the way households manage their financial activities. Traditionally, families rely on manual methods such as spreadsheets or handwritten notes to record and analyze expenses, which often leads to errors, inefficiencies, and a lack of real-time insights. To address these limitations, this system provides a centralized digital platform that automates expense tracking, categorizes transactions, and allows users to set, monitor, and adjust budget targets dynamically.

Through the use of ServiceNow's low-code capabilities, the application supports seamless data entry, secure storage, and generation of interactive financial reports for better visualization of spending patterns. The platform also incorporates notification features for budget thresholds and expense anomalies, enabling users to take timely corrective actions.

The findings demonstrate that ServiceNow is not limited to IT Service Management (ITSM) but is also a powerful framework for building scalable, user-friendly, and process-driven applications in non-IT domains. By integrating automation, analytics, and user-centric design, the Family Expense Management System empowers families to achieve greater financial transparency, improved accountability, and more informed decision-making in their daily lives.

Introduction

The complexity of modern household finance demands a centralized and efficient management solution. Effective money management is often cumbersome due to decentralized data across multiple platforms (bank apps, personal notes), which leads to significant friction and budget failure.

ServiceNow was chosen for this project due to its inherent strengths, which are highly beneficial even outside of traditional IT Service Management (ITSM):

1. Scalable Data Integrity: The platform offers a secure, reliable database structure perfect for sensitive financial records.
2. Robust Workflow Automation: Flow Designer enables complex, low-code logic for processes like purchase approvals and budget alerts.
3. User Experience (UX) Framework: The Service Portal provides a modern, responsive interface accessible across desktop and mobile devices.

The project's main objectives were to establish a single source of truth for all expenses, ensure proactive budget adherence via automated alerts, and empower all family members with clear financial visibility.

Problem Statement

In most households today, managing daily and monthly expenses remains a largely manual and fragmented process, relying on spreadsheets, notebooks, or multiple third-party mobile applications. While these tools offer basic tracking capabilities, they fail to provide a centralized and automated system that can offer a real-time understanding of the family's overall financial position. As a result, families often struggle to maintain control over their spending, leading to budget overruns, poor financial visibility, and a lack of proactive decision-making.

The core problem lies in the decentralized and retrospective nature of traditional household financial tracking. Financial records are often scattered across multiple tools, each serving an isolated purpose such as recording expenses, storing receipts, or calculating totals. This fragmentation prevents families from having a single, unified view of their income and expenses. Consequently, by the time a family realizes that they have exceeded their budget in a particular category—such as groceries, utilities, or entertainment—it is already too late to take corrective action.

Methodology/System Design

Design Approach

The Family Expense Management System was designed and developed using an Agile methodology, with a strong focus on the end-user experience — in this case, the family members who interact with the system on a daily basis. The project followed an iterative development cycle consisting of planning, prototyping, testing, and refinement, ensuring that user feedback and evolving requirements were continuously integrated into the design process. This approach promoted adaptability, faster development cycles, and the delivery of a user-friendly and feature-rich application.

The design philosophy centered on three key principles: simplicity, speed of data entry, and visual feedback. Since the target users are non-technical individuals, the interface was designed to be intuitive, with minimal steps required to perform common tasks such as adding an expense or viewing a report. The goal was to minimize user friction and make expense management a seamless part of daily household routines rather than a burdensome chore. Real-time visual elements like dynamic dashboards, charts, and notifications were incorporated to provide immediate insights and reinforce proactive financial awareness

System Architecture

At a high level, the architecture can be divided into four key layers: Presentation Layer, Application Logic Layer, Data Layer, and Integration Layer. Each layer performs a distinct role but works cohesively to deliver a smooth user experience.

1. Presentation Layer (User Interface)

The Service Portal serves as the main user-facing interface for the Family Expense Management System. It provides a clean, intuitive design that enables users to interact with the system through dashboards, forms, and widgets.

- Key Components:
  - Expense Entry Forms: Allow users to quickly log new expenses, select categories (e.g., groceries, utilities, transportation), and attach receipts if needed.
  - Budget Dashboard: Displays real-time data visualizations such as bar charts and pie charts, showing spending distribution and remaining budget for each category.
  - Notifications & Alerts: Provides visual cues and pop-up messages when budget thresholds are reached or exceeded.
  - Responsive Design: Ensures compatibility with desktop, tablet, and mobile devices for ease of access by all family members.

2. Application Logic Layer

This layer is the core of the application, where business rules, automation logic, and workflows are implemented. It defines how data is processed, validated, and displayed.

- Key Components:
  - Flow Designer: Used to create and manage all automation workflows—such as categorizing expenses, sending notification alerts, or generating monthly reports.
  - Business Rules & Script Includes: Enforce validation rules (e.g., preventing overspending beyond the set budget) and handle calculations dynamically.
  - Scheduled Jobs: Automatically generate periodic summaries and send them to family members via email notifications.
  - Access Control Rules: Define user roles (e.g., Admin, Parent, Member) and restrict access to certain modules or records to maintain data security.

3. Data Layer (Database and Storage)

The Data Layer manages how all financial information is stored, retrieved, and related within the ServiceNow platform. The data model was custom-designed to suit household budgeting requirements.

- Custom Tables Created:
    - Expense Records [x_family_finance_expense]: Stores detailed expense entries including date, amount, category, and payment method.
    - Budget Targets [x_family_finance_budget]: Holds category-wise budget limits and current utilization statistics.
    - User Profiles [x_family_finance_user]: Manages family member information and role-based access settings.
    - Reports [x_family_finance_report]: Contains pre-generated or custom report data for quick dashboard rendering.

4. Integration Layer

Although primarily a standalone application, the system is designed with future scalability in mind and supports integration with external platforms for advanced functionality.

- Potential Integrations:
    - Email Notifications: Integrated via ServiceNow’s native Notification engine to deliver alerts and monthly summaries.
    - External Data Sources (Optional): Can be connected through ServiceNow’s REST API integration framework for importing data from online banking or payment applications.
    - Analytics Extensions: Future enhancements may include integration with ServiceNow Performance Analytics or third-party visualization tools such as Power BI for advanced data insights.

User Interface (UI) and User Experience (UX)

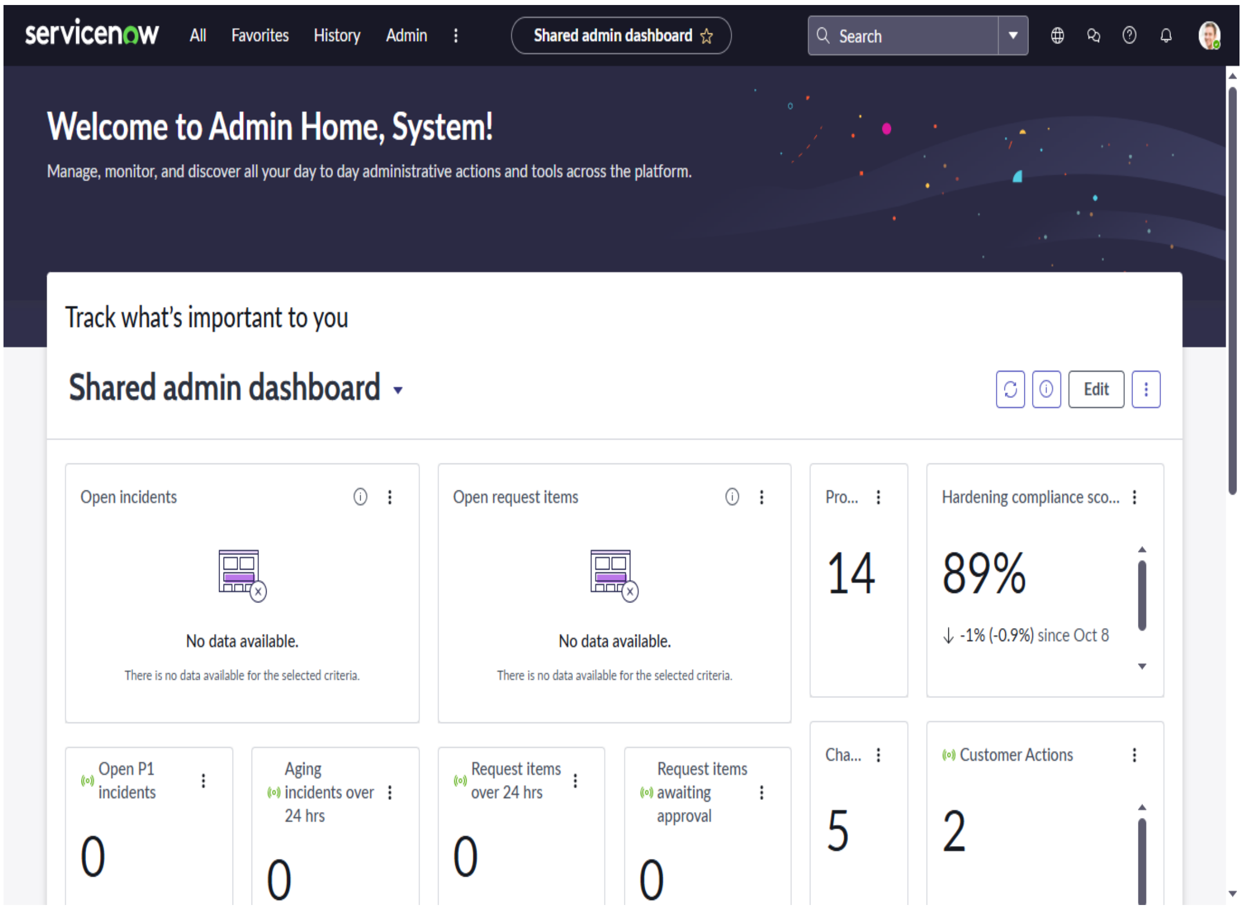
The UX is centered on a low-friction, high-impact approach:

- Forms: The Expense Entry form is streamlined, using Client Scripts to auto-populate fields (like Date and User) and ensure mandatory categorization, speeding up the logging process.
- Layouts: The Service Portal Dashboard is the main view, featuring responsive widgets with clear visualizations from the ServiceNow Reporting engine. Key KPIs—such as "Budget Remaining" and "Monthly Spending Breakdown"—are prominently displayed using donut charts and gauges.
- User Flows: The core flow is designed to ensure that data entry immediately triggers backend logic, so the user sees their updated remaining budget within seconds, reinforcing positive financial habits.

Implementation Details

Setting up ServiceNow Instance

1. Sign up for a developer account on the ServiceNow Developer site "https://developer.servicenow.com".
2. Once logged in, navigate to the "Personal Developer Instance" section.
3. Click on "Request Instance" to create a new ServiceNow instance



Welcome to Admin Home, System!

Manage, monitor, and discover all your day to day administrative actions and tools across the platform.

Track what's important to you

Shared admin dashboard

Open incidents

No data available.

There is no data available for the selected criteria.

Open request items

No data available.

There is no data available for the selected criteria.

Pro... 14

Hardening compliance sco... 89% ↓ -1% (-0.9%) since Oct 8

Open P1 incidents 0

Aging incidents over 24 hrs 0

Request items over 24 hrs 0

Request items awaiting approval 0

Cha... 5

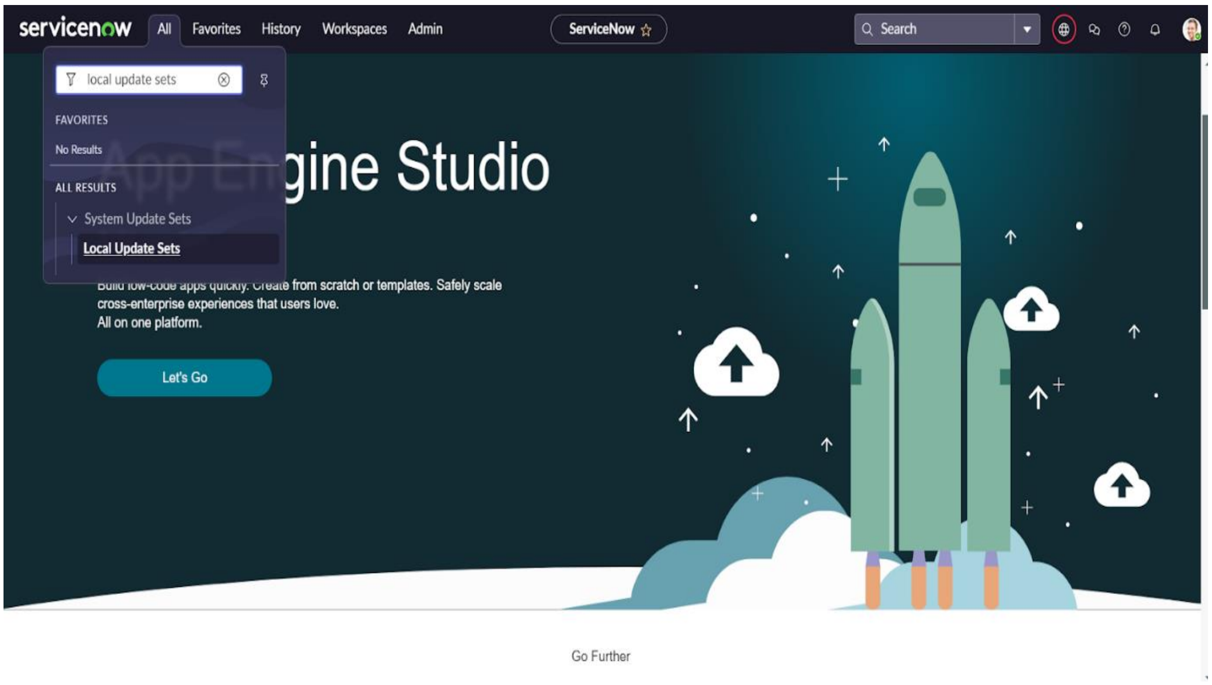
Customer Actions 2

.

4. Fill out the required information and submit the request.
5. You'll receive an email with the instance details once it's ready.
6. Log in to your ServiceNow instance using the provided credentials.
7. Now you will navigate to the ServiceNow.

Creation of New Update Set

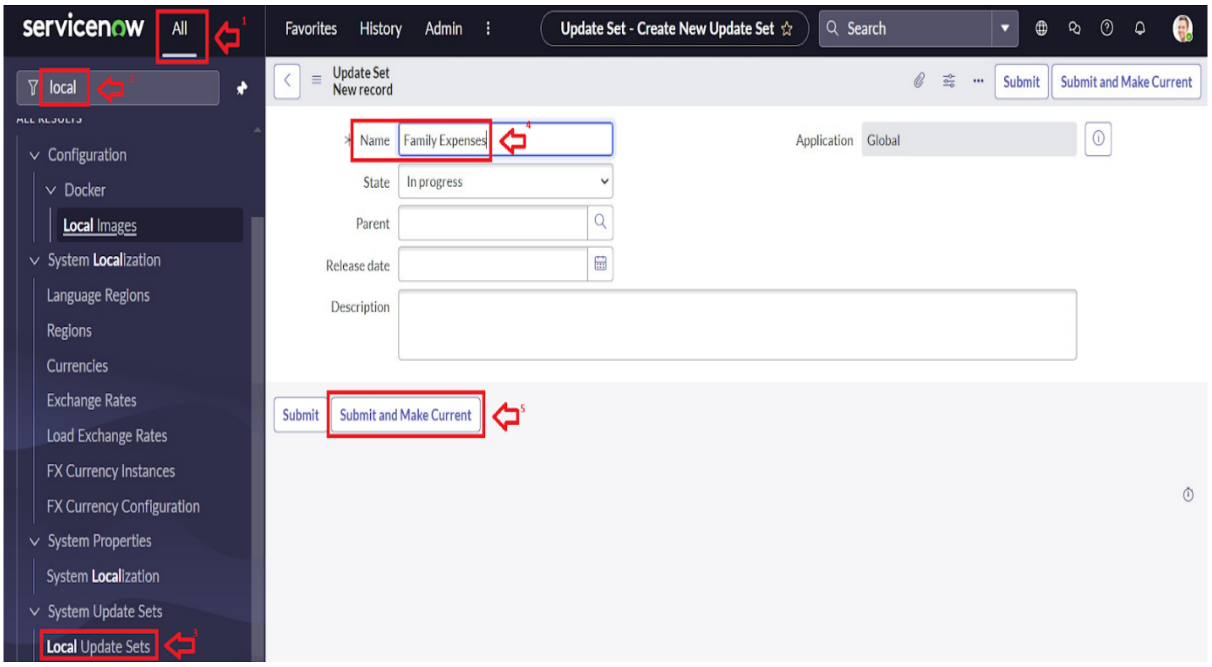
1. Go to All >> In the filter search for Local Update set > click on New.



2. Enter the Details as:

   Name : Family Expenses

3. Then click on Submit and Make current.



Name: Family Expenses
State: In progress
Application: Global

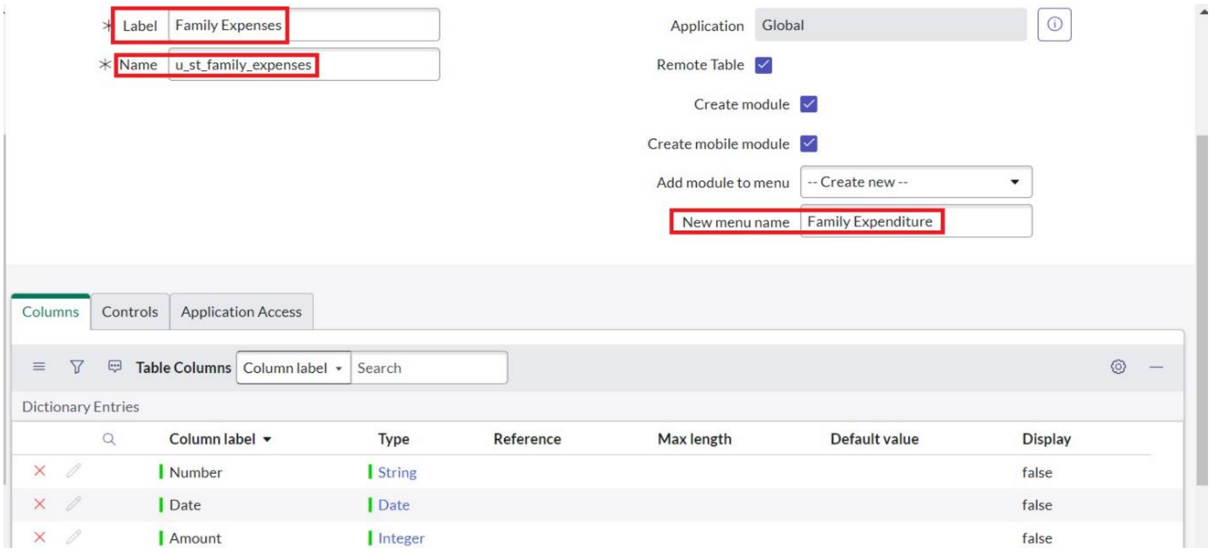
Creation of Family Expenses Table

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



Label: Family Expenses
Name: u_st_family_expenses
Application: Global
Remote Table: ✓
Create module: ✓
Create mobile module: ✓
Add module to menu: -- Create new --
New menu name: Family Expenditure

Column label	Type	Reference	Max length	Default value	Display
Number	String				false
Date	Date				false
Amount	Integer				false

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

Creation of Columns(Fields)

1. Near Columns Double click near insert a new row.
2. Give the details as:

Column label : Number

Type : String

3. Double click on insert a new row again
4. Give the details as:

Column label : Date

Type : Date

5. Double click on insert a new row again
6. Give the details as:

   Column label : Amount

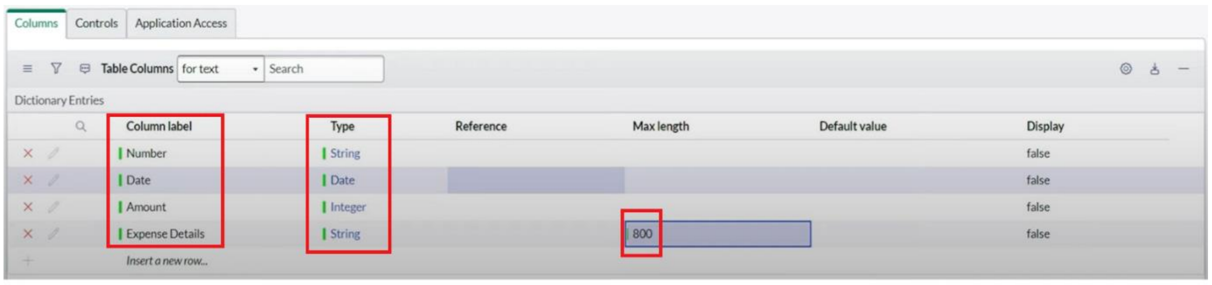
   Type : Integer

7. Double click on insert a new row again
8. Give the details as:

   Column label : Expense Details

   Type : String

   Max length : 800



Column label	Type	Reference	Max length	Default value	Display
Number	String				false
Date	Date				false
Amount	Integer				false
Expense Details	String		800		false

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

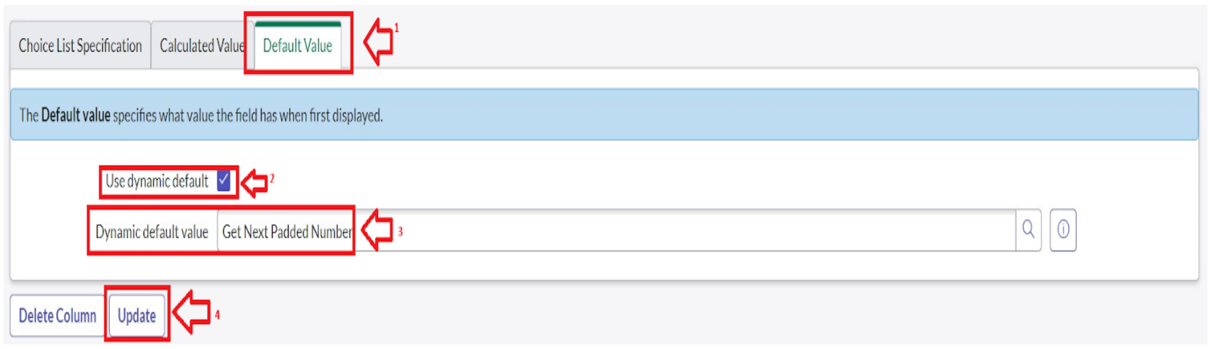
Making Number Field an Auto-Number

1. Double click on the Number Field/Column.
2. Go down and double click on Advanced view
3. In Default Value:

   Use dynamic default : check the box

   Dynamic default value : Get Next Padded Number

4. Click on Update.



The **Default value** specifies what value the field has when first displayed.

Use dynamic default ✓

Dynamic default value: Get Next Padded Number

5. Go to All >> In the filter search for Number Maintenance >> select Number Maintenance
6. Click on New.
7. Enter the below Details:

   Table : Family Expenses

   Prefix : MFE

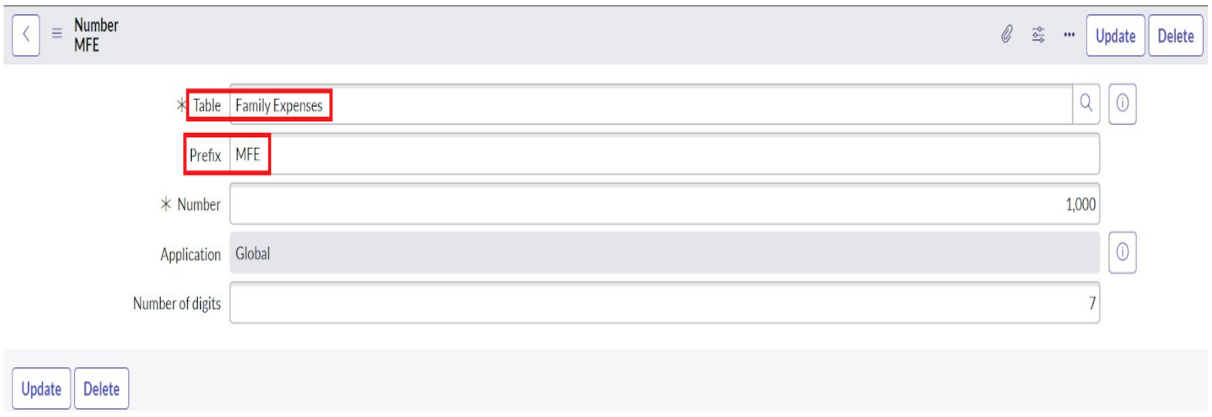
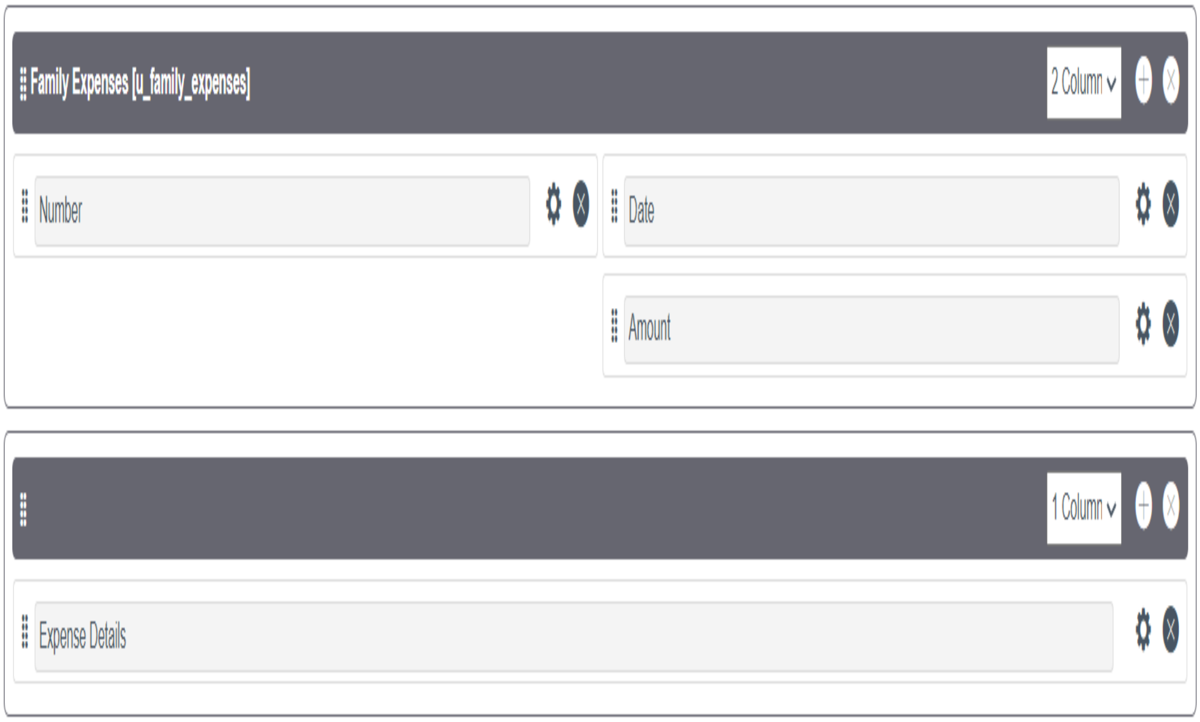


Table	Family Expenses
Prefix	MFE
Number	1,000
Application	Global
Number of digits	7

8. Click on Submit

Configure the Form

1. Go to All >> In the filter search for Family Expenses >> Open Family Expenses
2. Click on New
3. Go to the Header and right click there>> click on Configure >> Select Form Design
4. Customize or Drag Drop the form as per your requirement.



5. Make Number Read-Only Field by clicking on the gear icon and checking Read-Only
6. Make Date, Amount Mandatory Field by clicking on the gear icon and checking Mandatory
7. Click on Save.

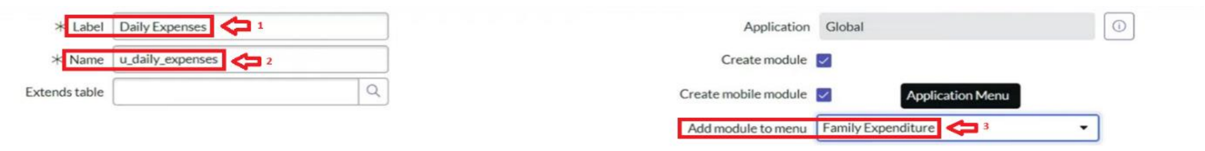
Creation of Daily Expenses Table

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


Label: Daily Expenses 1
Name: u_daily_expenses 2
Extends table:
Application: Global
Create module ✓
Create mobile module ✓
Application Menu
Add module to menu: Family Expenditure 3

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

Creation of Columns(Fields)

1. Near Columns Double click near insert a new row.
2. Give the details as:

   Column label : Number

   Type : String

3. Double click on insert a new row again
4. Give the details as:

   Column label : Date

   Type : Date

5. Double click on insert a new row again
6. Give the details as:

   Column label : Expense

   Type : Integer

7. Double click on insert a new row again
8. Give the details as:

   Column label : Family Member Name

   Type : Reference

   Max length : 800

9. Double click on insert a new row again
10. Give the details as:

    Column label : Comments

    Type : String

    Max length : 800

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

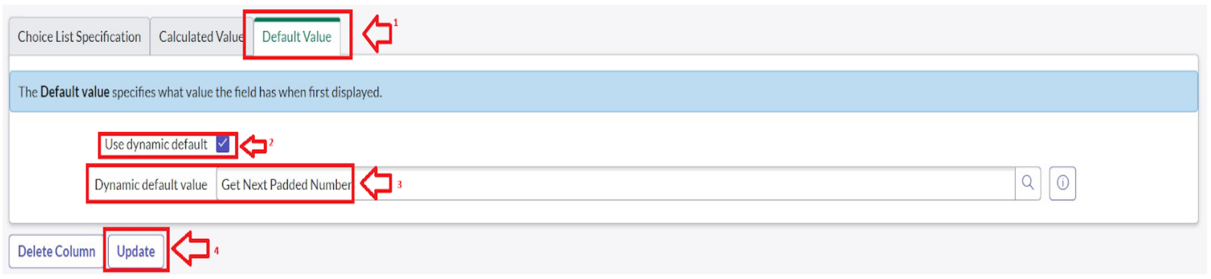
Making Number Field an Auto-Number

1. Double click on the Number Field/Column.
2. Go down and double click on Advanced view
3. In Default Value:

    Use dynamic default : check the box

    Dynamic default value : Get Next Padded Number

4. Click on Update.



The **Default value** specifies what value the field has when first displayed.

Use dynamic default ☑

Dynamic default value: Get Next Padded Number

5. Go to All >> In the filter search for Number Maintenance >> select Number Maintenance
6. Click on New.
7. Enter the below Details:

    Table : Family Expenses

Prefix : MFE

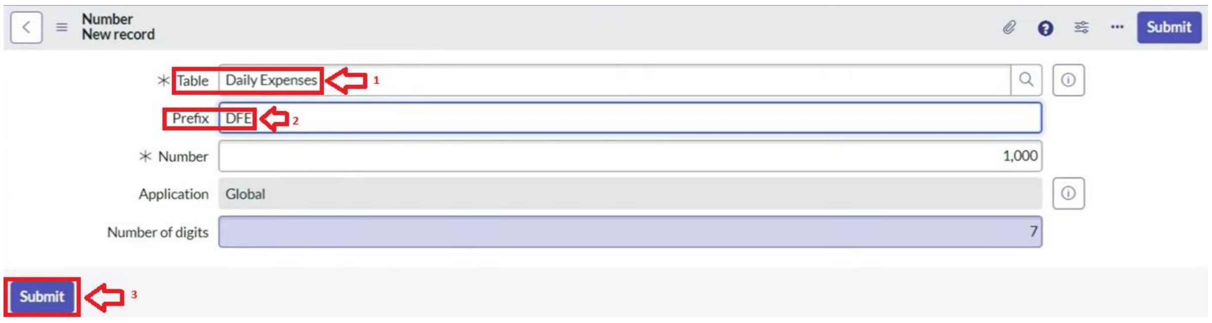
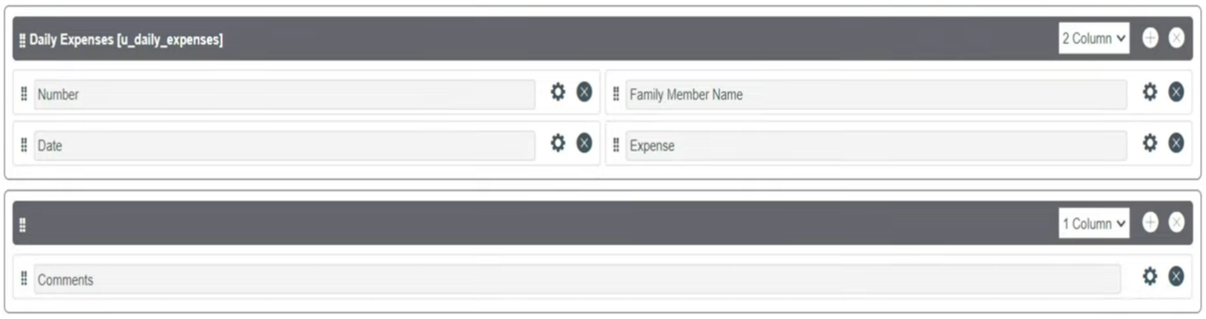

Number
New record

Table	Daily Expenses
Prefix	DFE
Number	1,000
Application	Global
Number of digits	7

8. Click on Submit.

Configure the Form

1. Go to All >> In the filter search for Daily Expenses >> Open Daily Expenses
2. Click on New
3. Go to the Header and right click there>> click on Configure >> Select Form Design
4. Customize or Drag Drop the form as per your requirement.


Daily Expenses [u_daily_expenses] — 2 Column

Number	Family Member Name
Date	Expense

1 Column

Comments

5. Make Number Read-Only Field by clicking on the gear icon and checking Read-Only
6. Make Date, Family Member Name Mandatory Field by clicking on the gear icon and checking Mandatory
7. Click on Save.

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

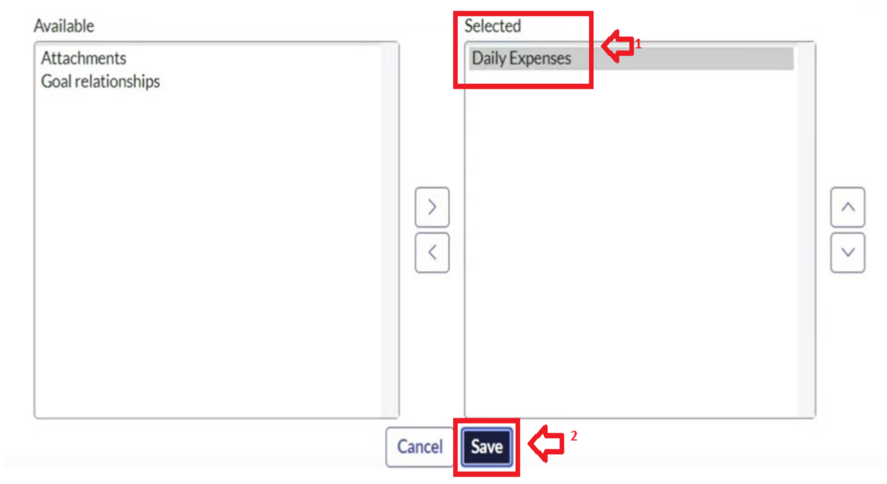
    Applies to table : Select Family Expenses

    Daily Expenses : Select Daily Expenses

4. Click Save.

Configuring Related List on Family Expenses

1. Go to All >> In the filter search for Family Expenses >> Open Family Expenses
2. Click on New
3. Go to the Header and right click there>> click on Configure >> Select Related Lists
4. Add Daily Expenses to the Selected Area.
5. Click on Save



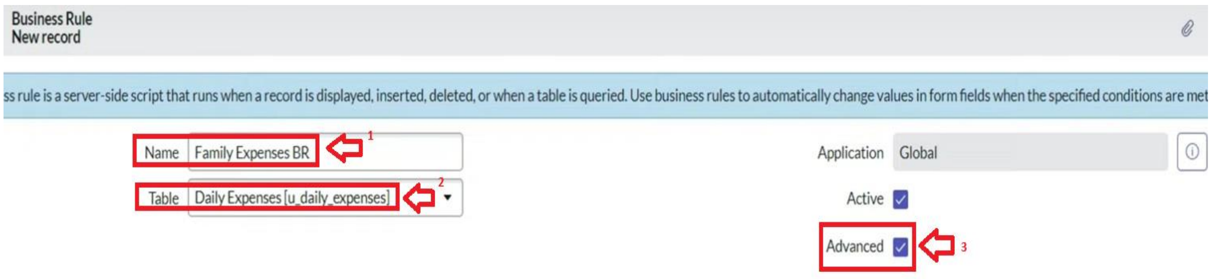
Creation of Business Rules

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

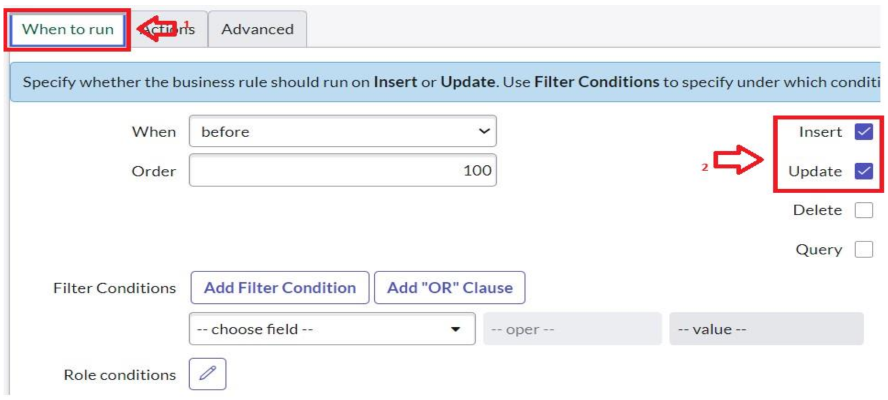


Business Rule
New record

ss 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

Name	Family Expenses BR	Application	Global
Table	Daily Expenses [u_daily_expenses]	Active	✓
		Advanced	✓

4. In when to run Check Insert and Update

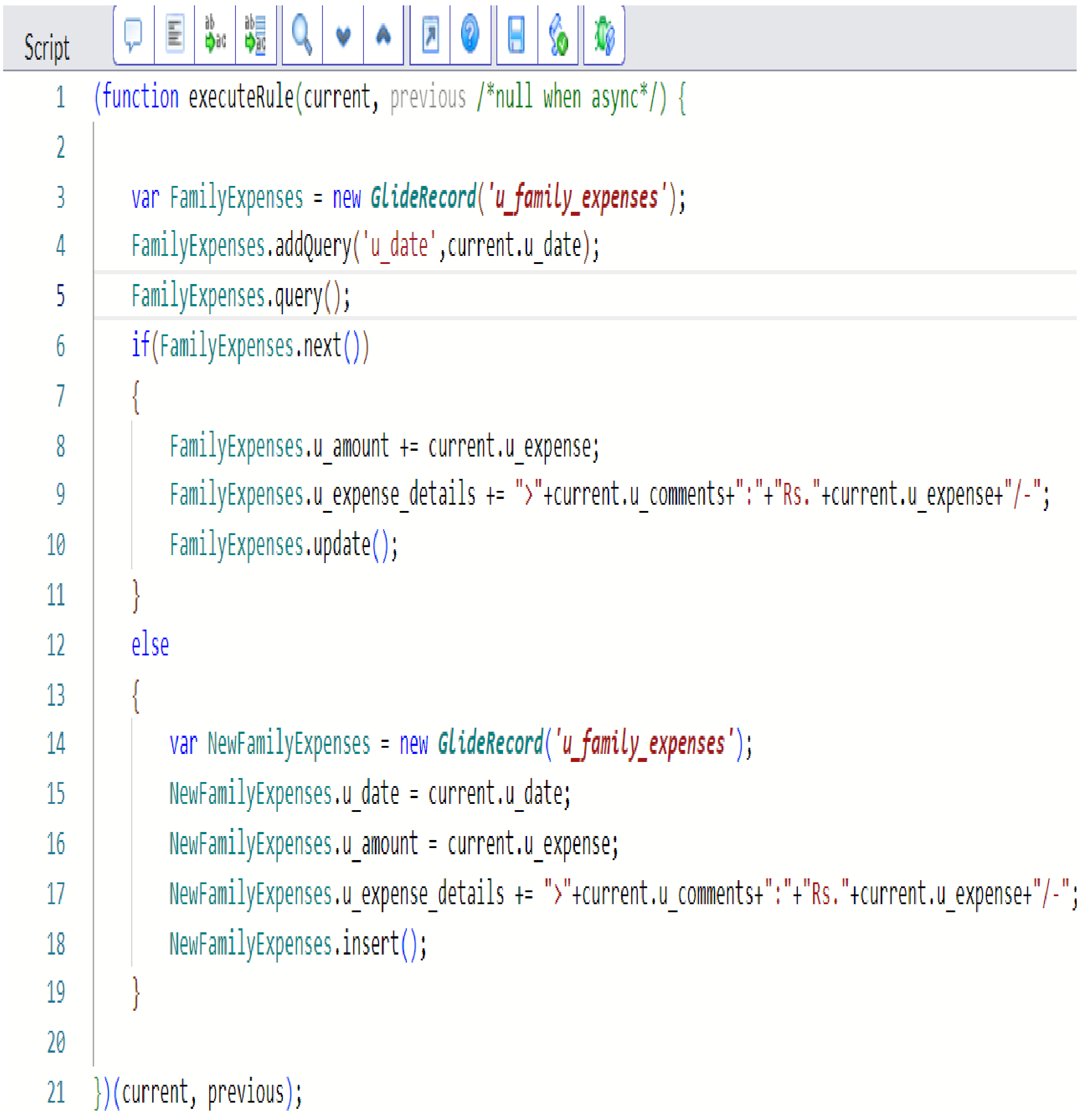


Specify whether the business rule should run on **Insert** or **Update**. Use **Filter Conditions** to specify under which conditi

When	before	Insert	✓
Order	100	Update	✓
		Delete	
		Query	

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);
```



```
(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);
```

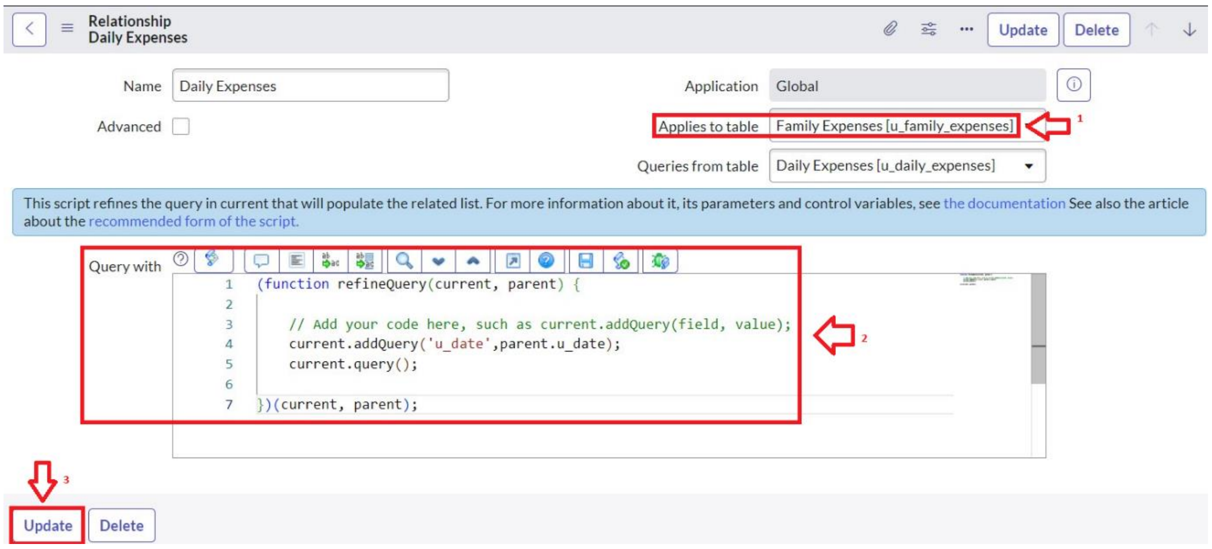
6. Go to the Header and right click there>> click on Save.

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) {

    // 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.



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

```
(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);
```

Conclusion and Future Scope

The Family Expenses using Service Now successfully demonstrates how ServiceNow can automate and streamline the management of family expenses. The system consolidates daily expenses, updates totals automatically, and maintains detailed records, reducing manual work and improving data accuracy. Automated scripts and workflows make the process efficient, user-friendly, and scalable.

Future Scope

- Integrating the system with financial management or banking APIs for automatic transaction import.
- Implementing reporting dashboards for expense analysis by date, category, or month.
- Adding notification systems to alert users when spending limits are exceeded.
- Expanding workflow automation to categorize expenses, manage budgets, and track recurring costs.