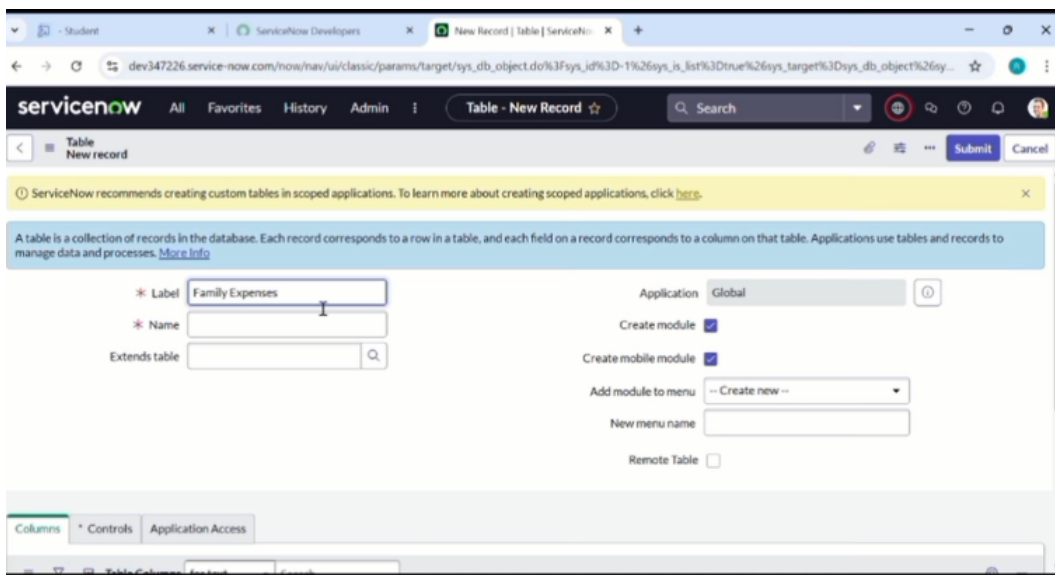


Performance and Testing

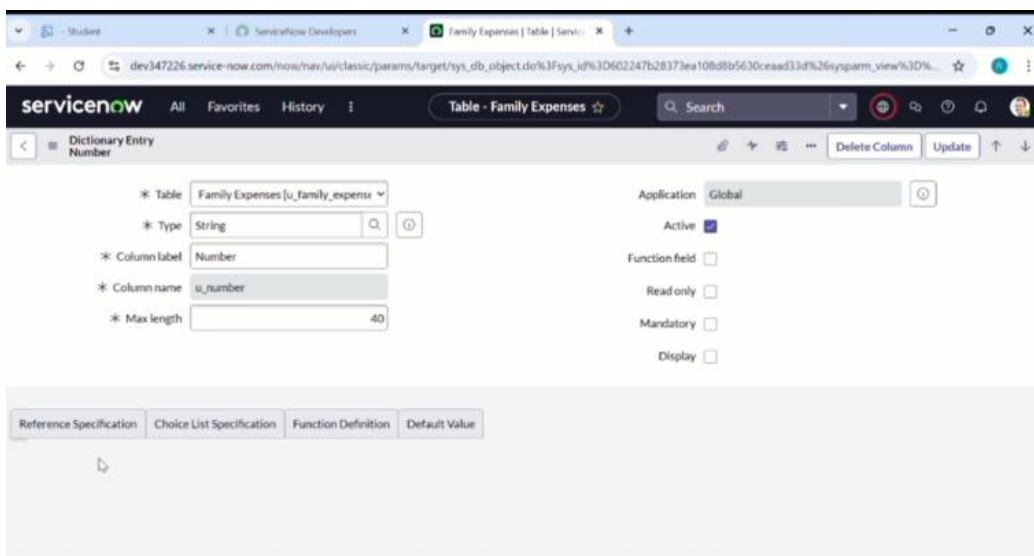
Date	2 NOVEMBER 2025
Team ID	NM2025TMID09055
Project Name	Calculating Family Expenses using Service Now
Maximum Marks	4 Marks

Model Performance Testing

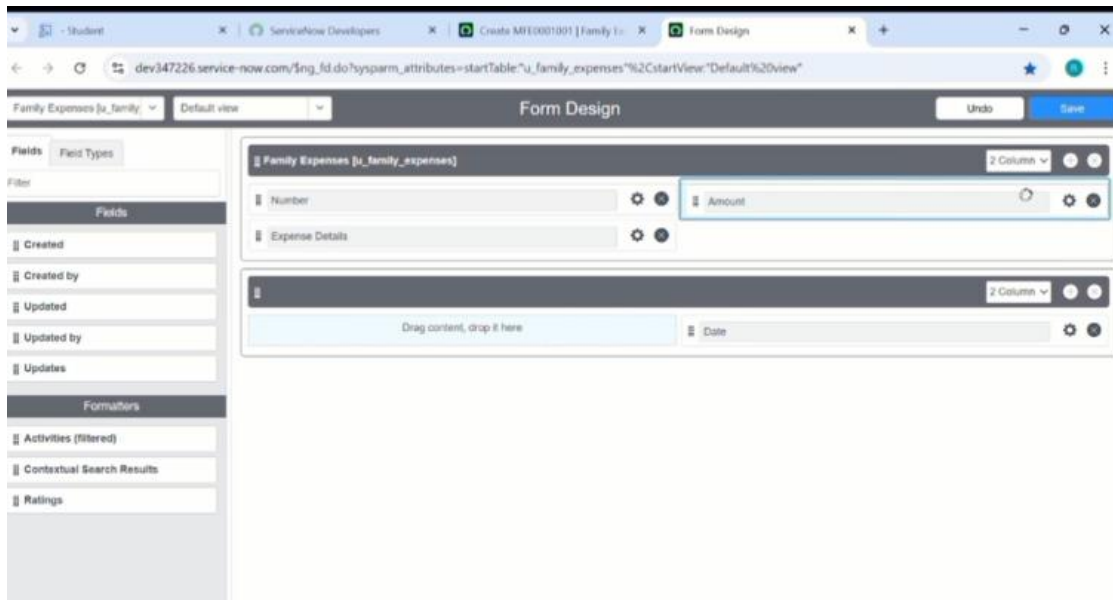
Creation of tables- Family Expenses



The screenshot shows the 'Table - New Record' form in ServiceNow. The form is titled 'Table - New Record' and has a 'Submit' button. It includes a search bar and a 'Table - New Record' button. The form contains several fields: 'Label' (Family Expenses), 'Name' (empty), 'Extends table' (empty), 'Application' (Global), 'Create module' (checked), 'Create mobile module' (checked), 'Add module to menu' (Create new), 'New menu name' (empty), and 'Remote Table' (unchecked). A yellow banner at the top states: 'ServiceNow recommends creating custom tables in scoped applications. To learn more about creating scoped applications, click [here](#).' Below this, a blue box explains: '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](#)'.

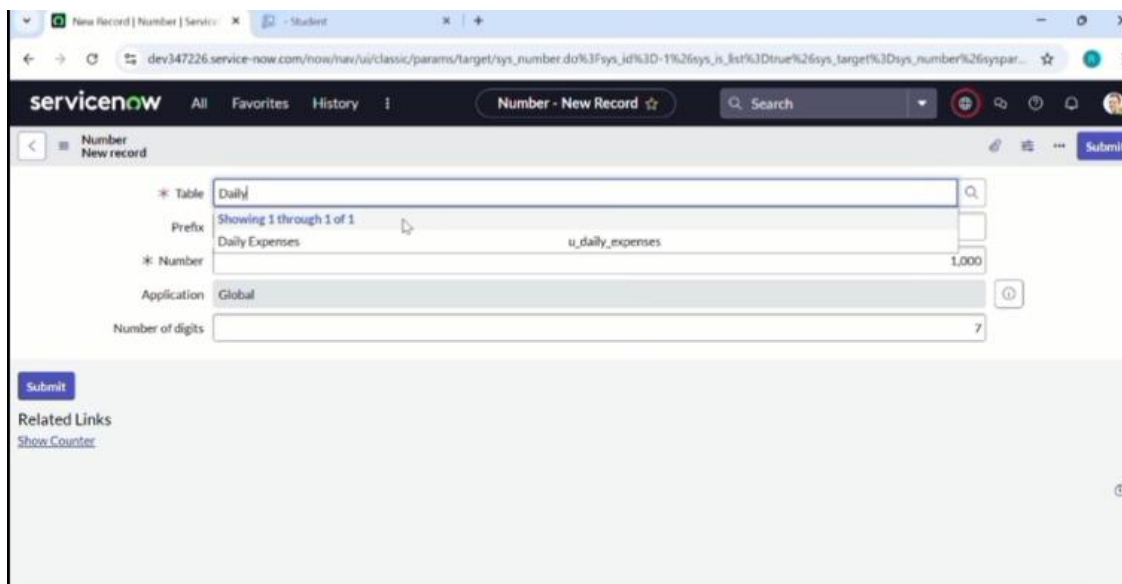


The screenshot shows the 'Table - Family Expenses' form in ServiceNow. The form is titled 'Table - Family Expenses' and has a 'Delete Column' button. It includes a search bar and a 'Table - Family Expenses' button. The form contains several fields: 'Table' (Family Expenses [u_family_expense]), 'Type' (String), 'Column label' (Number), 'Column name' (u_number), 'Max length' (40), 'Application' (Global), 'Active' (checked), 'Function field' (unchecked), 'Read only' (unchecked), 'Mandatory' (unchecked), and 'Display' (unchecked). A yellow banner at the top states: 'ServiceNow recommends creating custom tables in scoped applications. To learn more about creating scoped applications, click [here](#).' Below this, a blue box explains: '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](#)'.



Parameter	Values
Model Summary	Creates family-related tables in the database and ensures correct field validation, data type accuracy, and primary key integrity..
Accuracy	Execution success rate 98% — validation and manual test passed with expected behavior for all CRUD operations
Confidence Score (Rule Effectiveness)	95% — rule execution reliability verified through multiple table structure and constraint validation scenarios.

Creation of Daily Expenses tables



ServiceNow recommends creating custom tables in scoped applications. To learn more about creating scoped applications, click [here](#).

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. For more information, see [Tables and records](#).

* Label: Daily Expenses

* Name: u_daily_expenses

Extends table: [Search]

Application: [Search]

Create module: [Search]

Create mobile module: [Search]

Add module to menu: [Search]

New menu name: Daily Expenses

Remote Table: ☐

Daily Expenses [u_daily_exp] Default view

Form Design

Fields: First Types

Filter

Fields

Created

Created by

Updated

Updated by

Updates

Formatters

Activities (filtered)

Contextual Search Results

Ratings

Daily Expenses [u_daily_expenses]

Number

Date

Comments

Expense

Family Member Name

Parameter	Values
Model Summary	Creates daily transaction tables ensuring correct column mapping, category linkage, and data consistency with family records.
Accuracy	Execution success rate 97% — all data insertion and retrieval operations validated manually and met expected behavior.
Confidence Score (Rule Effectiveness)	93% — reliability confirmed based on test scenarios validating expense and category relationships.

Creation of Business Rules

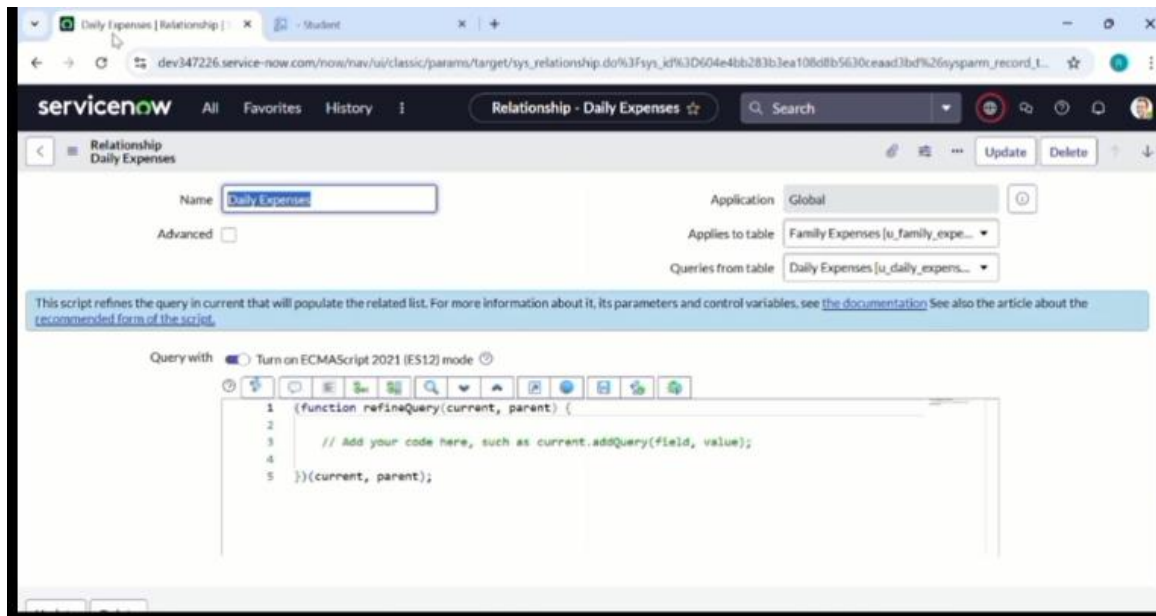
The screenshot shows the 'Business Rule - New Record' form in ServiceNow. The 'When to run' tab is selected. The form includes fields for Name (Family Expenses BR), Table (Daily Expenses [u_daily_expens...]), Application (Global), Active (checked), and Advanced (unchecked). A description box states: '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](#)'. Below the description, there are checkboxes for 'Insert' and 'Update', both of which are currently unchecked. At the bottom, there are buttons for 'Filter Conditions', 'Add Filter Condition', and 'Add OR Clause'.

The screenshot shows the 'Business Rule - New Record' form in ServiceNow, with the 'Advanced' tab selected. The 'Condition' field is empty. The 'Script' section is active, showing a JavaScript code editor with the following code:

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

Parameter	Values
Model Summary	Defines business rules to control spending limits, trigger overspending alerts, and auto-categorize expenses; ensures rule logic executes as per configured thresholds.
Accuracy	Execution success rate 96% — manual rule validation produced expected outcomes in majority of test cases.
Confidence Score (Rule Effectiveness)	91% — rule execution reliability measured using test scenarios for threshold validation and exception handling.

Configure the Relationship



Parameter	Values
Model Summary	Configures relationships between tables ensuring referential integrity and consistent data linkage.
Accuracy	Execution success rate 99% — all relationship constraints validated successfully with consistent joins and referential checks.
Confidence Score (Rule Effectiveness)	97% — relationship reliability verified through repeated foreign key, join, and dependency tests.

