**Internship Report**

A Salesforce Developer Virtual Internship Report Submitted to

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, ANANTAPUR**

In partial fulfillment of the

requirements for the award of

**BACHELOR OF TECHNOLOGY**

**IN**

**Computer Science and Systems Engineering**

*Submitted by*

|  |  |
| --- | --- |
| **Gopalam** | **Saikumar 20121A2917** |



Department of Computer Science and Systems Engineering

**SREE VIDYANIKETHAN ENGINEERING COLLEGE**

(AUTONOMOUS)

(Affiliated to JNTUA, Ananthapuramu, Approved by AICTE, Accredited by NBA & NAAC)

Sree Sainath Nagar, Tirupati – 517 102, A.P., INDIA

2020-2024



**INTERNAL EXAMINER EXTERNAL EXAMINER**

**CERTIFICATE:**



ABSTRACT

Salesforce Developer Catalyst is brought in collaboration with NASSCOM FutureSkills. It is a career building for students, job seekers, and mid-career professionals looking to up-skill and enhance career prospects. We can learn in-demand Salesforce skills, earn credentials and connect to opportunities in the Salesforce ecosystem. Salesforce is one of the world’s first cloud-based CRM software companies. Salesforce solves the problems of lagging innovation and a lack of mobile specialists with a revolutionary approach to app development for the social and mobile-first world. Through salesforce developer catalyst we can Meet the tools and technologies that power development on the Salesforce platform, get introduced to the platform, navigate use cases, and build custom functionality, learn about Salesforce multi-tenancy architecture and create a sample application and also earn two superbadges: apex specialist and process automation specialist. Salesforce delivers breakthrough productivity for all users because it puts the customer—employees, partners, consumers, and devices. Salesforce has introduced artificial intelligence (AI) on its Einstein platform, helping to improve the analytics process. Developers at Salesforce developer training evaluate business processes and develop CRM workflows, and custom solutions for different business needs.It offers quality customer service by displaying customer engagement and problem-solving. Salesforce helps companies monitor consumer experiences through their marketing cloud. This is by offering multichannel marketing campaigns. The salesforce modules allow companies to interact directly with their customers and allow their customers to connect with each other. Salesforce comes with a lot of standard functionality, or out-of-the-box products and features that you can use to run your business.Salesforce provides leads and opportunities to manage sales, cases and communities for customer engagement, customizable Salesforce mobile app, slack, chatter, and communities to connect your company and marketing Cloud to manage your customer journeys. Everything it offer resides in the trusted, multitenant cloud

**TABLE OF CONTENTS**

**CERTIFICATE** i

**ABSTRACT** iv

**TABLE OF CONTENT** v

**LIST OF FIGURES**  vi

**LIST OF TABLES** viii

**INTRODUCTION** 1

[**MODULE1: Building a Data Model for a Travel Approval App**](https://docs.google.com/document/d/1c9vKCfg8zc8JariTH9c2iC6a23qRPMFAEy0mfADNdlc/edit#heading=h.tyjcwt)

* 1. Create a Travel Approval Lightning App 5-7
  2. Create a Department Object 8-12
  3. Create a Travel Approval Object 12-13
  4. Create an Expense Item Object 13-15

1.5 Import Data and Test the App 15-20

**MODULE 2: Formulas and Validation**

2.1 Use Formula Fields 22-34

2.2 Implement Roll-Up Summary Fields 34-37

2.3 Create Validation Rules 38-43

**CONCLUSION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **List Of Figures** | |  | |
| **S.NO** | | **Figure name** | | **Page No** | |
| 1 | | Fig 1: Setup | | 3 | |
| 2 | | Fig 2: New Lightning App | | 4 | |
| 3 | | Fig 3: Navigation Items | | 4 | |
| 4 | | Fig 4: Travel App | | 5 | |
| 5 | | Fig 5: Object Manager | | 6 | |
| 6 | | Fig 6: Custom Object | | 6 | |
| **7** | | Fig 7: Optional Features | | 7 | |
| 8 | | Fig 8: New Custom Object Tab | | 8 | |
| 9 | | Fig 9: Save Tab | | 9 | |
| 10 | | Fig 10: Launch Wizard | | 14 | |
| 11 | | Fig 11: Click Next | | 15 | |
| 12 | | Fig 12: Recently Viewed | | 16 | |
| 13 | | Fig 13: Contact Management | | 16 | |
| 14 | | Fig 14: Travel Approval | | 17 | |
| 15 | | Fig 15: Formula Fields | | 20 | |
| 16 | | Fig 16: Formula Editor | | 22 | |
| 17 | | Fig 17: Sum | | 26 | |
| 18 | | Fig 18: Pyramid Construction | | 29 | |
| 19 | | Fig 19: Opportunities | | 31 | |
| 20 | | Fig 20: Pyramid Emergency Generator-1 | | 32 | |
| 21 | | Fig 21: Pyramid Emergency Generator-2 | | 33 | |
| 22 | | Fig 22: Review the errors on page | | 35 | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | | **List Of Figures** | |  | |
| **S.NO** | | **Figure name** | | **Page No** | |
| 1 | | Table 1: Table containing some details | | 7 | |
| 2 | | Table 2: Table containing some details | | 10 | |
| 3 | | Table 3: Table containing some details | | 12 | |
| 4 | | Table 4: Table containing some details | | 17 | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |  |
|  | |  | |  | |  |
|  | |  | |  | |  |
|  | |  | |  | |  |
|  | |  | |  | |  |

**INTRODUCTION**

Our organization has decided to implement a custom travel approval app. Up until now, you had a travel approval process in place based on emailing spreadsheets. It had no central repository or enforcement of the process, resulting in the inability to report on travel approval activities across the organization. You need to create an application that meets these requirements:

• Each employee must submit an electronic request in the system for future travel.

• Each request includes a list of estimated expenses for airfare, hotel, rental car, and so on.

• Each request must be approved by the employee’s manager, and all out-of-state travel must be approved by a travel coordinator.

• Managers need reports and dashboards to track key travel request trends and KPIs.

• In addition, employees and managers must be able to access their travel requests, approvals, and dashboards via mobile device.

We’ve got a lot of data in our organization. Your users need to access and understand this data at a glance without doing a bunch of calculations in their heads. Enter formula fields, the powerful tool that gives you control of how your data is displayed.

• Let’s say you wanted to take two numeric fields on a record and divide them to create a percentage. Or perhaps you want to turn a field into a clickable hyperlink for easy access to important information from a record’s page layout. Maybe you want to take two dates and calculate the number of days between them. All these things and more are possible using formula fields.

• Let’s look at a specific example. What if you wanted to calculate how many days are left until an opportunity’s close date? You can create a simple formula field that automatically calculates that value. By adding the value to the Opportunity page layout, your users can quickly access this key information. You can also add this field to reports and list views for instant access.

**MODULE-1**

**Build a Data Model for a Travel Approval App**

**1.1 Create a Travel Approval Lightning App:**

For this project, you need to create a new Trailhead Playground. Scroll to the bottom of this page, click the playground name, then click Create Playground. It typically takes 3–4 minutes to create a new Trailhead Playground.

Note: Yes, we really mean a brand-new Trailhead playground! If you use an existing org or playground, you can run into problems completing the challenges.

Open your new Trailhead Playground by clicking Launch at the bottom of this page. Your playground opens in a new browser tab or window. Keep the playground window open while you do this project. After you complete the project steps in your playground, come back to this window and click Verify step at the bottom of this page.

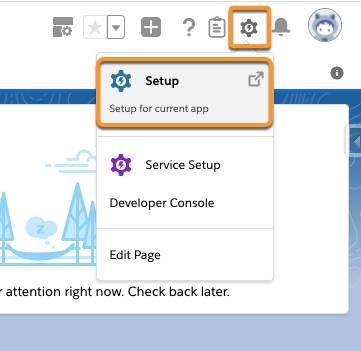
1. In your Trailhead Playground, click Setup and then select **Setup**. 

Fig-1: Setup

2. In the Quick Find box, enter App Manager and select App Manager.

3. Click New Lightning App

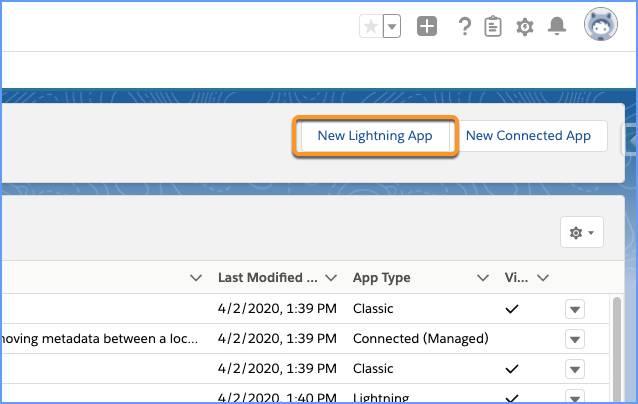


Fig-2: New Lightning App

4. In the App Details & Branding window, enter these details.

* 1. For App Name, enter Travel App.
  2. For the Image, click **Upload** and select **travel.png** from the files you downloaded above.

5. Click **Next**.

6. On theApp Options screen, select **Standard navigation** and click **Next**.

7. On the Utility Items screen, click **Next**

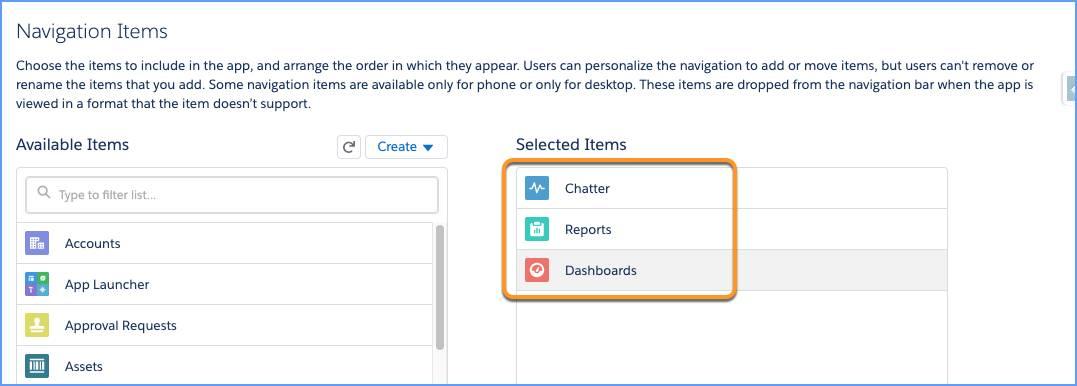
8. On the Navigation Items screen, select **Chatter, Reports,** and **Dashboards** from the Available Items list, and move them to the Selected Items list using the arrow. Then click **Next**. 

Fig-3: Navigation Items

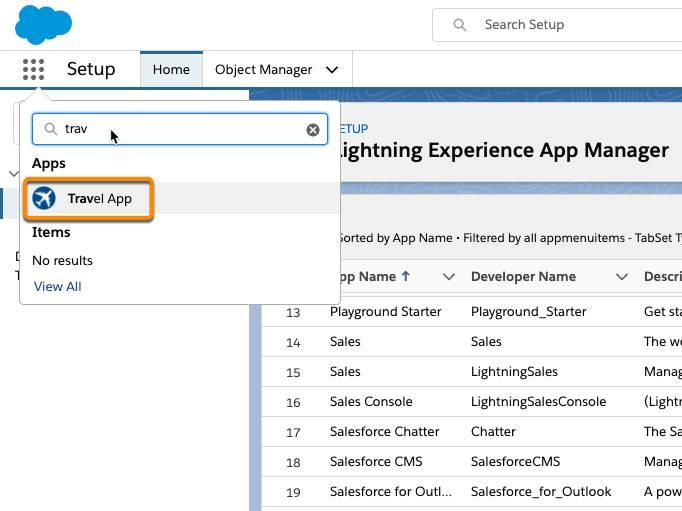
9. On the User Profiles screen, select **System Administrator** and add it to Selected Profiles, then click **Save & Finish**. Now navigate to your new Travel App application to see what it looks like so far. Click  App Launcher, then search for and select **Travel App**.

Fig-4: Travel App

At this point, your window should have three tabs that you defined earlier: Chatter, Reports, and Dashboards.

Now that you’ve created the travel app, and added three tabs to the app, you’re ready to create the data model.

**1.2 Create a Department Object:**

The next step in creating your organization’s travel approval app is to create the data model. The first object to create is the Department object. This object stores information about your departments, such as name and cost center code.

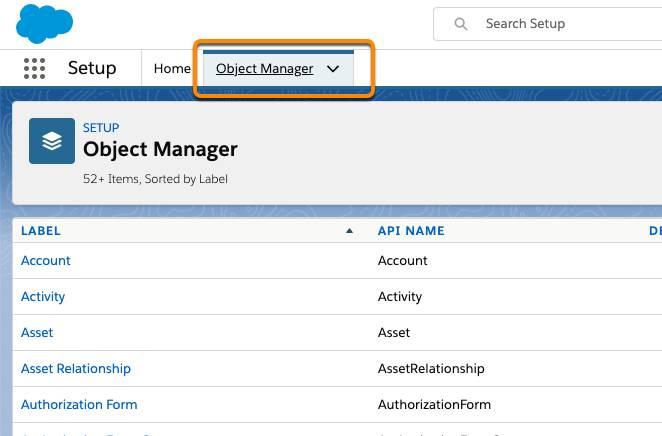
1. From Setup, click **Object Manager**.

Fig-5: Object Manager

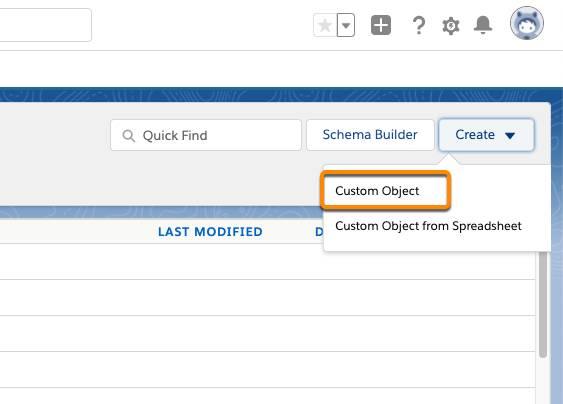
1. Click **Create**,then select **Custom Object**.

Fig-6: Custom Object

1. Enter these details:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Label | Department |
| Plural Label | Departments |
| Object Name | Department (this field autopopulates) |
| Record Name | Department Name |
| Data Type | Text |

Table 1: Table containing some details

1. Now scroll down and select these options:

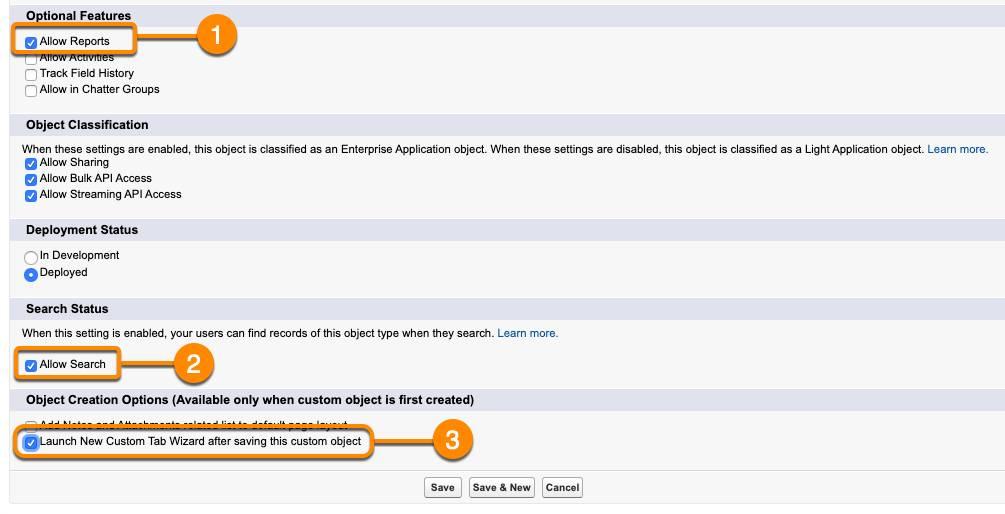
* Allow Reports (1)
* Allow Search (2)
* Launch New Custom Tab Wizard after saving this custom object (3) 

Fig-7: Optional Features

5. Click **Save**.

6. Now, define your tab settings. Click the Lookup iconnext to the Tab Style field and select **Books**.



Fig-8: New Custom Object Tab

7. Click **Next**. Leave the next screen as is and click **Next**.

8. Next, choose which applications to associate with this tab.

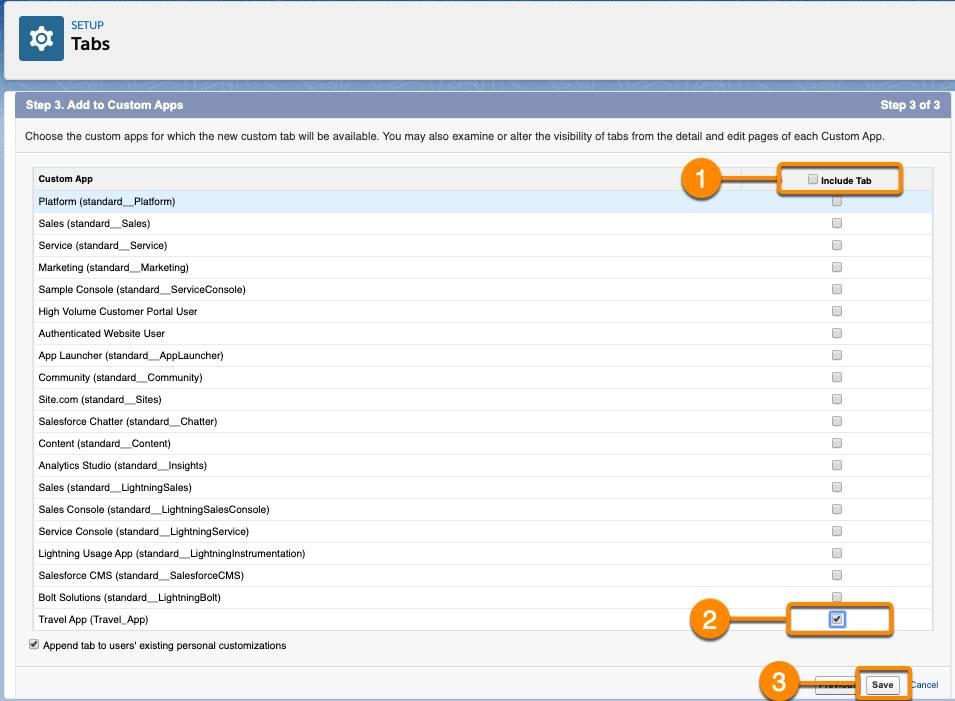
* Click the **Include Tab** (1) option at the top of the column to deselect all apps.
* Select only the **Travel App** (2) app from the list.
* Click **Save** 

Fig-9: Save Tab

9. On the next screen, click **Fields & Relationships**, then click **New**.

10. Select **Text** as the data type, then click **Next**.

11. Enter these details.

* For Field Label, enter Department Code
* For Length, enter 10
* Select **Required**
* Select **Unique**, then select **case-sensitive**

12. Click **Next**.

13. Leave the Establish field-level security screen as is, and click **Next**.

14. Leave the Add to page layouts screen as is, and click **Save**.

Now that you’ve created the Department Object, you’re ready to move on to the next step, creating the Travel Approval object. The Travel Approval object stores basic information about where employees are traveling and trip duration.

**1.3 Create a Travel Approval Object:**

In the previous steps, you created the travel app using Lightning App Builder and created the first custom object for Departments. Now you create the object to store the Travel Approval requests.

1. Navigate back to **Object Manager**.
2. Click **Create** then select **Custom Object**.
3. Enter these details:

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Label | Travel Approval |
| Plural Label | Travel Approvals |
| Object Name | Travel\_Approval (this field auto-populates) |
| Record Name | Travel Approval # |
| Data Type | **Auto Number** |
| Display Format | TA-{00000} |
| Starting Number | 1 |

Table 2: Table containing some details

4. Click **Save**.

5. Click Lookup icon next to Tab Style and select **Airplane**.

6. Click **Next**.

7. Keep the Add to profiles screen as is, and click **Next**.

8. In the Add to Custom Apps screen, select the **Include Tab** checkbox to deselect all apps. Select **Travel App**,then click **Save**.

9. On the object configuration page, click **Fields & Relationships**, then click **New**.

10. Select **Text Area** as the data type, then click **Next**.

11. For Field Label, enter Purpose of Trip.

12. Click **Next**.

13. Leave the Establish field-level security screen as is, and click **Next**.

14. Keep the Add to page layouts screen as is, and click **Save**.  
 To create the remaining fields for the Travel Approval object, repeat step 9 to begin, then follow the parameters below.

15. Create the **Status** field with a **Picklist** data type. Click **Enter values, with each value separated by a new line**. Use the following values:

* 1. New
  2. Submitted
  3. Pending Approval
  4. Approved
  5. Rejected
  6. Draft

16. Create the **Trip Start Date** field with a **Date** data type.

17. Create the **Trip End** **Date** field with a **Date** data type.

18. Create the **Out-of-State** field with a **Checkbox** data type.

19. Create the **Destination State** field with a **Text** data type. Set the text **Length** to 2.

20. Create the **Department** field with a **Lookup Relationship** data type, then select **Department** for the Related To field. Click **Next** 4 times, then **Save**.

You’ve just created the Travel Approval object and created a lookup relationship to the Department object. In the next step, you create the Expense Item object, where each expense line item for the travel request is created.

**1.4** **Create an Expense Item Object:**

In the previous steps, you created the travel app and custom objects for Department and Travel Approval. The final object you create is the Expense Item object. This allows the app to provide estimates for trip costs such as airfare, hotel, or rental car.

1. Navigate back to **Object Manager**.
2. Click **Create** then select **Custom Object**.
3. Enter these details:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Label | Expense Item |
| Plural Label | Expense Items |
| Starts with vowel sound | *Check/select this option* |
| Object Name | Expense\_Item (this field autopopulates) |
| Record Name | Expense Item Number |
| Data Type | Auto Number |
| Display Format | E-{00000} |
| Starting Number | 1 |

Table 3: Table containing some details

1. Select **Allow Reports**.  
     
   **Note:** Don’t select Launch new Custom Tab Wizard after saving this custom object. You don’t need a separate tab for just **Expense Items**. Users will use the **Travel Approval** tab to navigate to specific records and then drill down to expense items from the select travel approval.
2. Click **Save**.

Next, you’re directed to the standard object configuration page for the Expense Item object, where you create its fields Start with the Amount field.

1. Click **Fields & Relationships**, and click **New**.
2. For data type, select **Currency**.
3. Enter these details.
   * For Field Label, enter Amount
   * For Length, enter 16
   * For Decimal places, enter 2
   * Select **Required**
4. Click **Next, Next**, then **Save & New**.

Next, create the **Expense Type** field.

1. Select **Picklist** as the data type.
2. Select **Enter values, with each value separated by a new line.**
3. Add these values:
   * Airfare
   * Hotel
   * Rental car
   * Meals
   * Other
4. Select **Required**.
5. Click **Next**, **Next,** then **Save & New**.

Create the **Travel Approval** field.

1. Select **Master-Detail Relationship** data type, click **Next**.
2. Select **Travel Approval** from the Related To menu.
3. Click **Next** four times, then click **Save**.

Now, see how the travel approval app looks. Click App Launcher icon to open the App Launcher, then search for and select **Travel App** with the two new tabs you created.

**1.5 Import Data and Test the App:**

You now have a fully functional application. You built the data model, and in parallel behind the scenes, the user interface was created for you. And a full SOAP and REST API layer was automatically created to integrate your Travel Approval objects with objects in other systems. In this final step, you import some data and test the app.

Before you test your application, first load some sample data into the Departments object.

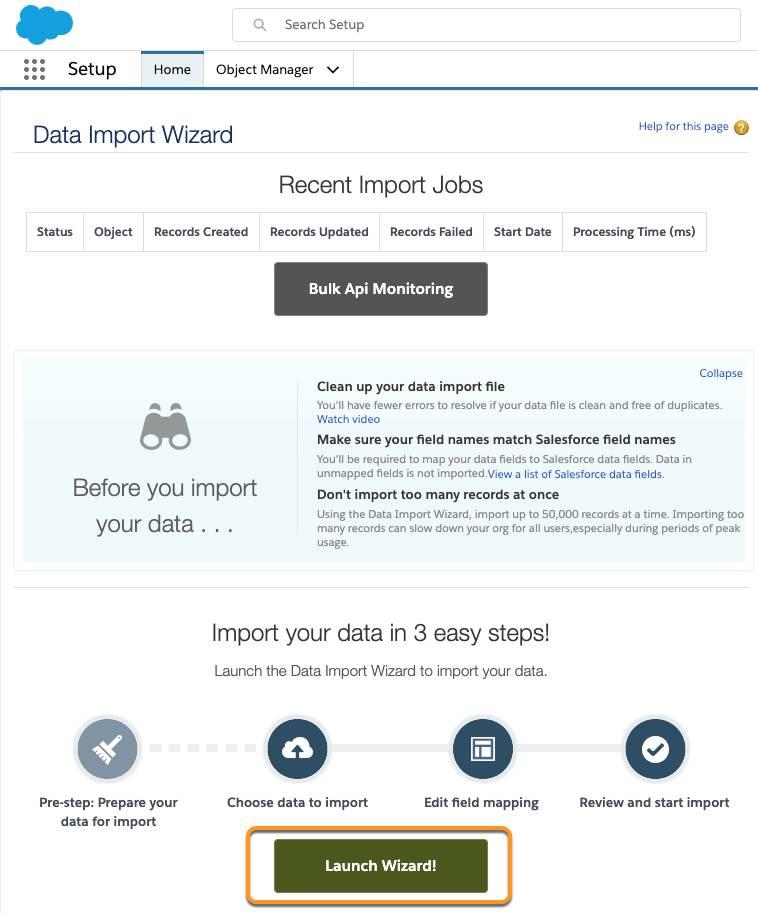
1. From Setup, click the **Home** tab.
2. In the Quick Find box, enter Data Import and select **Data Import Wizard**.
3. Click **Launch Wizard!** 

Fig-10: Launch Wizard

1. Click the **Custom Objects** tab and select the **Departments** object.
2. Next, select **Add new records**.
3. Drag and drop the **Departments.csv** file you downloaded in Step 1 - Create a Travel Approval Lightning App to the Drag CSV file here to upload section, or click the CSV icon and browse to select your file. Select **Next**.

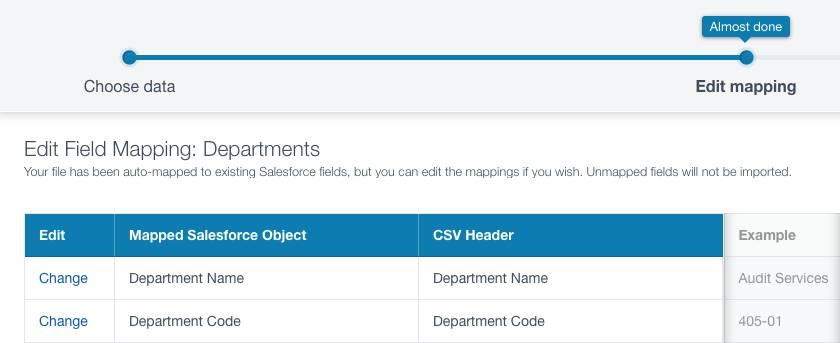
1. Since the field names in the CSV file (CSV Header) are the same as the field names in your object (Mapped Salesforce Object), the fields are automatically mapped. Click **Next**. 

Fig-11: Click Next

8. The next screen gives you a summary of your data import. Click **Start Import**.

9. Click **OK** on the popup.

This takes you to the bulk import summary window that shows that the process has completed, and 16 records have been successfully imported or processed. You’ll also get an email notification confirming the import.

**Testing the App:**

Now that you’ve imported department data, you can test out the app.

1. Click App Launcher icon to open the App Launcher and navigate back to your Travel Approval app.

2. Click the **Departments** tab.

3. Click the **Recently Viewed** (1), and select **All** (2).

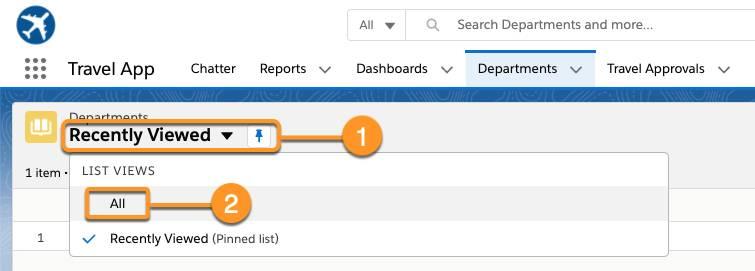


Fig-12: Recently Viewed

1. Click any one of the departments to drill down into the Department details. For example, click **Contract Management** and check out the department details including the Department Code field you created, along with some default fields.

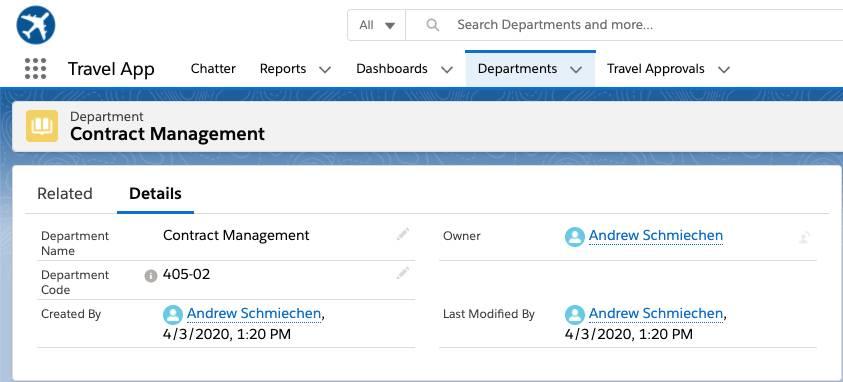


Fig-13: Contact Management

Now, create a sample travel approval and associate a few expense estimates to it.

1. Click the **Travel Approvals** tab and click **New**.

2. Enter these details.

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Purpose of Trip | Attend Dreamforce |
| Status | **Draft** |
| Trip Start Date | *(pick any date you want)* |
| Trip End Date | *(pick any date you want)* |
| Out-of-State | *Check/select this option* |
| Destination State | CA |
| Department | Technology |

Table 4: Table containing some details

3. Click **Save**. You now have a travel approval record.

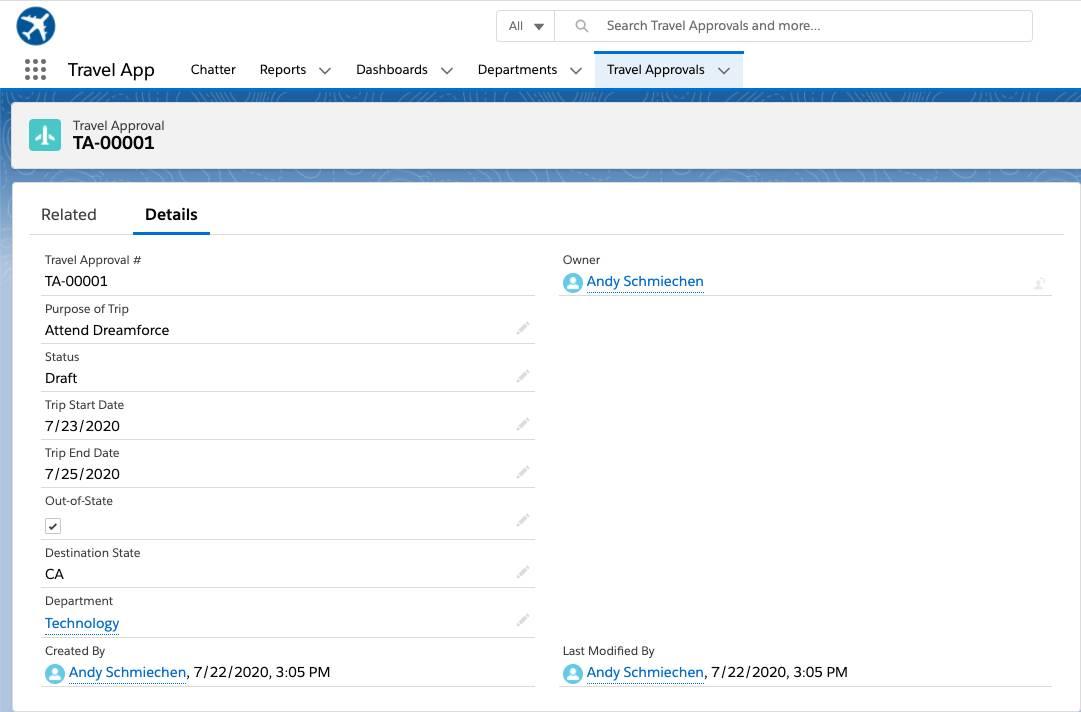


Fig-14: Travel Approvals

Finally, add expense estimates for airfare and hotel to the travel approval. Click the **Related** tab.

1. Click **New**, then enter these details:
   * Amount: 450
   * Expense Type: **Airfare**
2. Click **Save**.
3. Click **New** then enter these details.
   * Amount: 870
   * Expense Type: **Hotel**
4. Click **Save**.

We’ve successfully created a travel app for an organization.

**MODULE-2**

**Formulas and Validation**

**2.1 Use Formula Fields:**

**Introduction to Formula Fields**

You’ve got a lot of data in your organization. Your users need to access and understand this data at a glance without doing a bunch of calculations in their heads. Enter formula fields, the powerful tool that gives you control of how your data is displayed.

Let’s say you wanted to take two numeric fields on a record and divide them to create a percentage. Or perhaps you want to turn a field into a clickable hyperlink for easy access to important information from a record’s page layout. Maybe you want to take two dates and calculate the number of days between them. All these things and more are possible using formula fields.

Let’s look at a specific example. What if you wanted to calculate how many days are left until an opportunity’s close date? You can create a simple formula field that automatically calculates that value. By adding the value to the Opportunity page layout, your users can quickly access this key information. You can also add this field to reports and list views for instant access.

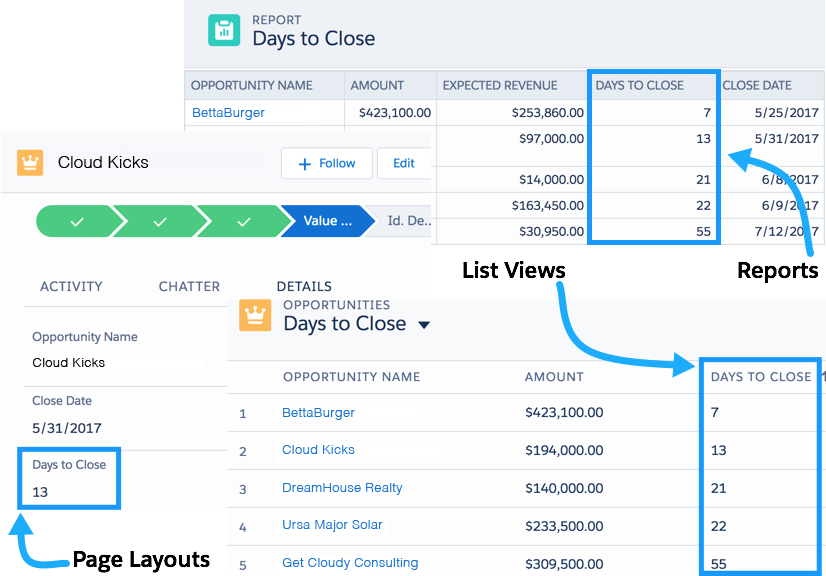


Fig-15: Formula Fields

When you’re first learning formulas, it’s best to start with simple calculations and build up to more complex scenarios. But even simple formulas can provide valuable information.

In this unit, we take you through the basics of using the formula editor and introduce you to formula syntax through several basic examples. We also touch on troubleshooting problems with your formula fields. Now let’s have some fun!

**Ready to Get Hands-on with Formulas?**

Launch your Trailhead Playground now to follow along and try out the steps in this module. To open your Trailhead Playground, scroll down to the hands-on challenge and click **Launch**. You also use the playground when it’s time to complete the hands-on challenges.

**Find the Formula Editor**

Before we dive into writing formulas, let’s locate the formula editor and get to know its features.

You can create custom formula fields on any standard or custom object. To start, we’ll create a formula on the Opportunity object. Follow these steps to navigate to the formula editor.

1. From Setup, open the Object Manager and click **Opportunity**.
2. In the left sidebar, click **Fields & Relationships**.
3. Click **New**.
4. Select **Formula** and click **Next**.
5. In **Field Label**, type My Formula Field. Notice that **Field Name** populates automatically.
6. Select the type of data you expect your formula to return. For example, if you want to write a formula that calculates the commission a salesperson receives on a sale, you select Currency. For now, pick **Text**.
7. Click **Next**. You’ve arrived at the formula editor! Time for our tour.

**Use the Formula Editor**

This image highlights the most important parts of the formula editor.

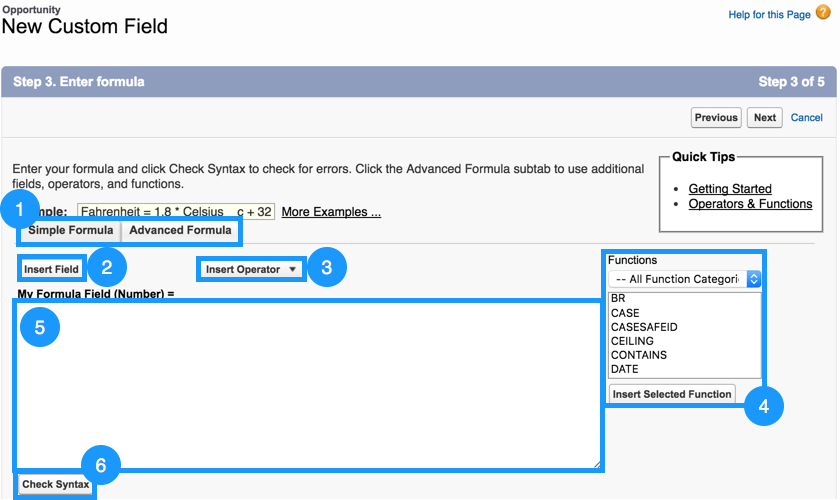


Fig-16: Formula Editor

1. The formula editor comes in two flavors: Simple and Advanced. It’s tempting to use the Simple editor, but we always recommend using the Advanced editor. Advanced doesn’t mean more complicated. It means more tools for you to create powerful formulas.
2. The **Insert Field** button opens a menu that allows you to select fields to use in your formula. Inserting from this menu automatically generates the correct syntax for accessing fields.
3. The **Insert Operator** button opens a dropdown list of the available mathematical and logical operators.
4. The Functions menu is where you view and insert formula functions. Functions are more complicated operations that are preimplemented by Salesforce. Some functions can be used as-is (for example, the TODAY() function returns the current date), while others require extra pieces of information, called parameters. The LEN(text) function, for instance, finds the length of the text you input as a parameter.The formula LEN("Hello") returns a value of 5.
5. The text area is where you enter your formula. When writing formulas, keep in mind that:
   * Whitespace doesn’t matter. You can insert as many spaces and line breaks as you want without affecting the formula’s execution.
   * Formulas are case sensitive. Pay attention to capitalization of field and object names.
   * When working with numbers, the standard order of operations applies.
6. Once you’ve written a formula, you can use the **Check Syntax** button to ensure that everything is in working order before saving. If your formula has issues, the syntax checker alerts you to specific problems.

We don’t need to continue creating this formula field, so click **Cancel**. Now that you know your way around, let’s put the editor to use with some simple examples.

**Example 1: Display an Account Field on the Contact Detail Page**

Record detail pages contain a ton of information, but sometimes it’s not enough. Sometimes you need more! For your first formula, let’s do something simple. Let’s take a single field from an Account and show it on a Contact using what’s called a *cross-object formula*. Let’s take a look.

First create a Contact. If you’ve never created a Contact before, click the **App Launcher** App Launcherand select **Contacts**. Then go to the Contacts tab and click **New**. Enter any value for Last Name. For Account Name, enter an existing account such as United Oil & Gas Corp. Click **Save**. Next we create a formula to display the account number on the Contact page.

1. From Setup, open the Object Manager and click **Contact**.
2. In the left sidebar click **Fields & Relationships**.
3. Click **New**.
4. For the field type, select **Formula** and click **Next**.
5. Call your field Account Number and select **Text** for the formula return type. Click **Next**.
6. In the Advanced Formula Editor, click **Insert Field**. Select **Contact** | **Account** | **Account Number** and then click **Insert**. Click **Check Syntax**. If there are no syntax errors, click **Next**. It’s unlikely that you’ll find a syntax error in a simple formula like this one, but it's a good idea to get in the habit of checking syntax for every formula.
7. Click **Next** to accept the field-level security settings, then click **Save.**

Congratulations, you’ve written your first formula! Now it’s time to see what you’ve done. Open the detail page for the Contact object you just created and find your new Account Number formula field. Cool!

**Example 2: Display the Number of Days Until an Opportunity Closes on a Report**

You can also use formula fields in reports to increase the visibility of important information. Say, for example, you wanted a report column that displays the number of days until an opportunity is closed. First, create an Opportunity to test our formula.

If you’ve never created an Opportunity before, click App Launcher and select **Opportunities**. go to the Opportunities tab and click **New**. Fill in any value for the Opportunity Name, select any Stage, and set a close date that’s at least 3 days in the future. Click **Save**.

Then take these steps to create a custom formula field called Days to Close on the Opportunities object with a Number return type.

1. From Setup, open the Object Manager and click **Opportunity**.
2. In the left sidebar click **Fields & Relationships**.
3. Click **New**.
4. Select **Formula** and then click **Next**.
5. In the Field Label text area, type Days to Close.
6. Select the **Number** radio button.
7. Click **Next** to open the formula editor.
8. To find the difference between the opportunity close date and today’s date subtract one from the other.
   * Click **Insert Field** and select **Opportunity | Close Date** and click **Insert**.
   * From the **Insert Operator** menu, select **- Subtract**.
9. But how do we tell our formula that we need today’s date? Luckily, there’s a function called TODAY() that updates to match the current date.
   * In the **Functions** menu on the right side of the editor, select **TODAY**.
   * Click **Insert Selected Function**.
10. Click **Check Syntax**. If there are no syntax errors, click **Next**.
11. Click **Next** to accept the field-level security settings, then click **Save.**

Now it's time to put your new formula field in a report.

1. From App Launcher, open the **Reports** tab and click **New Report**.
2. Enter **Opportunities** in the Search Report Types... field. Select **Opportunities** and click **Continue**. Your opportunity appears in the Report Preview panel.
3. Make sure **Update Preview Automatically** is enabled.
4. In the Add column... field on the left side of the page, enter **Days to Close**.  This field is the formula field you just created. A column with the field containing the calculated value is automatically added to the report.

You won't need the report again for this unit. You can discard it and move on to the next example.

**Example 3: Find Distinct Objects Using the Power of One**

Organizations often want to count the number of unique objects in a report with hundreds of records. Say, for example, you have a hundred opportunities listed in a report, but only a handful of users own all these opportunities. How do you find the number of distinct users? This task sounds difficult, but it’s one of the easiest formulas you can write. It’s called the Power of One.

To write this formula, create a custom formula field on the User object. Name it Unique Users, give it a **Number** return type, and select **0** from the Decimal Places dropdown list. Click **Next** to open the formula editor. For this formula, you don’t need to insert any fields, operators, or functions. Instead, enter the number **1**.

Yes, that’s really it! Save your formula as usual, and then click the **Reports** tab to see it in action. Create an Opportunities report and make sure that **Update Preview Automatically** is enabled. In the Add column... field on the left side of the page, enter Unique Users to add a column with the new field to the report. You should see something similar to the following (your report may appear different, depending on what columns you have displayed).



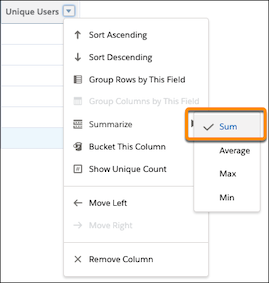
Here, we have five opportunities between two distinct users. Where does our formula come in? On the Unique Users column, click the dropdown menu and select **Summarize** and then **Sum**.

Fig-17: Sum

The number of distinct users appears at the bottom of the column. As you can see, even simple formulas provide powerful insights into your organization.

**Debug Formulas**

Syntax errors are an inevitable part of working with formulas. The **Check Syntax** button in the editor is an important tool for debugging your formulas. The syntax checker tells you what error it encountered and where it’s located in your formula. Here are some common syntax issues.

1. **Missing parentheses:** This error most often occurs when the number of opening parentheses doesn’t match the number of closing parentheses. It can be particularly difficult to avoid this error if you’re using several functions at once. Try breaking your function into multiple lines so it’s easier to tell which sets of parentheses belong together.

You’ll also see this error if you forget a comma between two function parameters. This error is confusing because the actual problem doesn’t match up with the syntax checker. If you’re certain your parentheses are correct, double check that the commas in your function are correct as well.

1. **Incorrect parameter type:** If you give a function a number parameter when it expects text (or any other combination of data types), this is the error you see. Always check the help text or the documentation, so you know what kind of parameters a function accepts.
2. **Incorrect number of parameters for function:** If you input too many or too few parameters into a function, the syntax checker alerts you. Again, check the help text or documentation for guidelines on inputting parameters to specific functions.

**Formula result is incompatible with formula return type:** You see this error if you select one data type when creating the formula field, but write a formula that returns a different data type. In the example below, you can see that My Account Formula expects to return a number (shown in parentheses next to the formula name), but the TODAY() function returns a date. The error tells you what the expected data type is, but you can always reference the documentation beforehand to avoid the error.

1. **Field does not exist:** This error indicates that you’ve included a field in your formula that your object doesn’t support. In this case, check your spelling and capitalization. If you can’t find any mistakes, try inserting the field from the Insert Field menu again to make sure you’re referencing it correctly.

Another reason you see this error is if you forget to put quotation marks around a text literal or a hyperlink.

1. **Unknown function**: In this case, check that Salesforce supports the functions you’re using. You also get this error for misspelled functions.

**Further Examples**

Let’s look at a few more examples. You can create these formulas yourself or simply read through.

* 1. This formula creates a hyperlink to an external website using the HYPERLINK() function. Adding hyperlinks to page layouts helps your users access important information quickly from the detail pages.
  2. If you want to apply a discount to an opportunity amount, you can use the following formula. In this case, we’re applying a 12% discount and then rounding the result to two decimal places using the ROUND() function.
  3. This formula is a checkbox formula that determines whether a particular opportunity is a “big” opportunity. It checks whether the number of employees at the opportunity account’s associated company is greater than 1,000 AND whether the opportunity amount is greater than $10,000. If both statements are true, the field appears as a checked box on the Opportunity page layout. Otherwise, it appears as a blank box.

The formulas documentation contains numerous examples for many different use cases. While you’re browsing these examples, keep in mind that many of them contain advanced concepts that weren’t covered in this unit. Make sure you’re comfortable with the information presented here before tackling these formulas

**2.2 Implement Roll-Up Summary Fields:**

**Introduction to Roll-Up Summary Fields**

While formula fields calculate values using fields within a single record, roll-up summary fields calculate values from a set of related records, such as those in a related list. You can create roll-up summary fields that automatically display a value on a master record based on the values of records in a detail record. These detail records must be directly related to the master through a master-detail relationship.

You can perform different types of calculations with roll-up summary fields. You can count the number of detail records related to a master record, or calculate the sum, minimum value, or maximum value of a field in the detail records. For example, you might want:

1. A custom account field that calculates the total of all related pending opportunities.
2. A custom order field that sums the unit prices of products that contain a description you specify.

**Defining a Roll-Up Summary Field**

Since roll-up summary fields are based on master-detail relationships, it’s useful to review object relationships before creating a roll-up summary field.

**Master-Detail Relationships**

Master-detail relationships closely link objects together so that the master record controls specific behaviors of the detail and subdetail record.

You define a roll-up summary field on the object that is on the master side of a master-detail relationship. For example, you can create a roll-up summary field on the Account object, summarizing related opportunities:

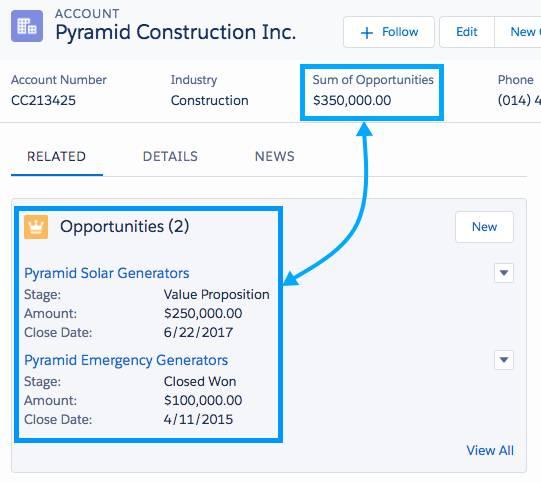


Fig-18: Pyramid Construction

There are a few different types of summaries you can use.

|  |  |
| --- | --- |
| **Type** | **Description** |
| COUNT | Totals the number of related records. |
| SUM | Totals the values in the field you select in the Field to Aggregate option. Only number, currency, and percent fields are available. |
| MIN | Displays the lowest value of the field you select in the Field to Aggregate option for all directly related records. Only number, currency, percent, date, and date/time fields are available. |
| MAX | Displays the highest value of the field you select in the Field to Aggregate option for all directly related records. Only number, currency, percent, date, and date/time fields are available. |

**Creating the Summary Field**

1. From Setup, open Object Manager and click **Account**.
2. On the left sidebar, click **Fields & Relationships**.
3. Click **New**.
4. Choose the Roll-Up Summary field type, and click **Next**.
5. For Field Label, enter Sum of Opportunities and click **Next**.
6. The Summarized Object is the detail object that you want to summarize. Choose Opportunities.
7. Choose the SUM summary type and choose Amount as the Field to Aggregate.
8. Click **Next**, **Next**, and **Save**.

**Examples of Roll-Up Summary Fields**

Here are more examples of detail data rolling up to master records.

**Date Opportunity First Created**

A roll-up field was created on the Accounts object. The MIN of all Created Date fields on the Opportunities object displays the earliest date an opportunity was created related to an account.

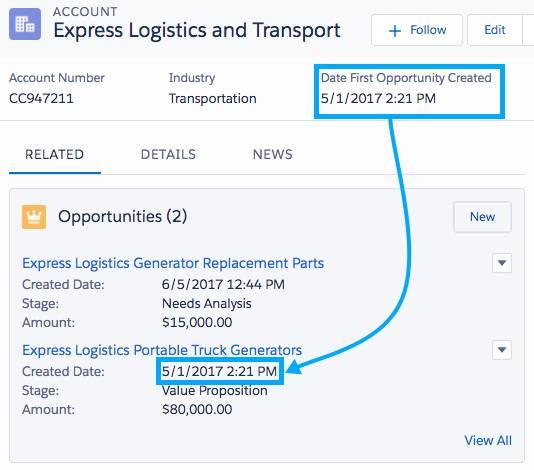


Fig-19: Opportunities

**Total Price of All Products Related to an Opportunity**

A roll-up field was created on the Opportunities object. Total Price is summarized on the Opportunity Product object to find the grand total of all products related to an opportunity.

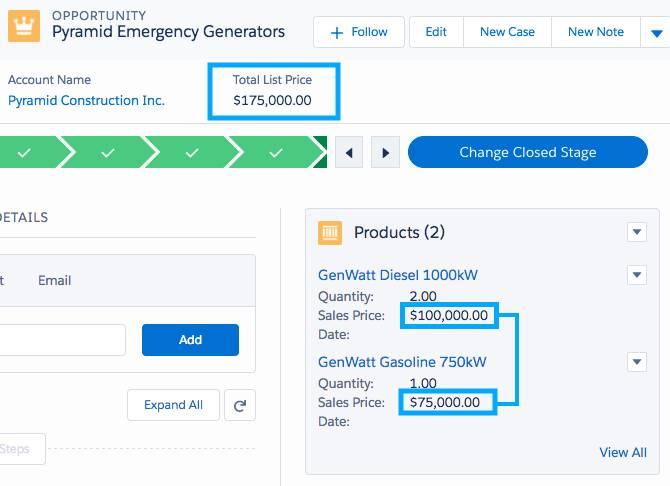


Fig-20: Pyramid Emergency Generators-1

**Minimum List Price of An Opportunity**

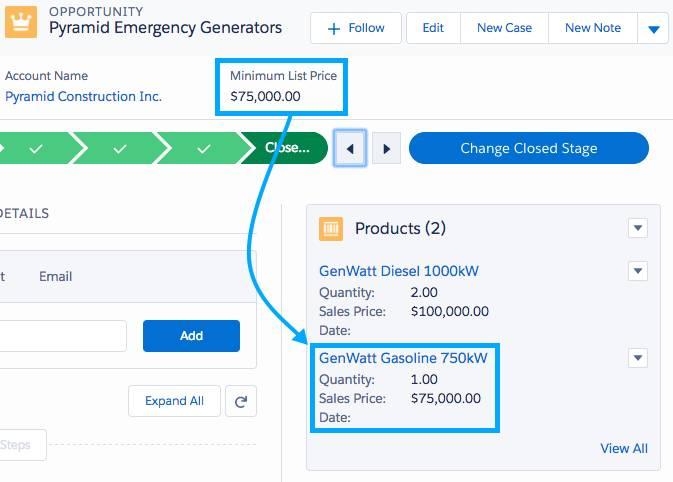
A roll-up field was created on the Opportunities object. List Price is summarized on the Opportunity Product object to find the product with the lowest price related to an opportunity.

Fig-21: Pyramid Emergency Generators-2

**2.3 Create Validation Rules:**

**Introduction to Validation Rules**

Validation rules verify that data entered by users in records meets the standards you specify before they can save it. A validation rule can contain a formula or expression that evaluates the data in one or more fields and returns a value of “True” or “False.” When the validation rule returns a value of "True", this confirms that the data entered by the user contains an invalid value. Validation rules can also include error messages to display to users when they enter invalid values based on specified criteria. Using these rules effectively contributes to quality data. For example, you can ensure that all phone number fields contain a specified format or that discounts applied to certain products never exceed a defined percentage.

**Defining Validation Rules**

You can create validation rules for objects, fields, campaign members, or case milestones. In these steps, we create a validation rule that fires when a user tries to save an account with an account number of incorrect length.

**Creating a Validation Rule**

1. From Setup, go to Object Manager and click **Account**.
2. In the left sidebar, click **Validation Rules**.
3. Click **New**.
4. Enter the following properties for your validation rule:  
     a.RuleName:Account\_Number\_8\_Characters  
     b. Error Condition Formula: LEN( AccountNumber) != 8
5. Error Message: Account number must be 8 characters long.
6. To check your formula for errors, click **Check Syntax**.
7. Click **Save** to finish.

Here’s how a validation rule’s error message can appear when a user types an incorrect account number format into a field.

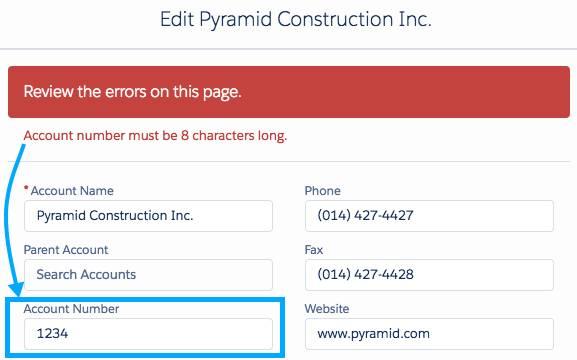


Fig-22: Review the errors on this page

**Examples of Validation Rules**

Here are some validation rule examples that you can try out yourself.

**Account Number Is Numeric**

The AND function returns a value of "True" if all values in the formula are true, and a value of "False" if one or more values are false. The ISBLANK function determines if an expression has a value. The ISNUMBER function determines if an expression's value is a number. The NOT function determines if the inverse of an expression is true. In the example, the validation rule determines if an account number is both not blank and not a number. A value of "True" indicates that the data entered by the user contains an invalid value. That is, if the user enters a non-numeric value for an account number, the validation rule returns a response of "True" and sends an error message.

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Description:** | Validates that the Account Number is numeric if not blank. |
| **Formula:** | AND(     NOT(ISBLANK(AccountNumber)),     NOT(ISNUMBER(AccountNumber))  ) |
| **Error Message:** | Account Number is not numeric. |
| **Error Location:** | Account Number |

**Date Must Be in the Current Year**

The YEAR function returns the four-digit year of a given date. The TODAY function returns the current date. The <> (Not Equal) operator determines if a value is not equal to another value (if it is either less than or greater than the other value.) In the example, the validation rule determines if the year of a given date is not equal to the year of today’s date. A value of "True" indicates that the data entered by the user contains an invalid value. That is, if the user enters a date that is not in the current year, the validation rule returns a response of "True" and sends an error message.  
**Number Range Validation**

In the example, the validation rule determines if the difference between two values (Salary Max and Salary Min) is greater than $20,000. A value of "True" indicates that the data entered by the user contains an invalid value. That is, if the user enters two values whose difference exceeds the $20,000 salary range, the validation rule returns a response of "True" and sends an error message.

**Website Extension**

The AND function returns a value of "True" if all values in the formula are true, and a value of "False" if one or more values are false. The <> (Not Equal) operator determines if a value is not equal (is either less than or greater than) another value. In the example, if the user enters a website URL with an extension that is not equal to (is either greater than or less than)  all six of the valid extensions, the validation rule returns a response of "True" and sends an error message. If the user enters a website URL with an extension that is identical to (is not greater than or less than) one of the valid extensions, the validation rule returns a response of "False" and does not send an error message, because the data the user entered is valid.

**Valid Billing Country**

The OR function returns a "True" response if one or more expressions in the formula are true, and returns a "False" response if all expressions are false. The LEN function returns the number of characters in a specified text string. In the example, the validation rule determines if the value that the user entered for a Billing Country code is either one character (instead of the required two), or does not contain one of the valid two-character codes. If either of these conditions is true, the validation rule returns a value of "True" and sends an error message. If the user enters a valid Billing Country code, both expressions in the formula are false: the LEN is not 1, and the data does contain one of the valid values. In this case, the validation rule returns a value of "False" and does not send an error message.

**Conclusion**

In Salesforce Developer Catalyst by customizing page layouts and creating cross-object formulas, validation rules, roll-up summaries, and formulas, you're giving the team more bang for their buck with their Travel Approval app. With consistent, complete data they can easily utilize and find at a glance, they can easily manage all the data.

By creating picklists, lookup filters, formula fields, and record types; set up page layouts and field history tracking; and established validation rules. Using these tools, you’ve enabled the ops, sales, and service teams to do their jobs better and more efficiently by customizing how they work with the important data they need day to day.

In Lighting Platform Customized page layouts and enabled feed tracking on Chatter. With all this hard work, we have made a Approval App. We’ve also made Travel Approval App much more efficient. Now we have a more user-friendly app and much easier access to the information we need to do this jobs well. Approval process automates help the users view open approval requests by adding the Items to Approve component to their Home page.

We’ve successfully created a travel app for your organization. To build the app, we created custom objects for Departments, Travel Approvals, and Expense Items, and you imported department data. Finally, we’ve tested the app by creating the first Travel Approval record. In the next project, we learn how to customize the user interface, work with list views, and enable Chatter.