



360 SmartDocs

Word Add-In — Complete User Guide

PRODUCT GUIDE • VERSION 1.0

Table of Contents

1. Introduction & Overview
2. Getting Started
 1. Signing In
3. Template Listing Page
4. Building a Template — The CREATE Tab
 1. Selecting a Primary Object
 2. Selecting an Action
5. Action: Field
 1. Understanding Field Types
 2. Drilling into Reference Fields
 3. Global Variables (Dates, Running User, Running Org)
 4. Step-by-Step: Inserting a Field
 5. Field Formatting Options
6. Action: Conditions
 1. If Block vs If/Else Statement
 2. Nested Conditions
 3. Operators & Multiple Conditions
 4. Step-by-Step: Building a Condition
 5. Real-Life Examples
 6. Custom Conditions (Direct Editing)
7. Action: View
 1. Three Layout Types
 2. Step-by-Step: Creating a View

3. Real-Life Examples
4. Summarize / Aggregate (Advanced)
5. Calc — Expression Evaluator (Advanced)
6. Filtering Loop Results with `filter`
7. Tables — Advanced Layouts & Helpers
 - Vertical Tables with `:vt`
 - Serial Numbers with `iterator`
 - Splitting Wide Tables with `splitTable`
 - Closing Tag Forms — Table, Loop & Grid
8. Action: Image
 1. Looping over Files (ContentDocument) for Images
9. Action: QR Code
10. Action: Barcode
11. Copy vs. Insert
12. Preview Tab — Test Your Template
13. Live Tab — Publish Your Template
14. Tag Syntax Quick Reference
15. Tips & Best Practices
16. Troubleshooting

1. Introduction & Overview

The **360 SmartDocs Word Add-In** is a Microsoft Word sidebar that connects directly to your Salesforce organization. It allows you to build smart document templates — Word files that automatically populate with real Salesforce data when a document is generated.

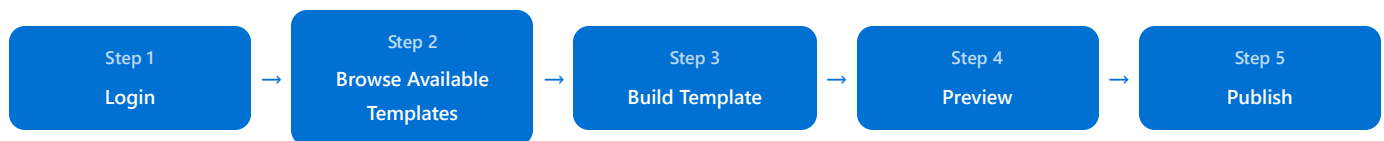
Instead of manually updating every quote, contract, or proposal, you place **merge tags** (placeholders like `{Account.Name}` or `{Amount}`) into your Word document. When a user generates a document from Salesforce, those tags are automatically replaced with live data from the selected record.

Who this guide is for: This guide is written for anyone who wants to create or edit document templates — no technical knowledge required. Every step is explained from scratch.

Related Documentation:

- **360 SmartDocs Installation & Setup Guide** — For administrators: installing the package, configuring OAuth, assigning permissions, and initial setup.
- **360 SmartDocs Feature & User Guide** — For all users: generating documents from Salesforce records, bulk generation, template management, and more.

How It Works



Full Microsoft Word Features Supported

Your templates are real **.docx** Word files. You have complete freedom to use all standard Word features:

- Headers and footers with page numbers
- Tables with custom borders, shading, and merged cells
- Fonts, colors, bold, italic, underline, strikethrough
- Bullet lists, numbered lists, multi-level lists
- Images, shapes, text boxes, and SmartArt
- Page breaks, section breaks, columns
- Styles, themes, and document templates
- Mail-merge-style layouts

The add-in simply helps you insert dynamic Salesforce placeholders (merge tags). Everything else in the document behaves exactly like a normal Word file.

Important: Document templates must be created and edited in **Microsoft Word using this Add-In**.

You cannot create or edit templates with merge fields directly within Salesforce. The Salesforce template record stores metadata (name, object, document type), but the actual template file (`.docx`) must be designed here in Word using the Add-In. Attempting to create templates without the Add-In may result in incorrect template structure, missing merge fields, or unusable documents.

Manual Tag Entry: You do not have to use the add-in for every tag. If you know the field name and the tag format, you can type merge tags directly into your Word document without using the add-in UI at all. The add-in is a helper tool — not a requirement. See [Section 12 — Tag Syntax Quick Reference](#) for all supported formats.

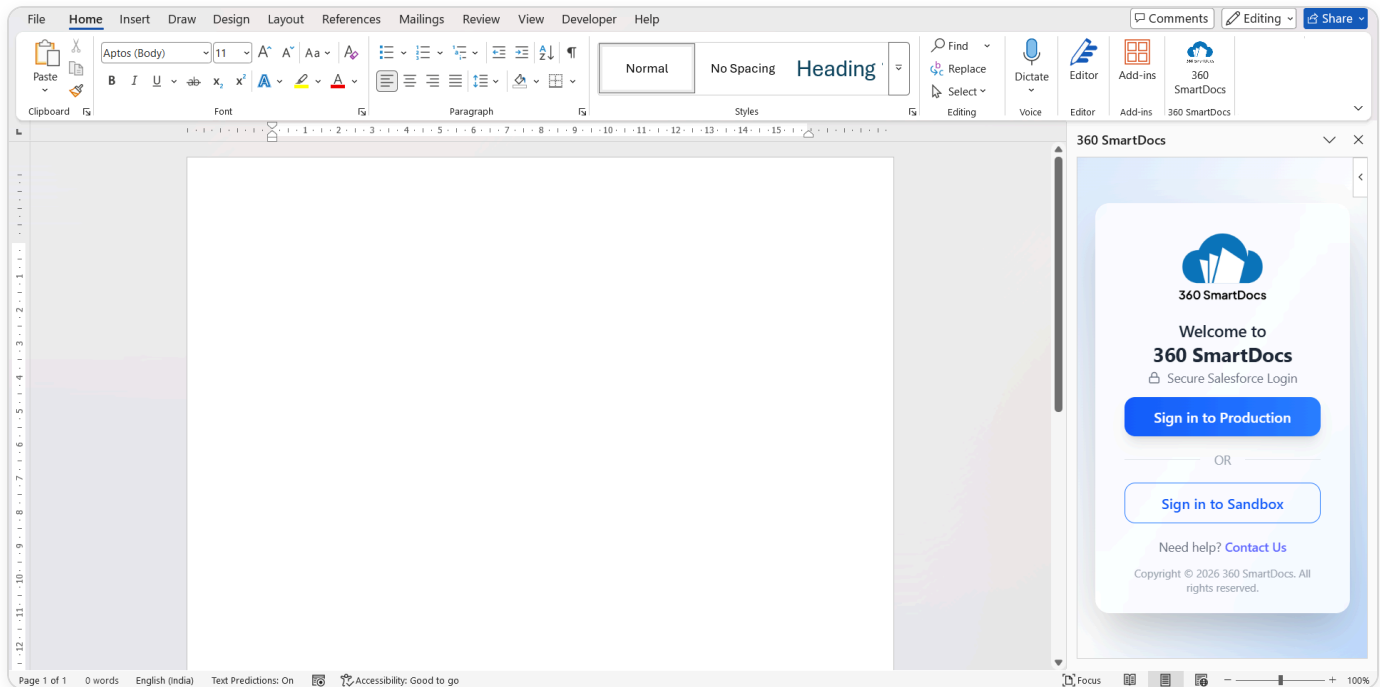


Figure 1: The 360 SmartDocs add-in panel open alongside a Word document

2. Getting Started

Prerequisites

- A Salesforce account (Production or Sandbox)
- Microsoft Word for Desktop (Windows or Mac) **or** Word Online (Microsoft 365 in a browser)
- **360 SmartDocs managed package installed and authorized in your Salesforce org** — if not already done, your Salesforce administrator must:
 1. Install the 360 SmartDocs managed package from the Salesforce AppExchange into your org (Production or Sandbox).
 2. Open the **App Launcher**, navigate to the **360 SmartDocs** app, and click the **Smart Docs Setup** tab.
 3. Select **License Details** from the left sidebar and click "**Authorize & Sync with Server**" to complete the OAuth authorization. Once done, the license status will show **Active**.
- The 360 SmartDocs add-in installed in Word (contact your administrator if not yet installed)

How to Open the Add-In

- 1 Open Microsoft Word on your computer.
- 2 Go to the **Home** tab in the Word ribbon.
- 3 Click **Add-ins** or **My Add-ins**.
- 4 Find **360 SmartDocs** in the list and click **Add**.
- 5 The 360 SmartDocs panel will appear on the right side of your Word window.

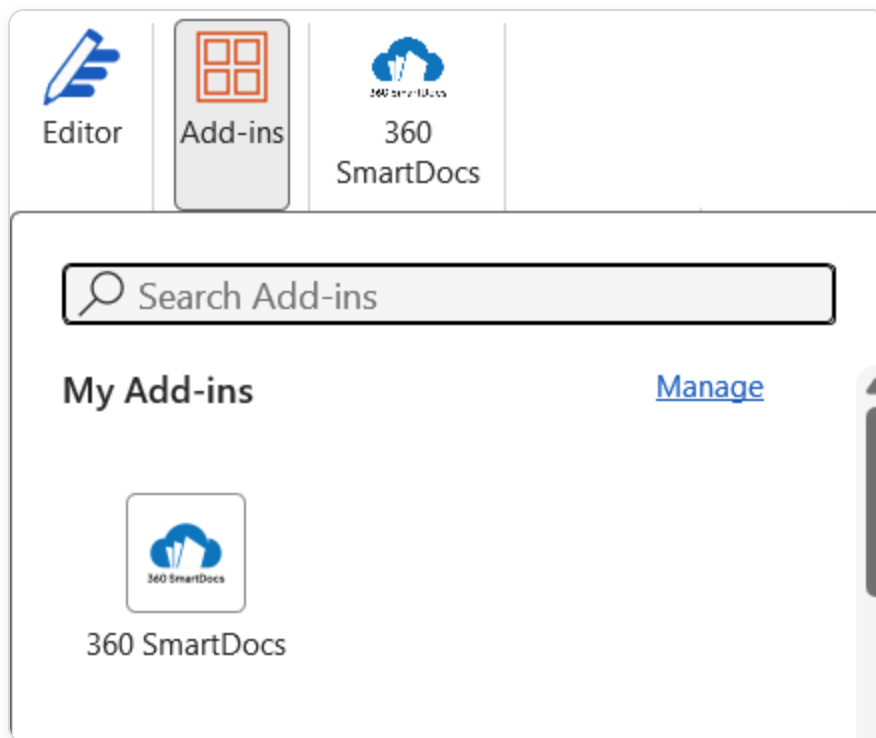


Figure 2: Opening the add-in from Word's Insert tab → Add-ins

2.1 Signing In

When you first open the add-in, you will see the login screen with two buttons:

Button	What it connects to	When to use
Sign in to Production	Your live Salesforce org (real data, real customers)	When building templates for actual use
Sign in to Sandbox	Your test/sandbox Salesforce org (safe to experiment)	When building and testing templates before go-live

1 Click either **Sign in to Production** or **Sign in to Sandbox** depending on your environment.

2 A Salesforce login window will open in your browser or as a dialog.

3 Enter your Salesforce username and password and click **Log In**.

4 If prompted, click **Allow** to authorize 360 SmartDocs to access your Salesforce data.

5 You will be redirected back to the add-in, now logged in and showing your templates.

Login Failed? If you see an authentication error, contact your Salesforce administrator. They may need to check the External Client App settings or grant you access to the 360 SmartDocs application.

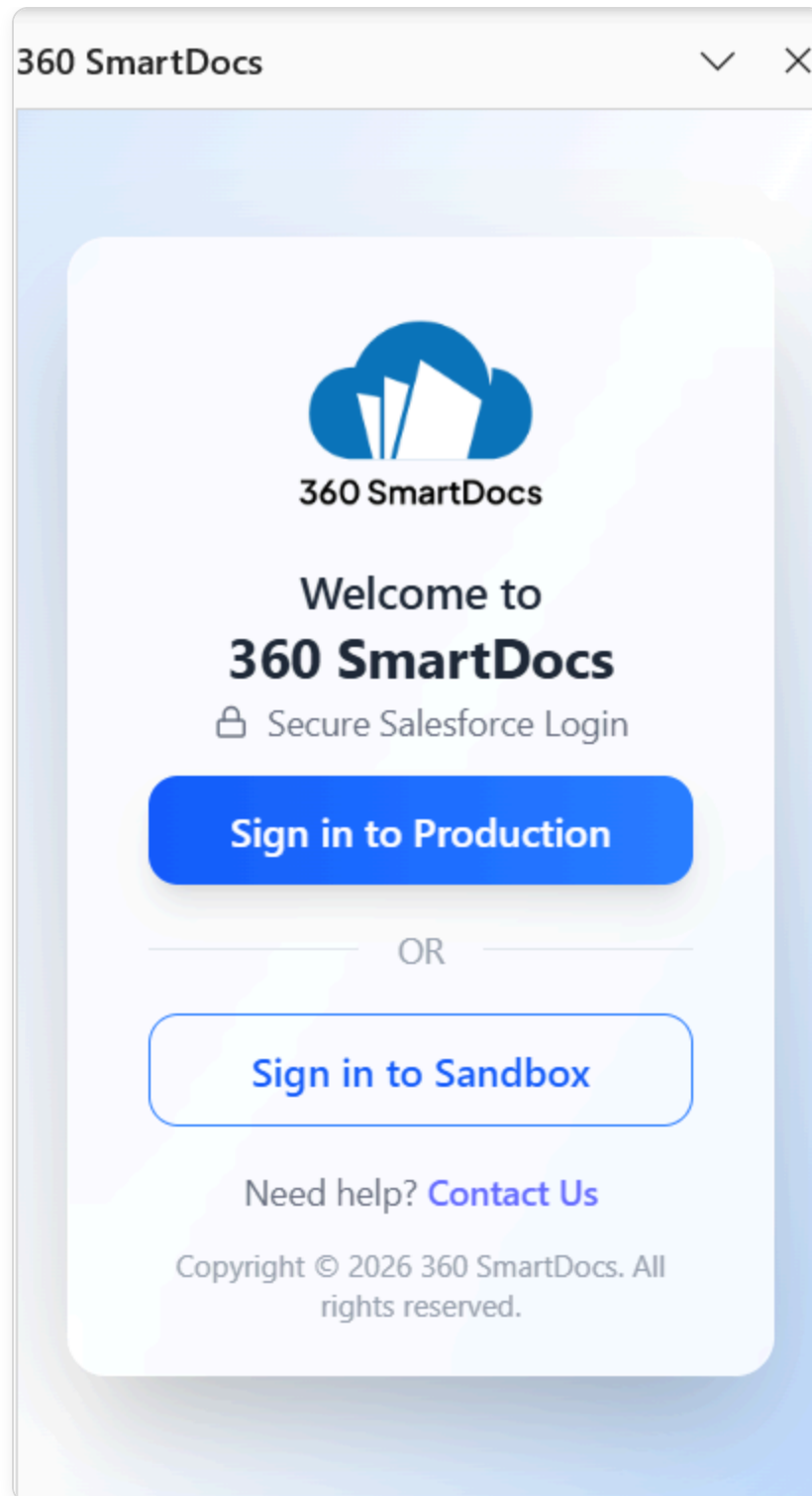
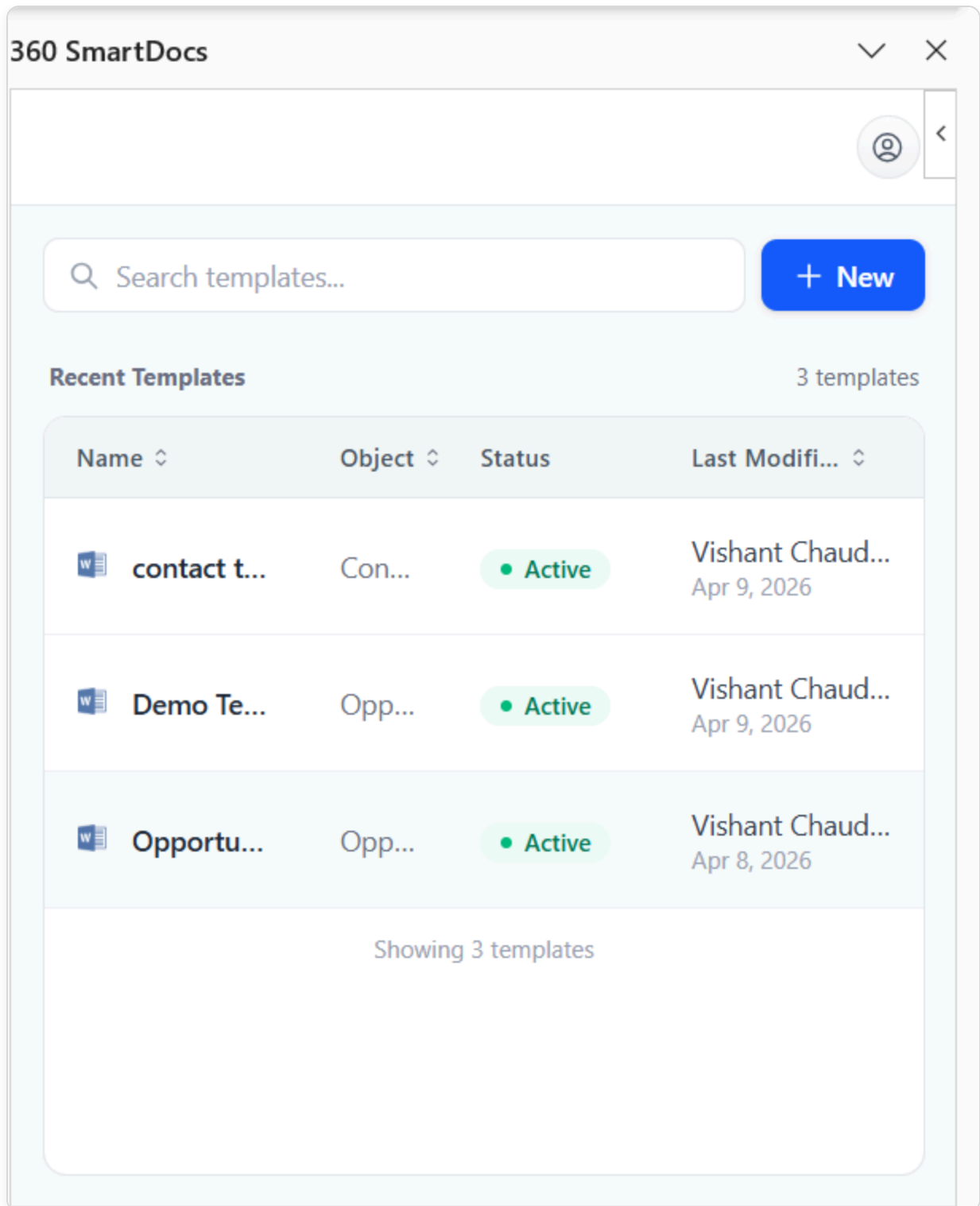


Figure 3: The login screen — choose Production or Sandbox

3. Template Listing Page




After signing in, you will see the **Template Listing** page — a list of all document templates saved in your Salesforce org.



360 SmartDocs

Search templates... **+ New**

Recent Templates 3 templates

Name	Object	Status	Last Modified
 contact t...	Con...	Active	Vishant Chaud... Apr 9, 2026
 Demo Te...	Opp...	Active	Vishant Chaud... Apr 9, 2026
 Opportu...	Opp...	Active	Vishant Chaud... Apr 8, 2026

Showing 3 templates

Figure 4: The Template Listing page

Searching Templates

Type in the search box at the top of the list. The list filters in real-time as you type. Click the × (clear) button to reset the search.

Sorting Columns

Click any column header to sort the list:

- **Name** — sort alphabetically by template name
- **Object** — group by the Salesforce object (Opportunity, Account, etc.)
- **Status** — sort by Active vs Draft
- **Last Modified** — sort by most recently edited

Click the same column header again to reverse the sort order.

Status Indicators

Indicator	Meaning
• Active (green dot)	Template is live and available to users for document generation
• Draft (gray dot)	Template is still being built — not yet available to users

Creating a New Template

Click the **New** button in the top-right corner of the listing. A new blank template will open and the add-in will move to the CREATE tab.

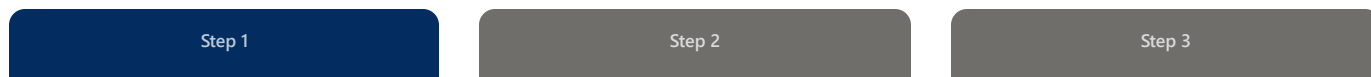
Opening an Existing Template

Click any row in the list to open that template. The template file will open in Word and the add-in will load the template's settings.

Infinite Scroll: If you have many templates, the list loads 20 at a time. Scroll to the bottom to automatically load more.

4. Building a Template — The CREATE Tab

When creating or editing a template, the add-in shows a **3-step wizard** at the top of the panel:



CREATE

PREVIEW

LIVE

In the **CREATE** tab, you:

1. Select a **Primary Object** — the Salesforce record type this template is for
2. Select an **Action** — what type of content to insert (Field, Conditions, View, or Image)
3. Configure the details and click **Insert** or **Copy** to place the tag in Word

Repeat steps 2–3 for every piece of dynamic content in your template.

360 SmartDocs

← 👤 <

1 — 2 — 3
CREATE **PREVIEW** **LIVE**

Object & Action

Search Object

 ▾

Select Action

 ▾

[Continue to Preview](#)

Figure 5: The CREATE tab — 3-step wizard and Object & Action selection

4.1 Selecting a Primary Object

The **Primary Object** is the type of Salesforce record your document is based on. For example:

- **Quote** — for sales quote documents
- **Opportunity** — for proposals and deal summaries
- **Account** — for account profiles and reports
- **Contact** — for contact sheets or letters
- **Order** — for order confirmations

1 Click the **Select Object** dropdown.

2 Type to search for your object (e.g., type "opportunity" to find Opportunity).

3 Select your object from the list.

Important: The Primary Object **cannot be changed** after the template has been saved for the first time. Choose carefully before saving.

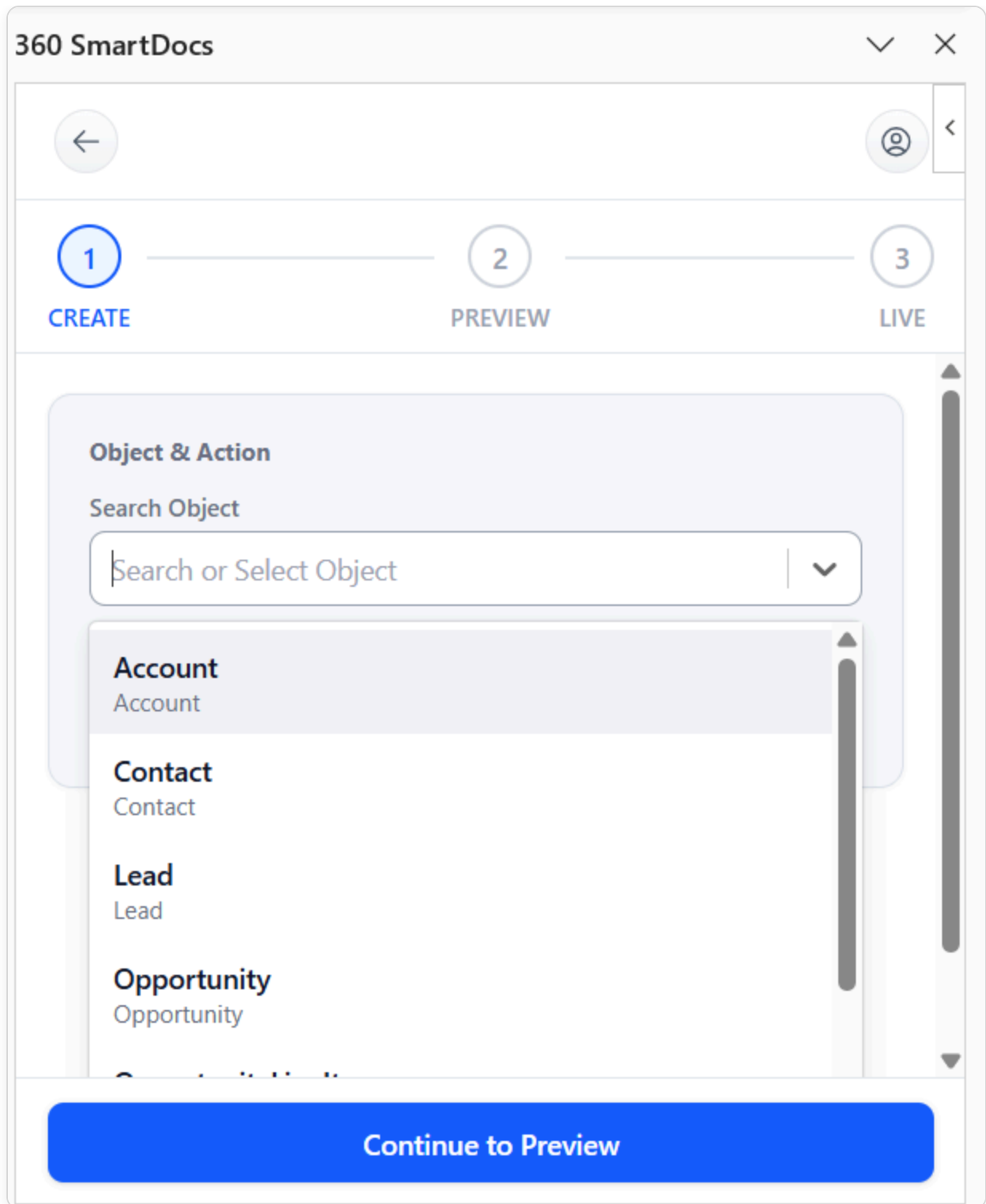


Figure 6: Selecting a Primary Object from the dropdown

4.2 Selecting an Action

After selecting an Object, choose an **Action** from the Action dropdown. The Action determines what type of content the add-in will help you insert:

Action	What it inserts	Example use
Field	A single field value from the record	Opportunity name, amount, close date
Conditions	Text that shows or hides based on a rule	Show "Approved" section only if approved
View	A list of related child records (table, loop, or grid)	All line items on an Opportunity
Image	A dynamic image from the record	Company logo, product photo







5. Action: Field

What it does: Inserts a single field value from a Salesforce record into your Word document. When a document is generated, the tag is automatically replaced with the actual value from the record.

Example: You place the tag `{Account . Name }` in your template. When a user generates a document from an Opportunity where the Account is "Acme Corporation", the tag becomes *Acme Corporation* in the final document.

5.1 Understanding Field Types

When you open the "Select Field" dropdown, fields are grouped by type. Each type has a small icon to help you identify it:

Icon	Field Type	Examples
	Text / String	Account Name, Status, Description, Subject
	Number / Currency / Percent	Amount, Discount %, Quantity, Annual Revenue
	Date / Date-Time	Close Date, Created Date, Expiration Date
	Checkbox (True/False)	Is Active, Closed Won, Has Attachment
	Lookup / Reference (drillable)	Account (on Opportunity), Owner, Campaign, Quote
	Global Variables	TODAY, Running User (you), Running Org (your company)

5.2 Drilling into Reference (Lookup) Fields

Reference fields (🔗 icon) link to another Salesforce object. For example, an Opportunity has a reference to an Account. You can click the > arrow next to a reference field to "drill in" and see that related object's fields.

Maximum depth: 5 levels

Example: Getting the Account Owner's Email from an Opportunity

- 1 On an Opportunity template, open the field selector.
- 2 Find **Account** (reference field 🔗) and click the > arrow — you are now inside Account fields.
- 3 Find **Owner** (reference field 🔗) and click the > arrow — you are now inside User (Owner) fields.
- 4 Select **Email**.
- 5 The resulting tag is: `{Account.Owner.Email}`

More Drilling Examples

Path	Result Tag	What it shows
Quote → Account → Name	<code>{Account.Name}</code>	The account's company name
Quote → Owner → Email	<code>{Owner.Email}</code>	The quote owner's email
Quote → Account → Owner → Manager → Department	<code>{Account.Owner.Manager.Department}</code>	Manager's department (4 levels deep)
Opportunity → Account → BillingCity	<code>{Account.BillingCity}</code>	The account's billing city

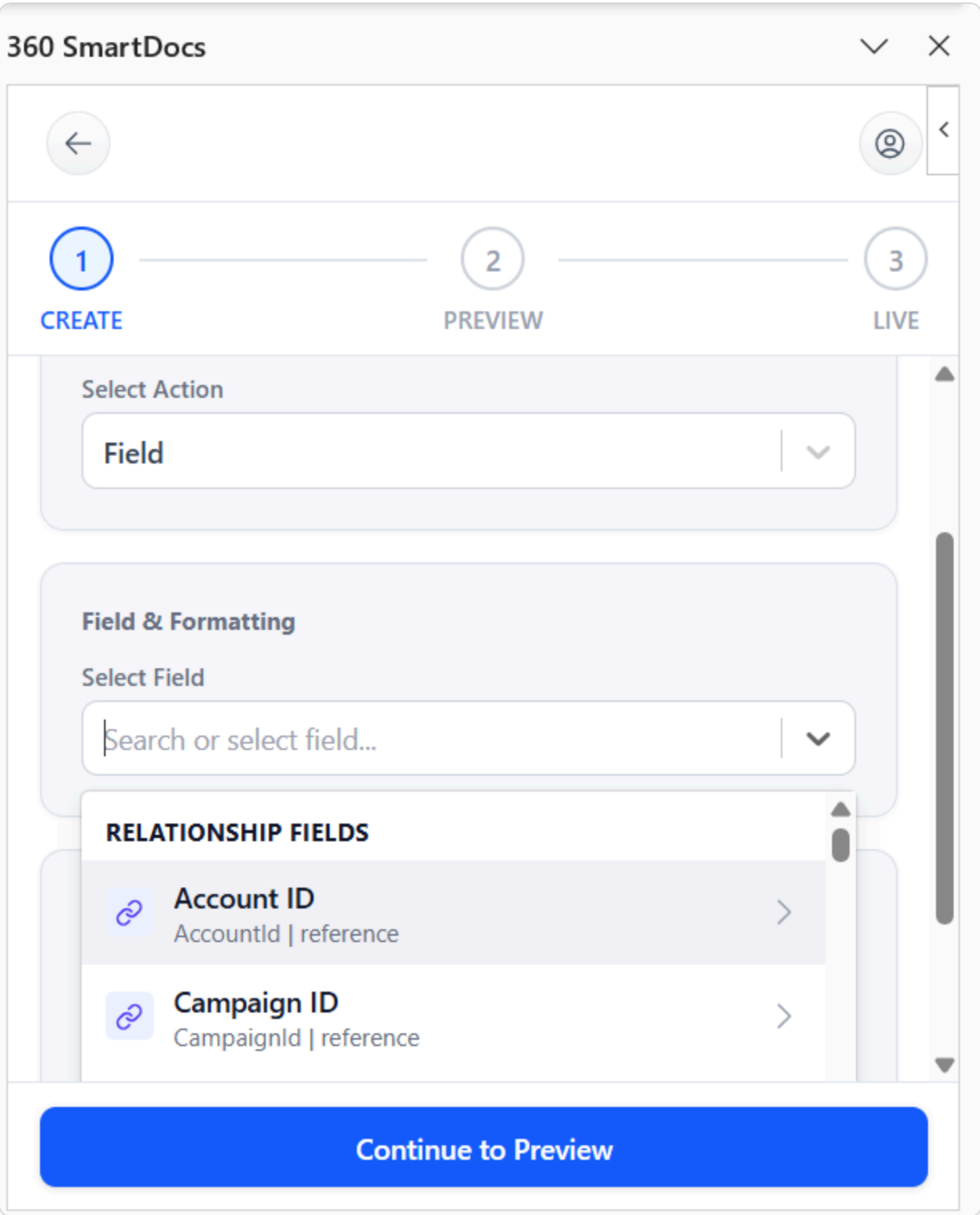


Figure 7: The field selector showing field types with icons

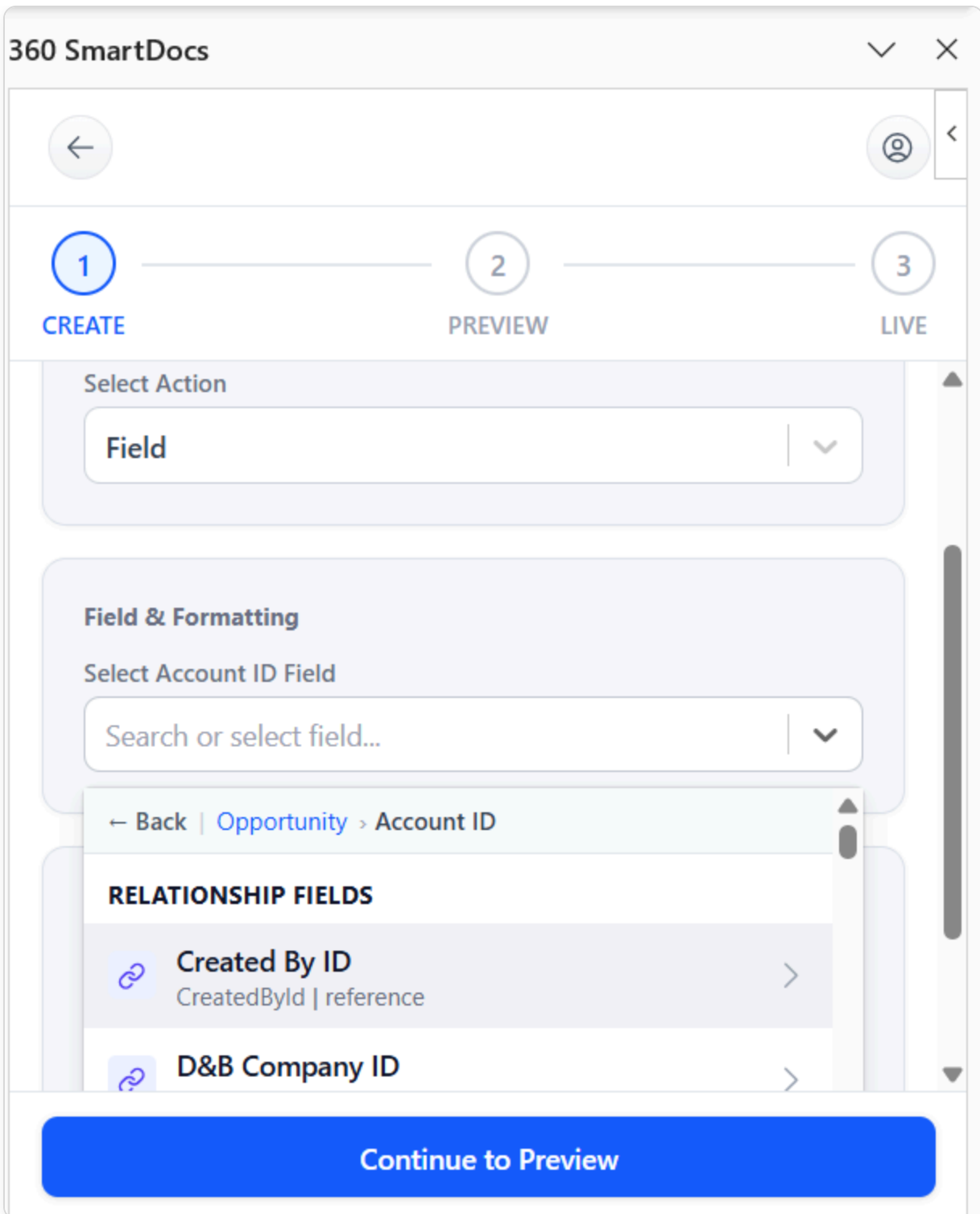



Figure 8: Drilling into a reference field — click the > arrow to navigate deeper

5.3 Global Variables

Global Variables are available in the  **Globe** group in the field selector. These values are **automatically calculated** each time a document is generated — they do not come from the Salesforce record.

Date Literals

Tag	What it inserts	Real-world use case
{TODAY}	Today's date	"Generated on: {TODAY formatDate:'MM/DD/YYYY'}" — auto-stamps the current date on every document
{YESTERDAY}	Yesterday's date	Daily sales reports: "Activity as of {YESTERDAY formatDate:'MM/DD/YYYY'}"
{TOMORROW}	Tomorrow's date	Urgent deadlines: "Response required by {TOMORROW formatDate:'MMMM DD, YYYY'}"
{THIS_WEEK}	Monday of the current week	Weekly status reports: "Week of {THIS_WEEK formatDate:'MMM DD'}"
{THIS_MONTH}	1st day of the current month	Monthly invoices: "Invoice period: {THIS_MONTH formatDate:'MMMM YYYY'}"
{THIS_QUARTER}	1st day of the current fiscal quarter	Quarterly reviews: "Period starting {THIS_QUARTER formatDate:'MM/DD/YYYY'}"
{THIS_YEAR}	January 1st of the current year	Annual reports: "Year-to-date from {THIS_YEAR formatDate:'MM/DD/YYYY'}"
{CURRENT_YEAR}	Just the 4-digit year number	Copyright footers: "© {CURRENT_YEAR} {Organization.Name}"
{CURRENT_MONTH}	Month as a number (1–12)	Custom date labels
{CURRENT_DAY}	Day of month as a number	Custom date labels

Running User — The Person Generating the Document

These tags automatically resolve to the details of whoever is generating the document at the time. This means different users will see their own name and contact info in generated documents.

Tag	What it inserts	Example use
<code>{User.FirstName}</code>	First name of the generating user	Personalized "Prepared by" footer
<code>{User.LastName}</code>	Last name	Same
<code>{User.Email}</code>	Email address	"Questions? Contact <code>{User.Email}</code> "
<code>{User.Title}</code>	Job title (e.g., Account Executive)	Show the rep's role below their name
<code>{User.Department}</code>	Department name	"From the <code>{User.Department}</code> team"
<code>{User.Phone}</code>	Direct phone number	"Call your rep: <code>{User.Phone}</code> "

Example — Document footer that personalizes itself per rep:

```
Prepared by: {User.FirstName} {User.LastName}
Title: {User.Title} | Department: {User.Department}
Email: {User.Email} | Phone: {User.Phone}
```

Every time a different sales rep generates this template, their own contact information appears automatically.

Running Org — Your Salesforce Organization

These tags reflect your organization's Salesforce settings.

Tag	What it inserts	Example use
<code>{Organization.Name}</code>	Your company/organization name	Letterheads, copyright footers
<code>{Organization.InstanceName}</code>	Salesforce instance identifier	Internal reference only

Example — Consistent letterhead across all templates:

```
{Organization.Name}
Confidential — © {CURRENT_YEAR} {Organization.Name}. All rights reserved.
```

5.4 Step-by-Step: Inserting a Field

- 1 Make sure you have selected a **Primary Object** (e.g., Opportunity).
- 2 From the **Action** dropdown, choose **Field**.
- 3 Click the **Select Field** dropdown to open the field browser.
- 4 Browse or type to search for your field (e.g., type "account" to find Account-related fields).
- 5 If you see a field with a > chevron (reference/lookup field), click it to drill into that related object and see its fields.
- 6 Click your target field to select it.
- 7 If the field is a **number**, **currency**, or **date** type, configure formatting options that appear below (see Section 5.5).
- 8 Click **Insert** to place the tag at your cursor in Word, or **Copy** to copy it to your clipboard.

Tip: Click inside your Word document first to position your cursor before clicking Insert. The tag will be placed exactly where your cursor is.

5.5 Field Formatting Options

Number, Currency & Percent Fields

When you select a numeric field, formatting options appear below the field selector.

Decimal Places (0–10): How many digits to show after the decimal point.

- Example: 12345.6789 with Decimal Places = 2 → **12,345.68**
- Example: 12345.6789 with Decimal Places = 0 → **12,346**

Number Format: The style of thousands and decimal separators.

Format Name	Example Output	When to use
US — 10,000.5	12,345.68	United States documents
European — 10.000,5	12.345,68	European documents (Germany, France, etc.)
International — 10 000,5	12 345,68	International / French style
Indian — 1,00,000	1,23,456.68	India and South Asia

Currency Symbol: Adds a currency symbol before the number.

- Options: \$ (USD) € (Euro) £ (Pound) ¥ (Yen/Yuan) ₹ (Rupee)
- Example: Amount = 50000, Symbol = \$, Decimals = 2 → **\$50,000.00**

360 SmartDocs

←

1 CREATE

2 PREVIEW

3 LIVE

Field & Formatting

Select Field

Opportunity > Amount

Decimal Places

2

Format

10,000,000.5

Symbol

\$

Output

{Amount | format:2:'10,000,000.5': '\$'}

Continue to Preview

Figure 9: Number formatting options

Steps for number formatting:

1 Select a numeric, currency, or percent field from the field selector.

2 Set the **Decimal Places** (e.g., 2 for money, 0 for whole numbers).

3 Choose the **Number Format** appropriate for your document's region.

4 Choose a **Currency Symbol** if the field represents money.

5 Click **Insert** or **Copy**.

Number Tag Syntax — `toFixed` vs `format`

The add-in generates one of two filters depending on what you choose. You can also type these directly:

- `toFixed:N` — controls decimal places only. Use this when you want to keep your org's default number formatting and just override the decimal precision.
 - Example: `{TotalPrice | toFixed:2}`
- `format:decimals:'pattern'` — full number format (decimal places + thousands/decimal separators). Use this when you need a specific regional pattern regardless of org defaults.
 - Example: `{Amount | format:0:'10,000,000.5'}` → integer with US-style thousands separator
- `format:decimals:'pattern':'symbol'` — same as above, with a currency symbol prepended.
 - Example: `{Amount | format:2:'10,000,000.5':'$'}` → **\$1,234,567.89**

Pattern values match the Number Format dropdown choices: `'10,000.5'` (US), `'10.000,5'` (European), `'10 000,5'` (International), `'1,00,000'` (Indian).

Which one should I use? Pick `toFixed` when you only care about decimal places and want the org's default number formatting. Pick `format` when you need a specific separator pattern or currency symbol.

Date & DateTime Fields

When you select a Date or DateTime field, formatting options appear. Both Date and DateTime fields support every date format below; DateTime fields additionally let you append a time format.

Date Format (works for both Date and DateTime fields):

Format	Example Output
MM/DD/YYYY	12/31/2024
DD/MM/YYYY	31/12/2024
YYYY/MM/DD	2024/12/31
MMMM DD, YYYY	December 31, 2024
MMM DD, YYYY	Dec 31, 2024
DD MMMM YYYY	31 December 2024

Time Format (DateTime fields only — append to any date format above when a time component is needed):

Format	Example Output
hh:mm A	03:45 PM
HH:mm	15:45
hh:mm:ss A	03:45:09 PM
HH:mm:ss	15:45:09

If a DateTime field doesn't need a time component, just pick a date format and "None" in time format — it will render as a plain date.

Add Days / Subtract Days: Offsets the date by a set number of days from the field's value. Both `addDays` and `subtractDays` accept positive or negative integers.

- **Add days (future):** `addDays:30` → 30 days after the field value (e.g., Close Date + 30 days)
- **Subtract days (past):** `subtractDays:7` → 7 days before the field value (e.g., one week before the contract end)
- Both filters accept negative numbers too — `addDays:-7` is equivalent to `subtractDays:7`.
- **Use case:** Payment due date = `{InvoiceDate | addDays:30}`. Expiry reminder = `{ContractEndDate | subtractDays:14}`.

360 SmartDocs

←

1 CREATE 2 PREVIEW 3 LIVE

Field & Formatting

Select Field

Opportunity > Close Date

Date Format

01/31/2025

Add Days (use negative to subtract)

0

Output

```
{CloseDate | formatDate: 'MM/DD/YYYY'}
```

Continue to Preview

Figure 10: Date formatting options

Steps for date formatting:

1 Select a Date or DateTime field.

2 Choose a **Date Format** from the dropdown.

3 (DateTime only) Choose a **Time Format** if you need a time component; leave it blank for date-only output.

4 (Optional) Enter a number in **Add Days** or **Subtract Days** to offset the date (positive or negative).

5 Click **Insert** or **Copy**.

Tag Examples

```
{Name}
{Account.Name}
{Account.Owner.Email}
{CloseDate | formatDate:'MMMM DD, YYYY'}
{CreatedDate | formatDate:'MM/DD/YYYY HH:mm:ss'}
{LastModifiedDate | formatDate:'DD/MM/YYYY hh:mm A'}
{Amount | format:2:'10,000,000.5': '$'}
{TotalPrice | toFixed:2}
{CloseDate | addDays:30 | formatDate:'MM/DD/YYYY'}
{ContractEndDate | subtractDays:14 | formatDate:'MM/DD/YYYY'}
{TODAY | formatDate:'MM/DD/YYYY'}
{User.FirstName} {User.LastName}
{Organization.Name}
© {CURRENT_YEAR} {Organization.Name}
```

6. Action: Conditions

What it does: Shows or hides a block of text in the document based on a rule you define. Think of it as: "*if this condition is true, then show this content.*"

Example: Only show a "Terms & Conditions" section if the customer is an Enterprise client. For standard clients, that section simply doesn't appear.

6.1 If Block vs. If/Else Statement

Type 1 — If Block

Content is shown **only when the condition is TRUE**. If the condition is FALSE, nothing is shown — the section is completely hidden and takes up no space in the document.

- **Best for:** Large optional blocks of content — entire paragraphs, multi-line clauses, full sections
- **Real examples:**
 - Show a full "Terms & Conditions" section only for Enterprise tier customers
 - Show a "California Residents" legal clause only when Billing State = "CA"
 - Show an "Installation Instructions" block only when Product Type = "Hardware"
 - Show a "Payment Plan Details" section only when a payment plan was selected
 - Show a "Warranty Information" section only for physical products

Type 2 — If/Else Statement

Shows **Content A** when the condition is TRUE, or **Content B** when it is FALSE. Something always appears — it just changes based on the data.

- **Best for:** Short inline labels, single words, or one-line phrases that need two versions
- **Real examples:**
 - Amount > \$10,000 → show "Premium Client" else show "Standard Client"
 - Is Active → show "Active Account" else show "Inactive Account"
 - Stage = "Closed Won" → show "✓ Closed" else show "○ Open"
 - Discount > 0 → show "Discounted Pricing Applied" else show "Standard Pricing"

How this renders in the template: The add-in's **If/Else Statement** produces an inline ternary like `{Amount > 10000 ? "Premium Client" : "Standard Client"}`. The TRUE and FALSE values can be either literal text (quoted) or Salesforce field names (unquoted) — for example `{IsActive ? active_label__c : inactive_label__c}`. If you need **block-level** alternate content (multiple lines or paragraphs in the FALSE branch), type it into Word manually as an *inverted-section* pair: take the `{#Condition}...{/}` If Block and add a second block immediately after it with `#` replaced by `^` — `{^Condition}...{/}`. That second block is what renders when the condition is false.

6.2 Nested Conditions

You can place a condition inside another condition to build complex logic. This is done by typing tags directly into your Word document (not available via the add-in UI buttons).

Pattern 1 — If Block inside an If Block

Show inner content only when BOTH the outer AND inner conditions are true:

```
{#Region == "North America"}
North American Standard Terms Apply.

  {#CustomerType == "Enterprise"}
  Enterprise SLA: 4-hour guaranteed response time, 24/7 support.
  {/}

{/}
```

Result: All North American customers see the first line. Only North American Enterprise customers also see the SLA line.

Pattern 2 — If/Else Statement inside an If Block

Show a label that changes, but only within a larger conditional section:

```
{#Status == "Active"}
This account is currently active.
Account tier: {Amount > 50000 ? "Premium" : "Standard"}
{/}
```

Pattern 3 — Nested If/Else Block (Multiple Tiers)

Show different content for three or more tiers:

```
{#Amount > 100000}
Platinum Client – Dedicated Account Manager Assigned
{/}
{^Amount > 100000}
  {#Amount > 50000}
  Gold Client – Priority Support Included
  {/}
  {^Amount > 50000}
  Standard Client – Self-Service Portal Access
  {/}
{/}
```

Result: Platinum for >\$100k, Gold for \$50k–\$100k, Standard for everything else.

How to add nested conditions: Build the outer condition first using the add-in, then position your cursor inside that block in Word and type or Insert the inner condition. Make sure every `{#...}` opening tag has a matching `{/...}` closing tag. Use the Preview tab to test all combinations.

6.3 Operators & Multiple Conditions

Comparison Operators

Operator	Meaning	Example
==	Equals exactly	Status == "Approved"
!=	Does not equal	Status != "Draft"
>	Greater than	Amount > 10000
<	Less than	Discount < 5
>=	Greater than or equal to	Score >= 80
<=	Less than or equal to	Days <= 30
contains	Field value contains this text	Name contains "Corp"
startsWith	Field value starts with this text	Email startsWith "info@"
endsWith	Field value ends with this text	Email endsWith "@gmail.com"

Multiple Conditions

Click "**Add Condition**" in the add-in to add more condition rows. Connect them with:

- **AND (&&)** — ALL conditions must be true for the content to show
- **OR (||)** — if ANY one condition is true, the content shows

6.4 Step-by-Step: Building a Condition

- 1 Make sure your **Primary Object** is selected.
- 2 From the **Action** dropdown, choose **Conditions**.
- 3 Choose the condition type: **If Block** or **If/Else Statement**.

- 4 Select a **Field** from the dropdown (e.g., StageName).
- 5 Choose an **Operator** from the dropdown (e.g., ==).
- 6 Enter a **Value** in the text box (e.g., type `Closed Won`). This is the value you are comparing against.
- 7 (Optional) Click **Add Condition** to add another rule, then choose AND or OR to connect them.
- 8 In the **Content** text box, type the text or paragraph to show when the condition is **TRUE**.
- 9 (If/Else only) In the **Alternate Content** text box, type the text to show when the condition is **FALSE**.
- 10 Click **Insert** or **Copy**.

360 SmartDocs

←

1 CREATE 2 PREVIEW 3 LIVE

Select Action

Conditions

Condition to be met

If Block

1

Select Field	Operator	Value
Select Field	[==]	Search or ins

+ Add Condition

Continue to Preview

Figure 11: Setting up an If Block condition

360 SmartDocs

∨
✕

←

👤
<

1
CREATE

2
PREVIEW

3
LIVE

Condition to be met

If / Else Statement
∨

1

Select Field	Operator	Value
<div style="border: 1px solid #ccc; padding: 2px; display: flex; justify-content: space-between; align-items: center;"> Select Field ∨ </div>	<div style="border: 1px solid #ccc; padding: 2px; display: flex; justify-content: space-between; align-items: center;"> [==] ∨ </div>	<div style="border: 1px solid #ccc; padding: 2px; display: flex; justify-content: space-between; align-items: center;"> Search or ins ∨ </div>

+ Add Condition

Select Content Value

if Value

Select or search Field or enter value
∨

Continue to Preview

Figure 12: If/Else statement with Content and Alternate Content

6.5 Real-Life Examples

Example 1 — If Block: Finance Approval Notice

Show the approval paragraph only when Approval Status equals "Approved":

```
{#ApprovalStatus == "Approved"}
This quote has been reviewed and approved by the Finance department.
All pricing and terms are final.
{/}
```

Example 2 — If/Else Block: Deal Tier Label

Show "ENTERPRISE DEAL" for amounts over \$50,000, otherwise "STANDARD DEAL":

```
{#Amount > 50000}
ENTERPRISE DEAL – VP Approval Required
{/}
{^Amount > 50000}
STANDARD DEAL – Standard Terms Apply
{/}
```

Example 3 — Multiple Conditions (AND): Priority Customer Notice

Show a notice only for customers who are Active AND New:

```
{#Status == "Active" && Type == "New Customer"}
★ Priority New Customer – Please attach the Onboarding Checklist.
{/}
```

Example 4 — Multiple Conditions (OR): Payment Processing Note

Show a note for customers paying by Check OR Wire Transfer:

```
{#PaymentMethod == "Check" || PaymentMethod == "Wire Transfer"}
Please allow 5-7 business days for payment processing.
{/}
```

6.6 Custom Conditions — Direct Document Editing (Advanced)

This is the same **inline If/Else Statement syntax** described in [6.1](#) — the same format the add-in's **If/Else Statement** button generates for you. You only need to type it by hand when the UI button doesn't cover what you need: for example, using a Salesforce field name (unquoted) as the TRUE or FALSE value instead of literal text, combining comparison operators the button doesn't expose, or placing the condition inside a nested block.


Syntax: {field operator value ? "text if true" : "text if false"}

Selecting a child relationship works the same way as selecting a field in the Field action (see [Section 5](#)):

1

Click the **Select Field** dropdown to open the field browser.

2

Browse or type to search for a relationship field (look for the  reference icon).

3

Click the > chevron next to a reference/lookup field to drill into that related object.

4

Continue drilling deeper through relationship fields as needed (up to 5 levels).

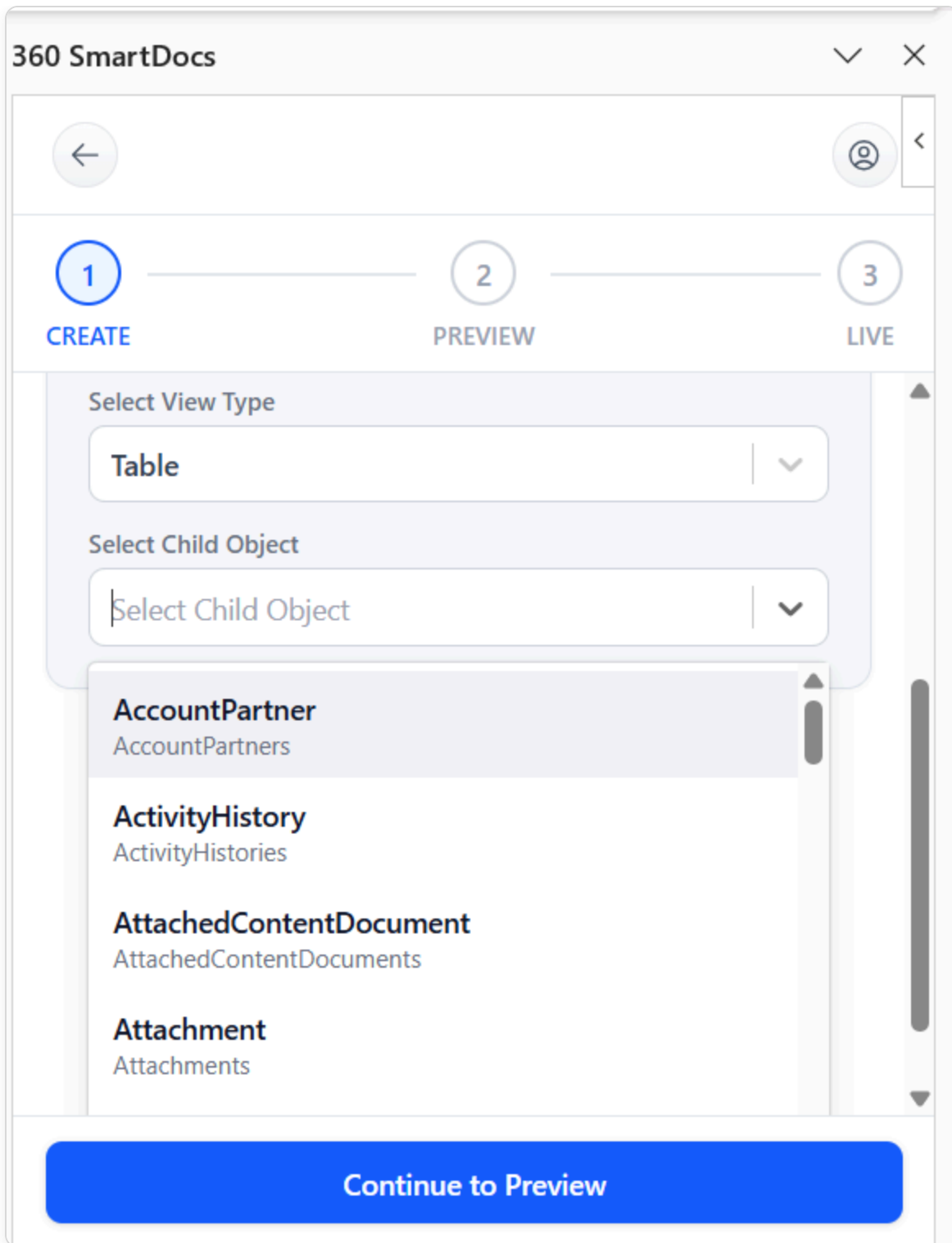


Figure 15: Selecting the child relationship using the field dropdown

Step 3: (Optional) Using Parent Fields Inside a Loop

Inside a loop, you can reference fields from the *parent* record (the primary object). For example, inside an OpportunityLineItems loop, you can display the parent Opportunity's name on every row.

- Use dot notation referring to the parent: `{Opportunity.Name}` inside an OpportunityLineItems loop
- This is useful for things like showing the Opportunity Name as first column for each line item row

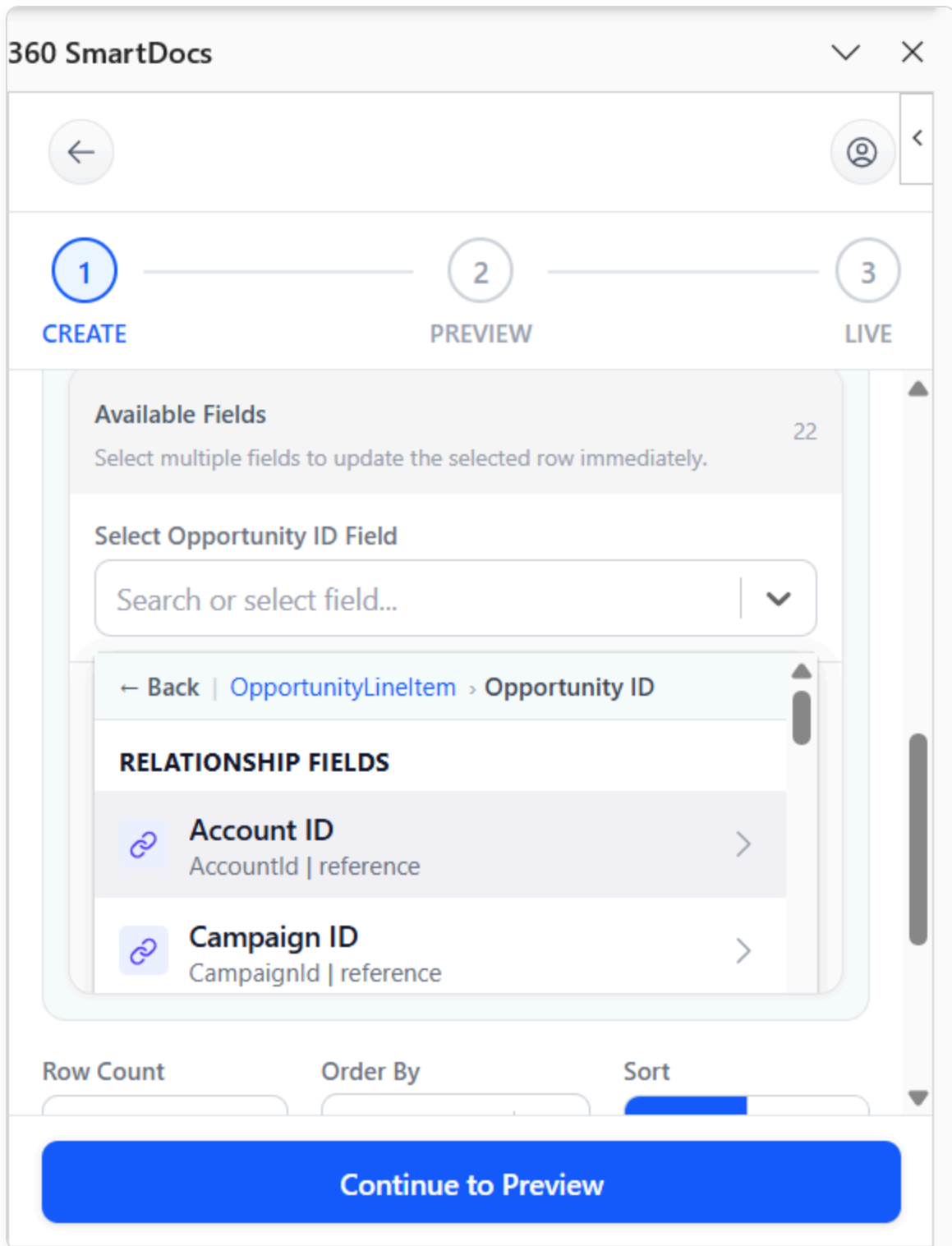


Figure 16: Accessing parent record fields inside a loop

Step 4: Add Fields to "Selected Fields"

Selecting fields for the view works the same way as the Field action (see [Section 5](#)):

- 1 Click the **Select Field** dropdown to open the field browser.

2

Browse or type to search for your desired field.

3

If you see a field with a > chevron (reference/lookup field), click it to drill into that related object and see its fields.

4

Click your target field to select it — it will be added to the **Selected Fields** list.

5

Repeat to add more fields. Fields appear in the Selected Fields list in the order they were selected.

6

To remove: Click the remove button next to a field in the Selected Fields list.

Each selected field is shown with a number indicating its position in the list.

360 SmartDocs

← (Back) | (User Profile) | (Close)

1 CREATE — 2 PREVIEW — 3 LIVE

Select Field

Search or select field... | (Dropdown Arrow)

Selected Fields 5

Arrange the output order here.

1 Name × | 2 Product Name ×

3 List Price × | 4 Quantity ×

5 Total Price ×

Row Count: [] | Order By: Select Field | (Dropdown Arrow) | Sort: ASC | DESC

Continue to Preview

Figure 17: Selecting fields using the dropdown and viewing them in the Selected Fields list

Step 5: Reorder Your Fields

The order of fields in "Selected Fields" = the column order (for Table) or display order (for Loops/Grid) in the document. By default, fields appear in the order they were selected.

To reorder:

- **Drag and drop** a field in the Selected Fields list to your desired position.

- Each field displays a number indicating its current position — the numbering updates automatically as you reorder.

The screenshot shows the 360 SmartDocs interface. At the top, there's a header with the title '360 SmartDocs' and navigation icons. Below the header is a progress bar with three steps: '1 CREATE', '2 PREVIEW', and '3 LIVE'. The main content area is titled 'Select Field' and contains a search box 'Search or select field...'. Below this is a 'Selected Fields' section with the instruction 'Arrange the output order here.' and a count of '5'. The selected fields are: '1 Name', '2 Product Name', '3 Quantity', '4 List Price', and '5 Total Price'. Each field has a small 'x' icon to remove it. At the bottom of the interface, there are three controls: 'Row Count' (an empty input field), 'Order By' (a dropdown menu showing 'Select Field'), and 'Sort' (two buttons: 'ASC' and 'DESC'). A large blue button labeled 'Continue to Preview' is positioned at the very bottom.

Figure 18: Reordering Selected Fields by dragging and dropping

Step 6: Set Row Count (Row Limit)

Enter the maximum number of child records to include in the document.

- Example: Enter **10** → only the first 10 records are shown
- Leave blank or enter **0** for unlimited (all records)

- **Recommendation:** Set a reasonable limit (e.g., 50 for line items) to prevent unexpectedly long documents

Step 7: Set Sort Field

Choose which field to use for sorting the child records. Example: sort Opportunity Line Items by "Quantity" or "Unit Price".

Step 8: Set Sort Direction

- **ASC** (Ascending): A → Z, 0 → 9, Oldest → Newest
- **DESC** (Descending): Z → A, 9 → 0, Newest → Oldest

Step 9: (Grid only) Set Grid Columns

Choose how many cards appear per row: **2**, **3**, or **4** columns.

Step 10: Insert or Copy

Click **Insert** to place the view block at your cursor in Word, or **Copy** to paste it manually.

The screenshot shows the 360 SmartDocs interface. At the top, there's a title bar with "360 SmartDocs" and navigation icons. Below that, a progress bar shows three steps: 1. CREATE, 2. PREVIEW, and 3. LIVE. The current step is PREVIEW. In the center, there's a preview of a table with three columns: "Quantity", "List Price", and "Total Price". Below the table, there are controls for "Row Count" (set to 5), "Order By" (set to "Quantity"), and "Sort" (set to "DESC"). At the bottom, there are two large blue buttons: "+ Insert" and "Continue to Preview".

Row Count: 5

Order By: Quantity

Sort: ASC | **DESC**

Preview

```
{#OpportunityLineItems |
orderBy:'Quantity':'desc' | limit:5}
{Opportunity.Name} | {Product2.Name} |
{Quantity} | {ListPrice} | {TotalPrice}
```

+ Insert

Continue to Preview

Figure 19: The generated view tag and output options

7.3 Real-Life Examples

Example 1 — Table: Quote Line Items on a Sales Quote

Scenario: You're building a Sales Quote document. The customer wants to see all products with their quantities and prices in a clean table.

Setting	Value
View Type	Table
Child Relationship	QuoteLineItems
Selected Fields	Product Name, Description, Quantity, Unit Price, Total Price
Sort Field	Product Name
Sort Direction	ASC
Row Count	50

What the final document looks like:

Product Name	Description	Qty	Unit Price	Total
Support Plan Annual	24/7 support coverage	1	\$200.00	\$200.00
Widget Lite	Basic model	10	\$49.00	\$490.00
Widget Pro	Standard model	5	\$99.00	\$495.00

Example 2 — Loops: Contact Directory on an Account Profile

Scenario: You're creating an Account Profile document that lists all key contacts at the customer's company, formatted as labeled blocks — one per person.

Setting	Value
View Type	Loops
Child Relationship	Contacts
Selected Fields	First Name, Last Name, Title, Email, Phone
Sort Field	Last Name
Sort Direction	ASC
Row Count	10

What the final document looks like:

Name: Sarah Johnson
 Title: VP of Procurement
 Email: s.johnson@acme.com
 Phone: (415) 555-0101

Name: Michael Chen
 Title: IT Director
 Email: m.chen@acme.com
 Phone: (415) 555-0202

Example 3 — Grid: Product Brochure Catalog

Scenario: You're creating a product brochure and want to display products as compact cards — 3 per row — showing the name, description, and price.

Setting	Value
View Type	Grid
Child Relationship	Products (or OpportunityLineItems)
Selected Fields	Product Name, Short Description, Price
Sort Field	Name
Sort Direction	ASC
Grid Columns	3

What the grid layout looks like (3 cards per row):

Widget Lite	Widget Max	Widget Pro
Basic model	Premium model	Standard model
\$49.00	\$199.00	\$99.00

7.4 Summarize / Aggregate (Advanced — Direct Document Entry)

The system supports summing a numeric field across all child records. This feature is not available in the add-in's UI but works when you type the tag directly into your Word document.

Syntax: {RelationshipName | sumBy:'FieldName'}

```
Total Amount: {OpportunityLineItems | sumBy:'TotalPrice'}
```

Grand Total (formatted option 1):

```
{OpportunityLineItems | sumBy:'TotalPrice' | toFixed:2}
```

Grand Total (formatted option 2):

```
{OpportunityLineItems | sumBy:'TotalPrice' | format:2:'10,000,000.5':'$'}
```

Total Items Ordered:

```
{OpportunityLineItems | sumBy:'Quantity'}
```

1

Click in your Word document where you want the total to appear (e.g., below your line items table).

2

Type: {OpportunityLineItems | sumBy:'TotalPrice'}

3

Replace OpportunityLineItems with your actual child relationship name.

4

Replace TotalPrice with the exact Salesforce API field name (case-sensitive).

5

Add formatting if needed: | format:2:'10,000,000.5':'\$' for a currency total, or | toFixed:2 for decimals only.

6

Go to the **Preview** tab to verify the sum calculates correctly.

Case-Sensitive Field Names: Field names in sumBy must match the Salesforce API name exactly (e.g., TotalPrice not Total Price). Check in Salesforce Setup → Object Manager if unsure.

7.5 Calc — Expression Evaluator (Advanced — Direct Document Entry)

The `calc` filter is a powerful, unified expression evaluator that lets you perform arithmetic, comparisons, and logical operations directly in your template tags. It works on both arrays (child records) and scalar values (text, numbers). This feature is not available in the add-in's UI — type the tags directly into your Word document.

Syntax: `{Input | calc:'expression'}`

Arithmetic on Child Record Fields

Use field names inside the expression to sum that field across all child records, then perform arithmetic on the result. You can use `sumBy(Field Name)` or just use the field name directly — both are equivalent.

Sum of TotalPrice:

```
{OpportunityLineItems | calc:'TotalPrice'}
```

Same as above using explicit `sumBy`:

```
{OpportunityLineItems | calc:'sumBy(TotalPrice)'}
```

Multiply two field sums:

```
{OpportunityLineItems | calc:'UnitPrice * Quantity'}
```

Compound expression with parentheses:

```
{OpportunityLineItems | calc:'(UnitPrice * Quantity) / 100'}
```

Formatted result:

```
{OpportunityLineItems | calc:'(UnitPrice * Quantity) / 100' | toFixed:2}
```

```
{OpportunityLineItems | calc:'TotalPrice' | format:2:'10,000,000.5': '$'}
```

Percentage Discount / Markup

Use `N%` after `+` or `-` to add or subtract a percentage of the preceding value. This follows common business logic for discounts and markups.

Subtract 10% discount from total:

```
{OpportunityLineItems | calc:'TotalPrice - 10%' | toFixed:2}
```

Add 15% markup:

```
{OpportunityLineItems | calc:'TotalPrice + 15%' | toFixed:2}
```

Chain with other operations:

```
{OpportunityLineItems | calc:'(UnitPrice * Quantity) - 5%' | toFixed:2}
```

How percentage works: `TotalPrice - 10%` means "sum of TotalPrice minus 10% of that sum." If the sum is 1000, the result is $1000 - 100 = 900$.

Using Parent Record Values (\$1, \$2, ...)

Filters cannot access parent record fields directly. To use a parent field value inside a calc expression, pass it as an extra argument after the expression string and reference it with `$1`, `$2`, etc.

```
Subtract parent Amount from line item total:
{OpportunityLineItems | calc:'TotalPrice - $1':Amount | toFixed:2}

Use two parent fields:
{OpportunityLineItems | calc:'TotalPrice - $1 + $2':Amount:Discount__c | toFixed:2}
```

How \$N works: `$1` refers to the first value after the expression string, `$2` the second, and so on. These values are resolved from the parent Salesforce record by the template engine.

Length — Count Records or Characters

Use the `length` keyword to get the count of child records in an array or the character count of a text field.

```
Number of line items:
{OpportunityLineItems | calc:'length'}

Length of a text field:
{Name | calc:'length'}

Arithmetic with length:
{OpportunityLineItems | calc:'length * 2'}
```

Conditions — Show / Hide Content

Use comparison and logical operators inside `calc` with the `{#...}{/}` conditional block syntax to show or hide sections of your document based on calculated values.

Comparison operators: `>`, `<`, `>=`, `<=`, `==`, `!=`

Logical operators: `&&` (AND), `||` (OR), `!` (NOT)

Show content only if line items exist:

```
{#OpportunityLineItems | calc:'length > 0'}  
  This opportunity has {OpportunityLineItems | calc:'length'} line items.  
{/}
```

Show content if more than 5 items:

```
{#OpportunityLineItems | calc:'length > 5'}  
  Large order – qualifies for bulk discount.  
{/}
```

Combined condition (AND):

```
{#OpportunityLineItems | calc:'length > 0 && length < 100'}  
  Standard processing applies.  
{/}
```

Compare length against a parent field value:

```
{#OpportunityLineItems | calc:'length > $1':MinItems__c}  
  Minimum item requirement met.  
{/}
```

All Supported Operators

Operator	Description	Example
<code>+</code>	Addition	<code>calc:'TotalPrice + 100'</code>
<code>-</code>	Subtraction	<code>calc:'TotalPrice - Discount'</code>
<code>*</code>	Multiplication	<code>calc:'UnitPrice * Quantity'</code>
<code>/</code>	Division (0 returns 0)	<code>calc:'TotalPrice / Quantity'</code>
<code>%</code>	Modulo (remainder)	<code>calc:'Quantity % 3'</code>
<code>**</code> or <code>^</code>	Exponentiation (power)	<code>calc:'TotalPrice ** 2'</code>
<code>N%</code>	Percentage of left operand	<code>calc:'TotalPrice - 10%'</code>
<code>-</code> (unary)	Negation	<code>calc:'-TotalPrice'</code>
<code>()</code>	Grouping	<code>calc:'(A + B) * C'</code>
<code>></code> <code><</code> <code>>=</code> <code><=</code>	Comparison	<code>calc:'length > 0'</code>
<code>==</code> <code>!=</code>	Equality / Inequality	<code>calc:'length == 5'</code>
<code>&&</code>	Logical AND	<code>calc:'length > 0 && length < 100'</code>
<code> </code>	Logical OR	<code>calc:'length == 0 length > 10'</code>
<code>!</code>	Logical NOT	<code>calc:'!(length > 5)'</code>

Complete Examples

Use Case	Tag
Sum a field	<code>{OpportunityLineItems calc:'TotalPrice'}</code>
Sum with formatting	<code>{OpportunityLineItems calc:'TotalPrice' toFixed:2}</code>
Subtract 10% discount	<code>{OpportunityLineItems calc:'TotalPrice - 10%' toFixed:2}</code>
Add 15% markup	<code>{OpportunityLineItems calc:'TotalPrice + 15%' toFixed:2}</code>
Multiply two field sums	<code>{OpportunityLineItems calc:'UnitPrice * Quantity' toFixed:2}</code>
Compound expression	<code>{OpportunityLineItems calc:'(UnitPrice * Quantity) / 100' toFixed:2}</code>
Use parent record value	<code>{OpportunityLineItems calc:'TotalPrice - \$1':Amount toFixed:2}</code>
Count child records	<code>{OpportunityLineItems calc:'length'}</code>
Count string characters	<code>{Name calc:'length'}</code>
Condition: has items	<code>{#OpportunityLineItems calc:'length > 0'}...{/}</code>
Condition: compare to value	<code>{#OpportunityLineItems calc:'length > \$1':MinItems__c}...{/}</code>
Condition: AND logic	<code>{#OpportunityLineItems calc:'length > 0 && length < 50'}...{/}</code>
Negation	<code>{OpportunityLineItems calc:'-TotalPrice'}</code>
Exponentiation	<code>{OpportunityLineItems calc:'TotalPrice ** 2'}</code>

1

Click in your Word document where you want the calculated value or conditional block.

2

Type the tag directly — for example: `{OpportunityLineItems | calc:'TotalPrice - 10%' | toFixed:2}`

3 Replace `OpportunityLineItems` with your child relationship name.

4 Replace field names in the expression with exact Salesforce API field names (case-sensitive).

5 To use a parent record field, add it after the expression string: `| calc:'TotalPrice - $1':Amount`

6 Add formatting if needed: `| toFixed:2` or `| format:2:'10,000,000.5': '$'`

7 Go to the **Preview** tab to verify the result.

Case-Sensitive Field Names: Field names inside `calc` expressions must match the Salesforce API name exactly (e.g., `TotalPrice` not `totalprice`). Check in Salesforce Setup → Object Manager if unsure.

Division by Zero: Dividing by zero returns `0` instead of an error. The same applies to modulo by zero.

7.6 Filtering Loop Results with `filter`

What it does: Narrows a child-relationship loop down to only the records that match a SOQL-style WHERE condition. Use it when you do not want every related record to render — e.g., show only open opportunities, only high-priority cases, or only PDF attachments.

Why `filter` is special: it lets you write a real Salesforce WHERE clause directly inside the tag. If you are iterating a child relationship and need to exclude some records, `filter` is almost always the right choice — far more readable than building conditions inside the loop body.

Quote convention. The `filter` operator wraps its condition in **double quotes** (`filter:"..."`) so the inner SOQL string literals can stay as single quotes. Every other operator (`limit`, `orderBy`, `formatDate`, `format`, `toFixed`, `sumBy`, `calc`, `Img`, `QRCode`, `Barcode`, `splitTable`) uses **single quotes** for string arguments.

Example: `filter:"StageName!='Closed Won' AND StageName!='Closed Lost'"` — outer double quotes wrap the whole condition, inner single quotes wrap each value.

Operator ordering. When chaining operators with `|`, **filter must come first**. Place every other operator (`orderBy`, `limit`, etc.) *after* the filter so they operate on the already-filtered subset.

✓ Correct: `{#Opportunities | filter:"Amount>10000" | orderBy:'CloseDate':'desc' | limit:5}`

✗ Wrong: `{#Opportunities | orderBy:'CloseDate':'desc' | filter:"Amount>10000" | limit:5}`

Supported Operators

Operator	Meaning	Example
<code>= / !=</code>	Equal / not equal	<code>StageName='Closed Won'</code>
<code>> / < / >= / <=</code>	Numeric / date comparison	<code>Amount>10000</code>
<code>LIKE</code>	Pattern match — <code>%</code> = any chars, <code>_</code> = single char	<code>Product2.Name LIKE 's%'</code>
<code>IN (...)</code> / <code>NOT IN (...)</code>	Membership in a list	<code>Priority IN ('High','Critical')</code>
<code>AND / OR / NOT</code>	Boolean combinators (uppercase)	<code>Status!='Closed' AND Priority='High'</code>

Examples

Example 1 — Exclude closed opportunities:

```
{#Opportunities | filter:"StageName!='Closed Won' AND StageName!='Closed Lost'"}
{Name} - {StageName} - ${Amount | toFixed:2}
{/}
```

Example 2 — New high-priority cases only:

```
{#Cases | filter:"Status='New' AND Priority IN ('High','Critical')}
{CaseNumber}: {Subject}
{/}
```

Example 3 — PDF files only (via ContentDocumentLinks):

```
{#ContentDocumentLinks | filter:"ContentDocument.FileType='PDF'"}
{ContentDocument.Title}
{/}
```

Example 4 — Negation with NOT :

```
{#Contacts | filter:"NOT (Title LIKE 'Junior%')"}
{Name} – {Title}
{/}
```

Example 5 — Filter first, then sort & limit (the recommended chain order):

```
{#Opportunities | filter:"Amount>10000" | orderBy:'CloseDate':'desc' | limit:5}
{Name} – {CloseDate | formatDate:'MM/DD/YYYY'} – ${Amount | toFixed:2}
{/}
```

Dotted field paths are supported. You can reference fields on parent records of the child item — for example `ContentDocument.FileType` on a `ContentDocumentLink`, or `Account.Industry` on a `Contact`. The system automatically adds these fields to the underlying Salesforce query so they are available at evaluation time.

7.7 Tables — Advanced Layouts & Helpers

The standard **Table** layout (Section 7.1) covers most needs, but three add-on capabilities make table-based templates much more powerful: a **vertical / transposed** table layout, an auto-incrementing **serial-number** token, and an automatic **wide-table splitter**. All three are designed to drop into any Word table you already have, with no UI button required.

Vertical Tables with `:vt`

What it does: Transposes a child-relationship loop into a **two-column label/value table**. Field labels go down the left column; the looped record's field values fill the right column. Useful for one-record-per-page summary sheets, contract detail pages, opportunity briefs, and any document where a row-per-field layout is more readable than a row-per-record table.

This is a special directive, not a pipe modifier. Vertical-table loops use the `:vt` prefix on both opening and closing tags — `{:vt#RelationshipName} ... {:vt/}`. There is no `| vertical` pipe modifier.

How to Lay Out the Word Table

- 1 In your Word document, insert a normal **two-column table** with one row per field you want to display.
- 2 In the **left column**, type the human-readable label for each field (e.g., "Opportunity Name", "Stage", "Close Date").
- 3 In the **value cell of the first row** (top-right), place the opening tag: `{:vt#RelationshipName}` followed immediately by the first field (e.g., `{Name}`).
- 4 In every middle row's right cell, place just the field tag (e.g., `{StageName}` , `{CloseDate}` , etc.).
- 5 In the **value cell of the last row** (bottom-right), place the last field tag followed immediately by the closing tag `{:vt/}` .
- 6 Optionally add a filter on the opening tag: `{:vt#Opportunities | filter:"..."}` .

Examples

Example 1 — Opportunity brief (two-column Word table, one record per page):

```
+-----+-----+
| Opportunity Name | {:vt#Opportunities | filter:"StageName!='Closed Won'"}{Name} |
| Stage           | {StageName} |
| Close Date     | {CloseDate | formatDate:'MM/DD/YYYY'} |
| Amount         | ${Amount | format:2:'10,000,000.5': '$'} |
| Probability     | {Probability}% |
| Owner          | {Owner.Name} |
| Type           | {Type} |
| Lead Source    | {LeadSource} |
| Next Step      | {NextStep}{:vt/} |
+-----+-----+
```

Example 2 — Contact card (compact 4-row layout):

```
+-----+-----+
| Contact Name | {:vt#Contacts}{Name} |
| Title       | {Title} |
| Email       | {Email} |
| Phone       | {Phone}{:vt/} |
+-----+-----+
```

Example 3 — Open case summary with filter:

```
+-----+-----+
| Case Number | { :vt#Cases | filter:"Status!='Closed'" } {CaseNumber} |
| Subject     | {Subject} |
| Priority    | {Priority} |
| Status     | {Status} |
| Created    | {CreatedDate | formatDate:'MM/DD/YYYY hh:mm A'} |
| Owner      | {Owner.Name} { :vt/ } |
+-----+-----+
```

One record per layout instance. The vertical table repeats the entire 2-column structure once per matching record. If you filter down to a single record, you get one card; if the filter returns five records, you get five back-to-back cards in the rendered document.

Serial Numbers with `{ $iterator }`

What it does: Inside any loop — regular `{ #... } { / }`, vertical `{ :vt#... } { :vt/ }`, or grid `{ :#grid ... } { :/grid }` — the special token `{ $iterator }` renders the current row's auto-incrementing number. Drop it in the first column of any table to produce an "S.No." column without having to manage the count yourself.

Why use it: any table with a serial-number column (line items, asset lists, attendee rosters, search results) becomes a one-line addition. The numbering automatically reflects the loop's filtered, sorted, and limited result — so you always get a clean 1, 2, 3, ... sequence regardless of how the underlying records were narrowed down.

Examples**Example 1 — Simple S.No. column in a product line-item table:**

```
+-----+-----+-----+-----+
| S.No. | Product Name      | Quantity | Total      |
+-----+-----+-----+-----+
| { #OpportunityLineItems } { $iterator } | {Product2.Name} | {Quantity} | $ {TotalPrice | toFixed:
```

Example 2 — S.No. with filter, sort & limit (the most common real-world pattern — filter narrows the list, iterator numbers what is left):

```
+-----+-----+-----+-----+-----+
| S.No. | Product Name | List Price | Quantity | Total Price |
+-----+-----+-----+-----+-----+
| {#OpportunityLineItems | orderBy:'Quantity':'desc' | filter:"Product2.Name LIKE 's%'" | lim
```

The same closing could be written in full as `{/OpportunityLineItems | orderBy:'Quantity':'desc' | filter:"Product2.Name LIKE 's%'" | limit:5}` — both forms render identically (see *Closing Tag Forms* at the end of this section, Section 7.7).

Example 3 — Inside a grid, where every tile carries its own number:

```
{:#grid Contacts | size:6:3}
#{${iterator}} – {Name}
{Email}
{:/grid}
```

Example 4 — Inside a vertical table (rare, but valid — useful when looping cards and you want each card numbered):

```
+-----+-----+
| Card # | {:vt#Opportunities | limit:3}#{${iterator}} |
| Name   | {Name} |
| Stage  | {StageName} |
| Amount | ${Amount | toFixed:2}{:vt/} |
+-----+-----+
```

The iterator is per-loop and 1-based. Numbering starts at 1 for each loop. Nested loops reset their own counter — `{${iterator}}` always refers to the *innermost* loop. To show the outer-loop index inside a nested loop, capture it in an outer-cell field before entering the inner loop.

Outside a loop, `{${iterator}}` renders nothing. Like all loop content, the iterator token only has meaning inside a `{#...}{/}` / `{:vt#...}{:vt/}` / `{:#grid ...}{:/grid}` block.

Splitting Wide Tables with `splitTable`

What it does: Solves the "my table is wider than the page" problem. Place a single marker tag on its own line immediately above any table; after the document renders, that table is automatically sliced into multiple narrower tables stacked vertically. The marker line itself is removed from the output.

When to use it: any time a generated table has too many columns to fit the page. Common cases — line items with 10+ tracking fields, comparison matrices across many products, financial schedules with monthly columns, audit logs with many metadata fields.

Syntax

```
{"" | splitTable:maxCols:keyCols:repeatKey}
```

Parameter	Type	Meaning
maxCols	positive integer	Maximum columns per chunk after splitting
keyCols	non-negative integer	Number of leftmost columns to treat as "keys" (identifiers like S.No., Name, ID)
repeatKey	true / false	If true, the key columns appear in every chunk for context. If false, the table is simply sliced into contiguous chunks.

Placement rule. The marker tag must be on its **own paragraph (its own line)**, immediately above the table you want to split. The paragraph is consumed and removed from the rendered output, so no blank line is left behind.

Examples

Example 1 — Simple slice (the user's reference example):

```
{"" | splitTable:5:1:false}
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Co11 | Co12 | Co13 | Co14 | Co15 | Co16 | Co17 | Co18 | Co19 | C10  | C11  |
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ... |
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

A 12-column table is split into three stacked tables: columns [0-4], [5-9], [10-11]. No column is repeated.

Example 2 — Repeat the ID column in every chunk for readability:

```
{"" | splitTable:6:1:true}
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Col1 | Col2 | Col3 | Col4 | Col5 | Col6 | Col7 | Col8 | Col9 | C10 | C11 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

The same 12-column table is split into chunks `[ID, Col1–Col5]`, `[ID, Col6–Col10]`, `[ID, Col11]`. The first column (ID) repeats in every chunk so each row is still identifiable.

Example 3 — Composite key (first two columns repeat):

```
{"" | splitTable:8:2:true}
+-----+-----+-----+-----+-----+-----+-----+-----+
| Name | Region | A | B | C | D | E | F | G | H |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

A 10-column table is split into chunks `[Name, Region, A–F]`, `[Name, Region, G, H]`. The two identifier columns repeat in every chunk.

Example 4 — On top of a looped line-item table (combine with `filter` / `orderBy` / `$iterator`):

```
{"" | splitTable:5:2:true}
+-----+-----+-----+-----+-----+-----+-----+-----+
| S.No. | Product Name | List Price | Quantity | Discount | Tax | ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
| {#OpportunityLineItems | filter:"Quantity>0"}{$iterator} | {Product2.Name} | {ListPrice} |
```

Constraints to remember:

- `maxCols` must be a positive integer.
- `keyCols` must be non-negative.
- When `repeatKey=true`, `maxCols` must be strictly greater than `keyCols` (otherwise every chunk would only contain key columns).
- Tables with merged cells (`gridSpan` or `vMerge`) are skipped — the engine cannot reliably re-split them.

Closing Tag Forms — Table, Loop & Grid

All three loop layouts in the add-in — the regular **Loop** (`{#...}{/}`), the **Vertical Table** (`{:vt#...}{:vt/}` , above), and the **Grid** (`{:#grid ...}{:/grid}` , see Section 8.1) — follow the same closing-tag rule: **each loop layout has two valid closing-**

tag forms, and either form works regardless of whatever pipe modifiers (`filter` , `orderBy` , `limit` , etc.) you put on the opening tag.

Two valid closing forms for every loop layout

Layout	Shorthand closing (always valid)	Mirrored closing (also valid)
Loop (regular)	<code>{/}</code>	<code>{/RelationshipName filter:"..." orderBy:'F':'desc' limit:N}</code> — repeat the opening's text after the <code>/</code>
Vertical Table	<code>{:vt/}</code>	(shorthand only — vertical loops always close with <code>{:vt/}</code>)
Grid	<code>{:/grid}</code>	(shorthand only — grid loops always close with <code>{:/grid}</code>)

Concrete example for a regular Loop — these two render identically:

```
{#OpportunityLineItems | orderBy:'Quantity':'desc' | filter:"Product2.Name LIKE 's%'" |
{Product2.Name} – {Quantity}
{/}
```

```
{#OpportunityLineItems | orderBy:'Quantity':'desc' | filter:"Product2.Name LIKE 's%'" |
{Product2.Name} – {Quantity}
{/OpportunityLineItems | orderBy:'Quantity':'desc' | filter:"Product2.Name LIKE 's%'" |
```

Pick whichever is easier to read — the shorthand is the simplest and works in every case.

8. Action: Image

What it does: Inserts a dynamic image from a Salesforce record into your document. The image is fetched and embedded automatically when the document is generated.

Common uses:

- Company or account logo in the document header
- Product photos in a catalog or proposal

- User profile photos in contact sheets
- Signature images in contracts

Step-by-Step: Inserting an Image

1 Select your **Primary Object**.

2 From the **Action** dropdown, choose **Image**.

3 Choose the **Source Type**:

- **URL** — a publicly accessible image URL.
- **Salesforce File Id** — a `ContentDocumentId` or `ContentVersionId` of a file stored in Salesforce Files.

4 Enter the value based on the source type you chose:

- For **URL** — paste the full image URL (e.g., `https://example.com/logo.png`).
- For **Salesforce File Id** — paste the Content Document Id (15/18-char Id starting with `069`) or Content Version Id (starting with `068`).

5 Set the **Width** and **Height** (in pixels).

6 Click **Insert** or **Copy**.

360 SmartDocs

←

1 CREATE

2 PREVIEW

3 LIVE

Search Object

Opportunity

Select Action

Image

Image Setup

Source Type

Salesforce File Id

Resource Id

069dM00000OdBPBQA3

Width

300

Height

300

Continue to Preview

Figure 20: Image action — choose source type, enter URL or File Id, then set size

Image Tag Syntax

Image tags use the `%Img(...)` function with three arguments: the source (URL or File Id, in quotes — or a Salesforce field name unquoted), width, and height in pixels.

```
{%Img('Salesforce File Id', width, height)}
{%Img('https://your-url/image.png', width, height)}
{%Img(FieldApiName__c, width, height)}
```

Examples

Tag	What it inserts
<code>{%Img('069dM000000dBPBQA3', 300, 300)}</code>	File from Salesforce by ContentDocumentId, sized 300×300px
<code>{%Img('068dM00000Jd3vxQAB', 300, 300)}</code>	File from Salesforce by ContentVersionId, sized 300×300px
<code>{%Img('https://monochrome-watches.com/app/uploads/2025/12/2026-Land-Rover-Defender-Dakar-DX7-R-Featured-1536x1024.jpg', 300, 300)}</code>	Image fetched from a public URL, sized 300×300px
<code>{%Img(ImageId__c, 300, 300)}</code>	Resolves the field value at runtime (the field should contain a URL or a Salesforce File Id), sized 300×300px

Quoted vs unquoted source: Wrap a literal URL or File Id in single quotes. Use a bare field API name (no quotes) when the source value lives on the record and should be resolved at generation time.

Real-Life Scenario: Using a Merge Field as the Image Source

If the image source (a URL or a Salesforce File Id) is already stored on the record in a field, pass the **field API name unquoted** as the first argument. The system reads the field value at generation time and uses it as the image source.

```
{%Img(FieldAPIName, 300, 300)}
```

Example — the `ImageId__c` field on the record holds an image URL or a Salesforce File Id:

```
{%Img(ImageId__c, 300, 300)}
```

This is the most common pattern for record-driven templates: each generated document automatically pulls the right image (logo, signature, product photo, etc.) from the record being merged.

Real-Life Scenario: Professional Document Header

Place this at the very top of your Word template to create a branded header on every generated document:

```
{%Img(Account.LogoUrl__c, 200, 60)}
{Organization.Name}
Prepared by: {User.FirstName} {User.LastName} | {User.Title}
Date: {TODAY | formatDate:'MMMM DD, YYYY'}
```

Supported image formats: PNG, JPEG/JPG, SVG

8.1 Looping over Files (ContentDocument) for Images

What it does: Pair the `{%Img(...)}` action with a loop over the `ContentDocumentLinks` relationship to render **every image attached to the current record** automatically — no need to know the file Ids in advance. Each iteration of the loop exposes a `ContentDocument` sub-record; you reference its fields with the `ContentDocument.` prefix.

Common uses: photo galleries on inspection reports, before-and-after image sets on cases, product image grids on quotes, signature pages, evidence pages on insurance claims.

Always include a filetype filter. Loop over `ContentDocumentLinks` with `filter:"ContentDocument.FileType=..."` so only image filetypes are rendered. Without the filter, the engine will try to embed every file (PDFs, Word docs, spreadsheets, etc.) as an image and fail silently — your document will show empty image boxes.

Recommended Pattern — Grid Layout

The grid directive `{:#grid ...}{:/grid}` is the cleanest layout for image galleries because you can specify rows × columns directly. The relationship to loop is `ContentDocumentLinks`, and the image source for `{%Img(...)}` is

