

SLATE PRESENTS WEBINAR

Stop Copy Pasting. Start Relating.

Using Related Records as a communication logic layer

David Dysart
Stevens Institute of Technology

The message works when the system understands the relationship.



INSTITUTIONAL SETTING

Stevens Institute of Technology

- Private research university in Hoboken, New Jersey
- Roughly 4,000 undergraduate students and 4,000 graduate students

PRESENTER

David Dysart

- Associate Director of Data Analytics, Enrollment Management
- Stevens Institute of Technology
- ddysart@stevens.edu



INSTITUTIONAL SETTING

Why this theme works

A novel shared activity can be more relational than a generic nice moment.

Systems communicate better when they understand who is connected to whom.

- Presentation goal: reusable Slate patterns, shown through a human frame
- Relationships are built through context, attention, and shared experience.

A relationship is specific. A good Slate build should be specific too.



Today's Webinar

A technical story, carried by one relationship idea.

This is not three separate tricks. It is one architecture used three ways.

- A clear mental model for Related Records
- Three working patterns for tour guides, school visits, and campus events
- A practical method for joining data, selecting content, testing branches, and building safe fallbacks

By the end of this session, you should be able to identify one relationship in your process and turn it into reusable communication logic.



HOW TO LISTEN TO THIS SESSION

Follow the relationship first

For every example, ask:

- What two records are connected?
- Where does that connection live?
- What information does the message need?
- What decision should Slate make?
- What happens when the data is missing?

Relationship. Retrieval. Decision. Message.



CORE PROBLEM

The copy paste trap

Most communication systems fail at the relationship layer.

- Template sprawl makes event edits risky
- Staff voice gets flattened into generic language
- Relationship context stays trapped in fields or out-of-system
- Manual follow up depends on memory and time

The system has data. Messaging fails when it does not understand the relationship.



CORE IDEA

The relationship logic pattern

Related Records hold context. Configurable Joins retrieve it. Merge logic speaks with it.

- Relationship context
- Configurable Join
- Exports
- Liquid logic
- Rendered message

RARE: Related Automation and Relationship Engine.



ONE PATTERN

Three proofs of the same pattern

Change the relationship. Keep the architecture.

- Tour Guides: student voice
- High School Visits: counselor voice
- Campus Events: operations voice

The project changes. The relationship logic stays the same.



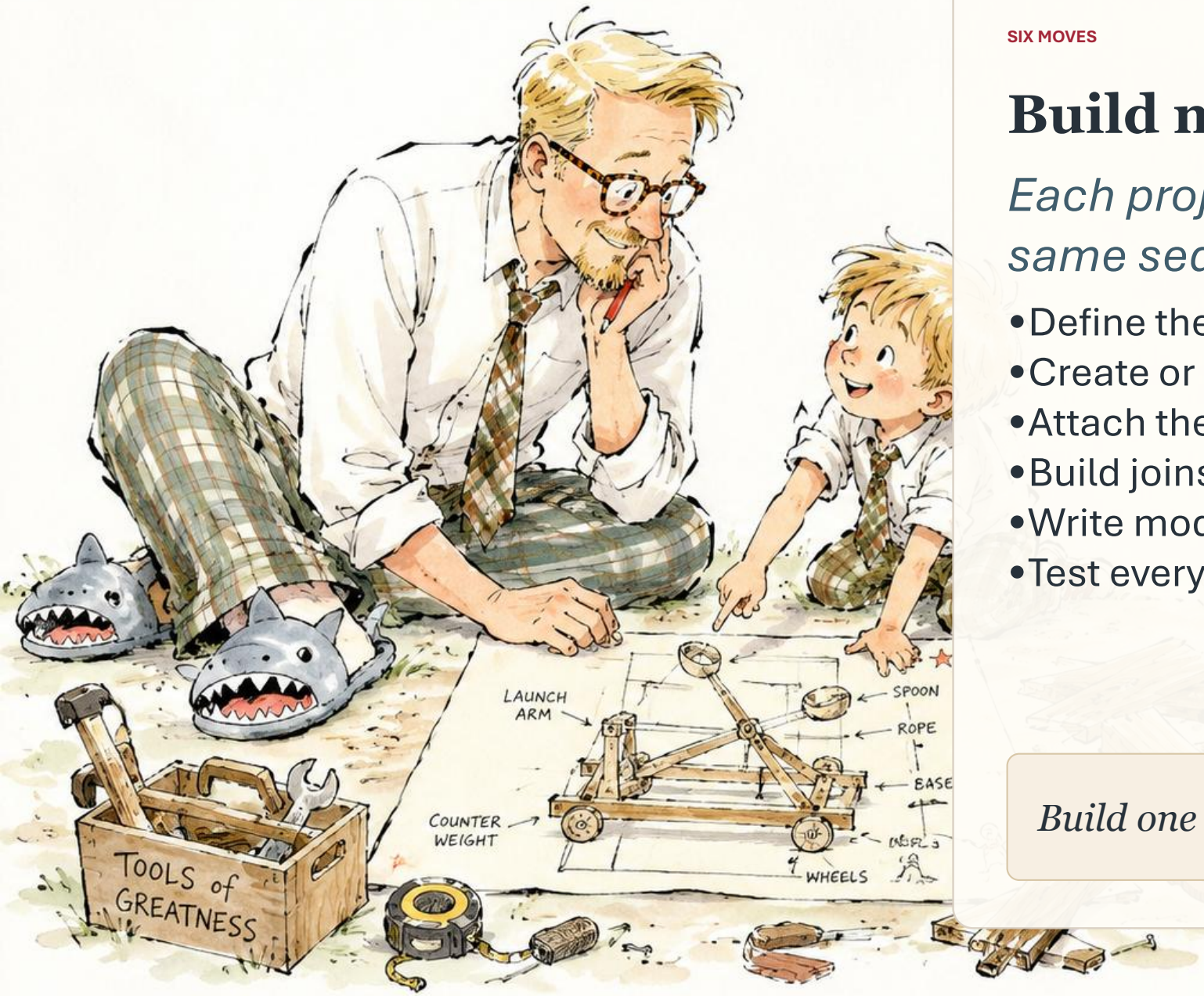
TECHNICAL SKELETON

Shared architecture

The same structure sits under all three builds.

- Event or Form Response
- Related Record context
- Configurable Join path and exports
- Content decision
- Message output

Relating starts with the context, not the template.



SIX MOVES

Build map

Each project can be explained with the same sequence.

- Define the relationship
- Create or reuse the dataset
- Attach the record
- Build joins and exports
- Write modular copy
- Test every branch

Build one clear pattern, then reuse it.



TOOL CHOICE

When Related Records Are the Right Tool

Use when the relationship changes the message.

Use Related Records when

One event points to a person, office, or detail
The sender voice depends on a relationship
The same template needs different instructions

Use plain fields when

The value belongs directly on the event
The detail is stable for every attendee
Future reuse is unlikely

Use separate templates when

The audience/event is meaningfully different
The timing is unrelated
The owner needs full control

RULE OF THUMB

Does another relationship need to shape this message?



END TO END PATH

Follow One Relationship Through Slate

One visitor, one event, one related guide, one rendered message.

Visitor registers
The form response creates the source row.

Visitor attends
Attendance qualifies the recipient.

Event has guide
The relationship is attached to the event.

Join finds guide
Configurable Join retrieves the right row.

Field joins pull story
Major, advice, interests, and experience become exports.

Exports compare
Flags turn raw values into simple decisions.

Liquid chooses copy
The right paragraph renders for that student.

Email sends
The message sounds specific because the relationship was found.

Process Improvement

The Related Record process adds personalization at scale



STUDENT VOICE

Project 1: Tour Guides

The easiest way to see the value is a message from the person the visitor actually met.

- A tour guide dataset stores voice and experience
- The guide is attached to the event as a Related Record
- The email compares guide data to visitor data

The guide is the relationship. The email is the outcome.

BUILD CHOICE

Start with a student worker dataset

Capture the fields that the email will use later.

- Use fields for topics that need matching
- Collect information with Forms
- Ask student workers for reusable first person copy
- Store experiences that can become message branches

Scope ▲	Category/Tab	Name	Prompt	ID
Student Workers		Club Sports	stu_club_sports	stu_club_sports
Student Workers		Advice to Prospective Students		stu_advice
Student Workers		Scholar Program	special_program	stu_special_program
Student Workers		First Name		stu_first_name



The dataset is the voice bank.

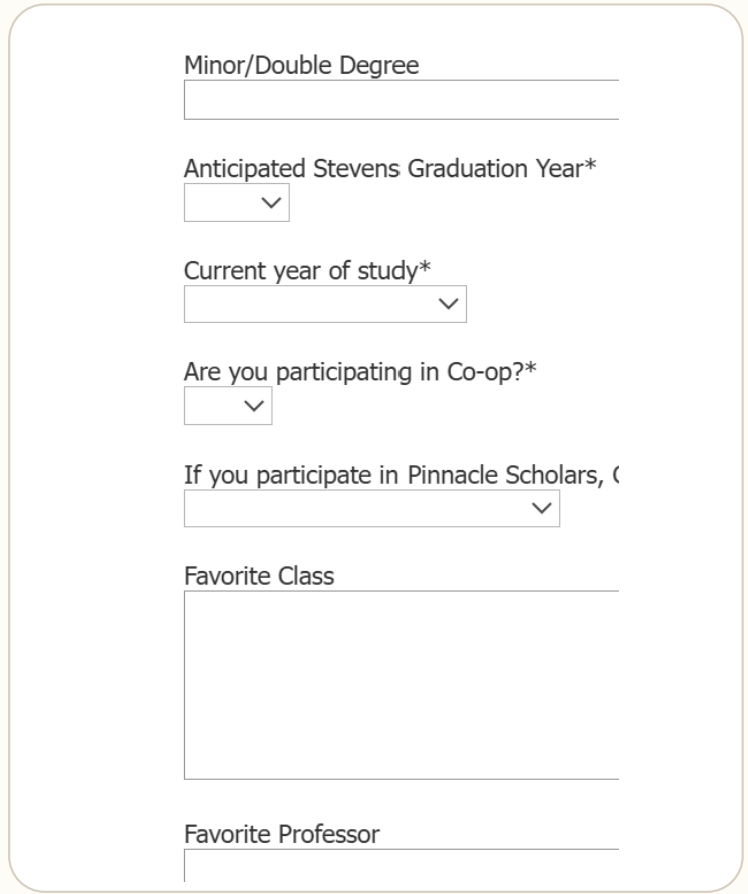
RELATED RECORDS

Collect Tour Guide information

A well-populated tour guide dataset gives more opportunities for meaningful connections and relationship building

- Codify open text and selectable data
- Embed merge fields and text directly into emails
- Write logic and match against data in the exports to dynamically display relevant text





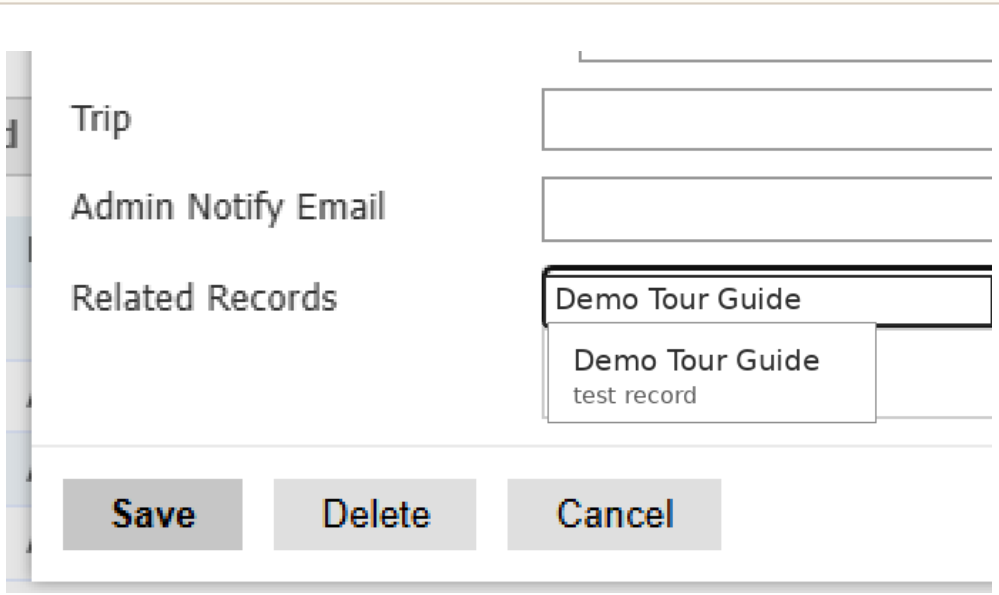
This is the moment the relationship becomes possible in Slate

RELATED RECORDS

Attach the guide to the event

The Related Record gives the message a person to relate through.

- Add the guide as a Related Record on the event
- Use the event relationship to shape later communications





This is the moment the relationship becomes available to Slate.



SEND LOGIC

Recipient logic starts with attendance

Then the attendee/guide relationship controls who the message comes from and what it says.

- The event relationship is required before sending
- The attendee record supplies student data for comparisons
- The Related Record supplies the intersection of the relationship

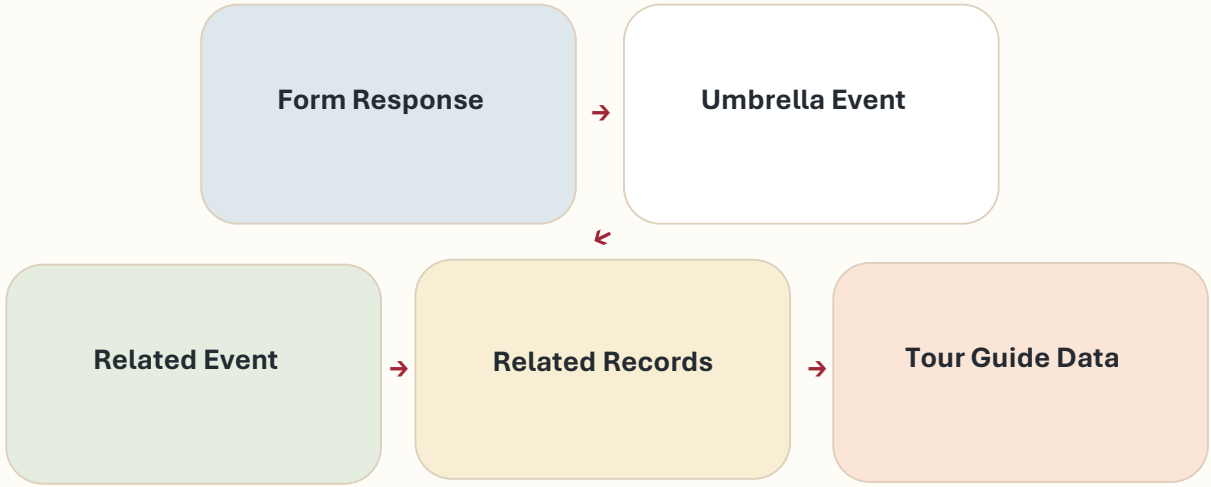
Attendance starts the send. Relationship context shapes it.



CONFIGURABLE JOIN

Step One. The Join Finds the Person

The first job is proving Slate reached the correct Related Record row.



BUILDER CHECK

First prove Slate found the right row. Then ask the row for story fields.

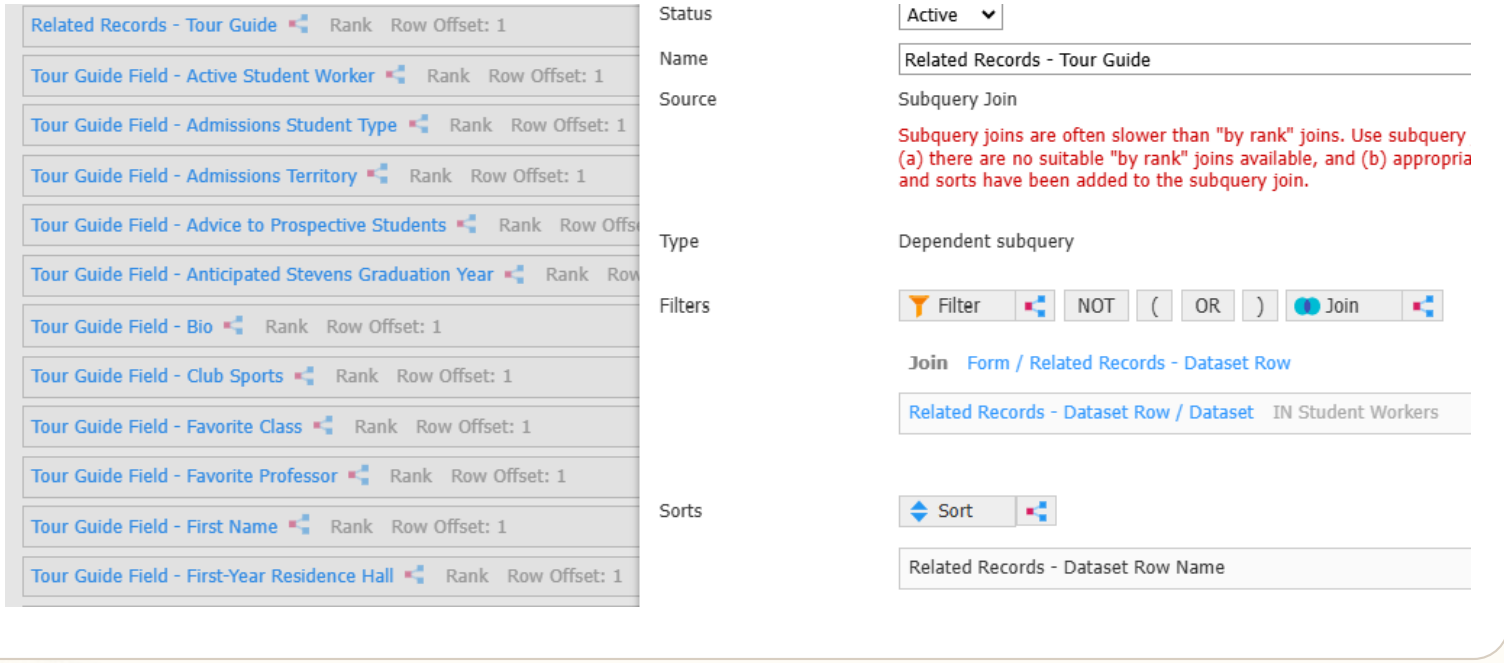


CONFIGURABLE JOIN PATH

Join from response to event to guide

The first join retrieves the correct related person.

- Start from the Form Response
- Join to the Form/Event that owns the event
- Join to the related event that keeps the tour guide
- Join to Related Records for the guide dataset
- Filter to the right dataset when several related records exist



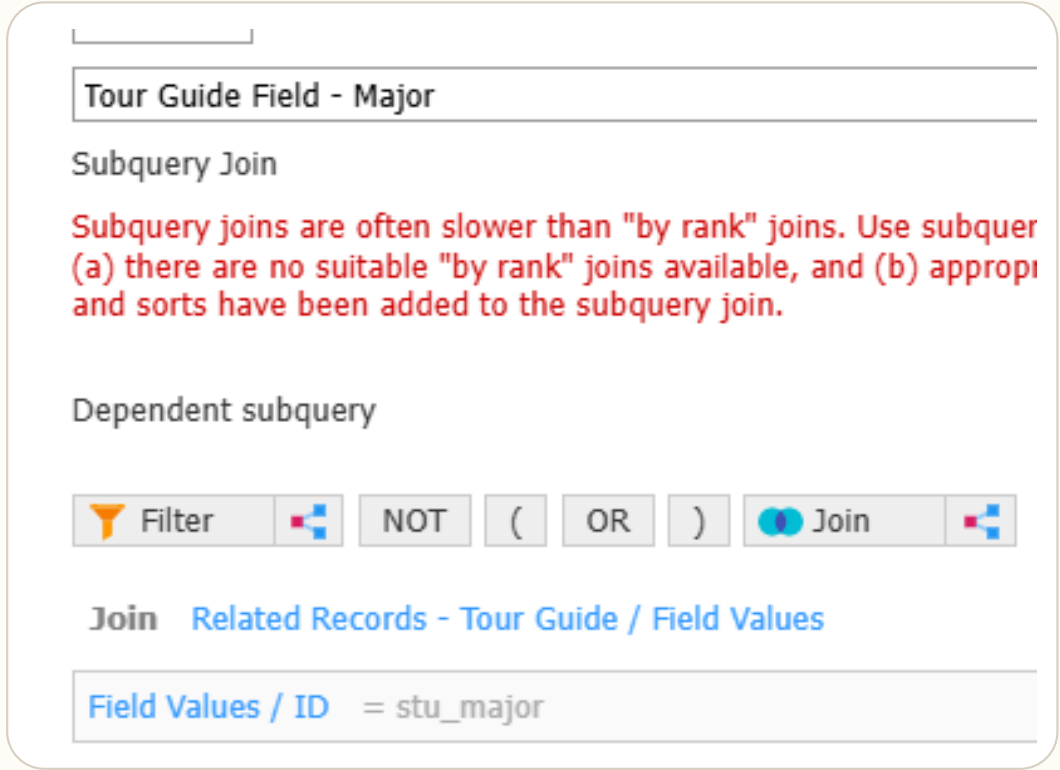


BUILDER REALITY

Every usable field gets its own join

Tedious once. Powerful every time after that.

- The related row alone does not expose every field
- Join to field values for each data point you need
- Name each join with the copy decision in mind
- Build once and suitcase later



The field joins are the price of reusable personalization.

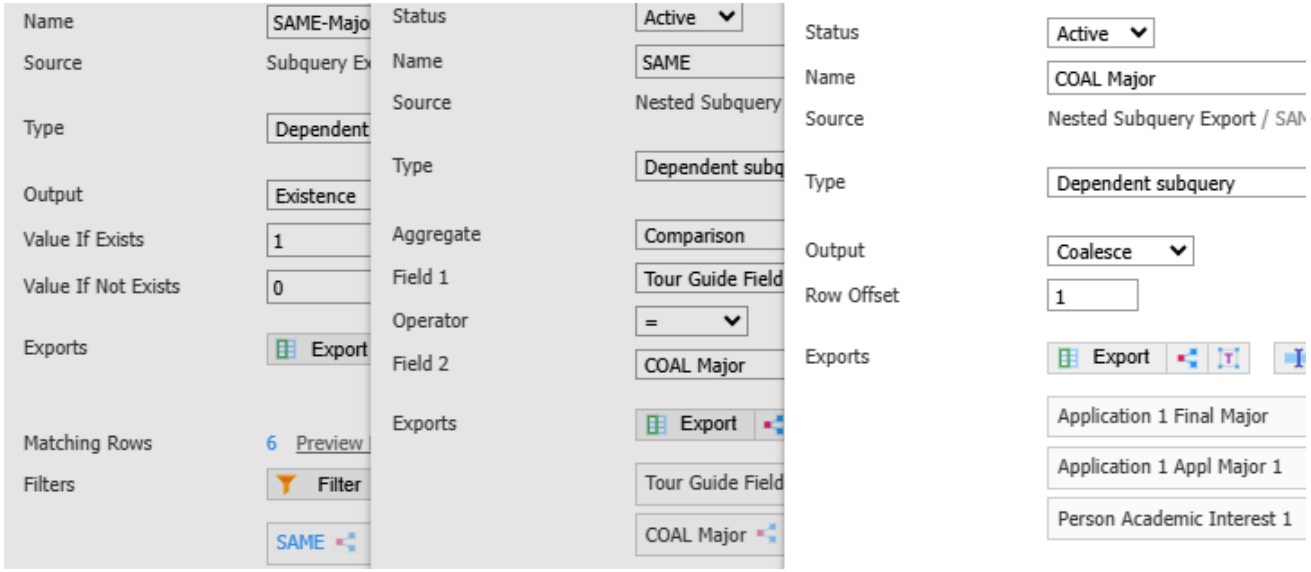


BUILDER REALITY

Comparing Attendee/Guide Data

Build relationship while driving logic and text

- Nest Comparison Filters inside Existence Export
- If Attendee major = Guide major, Slate exports a '1' else '0'
- Coalesce export the first field with data present
- Coalesce major/interest fields so the most relevant field is compared



Code Existence exports consistently across Slate (1/0, Yes/No, Y/N, etc.). Consistency > what you use



FIELD VALUE RETRIEVAL

Step Two. Field Joins Find the Story

The related row tells you who. Field joins tell you what they can say.

Guide major

Used for major/same school copy.

Guide residence

For campus life branches & narrative.

Guide advice

Used when match signals are weak.

Guide activity

Used for involvement copy.

NAMING RULE

Name joins by the copy decision they support. Export raw values first, flags second.



MERGE PREP

Export comparisons and simplify

The email should be simple decisions even when logic is complex.

- Create flags for shared major, program, geography, or activity
- Use ones and zeros consistently across projects
- Coalesce related topics into one best available talking point
- Protect the copy when some data is missing

EXISTS-Stevens-Summer Existence Exists: 1 Not
SAME-Majors Existence Exists: 1 Not Exists: 0
COAL-Varsity-Org-Scholar-CoOp-Club-Class Coalesce
Applied Existence Exists: 1 Not Exists: 0
~~~ Tour Guide Data
Related Records - Tour Guide Data
Related Records - Tour Guide Name
Tour Guide Field - Varsity Sports Prompt Value
Tour Guide Field - Student Organizations Prompt Value
Tour Guide Field - Stevens Degree Progress Prompt Value
Tour Guide Field - Scholar Program Prompt Value
Tour Guide Field - Residence Hall Prompt Value
Tour Guide Field - Pronouns Prompt Value
Tour Guide Field - Profile Photo Prompt Value
Tour Guide Field - Preferred Prompt Value

Status: Active
Name: COAL-Varsity-Org-Scholar-CoOp-Club-Class
Source: Subquery Export
Type: Dependent subquery
Output: Coalesce
Row Offset: 1
Exports: Export

Tour Guide Field - Varsity Sports Prompt Value
Tour Guide Field - Student Organizations Prompt Value
Tour Guide Field - Scholar Program Prompt Value
Tour Guide Field - Participating in Co-Op Prompt Value
Tour Guide Field - Club Sports Prompt Value
class
~~~



LIQUID LOGIC

Conditional copy produces different messages

Logic selects the right paragraph for each visitor.

- Shared major gets one paragraph
- Different major gets a collaboration paragraph
- Fallbacks protect the message when data is missing

It was so great meeting you on the tour today. I really enjoyed talking about {{Tour-Guide-Field---Major-Prompt-Value}}. {% if {{SAME-Major…

I remember being in your shoes, trying to picture what studying {{Person-Academic-Interest-1}} here would actually feel like. Spoiler: it's even better than I imagined. The professors are awesome, and there's always something happening in {{COAL-Varsity-Org-Scholar-CoOp-Club-Class}}. {% else %}

Even though I'm majoring in {{Tour-Guide-Field---Major-Prompt-Value}} I get to collaborate with {{Person-Academic-Interest-1}} students all the time. One of the best things about Stevens is how connected everything is.
{% endif %} {% if {{EXISTS-Stev…

The logic is technical. The output should sound like a person.



DECISION LOGIC

The Decision Table Behind the Email

A readable table is often clearer than talking through nested conditions.

Signal available	Slate action	Human result
Same major	Shared major paragraph	Visitor feels recognized
Same geo	Same geo paragraph	Connection without false precision
Shared experience	Guide advice paragraph	Specific memory
Weak match	General guide note	Still warm and useful
Missing guide	Safe fallback or suppress	Avoids an odd email

WRITING RULE

Build the table before the Liquid. The table is your content map.



RENDERED OUTCOME

What the student sees

Same build, different copy paths.

- The visitor sees a normal message
- The guide voice stays specific
- The branch logic stays hidden
- Attendee/Guide Majors match vs. different majors

It was so great meeting you on the tour today. I really enjoyed talking about Cybersecurity.

I remember being in your shoes, trying to picture what studying Cybersecurity here would actually feel like. Spoiler: it's even better than I imagined. The professors are awesome, and there's always something happening in Stevens Trivia Club.

It was so great meeting you on the tour today. I really enjoyed talking about Cybersecurity.

Even though I'm majoring in Cybersecurity I get to collaborate with Economics students all the time. One of the best things about Stevens is how connected everything is.

Anyway, I just wanted to say thanks again for coming. If you have any questions about classes, dorm life (Castle Point Hall is where I live!), or even the best place to get coffee between lectures, hit reply. I'd be happy to help.

Students experience the relationship, not the mechanism.



SEQUENCE DESIGN

The Full Three Email Sequence

The same relationship can support more than one message.

Email 1

Timing: next day/soon after to be fresh
Purpose: guide voice and strongest match
Decision: best shared signal

Email 2

Timing: two to four days later
Purpose: campus life or academic angle
Decision: next strongest useful topic

Email 3

Timing: later in the visit path
Purpose: next best action
Decision: apply, visit again, or ask a question

PAYOFF

The sequence should feel like follow through and avoid repetition.



WHAT CARRIES FORWARD

Tour Guide lessons

The first proof gives us the reusable moves.

- Relationship first: the guide is attached to the event
- Data second: fields become decision points
- Copy third: modular paragraphs still read as one message

Student voice proves the pattern because the relationship is obvious.



COUNSELOR VOICE

Project 2: High School Visits

The system should remember who met whom.

- Standalone Deliver
- Send to the specific school contact
- Use the counselor tied to the visit
- Pull staff voice through Content Blocks

A staff relationship can be automated without flattening the voice.

TARGETING MODEL

High school visit relationship model

The message is sent to the person the counselor actually interacted with.

- School organization
- School contact
- Visit event
- Counselor voice



Joins

Form by Related records — Rank — Row Offset: 1

Organizations

Form by Location Record — Rank — Row Offset: 1

Form by Location Record / User

The contact and visit need to point to the same school.



SAFETY CHECKS

Guardrail the joins

Personalization scales only when wrong sends are hard to make.

- Contact belongs to the same school
- Visit is recent
- Contact is active
- Email address exists

A personalized email to the wrong person is worse than a generic one.



GUARDRAILS

High School Visit Safety Checks

Written checklist helps ensure emails will send smoothly

Check	Why it matters
Contact: same school	Prevents mismatched outreach
Visit is recent	Keeps follow up timely
Contact is active	Protects list quality
Visit user exists	Selects counselor voice
Content Block exists	Keeps the message complete
Email exists	Keeps the send clean

SAFETY PRINCIPLE

A safety checklist helps prevent errant logic, bad sends, and confusion



VOICE STRATEGY

Counselor voice lives in Content Blocks

The build uses the staff-written paragraph based on the visit user.

- The visit user identifies the counselor
- A naming convention selects the right greeting
- Standard content follows after the voice block
- Staff can update their reusable paragraph

{{Organization-Contacts-First-Name}},

{{hs_visit_org_contact_Form-User | snippet:
"counselor_hs_visit_counselor_greeting"}}

I want to provide follow-up information about Stevens that you can share

Store the voice once. Let the context pick it repeatedly.



STAFF VOICE

How the Content Block Code Is Built

A small naming system lets the visit user select the right counselor paragraph.

Block key → **Use case code** → **Staff code** → **Counselor paragraph**

Example

key = counselor_greeting
use_code = hs_visit_org_contact_
visit_user = ddysart
block code = hs_visit_org_contact_ddysart

- Use predictable names
- Keep the block small
- Store staff voice in one editable content block
- Add a default voice for missing codes

CONTENT RULE

Content Block produces personalized, modular content.

STAFF VOICE

How the Content Block Key Is populated

An Export pulls the content block formatting and customization together.

- Establish the general code with static text
- Dynamically pull in the event's User to customize the introduction



Name	hs_visit_org_contact_Form-User
Source	Subquery Export
Type	Dependent subquery
Output	Concatenate
Row Separator	\n = New Line; \t = Tab;
Row Offset	1
Row Limit	
Export Separator	
Exports	Export
	hs_visit_org_contact_
	User Full Name

USE

Add the export name to the content block pulled in the email body

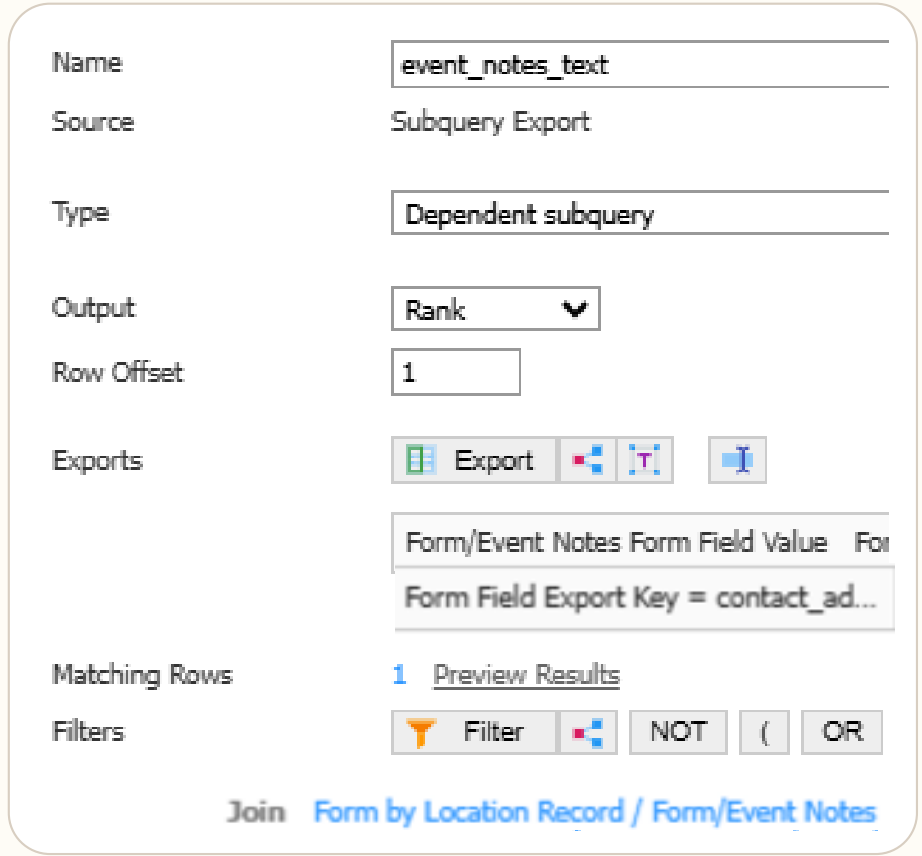


STAFF VOICE

Event Note Context

Counselors can also add event-specific text that is merged into the email

- Counselor adds open text through the event's Event Note
- Export the text value



ALTERNATE

Event Notes can be used as a pilot before a full Related Records process and as even higher touch personalization.



STAFF VOICE

Event Note Context

Counselors can also add event-specific text that is merged into the email

- With the merge field available, add to mailing
- Add Liquid Markup to hide merge field and line breaks when empty

```
{{hs_visit_org_contact_Form-User | snippet:
"counselor_hs_visit_counselor_greeting"}}

{% if {{event_notes_... {{event_notes_text}}

{% endif %} I want to provide follow-up information
```

ALTERNATE

Event Notes can be used as a pilot before a full Related Records process



WHAT THE CONTACT RECEIVES

Rendered counselor outcome

A relationship-specific message for the school contact.

- Recipient comes from the organization contact added as a Related Record
- Sender voice comes from the event counselor
- Message content can branch by visit type or persona

The contact receives useful follow up from the person they met.



WHAT CARRIES FORWARD

High School Visit lessons

The pattern shifts from student voice to staff relationship management.

- Match the recipient
- Match the sender voice
- Match the content

The same structure works when the relationship is staff to contact.



OPERATIONS VOICE

Project 3: Campus Events

Event details can become relationship context too.

- Umbrella Event Template Communications
- Event details become related records
- Related events determine the instructions
- Templates stay stable while logistics change

Operations voice is still a relationship with the attendee.



COMMUNICATION DEBT

Why event templates get messy

Small logistical differences create a lot of recurring risk.

- Same event type, different parking location
- Same day, different check in path
- Weather or construction changes need fast updates
- Template edits become operational risk

The template should stay stable. The details should change safely.



DETAILS AS DATA

Create the Event Details dataset

One dataset can represent parking, check in, updates, and special instructions.

- Use a readable name that maps to a Content Block key
- Attach detail records to the event that owns the logistics
- Let the umbrella communication retrieve those details later

Controlled inputs matter. A misspelled detail category breaks retrieval.



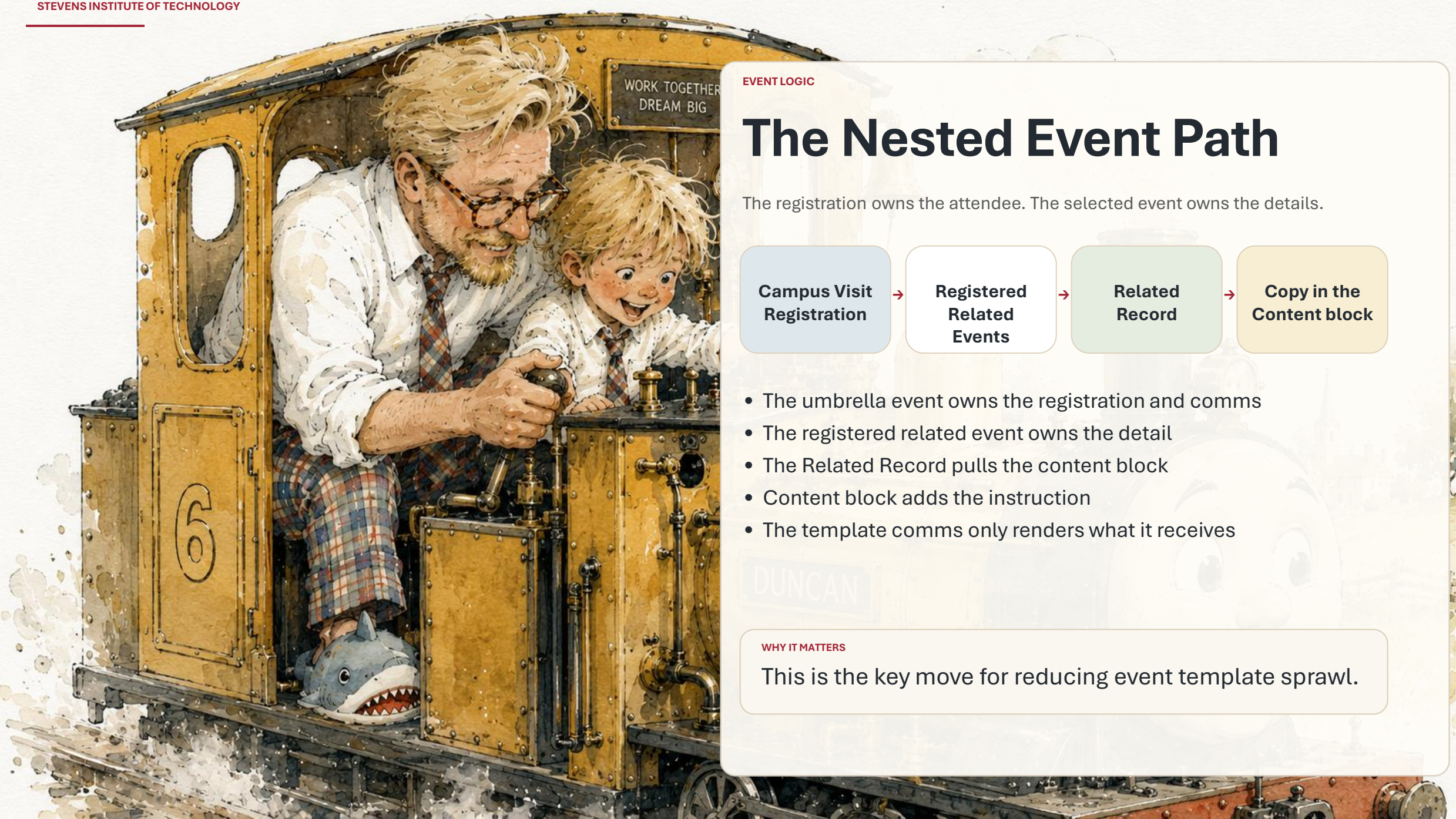
RELATED EVENT LOGIC

Umbrella events inherit context

A visit registration can retrieve details from the selected session or activity.

- Match the related event to the same day
- Pull the relevant detail record from that event
- Render the correct instructions inside the umbrella communication

The chosen activity carries the details the attendee needs.



EVENT LOGIC

The Nested Event Path

The registration owns the attendee. The selected event owns the details.

Campus Visit Registration → **Registered Related Events** → **Related Record** → **Copy in the Content block**

- The umbrella event owns the registration and comms
- The registered related event owns the detail
- The Related Record pulls the content block
- Content block adds the instruction
- The template comms only renders what it receives

WHY IT MATTERS

This is the key move for reducing event template sprawl.

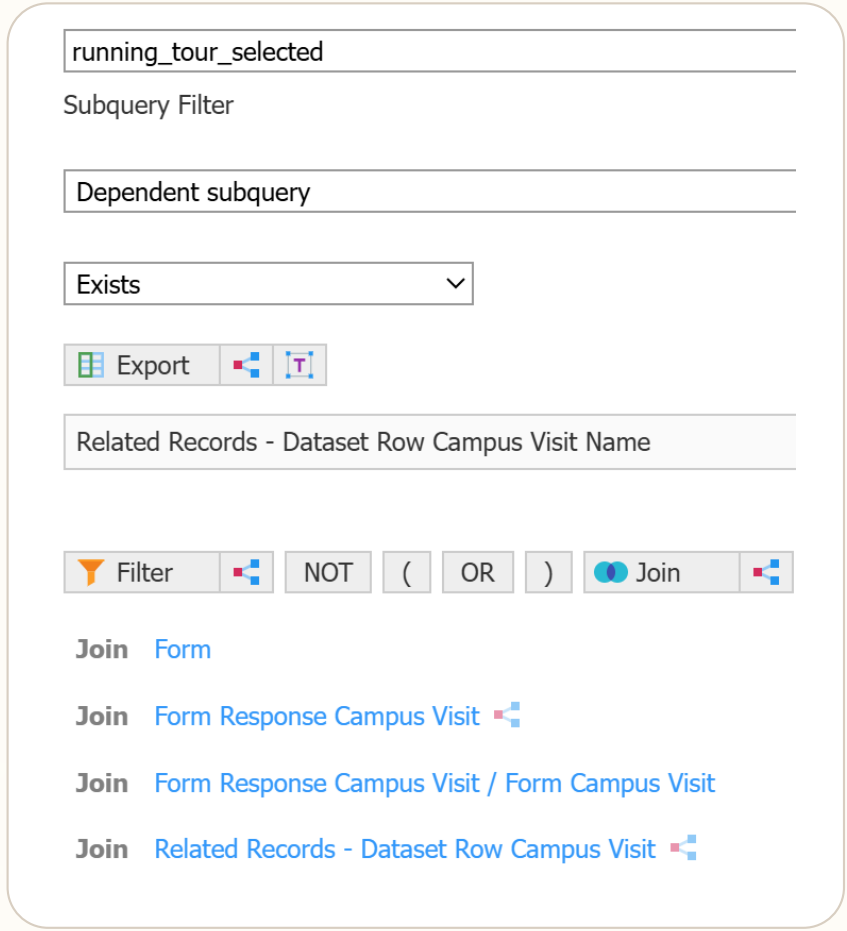


CONFIGURABLE JOIN PATH

Join path reaches through the related event

Retrieve the detail record attached to the selected session or activity.

- Join from the recipient to the umbrella form
- Join to the related event response
- Filter to the correct related event template
- Join to the event detail related record



When several related events exist, sorting and filtering matter.



READABLE COPY

Content Blocks turn values into directions

A retrieved value becomes clear attendee-facing text.

- Retrieved value supplies the Content Block code
- The Content Block stores the actual directions
- A single template can render many event scenarios

Check-In Location: {{campus_visit_check_in | snippet: "event_details"}}
Parking: {{campus_visit_parking | snippet: "event_details"}}
{{campus_visit_parking_to_check_in | snippet: "event_details"}}

The event team changes the related record. The template keeps doing its job.

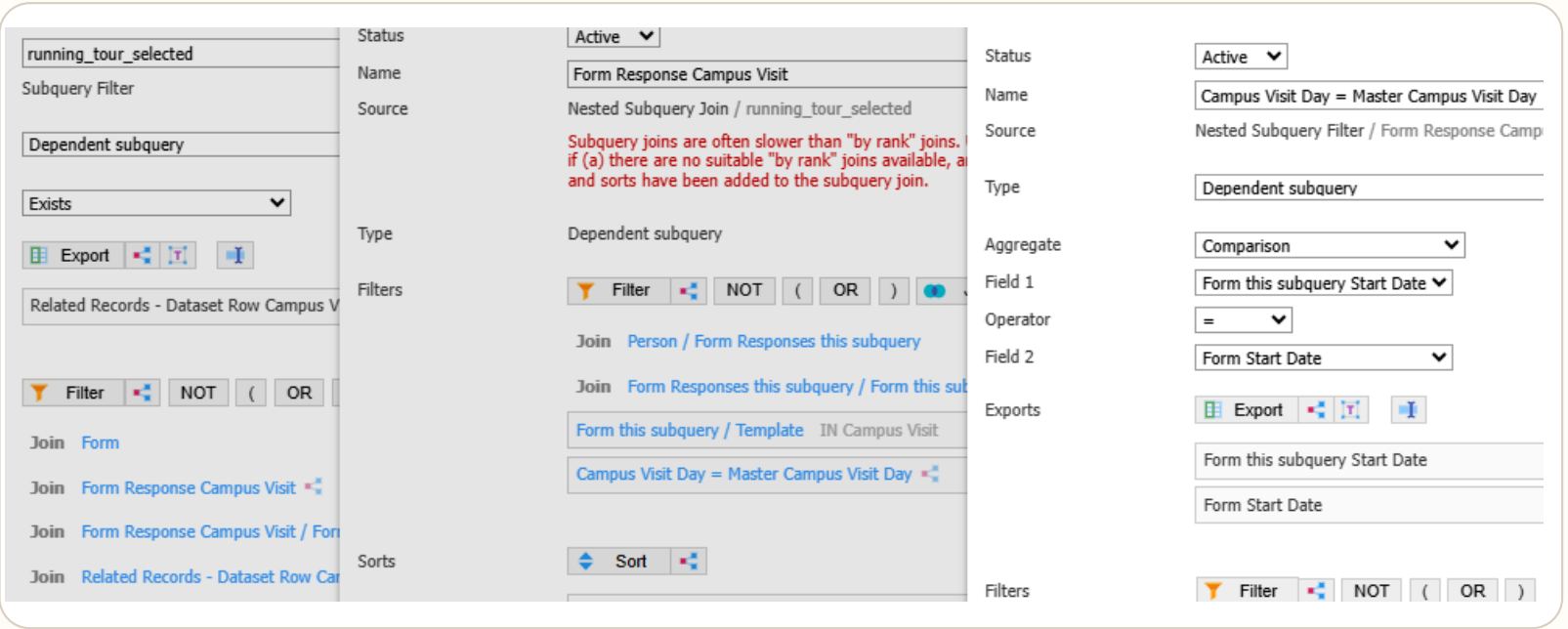
SEND DECISIONS

Use Related Records as send criteria

Some messages should send only when a specific event detail exists using Conditions Filter.

- Send a waiver only when a related event with specific Related Record is attached and registered
- Send a parking update, weather update, cancellation when the event has that Related Record





Related Records can decide what the email says and whether it sends.

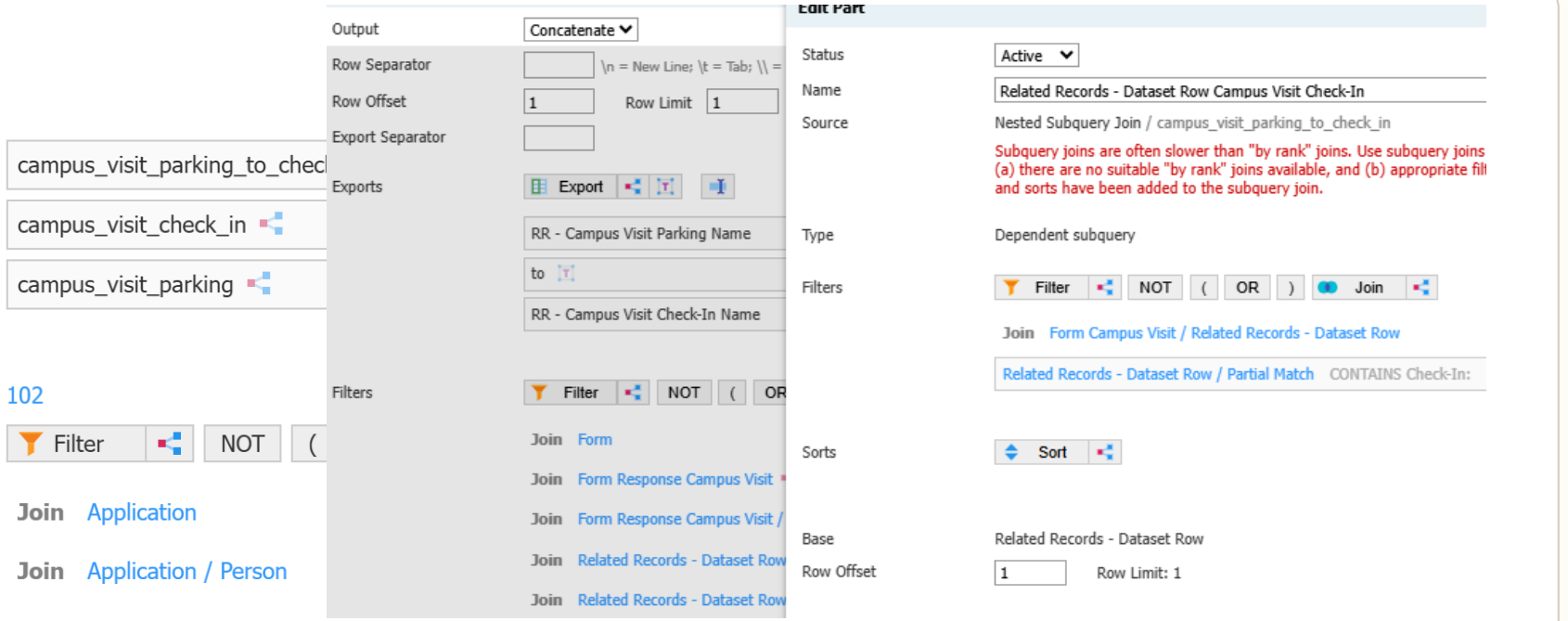
SEND DECISIONS

Use Related Records to pull in Content Blocks

Some messages need dynamic text based on specific details

- Umbrella Event contains comms and main registration
- Template comm's Conditions join to the related event on the same day
- Join to that event's Event Details Related records
- Render the value to pull the correct content Block



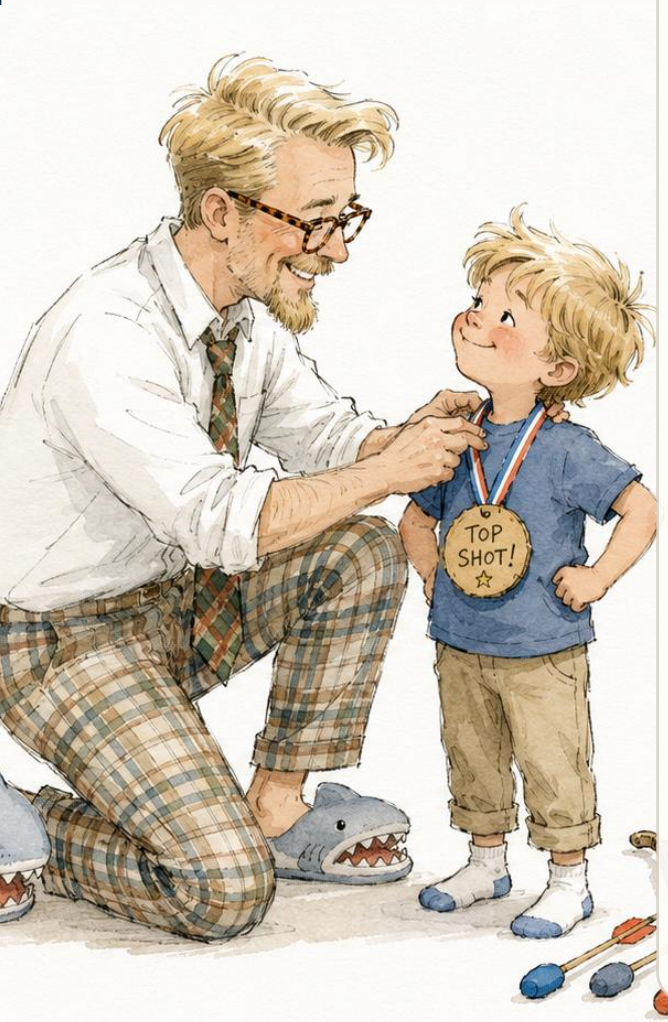


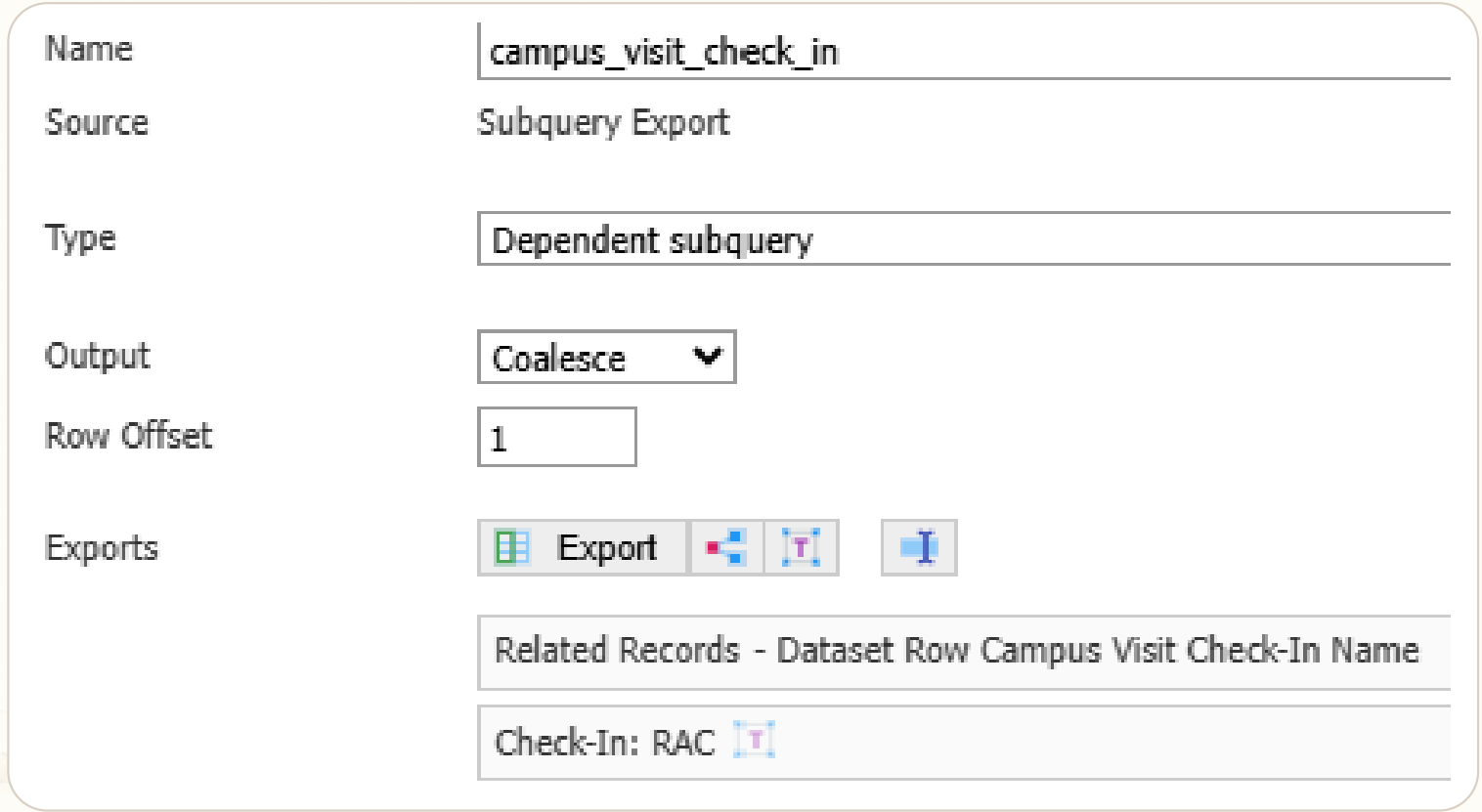
SEND DECISIONS

Design fallback values

Build defaults and fallback values into Exports

- Design a default Content Block code, email, and Condition Export for your most common state
- Fallback also ensures text if staff forget to add a Related Record or technical error
- Use Coalesce Exports to pull in this default/fallback text
- Defaults/Fallbacks in the Exports vs the Content Block Default give more flexibility in the CB Key







FAILURE MODES

Where This Build Fights Back

Technical trust comes from showing what breaks and how you protect against it.

- Several related records or the wrong dataset is retrieved
- Guide exists, field value is missing
- Two related events occur on the same day
- Content Block key does not match the retrieved value
- Old school visit becomes eligible
- Event team skips the required detail record
- Copy renders cleanly, voice sounds odd as rendered

DESIGN STANCE

The build should fail safely. Suppress, fallback, or expose the missing setup to staff.



DEBUG ORDER

How to Debug It

Check the relationship path before rewriting the copy.

Confirm relationship
Does the event have the Related Record?

Confirm recipient
Does this source row belong in the list?

Inspect joins
Did Slate retrieve the expected row?

Export raw values
Can you see the values before logic?

Test flags
Do binary outputs match the record?

Test joins
Does each join correctly identify the datapoint?

Render final
Does the email read cleanly?

Test missing data
Does the fallback behave safely?

PRACTICAL RULE

Most fixes happen before the Liquid. Prove the data path first.



PRACTICAL SEQUENCE

Implementation order

Build one relationship pattern before scaling to all three projects.

- Define the relationship
- Create or reuse the dataset
- Attach the record
- Build joins and exports
- Write modular copy
- Test every branch

Start with one relationship, then reuse the pattern: Webinar Host/Topic emails



AUDIENCE EXERCISE

Build Your Own Relationship Pattern

Pick one recurring message that changes because of one person, event, or detail.

Question	Your answer
Who or what is related?	
Where does the relationship live?	
Which facts change the message?	
Who receives it?	
Whose voice does it use?	
What is the safe fallback?	
Who maintains the record?	

TAKEAWAY

The goal is one relationship, one clean path, and one message that feels specific.



FINAL LINE

Takeaways and resources

Relationship context is the engine. Communications are the output.

- Use Related Records to preserve the relationship behind the interaction
- Use Configurable Joins to make that relationship available to the message
- Use exports and Liquid logic to decide what to say
- Use Content Blocks to keep copy reusable and editable
- Copy the build so the next project starts faster

- Suitcases of the email html and Rec Lists
- PPTs of this session and the Tour Guide Autoresponder Show & Tell
- 75-minute live technical build of the Configurable Joins and Liquid Markup
- Rehearsal Videos

Start with one relationship. Build the logic once. Reuse the pattern.



Questions and resources

- Questions now or email me
- David Dysart
- Associate Director of Data Analytics, Enrollment Management
- Stevens Institute of Technology
- ddysart@stevens.edu

- Suitcases of the email html and Rec Lists
- PPTs of this session and the Tour Guide Autoresponder Show & Tell
- 75-minute live technical build of the Configurable Joins and Liquid Markup
- Rehearsal Videos