Our best price of the year.
Get 20% off new memberships for a limited time.

Home
Library
Profile
Stories
Stats

Following
Alex Rowe
Roger Martin
Dhananjay Garg
The Medium Blog
M.G. Siegler
Anne Bonfert
Krishna Avva
Dr. Benjamin Hardy
Mr. Riyas
Shannon Ashley
More

Batching Record Updates for a Scheduled Flow

Pranav Nagrecha 5 min read · Feb 16, 2023

51

Flows are a powerful tool for automating business processes and making data updates in Salesforce. However, when processing large amounts of data, it can be challenging to avoid errors such as governor limit exceeded errors, DML statement exceptions, and record-locking conflicts. To help address these challenges, you can use the `RecordUpdater` Apex class to batch your data updates.The `RecordUpdater` Apex class is a reusable solution for updating records in Salesforce. By using the `@InvocableMethod` annotation it can be easily called from a flow.

Why create this — A real business issue!

While working on a Scheduled flow that closed opportunities based on open date and other

engagement conditions, I encountered an issue with the flow. The flow was generating errors like

- UNABLE_TO_LOCK_ROW
- CANNOT_INSERT_UPDATE_ACTIVATE_ENTITY

because of the supporting apex triggers and Process Builders that updated accounts, contacts, tasks, and a custom object when an opportunity is closed.

To resolve the issue, I tried a different approach. I extracted all the conditions from the flow and used a data loader with controlled batch sizes to close the opportunities. This approach was successful and did not result in any errors. I wanted to replicate that in a flow, so below is the apex class I created....

Apex Class

Name: **RecordUpdater**

```
global class RecordUpdater {
```

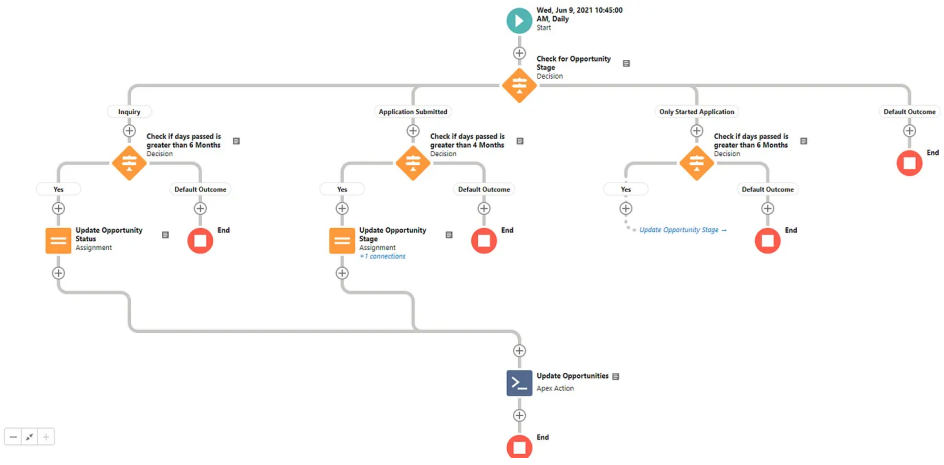
```
    // Declare an invocable method to update records wit
    @InvocableMethod(label='Update Records' description=
    global static void updateRecords(List<RecordUpdateRe
        // Loop through each record update request
        for (RecordUpdateRequest request : requests) {
            // Get the list of records to update
            List<SObject> recordsToUpdate = request.reco
            // Get the specified batch size or default t
            Integer batchSize = request.batchSize;
            if (batchSize == null) {
                batchSize = recordsToUpdate.size();
            }
            // Keep track of the current start index for
            Integer startIndex = 0;
            // Continue updating records as long as ther
            while (startIndex < recordsToUpdate.size())
                // Determine the end index for the curre
                Integer endIndex = Math.min(startIndex +
                // Create a new list to store the record
                List<SObject> batch = new List<SObject>(
                // Loop through the records to update fo
                for (Integer i = startIndex; i < endInde
                    // Add each record to the batch list
                    batch.add(recordsToUpdate.get(i));
                }
                // Update the records in the current bat
                update batch;
                // Update the start index to the end ind
                startIndex = endIndex;
            }
        }
    }

    // Declare a class to store information about a reco
    global class RecordUpdateRequest {
        // Declare a required variable to store the reco
        @InvocableVariable(required=true)
        global List<SObject> records;

        // Declare a variable to store the specified bat
        @InvocableVariable
        global Integer batchSize;
    }

}
```

There are several benefits to using the Apex class in a

flow:

1. Performance: By reducing the amount of data processed in a single transaction. This can help avoid exceeding governor limits and reduce the risk of encountering errors such as timeout exceptions and DML statement exceptions.
2. Scalability: By allowing you to process larger amounts of data all within Salesforce reducing dependency on ETL Platforms for some things.
3. Reliability: Batching records can improve the reliability of your flows and transactions by reducing the risk of encountering errors such as governor limit exceeded errors, DML statement exceptions, and record-locking conflicts.

Example



We go through a certain set of decisions to determine if we need to close this opportunity

In the Assignment Variable (Could be either, but I am going to use the center one called *Update Opportunity Stage, Status, and Role)* we first set the $Record variable values to what we want them to be and then add the $Record variable to another Record Collection variable that will be used further down in the Apex. We will call that collection variable Opportunity4Update

Assignment: Update Opportunity Stage, Status, and Role (Middle)

Collection Variable: Opportunity4Update

The updated values are now connected to the Apex Action

Apex Action: Update Opportunities

because this is a scheduled flow we don’t get a choice to pick a record we want to debug with. I tried it with a record to see how did it perform with a Flow Record Update to the Apex. Here are the results — // These are results form a single record debugging.

Alternate Approach

Instead of using the Apex Class to simply control the batch size because of errors you are facing (in this example let's stick to Opportunities closing), You can do the following:

- Keep the current scheduled flow and instead of closing an opportunity, Just update a new checkbox called *'trigger opportunity close'*
- Create a new Record Trigger flow to run only on record update and when *'trigger opportunity close'* = true.
- Create a dummy scheduled path and define a batch

size. (This is an example for the below flow, but the logic remains the same)

Dummy Schduled Path

- Close the opportunity via the Record Triggered flow with controlled batch size.

I created a flow that generates total compensation and individual payment records based on enrollments in a course. The flow is triggered when enrollments are updated and calculates the total compensation for the course. Then, it creates individual payment records that are used to pay out the faculty on a bi-weekly basis. The payment records are continually adjusted based on changes in the number of students enrolled or dropped. This approach ensures that faculty members are fairly compensated for their work and that the payment records are accurate and up-to-date. Because of system timeouts and other issues, this was a workaround for the Apex class as all I was looking for was to control the batch size.

Salesforce Development
Salesforce
Apex
Automation
Flow

Written by Pranav Nagrecha

51 followers · 43 following

Edit profile

Salesforce Administrator/Developer/Architect/Business Analyst/QA

No responses yet

Pranav Nagrecha he

What are your thoughts?

More from Pranav Nagrecha

Pranav Nagrecha

Boomi—How to convert Timezones in a Process

I use Google to the extreme, I know there are smarter peopl...

May 17, 2022

Pranav Nagrecha

How We Built a Real-Time Integration...

Say goodbye to 15-minute polling delays—here's how we...

May 14 13

Pranav Nagrecha

Stop Guessing. Start Designing: Order of...

Your save isn't haunted—it's a kitchen line: before-save, after...

Aug 26 2

Pranav Nagrecha

Roles, Profiles, and Permission Sets in...

Introduction

Aug 21 19

See all from Pranav Nagrecha

Recommended from Medium

Tosny

7 Websites I Visit Every Day in 2025

If there is one thing I am addicted to, besides coffee, it i...

Sep 23 · 8.9K · 311

Vaishnavi R

Database. Stateful in Salesforce

If you have ever worked with batch classes in Apex, you hav...

Jul 23 · 2

Musa Ndlala

Understanding Standard Objects and...

The Moment You Realize Salesforce Isn't Just Data—It's ...

Sep 15 5

Jeremy Carmona

How I Use Salesforce Templates to Save...

Third in the "Salesforce, Simplified" series

Jun 12 12.1K 22

Raja Saha

Generating Unique Identifiers in Salesforc...

In Salesforce, unique identifiers play a vital role in managing...

Jun 30

In Generative ... by Adham Khal...

Stanford Just Killed Prompt Engineering...

ChatGPT keeps giving you the same boring response? This...

Oct 19 18.5K 449

See more recommendations