SALESFORCE MARKETING CLOUD • CONTACT DATA HYGIENE

SFMC Bulk Delete Contacts 2026: Why Subscriber.Remove() Fails

Subscriber.Remove() runs without errors but never touches All Contacts. Full deletion requires the Contacts REST API DeleteByListReference endpoint. Here is the SQL to identify records and the SSJS script that actually deletes them.

by Genetrix Technology

Published June 10, 2026 • genetrix.tech/blogs

Genetrix Consulting LLC
304 S. Jones Blvd #5631, Las Vegas, NV 89107, USA | genetrix.tech

SFMC Bulk Delete Contacts 2026: Why Subscriber.Remove() Fails

By Genetrix Marketing • Published June 10, 2026 • https://genetrix.tech/blogs/sfmc-bulk-delete-contacts-from-all-contacts-2026-why-subscriberremove-fails-and/

Subscriber.Remove() runs without errors but never touches All Contacts. Full deletion requires the Contacts REST API DeleteByListReference endpoint. Here is the SQL to identify records and the SSJS script that actually deletes them.

A client came to us with a seemingly simple request: remove all converted leads from the All Contacts list in SFMC. Their CRM sync had been running for two years and All Contacts had ballooned with Lead records that no longer existed in Salesforce. It was eating into their contact credits and slowing down automations.

Their developer had already tried using Subscriber.Remove() in a loop script. The script ran without errors. The contacts were still there. This is the first thing almost everyone tries, and it is the wrong tool for the job — and understanding exactly why it fails is what makes the correct approach make sense.

Why Subscriber.Remove() Does Not Actually Delete Contacts

Subscriber.Remove() from the Core SSJS library removes a subscriber from a specific list. That is all it does. It does not touch the All Contacts list, the contact data model, or any of the channel-specific contact records. The subscriber record in All Contacts remains completely intact after calling it.

> **Common misconception:** Running Subscriber.Remove() or iterating through records and calling the Core Library subscriber delete functions will return no errors even though the contact still exists in All Contacts. This is what makes it particularly deceptive — the script appears to succeed, but nothing has actually been deleted from the contact data model. You will not know it failed until you check the All Contacts count and find it unchanged.

Full contact deletion from All Contacts requires going through the Contacts REST API, specifically the DeleteByListReference endpoint. This is the only method that actually removes a contact from the SFMC contact data model across all apps — Email, MobileConnect, MobilePush, and all sendable Data Extensions. It is also asynchronous, which means the deletion is queued and processed in the background, sometimes taking several hours depending on volume.

What You Need Before Running This

Contact Deletion Enabled — must be enabled at the parent Business Unit level in Setup. If you do not see the option, raise a support case.

Installed Package with Contacts Scope — your API integration package needs Contacts read and write scope. Create one in Setup under Apps if you do not have it.

Sendable Data Extension — the DE used as the deletion source must be marked as sendable with a SubscriberKey send relationship. Non-sendable DEs are not supported by the endpoint.

> **Suppression period:** SFMC applies a suppression period after deletion, defaultable between 0 and 30 days (platform default is 2 days). During this window, deleted contacts cannot be re-added to the system. If you are running a regular sync from Salesforce CRM, be aware that any contacts you delete may be blocked from re-entering if they re-appear in the sync before the suppression window expires.

Step 1: Identify the Contacts to Delete

Before anything else, you need to populate a sendable Data Extension with the SubscriberKeys of the contacts you want to remove. The REST API endpoint reads from this DE — it does not accept a dynamic query, it reads a pre-populated list.

SQL Query — Identify orphaned Lead contacts for deletion

```
/* Find Lead records in All Contacts that no longer exist
   in the Salesforce Leads sync DE.
   Salesforce Lead IDs start with '00Q'.
   Salesforce Contact IDs start with '003'.
   Adapt the JOIN to your own synced Lead/Contact DE names. */

SELECT
    ac.SubscriberKey

FROM _EnterableContacts ac

LEFT JOIN Lead_Salesforce lead
    ON ac.SubscriberKey = lead.Id

WHERE
    ac.SubscriberKey LIKE '00Q%'
    AND lead.Id IS NULL
```

Field	Type	Len	Required
SubscriberKey (PK)	Text	254	Required
DeleteReason	Text	200	Optional — useful for auditing
AddedDate	Date	-	Optional — useful for auditing

Step 2: The SSJS Deletion Script

Once the staging DE is populated, this script authenticates against the Marketing Cloud REST API and fires the DeleteByListReference request. Drop it into an SSJS Script Activity in Automation Studio, schedule it to run after your SQL query populates the staging DE, and it handles the rest.

SSJS Script Activity — Bulk Contact Deletion via REST API

```
<script runat="server">
Platform.Load("Core", "1");

var clientID      = "YOUR_CLIENT_ID";
var clientSecret  = "YOUR_CLIENT_SECRET";
var clientBase    = "YOUR_28_CHAR_SUBDOMAIN";
var deletionDEKey = "YOUR_DELETION_DE_EXTERNAL_KEY";

var contentType = "application/json";
var accessToken = "";
var operationID = "";

try {
    var authUrl     = "https://" + clientBase + ".auth.marketingcloudapis.com/v2/token";
    var authPayload = Stringify({
        "grant_type": "client_credentials",
        "client_id": clientID, "client_secret": clientSecret
    });
    var authResponse = HTTP.Post(authUrl, contentType, authPayload);
    var authObj      = Platform.Function.ParseJSON(authResponse["Response"][0]);
    accessToken      = authObj.access_token;
    if (!accessToken) { Write("Authentication failed."); return; }

    var deleteUrl = "https://" + clientBase
        + ".rest.marketingcloudapis.com"
        + "/contacts/v1/contacts/actions/delete?type=listReference";

    var deletePayload = Stringify({
        "deleteOperationType": "ContactAndAttributes",
        "targetList": { "listType": { "listTypeID": 3 }, "listKey": deletionDEKey },
        "deleteListWhenCompleted": false,
        "deleteListContentsWhenCompleted": true
    });

    var headerNames  = ["Authorization"];
    var headerValues = ["Bearer " + accessToken];
    var deleteResult = HTTP.Post(deleteUrl, contentType, deletePayload, headerNames, headerVa
lues);
    var resultObj    = Platform.Function.ParseJSON(deleteResult["Response"][0]);
    operationID      = resultObj.operationID;
    Write("Deletion queued. OperationID: " + operationID);
} catch(ex) {
    Write("Error: " + Stringify(ex));
}
</script>
```

> **Save the OperationID.** The deletion runs asynchronously and can take several hours for large volumes. Log the operationID to a Data Extension so you can check deletion status later.

Before You Run This

- Verify Contact Deletion is enabled in your parent Business Unit — go to Setup and search for Contact Delete Settings.
- Test your SQL query first in Query Studio against a small, clearly identifiable set of records before running it against your full population.
- The deletion DE must be sendable with SubscriberKey as the send relationship field. Check this in Contact Builder before running the script.

- Run a contact count against the staging DE after the SQL query and before the SSJS script. That number should match what you expect to delete.
- The deleteListContentsWhenCompleted: true flag will clear the DE contents after deletion runs. Set it to false if you want to keep the deletion log, and archive the records separately beforehand.
- Do not schedule this automation to run more frequently than your suppression period. If suppression is 2 days and you run daily, deleted contacts from the previous day may not be re-addable yet.
- For volumes over 1 million records, split into multiple batches and run sequentially. The API processes one deletion request at a time per account.

Frequently Asked Questions

Does this delete the contact from Salesforce CRM as well?

No. This deletes the contact from the SFMC contact data model only — All Contacts, all sendable Data Extensions, Email Studio subscriber records, and channel-specific records in MobileConnect and MobilePush. The corresponding Lead or Contact record in Salesforce CRM is completely untouched.

What is the difference between ContactAndAttributes and AttributesOnly?

ContactAndAttributes performs a full deletion — it removes the contact record and all associated data across the entire SFMC contact model. AttributesOnly only removes channel-specific information while leaving the core contact record intact. For most use cases including GDPR compliance requests and clearing converted leads, you want ContactAndAttributes.

How do I check whether the deletion actually completed?

Use the operationID returned by the initial API call and make a GET request to the contacts/v1/contacts/analytics/deleterequests endpoint filtering by that ID. When the status field shows Completed, the deletion is done.

Can I run this from a CloudPage instead of Automation Studio?

Technically yes, but we do not recommend it for bulk deletions. CloudPages have execution timeouts that can interrupt large operations, and you lose the scheduling and logging capabilities that Automation Studio provides. For one-time small-scale deletions during setup or testing, a CloudPage is fine. For anything recurring or at volume, use a Script Activity in Automation Studio.