Social Studio Next — Comprehensive User & Administrator Guide

Version: 1.0 **Namespace:** `rissnext` **API Version:** v62.0 **Package Type:** Second-Generation Managed Package (2GP)

Table of Contents

Part 1: Application Overview

What Is Social Studio Next?

Social Studio Next (SSN) is a Salesforce-native social media management platform that enables organizations to publish content, monitor brand conversations, engage with audiences, and analyze performance — all within the Salesforce ecosystem. It integrates deeply with Service Cloud for case management, Data Cloud for unified customer profiles, and Slack for team notifications.

Supported Platforms

Platform	Publishing	Listening	Reply/DM	Analytics
X (Twitter)	Yes	Yes	Yes	Yes
Facebook	Yes	Own Pages	Yes	Yes
Instagram	Yes	Hashtags (30/week)	Yes	Yes
LinkedIn	Yes	Own Pages	Yes	Yes
TikTok	Yes (Video only)	No	No	Limited

Key Capabilities

- **Multi-Platform Publishing** — Compose once, publish to multiple platforms simultaneously with platform-specific content variants
- **Content Calendar** — Visual month view of scheduled, pending, and published posts
- **Approval Workflows** — Multi-level approval chains with SLA tracking and auto-escalation

- **Recurring Posts** — Schedule daily, weekly, biweekly, or monthly recurring content
- **Social Listening** — Keyword-based brand monitoring, competitor tracking, and campaign monitoring
- **AI Sentiment Analysis** — Automated sentiment scoring, emotion detection, entity extraction, and crisis flagging
- **Unified Inbox** — Threaded conversation view for customer engagement across all platforms
- **CRM Case Integration** — Auto-create or manually create Service Cloud cases from social posts
- **Analytics Dashboard** — Engagement metrics, sentiment trends, share of voice, and platform comparisons
- **Crisis Detection** — Automated alerts and case creation when negative sentiment or crisis keywords are detected
- **Data Cloud Integration** — Unified customer profiles linking social authors to CRM contacts via the Ingestion API
- **Slack Notifications** — Real-time alerts for approvals, crisis events, and publishing status
- **Digital Asset Management** — Browse Salesforce Files (ContentVersion) directly from the post composer
- **Data Privacy** — PII redaction and GDPR right-to-erasure processing

---

Part 2: Getting Started

Permission Sets & Roles

Social Studio Next includes four permission sets that control access to features and data. Assign the appropriate permission set to each user based on their role.

SSN_Admin

Audience: System administrators, social media directors

Capability	Access
All objects	Full CRUD + ViewAll + ModifyAll
OAuth & account management	Full access
Approval chain configuration	Full access
Analytics	Full access
Listening topics	Full access
Notification preferences	Full access

SSN_Manager

Audience: Social media managers, content strategists, team leads

Capability	Access
Authored posts	Create, Read, Update, Delete

Capability	Access
Social posts	Read only
Listening topics	Create, Read, Update, Delete
Social accounts	Read only
Social authors	Read, Update (link to contacts)
Approval chains	Read only
Analytics	Full access

SSN_Creator

Audience: Content creators, copywriters, designers

Capability	Access
Authored posts	Create, Read, Update, Delete (own only)
Post media	Create, Read, Update, Delete (own only)
Social posts	Read only
Listening topics	Read only
Analytics	Limited

SSN_Agent

Audience: Customer service representatives, social support agents

Capability	Access
Social posts	Read only
Inbox conversations	Read, Reply
Cases	Create, Read, Update
Social authors	Read only
Authored posts	Read only

Application Navigation

After installation, the **Social Command Center** Lightning app appears in the App Launcher. It contains these tabs:

Tab	Description	Roles
Dashboard	KPI summary, sentiment trends, volume charts, trending topics	All
Publishing	Post composer, content calendar, approval manager	Admin, Manager, Creator

Tab	Description	Roles
Listening	Topic management, social feed, crisis alerts	Admin, Manager, Agent
Inbox	Threaded conversations, reply composer	Admin, Agent
Analytics	Engagement metrics, share of voice, competitor benchmarks	Admin, Manager
Settings	Account management, OAuth, topics, notifications, Data Cloud	Admin

Dashboard Overview

The Dashboard provides a real-time snapshot of your social media performance:

- **KPI Cards** — Total Posts, Total Engagement (likes + shares + comments), Average Sentiment Score, Active Listening Topics
- **Sentiment Trend Chart** — Line chart showing sentiment over time (powered by Chart.js)
- **Volume by Platform** — Bar chart showing post volume per platform
- **Trending Topics** — Top 10 topics by post volume

Date Range Filters: Last 24 Hours, Last 7 Days, Last 30 Days, Last 90 Days

---

Part 3: End-User Feature Guides

3.1 Publishing

Creating a New Post

1. Navigate to the **Publishing** tab
2. Click **New Post** to open the Post Composer
3. Write your content in the main text area
    - A real-time **character counter** shows remaining characters per platform
    - Content is validated against each platform's limits (e.g., 280 for X/Twitter)
4. Select **Target Platforms** — check one or more: X/Twitter, Facebook, Instagram, LinkedIn, TikTok

Content Variants

For platform-specific messaging, use **Content Variants**:

1. After selecting multiple platforms, click **Add Variant** for any platform
2. Write alternate content for that platform
3. Each platform will receive its variant instead of the main content
4. Variants are stored as JSON in the `Content_Variants__c` field

Media Uploads

1. Click **Upload Media** or drag-and-drop files into the media area
2. Alternatively, click **Browse Files** to open the DAM Browser and select from Salesforce Files
3. Add **Alt Text** for accessibility (recommended for all images)

4. Reorder media items via drag-and-drop for carousel posts

Platform Media Constraints:

Platform	Max Images	Max Video Size	Max Video Duration	Formats
X/Twitter	4	512 MB	140 sec	JPEG, PNG, GIF, MP4
Facebook	10	10 GB	4 hours	JPEG, PNG, GIF, MP4, MOV
Instagram	10	10 GB	90 sec (reels)	JPEG, PNG, MP4
LinkedIn	9	10 GB	—	JPEG, PNG, MP4
TikTok	0 (video only)	287.6 MB	—	MP4

Scheduling Posts

- **Publish Now** — Post is published immediately upon approval (or immediately if no approval chain)
- **Schedule** — Select a future date and time; the post enters the publishing queue and is published at the scheduled time by the `PostPublishSchedulable` batch job

Recurring Posts

1. Toggle **Is Recurring** on in the composer sidebar
2. Select a **Pattern**: Daily, Weekly, Biweekly, or Monthly
3. For Weekly/Biweekly: select a **Day of Week**
4. For Monthly: select a **Day of Month** (1-28)
5. Set an **End Date** for the recurrence
6. On first publish, the system automatically clones the post for the next occurrence

The recurrence rule is stored as JSON:

```
{
  "pattern": "weekly",
  "endDate": "2026-12-31",
  "dayOfWeek": "MON"
}
```

UTM Parameters

1. Expand the **UTM Parameters** section in the composer
2. Fill in: Campaign, Source, Medium, Content, Term
3. The system automatically appends UTM parameters to all URLs found in your post content
4. If Bitly integration is configured, URLs are also shortened

First Comment Strategy (Instagram/Facebook)

1. When Instagram or Facebook is selected as a target platform, a **First Comment** field appears
2. Enter hashtags or supplementary content

3. After the main post is published, the system automatically posts the first comment as a reply
4. This is a common Instagram strategy to keep captions clean while maximizing hashtag reach

Content Calendar

The Content Calendar provides a visual month view of all posts:

- **Color coding by status:** Draft (gray), Pending Approval (yellow), Approved (blue), Scheduled (purple), Published (green), Failed (red)
- **Recurring indicator** — Posts with `Is_Recurring__c = true` show a recurring icon
- Click any post to view details or edit
- Navigate between months using arrow controls

3.2 Approval Workflows

Submitting a Post for Approval

1. Save your post as a **Draft**
2. Click **Submit for Approval**
3. The post status changes to **Pending Approval**
4. The Level 1 approver is notified (email, in-app, and/or Slack based on preferences)

Reviewing & Approving Posts (Managers)

1. Navigate to **Publishing > Approval Manager**
2. View the queue of pending posts
3. Click a post to see the full preview (content, media, target platforms)
4. Click **Approve** to advance the post or **Reject** with a reason
5. Approved posts move to **Scheduled** (if a schedule date is set) or **Approved** (ready for immediate publish)

Multi-Level Approval

Approval chains support up to 3 levels:

- **Level 1** — Initial review (e.g., content manager)
- **Level 2** — Secondary review (e.g., brand director)
- **Level 3** — Final review (e.g., VP, conditionally required)

Each level has a configurable SLA (hours). If the SLA is exceeded and auto-escalation is enabled, the post escalates to the next approver or the designated escalation user.

Rejection Flow

1. Approver clicks **Reject** and enters a reason
2. Post status reverts to **Draft**
3. Creator is notified of the rejection with the reason
4. Creator edits the post and resubmits

3.3 Social Listening

Creating a Listening Topic

1. Navigate to **Settings > Listening Topics** (Managers/Admins)
2. Click **New Topic**
3. Configure:
    - **Topic Name** — Descriptive name (e.g., "Brand Mentions", "Competitor XYZ", "#SummerCampaign")
    - **Topic Type** — Brand Monitoring, Competitor, Campaign, Crisis, or Industry
    - **Keywords** — Comma or newline-separated keywords to match
    - **Exclude Keywords** — Keywords to filter out (negative matching)
    - **Platforms** — Select which platforms to monitor
    - **Languages** — Optional language filters
    - **Geo Filters** — Optional geographic filters (JSON)
    - **Sentiment Threshold** — Link to a threshold for automated actions
    - **Volume Spike Threshold** — Posts per hour that triggers a spike alert
    - **Alert on Crisis** — Enable crisis alerting for this topic
    - **Is Active** — Toggle monitoring on/off
4. Click **Save**

Viewing the Social Feed

1. Navigate to the **Listening** tab
2. Use filters to narrow results:
    - **Topic** — Select a specific listening topic
    - **Platform** — Filter by social network
    - **Sentiment** — Filter by Positive, Negative, Neutral, or Mixed
    - **Date Range** — Custom or preset ranges
3. Posts appear in reverse chronological order showing:
    - Author handle and display name
    - Post content
    - Platform icon
    - Sentiment badge (color-coded)
    - Engagement metrics (likes, comments, shares)
4. Click any post for the detail view:
    - Full content and metadata
    - Sentiment analysis breakdown (score, confidence, emotions, topics, entities)
    - Author profile (influence score, interaction history)
    - Conversation thread (if threaded)
    - Related CRM case (if one exists)

Crisis Detection

When the sentiment engine flags a post as a crisis:

1. The **Crisis_Alert_Event__e** platform event is published
2. Slack notification is sent to the configured crisis channel

3. A Case is automatically created (if threshold is configured with `action_type = 'create_case'`)
4. Crisis posts appear with a red badge in the Listening feed

---

3.4 Inbox & Conversations

Conversation List

1. Navigate to the **Inbox** tab
2. Conversations are grouped by author handle + platform
3. Each conversation shows:
    - Author avatar and handle
    - Last message preview
    - Timestamp of last activity
    - Sentiment badge
    - Unread indicator (if new messages)

Conversation Detail

1. Click a conversation to open the detail view
2. View the full thread (oldest to newest)
3. See the **Author Profile Card**:
    - Influencer status
    - Average sentiment across interactions
    - Total interaction count
    - Linked CRM records (Contact, Lead, Account)
4. If a Case exists, click the case link to navigate to it

Replying to Posts

1. Scroll to the bottom of the conversation detail
2. Type your reply in the **Reply Composer**
3. Character counter shows platform-specific limits
4. Click **Send Reply**
5. The reply is posted to the platform in real-time
6. If a Case is linked, the case is updated with `Response_Posted__c = true`

Note: TikTok does not support API replies. Manual replies must be posted directly on TikTok.

Creating a Case from a Post

1. In the Listening feed or Conversation detail, click **Create Case**
2. A Case is created with:
    - Subject auto-generated from post content
    - Origin set to "Social Media"
    - Priority based on sentiment severity
    - Linked Social_Post__c and Social_Author__c
    - Contact auto-resolved from author's CRM links
3. The case appears in the standard Cases tab for assignment and resolution

3.5 Analytics

Dashboard Metrics

The Analytics tab provides comprehensive performance insights:

- **Engagement Overview** — Total likes, comments, shares, views across all platforms
- **Engagement Rate** — (Total engagement / Total views) as a percentage
- **Average Sentiment** — Mean sentiment score across analyzed posts
- **Platform Breakdown** — Engagement metrics broken down by platform (bar chart)

Sentiment Trends

A line chart showing sentiment over time, filterable by:

- Platform
- Date range
- Listening topic

Share of Voice

Shows your brand's volume compared to competitors:

- Horizontal bar chart showing volume percentages per listening topic type
- Data table with exact counts and percentages
- Filter by date range

Competitor Benchmarking

Compare your brand's performance against competitor topics:

- Engagement metrics side-by-side
- Sentiment comparison
- Volume trends

Campaign Performance

When posts are linked to Salesforce Campaigns:

- View engagement metrics per campaign
- Track publishing success rates
- Measure campaign sentiment

Top Posts

A ranked list of your highest-performing posts based on total engagement, showing:

- Post content preview
- Platform
- Engagement metrics
- Sentiment score

Part 4: Administrator Configuration Guide

4.1 Initial Setup Checklist

Step	Action	Details
1	Install package	Install Social Studio Next managed package from AppExchange or package install URL
2	Assign permission sets	Assign SSN_Admin, SSN_Manager, SSN_Creator, or SSN_Agent to users
3	Register platform apps	Create developer apps on each social platform (see Section 4.2)
4	Configure OAuth	Add platform credentials to SSN_OAuth_Config__mdt or OAuth_App__c
5	Connect social accounts	Use the Settings > Connected Accounts flow to OAuth-connect each brand account
6	Create listening topics	Define keywords, platforms, and alert thresholds for monitoring
7	Configure approval chains	Set up multi-level approval workflows for content review
8	Configure Slack (optional)	Add Slack incoming webhook URLs to SSN_Slack_Config__mdt
9	Configure Data Cloud (optional)	Set up Ingestion API for unified profiles (see Part 5)
10	Review sentiment thresholds	Activate/customize automated action thresholds

4.2 OAuth & Social Account Setup

Platform Developer App Registration

Before connecting social accounts, you must register a developer application on each platform.

X (Twitter)

1. Go to developer.twitter.com
2. Create a new Project and App
3. Enable **OAuth 2.0** with PKCE
4. Set **App permissions** to Read and Write
5. Add the **Callback URL**:
   `https://{yourDomain}.my.salesforce.com/apex/rissnext__SSNOAuthCallback`
6. Note the **Client ID** and **Client Secret**

7. Required scopes: `tweet.read`, `tweet.write`, `users.read`, `media.write`, `offline.access`, `like.read`, `like.write`

Facebook & Instagram

1. Go to developers.facebook.com
2. Create a new App (type: Business)
3. Add the **Facebook Login** product
4. Add the **Instagram Graph API** product
5. Set **Valid OAuth Redirect URI**: `https://{yourDomain}.my.salesforce.com/apex/rissnext__SSNOAuthCallback`
6. Note the **App ID** and **App Secret**
7. Required Facebook scopes: `pages_show_list`, `pages_read_engagement`, `pages_manage_posts`, `pages_manage_metadata`, `pages_manage_engagement`, `publish_video`, `read_insights`
8. Required Instagram scopes: `instagram_basic`, `instagram_content_publish`, `instagram_manage_comments`, `instagram_manage_insights`

> **Important:** Instagram publishing requires publicly accessible media URLs. Media files behind authentication will fail.

LinkedIn

1. Go to linkedin.com/developers
2. Create a new App
3. Request access to the **Community Management API** and **Advertising API** products
4. Add the **Authorized Redirect URL**: `https://{yourDomain}.my.salesforce.com/apex/rissnext__SSNOAuthCallback`
5. Note the **Client ID** and **Client Secret**
6. Required scopes: `openid`, `profile`, `w_member_social`, `r_organization_social`, `w_organization_social`

TikTok

1. Go to developers.tiktok.com
2. Create a new App
3. Apply for **Content Posting API** access
4. Add the **Redirect URI**: `https://{yourDomain}.my.salesforce.com/apex/rissnext__SSNOAuthCallback`
5. Note the **Client Key** (TikTok uses `client_key` instead of `client_id`) and **Client Secret**
6. Required scopes: `user.info.basic`, `video.publish`, `video.upload`, `video.list`

> **Important:** Unaudited TikTok apps can only post as PRIVATE. Apply for TikTok's audit process to enable public posting.

Configuring OAuth Credentials in Salesforce

Option A: Custom Metadata Type (Recommended for managed package)

1. Go to **Setup > Custom Metadata Types > SSN OAuth Config > Manage Records**

2. Create a record for each platform:
    - **Label**: Platform name (e.g., "X Twitter")
    - **Developer Name**: Must match platform identifier (`x_twitter`, `facebook`, `instagram`, `linkedin`, `tiktok`)
    - **Client_Id__c**: Paste the platform's Client ID / App ID / Client Key
    - **Client_Secret__c**: Paste the platform's Client Secret / App Secret

Option B: OAuth App Custom Object

1. Navigate to the SSN app
2. Go to **Settings > OAuth Apps**
3. Click **New OAuth App**
4. Fill in Platform, Client ID, Client Secret
5. Set **Is Active** to true

Connecting Social Accounts

1. Navigate to **Settings > Connected Accounts**
2. Click **Connect Account**
3. Select a platform
4. Click **Authorize**
5. You are redirected to the platform's OAuth consent screen
6. Grant the requested permissions
7. You are redirected back to Salesforce
8. A `Social_Account__c` record is created with:
    - `Token_Status__c = 'active'`
    - `Account_Handle__c` populated from the platform
    - `Account_Platform_Id__c` set to the platform's user/page ID
    - `Is_Active__c = true`

Token Management

- **Token Refresh**: The `TokenRefreshSchedulable` batch job runs periodically to refresh tokens before expiry
- **Token Status Values**: `active`, `expired`, `revoked`
- **Expired Tokens**: The account shows a warning badge. Re-authenticate by clicking **Reconnect** in Settings
- **Revoking Tokens**: Deactivate the account in SSN, then revoke access on the platform's developer console

4.3 Platform Configuration Reference

Platform specifications are stored as Custom Metadata (`Platform_Config__mdt`) and are read-only after installation. These values are used to validate content before publishing.

Platform	Max Text	Max Images	Max Video Size	Max Video Duration	Rate Limit
X/Twitter	280 chars	4	512 MB	140 sec	300 req / 900 sec

Platform	Max Text	Max Images	Max Video Size	Max Video Duration	Rate Limit
Facebook	63,206 chars	10	10,240 MB	14,400 sec	200 req / 3600 sec
Instagram	2,200 chars	10	10,240 MB	90 sec (reels)	25 posts / 24 hours
LinkedIn	3,000 chars	9	10,240 MB	—	Per-endpoint limits
TikTok	N/A (video only)	0	287.6 MB	—	Per-endpoint limits

4.4 Approval Chain Configuration

Creating an Approval Chain

1. Navigate to **Approval Chain** records (or use the App Launcher)
2. Click **New**
3. Configure:

Field	Description
Social Account	Link to specific account (leave blank for global chain)
Level 1 Approver	User who performs initial review
Level 1 Role	Role requirement for L1 (informational)
Level 1 SLA Hours	Maximum hours for L1 to respond
Level 2 Approver	(Optional) User for secondary review
Level 2 Role	Role requirement for L2
Level 2 SLA Hours	Maximum hours for L2 to respond
Level 3 Approver	(Optional) User for final review
Level 3 Role	Role requirement for L3
Level 3 SLA Hours	Maximum hours for L3 to respond
Level 3 Required When	Condition triggering L3 (e.g., "crisis topic", "external campaign")
Auto Escalation	Enable automatic escalation when SLA is exceeded
Escalation User	User who receives escalated posts
Is Active	Toggle this chain on/off

4. Click **Save**

How Approval Routing Works

1. When a post is submitted for approval, the system finds the applicable chain:
   - First checks for an active chain linked to the post's target `Social_Account__c`
   - Falls back to the first active global chain (no linked account)
2. Level 1 approver is notified
3. After L1 approval, L2 approver is notified (if configured)
4. After L2, L3 approver is notified (if the `Level_3_Required_When__c` condition is met)
5. Final approval moves the post to Scheduled or triggers immediate publish
6. If any level rejects, the post reverts to Draft with the rejection reason

4.5 Sentiment Threshold Configuration

Sentiment thresholds define automated actions triggered when incoming social posts match certain criteria.

Pre-Configured Thresholds

Threshold Name	Condition	Action	Case Priority
Crisis Signal	`Crisis_Flag = true`	crisis_alert	High
High Urgency Complaint	Sentiment=Negative, Urgency > 0.8	create_case	High
Influencer Negative	Influencer Score > 0.7, Sentiment=Negative	create_case	High
Toxicity Flag	Toxicity Score > 0.7	flag_for_review	Medium
Support Request	Intent = inquiry	create_case	Medium
Purchase Intent	Intent = compliment, Engagement > 0.8	sales_alert	Low
Negative Escalation	Sentiment=Negative, No Reply > 24h	escalate	High

Creating a Custom Threshold

1. Go to **Setup > Custom Metadata Types > Sentiment Threshold > Manage Records**
2. Click **New**
3. Configure:
   - **Threshold Name**: Descriptive name
   - **Condition Type**: `sentiment`, `toxicity`, `intent`, `crisis`, `engagement`
   - **Min Score / Max Score**: Numeric range for the condition
   - **Intent Value**: For intent-based conditions (e.g., `complaint`, `inquiry`)
   - **Action Type**: `create_case`, `crisis_alert`, `flag_for_review`, `sales_alert`, `escalate`
   - **Case Priority**: `High`, `Medium`, `Low`
   - **Is Active**: Toggle on/off
4. Click **Save**

Threshold Evaluation Flow

```
Social Post Ingested
  → Sentiment Scoring Queueable runs
    → Social_Sentiment__c record created
      → SocialSentimentTriggerHandler fires
```

```
          → SentimentService.evaluateThresholds()
            → For each active threshold:
                If condition matches → triggerAction()
                  → create_case: CaseCreationService.createCaseFromPost()
                  → crisis_alert: Crisis_Alert_Event__e published + Slack
notification
                  → flag_for_review: Post flagged
                  → sales_alert: Notification sent
                  → escalate: Case escalated
```

4.6 Slack Integration

Setting Up Slack Incoming Webhooks

1. Go to api.slack.com/apps
2. Create a new Slack App (or use an existing one)
3. Enable **Incoming Webhooks**
4. Click **Add New Webhook to Workspace**
5. Select the target channel
6. Copy the **Webhook URL**

Configuring Slack in SSN

1. Go to **Setup > Custom Metadata Types > SSN Slack Config > Manage Records**
2. Create records for each notification type:

Record Name	Notification Type	Webhook URL	Channel	Is Active
Approval Alerts	approval	`https://hooks.slack.com/services/...`	#content-approvals	true
Crisis Alerts	crisis	`https://hooks.slack.com/services/...`	#crisis-response	true
Publish Updates	publish	`https://hooks.slack.com/services/...`	#social-updates	true

Notification Format

- **Approval Requests**: Blue accent, includes post preview (200 char truncated) and approver name
- **Crisis Alerts**: Red accent, includes topic name, severity level, and details
- **Publish Notifications**: Green (success) or Red (failure), includes target platforms

4.7 Notification Preferences

Each user can configure their notification preferences per event type.

Event Types

Event	Description
new_post	New social post ingested
crisis_alert	Crisis-level sentiment detected
approval_ready	Post submitted for your approval
publish_success	Post successfully published
publish_failed	Post failed to publish
comment_reply	Reply to a brand post

Channels

Channel	Description
Email	Email notification
In-App	Salesforce in-app notification
Slack	Slack channel message (requires Slack config)

Users configure their preferences in **Settings > Notification Preferences** by toggling each channel on/off for each event type.

Part 5: Data Cloud Configuration for Social Listening

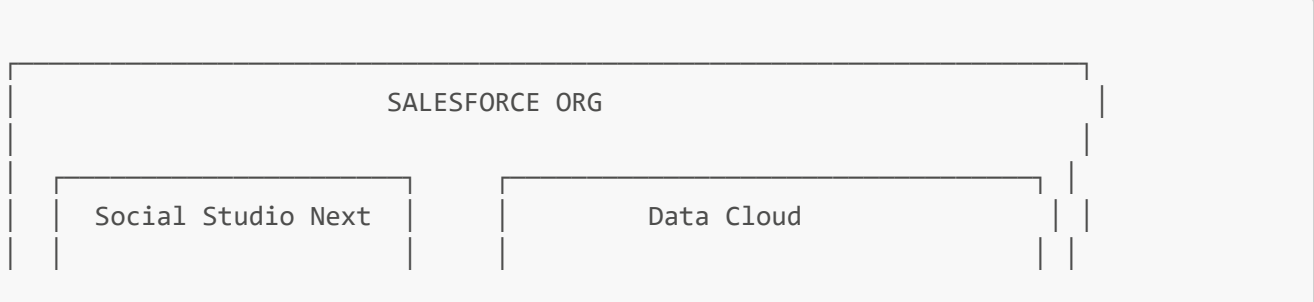
This section provides detailed instructions for configuring Salesforce Data Cloud to ingest social media data from Social Studio Next via the Ingestion API. This enables unified customer profiles, cross-platform identity resolution, and advanced social analytics.

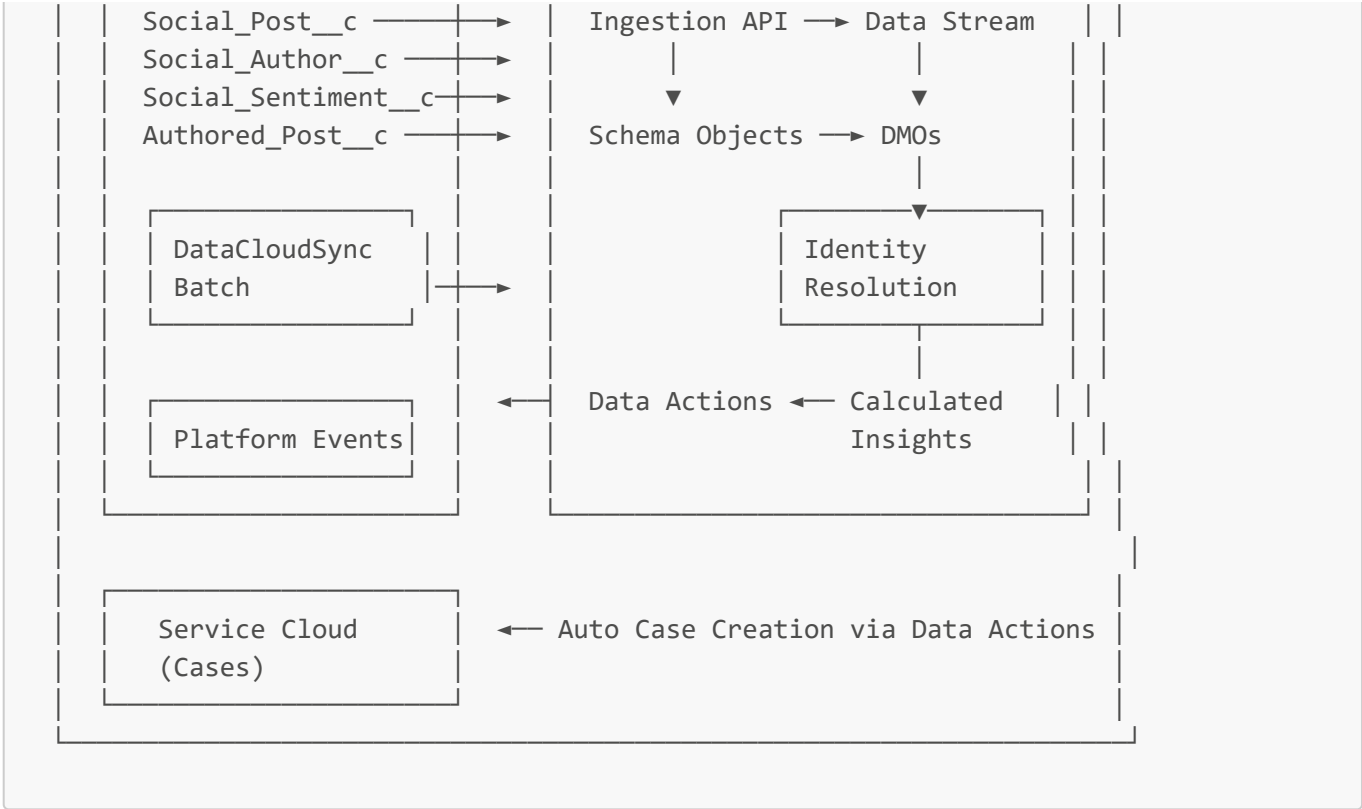
5.1 Prerequisites & Architecture Overview

Prerequisites

- **Salesforce Data Cloud license** — Data Cloud must be provisioned in your org
- **System Administrator** or **Data Cloud Architect** permission set
- **Social Studio Next** installed and configured with active social accounts
- **Connected App** configured for API access (see Section 5.2)

Architecture





Data Flow

1. **Social Studio Next** ingests social posts via platform APIs and scores sentiment
2. **DataCloudSyncBatch** periodically pushes SSN records to Data Cloud via the Ingestion API
3. **Data Cloud** maps ingested data to Data Model Objects (DMOs)
4. **Identity Resolution** links social authors to CRM contacts/leads via email, social handle, or external ID matching
5. **Calculated Insights** compute aggregated metrics (sentiment trends, engagement scores, share of voice)
6. **Data Actions** trigger Platform Events or Flows back in the CRM (e.g., crisis alerts, auto case creation)

5.2 Connected App Setup

Step 1: Create the Connected App

1. In Salesforce Setup, navigate to **App Manager**
2. Click **New Connected App**
3. Enter:
   - **Connected App Name**: `Data Cloud Social Ingestion`
   - **API Name**: `Data_Cloud_Social_Ingestion`
   - **Contact Email**: your admin email
4. Under **API (Enable OAuth Settings)**:
   - Check **Enable OAuth Settings**
   - **Callback URL**: `https://login.salesforce.com/services/oauth2/callback`
   - **Selected OAuth Scopes**:
     - `Access and manage your Data Cloud Ingestion API data (cdp_ingest_api)`
     - `Access and manage your data (api)`

    - `Perform requests on your behalf at any time (refresh_token, offline_access)`
5. Click **Save**
6. Note the **Consumer Key** and **Consumer Secret** (click to reveal after saving)

Step 2: Enable Client Credentials Flow

1. After saving, click **Manage** on the Connected App
2. Click **Edit Policies**
3. Under **OAuth Policies**:
    - Set **Permitted Users** to **Admin approved users are pre-authorized**
    - Check **Enable Client Credentials Flow**
    - Set **Run As** to an integration user with Data Cloud access
4. Click **Save**
5. Under **Manage Profiles** or **Manage Permission Sets**, add the appropriate Data Cloud permission sets

Step 3: Verify Permissions

The Run As user must have these permission sets assigned:

- `CdpCoreUser` or `CDPDataCloudUser`
- `DataCloudIngestionApiUser` (if available)

5.3 Authentication Flow

Authentication to the Data Cloud Ingestion API is a **two-step process**.

Step 1: Obtain Salesforce Access Token

Using Client Credentials Flow (recommended for server-to-server):

```
curl -X POST https://{myDomain}.my.salesforce.com/services/oauth2/token \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "grant_type=client_credentials" \
  -d "client_id={CONSUMER_KEY}" \
  -d "client_secret={CONSUMER_SECRET}"
```

Response:

```
{
  "access_token": "00D...!AQ...",
  "instance_url": "https://{myDomain}.my.salesforce.com",
  "id": "https://login.salesforce.com/id/{orgId}/{userId}",
  "token_type": "Bearer",
  "issued_at": "1708300000000"
}
```

Step 2: Exchange for Data Cloud Token

```
curl -X POST https://{myDomain}.my.salesforce.com/services/a360/token \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "grant_type=urn:salesforce:grant-type:external:cdp" \
  -d "subject_token={SALESFORCE_ACCESS_TOKEN}" \
  -d "subject_token_type=urn:ietf:params:oauth:token-type:access_token"
```

Response:

```
{
  "access_token": "{DATA_CLOUD_ACCESS_TOKEN}",
  "instance_url": "https://{tenant-id}.c360a.salesforce.com",
  "token_type": "Bearer",
  "issued_token_type": "urn:ietf:params:oauth:token-type:access_token",
  "expires_in": 7200
}
```

> **Important:** The `instance_url` from this response is your **tenant-specific base URL** for all Ingestion API calls. The `access_token` expires in 2 hours (7200 seconds).

Apex Implementation

```
public class DataCloudAuthService {

    private static final String TOKEN_ENDPOINT = '/services/oauth2/token';
    private static final String DC_TOKEN_ENDPOINT = '/services/a360/token';

    public static String getDataCloudToken(String consumerKey, String
consumerSecret) {
        // Step 1: Get Salesforce access token
        String baseUrl = URL.getOrgDomainUrl().toExternalForm();

        HttpRequest req = new HttpRequest();
        req.setEndpoint(baseUrl + TOKEN_ENDPOINT);
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/x-www-form-urlencoded');
        req.setBody(
            'grant_type=client_credentials'
            + '&client_id=' + EncodingUtil.urlEncode(consumerKey, 'UTF-8')
            + '&client_secret=' + EncodingUtil.urlEncode(consumerSecret, 'UTF-8')
        );

        Http http = new Http();
        HttpResponse res = http.send(req);
        Map<String, Object> tokenResponse = (Map<String, Object>)
JSON.deserializeUntyped(res.getBody());
        String sfAccessToken = (String) tokenResponse.get('access_token');
```

```
        // Step 2: Exchange for Data Cloud token
        HttpRequest dcReq = new HttpRequest();
        dcReq.setEndpoint(baseUrl + DC_TOKEN_ENDPOINT);
        dcReq.setMethod('POST');
        dcReq.setHeader('Content-Type', 'application/x-www-form-urlencoded');
        dcReq.setBody(
            'grant_type=urn:salesforce:grant-type:external:cdp'
            + '&subject_token=' + EncodingUtil.urlEncode(sfAccessToken, 'UTF-8')
            + '&subject_token_type=urn:ietf:params:oauth:token-type:access_token'
        );

        HttpResponse dcRes = http.send(dcReq);
        Map<String, Object> dcTokenResponse = (Map<String, Object>)
JSON.deserializeUntyped(dcRes.getBody());
        return (String) dcTokenResponse.get('access_token');
    }
}
```

5.4 Schema Definition

The Ingestion API requires an OpenAPI 3.0 YAML schema that defines the objects and fields being ingested.

Schema Constraints

Constraint	Limit
Max fields per object	1,000
Max object name length	80 characters
Max field name length	80 characters
Allowed characters	`a-z`, `A-Z`, `0-9`, `_`, `-`
Nested objects	Not allowed
Double underscores in field names	Not allowed
Object/field deletion after upload	Not supported
Field data type changes after upload	Not supported

Reserved Field Names (Do Not Use)

`date_id`, `location_id`, `dat_account_currency`, `dat_exchange_rate`, `pacing_period`, `pacing_end_date`, `row_count`, `version`

Data Type Mapping

Data Cloud Type	OpenAPI `type`	OpenAPI `format`	Example
Text	`string`	—	`"Hello World"`

Data Cloud Type	OpenAPI `type`	OpenAPI `format`	Example
Number (Integer)	`number`	`int32`	`42`
Number (Decimal)	`number`	`float`	`0.85`
Boolean	`boolean`	—	`true`
Date	`string`	`date`	`"2026-02-19"`
DateTime	`string`	`date-time`	`"2026-02-19T14:30:00.000Z"`
Email	`string`	`email`	`"user@example.com"`
URL	`string`	`url`	`"https://example.com"`
Phone	`string`	`phone`	`"+1-555-123-4567"`

Complete Schema File for Social Studio Next

Save this file as `ssn_ingestion_schema.yaml`:

```
openapi: "3.0.0"
info:
  title: Social Studio Next Ingestion Schema
  description: Schema for ingesting social media data into Salesforce Data Cloud
  version: "1.0.0"

components:
  schemas:

    # ─────────────────────────────────────────────
    # Social Post — Engagement category
    # Represents individual social media posts
    # ─────────────────────────────────────────────
    SocialPost:
      type: object
      required:
        - postId
        - platform
        - publishedDate
      properties:
        postId:
          type: string
          description: Unique platform-native post identifier
        platform:
          type: string
          description: "Source platform: x_twitter, facebook, instagram, linkedin,
tiktok"
        content:
          type: string
          description: Full text content of the post
        contentLanguage:
          type: string
```

```
      description: ISO 639-1 language code
    authorId:
      type: string
      description: Reference to SocialAuthor.authorId
    authorHandle:
      type: string
      description: Author's handle (e.g., @username)
    authorDisplayName:
      type: string
      description: Author's display name
    authorProfileUrl:
      type: string
      format: url
      description: URL to author's profile
    authorFollowerCount:
      type: number
      format: int32
      description: Author's follower count at time of post
    authorIsVerified:
      type: boolean
      description: Whether the author has a verified badge
    publishedDate:
      type: string
      format: date-time
      description: When the post was published (ISO 8601)
    permalink:
      type: string
      format: url
      description: Direct URL to the post on the platform
    postType:
      type: string
      description: "Type: post, image, video, link, poll, story, reel, thread"
    mediaType:
      type: string
      description: "Primary media type: text, image, video, carousel"
    isOwnedContent:
      type: boolean
      description: Whether this is the brand's own published content
    likeCount:
      type: number
      format: int32
      description: Number of likes/reactions
    shareCount:
      type: number
      format: int32
      description: Number of shares/retweets/reposts
    commentCount:
      type: number
      format: int32
      description: Number of comments/replies
    viewCount:
      type: number
      format: int32
      description: Number of views/impressions
```

```
        overallSentiment:
          type: string
          description: "Sentiment label: positive, negative, neutral, mixed"
        sentimentScore:
          type: number
          format: float
          description: Sentiment score from 0.0 (negative) to 1.0 (positive)
        matchedKeyword:
          type: string
          description: Listening topic keyword that matched this post
        listeningTopicId:
          type: string
          description: ID of the matched Listening Topic
        parentPostId:
          type: string
          description: ID of the parent post (for threading)
        caseId:
          type: string
          description: ID of the linked CRM Case
        dataCloudId:
          type: string
          description: Data Cloud record identifier
        modifiedDate:
          type: string
          format: date-time
          description: Last modification timestamp

    # ─────────────────────────────────────────────────
    # Social Author — Profile category
    # Represents social media user profiles
    # ─────────────────────────────────────────────────
    SocialAuthor:
      type: object
      required:
        - authorId
        - primaryPlatform
      properties:
        authorId:
          type: string
          description: Unique author identifier
        authorName:
          type: string
          description: Display name of the author
        primaryHandle:
          type: string
          description: Primary social media handle
        primaryPlatform:
          type: string
          description: "Primary platform: x_twitter, facebook, instagram,
linkedin, tiktok"
        profileUrl:
          type: string
          format: url
          description: URL to primary social profile
```

```
        email:
          type: string
          format: email
          description: Email address (if available from platform)
        isInfluencer:
          type: boolean
          description: Whether the author is flagged as an influencer
        followerCount:
          type: number
          format: int32
          description: Current follower count
        totalInteractions:
          type: number
          format: int32
          description: Total interactions with the brand
        averageSentiment:
          type: number
          format: float
          description: Average sentiment score across interactions
        lifetimeEngagementScore:
          type: number
          format: float
          description: Calculated lifetime engagement score
        firstInteractionDate:
          type: string
          format: date-time
          description: Date of first interaction with the brand
        lastInteractionDate:
          type: string
          format: date-time
          description: Date of most recent interaction
        crmContactId:
          type: string
          description: Linked Salesforce Contact ID
        crmLeadId:
          type: string
          description: Linked Salesforce Lead ID
        crmAccountId:
          type: string
          description: Linked Salesforce Account ID
        dataCloudId:
          type: string
          description: Data Cloud record identifier
        modifiedDate:
          type: string
          format: date-time
          description: Last modification timestamp

    # ─────────────────────────────────────────────
    # Social Sentiment — Other category
    # AI-generated sentiment analysis results
    # ─────────────────────────────────────────────
    SocialSentiment:
      type: object
```

```
        required:
          - sentimentId
          - postId
          - scoredDate
        properties:
          sentimentId:
            type: string
            description: Unique sentiment record identifier
          postId:
            type: string
            description: Reference to SocialPost.postId
          overallSentiment:
            type: string
            description: "Sentiment label: positive, negative, neutral, mixed"
          sentimentScore:
            type: number
            format: float
            description: Sentiment score 0.0-1.0
          sentimentConfidence:
            type: number
            format: float
            description: Model confidence 0.0-1.0
          intentClassification:
            type: string
            description: "Intent: complaint, compliment, inquiry, feedback,
  irrelevant"
          urgencyScore:
            type: number
            format: float
            description: Urgency level 0.0-1.0
          toxicityScore:
            type: number
            format: float
            description: Toxicity level 0.0-1.0
          crisisFlag:
            type: boolean
            description: Whether this post is flagged as a crisis
          emotionLabels:
            type: string
            description: JSON array of detected emotions
          extractedTopics:
            type: string
            description: JSON array of extracted topics
          extractedEntities:
            type: string
            description: JSON array of extracted entities
          influencerScore:
            type: number
            format: float
            description: Author influence rating 0.0-1.0
          modelVersion:
            type: string
            description: Sentiment model version identifier
          scoredDate:
```

```
        type: string
        format: date-time
        description: When the sentiment was analyzed
      dataCloudId:
        type: string
        description: Data Cloud record identifier
      modifiedDate:
        type: string
        format: date-time
        description: Last modification timestamp

  # ──────────────────────────────────────────────
  # Engagement Metric — Engagement category
  # Platform-reported engagement data per post
  # ──────────────────────────────────────────────
  EngagementMetric:
    type: object
    required:
      - engagementId
      - postId
      - metricDate
    properties:
      engagementId:
        type: string
        description: Unique engagement record identifier
      postId:
        type: string
        description: Reference to SocialPost.postId
      platform:
        type: string
        description: Platform this metric is from
      metricDate:
        type: string
        format: date-time
        description: When this metric was captured
      impressionCount:
        type: number
        format: int32
        description: Number of times content was displayed
      reachCount:
        type: number
        format: int32
        description: Unique users who saw the content
      clickCount:
        type: number
        format: int32
        description: Number of clicks on the content
      engagementRate:
        type: number
        format: float
        description: Engagement rate as a decimal (0.0-1.0)
      videoViewCount:
        type: number
        format: int32
```

```
      description: Video views (for video content)
    videoCompletionRate:
      type: number
      format: float
      description: Percentage of video watched (0.0-1.0)
    saveCount:
      type: number
      format: int32
      description: Number of saves/bookmarks
    profileVisitCount:
      type: number
      format: int32
      description: Profile visits attributed to this post
    dataCloudId:
      type: string
      description: Data Cloud record identifier
    modifiedDate:
      type: string
      format: date-time
      description: Last modification timestamp
```

5.5 Ingestion API Connector Setup

Step 1: Create the Connector

1. In Data Cloud, navigate to **Setup > Data Cloud Setup**
2. Under **Salesforce Integrations**, click **Ingestion API**
3. Click **New**
4. Enter the **Connector Name**: `SSN_Social_Ingest`
5. Click **Save**

Step 2: Upload the Schema

1. Click **Upload Schema**
2. Navigate to and select your `ssn_ingestion_schema.yaml` file
3. Preview the detected objects:
   - `SocialPost` (25+ fields)
   - `SocialAuthor` (18+ fields)
   - `SocialSentiment` (17+ fields)
   - `EngagementMetric` (13+ fields)
4. Click **Save**

The connector status will show **"Needs Data Stream"**. It changes to **"In Use"** once data streams are created and deployed.

Step 3: Note the Source API Name

After saving, note the **Source API Name** (e.g., `SSN_Social_Ingest`). This is used in the API endpoint URL:

```
POST https://{tenant-url}/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost
```

5.6 Data Stream Creation

Create a data stream for each schema object.

SocialPost Data Stream (Engagement)

1. Navigate to **Data Streams** in Data Cloud
2. Click **New** > **Ingestion API**
3. Select the `SSN_Social_Ingest` connector
4. Select **SocialPost** and click **Next**
5. Configure:
   - **Primary Key**: `postId`
   - **Category**: **Engagement**
   - **Event Time Field**: `publishedDate`
6. Click **Deploy**

SocialAuthor Data Stream (Profile)

1. Click **New** > **Ingestion API**
2. Select the `SSN_Social_Ingest` connector
3. Select **SocialAuthor** and click **Next**
4. Configure:
   - **Primary Key**: `authorId`
   - **Category**: **Profile**
   - **Record Modified Field**: `modifiedDate`
   - **Refresh Mode**: **Partial** (enables incremental updates)
5. Click **Deploy**

SocialSentiment Data Stream (Other)

1. Click **New** > **Ingestion API**
2. Select the `SSN_Social_Ingest` connector
3. Select **SocialSentiment** and click **Next**
4. Configure:
   - **Primary Key**: `sentimentId`
   - **Category**: **Other**
   - **Record Modified Field**: `modifiedDate`
5. Click **Deploy**

EngagementMetric Data Stream (Engagement)

1. Click **New** > **Ingestion API**
2. Select the `SSN_Social_Ingest` connector
3. Select **EngagementMetric** and click **Next**
4. Configure:

- **Primary Key**: `engagementId`
- **Category**: **Engagement**
- **Event Time Field**: `metricDate`

5. Click **Deploy**

> **Important:** The **Refresh Mode** for Profile and Other categories cannot be changed after creation. Choose **Partial** if you need incremental updates.

5.7 Data Mapping to DMOs

After data streams are deployed, map the ingested Data Source Objects (DSOs) to Data Model Objects (DMOs).

Map SocialAuthor to Contact Point Social (Standard DMO)

This maps social author profiles to the standard `ssot__ContactPointSocial__dlm` DMO, enabling identity resolution.

1. Go to **Data Model > Data Mapping**
2. Select the **SocialAuthor** data stream
3. Target DMO: **Contact Point Social**
4. Map fields:

Source Field (SocialAuthor)	Target Field (Contact Point Social)
`authorId`	`Id`
`primaryHandle`	`SocialHandleName`
`authorId`	`SocialHandleId`
`primaryPlatform`	`SocialNetworkProviderId`
`followerCount`	`FollowersCount`
`profileUrl`	`ProfilePictureURL`
`firstInteractionDate`	`ActiveFromDate`
`lastInteractionDate`	`ActiveToDate`

5. Also map to **Individual** DMO for identity resolution:

Source Field (SocialAuthor)	Target Field (Individual)
`authorId`	`Id`
`authorName`	`FirstName` (or use split logic)
`email`	(via Contact Point Email mapping)

Map SocialPost to Custom Social Post DMO

1. In Data Model, create a custom DMO: **SSN Social Post** (Category: Engagement)

2. Add fields matching the schema
3. Map all SocialPost data stream fields to the custom DMO fields
4. Set `publishedDate` as the **Event Time** dimension

Map SocialSentiment to Custom Sentiment DMO

1. Create a custom DMO: **SSN Social Sentiment** (Category: Other)
2. Add fields matching the schema
3. Map all SocialSentiment data stream fields to the custom DMO fields
4. Create a relationship to SSN Social Post via `postId`

Map EngagementMetric to Custom Engagement DMO

1. Create a custom DMO: **SSN Engagement Metric** (Category: Engagement)
2. Add fields matching the schema
3. Map all EngagementMetric data stream fields to the custom DMO fields
4. Set `metricDate` as the **Event Time** dimension
5. Create a relationship to SSN Social Post via `postId`

5.8 Identity Resolution

Identity Resolution links social authors to existing CRM contacts by matching identifiers across data sources.

Step 1: Create a Ruleset

1. Navigate to **Identity Resolution** in Data Cloud
2. Click **New Ruleset**
3. Name: `SSN_Author_Unification`
4. Description: "Links social media authors to CRM contacts and leads"
5. Click **Save**

Step 2: Configure Match Rules

Rule 1 — Email Match (High Confidence)

Setting	Value
Object	Contact Point Email
Match Type	Exact Normalized
Match On	Email Address
Description	Links authors who have an email matching an existing contact

Rule 2 — Social Handle Match

Setting	Value
Object	Contact Point Social
Match Type	Exact

Setting	Value
Match On	Social Handle Name + Social Network Provider
Description	Links known social handles across data sources

Rule 3 — Party Identification Match

Setting	Value
Object	Party Identification
Match Type	Exact
Match On	External ID (platform user ID)
Party Identification Type	Social
Description	Links by platform-native user IDs

Step 3: Configure Reconciliation Rules

When multiple sources disagree on a field value:

Field	Rule	Rationale
Name	Most Recent	Social profiles change names frequently
Email	Source Priority: CRM first	CRM data is verified
Follower Count	Most Recent	Follower counts change constantly
Platform	Most Frequent	Stabilize on the most-used platform

Step 4: Publish and Validate

1. Click **Publish** on the ruleset
2. Identity resolution runs within 24 hours
3. Validate results in **Profile Explorer**:
    - Search for a known contact
    - Verify linked social handles appear under their unified profile
    - Check for over-merging (multiple distinct people incorrectly merged)

5.9 Calculated Insights

Calculated Insights use SQL to compute aggregated metrics from Data Cloud data.

Sentiment Trend Analysis

```
-- Hourly sentiment trend across all social posts
SELECT
    DATE_FORMAT(ssn_SocialPost__dlm.publishedDate__c, 'yyyy-MM-dd HH:00:00') AS
time_bucket__c,
```

```
    ssn_SocialPost__dlm.platform__c AS platform__c,
    COUNT(ssn_SocialPost__dlm.postId__c) AS post_count__c,
    AVG(ssn_SocialPost__dlm.sentimentScore__c) AS avg_sentiment__c,
    SUM(CASE WHEN ssn_SocialPost__dlm.overallSentiment__c = 'positive' THEN 1 ELSE
0 END) AS positive_count__c,
    SUM(CASE WHEN ssn_SocialPost__dlm.overallSentiment__c = 'negative' THEN 1 ELSE
0 END) AS negative_count__c,
    SUM(CASE WHEN ssn_SocialPost__dlm.overallSentiment__c = 'neutral' THEN 1 ELSE
0 END) AS neutral_count__c
FROM
    ssn_SocialPost__dlm
WHERE
    ssn_SocialPost__dlm.publishedDate__c >= DATEADD(day, -30, CURRENT_TIMESTAMP)
GROUP BY
    time_bucket__c,
    platform__c
```

Engagement Scoring per Author

```
-- Calculate engagement score per unified individual
SELECT
    ssot__Individual__dlm.ssot__Id__c AS individual_id__c,
    SUM(ssn_SocialPost__dlm.likeCount__c
        + ssn_SocialPost__dlm.shareCount__c
        + ssn_SocialPost__dlm.commentCount__c) AS total_engagement__c,
    COUNT(ssn_SocialPost__dlm.postId__c) AS total_posts__c,
    AVG(ssn_SocialPost__dlm.sentimentScore__c) AS avg_sentiment__c,
    CASE
        WHEN SUM(ssn_SocialPost__dlm.likeCount__c
                 + ssn_SocialPost__dlm.shareCount__c
                 + ssn_SocialPost__dlm.commentCount__c) > 1000 THEN 'High Engager'
        WHEN SUM(ssn_SocialPost__dlm.likeCount__c
                 + ssn_SocialPost__dlm.shareCount__c
                 + ssn_SocialPost__dlm.commentCount__c) > 100 THEN 'Medium
Engager'
        ELSE 'Low Engager'
    END AS engagement_tier__c
FROM
    ssn_SocialPost__dlm
JOIN
    ssot__Individual__dlm
    ON ssn_SocialPost__dlm.authorId__c = ssot__Individual__dlm.ssot__Id__c
WHERE
    ssn_SocialPost__dlm.publishedDate__c >= DATEADD(day, -90, CURRENT_TIMESTAMP)
GROUP BY
    individual_id__c
```

Share of Voice Calculation

```
-- Share of voice by listening topic type
SELECT
    ssn_SocialPost__dlm.listeningTopicId__c AS topic_id__c,
    COUNT(ssn_SocialPost__dlm.postId__c) AS mention_count__c,
    ROUND(
        CAST(COUNT(ssn_SocialPost__dlm.postId__c) AS FLOAT)
        / CAST(SUM(COUNT(ssn_SocialPost__dlm.postId__c)) OVER () AS FLOAT)
        * 100, 2
    ) AS share_of_voice_pct__c,
    AVG(ssn_SocialPost__dlm.sentimentScore__c) AS avg_sentiment__c
FROM
    ssn_SocialPost__dlm
WHERE
    ssn_SocialPost__dlm.listeningTopicId__c IS NOT NULL
    AND ssn_SocialPost__dlm.publishedDate__c >= DATEADD(day, -30,
CURRENT_TIMESTAMP)
GROUP BY
    topic_id__c
```

Platform Comparison Metrics

```
-- Daily platform performance comparison
SELECT
    DATE_FORMAT(ssn_SocialPost__dlm.publishedDate__c, 'yyyy-MM-dd') AS
metric_date__c,
    ssn_SocialPost__dlm.platform__c AS platform__c,
    COUNT(ssn_SocialPost__dlm.postId__c) AS post_volume__c,
    AVG(ssn_SocialPost__dlm.sentimentScore__c) AS avg_sentiment__c,
    SUM(ssn_SocialPost__dlm.likeCount__c) AS total_likes__c,
    SUM(ssn_SocialPost__dlm.shareCount__c) AS total_shares__c,
    SUM(ssn_SocialPost__dlm.commentCount__c) AS total_comments__c,
    SUM(ssn_SocialPost__dlm.viewCount__c) AS total_views__c,
    AVG(
        CAST(ssn_SocialPost__dlm.likeCount__c
             + ssn_SocialPost__dlm.shareCount__c
             + ssn_SocialPost__dlm.commentCount__c AS FLOAT)
        / NULLIF(ssn_SocialPost__dlm.viewCount__c, 0)
    ) AS avg_engagement_rate__c
FROM
    ssn_SocialPost__dlm
WHERE
    ssn_SocialPost__dlm.publishedDate__c >= DATEADD(day, -30, CURRENT_TIMESTAMP)
GROUP BY
    metric_date__c,
    platform__c
```

Crisis Detection Scoring

```
-- Real-time crisis score aggregation
SELECT
    ssn_SocialSentiment__dlm.postId__c AS post_id__c,
    ssn_SocialSentiment__dlm.overallSentiment__c AS sentiment__c,
    ssn_SocialSentiment__dlm.sentimentScore__c AS sentiment_score__c,
    ssn_SocialSentiment__dlm.urgencyScore__c AS urgency__c,
    ssn_SocialSentiment__dlm.toxicityScore__c AS toxicity__c,
    ssn_SocialSentiment__dlm.crisisFlag__c AS is_crisis__c,
    CASE
        WHEN ssn_SocialSentiment__dlm.crisisFlag__c = true THEN 'Critical'
        WHEN ssn_SocialSentiment__dlm.toxicityScore__c > 0.7 THEN 'High'
        WHEN ssn_SocialSentiment__dlm.urgencyScore__c > 0.8
             AND ssn_SocialSentiment__dlm.overallSentiment__c = 'negative' THEN
'High'
        WHEN ssn_SocialSentiment__dlm.overallSentiment__c = 'negative' THEN
'Medium'
        ELSE 'Low'
    END AS crisis_severity__c
FROM
    ssn_SocialSentiment__dlm
WHERE
    ssn_SocialSentiment__dlm.scoredDate__c >= DATEADD(hour, -24,
CURRENT_TIMESTAMP)
```

5.10 Data Actions

Data Actions send near real-time events from Data Cloud to trigger automations.

Crisis Alert Data Action (Platform Event)

Step 1: Ensure Platform Event Exists

Social Studio Next includes the `Crisis_Alert_Event__e` platform event with fields:

- `Topic_Id__c` (Text)
- `Alert_Type__c` (Text)
- `Severity__c` (Text)
- `Details__c` (Long Text)

Step 2: Create the Data Action

1. In Data Cloud, go to the **Data Actions** tab
2. Click **New**
3. Select target type: **Salesforce Platform Event**
4. Select your Data Space
5. Source: **SSN Social Sentiment** (custom DMO) or the Crisis Detection Calculated Insight
6. Trigger condition: `crisisFlag = true` OR `crisis_severity = 'Critical'`
7. Map fields:

DMO/CIO Field	Platform Event Field

DMO/CIO Field	Platform Event Field
`postId`	`Topic_Id__c`
`crisis_severity`	`Severity__c`
`sentiment + urgency + toxicity`	`Details__c`

8. Click **Save** and **Activate**

Auto Case Creation (Data Cloud-Triggered Flow)

1. In Salesforce Setup, create a **Data Cloud-Triggered Flow**
2. Set the trigger:
   - Object: **SSN Social Sentiment** DMO
   - Condition: `overallSentiment = 'negative'` AND `urgencyScore > 0.8`
3. Add Flow actions:
   - **Create Record**: Case
     - Subject: "Social Media Alert - High Urgency"
     - Priority: High
     - Origin: Social Media
     - Description: Include post content and sentiment details
   - **Publish Platform Event**: `Social_Post_Event__e` (to notify SSN UI)
4. Activate the Flow

5.11 Streaming vs Bulk Ingestion

When to Use Each Pattern

Pattern	Use Case	Latency	Volume
Streaming	Real-time post ingestion, live monitoring	~3 min (micro-batch)	Up to 250 req/sec
Bulk	Historical data backfill, batch sync	Minutes to hours	Up to 15 GB per job

Both patterns can operate on the same data stream simultaneously.

Rate Limits

Limit	Streaming	Bulk
Payload size	200 KB per request	150 MB per CSV file
Max records per request	Limited by 200 KB	Unlimited (within file size)
Concurrent requests	5	20 jobs per hour
Requests per second	250 (across all endpoints)	N/A
Delete records per request	200	N/A
CSV files per job	N/A	100

Limit	Streaming	Bulk
Throttling	HTTP 429	HTTP 429

Streaming Insert Example

```
# Insert social posts via streaming API
curl -X POST \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost" \
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{
    "data": [
      {
        "postId": "tw_1234567890",
        "platform": "x_twitter",
        "content": "Just tried @YourBrand new product and absolutely love it!
#review",
        "contentLanguage": "en",
        "authorId": "auth_tw_98765",
        "authorHandle": "@happy_customer",
        "authorDisplayName": "Happy Customer",
        "authorFollowerCount": 1250,
        "authorIsVerified": false,
        "publishedDate": "2026-02-19T14:30:00.000Z",
        "permalink": "https://x.com/happy_customer/status/1234567890",
        "postType": "post",
        "mediaType": "text",
        "isOwnedContent": false,
        "likeCount": 42,
        "shareCount": 12,
        "commentCount": 5,
        "viewCount": 2100,
        "overallSentiment": "positive",
        "sentimentScore": 0.92,
        "matchedKeyword": "YourBrand",
        "modifiedDate": "2026-02-19T14:30:00.000Z"
      },
      {
        "postId": "tw_1234567891",
        "platform": "x_twitter",
        "content": "Terrible experience with @YourBrand support. 3 hours on
hold!",
        "contentLanguage": "en",
        "authorId": "auth_tw_11111",
        "authorHandle": "@frustrated_user",
        "authorDisplayName": "Frustrated User",
        "authorFollowerCount": 500,
        "authorIsVerified": false,
        "publishedDate": "2026-02-19T14:35:00.000Z",
        "permalink": "https://x.com/frustrated_user/status/1234567891",
```

```
        "postType": "post",
        "mediaType": "text",
        "isOwnedContent": false,
        "likeCount": 8,
        "shareCount": 3,
        "commentCount": 15,
        "viewCount": 800,
        "overallSentiment": "negative",
        "sentimentScore": 0.15,
        "matchedKeyword": "YourBrand",
        "modifiedDate": "2026-02-19T14:35:00.000Z"
      }
    ]
  }'
```

Response: `202 Accepted` — Data is queued for processing (~3 minute micro-batch cycle).

Bulk Insert Example

Step 1: Create a Bulk Job

```
curl -X POST \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost/jobs"
\
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{
    "object": "SocialPost",
    "operation": "upsert"
  }'
```

Response:

```
{
  "id": "750xx000000000001",
  "operation": "upsert",
  "object": "SocialPost",
  "state": "Open",
  "createdDate": "2026-02-19T15:00:00.000Z"
}
```

Step 2: Upload CSV Data

```
curl -X PUT \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost/jobs/7
```

```
50xx000000000001/batches" \
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}" \
  -H "Content-Type: text/csv" \
  --data-binary @social_posts_batch.csv
```

CSV Format:

```
postId,platform,content,authorId,authorHandle,publishedDate,likeCount,shareCount,c
ommentCount,viewCount,overallSentiment,sentimentScore,modifiedDate
tw_001,x_twitter,"Great product @YourBrand!",auth_001,@user1,2026-02-
19T10:00:00.000Z,50,10,5,1000,positive,0.88,2026-02-19T10:00:00.000Z
tw_002,x_twitter,"@YourBrand needs better support",auth_002,@user2,2026-02-
19T11:00:00.000Z,3,1,8,200,negative,0.25,2026-02-19T11:00:00.000Z
fb_001,facebook,"Love the new features from YourBrand",auth_003,user3,2026-02-
19T12:00:00.000Z,120,30,15,5000,positive,0.91,2026-02-19T12:00:00.000Z
```

Step 3: Close the Job

```
curl -X PATCH \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost/jobs/7
50xx000000000001" \
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{"state": "UploadComplete"}'
```

Step 4: Check Job Status

```
curl -X GET \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost/jobs/7
50xx000000000001" \
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}"
```

Response when complete:

```
{
  "id": "750xx000000000001",
  "operation": "upsert",
  "object": "SocialPost",
  "state": "JobComplete",
  "numberRecordsProcessed": 3,
  "numberRecordsFailed": 0
}
```

Delete Records Example

```
curl -X DELETE \
  "https://{tenant-
id}.c360a.salesforce.com/api/v1/ingest/sources/SSN_Social_Ingest/SocialPost" \
  -H "Authorization: Bearer {DATA_CLOUD_ACCESS_TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{
    "data": [
      {"postId": "tw_1234567890"},
      {"postId": "tw_1234567891"}
    ]
  }'
```

> **Limit:** Maximum 200 records per delete request.

Apex Streaming Ingestion Example

```
public class DataCloudIngestionService {

    @future(callout=true)
    public static void ingestSocialPosts(Set<Id> postIds) {
        List<rissnext__Social_Post__c> posts = [
            SELECT rissnext__Platform_Native_Id__c, rissnext__Platform__c,
                   rissnext__Content__c, rissnext__Content_Language__c,
                   rissnext__Author__c, rissnext__Author_Handle__c,
                   rissnext__Author_Display_Name__c,
rissnext__Author_Follower_Count__c,
                   rissnext__Author_Is_Verified__c,
rissnext__Published_Datetime__c,
                   rissnext__Permalink__c, rissnext__Post_Type__c,
                   rissnext__Is_Owned_Content__c,
                   rissnext__Like_Count__c, rissnext__Share_Count__c,
                   rissnext__Comment_Count__c, rissnext__View_Count__c,
                   rissnext__Overall_Sentiment__c, rissnext__Sentiment_Score__c,
                   rissnext__Matched_Keyword__c, rissnext__Listening_Topic__c,
                   rissnext__Data_Cloud_Id__c
            FROM rissnext__Social_Post__c
            WHERE Id IN :postIds
        ];

        if (posts.isEmpty()) return;

        List<Map<String, Object>> payload = new List<Map<String, Object>>();
        for (rissnext__Social_Post__c post : posts) {
            Map<String, Object> record = new Map<String, Object>{
                'postId' => post.rissnext__Platform_Native_Id__c,
                'platform' => post.rissnext__Platform__c,
                'content' => post.rissnext__Content__c,
                'contentLanguage' => post.rissnext__Content_Language__c,
```

```
                'authorHandle' => post.rissnext__Author_Handle__c,
                'authorDisplayName' => post.rissnext__Author_Display_Name__c,
                'authorFollowerCount' => post.rissnext__Author_Follower_Count__c,
                'authorIsVerified' => post.rissnext__Author_Is_Verified__c,
                'publishedDate' => post.rissnext__Published_Datetime__c != null
                    ? post.rissnext__Published_Datetime__c.formatGmt('yyyy-MM-
dd\'T\'HH:mm:ss.SSS\'Z\'')
                    : null,
                'permalink' => post.rissnext__Permalink__c,
                'postType' => post.rissnext__Post_Type__c,
                'isOwnedContent' => post.rissnext__Is_Owned_Content__c,
                'likeCount' => post.rissnext__Like_Count__c,
                'shareCount' => post.rissnext__Share_Count__c,
                'commentCount' => post.rissnext__Comment_Count__c,
                'viewCount' => post.rissnext__View_Count__c,
                'overallSentiment' => post.rissnext__Overall_Sentiment__c,
                'sentimentScore' => post.rissnext__Sentiment_Score__c,
                'matchedKeyword' => post.rissnext__Matched_Keyword__c,
                'modifiedDate' => Datetime.now().formatGmt('yyyy-MM-
dd\'T\'HH:mm:ss.SSS\'Z\'')
            };
            payload.add(record);
        }

        Map<String, Object> body = new Map<String, Object>{ 'data' => payload };
        String jsonBody = JSON.serialize(body);

        // Use the Data Cloud token obtained via DataCloudAuthService
        String dcToken = DataCloudAuthService.getDataCloudToken(
            'YOUR_CONSUMER_KEY', 'YOUR_CONSUMER_SECRET'
        );

        HttpRequest req = new HttpRequest();

req.setEndpoint('callout:SSN_DataCloud/api/v1/ingest/sources/SSN_Social_Ingest/Soc
ialPost');
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');
        req.setHeader('Authorization', 'Bearer ' + dcToken);
        req.setBody(jsonBody);
        req.setTimeout(30000);

        Http http = new Http();
        HttpResponse res = http.send(req);

        if (res.getStatusCode() != 202) {
            System.debug(LoggingLevel.ERROR,
                'Data Cloud ingestion failed: ' + res.getStatusCode() + ' ' +
res.getBody()
            );
        }
    }
}
```

5.12 Best Practices

Schema Design

- Plan your schema carefully before uploading — objects and fields **cannot be removed** once added
- Include a `modifiedDate` datetime field on all objects for incremental update support
- For Engagement-category objects, always include an Event Time datetime field
- Use clear, descriptive field names (no double underscores, no reserved names)

Data Quality

- Validate data before ingestion — invalid dates or mismatched types cause record-level failures
- Use ISO 8601 formats strictly: dates as `yyyy-MM-dd`, datetimes as `yyyy-MM-dd'T'HH:mm:ss.SSS'Z'`
- Ensure primary keys are truly unique to prevent unintended overwrites

Performance

- Keep streaming payloads well under the 200 KB limit
- Batch records efficiently within each request to minimize total API calls
- For large historical imports, use bulk ingestion over streaming
- Implement exponential backoff retry logic for HTTP 429 (rate limit) responses

Identity Resolution

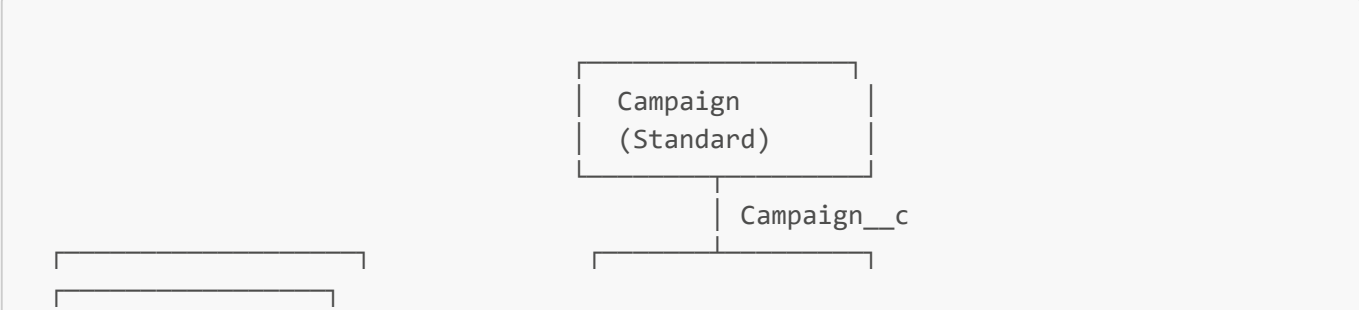
- Start with high-confidence match rules (exact email, exact external ID) before adding fuzzy matching
- Validate unified profiles in Profile Explorer after each rule change
- Monitor for over-merging (too many records being combined into one profile)
- Add additional match rules incrementally after measuring match outcomes

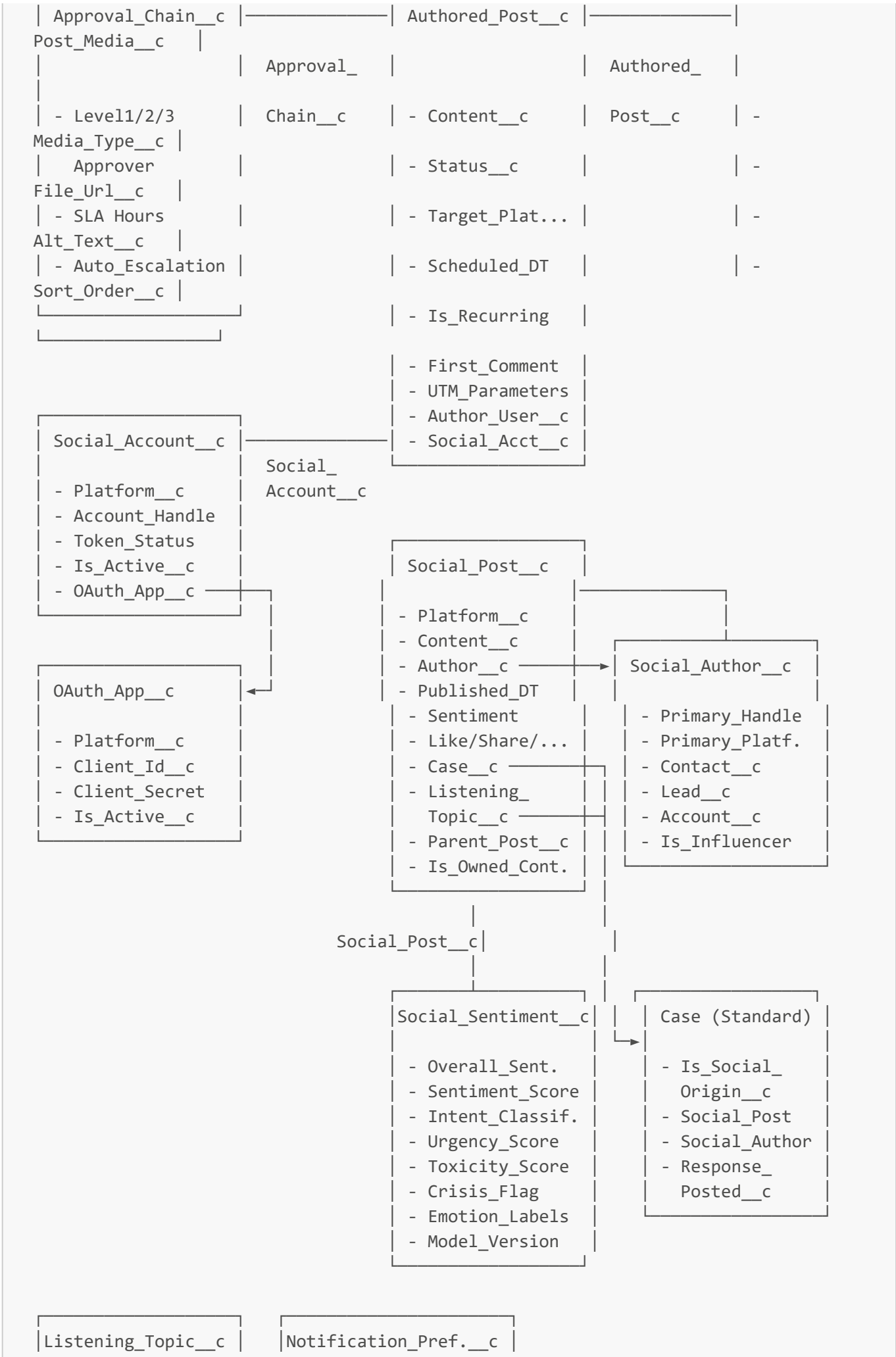
Monitoring

- Use the **Data Explorer** tab in Data Cloud to verify ingested records
- Check data stream health and processing status regularly
- Set up alerting at 80% of API capacity to avoid hitting hard limits during peak usage
- Review Data Cloud audit logs for ingestion errors

Part 6: Data Model Reference

Object Relationship Diagram




Approval_Chain__c
Approval_
Chain__c
- Level1/2/3 Approver
- SLA Hours
- Auto_Escalation
Authored_Post__c
Approval_
Chain__c
- Content__c
- Status__c
- Target_Plat...
- Scheduled_DT
- Is_Recurring
- First_Comment
- UTM_Parameters
- Author_User__c
- Social_Acct__c
Authored_
Post__c
Post_Media__c
Media_Type__c
File_Url__c
Alt_Text__c
Sort_Order__c
Social_Account__c
Social_
Account__c
- Platform__c
- Account_Handle
- Token_Status
- Is_Active__c
- OAuth_App__c
OAuth_App__c
- Platform__c
- Client_Id__c
- Client_Secret
- Is_Active__c
Social_Post__c
- Platform__c
- Content__c
- Author__c
- Published_DT
- Sentiment
- Like/Share/...
- Case__c
- Listening_Topic__c
- Parent_Post__c
- Is_Owned_Cont.
Social_Author__c
- Primary_Handle
- Primary_Platf.
- Contact__c
- Lead__c
- Account__c
- Is_Influencer
Social_Post__c
Social_Sentiment__c
- Overall_Sent.
- Sentiment_Score
- Intent_Classif.
- Urgency_Score
- Toxicity_Score
- Crisis_Flag
- Emotion_Labels
- Model_Version
Case (Standard)
- Is_Social_Origin__c
- Social_Post
- Social_Author
- Response_Posted__c
Listening_Topic__c
Notification_Pref.__c

```
  │                     │   │                         │
  │ - Keywords__c       │   │ - User__c               │
  │ - Exclude_KW__c     │   │ - Event_Type__c         │
  │ - Platforms__c      │   │ - Email_Enabled__c      │
  │ - Topic_Type__c     │   │ - InApp_Enabled__c      │
  │ - Is_Active__c      │   │ - Slack_Enabled__c      │
  │ - Alert_On_Crisis   │   └─────────────────────────┘
  └─────────────────────┘
```

Custom Metadata Types

CMDT	Purpose	Key Fields
`Platform_Config__mdt`	Platform specs	Max_Text_Length, Max_Images, Max_Video_Size, Rate_Limit, API_Base_Url
`Sentiment_Threshold__mdt`	Automated actions	Condition_Type, Min/Max_Score, Action_Type, Case_Priority
`SSN_OAuth_Config__mdt`	OAuth credentials	Client_Id, Client_Secret (per platform)
`SSN_Slack_Config__mdt`	Slack webhooks	Webhook_Url, Channel, Notification_Type, Is_Active

Platform Events

Event	Purpose	Key Fields
`Social_Post_Event__e`	Real-time post notifications	Post_Id, Platform, Sentiment, Action_Type, Payload
`Publishing_Event__e`	Publish status updates	Authored_Post_Id, Status, Platform, Message
`Crisis_Alert_Event__e`	Crisis detection alerts	Topic_Id, Alert_Type, Severity, Details

Scheduled Jobs

Job Class	Schedule	Purpose
`PostPublishSchedulable`	Every 5 min (configurable)	Publishes due scheduled posts
`RecurringPostSchedulable`	Hourly	Clones recurring posts for next occurrence
`TokenRefreshSchedulable`	Daily at 2 AM	Refreshes expiring OAuth tokens
`SocialPostIngestionBatch`	Hourly	Polls platform APIs for new social posts
`EngagementMetricsBatch`	Every 6 hours	Updates engagement metrics for owned posts

Job Class	Schedule	Purpose
`DataCloudSyncBatch`	Hourly	Syncs SSN records to Data Cloud via Ingestion API
`ApprovalEscalationSchedulable`	Hourly	Escalates overdue approval requests
`SentimentScoringQueueable`	On-demand (trigger)	Scores sentiment for new social posts

Part 7: Troubleshooting

Posts Failing to Publish

Symptom	Possible Cause	Solution
Status stuck at "publishing"	Batch job not running	Verify `PostPublishSchedulable` is scheduled in Setup > Scheduled Jobs
Status = "failed"	Check `Publish_Error__c` field	Error message contains the platform API error
"Token expired" error	OAuth token needs refresh	Go to Settings > Connected Accounts > Reconnect
"Rate limit exceeded"	Too many API calls	Wait for the rate limit window to reset (check Platform_Config__mdt)
"Unsupported media format"	Wrong file type	Check platform media constraints in Section 4.3
"Account inactive"	Social account disabled	Go to Settings > Connected Accounts > Activate

Sentiment Not Calculated

Symptom	Possible Cause	Solution
No Social_Sentiment__c record	Queueable didn't fire	Check if `SocialPostTriggerHandler` is active; verify trigger deployment
Sentiment score = 0	Content too short for analysis	Minimum content needed for keyword matching
Model version mismatch	Outdated scoring model	Verify `SSNSentimentScoringService` is current version

Cases Not Auto-Creating

Symptom	Possible Cause	Solution
Crisis posts don't create cases	Threshold not active	Verify `Sentiment_Threshold__mdt` has `Is_Active__c = true`
Wrong priority on cases	Threshold misconfigured	Check `Case_Priority__c` on the matching threshold
No contact linked to case	Author not linked to contact	Link Social_Author__c to Contact via the author detail page

Token Expiration

Symptom	Possible Cause	Solution
Token_Status = "expired"	Refresh token failed	Re-authenticate: Settings > Connected Accounts > Reconnect
Frequent token issues	Platform app permissions changed	Verify scopes on the platform developer console
Token refresh failing	Named Credential misconfigured	Check Named Credential settings in Salesforce Setup

Data Cloud Ingestion Issues

Symptom	Possible Cause	Solution
No data in Data Explorer	Data stream not deployed	Verify data stream status is "Active" in Data Cloud
HTTP 401 on ingestion	Token expired	Re-authenticate via the two-step token exchange
HTTP 429 throttling	Rate limit exceeded	Implement exponential backoff; reduce request frequency
Records failing validation	Invalid date format	Ensure all dates use ISO 8601: `yyyy-MM-dd'T'HH:mm:ss.SSS'Z'`
Schema upload fails	Invalid YAML	Validate YAML syntax; check field name constraints (no __)
Identity resolution not matching	Match rules too strict	Start with exact normalized; add fuzzy rules incrementally
Over-merging profiles	Match rules too loose	Remove or tighten fuzzy match rules; validate in Profile Explorer

Common Error Messages

Error	Meaning	Resolution

Error	Meaning	Resolution
`field 'Content__c' can not be filtered in a query call`	Long Text Area used in WHERE clause	Remove Long Text Area fields from SOQL WHERE/ORDER BY clauses
`Constructor not defined: [MockHttpCallout]`	Wrong constructor parameters in test	Check MockHttpCallout signature: `(Integer statusCode, String responseBody)`
`FIELD_CUSTOM_VALIDATION_EXCEPTION`	Validation rule blocking DML	Check validation rules on the target object
`UNABLE_TO_LOCK_ROW`	Record lock contention	Retry the operation; reduce batch sizes
`System.CalloutException: Unauthorized endpoint`	Named Credential not configured	Add the endpoint to Remote Site Settings or configure the Named Credential

This guide covers Social Studio Next version 1.0. For the latest updates, refer to the release notes included with each package version.