

SourceLace manual

Setting up SourceLace, connecting your systems, and how your data is protected

SourceLace version 0.1.0 · Made October 4, 2026

The latest version is at <https://sourcelace.onrender.com/docs>

Contents

Start here

- Getting started
- Admin guide
- Single sign-on (SSO)
- Plans and limits

Connect your systems

- Connecting your systems
 - Salesforce, Data 360 and Marketing Cloud
 - SAP S/4HANA, SuccessFactors and ECC
 - HubSpot
 - Mail and calendar
 - Files: Google Drive, SharePoint and OneDrive, Amazon S3
 - Service desks: ServiceNow, Zendesk and Jira
 - Data platforms: Snowflake, BigQuery and Databricks
 - Business apps: Dynamics 365, NetSuite and Gong
 - Databases: PostgreSQL, MySQL, SQL Server, Oracle Database and MongoDB
 - Oracle Fusion Cloud and Workday
- Build your own connector
- Google Sheets
- SAP ECC (on-premise)
- Slack and Microsoft Teams

Security and answers

- Security and privacy
- Frequently asked questions

Getting started

SourceLace lets the people in your organization ask questions of your business systems, such as Salesforce, SAP, HubSpot, Snowflake or your own databases, from the AI app they already use, with their own permissions in each system.

What SourceLace is

SourceLace sits between an AI app and your business systems. When someone asks "Which renewals over \$50,000 close this quarter?", their AI app asks SourceLace, SourceLace runs a read-only query against the system **as that person**, and the answer comes back with the rows it used.

Three things are true for every request:

- **Each person uses their own login.** People connect each system with their own account, so the system's own permissions, roles and sharing rules decide what they see. SourceLace never uses a shared admin account where per-person sign-in exists.
- **Reading is the default.** Queries are checked before anything is sent, and anything that could change data is refused. Changes are possible only where your admin turns them on, and only after the person sees a preview and confirms it.
- **Nothing is copied into a database.** Results are held in memory for up to 30 minutes, so they can be combined or charted, and are then gone. Every call is written to your organization's audit trail (what was asked, never the data that came back).

Two ways to use it

You can use SourceLace from any AI app that supports the Model Context Protocol (MCP), and from the SourceLace app. Both use the same sources, the same permissions and the same audit trail.

From your own AI app (MCP)

SourceLace works with Claude, ChatGPT, Cursor, VS Code, and any other MCP client, including an organization's own agent on its own model. Add SourceLace as a connector (some apps call it a "custom connector" or an "MCP server") with this address:

`https://sourcelace.onrender.com/mcp`

The app opens a browser window the first time, so you can sign in to SourceLace (with Google, Microsoft, or your organization's single sign-on). After that it stays signed in.

Where to add it, in the apps people use most (menu names change from time to time):

App	Where
Claude (claude.ai, desktop, mobile)	Settings → Connectors → Add custom connector. Name it SourceLace, paste the address, leave the advanced OAuth fields empty, then Connect . On Team and Enterprise plans an owner may need to add it first under Admin settings → Connectors .

App	Where
ChatGPT	Turn on Developer mode under Settings → Apps & Connectors → Advanced settings , then Create a connector with the address and OAuth authentication.
Cursor	Cursor Settings → MCP → Add new MCP server , with the address as its url. Click Login next to SourceLace.
VS Code	Run MCP: Add Server from the Command Palette, choose HTTP , and paste the address.
Any other MCP client	Use the address above. SourceLace follows the MCP authorization standard: the client registers itself and signs you in with PKCE.

When SourceLace is connected, the AI app sees SourceLace's tools (such as `list_sources`, `query` and `search_docs`). SourceLace itself sends nothing to any AI model in this setup: your AI app and its provider decide what happens to the answers, under your agreement with them.

From the SourceLace app

Open [the SourceLace app](#) and sign in. The app has a chat assistant that uses the same tools, plus pages for your sources, skills, projects and status. Admins also manage people, sources, groups, settings and the audit trail there. See [Admin guide](#).

The app's assistant is powered by Claude, through Anthropic's API, unless your admin has entered your organization's own Anthropic or OpenAI key (see [AI settings](#)).

Your first five minutes

1. **Sign in** to SourceLace, in the app or by adding the address above to your AI app. If you are told you are not allowed in, your organization's SourceLace admin needs to invite you or add your email domain.
2. **See your sources.** Ask "Which SourceLace sources do I have?" (the `list_sources` tool), or open **Data Sources** in the app. Each source shows whether you have connected it.
3. **Connect a source.** Ask "Connect salesforce:prod", or click **Connect** on the Data Sources page. A sign-in page opens for that system; sign in with your own account there. Databases open SourceLace's own sign-in form instead, for your own database login.
4. **Ask a question.** For example: "Show my open opportunities over 50,000, newest first." The AI looks up the fields first, writes a query in the system's own language, and shows you the rows.
5. **Combine and share.** Ask to join two results (for example Salesforce accounts with SAP customers), draw a chart, or, in the SourceLace app, make a Word, PowerPoint or PDF file from the results.

Changing records

Some sources can create or update records (and a few can delete them: Salesforce, HubSpot, and the email drafts SourceLace made), but only:

1. where your admin has turned on **Allow changes** for that source and listed the objects that can change;

2. after the AI shows you a **preview** with each field's current and new value; and
3. after **you confirm**. In the SourceLace app you click **Apply** on the preview card. In other AI apps, SourceLace asks you directly when the app can show its confirmation box; otherwise the preview includes a six-letter code that you type into the chat.

A preview expires after 10 minutes. If someone changed the record after the preview, SourceLace refuses and asks for a fresh preview. The source's own permissions still apply to every change.

Skills

Your organization can write **skills**: step-by-step how-tos the AI follows for a task, such as building a QBR deck or a pipeline review. Admins share skills with everyone, and anyone can keep their own. Skills work in the SourceLace app (the **Skills** page) and in every MCP client (the `list_skills` and `get_skill` tools). A skill cannot widen what anyone may read or change.

Asking about SourceLace itself

This manual is built into SourceLace. Ask your AI app or the SourceLace assistant questions such as "How do I connect Snowflake?" or "How long does SourceLace keep my data?": it searches this manual with the `search_docs` tool and answers with a link to the page. You can also download any page, or the whole manual, as a PDF from [the docs site](#).

Which systems SourceLace connects to

See [Connecting your systems](#) for the full list, who sets each one up, and what people can read and change in each.

Admin guide

For your organization's SourceLace admins: who can sign in, single sign-on, groups and access by group, sources, turning on changes, AI settings, the audit trail and how long things are kept.

Everything here is done in the [SourceLace app](#) by someone with the **admin** role. Most of it can also be done by asking an MCP client in plain words, which calls SourceLace's admin tools (`admin_people`, `admin_sources` and `admin_groups`). Single sign-on is set up in the app only, so a client secret never has to be typed into a chat. Every admin change is recorded in your organization's audit trail.

Roles

Role	Can
Member	Connect sources with their own logins, ask questions, propose changes (where allowed), keep their own skills, chats and projects.
Admin	Everything a member can, plus manage people, email domains, sources, groups, single sign-on, settings, shared skills, and read the audit trail, usage and invoices.

Every organization always keeps at least one admin: SourceLace refuses to remove or demote the last one.

What your organization can use (seats, connectors, AI questions, and features such as access by group) depends on its plan. See [Plans and limits](#).

People and sign-in

People sign in to SourceLace with **Google**, **Microsoft**, or your organization's own identity provider through [single sign-on](#). SourceLace never sees their password: it receives a confirmed email address from the identity provider and checks it against your organization's people.

There are two ways someone gets in:

- **Invited.** On the **Team** page, under **Invite someone**, enter their email address and choose **member** or **admin**. From chat: "Add bo@corvanta.com to SourceLace as a member". Invited people stay in even if you later remove the email domain they are at.
- **By email domain.** Under **Email domains**, list your domains (such as `corvanta.com`). Anyone who signs in with a verified address at one of them joins as a **member**. Public domains such as `gmail.com` are refused. If you remove a domain, the people who joined through it lose access.

Each person is one seat. The Team page shows how many seats are used of your plan's maximum; on Starter and Team, nobody new can join once it is reached (see [Seats](#)).

The Team page shows each person's role, how they joined (invite, domain or single sign-on) and when they last signed in. To change a role, pick it next to the person; to remove someone, remove them there. **Removing someone ends their SourceLace sign-in at once and deletes their sign-ins to every source.**

A few details about how sign-in works:

- **Microsoft accounts.** For a work or school account, SourceLace uses the person's Microsoft sign-in name (such as ana@corvanta.com), not the free-text email field, because only sign-in names are tied to domains your company has proved it owns. Add people by the address they sign in to Microsoft with. Guest accounts (someone from another company added to your directory) are refused; they sign in with their own company's account. Personal Microsoft accounts that sign in with a phone number cannot sign in.
- **One organization per person.** An email address belongs to one SourceLace organization.
- **How long sign-ins last.** In an MCP client, SourceLace's token lasts 1 hour and renews itself for up to 30 days; after that the person signs in again. In the SourceLace app, a session lasts 7 days.
- **People added by SourceLace's own server settings** (shown as joined by "settings") are managed by whoever runs your SourceLace server and cannot be changed on the Team page.

Single sign-on

Single sign-on is part of the Business and Enterprise plans. Your people can sign in through Okta, Microsoft Entra ID, Google Workspace, OneLogin, Ping, JumpCloud or any OpenID Connect provider, be added automatically on their first sign-in, and have their groups kept in step with your identity provider. You can also require single sign-on for your domains. See [Single sign-on \(SSO\)](#) for every step.

Sources

A **source** is one system your people can connect to, such as your production Salesforce org or a reporting database. Admins add sources on **Manage sources** (or from chat: "Add a SourceLace source salesforce:prod, kind salesforce, label Corvanta Salesforce").

Each source has:

Field	What it is
Kind	Which system it is, such as salesforce, snowflake or database. See Connecting your systems for every kind.
Name	The source's id, written kind:name, such as salesforce:prod or database:sales. Lowercase letters, numbers, dashes and underscores. People and their AI use this id. It cannot be renamed; add a new source instead.
Label	The name people see, such as "Corvanta Salesforce".
Options	Settings for that kind, such as a Salesforce login address or a Snowflake account. Each kind's page lists every option.
Allow changes and Objects that can be changed	Off by default. See Turning on changes .

How many sources of each kind you can add depends on your plan's connector classes and add-ons; connectors you cannot add are marked locked with the reason. See [Connectors](#).

Adding a source does not give anyone access to the data in it. **Each person still connects the source with their own login**, and sees only what that login can see.

Secrets in options. A few kinds take a secret as an option (the client secret for Oracle Fusion and Workday, and the shared secret for a custom connector). It is stored encrypted with your organization's own key, never shown again (the page shows (saved)), and never returned by any API or tool. To keep it while editing the source, leave the field as it is; to change it, type the new one. Type secrets on the Manage sources page, not into a chat with an AI.

Removing a source also removes everyone's sign-in to it and every group's access settings for it. Adding a source with the same name later starts afresh.

Turning on changes

SourceLace is read-only until you say otherwise. To let people create or update records in a source:

1. On **Manage sources**, edit the source and turn on **Allow changes**.
2. Under **Objects that can be changed**, list the objects, separated by commas, such as Opportunity, Case for Salesforce, deals for HubSpot, incident for ServiceNow, customer for NetSuite, invoices for Oracle Fusion, businessTitleChange for Workday or drafts for mail. * allows every object the source supports changes to.
3. Save. From chat: "Allow writes to Account and Opportunity on salesforce:prod".

Even then, every change is previewed first and applied only after the person confirms it, the person's own permissions in the system still apply, and, with access by group on, their groups must give them **read & write** on that object.

Some sources never accept changes through SourceLace, whatever this setting says: they are marked read-only on their pages and on the Manage sources page.

HubSpot, Zendesk and Jira: after you turn changes on or off, or change the objects, people connect that source again, so the system gives their sign-in the matching write permission. Until they do, the system refuses the change and SourceLace tells them to reconnect.

Access by group

Access by group lets you decide which people can use which sources, and which objects inside a source. For example: Sales can read Salesforce and change Opportunities; Finance can read SAP but not Salesforce.

It is **off** until you turn it on. While it is off, everyone in your organization can use every source. Access by group is part of the Business and Enterprise plans.

How it works:

- You create **groups** (such as "Sales") and add **people** to them. Only people already on the Team page can be added.
- Each group gets an **access level** per source: **no access**, **read**, or **read & write**.
- A group can also get a level for **one object** in a source, such as Opportunity. That replaces the group's whole-source level for that object: for example "read all of Salesforce, but no access to Opportunity", or "no access to Salesforce, except read on Account".
- Someone in several groups gets the **most** any of their groups gives, object by object.

- **Once it is on, someone in no group sees no sources at all.** That includes admins: admins can always manage groups, people and sources, but their own data access follows their groups. Put admins in a group if they also use the data.

To set it up, open **Groups**: create the groups, add people, set each group's access per source (and per object where needed), then turn **Access by group** on. From chat, the `admin_groups` tool does the same.

It only narrows access, never widens it. People still sign in to each source with their own account, so they see a record only if both their groups and the system itself allow it. **Read & write** does not by itself let anyone change data: the source must also allow changes to that object, and each change is confirmed. Groups are checked again when a change is applied, so taking access away also stops changes that were previewed but not yet applied.

Which objects a query reads. SourceLace works this out from the query before anything is sent: the object after `FROM` in SOQL, every table after `FROM` and `JOIN` in SQL, the entity set in OData, the object in JSON queries. When a query may reach objects it does not name (related fields such as `Account.Name`, subqueries, joins in some languages, `$expand`), it needs read access to the **whole source**. Someone with only object-level access gets a message asking them to query one object at a time.

Groups from your identity provider. Each group can carry an **external id**: the Okta group name, or the Microsoft Entra group's object id. With single sign-on and group sync on, every sign-in puts the person into the groups their identity provider lists and takes them out of the others that have an external id. Groups without an external id are managed by hand. See [Group sync](#).

Removing someone from the organization takes them out of every group. Chats saved before access was taken away keep what they already show; new questions follow the current groups.

AI settings

These apply to the assistant in the SourceLace app only. MCP clients use their own AI model, which SourceLace does not choose.

Open **Settings** → **AI model**:

- **Included with SourceLace** (the default): the assistant uses Claude, through SourceLace's Anthropic account. Your plan includes a number of AI questions per seat a month, shared by the organization, counted in credits. After them come any prepaid questions, then extra use up to the maximum you set under **Settings** → **AI use** (0 turns it off). At that maximum, chat in the app pauses until the 1st of the next month, or until you raise the maximum or add your own key. See [AI use](#). The **Usage** and **Invoices** pages show your organization's use.
- **Your own Anthropic key**: paste a key from your organization's Anthropic account. Anthropic then bills you directly and questions use no SourceLace credits. The key is encrypted with your organization's own key, never shown again, and never returned by any API. Send a new key only when you want to change it.
- **Your own OpenAI key**: paste a key from your organization's OpenAI account (create one under **API keys** in the OpenAI platform; it starts with `sk-`). OpenAI then bills you directly, and questions use no SourceLace credits. When you save, SourceLace makes one free call to

OpenAI to check that the key works and can use the chosen model, so a mistyped key or a model your account cannot use is caught straight away. The key is stored and protected exactly like an Anthropic key.

You can also choose which model the assistant uses: the Settings page lists the models available for each choice. A larger model gives better answers at a higher cost.

SourceLace keeps one key for your organization. To switch between your own Anthropic key and your own OpenAI key, paste the other provider's key. Switching to **Included with SourceLace** and back keeps your saved key. Saved chats continue with whichever model you choose.

The **Usage** and **Invoices** pages show token counts and an estimated cost for every choice, at the provider's list prices. With your own key, your provider's bill is the real figure.

If the assistant cannot answer, chat says why: for example that the provider refused the key, that your OpenAI account has run out of credit or reached its spending limit, that the provider is busy (try again in a minute), or that your account cannot use the chosen model (choose another in Settings).

What the assistant sends to the AI model is described in [Security and privacy](#).

The audit trail

Every tool call is recorded in your organization's audit trail, whether it succeeded, was refused or failed: from the SourceLace app, from every MCP client, and from admins. Admins see it on the **Audit Log** page (on the Business and Enterprise plans), with filters by time, person, action, source, object and outcome, and a text search.

Each entry holds **metadata only**: when, who, which tool and source, the query text, the object, the record id, the names of changed fields, how many rows came back, the outcome, the error type if any, and how long it took. It never holds row values, field values, passwords or tokens.

Entries are chained by hashes: each entry includes the hash of the one before it, so an entry that was changed, removed or reordered breaks the chain and is detected. If SourceLace cannot write an entry, the call fails rather than returning data it could not record. Before a change is sent to a system, SourceLace records that it is starting.

Sign-ins (Google, Microsoft and single sign-on, including refused ones) and every change to people, sources, groups and single sign-on settings are recorded too.

The same entries are the usage meter: the **Usage** page counts calls and rows per person, action and source, and AI questions and their estimated cost.

How long things are kept

What	Kept for
Query results	In memory only, for at most 30 minutes, then gone. Never written to a database.

What	Kept for
Saved chats in the SourceLace app, with the files made in them and project pins to them	30 days after the chat was last used, by default. Admins set the period under Settings → Chats , from 1 day up to the plan's longest (30 days on Starter, 90 on Team, 1 year on Business, 10 years on Enterprise). Anyone can delete their own chats at any time. Old chats are deleted automatically, checked every hour.
Skills and projects	Until someone deletes them.
Each person's sign-ins to sources	Until they disconnect, are removed from the organization, the source is removed, or the system ends the sign-in.
The audit trail	Kept. It is not deleted automatically today. Your plan's audit history (90 days to 7 years) is the least it is kept for.

See [Security and privacy](#) for what is stored and how it is protected.

Status, usage and invoices

- **Status** (everyone) shows whether the AI assistant is ready and, for each source, whether you are connected and how your calls went in the last 24 hours.
- **Usage** (admins) shows calls, rows and AI use over any period, per person, model, action and source.
- **Usage** also shows this month's AI credits: included, prepaid and extra use.
- **Invoices** (admins) shows your organization's AI use month by month, in credits, with the "Extra AI questions" line. These are estimates until billing is connected.

Getting help

Email support@sourcelace.com. The **Support** page in the app starts an email with your organization and browser filled in. Please don't send passwords, keys or customer data. Security questions: security@sourcelace.com.

Single sign-on (SSO)

Have your people sign in to SourceLace through your own identity provider (Okta, Microsoft Entra ID, Google Workspace, OneLogin, Ping, JumpCloud or any OpenID Connect provider), add them automatically on their first sign-in, require SSO for your domains, and sync their groups.

Single sign-on is part of the Business and Enterprise plans (see [Plans and limits](#)).

Status: in testing. Single sign-on is built to the OpenID Connect standard and covered by automated tests; tell support@sourcelace.com when you set it up so we can help with your first sign-in.

SAML-only identity providers are not supported yet; SAML support is planned. Almost every modern identity provider, including all of those above, offers OpenID Connect, so use that.

What it does

- **Sign-in.** The SourceLace sign-in page, and the page an MCP client opens, offers **Continue with SSO / work email** next to Google and Microsoft. The person types their work email; if its domain belongs to an organization with SSO, they are sent to that organization's identity provider, sign in as they do every day, and come back signed in to SourceLace.
- **New people are added automatically.** The first time someone signs in through SSO, SourceLace adds them to your organization as a **member**. Admins can make them admins on the Team page. The Team page shows them as joined by "Single sign-on".
- **Require SSO (optional).** People at your chosen domains can no longer sign in with Google or Microsoft: they are sent to SSO instead. Sessions they already had that did not come through SSO, in the SourceLace app and in MCP clients, stop working at once, so they sign in again through SSO.
- **Group sync (optional).** At every SSO sign-in, SourceLace reads the person's groups from your identity provider, puts them into the matching SourceLace groups, and takes them out of groups they are no longer in. See [Group sync](#).
- **Audit.** Every sign-in (including refused SSO sign-ins, and Google or Microsoft sign-ins that were sent to SSO) and every change to the SSO settings is recorded in your audit trail: who, when, how, and which groups changed. Never secrets or tokens.

How SourceLace checks a sign-in

SourceLace uses the OpenID Connect **authorization code flow with PKCE**, plus state and nonce. It reads your provider's addresses from `<issuer>/well-known/openid-configuration` and checks the ID token in full: its **signature** against your provider's published keys (only public-key algorithms such as RS256, never "none" or shared-secret ones), the **issuer**, the **audience** (your client id), the **expiry** and the **nonce**. It then takes the person's address from the `email` claim, requires it to be at one of the domains you chose for SSO, and, when the provider sends `email_verified`, requires that to be true.

Your client secret is encrypted with your organization's own key before it is stored. It is never shown again, never logged and never returned by any API: the settings screen only says it is set.

A saved secret is only ever sent to the identity provider it was saved for: changing the issuer URL or client id needs the secret to be entered again.

Before you start

1. On the **Team** page, make sure your email domains (for example corvanta.com) are listed under **Email domains**. SSO can only cover domains your organization already has.
2. Note the **redirect URL** to register in your identity provider. It is shown at the top of **Settings** → **Single sign-on (SSO)** in the SourceLace app. On this server it is:

`https://sourcelace.onrender.com/login/callback`

If the screen lists more than one address, register all of them.

Set up Okta

1. Sign in to the **Okta Admin Console**.
2. Go to **Applications** → **Applications** → **Create App Integration**.
3. Choose **OIDC - OpenID Connect**, then **Web Application**, and click **Next**.
4. Fill in:
 - **App integration name:** SourceLace
 - **Grant type:** leave **Authorization Code** ticked.
 - **Sign-in redirect URIs:** the redirect URL from [Before you start](#). Remove the example sign-out URI, or leave it empty.
 - **Assignments:** choose who may use SourceLace (everyone, or selected groups).
5. Click **Save**.
6. On the app's **General** tab, copy the **Client ID** and the **Client secret** (under *Client Credentials*; client authentication "Client secret"). You may tick **Require PKCE as additional verification**: SourceLace always uses PKCE.
7. **Issuer URL.** Your Okta domain, such as `https://corvanta.okta.com` (shown at the top right of the Admin Console, or under **Security** → **API**). If you use a custom authorization server instead, the issuer is its address, such as `https://corvanta.okta.com/oauth2/default`.
8. **Email.** Nothing to do: Okta sends email and email_verified with the email scope SourceLace asks for.
9. **Groups (only for group sync).**
 - With the Okta domain as issuer: open the app's **Sign On** tab → **OpenID Connect ID Token** → **Edit**. Set **Groups claim type** to **Filter**, the claim name to groups, and the filter to **Matches regex** `.*` (or, to send only some groups, **Starts with** SourceLace-). Click **Save**. SourceLace asks for the groups scope itself when group sync is on.
 - With a custom authorization server: **Security** → **API** → **Authorization Servers** → your server → **Claims** → **Add Claim**: name groups, include in **ID Token** (Always), value type **Groups**, filter **Matches regex** `.*`.
 - Okta sends **group names**. In SourceLace, set each group's **external id** to the Okta group's name exactly as it appears in Okta (case does not matter).

Set up Microsoft Entra ID

1. Sign in to the **Microsoft Entra admin center** (entra.microsoft.com).
2. Go to **Identity → Applications → App registrations → New registration**.
3. Fill in:
 - **Name:** SourceLace SSO
 - **Supported account types:** **Accounts in this organizational directory only (single tenant)**.
 - **Redirect URI:** platform **Web**, and the redirect URL from [Before you start](#).
4. Click **Register**. On the **Overview** page, copy the **Application (client) ID** (the client id) and the **Directory (tenant) ID**.
5. **Client secret.** Go to **Certificates & secrets → Client secrets → New client secret**, choose an expiry, click **Add**, and copy the secret's **Value** (not its Secret ID) right away. Put a reminder in your calendar: when it expires, create a new one and paste it into SourceLace.
6. **Issuer URL.** [https://login.microsoftonline.com/<Directory \(tenant\) ID>/v2.0](https://login.microsoftonline.com/<Directory (tenant) ID>/v2.0). It must contain your directory's id: /common or /organizations will not work, because SourceLace only accepts people from your own directory.
7. **Email claim.** Go to **Token configuration → Add optional claim**, choose token type **ID**, tick **email**, and click **Add** (accept the prompt to turn on the Microsoft Graph email permission). Each person's **Email** field in Entra must hold their address at your domain. Entra does not send `email_verified`; SourceLace relies on the token coming from your own directory and the address being at one of your SSO domains.
8. **API permissions.** The defaults (User.Read, plus openid, email and profile) are enough. If your organization requires it, click **Grant admin consent**.
9. **Groups (only for group sync).**
 - Go to **Token configuration → Add groups claim**. Choose **Groups assigned to the application** (recommended) or **Security groups**. Under **ID**, keep **Group ID**. Click **Add**.
 - If you chose *Groups assigned to the application*, go to **Enterprise applications → SourceLace SSO → Users and groups** and add the groups SourceLace should see (this needs Entra ID P1 or higher).
 - Entra sends **group object ids** (such as 0f3a6c1e-...). In SourceLace, set each group's **external id** to the Entra group's **Object Id** (from **Groups → All groups → the group**).
 - If someone is in more than 200 groups, Entra leaves the groups out of the token ("group overage"). SourceLace then leaves that person's groups unchanged for that sign-in. Assigning only the needed groups to the application avoids this.

Other providers

For Google Workspace, OneLogin, Ping, JumpCloud and others: create an **OpenID Connect web application** in the provider's admin console with the redirect URL from [Before you start](#), and note its **issuer URL**, **client id** and **client secret**. The issuer URL is the address before `/.well-known/openid-configuration` in the provider's discovery URL. For Google Workspace it is `https://accounts.google.com` (create an OAuth client of type *Web application* in Google Cloud Console). Make sure the provider sends the `email` claim; for group sync, a claim listing the person's groups (set its name under **Group claim** in SourceLace if it is not groups).

Turn it on in SourceLace

1. In the SourceLace app, as an admin, open **Settings → Single sign-on (SSO)**.
2. Enter the **Issuer URL**, **Client id** and **Client secret** from your identity provider.
3. Tick the **email domains** that should sign in through SSO.
4. Leave **Require SSO** off for now, and click **Save SSO settings**. SourceLace checks the issuer URL by fetching its discovery document; if that fails, it tells you why.
5. **Test it:** sign out, choose **Continue with SSO / work email** on the sign-in page, and enter your work email. You should land in SourceLace, signed in. Try it from an MCP client too if you use one.
6. **Require SSO (optional):** back in **Settings → Single sign-on**, turn on **Require SSO** and save. SourceLace only lets you do this once you yourself are signed in through SSO, so a setting that does not work cannot lock you out.
7. **Group sync (optional):** see below.

To turn SSO off, click **Turn off SSO**. People then sign in with Google or Microsoft again.

If your identity provider stops working while **Require SSO** is on, your admins cannot sign in either. Contact support@sourcelace.com.

Group sync

1. On the **Groups** page, set each group's **External id**: the Okta group name, or the Entra group's object id.
2. In **Settings → Single sign-on**, turn on **Sync groups at each sign-in** (the group claim is usually groups) and save.

People's groups update the next time they sign in through SSO. Only groups with an external id are synced; groups without one are managed by hand and never touched. To use the synced groups for access, turn on [access by group](#).

When something goes wrong

What you see	What to do
"Enter the issuer URL your identity provider shows, such as <code>https://corvanta.okta.com</code> ."	The issuer URL is empty or not an <code>https://</code> address. Copy it from your identity provider.

What you see	What to do
"The identity provider's discovery document names a different issuer..."	Copy the issuer URL exactly as your provider shows it. For Entra, use the one with your directory's tenant id, not /common or /organizations.
"... does not sign in to SourceLace with single sign-on. Go back and sign in with Google or Microsoft..."	That email domain is not ticked under SSO, or the domain is not on your Team page.
"Your identity provider did not accept the sign-in..."	Check the client id, client secret and redirect URL in both your identity provider and SourceLace. An expired Entra client secret causes this too.
"Your identity provider did not send an email address..."	Add the email claim to the SourceLace app in your identity provider (for Entra, step 7 above).
"Your identity provider has not verified this email address."	The provider sent <code>email_verified: false</code> . Verify the address in your identity provider.
"... is not at a domain your organization signs in with through SSO."	The person's address in the identity provider is at a domain not ticked for SSO in SourceLace.
"... already uses SourceLace with another organization, so it cannot sign in here."	That address belongs to another SourceLace organization. Contact support.
"The sign-in was not meant for SourceLace. Ask your admin to check SSO."	The token's audience or issuer does not match the client id or issuer saved in SourceLace.
"Single sign-on is no longer set up for your organization."	SSO was turned off while the person was signing in. Sign in with Google or Microsoft, or turn SSO back on.
Someone's groups did not change	Group sync only runs at an SSO sign-in, and only for groups with an external id. For Entra, check the person is in fewer than 200 groups, or assign only the needed groups to the app.

Plans and limits

What each SourceLace plan includes, how AI use is counted, and what happens at each limit. Your organization's plan is set by your SourceLace contact; admins see it on the **Manage sources**, **Team**, **Settings** and **Usage** pages.

Features your plan does not include are never hidden: the app shows them with the plan they are available on, such as "available on the Business plan".

The plans

	Starter	Team	Business	Enterprise
Seats (people who can sign in)	1 to 5	5 to 50	25 to 500	75 and up
How people use SourceLace	Their own AI app (Claude, ChatGPT, Cursor)	Their own AI app and the SourceLace app	Same	Same
AI questions included per seat a month, pooled	None	100	150	200
Standard connectors	3 systems	Unlimited	Unlimited	Unlimited
Business connectors	None	2 systems	5 systems	Unlimited
Enterprise connectors	None	None	Add-on only	3 included
Changes to records (preview and confirm)	Yes	Yes	Yes	Yes
Tool calls per person a month (fair use)	5,000	10,000	25,000	50,000
Access by group, single sign-on, audit log viewer	No	No	Yes	Yes
Audit history kept	90 days	1 year	3 years	7 years
Longest chat retention	30 days	90 days	1 year	10 years

Your SourceLace contact can change any of these for your organization, for example for a pilot.

Seats

A seat is one named person on the **Team** page, however they joined (invited, by email domain or through single sign-on). The Team page shows how many seats are used.

- **Starter and Team:** at the plan's maximum, nobody new can be invited or join by email domain until someone is removed or more seats are added. Admins see a notice at 90% of the maximum.
- **Business and Enterprise:** you can go over the maximum; the extra seats are counted and billed at the quarterly true-up.

Connectors

Every connector belongs to one of three classes:

Class	Connectors	Extra system, per month
Standard	Gmail and Google Calendar (google_workspace), Google Drive, Google Sheets, Microsoft 365 mail and calendar, SharePoint and OneDrive, Amazon S3, PostgreSQL, MySQL, SQL Server, MongoDB, Jira, Zendesk, HubSpot, custom connectors	\$100
Business	Salesforce (with Data 360), Marketing Cloud, Dynamics 365, ServiceNow, NetSuite, Gong, Snowflake, BigQuery, Databricks, Oracle Database	\$400
Enterprise	SAP S/4HANA, SAP SuccessFactors, SAP ECC on-premise, Oracle Fusion Cloud, Workday	\$1,500

A **system** is one source: two Salesforce orgs are two Business systems. Your plan includes a number of systems per class, and your contract can add more ("Enterprise: SAP S/4HANA x 1"). On **Manage sources**, connectors you can add are listed as usual; the others are marked locked, with the reason: the plan they are available on, what one more system costs, or "Ask your SourceLace contact". SourceLace checks this when a source is added, however it is added (the app or an AI app), so the limit cannot be passed by accident. Your contact can also switch a connector off for your organization, for example while it is in testing; its sources then stop working until it is switched on again.

AI use: credits and questions

AI use in the SourceLace app is counted in **credits**: one credit is one US cent. Each question costs what the AI model charged for it plus 30%, so a typical question is about 20 credits. The app shows credits as **questions** (20 credits each).

- **Included.** Each plan includes a pool of questions per seat a month, shared by the whole organization: heavy users borrow from light ones. Billed seats are the most of the seats bought, the plan's minimum and the people on the Team page. The pool renews on the 1st of each month.
- **Prepaid questions.** Packs of 5,000 questions (\$1,000) are used after the included questions and before any extra use.
- **Extra use.** After that, questions cost \$0.25 each (\$0.20 on Enterprise), up to a monthly **maximum** your admin sets under **Settings → AI use**. By default the maximum is 25% of the included questions; 0 turns extra use off. A maximum above the included amount needs a card on file (Team) or your SourceLace contact's approval (Business and Enterprise). Extra use is a separate line, "Extra AI questions", billed after the month ends.
- **At the maximum,** chat in the SourceLace app pauses until the 1st of next month, or until an admin raises the maximum (that applies at once). SourceLace keeps working in your own AI apps: they do their own thinking and use no credits.
- **Your own AI key.** With your organization's own Anthropic or OpenAI key (Settings → AI model), your provider bills you directly and questions use no credits.

Admins see a banner in the app at 80% and at 100% of the included questions, and at 80% of the maximum extra use. (Emails are planned.) The **Usage** page shows this month's credits, and

Invoices shows each month's credits, prepaid use and extra questions. Until billing is connected, these are estimates.

Fair use

Each plan has a fair-use number of tool calls per person a month (every lookup, from the app or an AI app). Nothing stops when someone passes it; admins see a notice, and repeated heavy use may be discussed at renewal.

Locked features

- **Starter:** the SourceLace app's chat is available on Team and up. Everything else works through your own AI app, as described in [Getting started](#).
- **Starter and Team:** access by group, single sign-on and the Audit Log page are available on Business and up. Every call is still recorded in your audit trail and counted on the Usage page. If you had groups or single sign-on before, they keep working; you can turn access by group off and remove SSO at any time.

Retention

Your admin chooses how long saved chats are kept, up to your plan's longest setting (see the table). A longer setting is refused with the plan that allows it. If your plan changes to one with a shorter limit, chats are kept for at most the new limit. The audit trail is kept at least as long as your plan's audit history; SourceLace does not delete audit entries today.

Connecting your systems

Every system SourceLace connects to, who sets each one up, what people can read and change, and what to do when a connection fails.

How connecting works

Connecting a system takes up to three steps, done by different people:

1. **Once per system, in that system** (some systems only): your admin for that system registers SourceLace there, for example an OAuth client in ServiceNow or a security integration in Snowflake. Some systems use an app SourceLace provides instead, which your admin may need to approve or install.
2. **Once, in SourceLace:** a SourceLace admin adds the **source** on **Manage sources**, with that system's options (see [Sources](#)).
3. **Each person, once:** they **connect** the source with their own login, from **Data Sources** in the SourceLace app or by asking their AI app to connect it. The system's sign-in page opens; they sign in there and come back. A connect link works once, for 10 minutes.

From then on, every request runs with that person's own sign-in, so the system's own permissions decide what they see and change. SourceLace keeps each person's sign-in encrypted with your organization's key and never shows it to anyone, including admins.

The redirect URL

Systems that use OAuth sign-in ask for the address they send people back to after they sign in. For SourceLace on this server it is:

`https://sourcelace.onrender.com/connect/callback`

Register exactly this address (with https, and no slash at the end) wherever a setup page asks for a redirect URL, callback URL or redirect URI for **connecting a source**. Single sign-on to SourceLace itself uses a different address, given on the [Single sign-on](#) page.

Every system SourceLace connects to

"In testing" means the connector is built against the vendor's documented API and covered by automated tests against a simulated system, but has not yet been used with a customer's live system. Tell support@sourcelace.com before you start with one, so we can help with the first connection.

System	Kind	Setup page	Reads	Changes	Status
Salesforce (Sales and Service Cloud)	salesforce	Salesforce	SOQL, schema, records	Create, update, delete	Available
Salesforce Data 360 (Data Cloud)	data360	Salesforce	SQL, schema, records	None	Available

System	Kind	Setup page	Reads	Changes	Status
Salesforce Marketing Cloud Engagement	sfmtc	Salesforce	Data extensions, journeys, content, send definitions	None	Set up with SourceLace support
SAP S/4HANA	s4hana	SAP	OData	None	SAP's public demo sandbox only
SAP SuccessFactors	successfactors	SAP	OData	None	SAP's public demo sandbox only
SAP ECC (on-premise)	ecc	SAP ECC	Gateway OData, through the SourceLace Agent	Create, update	In testing
HubSpot	hubspot	HubSpot	CRM search, schema, records	Create, update, delete	In testing
Gmail and Google Calendar	google_workspace	Mail and calendar	Mail, events, calendars	Email drafts only	In testing
Outlook mail and calendar (Microsoft 365)	microsoft365	Mail and calendar	Mail, events, calendars	Email drafts only	In testing
Google Drive	google_drive	Files	Search, folders, file text	None	In testing
SharePoint and OneDrive	microsoft_files	Files	Search, libraries, file text	None	In testing
Amazon S3	s3	Files	Keys, file text	None	In testing
Google Sheets	google_sheets	Google Sheets	One sheet's tabs, named by its link	Add rows, update cells	In testing
ServiceNow	servicenow	Service desks	Any table you can read	Create, update	In testing
Zendesk	zendesk	Service desks	Tickets, users, organizations, groups	Ticket create, update	In testing
Jira Cloud	jira	Service desks	Issues, projects	Issue create, update, comment	In testing
Snowflake	snowflake	Data platforms	SQL	None	In testing
Google BigQuery	bigquery	Data platforms	SQL	None	In testing
Databricks	databricks	Data platforms	SQL	None	In testing
Microsoft Dynamics 365 (Dataverse)	dynamics365	Business apps	OData, schema, records	Create, update	In testing
Oracle NetSuite	netsuite	Business apps	SuiteQL, schema, records	Create, update	In testing

System	Kind	Setup page	Reads	Changes	Status
Gong	gong	Business apps	Calls, transcripts, users, stats	None	In testing
PostgreSQL, MySQL, MariaDB, SQL Server, Oracle Database	database	Databases	SQL	None	In testing
MongoDB	mongodb	Databases	Find and read-only pipelines	None	In testing
Oracle Fusion Cloud (ERP, SCM, HCM)	oracle_fusion	Oracle Fusion and Workday	REST resources	Create, update	In testing
Workday	workday	Oracle Fusion and Workday	WQL, REST	Start a business title change	In testing
Your own system, through a connector you build	custom	Build your own connector	Your connector's own query language	Create, update, delete, if your connector allows it	New
Slack	slack	Slack and Teams	Conversations, messages, threads, search, people	Post a message or reply	In testing
Microsoft Teams	microsoft_teams	Slack and Teams	Channels, chats, messages, replies, search, people	Post a message or reply	In testing

Who sets up what

System	Your admin for that system	Your SourceLace admin	Each person
Salesforce, Data 360	Installs SourceLace's Salesforce app (a managed package) once, and decides who may use it	Adds the source with the org's login address	Signs in to Salesforce
Marketing Cloud	Creates a read-only installed package, with SourceLace support	Adds the source with the subdomain	Signs in to Marketing Cloud
HubSpot	May need to approve SourceLace's HubSpot app	Adds the source	Signs in and approves the app
Google (Gmail, Calendar, Drive, Sheets, BigQuery)	May need to allow SourceLace's Google app; for BigQuery, grants IAM roles	Adds the source	Signs in with Google
Microsoft (Outlook, SharePoint and OneDrive, Dynamics 365)	May need to grant admin consent to SourceLace's Microsoft app; for Dynamics, licenses and security roles	Adds the source	Signs in with Microsoft
ServiceNow, Zendesk	Registers an OAuth client in your instance	Adds the source with its Client ID	Signs in to ServiceNow or Zendesk
Jira Cloud	Nothing, unless your Atlassian organization restricts apps	Adds the source	Signs in and approves the site

System	Your admin for that system	Your SourceLace admin	Each person
Snowflake, Databricks	Creates an OAuth integration (Snowflake) or app connection (Databricks)	Adds the source with its client id	Signs in to Snowflake or Databricks
NetSuite	Turns on OAuth 2.0 and REST web services; may need to install SourceLace's integration; gives roles the permissions	Adds the source with the account id	Signs in and picks a role
Gong	Nothing to register	Adds the source	Signs in to Gong; Gong only lets technical admins approve the app
Databases	Your database administrator creates a read-only login per person and lets SourceLace's servers in	Adds the source with host and database	Signs in on SourceLace's form with their own database login
Amazon S3	Creates read-only AWS access, arranged with SourceLace support	Adds the source with buckets and region	Clicks Connect (no sign-in page)
Oracle Fusion, Workday	Registers an OAuth client in your identity domain or Workday tenant	Adds the source with the client's details	Signs in to Oracle or Workday
Your own connector	Your developers build and run the connector	Adds the source with its address and shared secret	Clicks Connect (no sign-in page)
SAP ECC (on-premise)	Your IT team runs the SourceLace Agent; your SAP Basis team trusts its CA (STRUST), maps certificates to users (CERTRULE) and activates the Gateway services	Creates the agent's token, then adds the source with the agent's id	Clicks Connect (no sign-in page)
Slack	May need to approve SourceLace's Slack app (Enterprise Grid: allow it per workspace)	Adds the source	Signs in to Slack
Microsoft Teams	Grants admin consent to SourceLace's Microsoft app (needed to read channel messages)	Adds the source	Signs in with Microsoft

How queries stay read-only

Every query is checked by SourceLace **before** anything is sent to your system: it must be in the language that source accepts, and anything that could change data is refused (for example INSERT, UPDATE, DELETE, MERGE, CREATE, DROP in SQL, more than one statement, or \$out and \$merge in MongoDB). Each setup page lists that source's checks. Where the system offers it, SourceLace also asks the system itself for read-only access: a read-only transaction in PostgreSQL and MySQL, read-only OAuth scopes, a byte cap in BigQuery.

A query returns at most 2,000 rows to SourceLace, of which 50 are shown to the AI at a time. Results are held in memory for 30 minutes so they can be joined, charted or made into a

document, and are never written to a database.

When something goes wrong

These messages can come from any source. Each setup page also lists the messages specific to that system.

What you see	What it means and what to do
"You have not connected <i>source</i> yet. Call <code>connect_source</code> to sign in with your own account."	Connect the source: ask your AI to connect it, or click Connect on Data Sources .
"Your sign-in to <i>source</i> has expired. Call <code>connect_source</code> to sign in again."	The system ended your sign-in (for example its refresh token expired or was revoked). Connect again.
" <i>source</i> did not accept your sign-in (...). Call <code>connect_source</code> to sign in again."	The system refused a fresh token too, often because your account or its permissions changed. Connect again; if it keeps happening, check your access in that system.
"This sign-in link has expired or was already used. Ask for a new one."	Connect links work once, for 10 minutes. Ask for a new link.
"Sign-in was not completed: ..."	The system's sign-in page reported a problem. The rest of the message is the system's own reason, such as a redirect URL that does not match or consent that was refused.
"There is no source called '...'. Call <code>list_sources</code> to see them."	The source id is mistyped, or the source was removed.
"Your groups in SourceLace do not give you access to ..."	Access by group is on and none of your groups gives you this. Ask a SourceLace admin.
"Writes to <i>object</i> are turned off for <i>source</i> . An admin can enable them per source and object."	A SourceLace admin must turn on Allow changes for that source and list the object.
" <i>source</i> does not accept changes through SourceLace."	That kind of source is read-only in SourceLace.
"The <i>kind</i> connector is switched off on this server. Ask whoever runs SourceLace to set ..."	SourceLace's app for that system is not set up on this server yet. Contact support@sourcelace.com .
"The <i>kind</i> connector is not available yet."	That kind of source is not built yet.
" <i>source</i> accepts ..., not ..."	The query was written in the wrong language for that source. The AI usually corrects this itself.
" <i>source</i> could not be reached." or "... did not answer in time. Try again shortly."	The system was down, slow, or blocked SourceLace's connection. Try again; for databases, check the firewall (see Databases).
"Result ... has expired or does not exist. Run the query again."	Results last 30 minutes. Run the query again.
"Change ... has expired, was already applied, or does not exist. Propose it again."	A change preview lasts 10 minutes and applies once. Ask for a new preview.
"SourceLace hit an unexpected problem ... Try again; if it keeps happening, tell your admin or SourceLace support."	Something SourceLace did not expect. It is logged for SourceLace's team (without your data); write to support if it repeats.

Salesforce, Data 360 and Marketing Cloud

How to connect Salesforce (Sales and Service Cloud), Salesforce Data 360 and Salesforce Marketing Cloud Engagement: what your Salesforce admin installs, the options, what people can read and change, and what error messages mean.

Salesforce

Each person signs in with **their own Salesforce login** (OAuth with PKCE). Salesforce then applies their profile, permission sets, field-level security and sharing rules to everything SourceLace reads or changes.

What your Salesforce admin does (once per org)

SourceLace signs people in through its own Salesforce **External Client App**, delivered as a **managed package**. Salesforce only lets people use an app that is installed in their org, so your admin installs it once. Ask support@sourcelace.com for the install link. Trying it in a sandbox first is a good habit.

1. **Install.** Open the install link while logged in to the org. For a sandbox, change `login.salesforce.com` in the link to `test.salesforce.com`. Choose **Install for Admins Only** (access is given in step 3), tick that you understand the package is not yet authorized for distribution through AppExchange, and click **Install**.
2. **Open the app's policies.** **Setup** → **External Client App Manager** → **SourceLace** → **Policies** → **Edit**.
3. **Decide who may use it.** Under **OAuth Policies**, set **Permitted Users** to **Admin approved users are pre-authorized**, save, and give access through a permission set: **Setup** → **Permission Sets** → **New** (for example "SourceLace users"), then under **Assigned Connected Apps / External Client Apps** add SourceLace, and assign the permission set to the people who will use it. Or leave **All users may self-authorize**, and each person approves the app the first time they connect.
4. **Optional:** set IP relaxation, session timeout and refresh token policy to your security team's standard. SourceLace works with the defaults.

Exact labels can differ a little between Salesforce releases; Salesforce's help page "External Client App OAuth policies" has the current names.

The app asks Salesforce for the scopes **Manage user data via APIs** (`api`) and **Perform requests at any time** (`refresh_token`, `offline_access`), and for Data 360, also `cdp_query_api` and `cdp_profile_api`. It requires PKCE. Its callback URL is SourceLace's redirect URL (<https://sourcelace.onrender.com/connect/callback>), which travels with the package.

Add the source (SourceLace admin)

On **Manage sources** → **Add a source**, choose **Salesforce**, or ask in chat: "Add a SourceLace source `salesforce:prod`, kind `salesforce`, label `Corvanta Salesforce`, login URL `https://corvanta.my.salesforce.com`".

Option	Type	Default	Example	What it is
login_url	Address	https://login.salesforce.com	https://corvanta.my.salesforce.com	Where people sign in. The org's My Domain (https://<name>.my.salesforce.com) works for everyone and is the safest choice. https://login.salesforce.com works for most production orgs, and https://test.salesforce.com (or the sandbox's My Domain, https://<name>--<sandbox>.sandbox.my.salesforce.com) for sandboxes.

Only Salesforce's own addresses are accepted. A Lightning address copied from the browser (https://<name>.lightning.force.com) is turned into the matching My Domain automatically. After someone signs in, SourceLace sends every request to the org address Salesforce gives back, so the login address only matters for signing in.

What people can do

- **Find objects and fields:** search_schema and describe_object list standard and custom objects and fields, picklist values and relationships, limited to what the person can see.
- **Query:** one SOQL SELECT, such as SELECT Name, Amount FROM Opportunity WHERE IsClosed = false ORDER BY Amount DESC LIMIT 20. Anything that is not a single read-only SOQL query is refused before Salesforce is called.
- **Read a record:** get_record with an 18- or 15-character record id, plus up to 3 related lists.
- **Change records** (create, update, delete), only on objects your SourceLace admin made writable (such as Account, Opportunity, Case). The preview checks the person's object and field permissions and, for existing records, their sharing access. An update is refused if the record changed in Salesforce after the preview. The result includes the previous values, so a change can be undone; deleted records go to Salesforce's Recycle Bin.

When something goes wrong

What you see	What to do
OAUTH_APP_BLOCKED, or "app is not installed"	The package is not installed in that org, or the person lacks access. See steps 1 and 3 above.
OAUTH_APPROVAL_ERROR_GENERIC, or "user is not admin approved"	Assign the person the permission set from step 3.
redirect_uri_mismatch	Contact SourceLace support: the redirect URL travels with the package.
"Salesforce sign-in did not work at ...: it answered with a redirect (or a web page) instead of a token. Check the login URL ..."	The login_url is mistyped or outdated. Use the org's current My Domain, or https://login.salesforce.com.
"The Salesforce login URL for ... must be login.salesforce.com, test.salesforce.com, or the org's My Domain"	Only Salesforce addresses are accepted in login_url.

What you see	What to do
"The Salesforce login URL for ... must be a plain https:// address."	Remove any path, port or ?... from login_url.
"Salesforce sign-in failed: ..."	Salesforce's own reason follows. Often the person's profile cannot use the app (see step 3), or their account is locked.
"Salesforce answered with a web page instead of data ... The org may be under maintenance, or its address changed"	Wait for maintenance to end, or connect again on Data Sources .
"Salesforce moved this org to a new address." or "Your Salesforce session has expired."	SourceLace refreshes the sign-in itself; if it then asks, connect again.
"Salesforce: MALFORMED_QUERY: ..." or "Salesforce: INVALID_FIELD: ..."	Salesforce's own message about the query. The AI usually fixes the query itself.
"Your Salesforce user cannot update Opportunity records." (or create, delete)	The person's Salesforce profile does not allow it. Change their permissions in Salesforce if they should.
"... cannot be set on update (read-only or not permitted for you)."	That field is read-only for the person in Salesforce.
"This ... record changed in Salesforce after the preview. Propose the change again."	Someone else changed the record. Ask for a new preview.

Data 360

Data 360 (formerly Data Cloud) lives inside your Salesforce org, so it uses the **same app and the same sign-in** as Salesforce: SourceLace signs the person in to Salesforce, then exchanges that sign-in for a Data 360 token for the same person, so Data 360 still decides what each person can see. Read-only.

What your Salesforce admin does: install the SourceLace package as above, and give each person a **Data 360 permission set**, such as **Data Cloud User** or **Data Cloud Admin** (names vary by release). Without one, signing in to Salesforce works but Salesforce refuses the Data 360 token.

Add the source: kind **Data 360** (data360), with the same login_url option as Salesforce.

What people can do:

- search_schema and describe_object list data model objects (names ending in __d1m), data lake objects (__d1l) and calculated insights (__cio), with their fields.
- query takes one SQL SELECT (or WITH ... SELECT), such as SELECT ssot__FirstName__c FROM ssot__Individual__d1m LIMIT 10. Anything that could change data and anything with more than one statement is refused before it reaches Data 360.
- get_record looks a row up by the object's primary key, and can add rows from related objects.

What you see	What to do
"Data 360 sign-in failed... Check that Data 360 is on in this org, that you have a Data 360 permission set, and that SourceLace's app has the cdp_query_api and cdp_profile_api scopes."	Give the person a Data 360 permission set. If it still fails, contact support. If they connected before Data 360 was set up, connect again.
"Data 360 has no object called ... that you can read. Try search_schema."	The object name is wrong, or the person cannot see it.

What you see	What to do
"Data 360 is taking too long to run this query. Try a narrower query."	Add filters or a LIMIT.

Marketing Cloud

Marketing Cloud Engagement is a separate account from your Salesforce org, with its own logins, so it uses its own app. Each person signs in with **their own Marketing Cloud login**, and Marketing Cloud applies their roles and business unit access. Everything is **read-only**.

Setup is done together with SourceLace support, because Marketing Cloud's app (an *installed package*) is registered in your Marketing Cloud account and its Client Id and Client Secret are added to SourceLace's server. Your Marketing Cloud admin:

1. Goes to **Setup → Platform Tools → Apps → Installed Packages** and clicks **New**. Name: "SourceLace".
2. Clicks **Add Component**, chooses **API Integration**, then **Web App**.
3. Sets the redirect URI to `https://sourcelace.onrender.com/connect/callback`.
4. Gives it read access only: **Data Extensions: Read, Journeys: Read, Saved Content: Read** and **Email: Read** (plus **SMS: Read** if you use SMS). Leaves every Write, Send and Delete box unticked.
5. Saves, and sends the **Client Id** and **Client Secret** to SourceLace support over a secure channel we agree with you. Notes the **Authentication Base URI**, such as `https://mc1a2b3c4d5e6f.auth.marketingcloudapis.com/`.

Add the source: kind **Marketing Cloud** (sfmc).

Option	Type	Default	Example	What it is
subdomain	Text	(required)	mc1a2b3c4d5e6f	The part of the Authentication Base URI before <code>.auth.marketingcloudapis.com</code> .
account_id	Number	(none)	123456789	Optional: the business unit's MID, to sign in to that business unit.

What people can do:

- `search_schema` finds data extensions, each named by its **external key**. `describe_object` lists its fields and marks the primary key.
- `query` (language `sfmc_rest`) takes a GET path, optionally with options after `?`. Only these are allowed: data extension rows (`/data/v1/customobjectdata/key/<key>/rowset`, such as `.../key/Customers/rowset?$pageSize=50&$filter=Tier eq 'Gold'`), the data extensions themselves (`/data/v1/customobjects`), journeys (`/interaction/v1/interactions`), Content Builder assets (`/asset/v1/content/assets`), and email and SMS send definitions (`/messaging/v1/email/definitions`, `/messaging/v1/sms/definitions`; read only, nothing is sent).
- `get_record` looks one data extension row up by its primary key value.
- Data extension keys with spaces, or punctuation other than `-`, `_` and `.`, are not supported yet.

What you see	What to do
"The sfmc connector is switched off on this server..."	The installed package has not been added to SourceLace yet. Contact support.
"Set the Marketing Cloud subdomain for ... (the part before .auth.marketingcloudapis.com)."	Fill in the subdomain option.
"The Marketing Cloud account_id for ... must be a number (the MID)."	Use the business unit's numeric MID.
"That Marketing Cloud endpoint is not on the read allowlist."	Only the read paths listed above are allowed.
"Marketing Cloud has no data extension with key ... that you can read."	Check the external key with search_schema.
"Marketing Cloud sign-in failed: ..."	Marketing Cloud's own reason follows. Check the person's Marketing Cloud user has access to the business unit.

SAP S/4HANA, SuccessFactors and ECC

What SourceLace can do with SAP today: SAP's public demo sandboxes for S/4HANA Cloud and SuccessFactors, read-only. Per-person sign-in to your own SAP systems is not built yet.

What is available today

SourceLace's SAP connectors read **SAP's public sandbox** on the SAP Business Accelerator Hub: real S/4HANA Cloud and SuccessFactors APIs with **shared demo data, not your company's**. The sandbox signs in with one API key held by SourceLace, not with each person's own SAP login, so it shows SAP questions working end to end but does not test per-person permissions. Everything is read-only.

Per-person sign-in to your own S/4HANA or SuccessFactors system is not built yet. If you want it, tell support@sourcelace.com which systems and how your people sign in to them today.

SAP S/4HANA

Add the source: kind **SAP S/4HANA** (s4hana). There is no sign-in page: connecting goes straight back to SourceLace.

Option	Type	Default	Example	What it is
auth	Text	(required)	sandbox	Must be sandbox: SAP's public demo sandbox.
services	List, separated by commas	(none)	API_CUSTOMER_MATERIAL_SRV	More OData services to offer in schema search, besides the built-in list.

What people can do:

- Objects are named SERVICE/EntitySet, such as API_BUSINESS_PARTNER/A_Customer or API_SALES_ORDER_SRV/A_SalesOrder. search_schema searches a built-in list of common APIs (business partners, customers, suppliers, sales orders, billing documents, deliveries, purchase orders, products, stock, receivables) plus the services option.
- describe_object reads the service's \$metadata: fields, labels, keys and navigation properties.
- query (language odata) takes SERVICE/EntitySet plus OData options: \$filter, \$select, \$expand, \$orderby, \$top, \$skip and \$inlinecount. For example
API_SALES_ORDER_SRV/A_SalesOrder?\$filter=SoldToParty eq '17100001'&\$select=SalesOrder,TotalNetAmount&\$top=20.
- get_record takes a key such as 17100001, or a composite key such as SalesOrder='1',SalesOrderItem='10', and navigation properties as related lists (up to 3).

SAP SuccessFactors

Add the source: kind **SAP SuccessFactors** (successfactors). As with S/4HANA, there is no sign-in page.

Option	Type	Default	Example	What it is
auth	Text	(required)	sandbox	Must be sandbox.
entities	List, separated by commas	(none)	cust_Vehicle, Background_Education	More entity sets to offer in schema search, including custom cust_ objects.

What people can do:

- Objects are entity sets such as User, PerPerson, EmpJob, EmpEmployment, F0Department OR Position.
- describe_object reads that entity's \$metadata.
- query (language odata) takes the entity set plus \$filter, \$select, \$expand, \$orderby, \$top, \$skip, \$inlinecount, and the effective-dating options asOfDate, fromDate and toDate (dates such as 2024-01-31). For example EmpJob?\$filter=department eq 'SALES'&\$select=userId,jobTitle,managerId&\$top=20.
- get_record takes a key such as admin, or an effective-dated key such as seqNumber=1L,startDate=datetime'2015-01-01T00:00:00',userId='admin'.

SAP ECC

SAP ECC systems in your own data centre (kind ecc) are read through your SAP Gateway OData services and a small SourceLace Agent inside your network, as each person's own SAP user, with no SAP passwords stored. It is **in testing**: see [SAP ECC \(on-premise\)](#).

When something goes wrong

What you see	What to do
"...: per-user SAP sign-in is not built yet. For now, this source needs "options": {"auth": "sandbox"} to use SAP's public sandbox."	Set the auth option to sandbox. Your own SAP system cannot be connected yet.
"The SAP sandbox needs an API key..." (or "The SuccessFactors sandbox ...")	SourceLace's sandbox key is not set up on this server. Contact support.
"'...' is not an allowed OData option. Use ..."	Only the options listed above are accepted.
"SAP: ... has no entity set ..." or "SuccessFactors has no entity set ..."	Check the name with search_schema, or add the service or entity with the source's options.
"Ask for at most 3 related lists at a time."	Ask for fewer related lists in one get_record.
"'...' in the entities option of ... is not an entity set name."	Fix the entities option: names only, separated by commas.

HubSpot

How to connect HubSpot's CRM: contacts, companies, deals, tickets, owners and custom objects, with optional confirmed changes. Each person signs in with their own HubSpot login.

Status: in testing. Built against HubSpot's documented API and covered by automated tests against a simulated HubSpot; not yet used with a customer's live account.

How sign-in works

SourceLace has its own HubSpot app. Each person signs in to HubSpot with **their own login**, chooses the HubSpot account, and approves the app. HubSpot then shows SourceLace only what that person could see in HubSpot itself.

What your HubSpot admin may need to do: to install an app into a HubSpot account, HubSpot may require the person to be a **Super Admin** or to have the **App Marketplace access** permission. If people cannot approve the app, a HubSpot admin either approves it first or gives them that permission. HubSpot may also warn that the app is not verified.

Permissions SourceLace asks for:

When	Scopes
Always (reading)	oauth, crm.objects.contacts.read, crm.objects.companies.read, crm.objects.deals.read, crm.objects.tickets.read, crm.objects.owners.read, crm.schemas.contacts.read, crm.schemas.companies.read, crm.schemas.deals.read, crm.schemas.tickets.read
Optional (custom objects, Enterprise plans only; HubSpot leaves them out for accounts without custom objects)	crm.objects.custom.read, crm.schemas.custom.read
Only for objects your SourceLace admin made writable	crm.objects.contacts.write, crm.objects.companies.write, crm.objects.deals.write, crm.objects.tickets.write, crm.objects.custom.write

With changes off, nobody is asked for write access.

Add the source (SourceLace admin)

On **Manage sources** → **Add a source**, choose **HubSpot** (hubspot).

Option	Type	Default	Example	What it is
hub_id	Number	(none)	12345678	Optional but recommended: the HubSpot account ID (in the account menu at the top right, or the address bar, as in <code>app.hubspot.com/contacts/12345678/...</code>). Many people belong to more than one HubSpot account; with <code>hub_id</code> set, SourceLace refuses a sign-in to any other account.

Changes are off unless you turn them on. HubSpot's object names are lowercase plurals: contacts, companies, deals, tickets, or a custom object's name as search_schema shows it (such as p12345678_cars). Owners cannot be changed. **After turning changes on or changing the objects, people connect HubSpot again** so their sign-in gets the new write permission.

What people can do

- search_schema and describe_object list objects and their properties.
- query takes the hubspot_search language: one JSON object in the shape of HubSpot's own search API, plus the object to search:

```
{"object": "deals",  
  "properties": ["dealname", "amount", "dealstage"],  
  "filterGroups": [{"filters": [{"propertyName": "amount", "operator": "GT", "value": "1000"}]}],  
  "sorts": [{"propertyName": "amount", "direction": "DESCENDING"}],  
  "limit": 100}
```

Up to 5 filter groups (a record matches if **any** group matches), each with up to 6 filters that must **all** match (18 filters in all). Operators: EQ, NEQ, LT, LTE, GT, GTE, BETWEEN (with value and highValue), IN and NOT_IN (with values), HAS_PROPERTY, NOT_HAS_PROPERTY, CONTAINS_TOKEN, NOT_CONTAINS_TOKEN. query searches text across the main fields; sorts takes one sort. {"object": "owners"} lists the users who can own records. Every key, operator and property name is checked before anything is sent.

- get_record reads one record by its id (digits only), with associated records as related lists.
- **Changes** (create, update, delete one record at a time), on the objects your admin made writable, after a preview and the person's confirmation. Deleting moves the record to HubSpot's recycling bin, where a HubSpot admin can restore it for 90 days.

Limits: SourceLace fetches 200 rows at a time up to its usual 2,000-row limit; HubSpot itself stops a search at 10,000 rows. HubSpot allows a few searches per second per account: when it asks SourceLace to slow down, SourceLace waits and tries again twice before giving up with a clear message. HubSpot sign-ins last 30 minutes and are refreshed automatically.

When something goes wrong

What you see	What to do
"You signed in to HubSpot account ..., but ... is account Connect again and choose that account."	The person picked the wrong HubSpot account on the sign-in page. Connect again and choose the account in hub_id.
"The HubSpot hub_id for ... must be the account's number, such as 12345678."	Fix the hub_id option: digits only.
"HubSpot: your sign-in does not include the permission this needs. If an admin just turned on writes, connect the source again..."	Connect HubSpot again. If it persists, the person's HubSpot role lacks the permission.
"HubSpot is limiting how fast SourceLace can call it. Wait a few seconds and try again."	Wait and ask again.
"HubSpot: that record does not exist, or you cannot see it."	Check the id; the person may lack access in HubSpot.
"... has no property ... Use describe_object to see the property names."	The query used a property name that does not exist on that object.

What you see	What to do
"HubSpot owners are users and cannot be changed through SourceLace."	Owners are read-only.
"HubSpot sign-in failed: ..."	HubSpot's own reason follows. Often the person cannot install apps in that account (see above).
"The hubspot connector is switched off on this server..."	SourceLace's HubSpot app is not set up on this server yet. Contact support.

Mail and calendar

How to connect Gmail and Google Calendar, or Outlook mail and calendar (Microsoft 365). SourceLace reads mail and calendars and can make email drafts. It never sends email.

Status: in testing. Built against Google's and Microsoft's documented APIs and covered by automated tests against simulated services; not yet used with a customer's live mailboxes.

What SourceLace can and cannot do

- **Read mail:** search messages by sender, subject, words, dates, folder, unread or attachment, and read one message's plain-text body and attachment names. Message lists show sender, recipients, subject, date and a snippet, not the body.
- **Read calendars:** events in a date range (up to a year at a time), with attendees, and the list of calendars.
- **Make drafts:** with changes turned on for drafts, SourceLace can create an email draft after the person confirms the preview. The draft waits in the person's **Drafts** folder until they review and send it themselves. SourceLace marks its drafts, and only ever updates or deletes drafts it made, never a draft the person wrote and never real mail.
- **Never:** send email, change calendar events, or delete messages.

Each person signs in with their own Google or Microsoft account, and sees only their own mailbox and the calendars their account can see.

Gmail and Google Calendar

Kind: **Google Workspace (Gmail and Calendar)** (`google_workspace`). No options.

How sign-in works. SourceLace has its own Google app for mail, calendar and files (separate from "Sign in with Google" to SourceLace). Each person signs in with their own Google account and approves these permissions:

Scope	Why
<code>https://www.googleapis.com/auth/gmail.readonly</code>	Read mail.
<code>https://www.googleapis.com/auth/gmail.compose</code>	Make drafts. Google has no drafts-only permission; this one would also allow sending, but SourceLace never sends, and refuses every call that is not on its short list of reads and draft calls.
<code>https://www.googleapis.com/auth/calendar.readonly</code>	Read calendars.

People must tick **every box** on Google's consent screen.

What your Google Workspace admin may need to do. If your organization restricts which third-party apps may use Google data, allow SourceLace's app in the **Google Admin console → Security → Access and data control → API controls → App access control** (look for SourceLace among the connected apps, and mark it trusted or limited as your policy requires). Google reviews apps that ask for Gmail access; if Google shows an "unverified app" warning or refuses the sign-in, contact support@sourcelace.com.

Outlook (Microsoft 365)

Kind: **Microsoft 365 (Outlook mail and calendar)** (microsoft365). No options.

How sign-in works. SourceLace uses its Microsoft app (the same one as "Sign in with Microsoft"). Each person signs in with their own work or school account and approves these **delegated** Microsoft Graph permissions, so SourceLace acts as that person with only their access:

Permission	Why
offline_access	Stay connected without signing in every hour.
User.Read	Know who signed in.
Mail.Read	Read mail.
Mail.ReadWrite	Make drafts.
Calendars.Read	Read calendars.

SourceLace does **not** ask for Mail.Send, so Microsoft itself would refuse to send mail for SourceLace.

What your Microsoft admin may need to do. People can usually approve these permissions themselves. If your organization requires an admin to approve apps, Microsoft shows **Need admin approval**; a Microsoft admin then grants consent for SourceLace in the **Microsoft Entra admin center → Enterprise applications** (or from the approval request).

Add the source (SourceLace admin)

Add a **Google Workspace** or **Microsoft 365** source on **Manage sources**. To allow drafts, turn on **Allow changes** and enter drafts under **Objects that can be changed**: it is the only object these sources accept changes to.

What people can do

query takes one small JSON object (language json), checked strictly before anything is sent. describe_object ON messages, events or calendars lists every key.

- **Mail:** {"object": "messages", "from": "ann@corvanta.com", "subject": "renewal", "text": "pricing", "after": "2026-09-01", "before": "2026-10-01", "folder": "inbox", "unread": true, "has_attachment": false}. Every key but object is optional; folder is inbox, sent, drafts or any. Newest first, at most 100 messages.
- **Calendar:** {"object": "events", "start": "2026-10-01", "end": "2026-10-15", "text": "QBR", "attendee": "corvanta.com", "calendar": "primary"}. start defaults to now and end to 30 days later, at most 366 days apart; at most 250 events, soonest first, with repeating events listed one by one. Times are in UTC.
- **Calendars:** {"object": "calendars"}.
- get_record with messages and an id returns the plain-text body (up to 20,000 characters) and attachment names; with events, the full description (up to 8,000 characters).

- **Drafts:** propose_change with object drafts and fields to, subject, body (plain text), and optionally cc and bcc (at most 50 recipients); then the person confirms.

When something goes wrong

What you see	What to do
"Google did not grant Connect again and tick every box on Google's consent screen."	Connect again and tick every box.
"Google: this sign-in does not allow that. Connect the source again and tick every box."	A permission was not granted. Connect again.
"Microsoft sign-in failed: ..."	Microsoft's own reason follows. If it mentions admin approval or consent, ask your Microsoft admin (see above).
"Microsoft 365: not found, or you cannot see it." or "Google: not found, or you cannot see it."	The message, event or calendar does not exist, or the person cannot see it.
"SourceLace only creates email drafts here: use object drafts. It never sends email or changes calendars."	Only drafts can be created.
"SourceLace only changes or deletes drafts that it created."	Drafts the person wrote themselves cannot be changed through SourceLace.
"Writes to drafts are turned off for ..."	A SourceLace admin turns on Allow changes with the object drafts.
"The google_workspace connector is switched off on this server..." (or microsoft365)	SourceLace's Google or Microsoft app is not set up on this server yet. Contact support.

Files: Google Drive, SharePoint and OneDrive, Amazon S3

How to connect file stores so people can find files where they already are and read the text inside them. Everything is read-only, and files are read only when someone asks, never copied or indexed.

Status: in testing. Built against each service's documented API and covered by automated tests against simulated services; not yet used with a customer's live files.

How it works

- **Read on demand, never indexed.** SourceLace does not copy, crawl or index anyone's files. When someone asks to read a file, SourceLace downloads that one file, pulls the text out in memory, and returns it in chunks. The chunks are held in memory like any other result (up to 30 minutes) and then forgotten.
- **Finding files uses each store's own search:** Google Drive's search, Microsoft Search for SharePoint and OneDrive, and key names for S3 (S3 cannot search inside files).
- **What it can read:** PDF, Word (.docx), Excel (.xlsx), PowerPoint (.pptx), CSV, plain text (and Markdown, JSON, HTML), and Google Docs, Sheets and Slides. Older .doc, .xls and .ppt files, images, and scanned PDFs without a text layer are not read; SourceLace says so instead of guessing. Password-protected PDFs cannot be read.
- **Limits:** files up to 25 MB; up to 400,000 characters of text per file (longer files are marked text_truncated); PDFs up to 1,000 pages; Office files that unpack to more than 250 MB are refused.

Source	Kind	Who decides what someone can read
Google Drive	google_drive	Each person signs in with their own Google account. Drive's own sharing applies.
SharePoint and OneDrive	microsoft_files	Each person signs in with their own work or school account. SharePoint and OneDrive permissions apply.
Amazon S3	s3	The source, not the person. See Amazon S3 .

How people use it

Every file source takes one query language, files: a small JSON object with an action.

Action	Example	What it does
search	<pre>{"action": "search", "text": "corvanta renewal"}</pre>	Finds files with the store's own search. "match": "name" matches names only; the default "content" also matches text inside files where the store supports it. Optional: "folder", "modified_after": "2026-09-01", "limit" (up to 200).

Action	Example	What it does
list	<code>{"action": "list", "folder": "root"}</code>	The files and folders in one folder. root is My Drive, your OneDrive, or the top of the bucket.
drives	<code>{"action": "drives"}</code>	Where files live: shared drives, SharePoint document libraries (add "text": "sales" to find sites by name), or the source's buckets.
read	<code>{"action": "read", "file": "<id>"}</code>	The file's text, in chunks of 4,000 characters ("chunk_chars" up to 20,000), 10 chunks at a time ("chunks" up to 50). "from_chunk": 10 reads further.

get_record with object files and a file id returns the file's details and the first 20,000 characters of its text.

Google Drive

Kind: **Google Drive** (google_drive). No options.

Each person signs in with their own Google account through SourceLace's Google app (the same app as Gmail and Calendar), and is asked only for <https://www.googleapis.com/auth/drive.readonly> ("See and download all your Google Drive files"). Connecting Drive never gives SourceLace someone's mail, and connecting mail never gives it their files.

Your Google Workspace admin may need to allow SourceLace's app under **Google Admin console** → **Security** → **Access and data control** → **API controls** → **App access control**, if your organization restricts third-party apps. Google reviews apps that ask for Drive access; if Google shows an "unverified app" warning or refuses, contact support@sourcelace.com.

SharePoint and OneDrive

Kind: **SharePoint and OneDrive** (microsoft_files). No options.

One source covers both: each person's OneDrive (`{"action": "list"}`) and every SharePoint document library they can open (`{"action": "drives", "text": "sales"}`). Search covers both too. File ids look like b!abc...~01XYZ...; use them exactly as returned.

Each person signs in with their own work or school account through SourceLace's Microsoft app, and approves these **delegated** Microsoft Graph permissions (SourceLace acts as the person, with only their access): offline_access, User.Read, Files.Read.All, Sites.Read.All. If your organization requires an admin to approve apps, a Microsoft admin grants consent in the **Microsoft Entra admin center** → **Enterprise applications**.

Amazon S3

Kind: **Amazon S3** (s3).

S3 has no per-person sign-in, so **S3 access is per source, not per person**. Each S3 source reads with one set of AWS credentials, and everyone who can use that source reads with that same access. The IAM policy behind those credentials decides which buckets and files the source

can read. To narrow **who in your organization** can use an S3 source, use [access by group](#). Until you do, everyone in your organization can use every S3 source you add, so only add buckets that everyone may read.

SourceLace makes only two kinds of S3 call, both reads: **ListObjectsV2** (list and search) and **GetObject** (read). With a role, it also calls STS **AssumeRole** for short-lived credentials. It signs requests itself (AWS Signature Version 4) and uses no AWS SDK.

1. Create read-only access (your AWS admin)

AWS credentials are kept in SourceLace's server settings, never in a source's options (which admins can see), so **S3 is set up together with SourceLace support**. Create a read-only IAM policy, such as (change corvanta-docs and the reports/ folder to yours; drop the Condition to allow the whole bucket):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::corvanta-docs",
      "Condition": {
        "StringLike": {
          "s3:prefix": ["reports/*"]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::corvanta-docs/report
s/*"
    }
  ]
}
```

Then, with support, choose one of:

- **A role SourceLace assumes:** put the policy on an IAM role in your account whose trust policy lets the AWS identity that support names assume it, with an **external id**. You give the source the role's ARN and external id, so no long-lived keys of yours leave your account.
- **Access keys:** an IAM user with the policy and an access key ("Application running outside AWS"), sent to support over a secure channel we agree with you. Support stores them under a name (such as default) in SourceLace's server settings.

2. Add the source (SourceLace admin)

Option	Type	Default	Example	What it is
buckets	List, separated by commas	(required)	corvanta-docs	The buckets this source reads.
region	Text	(required)	us-east-1	The buckets' AWS region.
prefix	Text	(none)	reports/	Only read under this folder. SourceLace enforces it too, before any call.
keys	Text	default	finance	The name of the credentials in SourceLace's server settings, as support tells you.
role_arn	Text	(none)	arn:aws:iam::123456789012:role/sourcelace-read	An IAM role to assume.
external_id	Text	(none)	corvanta-7f3a	The external id the role's trust policy expects.

People still click **Connect** once (there is no sign-in page; it only records that they connected).
`list_sources` shows the source's access as the key name or the role, not a person.

S3 search matches file names only, and looks through up to 10,000 keys per search: use prefix or folder to narrow big buckets.

When something goes wrong

What you see	What to do
"Google did not grant read access to Drive. Connect again and tick the box on Google's consent screen."	Connect again and tick the Drive box.
"'...' is ... MB; SourceLace reads files up to 25 MB."	The file is too large to read.
"The PDF is password-protected, so SourceLace cannot read it."	Remove the password, or read it another way.
"SourceLace could not read the text in '...': the file looks damaged or is in an unexpected format"	The file could not be read; open it in its own app to check.
"'...' is a folder. List it with ..."	Use <code>list</code> on a folder, <code>read</code> on a file.
"'...' is not a SharePoint or OneDrive id. Use an id from a search, list or drives result."	Use file ids exactly as SourceLace returned them.
"... needs a valid "buckets" (bucket names, separated by commas) in its options." (or "region", "role_arn", ...)	Fix that option on Manage sources .
"... uses the S3 keys named '...', which are not in SOURCELACE_S3_KEYS on the server."	The keys option names credentials SourceLace does not have. Check the name with support.
"AWS refused to let SourceLace assume the source's role (HTTP ...). Check role_arn, external_id and the role's trust policy."	Fix the role's trust policy, <code>role_arn</code> or <code>external_id</code> .
"Amazon S3 refused access (AccessDenied). The source's keys or role cannot read this; an admin can change its IAM policy."	Widen the IAM policy if the source should read it.
"This source only reads under ...".	The file is outside the source's prefix.
"'...' is not in this source's buckets (...)."	The folder is in a bucket the source does not list.
"Microsoft sign-in failed: ..."	Microsoft's own reason follows. If it mentions admin approval or consent, ask your Microsoft admin.

Service desks: ServiceNow, Zendesk and Jira

How to connect ServiceNow, Zendesk and Jira Cloud. People read with their own logins and, if an admin turns it on, make single-record changes after confirming them.

Status: in testing. All three are built against each vendor's published API and covered by automated tests against simulated systems; none has been used with a customer's live instance yet. Menu names move between releases: if a step doesn't match what you see, look for the nearest equivalent.

System	Reads	Changes (off unless an admin turns them on)
ServiceNow	Any table the person can read: incidents, tasks, problems, changes, requests, knowledge, users and so on	Create or update one record in a table the admin lists, such as incident. Comments and work notes are fields (comments, work_notes). No deletes.
Zendesk	Tickets (with comments), users, organizations, groups	Create a ticket, or update one: status, priority, assignee, tags, custom fields, a public reply (comment) or a private note (internal_note). No deletes.
Jira Cloud	Issues (with comments and change history) and projects	Create an issue, update its fields, or add a comment. No deletes, and no status changes yet (Jira moves status through workflow transitions).

Each person signs in with **their own login**, so ServiceNow, Zendesk and Jira only show and allow what that person could see and do there anyway. SourceLace never uses a shared admin account.

The redirect URL for all three is:

`https://sourcelace.onrender.com/connect/callback`

ServiceNow

ServiceNow lives at your own address (such as `corvanta.service-now.com`), so **your ServiceNow admin registers SourceLace once in your instance**, which gives a Client ID. That ID is not a secret. SourceLace registers as a *public client* that proves each sign-in with PKCE, so no client secret is created, sent or stored.

Register SourceLace (ServiceNow admin)

1. Open **All** (top left), type Application Registry, and open **System OAuth → Application Registry**.
2. Click **New**, then **Create an OAuth API endpoint for external clients**.
3. Fill in:
 - **Name:** SourceLace.
 - **Redirect URL:** the redirect URL above.
 - **Public Client:** tick it. This means "no client secret"; ServiceNow then requires PKCE, which SourceLace always uses. If your release has no **Public Client** box, this connector cannot be used on it yet; tell support the release name.

- **PKCE Required:** tick it if the box is there.
 - Leave **Access Token Lifespan** (30 minutes) and **Refresh Token Lifespan** (100 days) as they are. People connect again when the refresh token runs out.
4. Click **Submit**, open the new entry again, and copy the **Client ID**.

Add the source (SourceLace admin)

Kind: **ServiceNow** (servicenow).

Option	Type	Default	Example	What it is
instance	Text	(required)	corvanta	The name before .service-now.com (the full address also works).
client_id	Text	(required)	3f2a...	The Client ID from the Application Registry.

To allow changes, turn on **Allow changes** and list the tables, such as incident, change_request.

What people need in ServiceNow

- Read access to the tables they ask about (for example the itil role for incidents). ServiceNow's ACLs decide everything.
- Listing fields uses the data dictionary (sys_db_object, sys_dictionary, sys_choice). If someone cannot read those, SourceLace falls back to the fields of one record they can see.

Queries

query (language servicenow_query) takes one JSON object with table, query, fields, display_value and limit:

```
{
  "table": "incident",
  "query": "active=true^priority=1^ORDERBYDESCopened_at",
  "fields": ["number", "short_description", "state", "assigned_to"],
  "limit": 50
}
```

query is ServiceNow's own *encoded query*, the same text a list's filter gives with **Copy query**. SourceLace refuses tables that hold the instance's own credentials (such as oauth_credential, or anything ending in _credentials), and refuses javascript: in queries, because it would run script on the instance.

Zendesk

Zendesk also lives at your own address (such as corvanta.zendesk.com), so **your Zendesk admin creates an OAuth client for SourceLace**. It is a *public* client: no secret, PKCE on every sign-in.

Create the OAuth client (Zendesk admin)

1. Open **Admin Center** (the four-squares menu → Admin Center).
2. Go to **Apps and integrations → APIs → OAuth Clients** (on some accounts: **Zendesk API → OAuth Clients**) and click **Add OAuth client**.
3. Fill in:
 - **Name:** SourceLace. **Description:** "Read and update tickets as the signed-in person".

- **Client kind: Public.**
 - **Redirect URLs:** the redirect URL above.
 - **Identifier:** keep the suggested one or type sourcelace. Copy it.
4. Click **Save**. If Zendesk shows a secret anyway, you don't need it; don't paste it anywhere.

Add the source (SourceLace admin)

Kind: **Zendesk** (zendesk).

Option	Type	Default	Example	What it is
subdomain	Text	(required)	corvanta	The name before .zendesk.com.
client_id	Text	(required)	sourcelace	The OAuth client's Identifier .

Sign-in asks for the scope read, plus tickets:write when an admin has made tickets writable.

After turning changes on or off, people connect Zendesk again so their sign-in matches.

Queries and changes

- query (language zendesk_search) takes Zendesk's own search text, such as type:ticket status:open priority:urgent order_by:created sort:desc. Start with type:ticket, type:user or type:organization so every row has the same columns. One line, at most 1,000 characters. Zendesk returns at most 1,000 results per search, and new tickets can take a minute to appear in search.
- get_record on a ticket can add its comments.
- With tickets writable, changes can set subject, status, priority, type, assignee_id, group_id, requester_id, organization_id, tags, due_at, external_id and custom_field_<id>, and add either a public reply (comment) or a private note (internal_note). Updates use Zendesk's "safe update", so they are refused if someone else changed the ticket since the preview.

Jira Cloud

Jira Cloud sign-in goes through Atlassian's central login, so **SourceLace has one Atlassian app**. After someone signs in, SourceLace asks Atlassian which Jira sites their sign-in reaches and uses the one named in the source's site option. Your Jira admin has nothing to register, unless your Atlassian organization restricts which apps people may approve.

Permissions SourceLace asks for: read:jira-work (read projects and issues), read:jira-user (read people's names) and offline_access (stay connected), plus write:jira-work (create and edit issues and comments) only on sources where an admin made issue writable.

Add the source (SourceLace admin)

Kind: **Jira Cloud** (jira).

Option	Type	Default	Example	What it is
site	Text	(none)	corvanta.atlassian.net	The Jira site. Optional if people only ever reach one Jira site; needed if they reach several. A bare name such as corvanta means corvanta.atlassian.net.

To allow changes, turn on **Allow changes** with the object `issue`. **After turning changes on or off, people connect Jira again.**

Each person clicks **Connect**, signs in with their Atlassian account, and approves the site on Atlassian's consent page.

Queries and changes

- `query` (language `jql`) takes plain JQL, such as `project = CORV AND statusCategory != Done ORDER BY updated DESC`, or JSON to choose the columns: `{"jql": "project = CORV", "fields": ["summary", "status", "customfield_10010"]}`. Jira needs at least one condition before `ORDER BY`.
- `get_record` on an issue can add comments and changelog.
- With `issue` writable: create an issue, update its fields, or add a comment. Descriptions and comments are sent as Jira's rich text, built from plain lines. Status changes and deletes are not offered.

When something goes wrong

What you see	What to do
"... needs the instance option: the name before .service-now.com, such as corvanta."	Fill in <code>instance</code> .
"... needs the <code>client_id</code> option: the Client ID of the OAuth app registered in the instance's Application Registry"	Fill in <code>client_id</code> from the Application Registry.
"ServiceNow sign-in failed: ..."	ServiceNow's own reason follows. Check the redirect URL and that Public Client is ticked.
"... holds credentials and is not available through SourceLace"	That ServiceNow table is never readable through SourceLace.
"javascript: runs script on the instance and is not allowed in a query"	Remove <code>javascript:</code> from the encoded query.
"ServiceNow: that table or record does not exist, or you cannot see it."	Check the name; the person's ServiceNow roles may not allow it.
"SourceLace does not delete ServiceNow records. Change its state instead."	Deletes are not offered; update the state.
"... needs the subdomain option: the name before .zendesk.com, such as corvanta."	Fill in <code>subdomain</code> .
"... needs the <code>client_id</code> option: the Identifier of the OAuth client created in Zendesk's Admin Center"	Fill in <code>client_id</code> with the OAuth client's Identifier.
"Zendesk sign-in failed: ..."	Zendesk's own reason follows. Check the redirect URL and that the client kind is Public .
"SourceLace cannot set '...' on a ticket. Use ..."	Only the ticket fields listed above can be changed.

What you see	What to do
"Add either a comment or an internal_note in one change, not both."	Split it into two changes.
"Your account reaches several Jira sites (...). An admin sets the site option on ... to choose one."	Set the site option.
"Your Atlassian account did not give SourceLace access to Connect again and choose that site on Atlassian's consent page."	Connect again and pick the right site.
"Your Atlassian account did not give SourceLace access to any Jira site."	The person has no Jira site, or did not approve one. Connect again.
"the JQL needs a condition before ORDER BY, such as project = CORV"	Add a condition to the JQL.
"Status changes go through a workflow transition, which SourceLace does not offer yet."	Change status in Jira itself.
"The Jira connector is not available yet."	SourceLace's Atlassian app is not set up on this server yet. Contact support.

Data platforms: Snowflake, BigQuery and Databricks

How to connect Snowflake, Google BigQuery and Databricks for read-only SQL. Each person signs in with their own login, so the platform's own permissions apply to every query and schema lookup.

Status: in testing. Built against each vendor's documented REST APIs and covered by automated tests against simulated services; not yet used with a customer's live account.

What people can do

- `search_schema` lists tables and views the person can see; `describe_object` lists a table's columns and types.
- `query` (language `sql`) takes **one** `SELECT` or `WITH ... SELECT` statement in the platform's own SQL. Anything that could change data (`INSERT`, `UPDATE`, `DELETE`, `MERGE`, `CREATE`, `DROP`, `ALTER`, `SELECT ... INTO`, `CALL` and so on), and anything with more than one statement, is refused **before** it is sent. A few functions that reach outside SQL are refused too (Snowflake `SYSTEM$...` functions, Databricks `reflect`).
- Results are capped at 2,000 rows (50 shown at a time), and SourceLace stops reading once it has enough. Results are held in memory for 30 minutes and never written to a database.
- Every query has a time limit of about a minute. A query that runs longer is cancelled on the platform, and the person is asked to narrow it.
- These sources never accept changes. `get_record` is not used: warehouse tables have no record ids, so people query with a `WHERE` clause.
- SourceLace never uses a shared service account for these sources.

The redirect URL for all three is:

<https://sourcelace.onrender.com/connect/callback>

Snowflake

Who sets it up: your Snowflake admin (someone with `ACCOUNTADMIN`, or a role with `CREATE INTEGRATION`), once per Snowflake account.

1. Create the security integration (Snowflake admin)

In a Snowflake worksheet, run:

```
USE ROLE ACCOUNTADMIN;

CREATE SECURITY INTEGRATION SOURCELACE
  TYPE = OAUTH
  ENABLED = TRUE
  OAUTH_CLIENT = CUSTOM
  OAUTH_CLIENT_TYPE = 'PUBLIC'
  OAUTH_REDIRECT_URI = 'https://sourcelace.onrender.com/connect/callback'
  OAUTH_ENFORCE_PKCE = TRUE
  OAUTH_ISSUE_REFRESH_TOKENS = TRUE
  OAUTH_REFRESH_TOKEN_VALIDITY = 7776000; -- 90 days, then people sign in again
```

Why these settings:

- `OAuth_Client_Type = 'PUBLIC'` with `OAuth_Enforce_Pkce = TRUE`: SourceLace proves each sign-in with PKCE instead of a client secret, so there is no Snowflake secret for anyone to copy or for SourceLace to store.
- `OAuth_Issue_Refresh_Tokens`: people stay connected until the refresh token expires, instead of signing in every 10 minutes.

Snowflake blocks ACCOUNTADMIN, ORGADMIN and SECURITYADMIN from OAuth sign-ins by default. Keep it that way: people should use their everyday roles.

If the account has a **network policy**, it must allow SourceLace's outbound IP addresses; ask support@sourcelace.com for the current list.

2. Copy the client id

```
SELECT SYSTEM$SHOW_OAUTH_CLIENT_SECRETS('SOURCELACE');
```

Copy only `OAuth_Client_Id` from the result. SourceLace does not need either client secret: leave them where they are.

3. Add the source (SourceLace admin)

Kind: **Snowflake** (snowflake).

Option	Type	Default	Example	What it is
account	Text	(required)	corvanta-xy12345.snowflakecomputing.com	The account URL. The account identifier alone (corvanta-xy12345) works too. Only *.snowflakecomputing.com addresses are accepted.
client_id	Text	(required)	AbCdEf123...	OAuth_Client_Id from step 2.
warehouse	Name	each person's default	REPORTING_WH	The warehouse queries run on.
role	Name	each person's default	ANALYST	If set, people sign in with this role only (Snowflake shows it on the consent screen).
database	Name	(none)	SALES	Default database for queries. With it, search_schema looks in that database; without it, across the account.
schema	Name	(none)	PUBLIC	Default schema for queries.

How SourceLace talks to Snowflake: sign-in at `https://<account>/oauth/authorize`; queries through Snowflake's SQL API with a 60-second statement timeout, `MULTI_STATEMENT_COUNT = 1` (Snowflake itself refuses a second statement), and the query tag `sourcelace`, so your Query History shows exactly what SourceLace ran and for whom.

Google BigQuery

Who sets it up: SourceLace provides the Google app people sign in with. Your Google Cloud admin gives people the roles below, and your SourceLace admin adds the source with your project id.

Sign-in asks only for `https://www.googleapis.com/auth/bigquery.readonly`, plus `openid` and email (to show who is connected). If your Google Workspace restricts third-party apps, allow SourceLace's app under **Google Admin console** → **Security** → **Access and data control** → **API controls**.

What each person needs in Google Cloud

- **BigQuery Job User** (`roles/bigquery.jobUser`) on the project in the project option, so they can run queries there. Queries are billed to that project.
- **BigQuery Data Viewer** (`roles/bigquery.dataViewer`) on the datasets they should read, as they would for the BigQuery console.

Add the source (SourceLace admin)

Kind: **Google BigQuery** (`bigquery`).

Option	Type	Default	Example	What it is
project	Text	(required)	corvanta-data	The Google Cloud project id that runs, and pays for, queries.
location	Text	(none)	US, EU, europe-west2	Where the data lives.
max_bytes_billed	Number of bytes	10000000000 (10 GB)	50000000000	The most one query may scan.

Cost protection: every query is first sent as a **dry run**, which is free. If it would scan more than `max_bytes_billed`, it is refused with the size it would have scanned, before anything is billed. The real query is then sent with `maximumBytesBilled` set, so BigQuery enforces the cap too, and with the label `source=sourcelace`. Tables are named `dataset.table` (or ``project.dataset.table``).

Databricks

Who sets it up: your Databricks **account** admin, once per Databricks account. Works on AWS (`*.cloud.databricks.com`), Azure (`*.azuredatabricks.net`) and Google Cloud (`*.gcp.databricks.com`).

1. Create the app connection (Databricks account admin)

1. Open the account console: `accounts.cloud.databricks.com` (AWS), `accounts.azuredatabricks.net` (Azure) or `accounts.gcp.databricks.com` (Google Cloud).
2. **Settings** → **App connections** → **Add connection**.
3. **Name:** SourceLace. **Redirect URLs:** the redirect URL above.
4. **Access scopes: All APIs.** SourceLace itself only calls the SQL statement and Unity Catalog read endpoints; each person's own grants still decide what they can see. Sign-in asks for `all-apis` and `offline_access`.

5. **Untick "Generate a client secret"**. SourceLace signs in as a public client with PKCE, so there is no secret to store.
6. Leave the token lifetimes as they are (or set the refresh token to 90 days). Click **Add**, then copy the **Client ID**.

2. Pick the SQL warehouse

In the workspace, **SQL Warehouses → (the warehouse) → Overview**: copy the warehouse **ID** (16 characters, such as 1234567890abcdef). Each person needs **Can use** on it, plus the usual Unity Catalog grants (USE CATALOG, USE SCHEMA, SELECT) on what they should read. A stopped serverless warehouse starts on the first query, which can take a few seconds.

3. Add the source (SourceLace admin)

Kind: **Databricks** (databricks).

Option	Type	Default	Example	What it is
host	Text	(required)	corvanta.cloud.databricks.com	The workspace address. Only Databricks addresses are accepted.
client_id	Text	(required)	1a2b3c4d-...	The app connection's Client ID from step 1.
warehouse_id	Text	(required)	1234567890abcdef	The SQL warehouse ID from step 2.
catalog	Name	(none)	main	Default catalog. With it, search_schema looks only in that catalog.
schema	Name	(none)	sales	Default schema.

Tables are named `catalog.schema.table`. A query still running after 50 seconds is cancelled rather than left running.

When something goes wrong

What you see	What to do
"Set the Snowflake client_id option: the OAUTH_CLIENT_ID of the account's SourceLace security integration"	Fill in <code>client_id</code> (step 2).
"The Snowflake account option must be a host name such as corvanta-xy12345.snowflakecomputing.com."	Fix account.
"The Snowflake warehouse option must be a plain name such as ANALYTICS." (or role, database, schema)	Use a plain name, without quotes or dots.
"Snowflake sign-in failed: ..."	Snowflake's own reason follows. Check the redirect URI in the security integration matches exactly, and that the person is not signing in with a blocked role such as ACCOUNTADMIN.
"Snowflake took too long to run this query, so it was stopped. Try a narrower query."	Narrow the query.

What you see	What to do
"Google did not grant read access to BigQuery. Connect again and tick the BigQuery box on Google's consent screen."	Connect again and tick the box.
"Set the BigQuery project option to a Google Cloud project id, such as corvanta-data."	Fill in project.
"This query would scan ..., more than this source's limit of Select fewer columns, filter on the partition column, or ask an admin to raise max_bytes_billed."	Narrow the query, or raise max_bytes_billed.
"BigQuery: Access Denied: ..."	The person lacks BigQuery Job User on the project or Data Viewer on the dataset.
"Set the Databricks client_id option ..." or "Set the Databricks warehouse_id option ..."	Fill in the option from the steps above.
"The Databricks host option must be a host name such as corvanta.cloud.databricks.com."	Fix host.
"Databricks took too long to run this query, so it was stopped. Try a narrower query, or check that the SQL warehouse is running."	Narrow the query, or start the warehouse.
"'...' is not allowed in a read-only ... query." or "SELECT ... INTO is not allowed in a read-only query."	Only plain reads are allowed.

Business apps: Dynamics 365, NetSuite and Gong

How to connect Microsoft Dynamics 365 (Dataverse), Oracle NetSuite and Gong.

Status: in testing. All three are built against each vendor's documented API and covered by automated tests against simulated services; none has been used with a customer's live account yet.

Source	Kind	Reads	Changes	People sign in with
Microsoft Dynamics 365	dynamics365	OData queries, tables and columns, records with related rows	Create and update one record at a time, after the person confirms (off unless an admin turns it on). No deletes.	Their own work or school account
Oracle NetSuite	netsuite	SuiteQL queries, record types, records with sublists	Create and update one record at a time, after the person confirms (off unless an admin turns it on). No deletes.	Their own NetSuite login and role
Gong	gong	Calls, call details, transcripts, users, activity and interaction stats	None (read-only)	A Gong technical admin's login, company-wide (see Gong)

As everywhere in SourceLace, queries are checked before anything is sent, results are held in memory for 30 minutes and never written to a database, and sign-in tokens are encrypted and never shown. Gong transcripts are read only when someone asks for them.

The redirect URL for all three is:

<https://sourcelace.onrender.com/connect/callback>

Microsoft Dynamics 365

Each person signs in with their own Microsoft work or school account. Dataverse then applies that person's security roles, business units and field security to everything SourceLace reads or changes.

Who sets it up: SourceLace uses its own Microsoft app (the same one as "Sign in with Microsoft"), so there is nothing for you to register. Sign-in asks for `<org_url>/user_impersonation` (Dynamics CRM, delegated: "act as the signed-in person") and `offline_access` (stay connected).

What your Microsoft admin may need to do: if your organization requires an admin to approve apps, Microsoft shows **Need admin approval**. A Microsoft admin then grants consent for SourceLace in the **Microsoft Entra admin center → Enterprise applications**.

What each person needs: a **Dynamics 365 license and a security role** in that environment, as they would to use Dynamics 365 in the browser.

Add the source (SourceLace admin)

Kind: **Microsoft Dynamics 365** (dynamics365).

Option	Type	Default	Example	What it is
org_url	Web address	(required)	https://corvanta.crm.dynamics.com	The environment's address: Power Platform admin center → Environments → (your environment) → Environment URL , or the address bar when Dynamics 365 is open. Regional environments use crm4, crm11 and so on. Only https://<name>.crm<n>.dynamics.com addresses are accepted.

To allow changes, turn on **Allow changes** and list the tables by their entity set names as search_schema shows them, such as accounts, opportunities.

What people can do

- search_schema lists the tables they can see; describe_object accounts shows columns, choice values, lookups and related lists.
- query (language odata) takes an entity set name plus OData options \$select, \$filter, \$top (1 to 5,000), \$expand and \$orderby, such as accounts?\$select=name,revenue&\$filter=revenue gt 100000&\$orderby=revenue desc&\$top=20. Other options, functions, actions and batch calls are refused before anything is sent.
- Choices and lookups also come back with their display text, in a column named <column>_label.
- get_record accounts <id> reads one record (the id is a GUID), and can add related lists such as contact_customer_accounts.
- **Changes:** propose_change shows the current and new values. The update is sent with the record's version (ETag), so Dynamics refuses it if someone changed the record after the preview. Lookups are set with "<navigation property>@odata.bind": "/contacts(<id>)". Deleting is not offered.

Oracle NetSuite

Each person signs in with their own NetSuite login and picks a role. NetSuite then applies that role's permissions and restrictions to every query and every change. Changes are off until a SourceLace admin turns them on (see [Add the source](#)).

Who sets it up: SourceLace's NetSuite integration is installed into your account by your **NetSuite administrator**, once.

Set up (NetSuite administrator)

1. **Turn on the features:** Setup → Company → Enable Features → SuiteCloud. Under SuiteTalk (Web Services) tick **REST Web Services**; under Manage Authentication tick **OAuth 2.0**. Save.
2. **Install SourceLace's integration:** Setup → Integration → Manage Integrations → Install, using the **Application ID** that support@sourcelace.com gives you. Without this step, sign-in fails with "invalid client".

3. **Give people a role that can use it:** each person's role needs the permissions **REST Web Services** and **Log in using OAuth 2.0 Access Tokens** (both under **Setup**), plus view access to the records they should see (and create or edit access to the records they may change, if changes are on). NetSuite may refuse OAuth 2.0 sign-in with the Administrator role, so people should use their everyday role.

A sandbox (such as 1234567_SB1) is the safest place for a first try. NetSuite sign-ins end after the period set on the integration; people then connect again.

Add the source (SourceLace admin)

Kind: **Oracle NetSuite** (netsuite).

Option	Type	Default	Example	What it is
account_id	Text	(required)	1234567 or 1234567_SB1	Setup → Company → Company Information → Account ID. Sandboxes and release previews end in _SB1, _RP and so on.

Sign-in asks for the scope rest_webservices.

To allow changes, turn on **Allow changes** and list the record types as search_schema shows them, such as customer, contact. Nothing new is needed in NetSuite: the sign-in already includes REST web services, so nobody needs to connect again.

What people can do

- search_schema lists record types; describe_object salesorder shows the fields and sublists.
- query (language sql) takes one SuiteQL SELECT (or WITH ... SELECT), such as SELECT id, companyname, email FROM customer WHERE isinactive = 'F'. All transactions are in the transaction table, with a type column such as 'SalesOrd'. Anything that could change data, SELECT ... INTO and more than one statement are refused before NetSuite is called. Statements can be up to 20,000 characters; no LIMIT is needed, since SourceLace pages through results itself.
- get_record salesorder 42 reads one record by its internal id (digits only); add item (or another sublist) to get its lines, up to 50 per sublist.
- **Changes** (when an admin has turned them on for that record type): propose_change shows a preview (for an update, each field's current value next to the new one), and apply_change makes the change after the person confirms. A create returns the new record's internal id.
 - Only a record's own fields can be set: text, numbers, true/false, or another record's internal id for a reference field (such as subsidiary: 1). Sublists (such as a sales order's lines) and subrecords (such as addresses) are not changed. Unknown fields and fields NetSuite sets itself are refused in the preview, before anything is written.
 - The person's NetSuite role decides what is allowed. If the role cannot change that record, NetSuite's own message comes back.
 - If someone changed the record between the preview and the confirmation, the change is not made and the person is asked to preview it again.
 - **Deletes are refused.** NetSuite records are usually made inactive rather than deleted: update the record's isInactive field to true where the record type has one.

Gong

Gong is **read-only**: calls, call details (participants, Gong's brief, key points, topics, trackers and linked CRM records), transcripts, users, and activity and interaction stats.

Important: Gong's access is company-wide, not per person. Gong lets only its **technical administrators** authorize an app, and the sign-in gives access to the whole company's Gong data; Gong does not apply each person's own Gong permissions to API calls. So:

- Add a Gong source only where everyone who can use it may see all calls and transcripts. Narrow who can use it with [access by group](#).
- The person who connects must be a Gong technical admin.
- Gong does not always say which user signed in, so the audit trail may name the Gong company (gong:<api address>) instead of a person.

Who sets it up: SourceLace has its own Gong app. A Gong technical admin connects and approves it. Sign-in asks for `api:calls:read:basic`, `api:calls:read:extensive`, `api:calls:read:transcript`, `api:users:read`, `api:stats:interaction` and `api:stats:user-actions`.

Each Gong company has its own API address (such as `https://us-12345.api.gong.io`). Gong sends it with the sign-in, and SourceLace only ever calls that address (it must end in `.gong.io`).

Add the source (SourceLace admin)

Kind: **Gong** (gong).

Option	Type	Default	Example	What it is
api_base	Web address	(none: taken from the sign-in)	<code>https://us-12345.api.gong.io</code>	Optional. Pins the source to one Gong company: a sign-in to any other company is refused.

What people can do

query (language gong_json) takes one small JSON object:

Object	Example	Notes
calls	<code>{"object": "calls", "from": "2026-09-01", "to": "2026-10-01", "details": true}</code>	from and to default to the last 30 days (at most a year). Filter with <code>call_ids</code> , <code>user_ids</code> (the call's host) or <code>workspace_id</code> . <code>details</code> adds participants, brief, key points, topics and trackers.
transcripts	<code>{"object": "transcripts", "call_ids": ["7782342274025937895"]}</code>	Up to 20 calls. One row per stretch of speech, with the speaker's name.
users	<code>{"object": "users"}</code>	Everyone in the Gong company.
activity_stats	<code>{"object": "activity_stats", "from": "2026-09-01"}</code>	Calls hosted, attended, feedback and scorecards, per user.
interaction_stats	<code>{"object": "interaction_stats", "user_ids": ["2345994848423"]}</code>	Talk ratio, longest monologue, interactivity, patience, question rate, per user.

Ids are given as text. `get_record` calls `<id>` reads one call with its details and linked CRM records, and adds the transcript when transcript is asked for as a related list. Gong allows about 3 calls a second; when it asks SourceLace to slow down, SourceLace waits and tries again twice.

When something goes wrong

What you see	What to do
"... has no org_url. An admin sets it to the environment's address, such as <code>https://corvanta.crm.dynamics.com</code> ."	Fill in <code>org_url</code> .
"The org_url for ... must be a Dynamics 365 address such as <code>https://corvanta.crm.dynamics.com</code> ."	Copy the Environment URL exactly.
"Microsoft sign-in failed: ..."	Microsoft's own reason follows. If it mentions admin approval or consent, ask your Microsoft admin. If it mentions the scope, check <code>org_url</code> matches the Environment URL.
"Dynamics 365 has no table '...' that you can see. Use <code>search_schema</code> to find one."	Use the entity set name (accounts, not account), or the person's security role does not include that table.
"...' is not allowed. Use only <code>\$select</code> , <code>\$filter</code> , <code>\$top</code> , <code>\$expand</code> , <code>\$orderby</code> ."	Remove the other OData options.
"This record changed in Dynamics 365 after the preview. Propose the change again."	Someone else changed the record; propose the change again.
"Dynamics 365 accepts create and update through SourceLace, not delete."	Delete in Dynamics 365 itself.
"Dynamics 365: not found, or you cannot see it."	Check the id; the person's security role may not allow it.
"... has no account_id. An admin sets it to the NetSuite account id ..." or "The NetSuite account_id for ... must look like 1234567 or 1234567_SB1."	Fix <code>account_id</code> .
"NetSuite sign-in failed: ..."	NetSuite's own reason follows. "invalid client" means the integration is not installed in your account (step 2). Otherwise check the person's role has the two permissions in step 3, and is not Administrator.
"NetSuite did not say which user signed in."	Contact support.
"SELECT ... INTO is not allowed in a read-only query."	Only plain reads are allowed.
"NetSuite: not found, or you cannot see it."	Check the record type and id; the person's role may not allow it.
"NetSuite: your NetSuite role does not allow this. ..."	The person's role lacks create or edit access to that record type. Ask your NetSuite administrator.
"... has no field '...'. Use <code>describe_object</code> to see them." or "... is set by NetSuite and cannot be changed."	Use the field names <code>describe_object</code> lists, and leave out fields NetSuite fills in itself.
"... is a sublist; SourceLace changes a record's own fields only." (or subrecord)	Change lines and addresses in NetSuite itself.
"This ... record changed in NetSuite after the preview. Propose the change again."	Someone else changed the record; ask for a new preview.
"SourceLace does not delete NetSuite records. ..."	Set <code>isInactive</code> to <code>true</code> instead.
"Your NetSuite sign-in has expired." (or Dynamics 365, or Gong)	Connect again.
"You signed in to a different Gong company than ... uses. Connect again with the right Gong account."	Sign in to the Gong company named in <code>api_base</code> .

What you see	What to do
"The Gong api_base for ... must be Gong's API address, such as https://us-12345.api.gong.io."	Fix api_base, or clear it.
"Gong refused this: the sign-in lacks the permission it needs."	The person who connected is not a Gong technical admin, or did not approve every permission. Connect again.
"...' is not a Gong object. Use calls, transcripts, users, activity_stats, interaction_stats."	Use one of those objects.
"The dynamics365 connector is not available yet." (or netsuite, gong)	SourceLace's app for that system is not set up on this server yet. Contact support.

Databases: PostgreSQL, MySQL, SQL Server, Oracle Database and MongoDB

How to connect your own databases for read-only questions. Each person signs in with their own database login, so the database's own grants decide what they see.

Status: in testing. Built against each database's documented behaviour, and tested against a real PostgreSQL and simulated MySQL, SQL Server, Oracle Database and MongoDB servers; not yet used with a customer's live database.

Source	Kind	Covers
SQL database	database	PostgreSQL (including Amazon RDS, Aurora, Azure and Cloud SQL), MySQL and MariaDB, SQL Server and Azure SQL, and Oracle Database 12.1 or later (including Oracle Autonomous Database and Amazon RDS for Oracle). The admin picks the engine.
MongoDB	mongodb	MongoDB, including MongoDB Atlas.

Everything here is **read-only**: `propose_change` refuses changes to database sources.

How sign-in works

Databases have no "Sign in with..." page, so **each person types their own database username and password** into a sign-in form served by SourceLace. There is no shared SourceLace login to your database.

- The sign-in link works once, for 10 minutes, and stops after three wrong passwords.
- SourceLace keeps the password encrypted with your organization's own key, uses it only for that person's requests, and never logs it, shows it, or puts it in the audit trail.
- If the password changes later, SourceLace asks the person to connect again.

1. For your database administrator

Give each person their own read-only login

Create one login per person, with read access only. A group role makes this easy to manage. Narrow it as far as you like (some schemas, tables or collections): SourceLace only ever sees what the person's login can see.

PostgreSQL

```
-- Once: a group that may read the tables people should see.
CREATE ROLE sourcelace_readers NOLOGIN;
GRANT CONNECT ON DATABASE sales TO sourcelace_readers;
GRANT USAGE ON SCHEMA public TO sourcelace_readers;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO sourcelace_readers;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO sourcelace_readers;

-- For each person:
CREATE ROLE ann_lee LOGIN PASSWORD 'a long password' IN ROLE sourcelace_readers;
ALTER ROLE ann_lee SET default_transaction_read_only = on;
```

Use hostssl (not host) lines in pg_hba.conf for these logins, so they can only connect over TLS.

MySQL and MariaDB

```
-- Once:
CREATE ROLE 'sourcelace_reader';
GRANT SELECT, SHOW VIEW ON sales.* TO 'sourcelace_reader';

-- For each person:
CREATE USER 'ann_lee'@'%' IDENTIFIED BY 'a long password' REQUIRE SSL;
GRANT 'sourcelace_reader' TO 'ann_lee'@'%';
SET DEFAULT ROLE 'sourcelace_reader' TO 'ann_lee'@'%';
```

SQL Server

```
CREATE LOGIN ann_lee WITH PASSWORD = 'a long password';
USE Sales;
CREATE USER ann_lee FOR LOGIN ann_lee;
ALTER ROLE db_datareader ADD MEMBER ann_lee;
```

On Azure SQL you can create the user in the database directly (CREATE USER ann_lee WITH PASSWORD = '...') and add it to db_datareader. SourceLace signs in with SQL logins (username and password); Microsoft Entra sign-in to the database is not supported yet.

Oracle Database

Oracle signs people in as database users, and a user's name is also its schema. Create one user per person and give them SELECT through a role:

```
-- Once: a role that may read the tables people should see.
CREATE ROLE sourcelace_reader;
GRANT SELECT ON sales.orders TO sourcelace_reader;
GRANT SELECT ON sales.customers TO sourcelace_reader;
-- (or, on Oracle 23ai, every table of a schema at once: GRANT SELECT ANY TABLE ON SCHEMA sales
  TO sourcelace_reader;)

-- For each person:
CREATE USER ann_lee IDENTIFIED BY "a long password";
GRANT CREATE SESSION TO ann_lee;
GRANT sourcelace_reader TO ann_lee;
```

Existing named users work as they are: SourceLace reads with whatever they can already see. Grant SELECT (or READ), not EXECUTE: a function that changes data inside its own transaction (PRAGMA AUTONOMOUS_TRANSACTION) could get round Oracle's read-only transaction, so only the database's grants stop that.

SourceLace connects with the database's **service name**, such as SALESPDB or sales.corvanta.com (SELECT sys_context('USERENV', 'SERVICE_NAME') FROM dual shows it); the older SID form is not supported. No Oracle Client software is needed, and Oracle Database 12.1 or later works.

MongoDB

```
use sales
db.createUser({ user: "ann_lee", pwd: passwordPrompt(), roles: [{ role: "read", db: "sales" }] })
```

On MongoDB Atlas: **Database Access → Add New Database User**, password authentication, with the built-in role **read** on the database (under *Specific Privileges*).

Let SourceLace in

SourceLace connects from its own servers to your database, so the database must:

1. **Have an address reachable from the internet**, such as `db.corvanta.com` or an RDS endpoint. SourceLace's cloud refuses private addresses (10.x, 172.16-31.x, 192.168.x, localhost), so that no organization can point it at another's internal systems.
2. **Allow SourceLace's outbound IP addresses** in its firewall or security group (AWS security group, Azure SQL firewall rule, Atlas **Network Access → IP Access List**), and nothing else from the internet. Ask support@sourcelace.com for the current list.
3. **Listen on the port** your SourceLace admin enters (usually 5432, 3306, 1433, 1521 or 27017; an Oracle TLS listener is often on 2484, and Oracle Autonomous Database on 1522).

TLS (encryption)

Every connection is encrypted, and SourceLace checks the database's certificate and that it was issued for the host name your admin entered. Make sure:

- TLS is on in the database (`ssl = on` in PostgreSQL, `require_secure_transport = ON` in MySQL, *Force Encryption* in SQL Server, a TCPS listener endpoint in Oracle's `listener.ora`, `net.tls.mode: requireTLS` in MongoDB). Managed services have it on already. For Oracle Autonomous Database, allow TLS connections without a wallet (mutual TLS not required), with SourceLace's IP addresses on its access control list.
- The certificate is valid for the host name people connect to. If you use a different name (a CNAME), add it to the certificate.
- If the certificate was signed by a private certificate authority (Amazon RDS, some Azure servers, or your company's own CA), give your SourceLace admin that CA's certificate (a PEM file starting `-----BEGIN CERTIFICATE-----`). For Amazon RDS this is the regional bundle from AWS's *Using SSL/TLS to encrypt a connection* page.

2. For your SourceLace admin

Add the source on **Manage sources → Add a source**.

SQL database options

Kind: **SQL database** (database).

Option	Type	Default	Example	What it is
engine	One of postgres, mysql, sqlserver, oracle	(required)	postgres	The kind of database. MariaDB uses mysql; Azure SQL uses sqlserver; Oracle Database uses oracle.
host	Host name	(required)	db.corvanta.com	The server's name only: no <code>https://</code> , no port.
port	Number	5432, 3306, 1433 or 1521, by engine	5433	The port. For Oracle, the port of the database's TLS listener (often 2484).

Option	Type	Default	Example	What it is
database	Text	(required)	sales	The database to read. For Oracle, the service name , such as SALESPDB (not the SID).
ca_cert	PEM certificate	(none)	-----BEGIN CERTIFICATE----- ...	The certificate of a private CA (see TLS above). It may be pasted on one line.
tls	verify or off	verify	verify	off is accepted only for a database on the same computer as SourceLace, so in practice always verify.

MongoDB options

Kind: **MongoDB** (mongodb).

Option	Type	Default	Example	What it is
host	Host name	(required)	mongo.corvanta.com or cluster0.abcde.mongodb.net	The server's name, or the Atlas cluster address.
port	Number	27017	27018	The port (not used with srv).
database	Text	(required)	sales	The database to read.
auth_source	Text	the database option	admin	The database where users are defined.
srv	true or false	false	true	true for a mongodb+srv:// address, as MongoDB Atlas uses.
ca_cert, tls				As for SQL databases.

Then each person clicks **Connect** and signs in on SourceLace's form with their own username and password.

3. What people can do, and how it stays read-only

- **SQL databases** take sql: one SELECT (or WITH ... SELECT) in the engine's own dialect, on tables named schema.table as search_schema lists them (SELECT TOP 10 ... on SQL Server, FETCH FIRST 10 ROWS ONLY on Oracle, LIMIT 10 elsewhere). On Oracle, a table name without a schema is looked up in the person's own schema, and names written without quotes are matched in upper case, as Oracle stores them. get_record works on tables with a single-column primary key.
- **MongoDB** takes mongodb: one JSON object, either a find, such as {"collection": "orders", "filter": {"status": "open"}, "sort": {"placed": -1}, "limit": 50}, or an aggregation pipeline, such as {"collection": "orders", "aggregate": [{"_\$match": {"status": "open"}}, {"_\$group": {"_id": "\$region", "orders": {"_\$sum": 1}}}}. Ids and dates use MongoDB

Extended JSON, such as {"\$oid": "..."} and {"\$date": "2026-10-01T00:00:00Z"}.

Checked before anything is sent:

- SQL: a single statement starting with SELECT or WITH; nothing that writes (INSERT, UPDATE, DELETE, MERGE, CREATE, DROP, ALTER, TRUNCATE, GRANT, SELECT ... INTO, COPY, EXEC, CALL and so on), nothing that locks rows (FOR UPDATE, FOR SHARE), no system procedures (xp_..., sp_...), and no functions that reach outside the query or change state (such as pg_sleep, pg_read_file, dblink, nextval, LOAD_FILE, OPENROWSET, WAITFOR). On Oracle, also no PL/SQL packages (DBMS_..., UTL_... such as UTL_HTTP, OWA_..., HTP), no URI types (such as HTTPURITYPE), nothing in the SYS schema, no database links (orders@other_db) and no functions defined inside the query (WITH FUNCTION).
- MongoDB: only read stages (\$match, \$project, \$group, \$sort, \$limit, \$skip, \$unwind, \$lookup, \$facet and similar). \$out and \$merge (which write) are refused, and so are \$where, \$function and \$accumulator (which run code) anywhere in the query. system. collections are off limits.

Enforced on the database too:

Database	Read-only	Time limit
PostgreSQL	A READ ONLY transaction, always rolled back	statement_timeout
MySQL and MariaDB	START TRANSACTION READ ONLY, rolled back	max_execution_time (MariaDB: max_statement_time)
SQL Server	A transaction that is always rolled back, with ApplicationIntent=ReadOnly. SQL Server has no read-only transaction, so the login's db_datareader role matters most.	Query timeout
Oracle Database	SET TRANSACTION READ ONLY first, then the query, then always a rollback	call_timeout on the connection
MongoDB	MongoDB has no read-only session, so the read role and the checks above do this work. Reads prefer a secondary.	maxTimeMS

Each query may run for 30 seconds and returns at most 2,000 rows to SourceLace, 50 shown at a time. Results stay in memory for 30 minutes and are never written to a database. Each request opens its own connection and closes it straight after. Schema comes from the database's own catalog (for Oracle, ALL_TABLES, ALL_VIEWS, ALL_TAB_COLUMNS and ALL_CONSTRAINTS, leaving out the schemas Oracle itself maintains), which lists only what the person can read; for MongoDB, describe_object samples 20 documents and reports field names and types only.

When something goes wrong

What you see	What to do
"Set the option 'host': the database server's name, such as db.corvanta.com." (or 'database', 'engine')	Fill in that option.
"...' is not a host name. Give only the name, such as db.corvanta.com."	Remove https://, the port, and any path.
"TLS can only be turned off for a database on the same computer as SourceLace (localhost)..."	Remove tls: off.
"The option 'ca_cert' must be a PEM certificate (-----BEGIN CERTIFICATE-----).")"	Paste the whole CA certificate, including the BEGIN and END lines.

What you see	What to do
"... is a private network address, and this SourceLace server only connects to databases on the internet..."	Use the database's public address and allow SourceLace's outbound IP addresses.
"... could not be reached. Check the host and port, and that the database allows connections from SourceLace's IP addresses."	Check host, port and the firewall allowlist. If it ends "(the database does not know that service name)", fix the Oracle database option: it must be the service name, not the SID.
"... could not be reached securely: its TLS certificate could not be checked. An admin can set the source's ca_cert option..."	Set ca_cert, or fix the certificate's host name.
"The database refused that username and password too many times. Ask for a new sign-in link to try again."	Connect again for a new link, and check the username and password with your DBA.
"This sign-in link has expired or was already used. Ask for a new one."	Click Connect again.
"... no longer accepts your database username or password." (or MongoDB)	The password changed or the login was removed. Connect again.
"... did not finish the query in time. Narrow it down and try again."	Add filters or a smaller limit.
"... refused the query: ..."	The database's own reason follows, such as a missing grant or an unknown column.
"There is no table or view ... that you can read. Call search_schema to find it."	Check the name (schema.table); the login may lack a grant.
"... has no single-column primary key. Use query with a WHERE clause instead."	Use query instead of get_record.
"'...' is not a collection you can read. Call search_schema to list them."	Check the collection name and the login's read role.

Oracle Fusion Cloud and Workday

How to connect Oracle Fusion Cloud (ERP, SCM and HCM) and Workday. Both read, and both can also make changes once a SourceLace admin turns them on: Oracle Fusion creates and updates records, and Workday starts one business process, the business title change (see [Making changes](#)). Each person signs in with their own Oracle or Workday login, so Oracle and Workday show them only what their own roles and security groups allow. SourceLace never uses an administrator or integration system account, for reads or for changes.

Status: in testing. Built against Oracle's and Workday's documented REST APIs and covered by automated tests against simulated systems; not yet used with a customer's live Oracle pod or Workday tenant.

How this differs from other sources

There is no SourceLace-wide app for Oracle or Workday. **Your Oracle or Workday admin registers an OAuth client for SourceLace inside your own system**, and your SourceLace admin types its details into the source.

- The **client secret** is stored encrypted with your organization's own key. After saving, SourceLace only ever shows (saved). To keep it when editing the source, leave the field empty; to change it, type the new one.
- Type the secret on the web app's **Manage sources** page, not in a chat, so it never sits in a chat transcript.
- SourceLace sends a PKCE challenge with every sign-in. A public client (no secret) also works where your policy prefers it: leave the secret empty.

The redirect URL both systems ask for is:

`https://sourcelace.onrender.com/connect/callback`

Oracle Fusion Cloud

Register SourceLace (Oracle identity domain administrator, about 15 minutes)

You need the **Identity Domain Administrator** role for the identity domain your Fusion users sign in with.

1. Sign in to the **OCI Console** (cloud.oracle.com). Open **Identity & Security → Domains** and click the identity domain your Fusion users sign in with. (Older pods call this **Oracle Identity Cloud Service**; the steps are the same.)
2. On the domain's **Overview**, copy the **Domain URL**, such as `https://idcs-0123456789abcdef.identity.oraclecloud.com`. This is the `idcs_url` option.
3. Open **Integrated applications → Add application → Confidential Application → Launch workflow**. **Name:** SourceLace. Click **Next**.
4. Under **Client configuration**, choose **Configure this application as a client now:**
 - **Allowed grant types:** tick **Authorization code** and **Refresh token**, nothing else.

- **Redirect URL:** the redirect URL above.
 - **Client type: Confidential.**
 - **Token issuance policy → Resources:** tick **Add resources**, click **Add scope**, choose your pod's Fusion Applications resource, and add its scope (usually `urn:opc:resource:consumer::all`).
 - Click **Next**, then **Finish**.
5. Click **Activate** on the new application.
 6. On its **OAuth configuration** tab, copy the **Client ID**, the **Client secret (Show secret)**, and the **scope** shown next to the Fusion resource, exactly as shown with no added spaces, such as `urn:opc:resource:fa:instanceid=123456789urn:opc:resource:consumer::all`. SourceLace adds `offline_access` itself, so people stay signed in.
 7. Note the address people use for Fusion, such as `corvanta.fa.us2.oraclecloud.com`. This is the host option.

Add the source (SourceLace admin)

Kind: **Oracle Fusion Cloud (ERP, SCM, HCM)** (`oracle_fusion`).

Option	Type	Default	Example	What it is
host	Host name	(required)	<code>corvanta.fa.us2.oraclecloud.com</code>	The Fusion address. Must end in <code>.oraclecloud.com</code> .
idcs_url	Host name	(required)	<code>https://idcs-0123456789abcdef.identity.oraclecloud.com</code>	The identity domain URL from step 2. Must end in <code>.identity.oraclecloud.com</code> .
client_id	Text	(required)	<code>0a1b2c3d4e...</code>	From step 6.
client_secret	Secret	(none: public client)		From step 6. Stored encrypted; shown only as (saved).
scope	Text	(required)	<code>urn:opc:resource:fa:instanceid=123456789urn:opc:resource:consumer::all</code>	The Fusion scope from step 6.
resources	List, separated by commas	(none)	<code>receivablesCustomerAccountActivities,hcm/salaries</code>	Extra REST resources to list in <code>search_schema</code> .

What people can do

- `search_schema` lists common resources (Payables invoices, suppliers, purchase orders, requisitions, sales orders, items, projects, and HCM workers, absences, jobs and positions) plus any in `resources`. Any other Fusion REST resource can still be queried by name.
- `describe_object` reads the resource's own description from Oracle: its attributes and child resources.
- `query` (language `fusion_rest`) takes a resource with Oracle's REST options, such as `invoices?q=InvoiceAmount > 1000 and PaymentStatusFlag = 'N'&fields=InvoiceId,InvoiceNumber,InvoiceAmount&orderBy=InvoiceDate:desc&limit=50`. HCM resources start with `hcm/` and CX ones with `crm/`. The options allowed are `q`, `fields`, `orderBy`, `finder`, `expand`, `limit` (up to 500) and `offset`; anything else is refused before Oracle is called.

SourceLace pages through results itself.

- `get_record` reads one record by its key, plus up to three child resources (such as an invoice's `invoiceLines`).
- `propose_change` and `apply_change` create and update records, when changes are on (see [Making changes](#)).

Workday

Register SourceLace (Workday security administrator, about 15 minutes)

You need someone who can run the **Register API Client** task. Try it first in an implementation or sandbox tenant.

1. Search for the task **Register API Client** (not "for Integrations": that one is for system accounts, and SourceLace signs each person in with their own login).
2. Fill it in:
 - **Client Name:** SourceLace.
 - **Client Grant Type:** **Authorization Code Grant**. **Access Token Type:** **Bearer**.
 - **Redirection URI:** the redirect URL above.
 - **Refresh Token Timeout (in days):** how long people stay connected, such as 30, or tick **Non-Expiring Refresh Tokens** if your policy allows.
 - **Support Proof Key for Code Exchange (PKCE):** tick it if your tenant shows it.
 - Leave **Public Client** unticked.
 - **Scope (Functional Areas):** at least **Staffing, Organizations and Roles, Time Off and Leave, Jobs and Positions, Contact Information** and **System** (System holds Workday Query Language). Add others, such as Compensation or Benefits, for data people should read.
 - **Include Workday Owned Scope:** tick it.
3. Click **OK**. Workday shows the **Client ID** and **Client Secret** once: copy both now.
4. Search for **View API Clients** and open the **API Clients** tab. Copy:
 - **Workday REST API Endpoint**, such as `https://wd2-impl-services1.workday.com/ccx/api/v1/corvanta_preview`. Its host (`wd2-impl-services1.workday.com`) is the host option, and its last part (`corvanta_preview`) is the tenant option.
 - **Authorization Endpoint**, such as `https://impl.workday.com/corvanta_preview/authorize`. If its host differs from the REST API host, that host (`impl.workday.com`) is the `auth_host` option.

Add the source (SourceLace admin)

Kind: **Workday** (`workday`).

Option	Type	Default	Example	What it is
host	Host name	(required)	wd2-impl-services1.workday.com	The REST API host from step 4. Must end in .workday.com or .myworkday.com.
tenant	Text	(required)	corvanta_preview	The tenant name from step 4.
auth_host	Host name	the host option	impl.workday.com	The Authorization Endpoint's host, if different.
client_id	Text	(required)	MzY4...	From step 3.
client_secret	Secret	(none: public client)		From step 3. Stored encrypted; shown only as (saved).

What people can do

- `search_schema` lists the REST objects (workers, supervisory organizations, job profiles, job families, jobs) and the WQL data sources the person may use (such as `allActiveWorkers`).
- `describe_object` on a WQL data source lists its fields; on a REST object it lists the fields of a sample record and the related lists `get_record` can add.
- `query` takes wql, one Workday Query Language `SELECT ... FROM ...` such as `SELECT workdayID, fullName, businessTitle FROM allActiveWorkers WHERE isManager = true` (no LIMIT needed), or `workday_rest`, a REST collection with `search`, `limit` and `offset`, such as `workers?search=Lee&limit=20`.
- `get_record` reads one REST object by its Workday ID, plus related lists such as a worker's `directReports`, `supervisoryOrganizationsManaged` or `timeOffDetails`.
- `propose_change` and `apply_change` start a business title change, when changes are on (see [Making changes](#)).

Making changes

Changes are **off** until a SourceLace admin turns on **Allow changes** for the source and lists the objects under **Objects that can be changed** (such as `invoices` for Oracle Fusion, or `businessTitleChange` for Workday). As everywhere in SourceLace, the person first sees a preview with the current and new values, and nothing is sent until they confirm. Each change runs as **that person's own Oracle or Workday login**, so their own roles decide whether it is allowed.

Oracle Fusion: create and update

- **What can be changed:** any top-level Fusion REST resource the admin lists, by the same name query uses (such as `invoices`, `suppliers`, `purchaseRequisitions` or `hcm/workers`). A create sends one new record to the resource; an update changes one record.
- **Checked before anything is sent:** field names are checked against the resource's own description from Oracle. An unknown field, or one Oracle marks as not updatable, is refused in the preview. If Oracle says the resource does not allow create or update, that is refused too.
- **The preview** shows the record's current values of just the fields being changed.

- **If someone else changed the record** after the preview, the change is refused and the person is asked to propose it again. SourceLace uses Oracle's record version (ETag) when Oracle returns one; otherwise it compares the record's LastUpdateDate (or, if the resource has none, the values themselves) just before writing.
- **Deletes are refused.** Fusion records are closed or cancelled through their status (for example, cancel an invoice or close a purchase order), so ask for a status change instead.
- **What each person's roles decide:** whether they can create or change that kind of record at all (their Fusion duty roles and privileges, such as *Manage Payables Invoices*) and which records (their data security, such as business units). If Oracle refuses, SourceLace shows Oracle's own message.
- **Setup:** nothing new. The Fusion scope you already gave the OAuth client covers reads and changes; Fusion's roles are what limit changes. Make sure the people who should make changes have the right Fusion roles.

Workday: the business title change

Workday does not let outside apps edit records directly: every change runs a **business process** inside Workday, with its own approvals. SourceLace only offers changes that are a single, documented Workday REST call, so a change can never stop half way. Today that is one:

Object to allow	What it does	Field
businessTitleChange	Starts a Business Title Change for one worker	proposedBusinessTitle (text, up to 255 characters)

- It is proposed as an **update**, with object businessTitleChange and the worker's Workday ID as the record id. The preview shows the worker's current business title.
- **Approvals still apply.** The preview says so: this starts Workday's own business process, so if your Workday setup requires a manager or HR partner to approve, the new title shows only after they do.
- Creates, deletes, any other field and any other Workday object are refused in the preview, before anything is sent. Just before starting the change, SourceLace reads the worker's current title again and refuses the change if it moved since the preview.
- Once changes are on, search_schema lists businessTitleChange and describe_object shows its field.
- **What each person's security groups decide:** whether they may start a business title change for that worker, set by the **business process security policy for Business Title Change** in Workday (for example, workers for themselves, or managers for their team). If Workday refuses, SourceLace shows Workday's own message.
- **Setup:** the API client's scope must include **Staffing**, the functional area that holds the Business Title Change business process (already in the list in step 2 above). If your API client was registered without it, edit the client in **View API Clients** and add it. Then turn on changes for businessTitleChange on the source.
- **Not offered yet:** contact information changes, job changes, time off and other business processes.

When something goes wrong

What you see	What to do
"... needs the option '...', such as"	Fill in that option.
"The option '...' of ... must be a host name ending in ..., such as"	Give only the host name, from the steps above.
"... needs the option 'client_id': the client id of the OAuth client your admin registered for SourceLace."	Fill in <code>client_id</code> .
"... needs the option 'scope': the Fusion scope shown in your identity domain, such as ..."	Copy the scope from step 6, exactly.
"Oracle sign-in failed: ..." or "Workday sign-in failed: ..."	The system's own reason follows, usually a redirect URL that does not match exactly, a wrong client secret, or (Workday) a missing functional area.
"Oracle did not say which user signed in." or "Workday did not say which worker signed in."	Contact support.
"Oracle Fusion: that resource or record does not exist, or your roles do not let you see it."	Check the resource name with <code>search_schema</code> ; the person's Fusion roles may not allow it.
"Workday: that does not exist, or your security groups do not let you see it."	Check the name or id; the person's security groups may not allow it, or the API client lacks that functional area.
"Workday has no WQL data source '...' that you can use."	Check the data source name with <code>search_schema</code> .
"Oracle answered with a web page instead of data (HTTP ...)." (or Workday)	The host option is probably wrong, or the system is down for maintenance.
"Your Oracle sign-in has expired." or "Your Workday sign-in has expired."	Connect again.
"Oracle Fusion: your roles do not allow this. ..."	The person's Fusion roles do not allow creating or changing these records. Ask your Oracle admin.
"Oracle Fusion does not allow ... on ... through its REST API."	That resource cannot be created or updated through Oracle's REST API. Make the change in Fusion itself.
"This ... record changed in Oracle Fusion after the preview. Propose the change again." or "This worker changed in Workday after the preview. Propose the change again."	Someone else changed it; ask for a new preview.
"SourceLace does not delete Oracle Fusion records. ..."	Change the record's status instead, such as cancelling an invoice.
"SourceLace cannot change '...' in Workday. The changes it can propose are: ..."	Only <code>businessTitleChange</code> is offered.
"Workday changes go through business processes, so SourceLace can only start a business title change for one worker: ..."	Propose an update of <code>businessTitleChange</code> with the worker's Workday ID.

Build your own connector

How to connect SourceLace to a system it has no connector for, such as an in-house database, an internal API or a niche app, by building a small web service of your own. This page is the complete protocol: if something you need is not written here, it is not part of the protocol, so tell support@sourcelace.com and we will add it here.

Status: new. The protocol is version 1 and covered by automated tests; it has not yet been used with a customer's live connector. Admins add one as **Custom connector** on **Manage sources**.

How it works

A connector is a small HTTPS web service that you write and run, in any language. SourceLace calls it when someone asks about your system: "what objects are there", "what fields does this one have", "run this query", "get this record", and, if you allow it, "check this change" and "make this change". Every call is signed with a secret you share with SourceLace, and names the person who is asking, so your connector applies your own permissions.

Your code never runs inside SourceLace, and SourceLace never sends your connector anyone's passwords or tokens: only the signed request and the person's email, name (if known), organization id and SourceLace group names.

SourceLace does:

- signs people in, and checks that the person may use this source at all ([access by group](#));
- previews every change and applies it only after the person confirms it, and only when an admin has turned on changes for this source and object;
- writes every call to the audit trail;
- holds results in memory briefly (never in a database), caps the rows it keeps, and stops waiting for your connector after 30 seconds.

Your connector must:

- check the signature and timestamp of every request (see [Signing requests](#));
- **enforce your own system's permissions**: use person to decide what this person may read and change, and return or change nothing they may not;
- **keep** query **read-only**: whatever the query text says, /query must not change any data. Run it with a read-only login or connection, or accept only a format you can check. SourceLace only checks that a query is not empty and at most 10,000 characters, because only your connector knows its query language;
- never put passwords, keys or internal addresses in an answer: `error.message` is shown to the person.

Add a custom source (SourceLace admin)

1. Make a shared secret, a random string of **32 to 256 characters**, for example with `openssl rand -hex 32`. Keep it in your password manager and give it to whoever runs the connector.

2. On **Manage sources** → **Add a source**, choose **Custom connector** (custom), give it a name (such as `custom:inventory`) and a label, and fill in the options below.
3. To allow changes, turn on **Allow changes** and list the objects that may change. The connector must also say `"writes": true` in its manifest.
4. People connect the source once (there is no extra sign-in) and then ask about it like any other source.

Option	Type	Default	Example	What it is
url	Web address	(required)	<code>https://connector.corvanta.com</code>	The connector's address. Must start with <code>https://</code> , with no user name, password, <code>?query</code> or <code>#fragment</code> . A path is fine, such as <code>https://api.corvanta.com/source-lace-connector</code> .
secret	Secret, 32 to 256 characters	(required)		The shared secret. Stored encrypted with your organization's key and never shown again.

In SourceLace's cloud, the connector's address must be on the public internet: private addresses are refused.

Transport

Rule	Value
Base URL	The url option. Every path below is appended to it.
Scheme	<code>https://</code> only.
Methods	GET for the manifest; POST for everything else.
Request body	UTF-8 JSON, Content-Type: <code>application/json</code> . SourceLace sends compact JSON, but don't depend on spacing or key order: parse it.
Success	HTTP 200 with a UTF-8 JSON body. Any other status is an error (see Errors).
Redirects	Not followed. A 3xx answer is an error.
Time limit	SourceLace waits at most 30 seconds for the whole answer.
Size limit	An answer must be at most 10 MB (10,000,000 bytes).
Unknown fields	Both sides ignore JSON fields they don't know. Later versions may add optional fields; they will never change or remove the ones here.
Null	A field marked "or null" may be <code>null</code> or left out; both mean "no value".

Signing requests

Every request, including the GET, has two headers:

Header	Value
SourceLace-Timestamp	When SourceLace sent the request: whole seconds since 1970-01-01T00:00:00Z, as digits, such as 1767225600.

Header	Value
SourceLace-Signature	v1= followed by 64 lowercase hexadecimal characters.

SourceLace computes the signature like this, and your connector must check it the same way:

1. Take the raw request body **exactly as the bytes arrived**. Don't parse and re-encode it first. For the GET request the body is empty: zero bytes.
2. Build the message: the timestamp header's value as ASCII bytes, then one . (byte 0x2E), then the body bytes.
3. Compute HMAC-SHA256 of the message, keyed with the secret's UTF-8 bytes.
4. Write the 32-byte result as 64 lowercase hexadecimal characters and put v1= in front.

Refuse the request (HTTP 401, error code unauthorized) when either header is missing, the timestamp is not only digits, the timestamp is more than **300 seconds** before or after your own clock (keep your clock synchronized with NTP), or the signature is not exactly equal to the one you compute. Compare with a constant-time comparison, such as `hmac.compare_digest` in Python, `crypto.timingSafeEqual` in Node.js or `MessageDigest.isEqual` in Java.

To change the secret without downtime, a connector may accept two secrets for a while: add the new one next to the old one in your connector, change it in SourceLace, then remove the old one.

Worked example

Your code must produce exactly these signatures.

Secret	sourceLace-example-secret-0123456789
Timestamp	1767225600
Body (110 bytes, no line break at the end)	{"person":{"email":"ann@corvanta.com","name":"Ann Lee","tenant":"corvanta","groups":["Sales"]},"text":"stock"}
Message that is signed	1767225600. followed by the body
SourceLace-Signature	v1=fefe61f4029cc019764e6d73619e17f4798c43292f94e370244fd17116c27623

For a GET (empty body) with the same secret and timestamp, the message is 1767225600. and the signature is v1=5c4452cf2f35fa29be49a2e8d9efc44409c6dd800b9abf2c17bf0a27a56f8548.

Check it from a shell:

```
printf '%s' '1767225600.{\"person\":{\"email\":\"ann@corvanta.com\",\"name\":\"Ann Lee\",\"tenant\":\"corvanta\",\"groups\":[\"Sales\"]},\"text\":\"stock\"}' \\
| openssl dgst -sha256 -hmac 'sourceLace-example-secret-0123456789'
# ... fefe61f4029cc019764e6d73619e17f4798c43292f94e370244fd17116c27623
```

Endpoints

Every POST body has a person object. SourceLace verified who this is; once the signature checks, you can trust it.

person field	Type	Meaning
email	string, 1-320 characters	The person's email address, in lowercase.
name	string (up to 200) or null	Their display name, or null when SourceLace does not know it.
tenant	string, 1-100 characters	Their organization's id in SourceLace, such as corvanta. One connector can serve several organizations; use this to tell them apart.
groups	array of strings (up to 500)	The person's SourceLace group names (set by an admin, or synced from single sign-on). May be empty.

The lengths below are maximums your answers must respect. SourceLace treats an answer that breaks any rule here as a failed call, and does not use it.

GET /sourcelace/v1/manifest

What the connector is. No body. SourceLace asks for it when someone connects the source, and keeps the answer for 5 minutes.

Field	Type	Required	Meaning
protocol	integer, 1	no (default 1)	The protocol version. Only 1 exists.
name	string, 1-100	yes	The system's name, such as Corvanta inventory.
description	string, 1-1000	yes	What is in it, in a sentence or two.
version	string, 1-50	yes	Your connector's own version, such as 1.0.0.
query_language.name	string matching <code>^[a-z][a-z0-9_]{0,39}\$</code>	yes	A short name, such as inventory_sql.
query_language.description	string, 1-2000	yes	One paragraph that tells the AI how to write a query: the objects or tables, the syntax, and what is refused. Write it for a reader who has never seen your system.
query_language.examples	1 to 3 strings, each 1-1000	yes	Complete example queries.
writes	boolean	no (default false)	true when the two changes endpoints work.
objects_searchable	boolean	no (default false)	true when /objects filters by text itself. Otherwise SourceLace keeps the objects whose name, label or description contains the text.

```
{
  "protocol": 1,
  "name": "Corvanta inventory",
  "description": "Products, stock levels and warehouses from Corvanta's inventory system.",
  "version": "1.0.0",
  "query_language": {
```

```

    "name": "inventory_sql",
    "description": "One SQLite SELECT statement on the tables products and warehouses. Anything
that changes data is refused.",
    "examples": ["SELECT sku, name, in_stock FROM products WHERE in_stock < 100 ORDER BY in_stoc
k"],
  },
  "writes": true,
  "objects_searchable": false
}

```

POST /sourcelace/v1/objects

The objects (tables, entities, record types) this person can read.

- **Request:** person; text, 0-200 characters, what the person is looking for (empty means all).
- **Answer:** objects, at most 1,000, each with name (1-200, required, used in every other call), label (up to 200, or null) and description (up to 1,000, or null).

```

{"objects": [{"name": "products", "label": "Products", "description": "Everything we sell."}]}

```

POST /sourcelace/v1/describe

One object's fields.

- **Request:** person; object, a name from /objects.
- **Answer:** name, optional label and description, and fields: 1 to 1,000 fields with unique names, each with name (1-200), type, optional label and description, key (true for the one field whose value /record takes as id; at most one) and writable (true when a change may set it). If there is no such object, answer not_found.

Type	JSON value in rows
string	string
integer	number without a fraction
number	number
boolean	true or false
date	string, YYYY-MM-DD
datetime	string, ISO 8601 with a time zone, such as 2026-10-03T14:05:00Z
json	any JSON value

Any value may be null.

POST /sourcelace/v1/query

Run one read-only query in your query language.

- **Request:** person; query, 1-10,000 characters; max_rows, 1-10,000. SourceLace asks for one more row than it keeps, to tell whether a result was cut short.
- **Answer:** columns (at most 500 unique names, in display order), rows (at most max_rows objects whose keys are names from columns; a missing key means null) and truncated (true when more rows matched than you returned).
- If the query is not valid, or would change anything, answer invalid_query with a message that says how to fix it: SourceLace passes it to the AI, which uses it to try again.

```
{"columns": ["sku", "in_stock"], "rows": [{"sku": "SKU-2", "in_stock": 0}], "truncated": false}
```

POST /sourcelace/v1/record

One record by the value of its key field.

- **Request:** person; object; id, 1-500 characters (the key field's value, as text).
- **Answer:** row, an object of field name to value. If there is no such object or record, or the person may not see it, answer `not_found`.

```
{"row": {"sku": "SKU-1", "name": "Trail running shoe", "price": 129.0}}
```

Changes (optional): /changes/check and /changes/apply

POST /sourcelace/v1/changes/check and POST /sourcelace/v1/changes/apply, only when the manifest says "writes": true. SourceLace calls check to build the preview the person sees, and apply only after they confirm it.

A change, in both requests next to person, has operation (create, update or delete), object, record_id (null for create) and fields (at most 100 new values; {} for delete, at least one otherwise).

/changes/check must not write anything. It answers allowed (boolean, required), problems (up to 50 reasons, for the person), before (the record's current values of the fields being changed), record_name (shown in the preview) and version (the record's current version, such as a last-modified time; SourceLace sends it back to apply). If the record does not exist, answer `not_found`; if the person may not change it, forbidden, or allowed: false with problems.

/changes/apply makes the change. Its request also has version, as check returned it (or null). If the record's version is no longer version, don't write and answer conflict. It answers record_id, the record's id (the new id after a create).

```
{"record_id": "SKU-4"}
```

Errors

Any answer other than HTTP 200 is an error. Its body should be:

```
{"error": {"code": "not_found", "message": "There is no product SKU-9."}}
```

message is 1-1,000 characters and **is shown to the person** and their AI, so write it for them and never include secrets.

code	HTTP status	When
invalid_request	400	The request does not follow this protocol (you will mostly see this while testing).
invalid_query	400	The query is not valid in your language, or would change data.
unauthorized	401	The signature or timestamp check failed.
forbidden	403	This person may not do this.
not_found	404	No such object or record (or the person may not know it exists).

code	HTTP status	When
conflict	409	/changes/apply only: the record changed since /changes/check.
unavailable	503 (or 500 for an unexpected failure)	Your system cannot answer right now.

An error answer without this body, a timeout, a refused connection or an invalid answer is reported to the person as a failure of the connector, without details.

Limits

Limit	Value
Time per answer	30 seconds
Size per answer	10 MB
Query text	10,000 characters
Rows per query request	10,000 (and SourceLace keeps at most 2,000)
Objects per /objects answer	1,000
Fields per object	1,000
Columns per query	500
Fields per change	100
Signature clock tolerance	300 seconds
Shared secret	32 to 256 characters

Versioning

This is version 1, and the path contains it (/sourcelace/v1/). Additions that are optional on both sides stay in version 1, so ignore what you don't know. Anything that would break an existing connector will be version 2, under /sourcelace/v2/, and SourceLace will keep calling version 1 connectors.

Building one

Any language that can serve HTTPS and compute HMAC-SHA256 works:

1. Implement signature checking first and test it against the [worked example](#).
2. Implement the manifest and the four POST endpoints (objects, describe, query, record), and the two changes endpoints only if you want changes.
3. Answer errors in the shape above.
4. Run it behind HTTPS on the internet (any container platform, a virtual machine behind a reverse proxy, or your own Kubernetes), with the secret as a secret setting.

SourceLace also has a Python starter kit, `sourcelace-connector` (Python 3.11 or newer): you write one class with objects, describe, query and record methods plus a manifest, and the kit handles signing, validation and errors. Its `sourcelace-connector check <address> --secret <secret>` command calls every endpoint the way SourceLace does and prints a pass or fail list, for a

connector in any language, without changing anything. It is not yet published; ask support@sourcelace.com for a copy.

When something goes wrong

What you see	What to do
"Set the option 'url': the connector's address, such as <code>https://connector.corvanta.com</code> ."	Fill in url.
"The connector's url must start with <code>https://...</code> "	Use the connector's <code>https://</code> address.
"The option 'url' must be a plain address such as <code>https://connector.corvanta.com</code> , with no user name, password, <code>?query</code> or <code>#fragment</code> ."	Remove those parts from url.
"Set the option 'secret': the secret shared with the connector, 32 to 256 characters..."	Enter the shared secret.
"The address of ...'s connector (...) could not be found."	Check the host name in url.
"... is a private network address, and this SourceLace server only calls connectors on the internet..."	Give the connector a public <code>https://</code> address.
"... 's connector refused SourceLace's signature (...). An admin should check that the secret on SourceLace's Sources page matches the connector's, and that the connector's clock is right."	Make the secrets match, and sync the connector's clock.
"... 's connector did not answer within 30 seconds."	Make the connector faster, or ask for less.
"... 's connector sent an answer larger than 10 MB."	Return fewer rows or smaller values.
"... 's connector sent an answer that does not follow the SourceLace connector protocol, so SourceLace did not use it..."	Whoever maintains the connector runs <code>sourcelace-connector</code> check to see what is wrong.
"... 's connector failed (HTTP ...). Try again; if it keeps failing, tell your admin."	The connector answered with an error but no error body. Check its logs.
"... refused the query: ..."	The connector's own reason follows; rewrite the query.
"... does not accept changes: its connector is read-only."	The manifest says <code>"writes": false</code> .

Google Sheets

How to connect Google Sheets, so people can read a sheet by pasting its link and, when an admin allows it, add rows and change cells after confirming a preview.

Status: in testing. Built against Google's documented Sheets API and covered by automated tests against a simulated Google; not yet used with a customer's live sheets.

Source	Kind	Reads	Changes	People sign in with
Google Sheets	google_sheets	The tabs of one sheet, named by its link or id	Add a row, update cells of a row, after the person confirms (off unless an admin turns it on). No deletes.	Their own Google account

How it works

- **Each person signs in with their own Google account.** Google applies that person's own sharing: they read only sheets shared with them, and change only sheets they can edit.
- **A separate source from Google Drive.** A Google Drive source asks only for read-only Drive access and stays read-only. A Google Sheets source asks only for `https://www.googleapis.com/auth/spreadsheets`, plus `openid` and `email` so SourceLace knows who signed in for the audit trail. Connecting one never grants the other.
- **Sheets are named by link.** The Sheets permission cannot list or search files, so people paste the sheet's link (or its id). An admin can also list sheets the team uses often; their tabs then show in `search_schema`.
- **Nothing is stored.** Rows are read when asked and held in memory like any other result (up to 30 minutes), never written to a database.

Set up

SourceLace's Google app (whoever runs this SourceLace server)

Google Sheets signs in through SourceLace's own Google app, the same one as Gmail, Google Calendar and Google Drive, so there is no new app or redirect URL. In the Google Cloud project that holds that app, once:

1. **APIs & Services → Library:** search for **Google Sheets API** and click **Enable**.
2. **Google Auth Platform → Data Access** (or **OAuth consent screen → Scopes**): **Add or remove scopes**, then under **Manually add scopes** paste `https://www.googleapis.com/auth/spreadsheets`, click **Add to table**, tick it, then **Update** and **Save**. `openid` and `.../auth/userinfo.email` are usually there already; tick them if not.

`spreadsheets` is one of Google's "sensitive" scopes: while the app's consent screen is in testing, only its test users can connect. Opening it to everyone needs Google's verification.

Your Google Workspace admin may need to allow SourceLace's app under **Google Admin console → Security → Access and data control → API controls → App access control**, if your organization restricts third-party apps.

Add the source (SourceLace admin)

Kind: **Google Sheets** (google_sheets).

Option	Type	Default	Example	What it is
spreadsheets	List, separated by commas	(none)	https://docs.google.com/spreadsheets/d/1AbC.../edit	Optional. Links or ids of sheets your team uses often, at most 20. Their tabs show in search_schema. People can still use any other sheet shared with them by pasting its link.

To allow changes, turn on **Allow changes** and put * under **Objects that can be changed** (every sheet people can edit), or name tabs as <sheet id>/<tab name>. The sign-in already covers changes, so nobody needs to connect again; Google still refuses changes to sheets the person cannot edit.

What people can do

An object is one tab, written <sheet id>/<tab name>. Anywhere a sheet is named, its full link (<https://docs.google.com/spreadsheets/d/<id>/edit#gid=0>) or its bare id works.

- `describe_object` on a sheet's link lists its tabs. On <id>/<tab name> it lists the tab's columns (from row 1) and the size of its grid. A link with #gid=... picks that tab.
- `query` (language sheets) reads a tab. It takes one JSON object, such as {"sheet": "https://docs.google.com/spreadsheets/d/1AbC.../edit", "tab": "Pipeline", "range": "A1:H500"}. Only sheet is required. Without tab, SourceLace reads the tab the link points at, or the first tab; without range, the whole tab (up to the row limit). With header (the default, true), row 1 of the range names the columns, and an empty header cell is named by its column letter. `limit` returns at most that many rows. Every row also has `_row`, its row number in the sheet. Empty rows are skipped. Values come back as the sheet shows them (dates as dates, currency with its symbol, formula results).
- `get_record` with object <id>/<tab name> and a row number returns that row.

Changes

Through `propose_change` and `apply_change`, like every other source: the person sees a preview, and nothing is written until they confirm.

- **Add a row** (create): adds one row after the table on that tab. Fields are named by the column names in row 1 (any capitalization); columns not named stay empty. Names that are not in row 1 are listed as problems in the preview.
- **Update cells** (update): changes some cells of one row. The record id is the row number (the `_row` value, 2 or more). The preview shows each cell's current value, and a null value empties the cell. Just before writing, SourceLace reads the row and row 1 again; if either changed since the preview (for example someone inserted a row above, so row 12 is now a different row),

the change is refused and must be proposed again.

- **Deleting rows is not offered:** deleting a row changes the number of every row below it. To empty a row, update its cells to `null`.

Values are always written **RAW**: Google stores exactly what was given and never treats it as a formula. Text such as `=IMPORTXML(...)` or `+1 555 0100` stays plain text.

SourceLace only ever reads a sheet's title and tabs, reads cells, appends a row and sets cells. It has no call that deletes rows or tabs, clears ranges, or changes a sheet's structure or sharing.

When something goes wrong

What you see	What to do
"Google did not grant access to Google Sheets. Connect again and tick the box on Google's consent screen."	Connect again and tick the Sheets box.
"Google Sheets: that sheet was not found, or it is not shared with you."	Check the link; ask the sheet's owner to share it with you.
"Google Sheets: you do not have access to this sheet. Ask its owner to share it with you."	Ask the owner to share it.
"Google Sheets: you do not have edit access to this sheet. Ask its owner to share it with you as an editor."	Changes need edit access in Google.
"This sheet has no tab named '...'. Its tabs: ..."	Use one of the tab names listed.
"Google Sheets: that tab or range does not exist. describe_object on the sheet lists its tabs."	Check the tab name and range.
"Row 1 has no column named '...'. Use describe_object to see the column names."	Name fields exactly as row 1 does.
"Google Sheets: row 1 of ... is empty. SourceLace needs column names in row 1 to change rows."	Put column names in row 1 of the tab.
"Row ... of ... changed after the preview (or rows moved). Propose the change again."	Someone changed the sheet; ask for a new preview.
"Google Sheets rows cannot be deleted through SourceLace ..."	Update the row's cells to <code>null</code> instead.
"Google Sheets is limiting how fast SourceLace can call it. Wait a minute and try again."	Wait, then try again.
"The spreadsheets option of ... lists at most 20 sheets."	Shorten the spreadsheets option.
"The google_sheets connector is switched off on this server..."	SourceLace's Google app is not set up on this server yet. Contact support@sourcelace.com .

SAP ECC (on-premise)

How to connect an SAP ECC system in your own data centre: SourceLace reads (and, if you allow it, creates or updates) data through your SAP Gateway OData services, as each person's own SAP user, through a small SourceLace Agent that runs inside your network. No SAP passwords are stored anywhere and no inbound firewall rule is needed.

Status: in testing. Built against SAP's documented behaviour for SAP Gateway OData (version 2), client certificates and certificate-to-user mapping, and covered by automated tests against a simulated SAP Gateway; not yet used with a customer's live SAP system. Tell support@sourcelace.com before you start, so we can help with the first connection.

How it works

1. Your IT team runs the **SourceLace Agent**, a small program (or Docker container), on a machine inside your network that can reach SAP over HTTPS.
2. The agent opens **one outbound connection** to SourceLace over HTTPS (port 443) and keeps it open. SourceLace sends requests over it; nothing connects in to your network.
3. For each request, the agent maps the asking person to an **SAP user** (by a rule you choose) and calls SAP Gateway with a **client certificate for that user** that it signs itself and that lasts a few minutes. Your SAP system trusts the agent's certificate authority and maps each certificate to its SAP user, so SAP applies **that person's own authorizations**. This is the same pattern as SAP Cloud Connector's principal propagation.
4. The agent checks every request against **its own allowlist** first: only the Gateway services you list can be read, and only the entity sets you list under writes can be changed. Deletes, \$batch and anything outside your Gateway paths are refused, whatever SourceLace asks.

SourceLace reaches ECC **only through SAP Gateway OData services**, never through RFC or BAPI calls. If you want SourceLace to use a BAPI's logic, expose it as a Gateway service (for example with transaction SEGW) and add that service to the agent.

The step-by-step guide for your SAP Basis team (installing the agent, making its certificate authority, STRUST, CERTRULE or EXTID_DN, ICM settings and activating services in /IWFND/MAINT_SERVICE) is in SourceLace's repository as `docs/SAP_ECC.md`; support@sourcelace.com can send it to you.

Create the agent's token (SourceLace admin)

1. On **Manage sources**, under **On-premise agents**, click **Add an agent** and give it a name, such as Frankfurt DC.
2. SourceLace shows the agent's **token** (it starts with `sla_`) **once**. Copy it into the agent's token file, or give it to whoever installs the agent through your password manager. SourceLace keeps only a hash of it.
3. Once the agent runs, the list shows it **Online**, with the time it was last seen and the services it offers. **Revoke** stops a token at once and disconnects the agent; make a new one to replace it.

Each organization can have up to 20 agents, for example one per SAP system.

Add the source (SourceLace admin)

On **Manage sources** → **Add a source**, choose **SAP ECC (on-premise)** (ecc).

Option	Type	Default	Example	What it is
agent	Text	(required)	agt_0123456789ab	The id of the on-premise agent, shown under On-premise agents .
gateway_path	Text	/sap/opu/odata/sap	/sap/opu/odata/sap	Where your Gateway services live. The agent must list the same path.

Changes are off unless you turn them on. Name objects as SERVICE/EntitySet, such as ZSALES_ORDER_SRV/SalesOrderSet. A change is only possible when it is allowed in **both** places: here, and in the agent's own `writes` list. Records are created or updated one at a time; SourceLace never deletes records in SAP ECC.

Connect (each person)

Each person clicks **Connect** on **Data Sources** (or asks their AI app to connect). There is no sign-in page: SourceLace asks the agent which SAP user the person is, and shows it. People the agent cannot map are refused with "You have no SAP user in the SourceLace Agent's configuration".

What people can do

- `search_schema` lists the entity sets of the Gateway services the agent offers, read from each service's `$metadata` as the person.
- `describe_object` on SERVICE/EntitySet lists fields, labels, keys and navigation properties.
- `query` (language odata) takes SERVICE/EntitySet plus OData options: `$filter`, `$select`, `$expand`, `$orderby`, `$top`, `$skip`, `$inlinecount` and `sap-language`. For example ZSALES_ORDER_SRV/SalesOrderSet?\$filter=SoldTo eq '1000'&\$select=SalesOrder,NetValue&\$top=20. Only entity sets can be queried, never function imports.
- `get_record` takes a key such as 4711, or a composite key such as SalesOrder='4711',Item='10', and navigation properties as related lists (up to 3).
- `propose_change` and `apply_change` create or update one record, where allowed. SAP Gateway's CSRF token is handled by the agent. When the service sends ETags, SourceLace refuses to apply a change if the record changed after the preview; when it does not, the preview says so.

Security

- **No SAP passwords** exist in SourceLace or in the agent. Each certificate names one SAP user and expires within minutes.
- **The certificate authority's private key stays on the agent's machine.** SourceLace never has it, so SourceLace alone cannot sign in to SAP as anyone.

- **The agent decides what can be reached**, from its own configuration file: SAP's address and client, the Gateway services, the writable entity sets and who is which SAP user. SourceLace cannot change any of it.
- **The agent logs** request ids, methods, service names, HTTP status and timings, never row data, query options, tokens or keys.
- Rows from SAP pass through SourceLace's memory like any other source's and are never stored (How long things are kept).

When something goes wrong

What you see	What to do
"Your organization's SourceLace Agent is offline, so SAP cannot be reached right now..."	The agent is not running or cannot reach SourceLace over outbound HTTPS. A SourceLace admin sees its state under On-premise agents ; your IT team checks the agent's log.
"The SourceLace Agent (...) did not answer within 60 seconds."	SAP or the agent was too slow. Try again with fewer rows (\$top) or fields (\$select).
"You have no SAP user in the SourceLace Agent's configuration: ..."	Ask your SAP or IT team to map you: your email domain in the agent's <code>users.domains</code> , or your email in its <code>[users.table]</code> .
"The SourceLace Agent for ... refused this request: The Gateway service ... is not in the agent's <code>sap.services</code> list."	The agent's allowlist does not include that service. Your IT team can add it.
"SAP did not accept the certificate login for ... (HTTP 401)."	SAP does not trust the agent's certificate authority, or cannot map the certificate to a user. Your SAP Basis team checks STRUST and CERTRULE (or EXTID_DN).
"SAP refused this for ... (HTTP 403)..."	SAP's own authorization check refused it. Ask your SAP team for the access.
"SAP ECC: the service ... has no entity set ..."	Check the name with <code>search_schema</code> . Function imports cannot be queried.
"There is no on-premise agent ... in your organization."	The agent option names an agent that does not exist or was revoked. Use an id from On-premise agents .
"SAP ECC: the record changed after the preview."	Someone changed the record in SAP meanwhile. Propose the change again to see the current values.

Slack and Microsoft Teams

How to connect Slack and Microsoft Teams. SourceLace reads the channels, chats and direct messages each person is in, reads threads, searches messages, looks up people and reads files shared in conversations; in Teams it can also read meeting transcripts. If you allow it, it posts messages, creates channels, adds reactions and more, as the person, after they confirm a preview.

Status: in testing. Built against Slack's Web API and Microsoft Graph's documented Teams APIs, and covered by automated tests against simulated services; not yet used with a customer's live workspace or tenant.

What SourceLace can and cannot do

Kind	Reads	Changes	Who signs in
Slack (slack)	Conversations you are in, messages, threads, search, people, shared files	Post or schedule a message or thread reply (messages), create a channel (channels), create a canvas or update one you created (canvases), add an emoji reaction (reactions)	Each person, with their own Slack account
Microsoft Teams (microsoft_teams)	Teams and channels you joined, your chats, messages, channel replies, search, people, shared files; with meetings on, your meetings, their transcripts and recording details	Post a channel message, channel reply or chat message (messages), create a channel (channels), add a website tab (tabs), add a reaction (reactions)	Each person, with their own Microsoft work or school account

- **Only what you can see.** SourceLace uses each person's own access (Slack user tokens, Microsoft Graph delegated permissions). There is no bot or shared account, so a person sees only the conversations they are a member of.
- **Nothing changes without you.** Every change is off unless an admin allows changes to that object on the source. Each change is previewed (exactly what will happen, and where) and is made only after the person confirms.
- **Never:** edit or delete messages (yours or anyone's); delete or rename channels, canvases, tabs or reactions; change a canvas someone else created; post or react in a conversation that includes people outside your organization (direct or group messages with outside people, Slack Connect channels, Teams shared channels and chats with other organizations); or post to a channel you have not joined. Teams cannot schedule messages, so SourceLace refuses that for Teams.
- **Never stored.** Messages, files and transcripts are read when asked, held in memory for 30 minutes for joins and charts, and never written to a database.

Slack

Kind: **Slack** (slack).

How sign-in works. SourceLace has its own Slack app. Each person clicks Connect, chooses the workspace and allows SourceLace to see their conversations as them. SourceLace asks for these user scopes: channels:read, groups:read, im:read, mpim:read, channels:history, groups:history, im:history, mpim:history, search:read, users:read, users:read.email, files:read; and only for the objects you made writable: chat:write (messages), channels:write and groups:write (channels), canvases:write (canvases), reactions:write (reactions).

What your Slack admin may need to do. If your workspace requires app approval, or you are on Enterprise Grid, a Slack admin approves SourceLace's app (and, on Grid, allows it in each workspace).

Slack's limits for apps outside its Marketplace. Since May 2025 Slack lets apps that are distributed commercially but not yet approved for the Slack Marketplace read a channel or thread only once a minute, 15 messages at a time. Searching is not limited this way. If this slows you down, your Slack admin can create your organization's own internal Slack app (SourceLace support sends the manifest) and enter its Client ID and Client Secret in the source's options: internal apps keep Slack's normal limits.

Option	Example	What it does
team_id	T0123ABCD	Optional. Only sign-ins to this workspace or Enterprise Grid organization are accepted.
client_id	1234567890.1234567890	Optional: your organization's own Slack app.
client_secret		With client_id. Stored encrypted and never shown again.

Microsoft Teams

Kind: **Microsoft Teams** (microsoft_teams).

How sign-in works. Each person signs in with their Microsoft work or school account through SourceLace's Microsoft app (the same one as Microsoft sign-in, Outlook and SharePoint) and approves these delegated permissions: User.Read, offline_access, User.ReadBasic.All, Team.ReadBasic.All, Channel.ReadBasic.All, Chat.Read, Files.Read.All, ChannelMessage.Read.All; with meetings ON, Calendars.Read, OnlineMeetings.Read, OnlineMeetingTranscript.Read.All and OnlineMeetingRecording.Read.All; and only for the objects you made writable: ChannelMessage.Send and ChatMessage.Send (messages, reactions), Channel.Create (channels), TeamsTab.Create (tabs).

What your Microsoft admin needs to do. ChannelMessage.Read.All, OnlineMeetingTranscript.Read.All, OnlineMeetingRecording.Read.All and TeamsTab.Create always need admin consent: in the Microsoft Entra admin center, open **Enterprise applications** → SourceLace → **Permissions** → **Grant admin consent**. If your organization turned off user consent, the admin approves all of them there. Each person needs a license that includes Teams. Transcripts exist only for meetings where transcription was turned on and your Teams meeting policy allows it; Microsoft gives them for meetings the person organized, and may refuse them for meetings the person only attended.

Option	Example	What it does
channel_messages	off	Optional. off leaves out <code>ChannelMessage.Read.All</code> , so no admin consent is needed: people read and search their chats, but not channel messages.
meetings	on	Optional, off by default. on lets people read their Teams meetings, transcripts and recording details (needs admin consent for the transcript and recording permissions).

Queries

Both use the `chat_json` language: one small JSON object. Use `describe_object` for every key.

Object	Example
conversations	<code>{"object": "conversations", "types": ["public_channel", "private_channel"], "name": "sales"}</code> (Slack types: <code>public_channel</code> , <code>private_channel</code> , <code>mpim</code> , <code>im</code> ; Teams: <code>channel</code> , <code>chat</code>)
messages	<code>{"object": "messages", "conversation": "<id>", "after": "2026-10-01", "limit": 50}</code>
replies	<code>{"object": "replies", "conversation": "<id>", "thread": "<thread column>"}</code>
search	<code>{"object": "search", "text": "renewal", "from": "ann", "after": "2026-09-01"}</code>
users	<code>{"object": "users", "text": "ann@corvanta.com"}</code>
files	<code>{"object": "files", "conversation": "<id>"}</code> (in Teams, conversation is required)
file_text	<code>{"object": "file_text", "file": "<id column of files>"}</code> : the file's text in chunks, read on demand
meetings (Teams)	<code>{"object": "meetings", "after": "2026-09-01"}</code> (default: the last 30 days)
transcript, recordings (Teams)	<code>{"object": "transcript", "meeting": "<id column of meetings>"}</code>

Queries are checked before anything is sent: unknown keys, quotes, colons, brackets and ranges over a year are refused. At most 200 messages, 100 search results, 500 conversations or 200 people per query.

Changes, through `propose_change` (operation create unless noted):

Object	Fields
messages	conversation, text, optionally thread; Slack only: <code>post_at</code> (a time with a zone, at most 120 days ahead) to schedule it
channels	name, optionally private (true or false); Teams: team (required) and description
canvases (Slack)	title, markdown, optionally conversation for a channel's canvas; operation update with the canvas id replaces the markdown of a canvas you created

Object	Fields
tabs (Teams)	conversation (a channel), name, url (https)
reactions	conversation, message, emoji (Slack: an emoji name such as thumbsup; Teams: like, heart, laugh, surprised, sad or angry)

When something goes wrong

What you see	What to do
"Slack is limiting how often SourceLace can read conversation history..."	Wait a minute or use a search query; see Slack's limits above.
"You are not a member of that Slack conversation. Join it in Slack first."	Join the channel.
"Your Slack sign-in does not include the permission this needs..."	Connect again (usually after an admin allowed posting).
"Slack refused because of your organization's Slack settings: on Enterprise Grid..."	Ask your Slack org admin to approve the app for the workspace.
"Microsoft needs an administrator of your organization to approve SourceLace's Teams permissions first..."	Grant admin consent, or set channel_messages to off.
"Microsoft Teams: your account has no Teams license, or Teams is turned off for you."	Ask your Microsoft 365 admin for a Teams license.
"SourceLace does not send direct messages to people outside your organization."	By design: post from Slack or Teams yourself.
"Your sign-in to ... has expired. Call connect_source to sign in again."	The token was revoked or expired: connect again.
"Microsoft did not give that meeting's transcript or recordings..."	Check that transcription was on, that you organized the meeting, and that your Microsoft admin granted consent.
"Teams cannot schedule messages through SourceLace..."	Post now, or schedule it in Teams yourself.

Security and privacy

What SourceLace stores, what it never stores, how it protects what it keeps, and where your data goes. Written for your security reviewer: every statement here describes how SourceLace works today. Questions: security@sourcelace.com.

The short version

- **Each person uses their own login** to each system. SourceLace never uses a shared admin account for your systems (the exceptions, Amazon S3 and Gong, are explained below), so people only ever see and change what they already could.
- **Read-only by default.** Changes are off for every source until an admin turns them on for named objects, and even then every change is previewed and only sent after the person confirms it.
- **Your records are not stored.** Query results live in memory for at most 30 minutes and are never written to a database. The one exception is saved chats in the SourceLace app, which are encrypted and deleted after your retention period.
- **Secrets are encrypted with a key for your organization alone**, and never shown, returned or logged.
- **Everything is audited:** every call, by every person and every client, in a tamper-evident trail that holds metadata, not your data.

What SourceLace stores

What	Where and how
Your organization, its people and their roles, groups, sources and their options, skills and projects	SourceLace's database.
Each person's sign-ins to your systems (access and refresh tokens), database passwords, Oracle and Workday client secrets, custom connectors' shared secrets, single sign-on client secrets, your own AI key	SourceLace's database, encrypted with your organization's own key (see Encryption). Never shown again after saving, never returned by any API, never logged.
Saved chats in the SourceLace app: messages, the tables, charts and tool results in them, and files made in them	SourceLace's database, encrypted with your organization's key, and deleted after your retention period (30 days after last use by default; admins choose 1 to 3,650 days). People can delete their own chats at any time.
The audit trail	SourceLace's database. Metadata only (see The audit trail).
AI usage	One row per question: who, when, which model and token counts. Never the question or the answer.
Sign-in codes, one-time links and tokens SourceLace issues to MCP clients	Stored only as SHA-256 hashes, so a copy of the database cannot be replayed as tokens.

What SourceLace does not store

- **Rows from your systems.** Query results are held in memory, for the person who asked, for at most 30 minutes (up to 2,000 rows a query), then forgotten. They are never written to a database, a file or a log, outside saved chats.
- **Files from your file stores.** SourceLace reads a file only when someone asks, pulls its text out in memory, and does not copy, crawl or index files.
- **Passwords to SourceLace.** People sign in to SourceLace with Google, Microsoft or your single sign-on provider.

Encryption

- **In transit:** SourceLace is served over HTTPS. Connections from SourceLace to your systems use TLS, and SourceLace checks the certificate; for databases it also checks the certificate was issued for the host name your admin entered, and TLS cannot be turned off for any database other than one on the same computer as SourceLace.
- **At rest:** each organization has its **own random data key**. Secrets and saved chats are encrypted with it using Fernet (AES-128 in CBC mode with an HMAC-SHA256 integrity check). Your data key is itself stored only in encrypted ("wrapped") form, under a master key held by the hosting provider as a secret and never stored in the database. Without your data key, nothing encrypted for your organization can be read.

Access control

- **Signing in to SourceLace:** with Google, Microsoft, or your own single sign-on (OIDC, such as Okta or Microsoft Entra ID; see [Single sign-on](#)). Only people your admins add, or people from your organization's allowed email domains, can sign in. A session in the SourceLace app lasts 7 days.
- **MCP clients** sign in through SourceLace with standard OAuth. Their access tokens last 1 hour and are refreshed for up to 30 days, after which the person signs in again.
- **Roles:** admins manage people, sources, groups, single sign-on and AI settings; members use the sources they may use. See the [Admin guide](#).
- **Groups:** admins can limit each source to named groups of people, and sync groups from your identity provider. See [Access by group](#).
- **Your systems' own permissions** apply to every call, because each person signs in with their own login. SourceLace adds its own limits on top, never removes any.

Shared access: Amazon S3 and Gong

Two sources cannot use per-person logins, so their access is **per source**: Amazon S3 (one set of read-only AWS credentials or a role per source) and Gong (Gong only lets a technical admin authorize an app, for the whole company). Everyone who may use such a source reads with the same access. Add them only where that is acceptable, and narrow who can use them with groups.

Reads, changes and query checks

- Every query is checked **before** it is sent: one read-only statement in the system's own language, with anything that writes, deletes, locks, calls out or runs code refused. Databases are also queried inside read-only (or always rolled-back) transactions with a time limit. See [How queries stay read-only](#).
- **Changes** are off for every source until an admin turns them on for named objects. Then `propose_change` shows the current and new values, and nothing is sent until the person confirms: in SourceLace's own confirmation prompt where their app can show one, or by typing the confirmation code from the preview. Three wrong codes discard the preview, and previews expire after 10 minutes. Where the system supports it, the change is refused if the record changed after the preview.
- SourceLace never sends email: mail sources only make drafts, which the person sends themselves.

The audit trail

Every tool call is recorded, whether it succeeded, was refused or failed, from the SourceLace app, from every MCP client and from admins. So are sign-ins and every change to people, sources, groups and single sign-on.

Each entry holds: when, who, the tool, the source, the person's id in that system, the query text, the object, the record id, the **names** of changed fields, how many rows came back, the outcome, the error type, and how long it took. It **never** holds row values, field values, passwords, keys or tokens. Note that the query text is kept as written, so a filter value typed into a query (such as a customer name) appears in the trail.

Entries are chained by SHA-256 hashes per organization: editing, removing or reordering an entry breaks the chain, which SourceLace detects. If an entry cannot be written, the call fails rather than returning data that was not recorded. Admins read the trail on the **Audit Log** page. It is kept, and not deleted automatically today.

AI and your data

From your own AI app (MCP clients such as Claude, ChatGPT, Cursor or VS Code):

SourceLace sends nothing to any AI model. It returns results to your AI app, which sends them to its own AI provider under your agreement with that provider.

From the assistant in the SourceLace app: to answer, SourceLace sends the conversation to Anthropic's API: the person's questions, the assistant's earlier replies, your project instructions and skills used, and the results of the tools the assistant called, which can include rows from your systems. Your admin chooses whether this goes through SourceLace's Anthropic account (the default), your organization's own Anthropic key, or your organization's own OpenAI key. With your own key it is under your own agreement with that provider. Requests to OpenAI are sent with OpenAI's store option off, so OpenAI does not keep the conversation for later retrieval.

SourceLace does not train AI models on your data, and does not send it to any AI provider other than the one your admin chose.

Hosting and subprocessors

- **Hosting:** SourceLace runs on **Render**, in its **Oregon (United States)** region: the application and its database. A move to **AWS is planned**, not yet done.
- **Subprocessors:**

Company	What for	What it receives
Render	Hosting the application and database	Everything SourceLace stores (encrypted as described above) and processes.
Anthropic	The assistant in the SourceLace app (only when people use it)	The conversation and tool results, as described in AI and your data .
OpenAI (only if your admin enters your own OpenAI key)	The assistant in the SourceLace app, under your own agreement with OpenAI	The conversation and tool results, as described in AI and your data .
Google, Microsoft, or your single sign-on provider	Signing people in to SourceLace	The sign-in itself, under your own account with them.

Your connected systems (Salesforce, SAP and so on) receive only the calls each person makes with their own login.

Certifications and agreements

SourceLace does not yet hold any security certification or attestation. **SOC 2, ISO 27001 and HIPAA are not yet available**, so neither is a HIPAA Business Associate Agreement. **A GDPR data processing agreement (DPA) is not yet available** either. Data is hosted in the United States only today.

For a security questionnaire, a review call, or to report a vulnerability, email security@sourcelace.com.

Removing access and data

- **A person** can disconnect any source at any time, which deletes their stored sign-in for it, and delete their own chats.
- **An admin** can remove a person, remove a source, or turn changes off at any time.
- To close your organization's account and have its data deleted, email support@sourcelace.com.

Frequently asked questions

Short answers to the questions people ask most, with links to the full pages.

Using SourceLace

Which AI apps work with SourceLace?

Any app that supports MCP (the Model Context Protocol) with remote servers and sign-in, such as Claude, ChatGPT, Cursor and VS Code, plus the assistant in the SourceLace app itself. Add <https://sourcelace.onrender.com/mcp> as a connector or MCP server. See [Getting started](#).

Do I need the SourceLace app if I already use an AI app?

No. The app is optional: it adds saved chats, projects, charts and files, and is where admins manage people and sources. Everything else works from your own AI app.

Why can't I see a record I know exists?

SourceLace signs you in to each system with your own login, so it shows exactly what that system lets you see, nothing more. Check you can see the record in the system itself, and that you connected the right account (`list_sources` shows who you are connected as).

Why does a source say "not connected"?

Each person connects each source once, with their own login. Ask your AI app to connect it, or click **Connect** on the **Sources** page in the app. Sign-ins can also end on the system's side (a changed password, an expired refresh token); then just connect again.

How many rows can a question return?

Up to 2,000 rows per query, shown 50 at a time. Results are held for 30 minutes so your AI app can page through them or join them with another source's results. For more, narrow the query or ask for totals instead of rows.

Can SourceLace change records?

Only where an admin has turned on changes for named objects of a source, and only one record at a time. You always see a preview of the current and new values first, and nothing is sent until you confirm. Some sources never accept changes: Gong, data platforms, databases and file stores (Google Drive, SharePoint and OneDrive, Amazon S3). Workday accepts one kind of change, starting a business title change, which still goes through Workday's own approvals. Google Sheets accepts new rows and changed cells, never deleted rows. See [Turning on changes](#) and the table in [Every system SourceLace connects to](#).

Can I use a Google Sheet?

Yes, once an admin adds a Google Sheets source: paste the sheet's link into your question. You see only sheets shared with you, and change only sheets you can edit. See [Google Sheets](#).

Can SourceLace send email?

No. Mail sources can only make drafts, which wait in your Drafts folder until you send them yourself.

What are skills?

How-tos written by your organization, such as "build a QBR deck". Your AI app lists them at the start of a task and follows a matching one. See [Getting started](#).

Can I ask SourceLace how to set something up?

Yes. Ask your AI app or the assistant in the app: they search this manual with `search_docs` and read a page with `get_doc`, then answer with a link. This manual is also at <https://sourcelace.onrender.com/docs> and as a PDF.

For admins

Who sets up each system?

For most systems, SourceLace already has the app people sign in with, and your SourceLace admin only adds the source. For ServiceNow, Zendesk, Snowflake, Databricks, Oracle Fusion Cloud, Workday, NetSuite and databases, an admin of that system does a one-time setup first. See [Who sets up what](#).

What is the redirect URL?

<https://sourcelace.onrender.com/connect/callback>. Systems where your own admin registers SourceLace ask for it. See [The redirect URL](#).

Can I limit a source to some people?

Yes: [Access by group](#). Groups can be kept in step with your identity provider through [single sign-on](#).

Can we use our own AI key?

Yes, for the assistant in the SourceLace app: **Settings → AI model → Your own Anthropic key** or **Your own OpenAI key**. See [AI settings](#). Your own AI apps use their own provider and are not affected.

Why is a system "switched off on this server" or "not available yet"?

SourceLace's own app for that system has not been set up on the server you use yet. Contact support@sourcelace.com.

Can we connect a system SourceLace has no connector for?

Yes: your developers can build a small connector of their own, and an admin adds it as a **Custom connector** source. See [Build your own connector](#).

Which systems are in testing?

Salesforce and Salesforce Data 360 are available. SAP S/4HANA and SuccessFactors work with SAP's public demo sandbox only for now. Most other connectors are marked **in testing** (see [Every system SourceLace connects to](#)): built against the vendor's documented API and tested against simulated systems, but not yet used with a customer's live account. If you would like to be the first, tell support and we will try it with you.

Security and privacy

Does SourceLace store our data?

Not your records. Query results are held in memory for at most 30 minutes and never written to a database. The exception is saved chats in the SourceLace app, which are encrypted with your organization's own key and deleted after your retention period (30 days by default). See [Security and privacy](#).

Does SourceLace send our data to an AI model?

From your own AI app: no, SourceLace returns results to your app, which uses its own AI provider. From the assistant in the SourceLace app: yes, the conversation and tool results go to Anthropic, through SourceLace's account or your own key, or to OpenAI if your admin entered your own OpenAI key. See [AI and your data](#).

Where is SourceLace hosted?

On Render, in the Oregon (United States) region. A move to AWS is planned.

Is SourceLace SOC 2 certified? Do you sign a DPA?

Not yet. SOC 2, ISO 27001, HIPAA and a GDPR data processing agreement are not yet available. Email security@sourcelace.com for a security review.

Who can see what my people asked?

Your admins, in the audit trail: who called which tool on which source, the query text, and how many rows came back, never the rows themselves. Saved chats are private to the person who had them.

How do we remove someone?

Remove them under **People**. Their SourceLace sign-in stops working at once, and their sign-ins to your systems are deleted. If your organization admits everyone at an email domain, they will come back as a member if they sign in again, so also remove them in your identity provider or remove the domain.