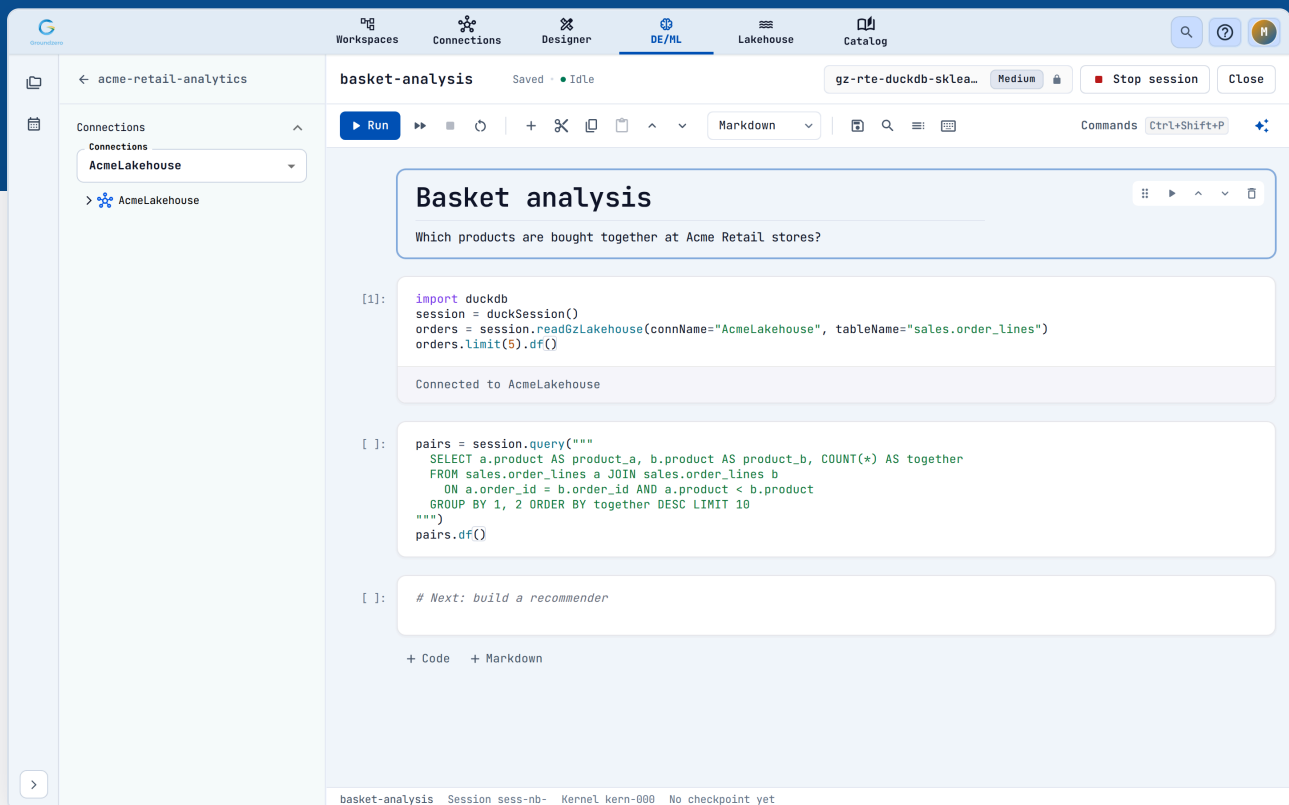




Feature Guide Build and Run

Create projects, run ETL jobs on a schedule, work in notebooks, build datasets, share models and connect your own tools.



Contents

About this guide	4
Find your task	4
Before you start	4
Name things the right way	5
How to read this guide	5
1 Projects	6
1.1 Create a project	6
1.2 Open a project	7
1.3 Rename or delete a project	7
1.4 Find a project	8
1.5 Troubleshooting	8
2 Runtimes and compute	9
2.1 Choose a runtime	9
2.2 Pick a runtime and compute size	9
2.3 Choose a size	11
2.4 How long a runtime runs	11
2.5 Add a custom runtime (administrators)	11
2.6 Limits on cost	11
2.7 Troubleshooting	12
3 ETL jobs	13
3.1 Is a job what you need?	13
3.2 Create a job	13
3.3 Write and run a job	14
3.4 Write job code	17
3.5 Run jobs together	21
3.6 Read the log and stop a run	21
3.7 Share a job	22
3.8 Troubleshooting	23
4 Scheduling and run history	24
4.1 Schedule a job	24
4.2 Find your way around the Scheduler	26
4.3 Change, run or stop a task	27
4.4 Check how runs went	29
4.5 Get an email when a run succeeds or fails	30
4.6 Troubleshooting	32
5 Notebooks	33
5.1 Create a notebook	33

5.2	Start a session	35
5.3	Run code and read the output	36
5.4	Use your connections and tables	38
5.5	Work with cells and modes	38
5.6	Keep your work safe	41
5.7	Move notebooks in and out	41
5.8	Share a notebook	42
5.9	Troubleshooting	42
6	Datasets	43
6.1	Create a dataset	43
6.2	Write and run a query	44
6.3	Choose between a SQL dataset and a code dataset	46
6.4	Read the results	47
6.5	Save a dataset	48
6.6	Use filters in a query	49
6.7	Troubleshooting	49
7	Models and MLflow	50
7.1	Log a model from a notebook	50
7.2	Look at experiments, runs and the registry	50
7.3	Share a model	51
7.4	Use a model	52
7.5	Troubleshooting	52
8	Connecting from MCP clients	53
8.1	Connect a client	53
8.2	What your assistant can do	55
8.3	Troubleshooting	56
9	Glossary	57

About this guide

This guide shows you how to build and run your work in Groundzero. You will create projects, write and run ETL jobs, put them on a schedule, work in notebooks, build datasets, train and share models, and connect your own tools.

Each chapter starts with the task and what you get at the end. It then gives numbered steps, a reference for the fields and options, how to check that it worked, and fixes for the messages you may see.

Screenshots show sample data for a company called Acme Retail. Names such as `AcmeP0SP0stgres` and `daily-sales-rollup` are examples, so use your own.

Find your task

I want to...	Go to
Keep my jobs, notebooks and models together	Projects
Choose where my code runs and how big it is	Runtimes and compute
Move and transform data with Python	ETL jobs
Run a job every morning, hour or week	Scheduling and run history
Explore data and try ideas interactively	Notebooks
Save a query that charts and dashboards can use	Datasets
Train, track and share machine learning models	Models and MLflow
Drive Groundzero from Codex, Claude Code, Cursor or VS Code	Connecting from MCP clients

Before you start

You need	Why
A signed-in Groundzero account with two-step verification set up	Groundzero asks for it before the console opens
At least one connection (on the Connections page)	Jobs, notebooks and datasets all read from a connection
Permission to use compute	To use any runtime except Standard you need a compute size. If the Compute list says Insufficient Permission , ask an administrator for access to compute
Permission to run ETL jobs	Without it, running a job is refused

Your administrator grants permissions. When an action is not allowed, Groundzero tells you with a message such as *You do not have "... permission on ...* or *You are not authorized to perform this action*. Send that message to your administrator.

Name things the right way

Most things you create follow one naming rule. The **Add** or **Save** button stays greyed out until the name is valid.

Item	Allowed characters	Rule
Project, job, notebook, dataset	Letters, digits, hyphen	Start with a letter. Do not end with a hyphen or use two hyphens in a row. No spaces or underscores. Example: <code>daily-sales-rollup</code>
Scheduled task	Letters, digits, dot, hyphen, underscore	No spaces. The form says <i>Name cannot contain space</i>
Imported model	Any text	Must be unique. The dialog says <i>Model Name already taken</i> or <i>Model Name is available</i>

A bad project, job or notebook name shows: *Name must start with a letter, contain only letters, numbers, and hyphens, and cannot end with or contain consecutive hyphens.*

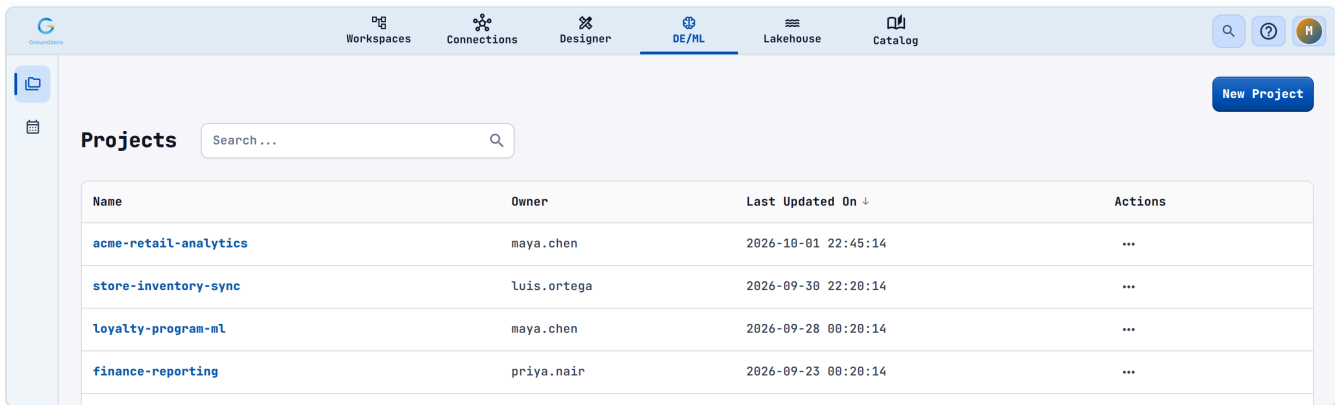
How to read this guide

- **Bold** is a button, menu item or field exactly as it appears on screen.
- `Monospace` is code, or a value you type.
- A **Tip** is a shortcut or good habit. A **Warning** is something that can lose work or cost time. A **Note** is useful background.

1 Projects

A **project** keeps related work together: its jobs, notebooks and models live inside it, and the Scheduler groups tasks by project. After this chapter you can create a project, find your way around it, and rename or delete it.

Projects live under **DE/ML**, in the **Projects** page of the left menu. A project has a name and an owner. The owner is the person who created it, unless an administrator changes it.



Name	Owner	Last Updated On ↓	Actions
acme-retail-analytics	maya.chen	2026-10-01 22:45:14	...
store-inventory-sync	luis.ortega	2026-09-30 22:20:14	...
loyalty-program-ml	maya.chen	2026-09-28 00:20:14	...
finance-reporting	priya.nair	2026-09-23 00:20:14	...

Figure 1. The Projects page lists every project you can see, with its owner and last update time.

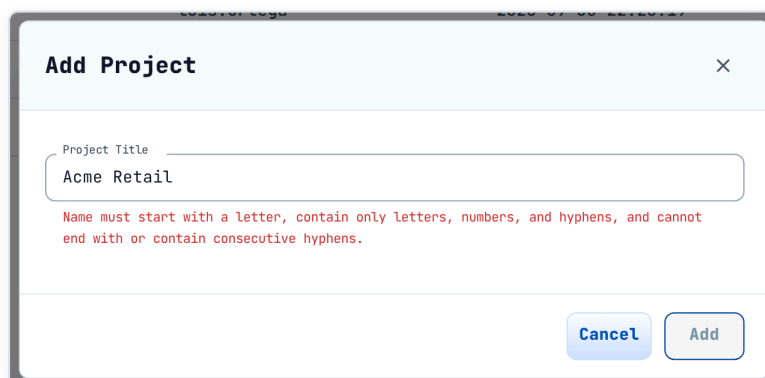
TIP

Use one project for each initiative or team, for example `acme-retail-analytics` or `finance-reporting`, so that everything for it travels together.

1.1 Create a project

You can create projects if your role allows it. If not, the page shows *You are not authorized to perform this action*.

1. Open **DE/ML**, then **Projects**.
2. Select **New Project**.
3. Type a **Project Title**. **Add** stays greyed out until the name is valid.
4. Select **Add**.



Add Project

Project Title

Acme Retail

Name must start with a letter, contain only letters, numbers, and hyphens, and cannot end with or contain consecutive hyphens.

Cancel Add

Figure 2. A name with a space is rejected. The helper text under the field states the rule.

What happens next: The message *Project Added successfully* appears and the project joins the list.

TIP

Use lower-case words joined by hyphens, such as `loyalty-program-ml`. The same rule applies to jobs, notebooks and datasets, so one habit covers all of them.

1.2 Open a project

Select a project name. The project opens on its **Jobs** tab, with **Notebooks**, **Models** and **Agents** tabs beside it. You can move between tabs with the left and right arrow keys. Home and End jump to the first and last tab.

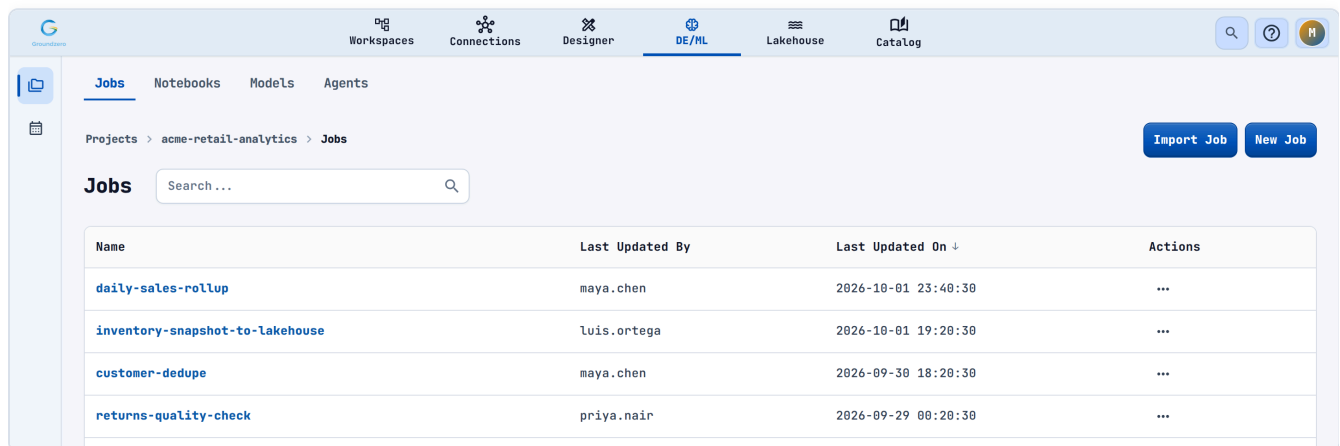


Figure 3. Inside a project, the breadcrumb reads Projects, the project name, then the active tab.

1.3 Rename or delete a project

Open the **Actions** menu (the three dots) on the project row.

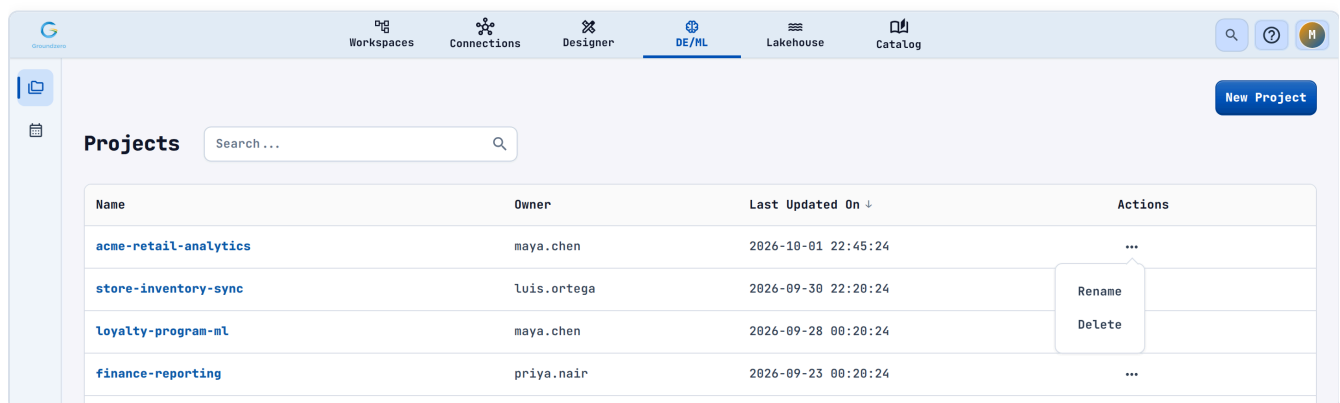


Figure 4. The row menu for a non-administrator offers Rename and Delete.

Action	What it does
Rename	Opens a dialog with the current name. The same name rule applies. You see <i>Project renamed successfully</i>
Delete	Asks <i>Are you sure you want to delete "name"?</i>
Change Owner	Lets an administrator pick a new owner. You see <i>Owner changed to ... successfully</i>

WARNING

A project that still contains jobs cannot be deleted. Only jobs block deletion, so check the Notebooks and Models tabs yourself before you delete a project.

1.4 Find a project

The search box filters by name as you type. The table shows each project's name, owner and the time of its last change (in your local time zone, newest first). It shows 20 rows at a time.

1.5 Troubleshooting

Message or symptom	Cause	Fix
<i>Name must start with a letter...</i>	The name has a space, starts with a digit, ends with a hyphen or has two hyphens in a row	Use letters, digits and single hyphens
<i>Project cannot be deleted as it contains jobs</i>	At least one job is still in the project	Delete the jobs first
<i>Failed to fetch users list</i>	The list of users for Change Owner did not load	Try again. If it keeps failing, contact an administrator

2 Runtimes and compute

Your code does not run in your browser. It runs on a **runtime** (also called an RTE) that Groundzero starts for you, on a **compute** size that sets how much memory and how many processors the runtime gets. After this chapter you can choose both for any job, notebook, dataset or scheduled task.

You choose them in the **runtime picker**, which appears wherever code runs.

2.1 Choose a runtime

Work down this list and stop at the first match.

1. You need one SQL query on one source, for a chart or dashboard: use **Standard**. It has no compute size to choose, and works only in datasets.
2. You need several connections, writes to a table, or Python beyond a few lines: use **DuckDB** or **Spark**.
3. The data is small to medium in size, and you want fast analysis, joins or ETL: use **DuckDB**. It starts quickly and is a good default for notebooks.
4. The data is large, or you need heavy parallel processing: use **Spark**.
5. You need a machine learning library such as scikit-learn, PyTorch or TensorFlow: pick the runtime whose name includes that library.
6. You need extra Python packages that no listed runtime has: ask an administrator for a custom runtime.

Runtime names follow the pattern *engine, then machine learning library, then version*. Your site may list different ones.

Runtime	Use it for	Notes
Standard	Datasets that query one SQL source	Only in the dataset editor. See Datasets
DuckDB	Fast analysis on one machine, including lakehouse tables	Good default for notebooks and small to medium data
Spark	Large, parallel data processing	Has Minimum Workers . Notebook sessions scale between 1 and 10 workers as the load changes
Custom	Anything that needs extra Python packages	Built by an administrator from a base runtime

TIP

Start on Standard or DuckDB and move to Spark when a job outgrows it.

2.2 Pick a runtime and compute size

1. Open the runtime picker in the editor toolbar. Until you choose, it reads **Select runtime**.
2. Open **Runtime** and choose a runtime.
3. Open **Compute** and choose a size.
4. For Spark, set **Minimum Workers** if you need more than the default. For work that needs GPUs, set **GPU Size**.

5. Close the panel.

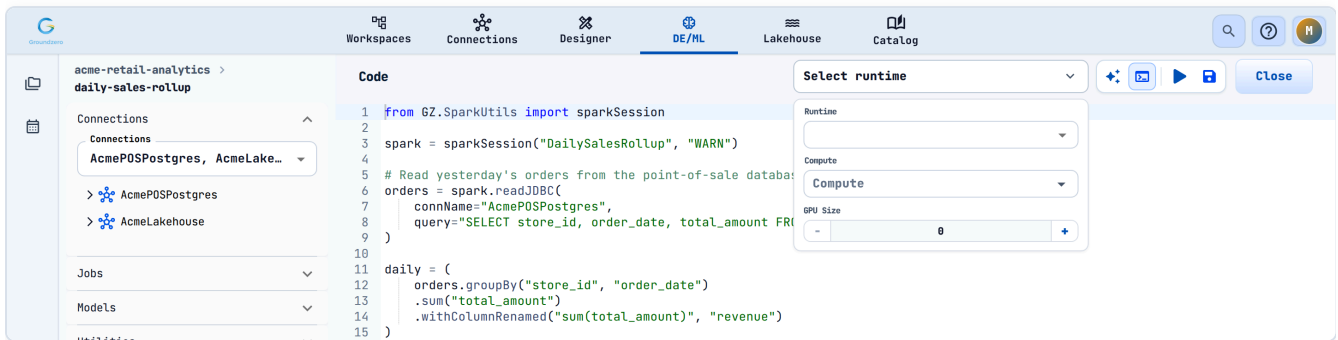


Figure 5. The picker in a job editor: a Runtime list, a Compute list, and a GPU Size stepper.

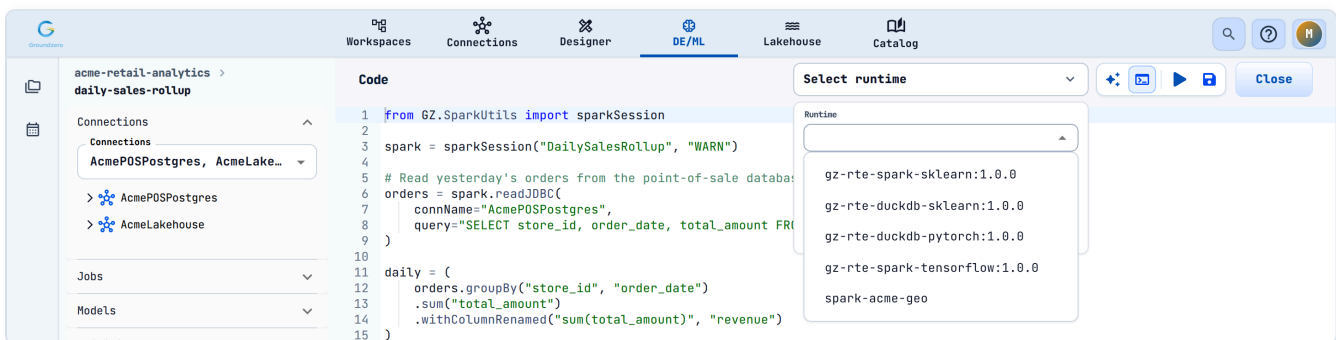


Figure 6. Opening the Runtime list shows the runtimes available to your site.

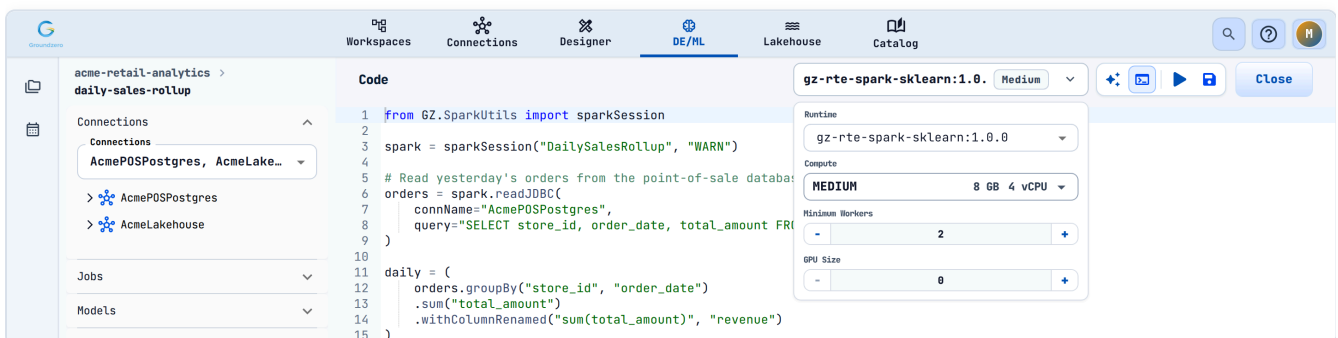


Figure 7. After you choose, the toolbar shows a short summary with the compute name as a badge.

What happens next: The picker shows your choice. The runtime starts when you run the job, start the notebook session or run the dataset.

Field	What it holds
Runtime	The runtimes your site makes available, for example <code>gz-rte-spark-sklearn:1.0.0</code> . If none are available, the list reads No RTEs Available
Compute	A size shown as name, memory and processors, for example Medium, 8 GB Memory, 4 vCPU. If no sizes are listed for you, it reads Insufficient Permission
Minimum Workers	Spark runtimes only. How many workers to keep. Starts at 2 and cannot go below 1
GPU Size	Every runtime except Standard. Starts at 0 and cannot go below 0

2.2.1 How the picker behaves in each place

Where	What to know
ETL job editor	Standard is not offered. Choose a runtime and compute size before Run
Notebook	Standard is not offered. The picker is locked while the session runs. Stop the session to change it
Dataset editor	Standard is offered when you use one SQL connection or none. Choosing another runtime turns the dataset into a code dataset
Scheduler task	Standard is not offered. The RTE field opens a Runtime and Compute dialog, and both are required

2.3 Choose a size

- Start with **Small** while you develop. Choose a bigger size when a run is slow or runs out of memory.
- For Spark, more memory per worker helps wide joins more than more workers do.
- Raise **Minimum Workers** only for work that runs in parallel. More workers than the work needs only adds cost.
- Use **GPU Size** only with runtimes built for GPUs, such as the PyTorch and TensorFlow ones.

TIP

Compute sizes are shared by your whole site. Pick the smallest size that finishes in a time you accept, and stop notebook sessions when you are done.

2.4 How long a runtime runs

- A **job run** starts a fresh runtime for that run and ends when the job ends.
- A **notebook session** keeps running until you select **Stop session** or **Stop Notebook**.
- **Dataset runs** on DuckDB and Spark reuse a session for a few minutes after the last run, so the next run starts faster. A dataset run waits at most 5 minutes for its result.

2.5 Add a custom runtime (administrators)

1. Open the admin console, then **Custom RTE**, then **Add custom RTE**.
2. In **Base Image**, choose the runtime to extend.
3. Type a **Name**. The final name appears under the field.
4. Under **Choose File (.txt file only)**, choose a text file that lists the extra Python packages.
5. Submit the form.

What happens next: You see *RTE creation is in progress, this may take a while*. The table lists each runtime's **Name**, **Created By**, **Created On** and **Status**, so you can watch the build finish. The new runtime then appears in every runtime list. If the build cannot start, you see *Failed to add RTE*.

2.6 Limits on cost

Jobs, notebooks and datasets have no cost limit of their own. The budget set in the admin console limits AI chat only. See the Analyze, AI and Admin guide.

2.7 Troubleshooting

Message or symptom	Cause	Fix
Compute shows Insufficient Permission	Your role cannot use compute sizes	Ask an administrator for access to compute
No RTEs Available	No runtime is visible to your site	Ask an administrator to enable a runtime
<i>please select Compute</i>	You pressed Run with no compute chosen	Choose a Runtime and a Compute size
<i>Please select RTE and Compute</i>	You ran a dataset with neither chosen	Choose both, or choose Standard where it is offered
<i>Invalid RTE.</i>	The runtime you chose cannot run this work	Choose a runtime from the list
<i>The dataset session stopped responding before it took the run. Run the dataset again to start a new session.</i>	The dataset session stopped answering	Run the dataset again

3 ETL jobs

An **ETL job** is Python code that reads data from your connections, transforms it and writes the result somewhere else. After this chapter you can create a job, run it, read its log, stop it, and share it with other projects.

Jobs live in a project, on the **Jobs** tab. A job can also run on a schedule (see Scheduling and run history).

3.1 Is a job what you need?

Use a job when	Use something else when
The work should run without you, on a schedule, or repeatedly with the same code	You are exploring data. Use a notebook
The result should be written to a lakehouse table, file store or database	You only need rows for a chart. Use a dataset
The work is too heavy for a dataset, such as Spark transformations or joins across sources	The data fits one SQL query. A dataset on Standard is quicker

You need a project, at least one connection, permission to run ETL jobs and a compute size you may use. Jobs never run on Standard.

3.2 Create a job

1. In a project, open the **Jobs** tab and select **New Job**.
2. Type a **Job Title** that follows the name rule.
3. Select **Add**.

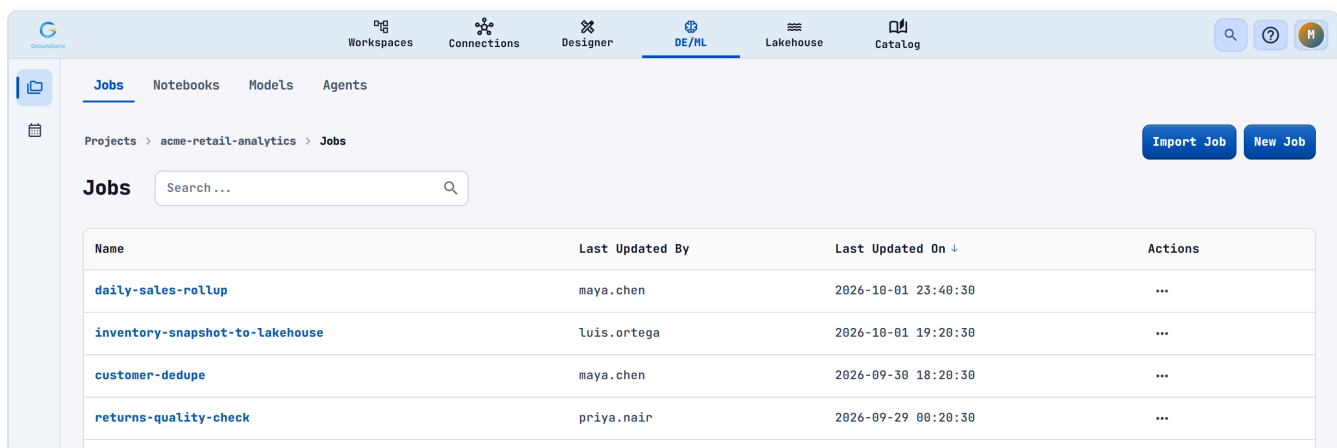
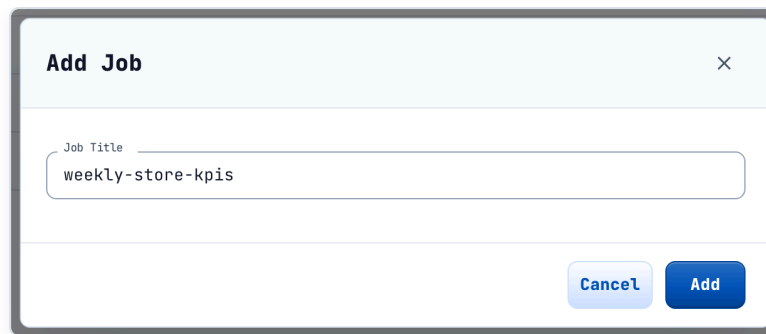


Figure 8. The Jobs tab. Import Job and New Job sit at the top right.

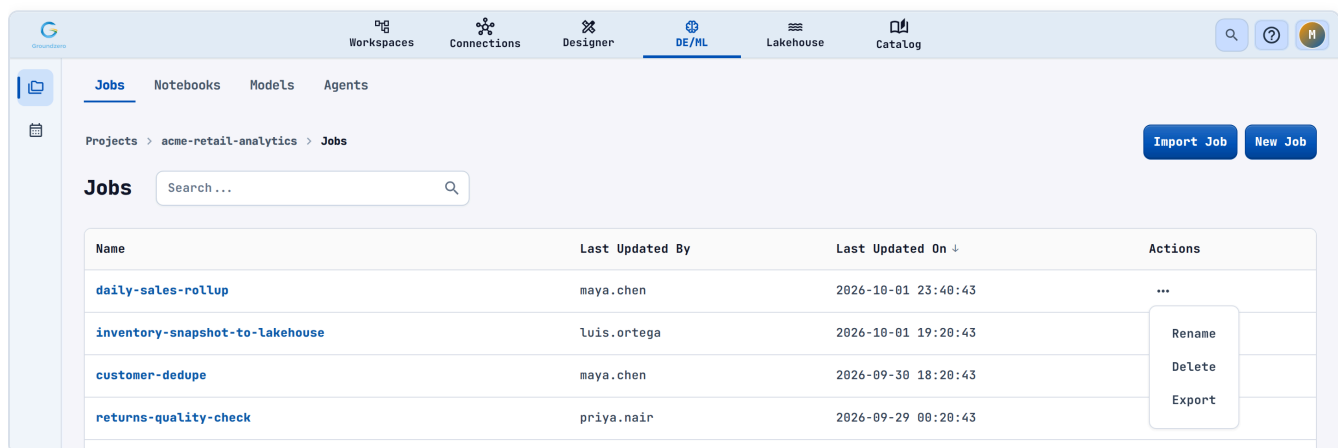


The 'Add Job' dialog box is a light blue window with a close button (X) in the top right corner. It contains a text input field labeled 'Job Title' with the value 'weekly-store-kpis' entered. At the bottom right, there are two buttons: 'Cancel' and 'Add'.

Figure 9. The New Job dialog asks only for a name.

What happens next: You see *Job Added successfully*, and the job appears in the list. Select its name to open the editor.

The list shows each job's name, who saved it last and when. The row menu has **Rename** (*Job renamed successfully*), **Delete** (*Job deleted successfully*) and **Export**.



The screenshot shows the Groundzero interface with the 'Jobs' tab selected. The breadcrumb path is 'Projects > acme-retail-analytics > Jobs'. There are 'Import Job' and 'New Job' buttons. A search bar is present. The table below lists jobs with columns for Name, Last Updated By, Last Updated On, and Actions.

Name	Last Updated By	Last Updated On ↓	Actions
daily-sales-rollup	maya.chen	2026-10-01 23:40:43	...
inventory-snapshot-to-lakehouse	luis.ortega	2026-10-01 19:20:43	Rename
customer-dedupe	maya.chen	2026-09-30 18:20:43	Delete
returns-quality-check	priya.nair	2026-09-29 00:20:43	Export

Figure 10. The row menu for a job.

3.3 Write and run a job

By the end of this task your job has run and you have read its log.

1. Select the job name to open the editor.
2. In the left drawer, open **Connections** and tick the connections the job uses. Your code can use only the connections you tick.
3. Write your code (see the examples below). The editor suggests connection, database, table and column names as you type.
4. Open the runtime picker and choose a **Runtime** and **Compute** size. See Runtimes and compute.
5. Select **Run**. Groundzero saves the job first. If the save fails, nothing runs.
6. Watch the **Messages** pane. It fills with log lines while the job runs.

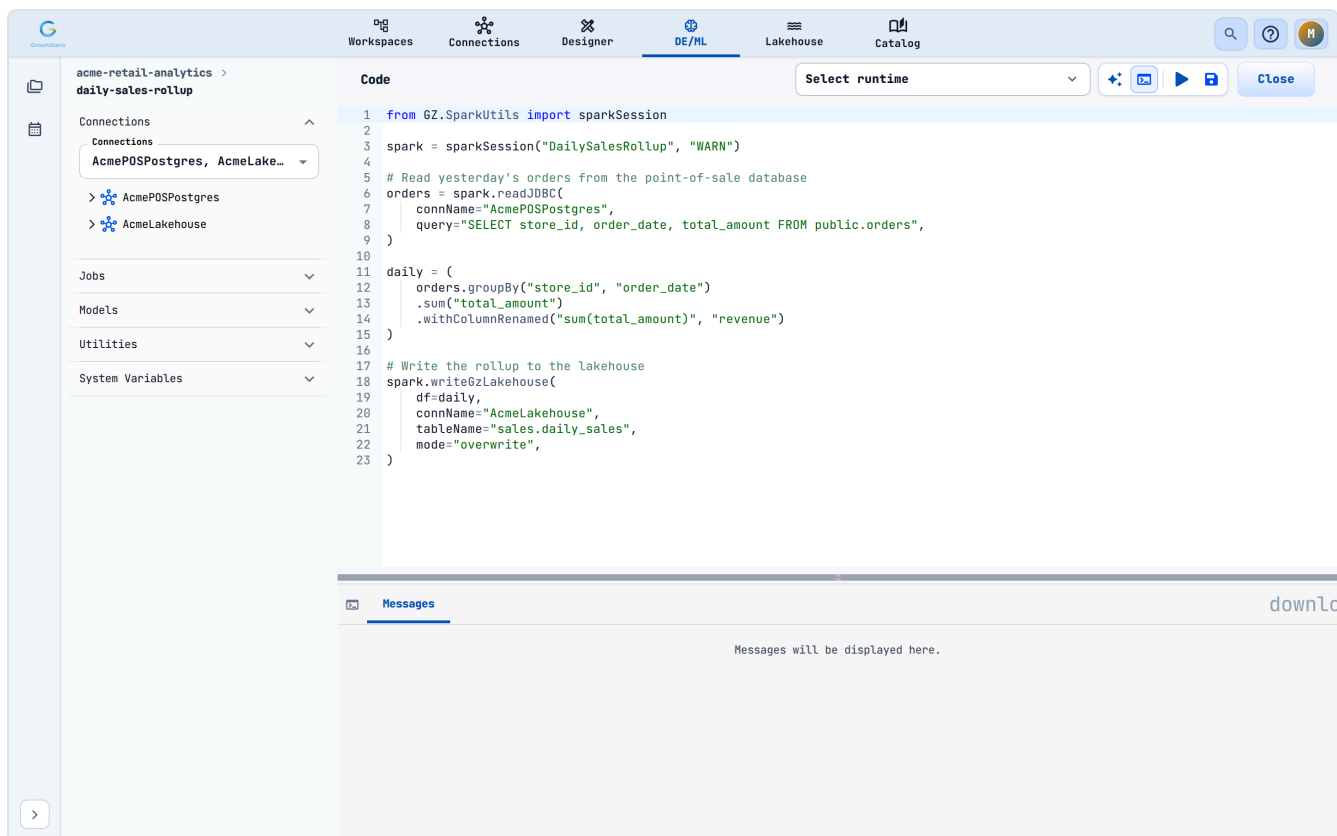


Figure 11. The editor. Left: Connections, Jobs, Models, Utilities and System Variables. Top right: runtime picker, assistant, output toggle, Run, Save, Close. Bottom: the Messages pane.

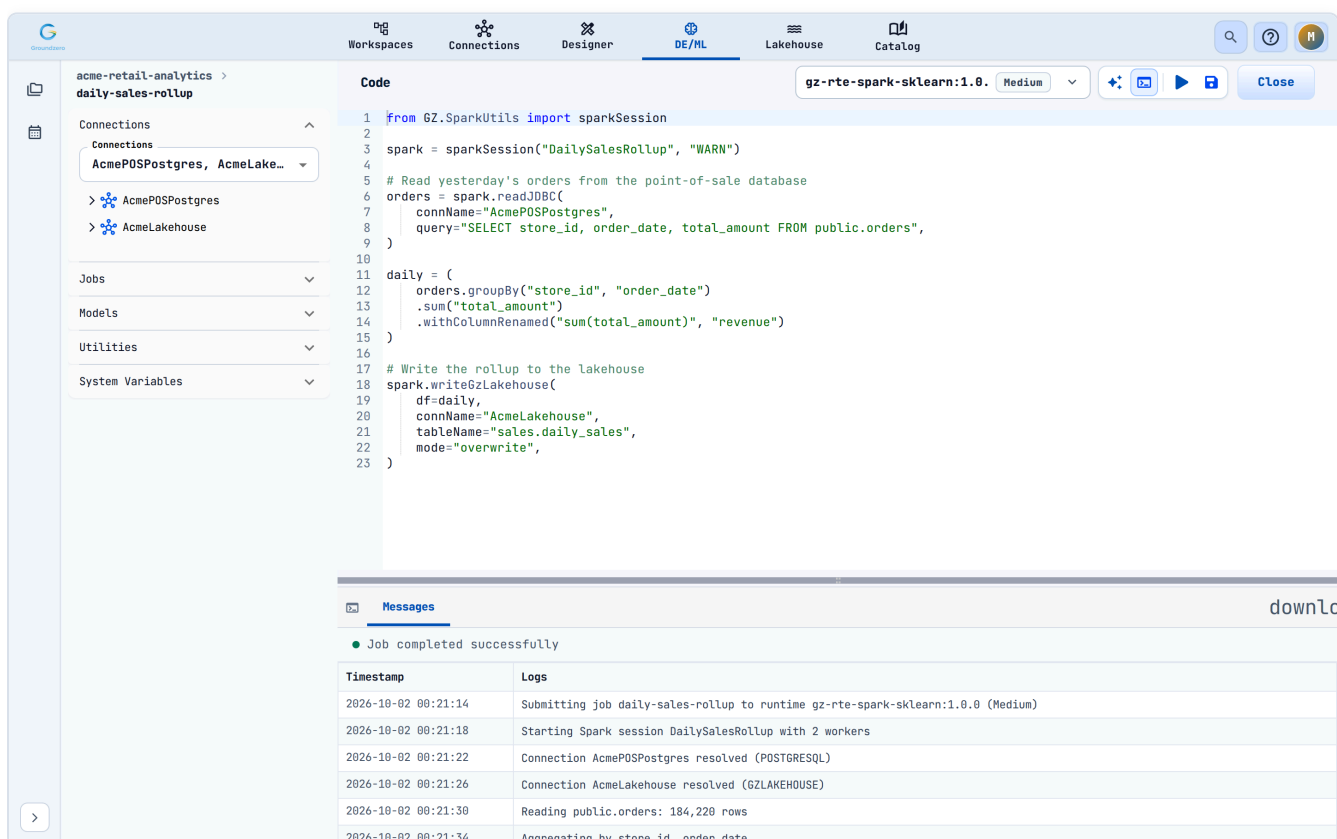


Figure 12. A finished run. The line above the table reads Job completed successfully. The table holds Timestamp and Logs columns.

How to check it worked: The message *ETL job submitted successfully* appears, and the log ends with *Job completed successfully*. If it ends with *Job failed*. Check the logs for the cause., read the last log lines.

TIP

Tick only the connections the job actually uses. The ticked connections are saved with the job and decide which connections your code can reach.

3.3.1 The editor toolbar

Control	What it does
Runtime picker	Opens the Runtime, Compute and GPU Size panel
Assistant (sparkle icon)	Opens the code assistant for this job. Type <code>@Logs</code> to hand it the latest log lines. See the Analyze, AI and Admin guide
Output pane toggle	Shows or hides the Messages pane
Run (play icon)	Saves, then starts the job. While a run is active the icon turns into a red Stop
Save (disk icon)	Saves code and settings. Greyed out while the code is empty. You see <i>Job saved successfully</i>
Close	Returns to the project

3.3.2 The left drawer

Section	Use it to
Connections	Tick the connections to use. Each one shows its databases, tables and columns
Jobs	Pick other jobs from the same project to run together with this one (sub-jobs)
Models	Pick registered models to load before your code starts
Utilities	Search ready-made code templates. Hover over one and select the copy icon. You see <i>Template name code copied to clipboard!</i>
System Variables	Copy placeholders to paste into code. You see <i>\${NAME} copied to clipboard!</i>

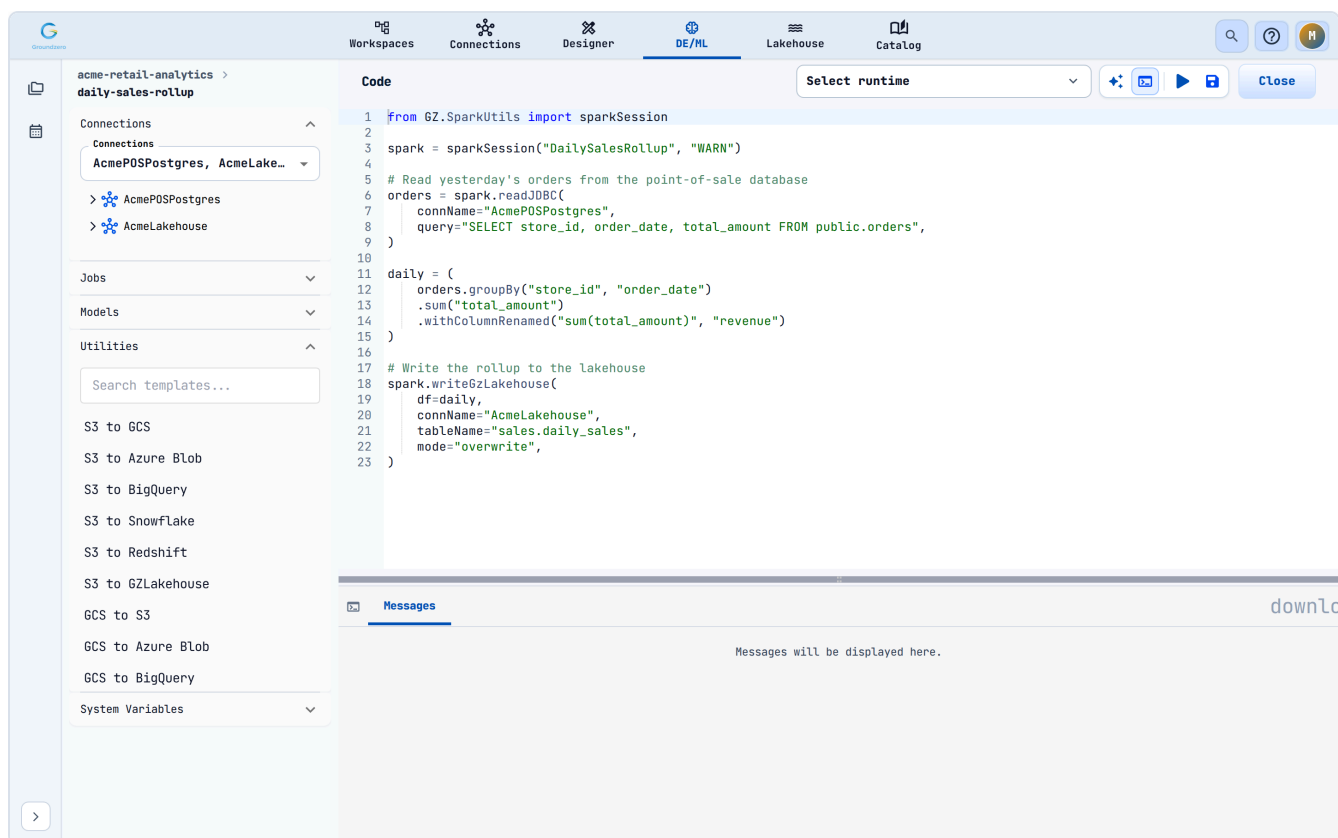


Figure 13. Utilities: search the templates, then copy one into your code.

3.4 Write job code

Every job starts by creating a session, then reads, transforms and writes through the connection names you ticked.

3.4.1 A Spark job

```
from GZ.SparkUtils import sparkSession

spark = sparkSession("DailySalesRollup", "WARN")

orders = spark.readJDBC(
    connName="AcmePOSPostgres",
    query="SELECT store_id, order_date, total_amount FROM public.orders",
)

daily = (
    orders.groupBy("store_id", "order_date")
    .sum("total_amount")
    .withColumnRenamed("sum(total_amount)", "revenue")
)

spark.writeGzLakehouse(
    df=daily,
    connName="AcmeLakehouse",
    tableName="sales.daily_sales",
    mode="overwrite",
)
```

`sparkSession(appName, logLevel="WARN")` returns a Spark session with extra methods that read from and write to your connections. Give it a meaningful application name.

3.4.2 A DuckDB job

```
from GZ.DuckDBUtils import duckSession

session = duckSession()
sales = session.readGzLakehouse(
    connName="AcmeLakehouse", tableName="sales.daily_sales"
)
session.register("sales", sales)
top = session.query("SELECT store_id, SUM(revenue) AS r FROM sales GROUP BY 1")
result_df = top.df()
```

3.4.3 Read and write other systems

Each connection type has its own read and write methods. You pass the **connection name** as it appears on the Connections page, never a password.

Connection	Read	Write or run
Amazon S3	<code>readS3(connName, fileName, format="parquet", schema=None, options=None)</code>	<code>writeS3(df, connName, filePath, format, mode, options=None, coalesce=False)</code>

Connection	Read	Write or run
JDBC databases (PostgreSQL, SQL Server, MySQL, Redshift)	<code>readJDBC(connName, query, options=None)</code>	<code>writeJDBC(df, connName, tableName, mode, options=None) ; executeJDBC(connName, query)</code>
Snowflake	<code>executeSnowflake(connName, query, options=None)</code>	<code>writeSnowflake(df, connName, tableName, mode="append", extraOptions=None) ; executeSnowflakeStatement(connName, statement)</code>
Google BigQuery	<code>executeBigQuery(connName, tableName, query=None)</code>	<code>writeBigQuery(df, connName, tableName, writeMethod, mode)</code>
Kafka	<code>readKafka(...)</code>	<code>writeKafka(...)</code>
REST	<code>readREST(...)</code>	<code>writeREST(...)</code>
GZ Lakehouse	<code>executeQuery(connName, sqlQuery, siteName=None)</code>	<code>writeGzLakehouse(df, connName, tableName, database=None, mode="overwrite", partitionBy=None, options=None, ...)</code>

Common DuckDB session methods are `readGzLakehouse(connName, tableName)`, `readS3File(connName, bucketName, filePath, fileType)`, `register(name, relation)` and `query(sql)`. Call `.df()` on a result to get a pandas DataFrame.

Not sure of the exact syntax? Open **Utilities** in the left drawer. It has templates for moving data between S3, GCS, Azure Blob, BigQuery, Snowflake, Redshift and GZ Lakehouse, such as *S3 to GZLakehouse* and *Snowflake to S3*. Replace each `{{PLACEHOLDER}}` value and any optional parameters in the comments. You can also ask the code assistant, or your MCP client, to look up the syntax for a connection type.

NOTE

Template placeholders such as `{{S3_CONNECTION_NAME}}` are for you to edit before you save. They are not system variables.

3.4.4 Use system variables

System variables are placeholders in the form `${NAME}`. Groundzero fills them in just before your code runs.

Placeholder	Value
<code>\${STRTTM}</code>	Session start time
<code>\${ENDTM}</code>	Session completion time (set when the code reaches that line)
<code>\${SITENM}</code>	Site name
<code>\${PRJNM}</code>	Project name
<code>\${JOBNM}</code>	Job name
<code>\${CLNTNM}</code>	Client name
<code>\${RTE}</code>	Runtime name
<code>\${RGN}</code>	Region

Placeholder	Value
<code>\${USR}</code>	User who started the run
<code>\${USRGRPS}</code>	The user's groups

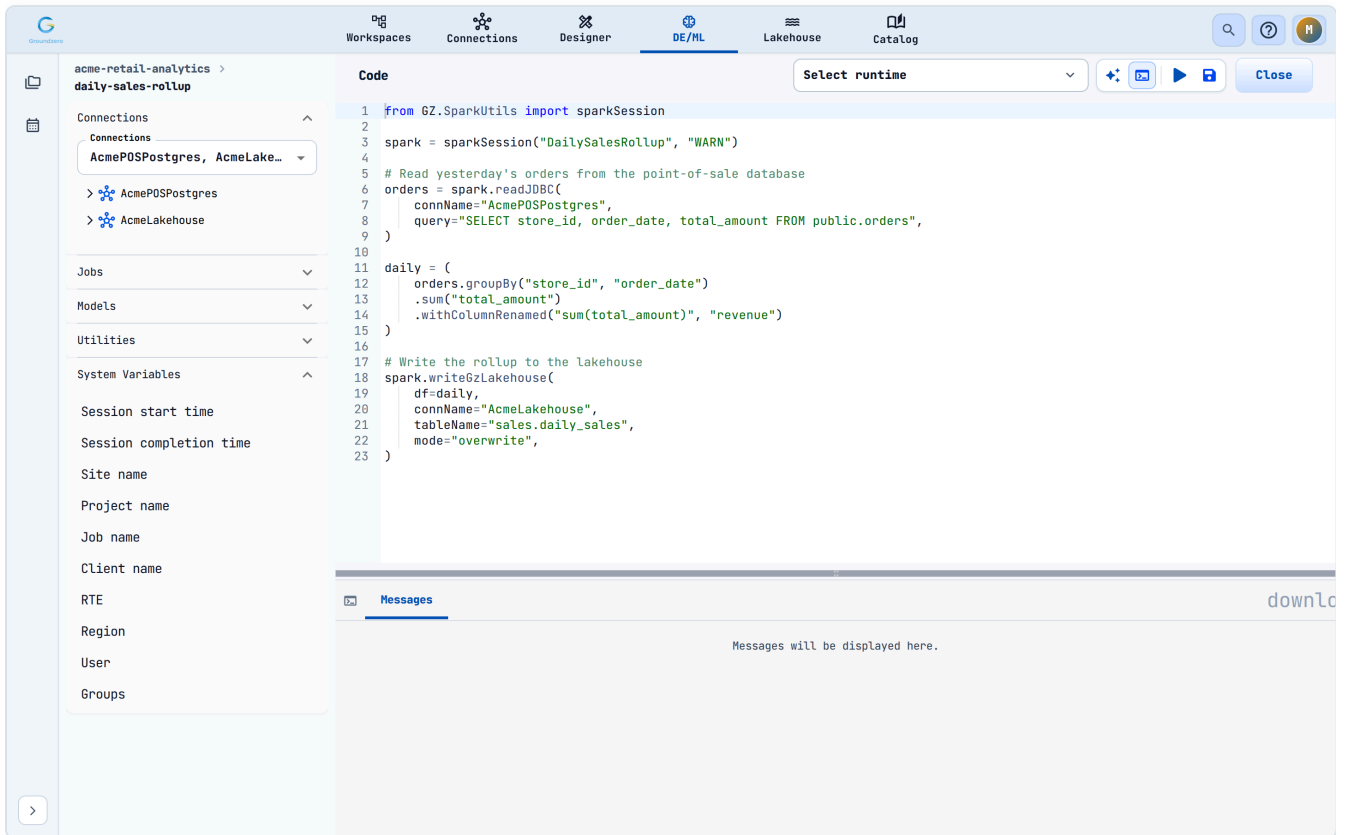


Figure 14. System Variables in the left drawer.

WARNING

Each placeholder is replaced by text that already has single quotes around the value, so you can drop it straight into a SQL statement. For example `${USR}` becomes `'maya.chen'`. If you need the bare value, remove the quotes in your code.

3.4.5 What your code may not do

Job code runs in a protected space on the Spark and DuckDB runtimes. This keeps credentials safe.

You cannot	What you see	Do this instead
Import network and system modules such as <code>socket</code> , <code>urllib</code> , <code>http</code> , <code>ssl</code> , <code>smtplib</code> , <code>asyncio</code> , <code>threading</code> , <code>multiprocessing</code> , <code>pickle</code> , <code>shutil</code> , <code>pathlib</code> , <code>glob</code> or <code>sqlite3</code>	<i>Import of 'name' is not allowed for security reasons</i>	Reach outside systems through a connection (REST, S3, a database)
Read secret environment variables	The values are hidden	Keep secrets in a connection
Read or write files outside <code>/tmp</code> , <code>/home/workspace</code> and <code>/home</code>	<i>Access to 'path' is restricted</i>	Write to <code>/tmp</code> or to a connection

You cannot	What you see	Do this instead
Call <code>os.system</code> -style functions or <code>sys.exit()</code>	<i>... is not allowed for security reasons</i>	Let the script end normally

3.5 Run jobs together

3.5.1 Run other jobs along with this one

1. Open **Jobs** in the left drawer.
2. Pick other jobs from the same project.
3. Select **Run**.

Groundzero runs the picked sub-jobs together with the main job and gives each its own saved connections. A run can include up to 255 sub-jobs, and each job can appear once. This needs a Spark, DuckDB or pandas runtime. Otherwise the run fails with *Invalid RTE for orchestration*.

3.5.2 Load models

Open **Models** in the left drawer and pick registered models. They are loaded before your code starts. See Models and MLflow.

3.6 Read the log and stop a run

The **Messages** pane shows lines about once a second while the job is logging. The download icon above the pane saves the log lines.

You see	What it means	What to do
<i>Job completed successfully</i>	The run finished with status Success	Nothing
<i>Job failed. Check the logs for the cause.</i>	The run failed	Read the last log lines
<i>Job was aborted / Job aborted successfully</i>	The run was stopped	Run again if you want it to finish
<i>Reconnecting...</i>	A log request failed. Groundzero is trying again	Wait
<i>Log streaming stopped after 2 minutes without a response. The job may still be running.</i>	The retries did not succeed	Select Resume log streaming
<i>No new logs for 15 minutes. The job may still be running.</i>	The run is quiet	Select Resume log streaming
<i>The run could not be found.</i>	Groundzero has no record of the run	Run the job again
<i>Reconnected to the running job</i>	You reloaded the page and Groundzero picked the run up again	Nothing

To stop a run, select the red **Stop** icon. *Job aborted successfully* confirms it. If a run cannot be stopped, a message gives the reason.

TIP

If you close the browser while a job runs, open the job again. Groundzero picks up the run and carries on showing the log.

3.6.1 If the run does not start

Message	Cause	Fix
<i>please select Compute</i>	No runtime and compute chosen	Choose both in the runtime picker
<i>please select connection</i>	No connection ticked	Tick a connection in the left drawer
<i>Failed to save job</i>	The save did not work, so the run did not start	Try again
<i>Failed to run job</i>	The run could not be started	Try again. If it keeps failing, contact an administrator
<i>Invalid parameters</i>	The run request was incomplete, for example no compute	Choose a runtime and compute
<i>Invalid RTE.</i>	The chosen runtime cannot run jobs	Choose a Spark or DuckDB runtime

3.7 Share a job

Published jobs form a shared library for your site.

3.7.1 Publish a job

1. In the Jobs list, open the row menu and select **Export**.
2. If no published job has that name, you see *name exported successfully*.
3. If one exists, the **Override Export** dialog asks whether to replace it. Choose **Override** to replace it, or cancel.

3.7.2 Import a job

1. On the Jobs tab, select **Import Job**.
2. Choose a published job in **Select a job** and select **Import**.
3. If your project already has a job with that name, the **Job Name Conflict** dialog offers **Replace**, **Keep Both** (adds a timestamp to the new name) and **Cancel**.

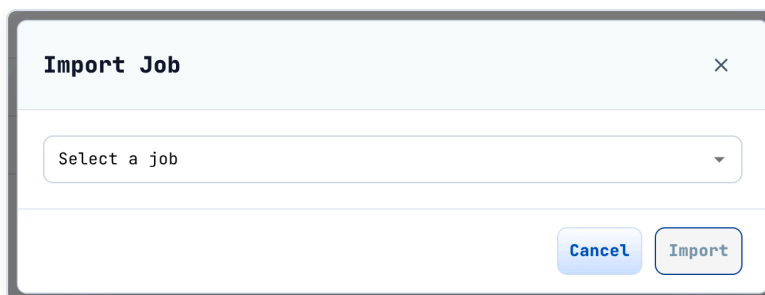


Figure 15. Import Job lists the jobs published on your site.

Administrators can review and delete published jobs, and import them across sites, in the admin console.

3.8 Troubleshooting

Symptom	Cause	Fix
The log table stays empty after Run	The runtime is still starting	Wait. The pane reads <i>Running job...</i> until lines arrive
<i>Import of 'x' is not allowed for security reasons</i>	The code imports a blocked module	Use a connection instead
The job cannot see a table	The connection is not ticked, or its tables have not loaded yet	Tick the connection and wait for its tables to appear
A run says Success but a table is missing	The write used <code>mode="append"</code> or a different <code>tableName</code> than you checked	Check <code>tableName</code> and <code>mode</code>

4 Scheduling and run history

The **Scheduler** runs a job at the time you choose, once or on a repeating schedule. Each scheduled job is a **task**. After this chapter you can schedule a job to run every morning, check how its runs went, and get an email when a run succeeds or fails.

Open it from **DE/ML**, then the calendar icon in the left menu (**Scheduler**).

Project Name	Schedule Name	Job Name	Start	End	Status	Actions
▼ acme-retail-analytics (2)						
	● daily-sales-rollup-0600	daily-sales-rollup	2026-09-20 00:22:36		Success	...
	● customer-dedupe-weekly	customer-dedupe	2026-09-12 00:22:36	2026-09-30 00:22:36	Failed	...
▼ store-inventory-sync (1)						
	● inventory-snapshot-hourly	inventory-snapshot-to-lakehouse	2026-09-29 00:22:36		Running	...
▼ finance-reporting (1)						
	● returns-quality-nightly	returns-quality-check	2026-09-26 00:22:36		Success	...

Figure 16. Tasks are grouped by project. A green dot beside a name means the task is active. A red dot means it is inactive.

Before you start, you need a saved job with working code. Run it by hand once first. You also need permission to run ETL jobs and to use compute (see [Runtimes and compute](#)).

4.1 Schedule a job

1. Select **New Task**.
2. Choose a **Select Project**, then a **Select Job**. The job list shows that project's jobs.
3. Select the **RTE** box. Choose a **Runtime** and **Compute** size, then select **Save** in the panel.
4. Choose a schedule type: **Streaming** (runs once) or **Batch** (repeats).
5. Type a **Name**. Use letters, digits, `.`, `-` and `_`. Spaces are rejected.
6. Add a **Description** if you like.
7. Pick the **Start date** and **Start time**.
8. For Batch, set **Recurrence** and **Ends**.
9. Select **Save**.

New Task

Select Project* ▼ Select Job* ▼

RTE* ● Streaming ○ Batch

Name* Description

Start date 10/02/2026 📅 Start time 12:22 AM 🕒

Cancel Save

Figure 17. The New Task form with the default Streaming type: one date and time.

What happens next: You see *Job scheduled successfully* and the task appears in the list. Its status shows the latest run: **Running**, **Success** or **Failed**.

How to check it worked: From the task's row menu choose **Run Once**, wait a couple of minutes, then choose **View Logs**.

4.1.1 Choose the schedule type

Type	What happens
Streaming (default)	The task runs once, at the start date and time you pick
Batch	The task repeats. You set how often, and when it stops

New Task

Select Project* acme-retail-analytics ▼ Select Job* daily-sales-rollup ▼

RTE* gz-rte-spark-sklearn:1.0.0 ○ Streaming ● Batch

Name* daily-sales-rollup-0600 Description

Start date 10/02/2026 📅 Start time 12:23 AM 🕒

RECURRENCE

Repeat Minutes ▼

Every 1 minute(s)

Run every 1 minute(s)

Ends ● Never ○ On

Cancel Save

Figure 18. Batch adds a Recurrence panel with Repeat and Ends.

4.1.2 Set how often it repeats

Repeat	What you set
Minutes	Every N minutes (0 to 59)

Repeat	What you set
Hourly	Every N hours (0 to 23)
Daily	At the start time, every day
Weekly	At the start time, on the weekdays you click
Monthly	The 1st of every month at the start time. Select Custom to choose specific dates
Yearly	Every N years (1 to 24)

Ends has two choices. **Never** keeps the task going. **On** shows an **End date** and **End time**. If the end is not after the start, the form says *End date and time must be after the start date and time*.

WARNING

Schedules follow UTC. Groundzero converts the start time you pick from your own time zone, so the repeating pattern follows UTC. A daily task started at 06:00 in India runs at 00:30 UTC every day.

4.2 Find your way around the Scheduler

Element	What it does
Select Project	Tick one or more projects to show only their tasks
Refresh control	Shows <i>Refreshed Ns ago</i> . The page refreshes itself. Select the circular arrow to refresh now
New Task	Opens the task form
Project groups	Select a project row to expand or collapse it. The number in brackets is its task count

Column	Meaning
Schedule Name	The task name, with its active or inactive dot
Job Name	The job that runs
Start	The first run time, in your local time zone
End	The stop time, if there is one
Status	The latest run: Running , Success or Failed
Actions	The row menu

4.3 Change, run or stop a task

Open the **Actions** menu on the task row.

The screenshot shows the Groundzero Scheduler interface. At the top, there's a navigation bar with tabs: Workspaces, Connections, Designer, DE/ML (selected), Lakehouse, and Catalog. Below the navigation bar, there's a sidebar with a 'New Task' button. The main area is titled 'Scheduler' and contains a table with columns: Project Name, Schedule Name, Job Name, Start, End, Status, and Actions. The table lists several tasks under the project 'acme-retail-analytics'. One task, 'customer-dedupe-weekly', is in a 'Failed' status. An actions menu is open for this task, showing options: View Details, Edit, Delete, Run Once, Abort (greyed out), Stop Schedule, Last 10 runs, View Logs, and Notify Email.

Project Name	Schedule Name	Job Name	Start	End	Status	Actions
acme-retail-analytics (2)						
	daily-sales-rollup-0600	daily-sales-rollup	2026-09-20 00:22:56		Success	...
	customer-dedupe-weekly	customer-dedupe	2026-09-12 00:22:56	2026-09-30 00:22:56	Failed	View Details, Edit, Delete, Run Once, Abort, Stop Schedule, Last 10 runs, View Logs, Notify Email
store-inventory-sync (1)						
finance-reporting (1)						

Figure 19. The row menu. Abort is greyed out because this task is not running.

I want to...	Choose	You see
Read how the task is set up	View Details	A summary: name, project, job, type, description, runtime, compute, start and end, and who last changed it
Change the schedule	Edit	The form with the current values. <i>name is successfully rescheduled</i>
Remove the task	Delete	<i>Are you sure you want to delete?</i> then <i>name is successfully deleted</i>
Run once more, now	Run Once	An extra run about two minutes from now with the same job, runtime and compute. <i>name executed successfully</i>
Stop a run in progress	Abort	<i>Successfully aborted.</i> Enabled only while the status is Running
Stop future runs	Stop Schedule	<i>name stopped successfully</i>
See recent runs	Last 10 runs	The run history page
Read the latest log	View Logs	A log dialog
Get emails about runs	Notify Email	The notification form

While a run is in progress, every action except **Abort** and **Notify Email** is greyed out.

Task Details ×

SCHEDULED TASK

daily-sales-rollup-0600

Batch acme-retail-analytics

TASK

Name	Project	Job	Type
daily-sales-rollup-0600	acme-retail-analytics	daily-sales-rollup	Batch

Description

Roll up POS orders each morning

RUNTIME

Runtime Image

gz-rte-spark-sklearn:1.0.0

Compute

Medium, 4 vCPU vCPU, 8 GB Memory

SCHEDULE

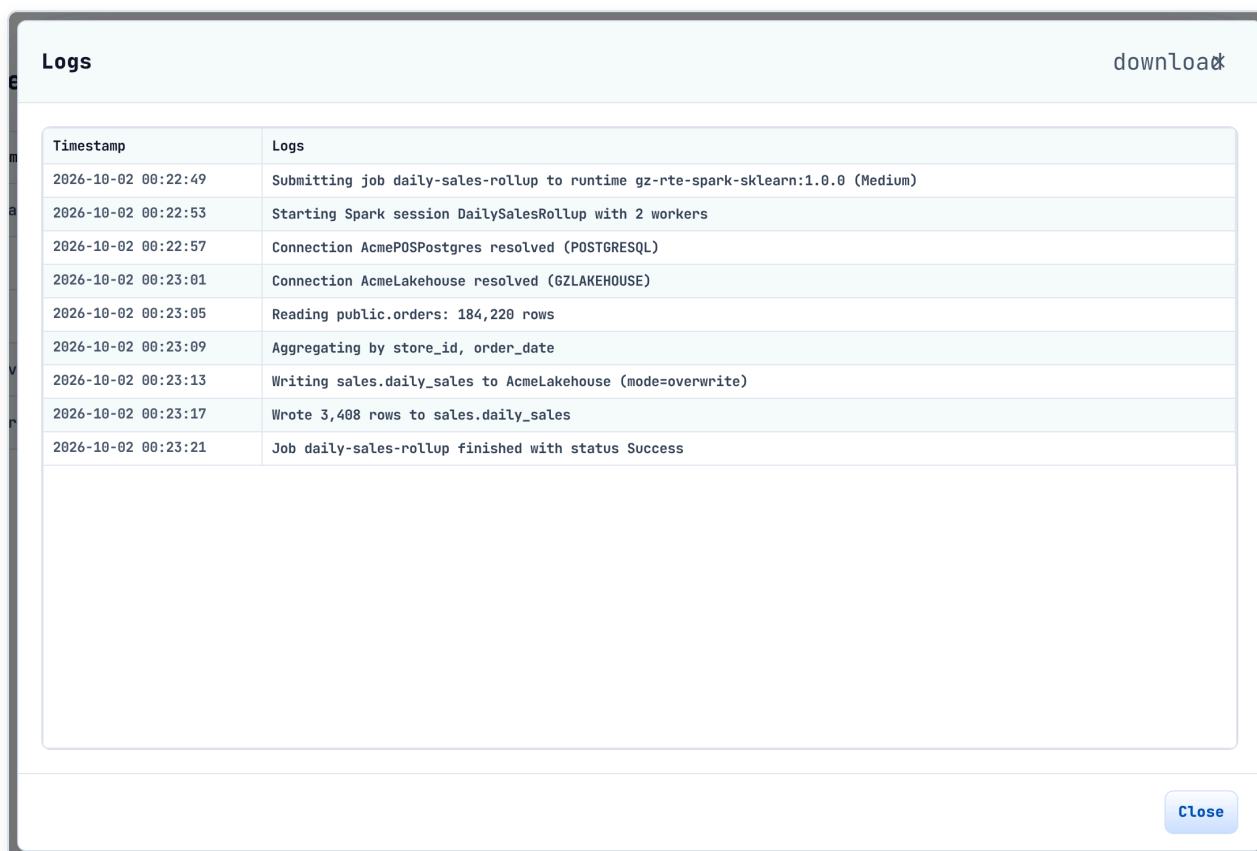
Start Time	End Time	Last Modified On	Last Modified By
2026-09-20 00:23:48	Never	2026-09-20 00:23:48	maya.chen

Figure 20. View Details shows how the task is set up, in three sections: Task, Runtime and Schedule.

4.4 Check how runs went

4.4.1 Read the log of a run

Select **View Logs**. A dialog shows a table with **Timestamp** and **Logs** columns and a download button.



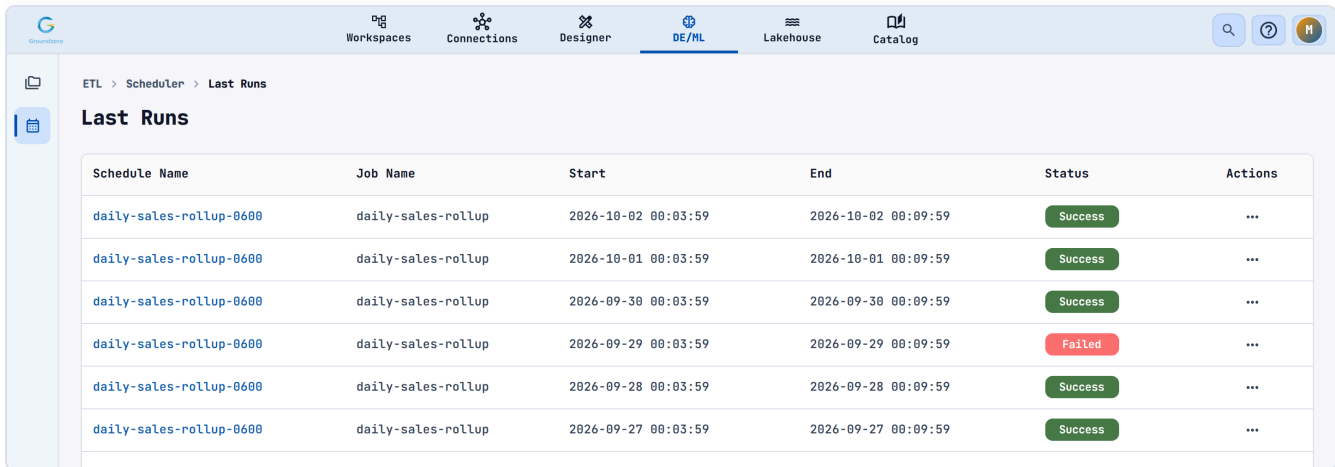
Timestamp	Logs
2026-10-02 00:22:49	Submitting job daily-sales-rollup to runtime gz-rte-spark-sklearn:1.0.0 (Medium)
2026-10-02 00:22:53	Starting Spark session DailySalesRollup with 2 workers
2026-10-02 00:22:57	Connection AcmePOSPostgres resolved (POSTGRESQL)
2026-10-02 00:23:01	Connection AcmeLakehouse resolved (GZLAKEHOUSE)
2026-10-02 00:23:05	Reading public.orders: 184,220 rows
2026-10-02 00:23:09	Aggregating by store_id, order_date
2026-10-02 00:23:13	Writing sales.daily_sales to AcmeLakehouse (mode=overwrite)
2026-10-02 00:23:17	Wrote 3,408 rows to sales.daily_sales
2026-10-02 00:23:21	Job daily-sales-rollup finished with status Success

Figure 21. Log lines for the latest run of a task.

Message	Meaning	What to do
No logs found	The run has not produced readable log lines yet	Try again in a moment
Execution Id not found	The task has not run yet	Use Run Once , or wait for the first scheduled run
Error in viewing logs	The log could not be shown	Try again, or download the log

4.4.2 See the last runs

Select **Last 10 runs**. The page lists up to ten recent runs with the task name, job name, start, end and status. Each row has **View Logs**.



Schedule Name	Job Name	Start	End	Status	Actions
daily-sales-rollup-0600	daily-sales-rollup	2026-10-02 00:03:59	2026-10-02 00:09:59	Success	...
daily-sales-rollup-0600	daily-sales-rollup	2026-10-01 00:03:59	2026-10-01 00:09:59	Success	...
daily-sales-rollup-0600	daily-sales-rollup	2026-09-30 00:03:59	2026-09-30 00:09:59	Success	...
daily-sales-rollup-0600	daily-sales-rollup	2026-09-29 00:03:59	2026-09-29 00:09:59	Failed	...
daily-sales-rollup-0600	daily-sales-rollup	2026-09-28 00:03:59	2026-09-28 00:09:59	Success	...
daily-sales-rollup-0600	daily-sales-rollup	2026-09-27 00:03:59	2026-09-27 00:09:59	Success	...

Figure 22. Run history for one task. A Failed row can be opened to read why.

4.4.3 Run a failed job again

Use **Run Once**, or fix the job and wait for the next scheduled time.

4.5 Get an email when a run succeeds or fails

1. Open the task's row menu and select **Notify Email**.
2. Choose the **Success** or **Failure** tab.
3. Fill in **To**, **From**, **Subject** and **Message**. Use the **Variables** tab to find placeholders for the subject and message. Select one to copy it (*Placeholder copied*).
4. Select **Save**.

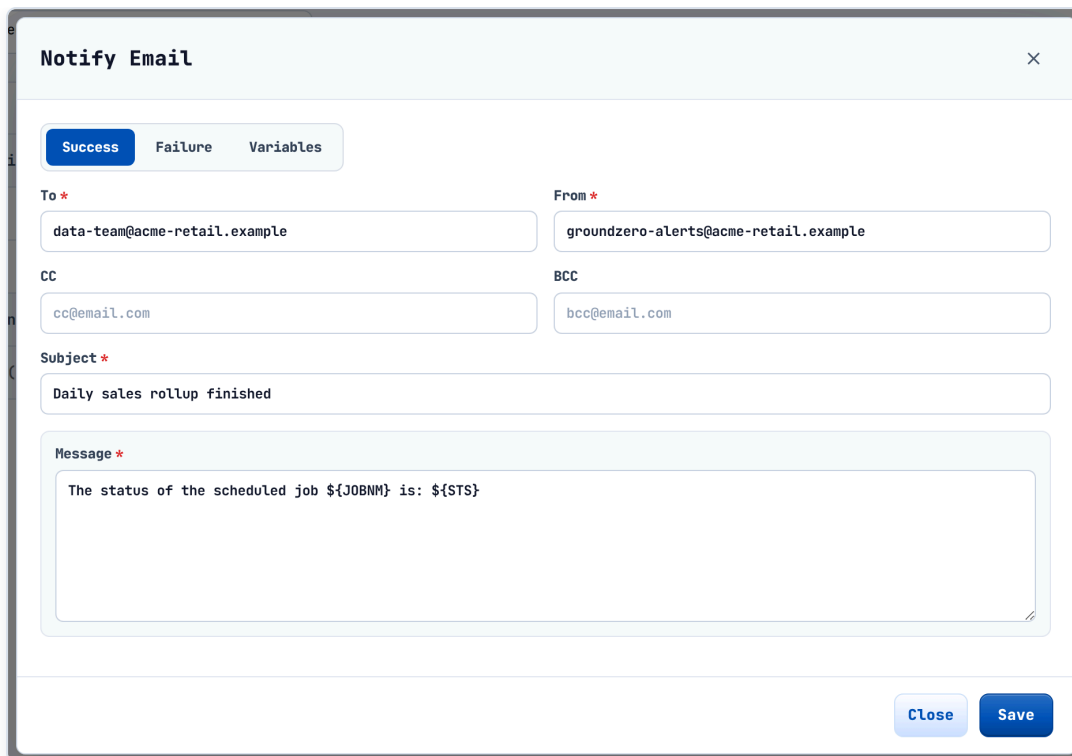


Figure 23. The Success tab. Required fields are marked with a red star.

Field	Required	Notes
To	Yes	A valid email address
From	Yes	A valid email address
CC, BCC	No	Shown in the form. Groundzero sends To, From, Subject and Message
Subject	Yes	Text. May include variables
Message	Yes	Text. May include variables

Save stays greyed out until To, From, Subject and Message are filled in. If a setting already exists, the dialog opens with it filled in, and **Save** updates it.

Placeholder	Value
{STRTTM}	Session start time
{ENDTM}	Session completion time
{SITENM}	Site name
{PRJNM}	Project name
{JOBNM}	Job name
{CLNTNM}	Client name
{STS}	Session status
{ERR}	Error message

Placeholder	Value
{RTE}	Runtime
{RGN}	Region
{USR}	User
{USRGRPS}	Groups

NOTE

Emails are sent through the email settings your administrators manage in the admin console under **Email**. See the Analyze, AI and Admin guide.

4.6 Troubleshooting

Symptom	Cause	Fix
<i>Name cannot contain space</i>	The task name has a space	Remove the spaces
<i>Error in scheduling name</i>	Groundzero rejected the schedule	Check the runtime, compute and dates, then try again
A task shows Failed	The job code failed	Open View Logs or the run history and read the last lines
Abort is greyed out	The task is not running	Abort works only while the status is Running
Save is greyed out in New Task	A required field is empty	Fill in project, job, runtime, compute and name
The task runs at an unexpected hour	Schedules follow UTC	Convert your start time to UTC

5 Notebooks

A **notebook** is an interactive document made of cells. You run a code cell and its output appears right under it, and Markdown cells hold your notes. After this chapter you can start a notebook session, run code, keep your work safe with checkpoints, and share a notebook.

Use notebooks to explore data, try a transformation or train a model before you turn the result into a job. Groundzero's notebook editor follows Jupyter habits: command and edit modes, the same keyboard shortcuts, `.ipynb` import and export, and checkpoints. Each notebook runs in its own **session** that you start and stop yourself.

Notebooks live in a project, on the **Notebooks** tab.

Use a notebook when	Use something else when
You want to explore, chart or try ideas interactively	The work should run without you. Use an ETL job
You are training or testing a machine learning model	You need a governed table for dashboards. Use a dataset

Before you start, you need a project, a connection to work with, and a runtime and compute size. Notebooks never run on Standard.

5.1 Create a notebook

1. Open a project and select the **Notebooks** tab.
2. Select **New Notebook**.
3. Type a **Notebook Name** that follows the name rule.
4. Select **Add**.

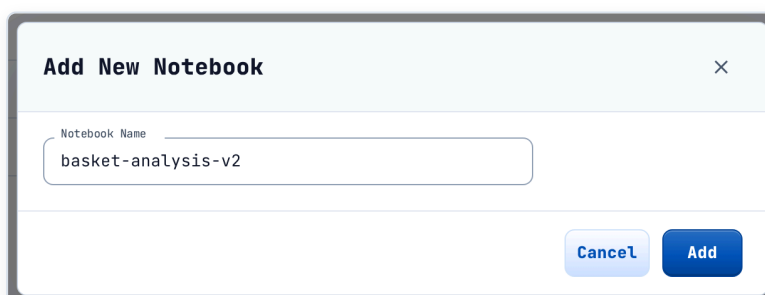
A screenshot of the 'Add New Notebook' dialog box. The dialog has a title bar with the text 'Add New Notebook' and a close button (X). Below the title bar is a text input field labeled 'Notebook Name' containing the text 'basket-analysis-v2'. At the bottom right of the dialog are two buttons: 'Cancel' and 'Add'.

Figure 24. The Add New Notebook dialog.

What happens next: You see *Notebook created successfully* and the notebook joins the list.

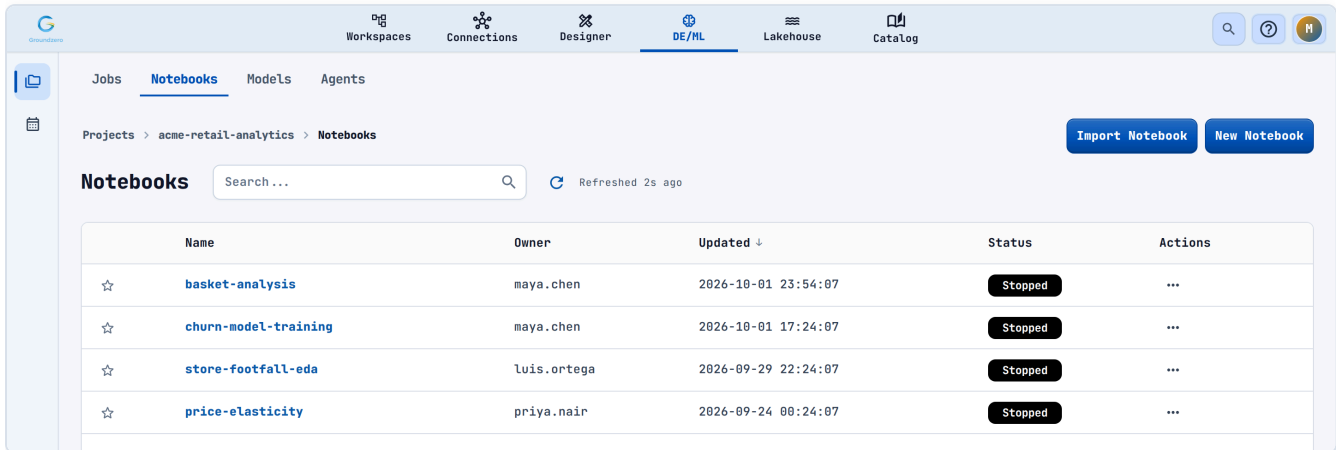


Figure 25. The Notebooks tab lists notebooks with their owner, last update and status.

Column	Meaning
Star	Marks a favourite. It is kept in your browser only, so it does not follow you to another browser
Name	Select it to open the notebook
Owner	The user who last changed the notebook
Updated	The time of the last change, in your local time
Status	Stopped or Running . The list updates by itself
Actions	Delete , Export , Rename and, while running, Stop Notebook

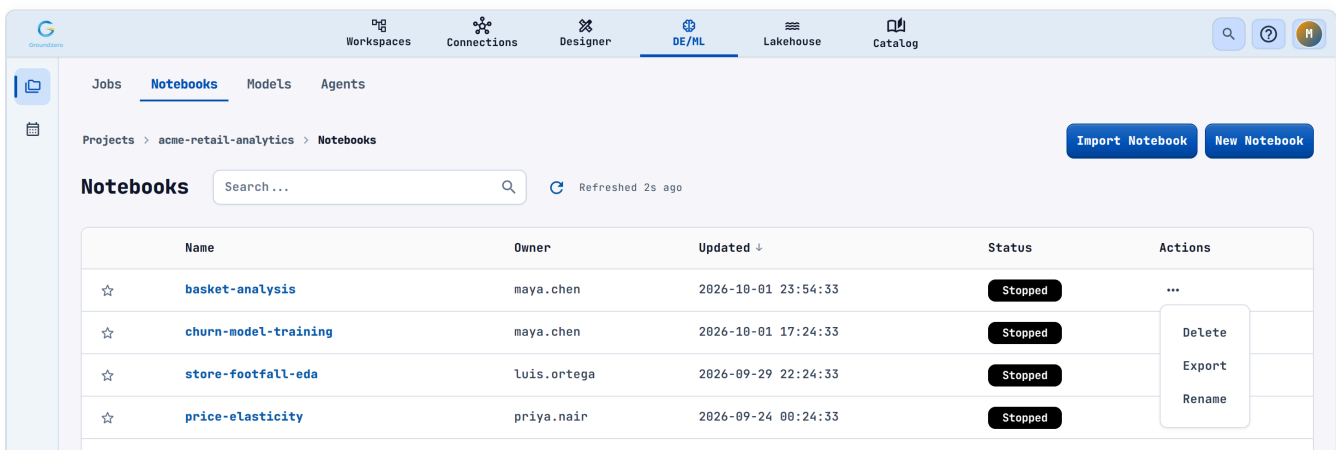


Figure 26. Row menu of a stopped notebook.

Action	What it does
Rename	Uses the same name rule. You see <i>Notebook renamed successfully</i>
Delete	Asks you to confirm. Greyed out while the notebook runs. You see <i>Notebook deleted successfully</i>
Export	Publishes a copy. See Share a notebook
Stop Notebook	Appears while running. Confirm in the dialog. You see <i>Notebook stopped successfully</i>

5.2 Start a session

By the end of this task your notebook is running and ready for code.

1. Select the notebook name.
2. In the left drawer, open **Connections** and tick the connections you need. Their databases and tables load as a tree.
3. In the header, open the runtime picker and choose a **Runtime** and **Compute**.
4. Select **Start session**.

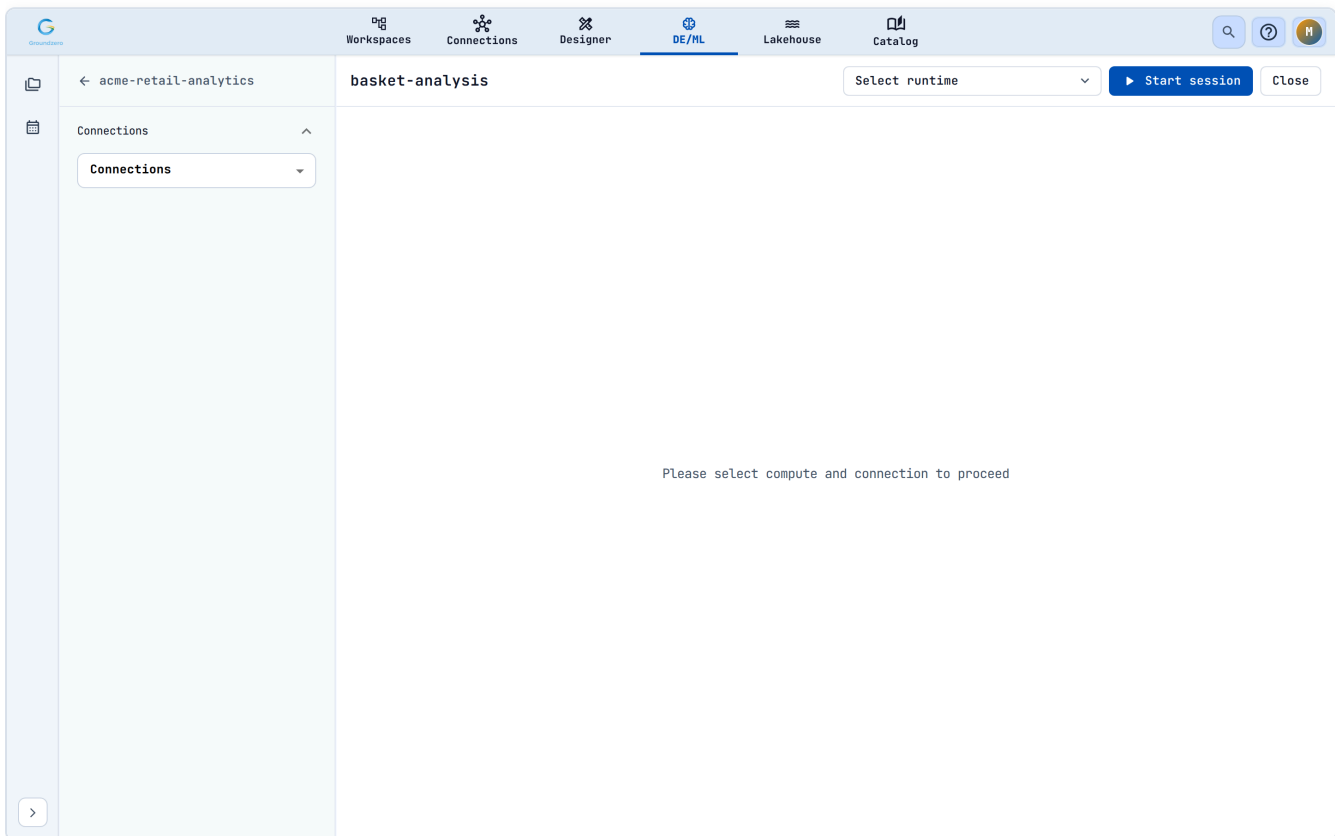


Figure 27. Before a session starts, the editor shows a prompt. It tells you what is missing: connection, compute or both.

What happens next: The page shows *Opening notebook session...*, *Starting notebook runtime...*, *Preparing notebook runtime...* and *Opening notebook workspace...*. This usually takes a minute or two. Then the status bar shows the session and kernel (the Python process that runs your cells).

Message	What it means and what to do
<i>Please select connection and Compute</i>	You started without both. Choose them
<i>Please select compute and connection to proceed</i>	You have not chosen yet. Choose both
<i>Please click on start to start the notebook</i>	Everything is set. Select Start session
<i>Notebook is already running, connecting to existing session</i>	You or a colleague already started it. Groundzero reconnects you
<i>Notebook session could not be started</i>	The runtime did not become ready. Read the message, then try again or choose a smaller or different runtime
<i>Notebook runtime did not become ready</i>	The runtime took too long. Try again

NOTE

The runtime picker is locked while a session runs, with the hint *Stop the session to change compute*. Stop the session, change the runtime and start again.

5.2.1 Stop a session

Select **Stop session** in the header, or **Stop Notebook** in the list. The page reads *Notebook session stopped; start the session when you are ready to continue*. Stopping ends the kernel, so variables in memory are lost. The saved notebook stays.

WARNING

A running session keeps using compute until you stop it. Stop sessions you no longer need.

5.3 Run code and read the output

1. Click a cell and type code.
2. Press Shift+Enter, or select **Run**.
3. Read the output under the cell.

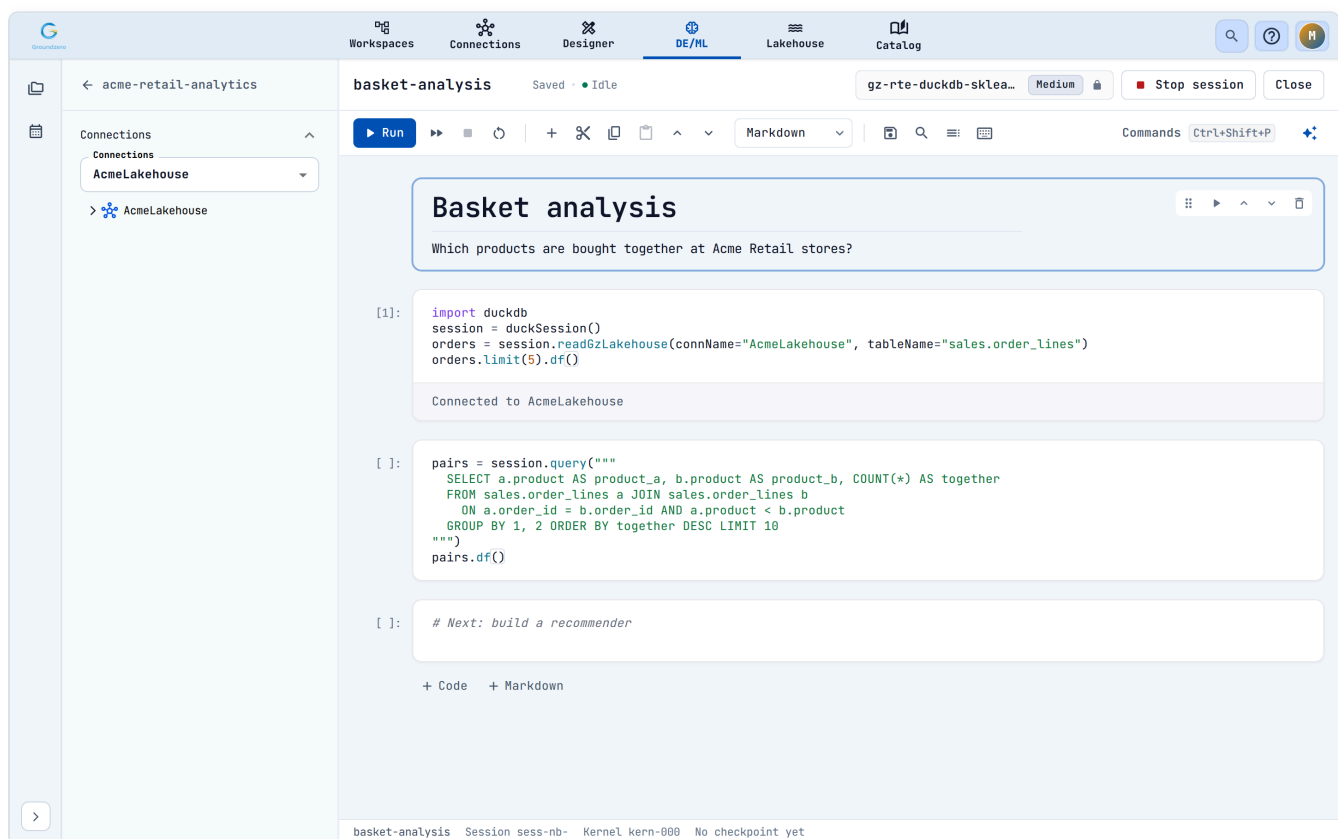


Figure 28. A running notebook. The status bar at the bottom shows the session, the kernel and the latest checkpoint.

How to check it worked: The kernel state in the header changes from Busy to Idle, and the output appears under the cell.

Code cells run one at a time, in the order you queued them. If a cell fails, or you choose **Stop after current cell**, the rest of the queue is dropped, and a message says *Run stopped at cell N* or *Run stopped after cell N*.

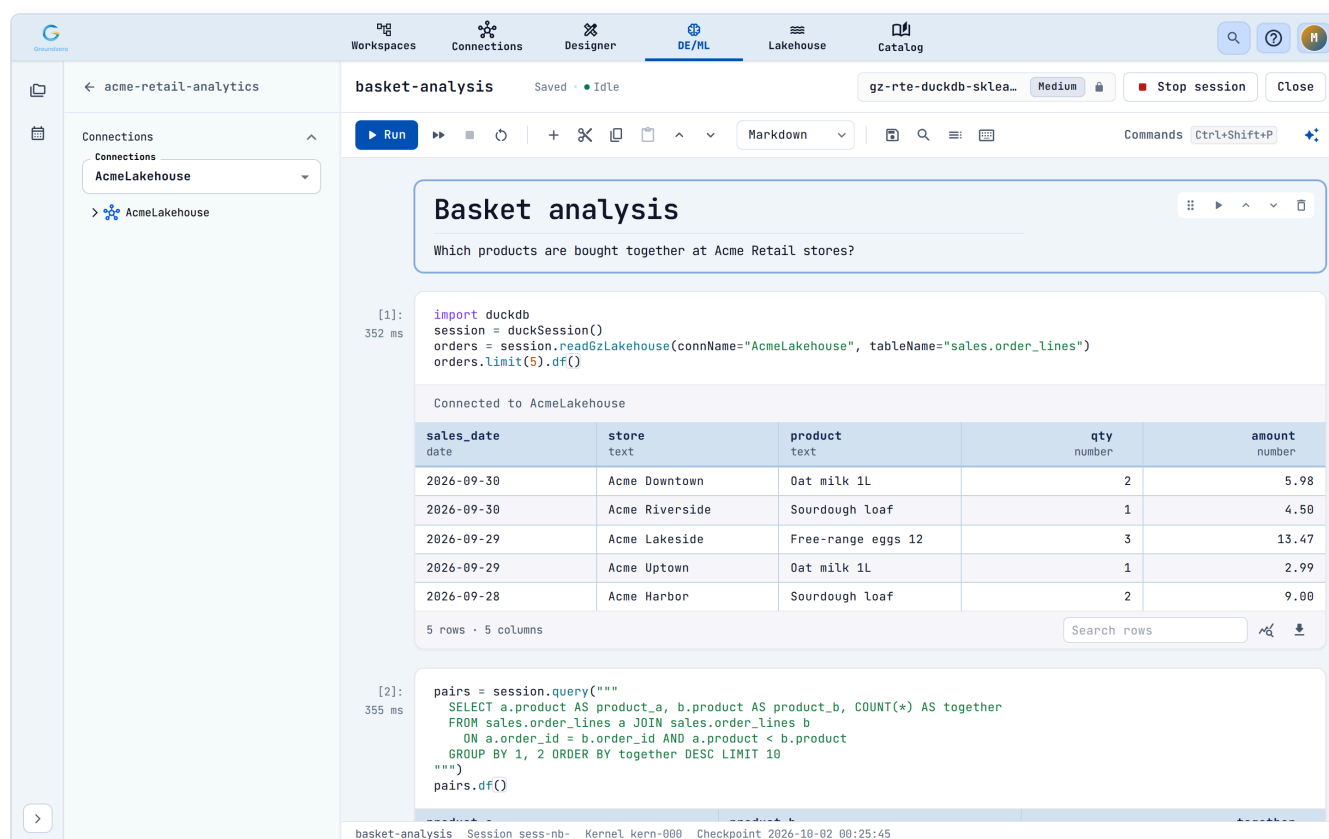


Figure 29. After Run all, outputs appear under each cell with timing and a table preview.

I want to...	Do this
Run the selected cells	Ctrl+Enter
Run and select the cell below	Shift+Enter
Run and insert a cell below	Alt+Enter
Run every code cell	Run all cells in the toolbar
Run all above, or the selected cell and below	Run menu in the command palette
Stop after the current cell	Run menu in the command palette
Stop a cell that is running	Interrupt kernel (press I twice)

Run all makes a silent checkpoint first, so you can go back.

5.3.1 What the output can show

- Text and error output, with colours.
- Markdown and HTML (cleaned for safety), images and JSON trees.
- A data grid under the cell when its last expression is a pandas, Spark or DuckDB frame. The grid shows up to **50,000 rows**. `df.head(5)`, `df.limit(100)`, `df.show(20)` and `df.take(10)` use their own row counts.

The grid footer has the row count, a **Search rows** box, a column summary and a CSV download. You can sort and filter up to 50,000 rows, and the grid shows 500, 1,000, 2,500 or 5,000 rows per page. If the grid shows fewer rows than your table, the preview is capped, so aggregate or filter in code.

TIP

Add `LIMIT` to exploratory SQL. Groundzero warns you before an unbounded query: *SELECT without LIMIT — large results may be slow. Add LIMIT n.* This applies to `%%sql` cells and to SQL strings in Python such as `spark.sql` and `session.sql`.

5.4 Use your connections and tables

The left drawer shows each ticked connection as a tree of databases, tables and columns. Hover over a node for two actions.

Action	Result
Insert notebook snippet (plus icon)	Adds a snippet for that connection, table or column at the cursor
Copy name	Copies the name. You see <i>Copied to clipboard</i>

As you type, the editor suggests connection, database, table and column names, together with the kernel's own suggestions. Press Tab to complete and Shift+Tab for help on the code at the cursor.

5.5 Work with cells and modes

- **Command mode** (blue bar beside the cell): keys act on cells. Press Enter to edit a cell.
- **Edit mode**: you type in the cell. Press Esc to go back to command mode.
- **Code** cells run on the kernel. **Markdown** cells show text, maths and tables. **Raw** cells are kept as typed.
- Select several cells with Shift+Up or Shift+Down and act on them together.
- Drag a cell by its handle to reorder it.

Press **H** in command mode, or select the keyboard icon, to see every shortcut.

Command mode (press Esc)		Edit mode (press Enter)	
Keys	Action	Keys	Action
Enter	Edit the selected cell	Esc	Leave the editor (command mode)
Shift+Enter	Run cell and select below	Shift+Enter	Run cell and select below
Ctrl+Enter Meta+Enter	Run selected cells	Ctrl+Enter	Run cell
Alt+Enter	Run cell and insert below	Alt+Enter	Run cell and insert below
A	Insert cell above	Ctrl+Shift+-	Split cell at cursor
B	Insert cell below	Tab	Complete (kernel and Groundzero)
D D	Delete selected cells	Shift+Tab	Show documentation for the name at the cursor
Z	Undo cell deletion	Ctrl+/	Toggle comment
M	Change to Markdown	Ctrl+F	Find in this cell
Y	Change to code	Ctrl+Alt+ArrowUp	Add a cursor above
R	Change to raw	Ctrl+S	Save notebook
C	Copy selected cells		
X	Cut selected cells		
V	Paste cells below		
Shift+V	Paste cells above		

Figure 30. Keyboard shortcuts for command mode and edit mode.

5.5.1 The header

Element	Meaning
Notebook name	Select it to rename in place. Press Enter to save, Esc to cancel
Saved / Saving... / Unsaved changes / Save failed	Whether your work is saved. Retry save appears on failure
Kernel state	Starting, Idle, Busy, Restarting, Reconnecting, Disconnected or Kernel stopped
Runtime picker	Runtime and compute badge. Locked while the session runs
Stop session / Start session	Controls the session
Close	Returns to the project's Notebooks tab

5.5.2 The toolbar

Control	What it does
Run	Runs the selected cells and selects the one below (Shift+Enter)

Control	What it does
Run all cells	Runs every code cell from the top
Interrupt kernel	Stops the running cell (press I twice)
Restart kernel	Restarts and clears memory (press R twice)
Insert, cut, copy, paste, move up, move down	Edit cells
Cell type	Code, Markdown or Raw
Save, Find and replace, Table of contents, Keyboard shortcuts	Utilities
Commands (Ctrl+Shift+P)	Opens the command palette

5.5.3 Find and replace, and the table of contents

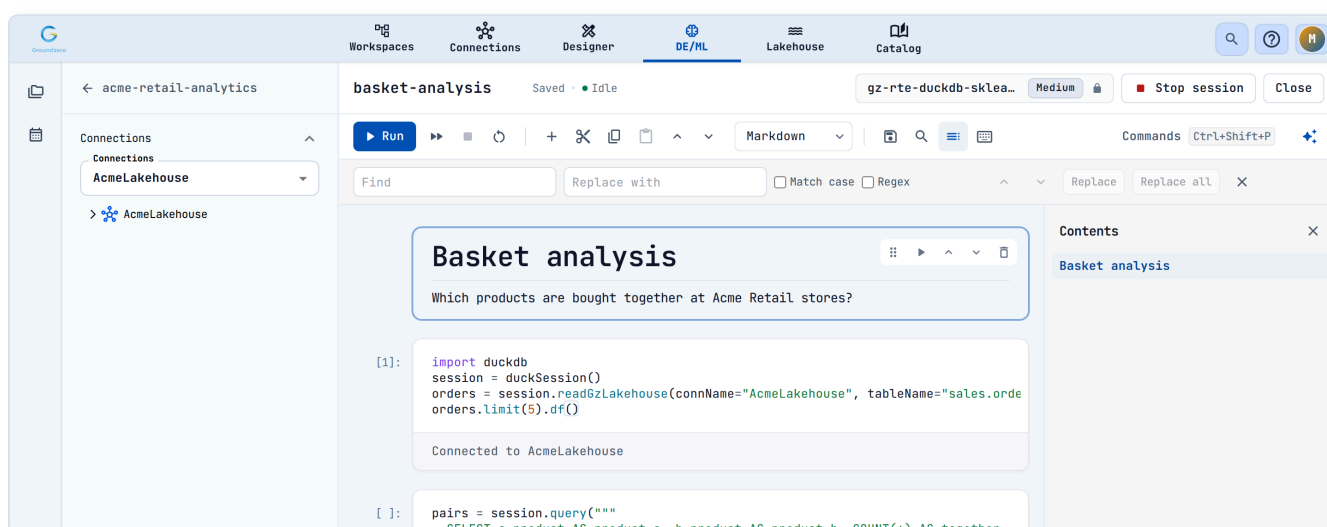


Figure 31. Find and replace bar (top) and the table of contents (right).

- **Find and replace** searches all cells. It has **Match case** and **Regex** options, and **Replace** and **Replace all**.
- **Table of contents** lists the Markdown headings and jumps to them.

5.5.4 The command palette

Press Ctrl+Shift+P to search every command.

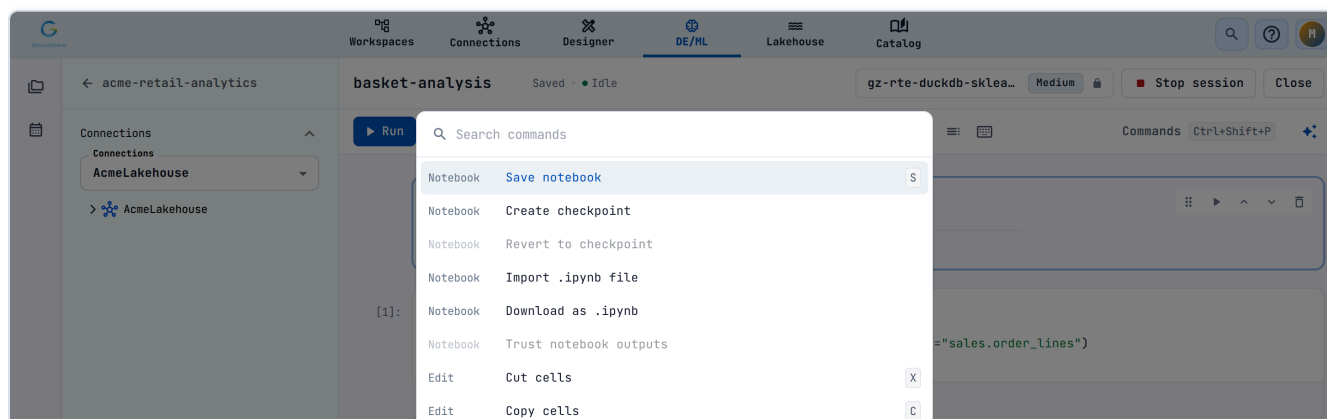


Figure 32. Ctrl+Shift+P opens a searchable list of every command, grouped as Notebook, Edit, View, Cell, Run and Kernel.

Group	Commands
Notebook	Save, Create checkpoint, Revert to checkpoint, Import .ipynb file, Download as .ipynb, Trust notebook outputs
Edit	Cut, copy, paste above or below, delete, undo delete, split, merge, move, find and replace, clear output, clear all outputs
View	Table of contents, line numbers, show or hide input or output, output scrolling, keyboard shortcuts
Cell	Insert code or Markdown cell, change type
Run	Run variants, stop after current cell
Kernel	Interrupt, restart, restart and clear outputs, restart and run all, reconnect

5.6 Keep your work safe

- **Autosave** saves the notebook 3 seconds after you stop typing. Press Ctrl+S to save at once. *Notebook saved* confirms it.
- A **checkpoint** is a saved restore point. The status bar shows the latest one, for example *Checkpoint 2026-10-02 00:10:04*, or *No checkpoint yet*.
- Choose **Create checkpoint** to make one, and **Revert to checkpoint** to go back to the latest.

5.6.1 Confirmations you may see

Command	What the dialog says
Restart kernel	<i>Every variable in memory is lost. Cell outputs stay in the notebook.</i>
Restart and clear outputs	<i>Every variable in memory and every cell output is lost.</i>
Restart and run all	<i>Every variable in memory is lost, then all code cells run from the top.</i>
Revert to checkpoint	<i>Every change made after that checkpoint is lost, including saved changes.</i>
Import a <code>.ipynb</code> file	<i>The cells of this notebook are replaced by the N cells in file...</i> Download the current notebook first

If the connection to the kernel drops, the editor tries again a few times, then shows **Disconnected**. Choose **Reconnect to kernel**. A cell that was running shows *The kernel connection was lost*. If the editor cannot open a notebook, it shows *Unable to open notebook* with the hint *Check that the notebook session is running, then try again* and a **Try again** button.

5.7 Move notebooks in and out

- **Download as .ipynb** saves the notebook to your computer.
- **Import .ipynb file** replaces the cells of the open notebook. Imported outputs stay untrusted until you choose **Trust notebook outputs**.

5.8 Share a notebook

Notebooks have no share link. Share them in one of two ways.

1. **Publish and import.** Choose **Export** on the Notebooks list to publish a copy to your site. Anyone on the site can then choose **Import Notebook** to add a copy to their own project.
2. **Permissions.** Administrators decide who can see and edit notebooks and projects.

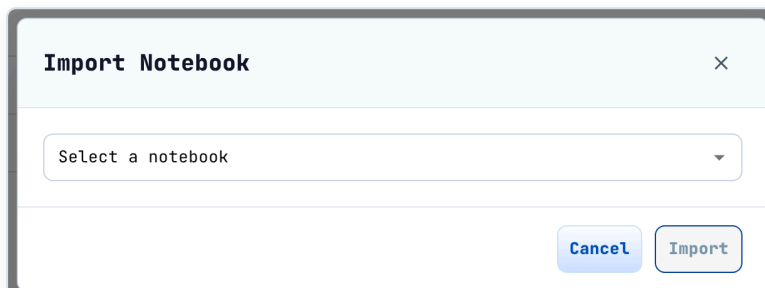


Figure 33. Import Notebook lists the notebooks published on the site.

Step	What you see
Export	<i>name exported successfully</i>
Export again with the same name	The dialog Overwrite Export Notebook . Choose to replace it
Import	<i>Notebook name imported successfully</i>
Import a name that already exists in the project	The dialog Notebook Name Conflict , with Replace (<i>Notebook name replaced successfully</i>) or Keep Both (<i>... kept and new notebook imported successfully</i>)

Administrators can list, import across sites and delete published notebooks in the admin console.

5.9 Troubleshooting

Symptom	Cause	Fix
<i>Unable to open notebook</i>	The session is not running or cannot be reached	Start the session, then select Try again
<i>Save failed</i>	The notebook could not be saved	Select Retry save . Copy your work if it keeps failing
<i>A cell says The kernel is not connected yet.</i>	The kernel is still connecting	Wait a moment and run again
The output grid shows fewer rows than the table	The preview is capped at 50,000 rows	Aggregate or filter in code
A variable you defined is gone	The kernel restarted	Run the cells again

6 Datasets

A **dataset** is a saved, named result. You define it with a SQL query on a connection, or with Python code that runs on a runtime. Groundzero runs it when asked and shows the rows in pages. After this chapter you can build a dataset, run it, page through the results and save it for charts and dashboards to use.

Datasets live under **Designer**, then **Datasets**. Visualizations, filters and dashboards are built on datasets. This chapter covers the dataset itself.

Use a dataset when	Use something else when
Charts and dashboards need to share a governed table	The work writes data somewhere. Use an ETL job
One SQL query, perhaps with a small Python step, gives you what you need	You are exploring. Use a notebook
You want to page quickly through up to thousands of rows	The data must be rebuilt on a schedule into a table. Use a job and the Scheduler

Before you start, you need at least one connection. A SQL dataset on **Standard** needs no compute. A code dataset needs a runtime and a compute size.

6.1 Create a dataset

1. Open **Designer**, then **Datasets**.
2. Select **New Dataset**.
3. Type a name in **Enter Name** (same name rule as projects).
4. Select **Add**, then open the new dataset from the list.

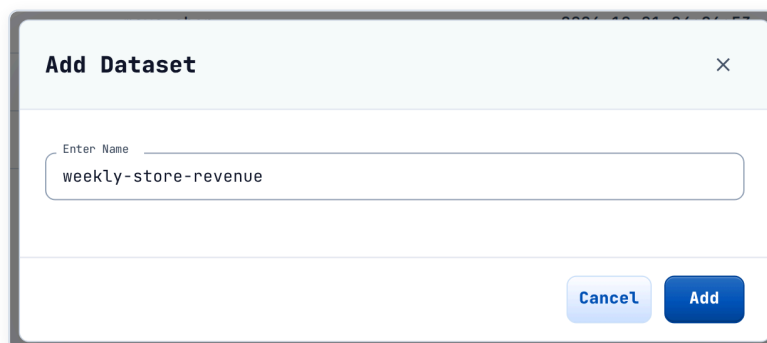


Figure 34. The Add Dataset dialog.

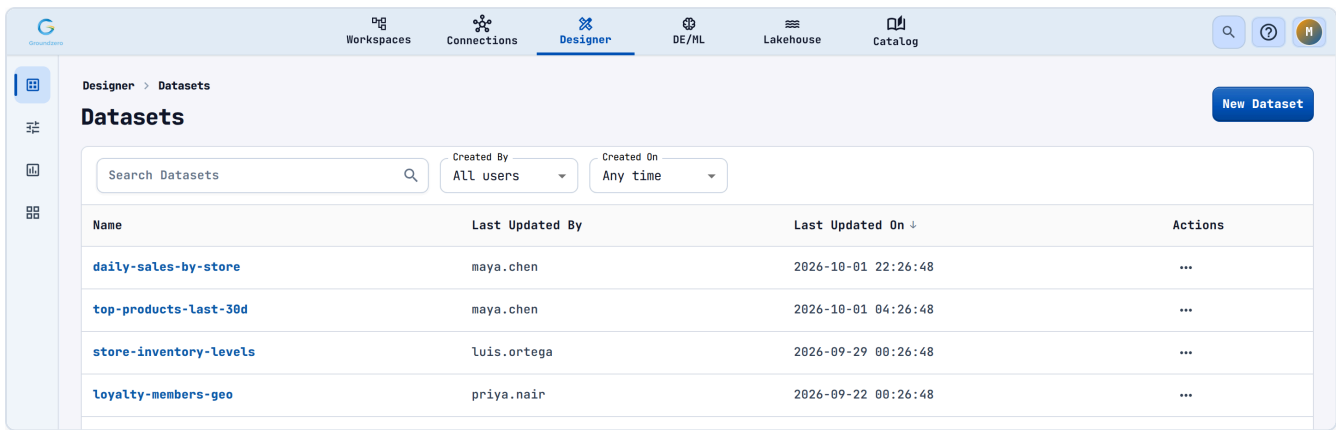


Figure 35. The Datasets list, with search and two filters above the table.

Control	Meaning
Search Datasets	Filters by name as you type
Created By	Shows datasets of one user, or All users
Created On	Any time, Today, Last 7 days or Last 30 days
Columns	Name, Last Updated By, Last Updated On, Actions (Rename, Delete)

Rename and Delete are in the row menu. A deleted dataset shows *name deleted successfully*. A failure shows *Error deleting name*.

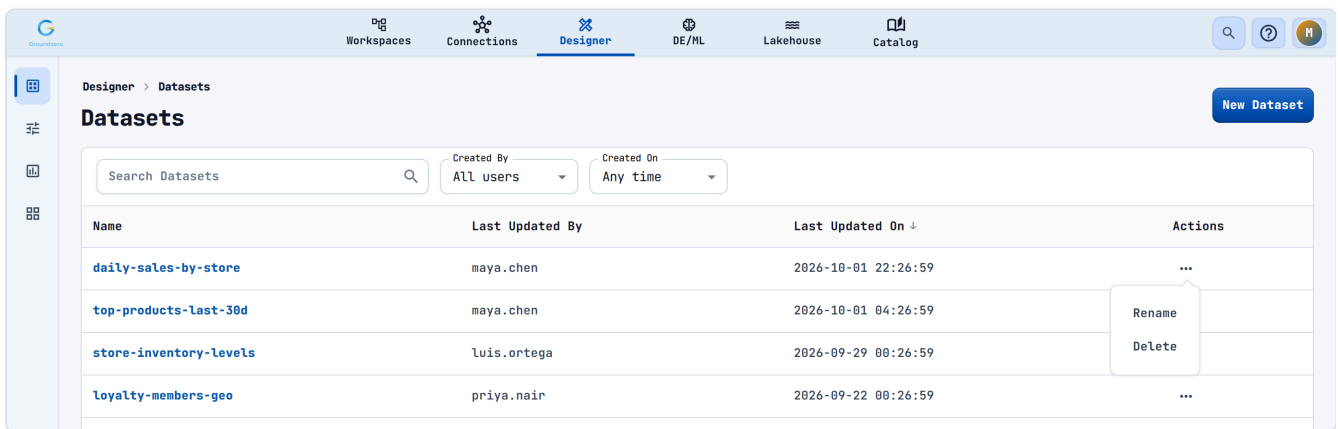


Figure 36. Row menu of the Datasets list.

6.2 Write and run a query

By the end of this task you have rows on screen.

1. Open the dataset.
2. In the left drawer, open **Connections** and tick the connection to query. It shows its databases, tables and columns.
3. On the **Query** tab, write your SQL. A table's menu in the drawer inserts the table name at the cursor.
4. Leave the runtime picker on **Standard**, or choose another runtime (see below).
5. Select **Run**. The button turns into a red **Stop** while it runs.

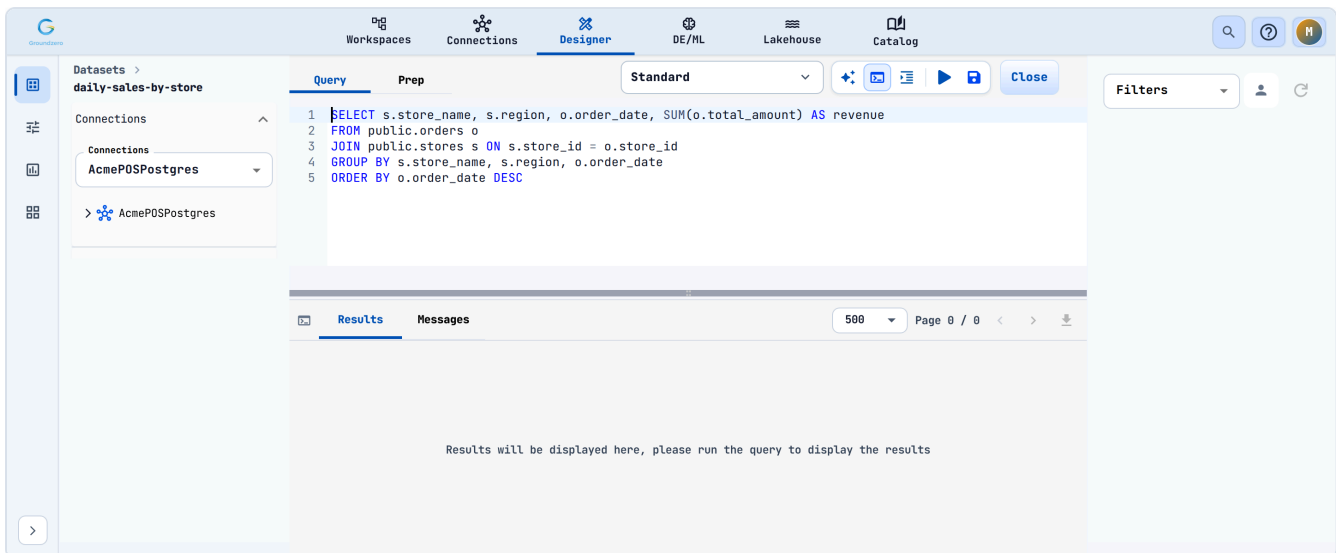


Figure 37. A SQL dataset on the Standard runtime. Left: Connections, Models, System Variables. Centre: the Query and Prep tabs. Right: Filters. Bottom: Results and Messages.

How to check it worked: The **Results** tab shows the response time, the row count and the rows, and **Messages** shows success. **Run** and **Save** stay greyed out until a connection is ticked.

Area of the editor	Purpose
Connections	Pick one or more connections. A refresh icon on a connection reloads its tables
Models	Pick registered models to load into the dataset (see Models and MLflow). This makes it a code dataset
System Variables	<code>\${SITENM}</code> , <code>\${CLNTNM}</code> , <code>\${RGN}</code> , <code>\${USR}</code> and <code>\${USRGRPS}</code> . Select the copy icon beside one. You see <code>\${NAME}</code> copied to clipboard!
Query tab	SQL on the ticked connection
Prep tab	Python that turns the query result into the final result. The default, <code>transformedData = rawData</code> , passes the rows through unchanged
Runtime picker	Standard, or a runtime with a compute size
Assistant	The code assistant for this dataset
Output toggle, Format SQL , Run , Save , Close	Toolbar buttons. Format SQL shows only on the Query tab
Filters panel	Dataset filters and their values. See the Analyze, AI and Admin guide

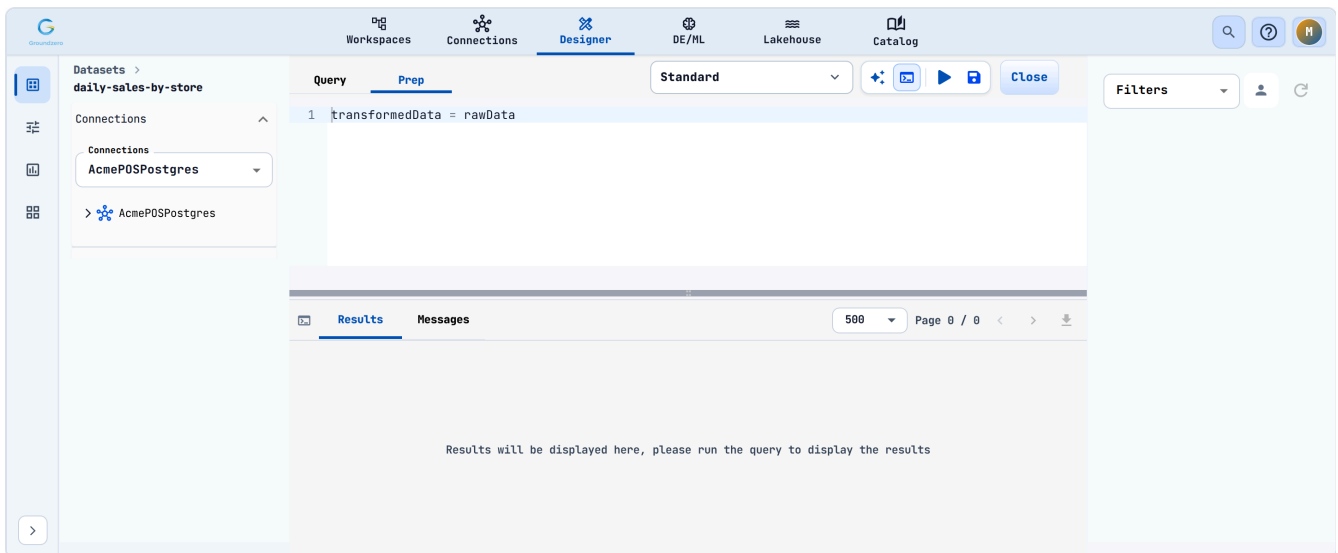


Figure 38. The Prep tab holds Python. `rawData` is the query result. Your code assigns the final rows to `transformedData`.

6.3 Choose between a SQL dataset and a code dataset

The editor picks the mode from your connection and runtime.

	SQL dataset	Code dataset
Tabs	Query and Prep	Prep only
Runtime	Standard	A runtime with a compute size
Connections	One SQL connection, or none	Several connections, or one that is not SQL
Page sizes	50, 100, 500, 1,000	50, 100, 500, 1,000, 10,000
Default page size	500	10,000

Standard works with these SQL connections: SQL Server, PostgreSQL, MySQL, Oracle, Teradata, DB2, SAP HANA, Exasol, Snowflake, Redshift, Google BigQuery, Athena, Cassandra, Cosmos DB, Elasticsearch, TimescaleDB, Firebolt, SkySQL and GZ Lakehouse.

You get a code dataset when you choose another runtime, tick more than one connection, tick a connection that is not in that list, or pick a model. The Query tab then disappears, and your Prep code is the whole dataset.

6.3.1 Choose a runtime

Open the runtime picker in the toolbar.



Figure 39. The Runtime list for a dataset starts with Standard, followed by your site's runtimes.

Standard has no compute list. Any other runtime shows **Compute** and, for Spark, **Minimum Workers**. See Runtimes and compute.



Figure 40. With Standard selected there is nothing else to set.

6.4 Read the results

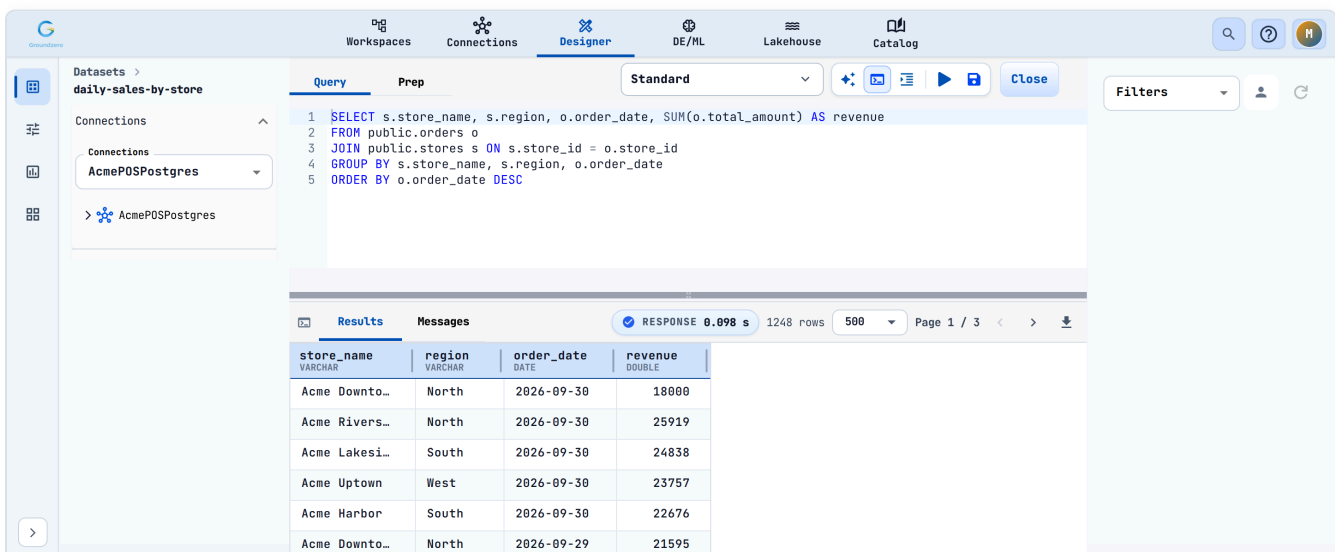


Figure 41. Results show the Response time, the row count, the page size, the page counter and a CSV download.

Item	Meaning
Response	How long the request took. Reads Running with a spinner during a run
Row count	Total rows, for example <i>1248 rows</i>
Page size	Rows per page
Page 1 / 3 with arrows	Move through pages
Download icon	<i>Download current data as CSV</i> saves the page you see. You see <i>CSV downloaded successfully</i>

Each column header shows its name and data type. The **Messages** tab shows success or the error text, in red when it contains *error*, *failed* or *not*.

6.4.1 Change the page size

Choose a size from the page-size list. Groundzero runs the dataset again from page 1. The next-page arrow works while more rows exist.

The screenshot shows the Groundzero Designer interface. On the left, there's a sidebar with 'Datasets' and 'Connections'. The main area is divided into 'Query' and 'Prep' tabs. The 'Query' tab is active, showing a SQL query:

```
1 SELECT s.store_name, s.region, o.order_date, SUM(o.total_amount) AS revenue
2 FROM public.orders o
3 JOIN public.stores s ON s.store_id = o.store_id
4 GROUP BY s.store_name, s.region, o.order_date
5 ORDER BY o.order_date DESC
```

Below the query, there's a 'Results' tab showing a table with 4 columns: store_name (VARCHAR), region (VARCHAR), order_date (DATE), and revenue (DOUBLE). The table has 6 rows of data. To the right of the table, there's a dropdown menu for 'Page size' with options 50, 100, 500, and 1000. The current page size is 500. Above the table, it says 'RESPONSE 0.031 s' and '1248 rows'. At the bottom right, it says 'Page 1 / 3' with navigation arrows and a download icon.

Figure 42. Page size options for a Standard dataset: 50, 100, 500 and 1,000.

A code dataset shows at most 10,000 rows per page.

6.4.2 Get fresh results

Groundzero keeps the result of an identical query and page in your browser for 10 minutes, so running it again within that time shows the stored result. Reload the page to start fresh.

TIP

Use the circular refresh arrow beside **Filters** to reload filter values. Use the refresh icon on a connection to reload its tables after a table changes.

6.5 Save a dataset

Select **Save**. Groundzero saves the connections, queries, runtime, compute, minimum workers, GPU size, models and filters.

What happens next: You see *Dataset updated*. A failure reads *Error occurred while updating*.

6.6 Use filters in a query

Filters you attach to a dataset put their values into your code as `{{placeholder}}` tokens. Before a run, the editor checks that every token has a value.

Message	Cause	Fix
Select values for: <code>{{region}}</code>	A placeholder in the code has no filter value yet	Pick a value in the Filters panel
Select values for all filters before running the dataset	An <code>IN</code> clause has no values	Pick values for every filter

6.7 Troubleshooting

Message	Cause	Fix
Please select Connection	You ran with no connection	Tick one in the left drawer
Please select Compute	A runtime is chosen but no compute	Choose a Compute size
Please select RTE and Compute	Neither is chosen	Choose both
Connection details are not valid / The connection selected is not available	The connection was edited or removed	Pick another, or fix it on the Connections page
Error occurred while running the code	The run failed	Open Messages to read why
No SQL query to format / Unable to format this SQL query	Format SQL on an empty query or one that cannot be formatted	Write or fix the SQL
No data available to download	You downloaded CSV with no rows	Run the dataset first
The dataset session stopped responding before it took the run. Run the dataset again to start a new session.	The dataset session stopped	Run again

7 Models and MLflow

Groundzero uses **MLflow** to track machine learning experiments and manage models. After this chapter you can log a model from a notebook, look at your experiments and runs, share a model with other projects, and use a model in a job or dataset.

Notebook sessions are already set up to record to MLflow for your site. A model you train and log with MLflow shows up on the project's **Models** tab.

Use models when	Notes
You train models in notebooks or jobs and want to compare runs	Log runs with MLflow in your code
Another team's trained model should be reused	Import the published model
A job or dataset should score data with a model	Pick the model under Models in the job or dataset editor

You need a project, and models logged by you or shared with you. The list shows the models you may see. You also need a compute size to open the MLflow screen. Groundzero uses the smallest one for that.

7.1 Log a model from a notebook

Inside a notebook, log with the MLflow library as you would anywhere. Nothing needs configuring. After the run finishes, the model is listed on the Models tab.

```
import mlflow
import mlflow.sklearn
from sklearn.ensemble import RandomForestClassifier

with mlflow.start_run(run_name="churn-baseline"):
    model = RandomForestClassifier(n_estimators=200).fit(X_train, y_train)
    mlflow.log_param("n_estimators", 200)
    mlflow.log_metric("accuracy", model.score(X_test, y_test))
    mlflow.sklearn.log_model(model, "model")
```

Choose a runtime that contains your library, such as one of the `sklearn`, `tensorflow` or `pytorch` runtimes, or ask an administrator for a custom runtime.

7.2 Look at experiments, runs and the registry

1. Open the project's **Models** tab.
2. Open the row menu of a model and select **View in MLflow**.
3. Wait a few seconds. The message *Please wait, you will be redirected to the MLflow UI* appears, and MLflow opens in a new browser tab.
4. In MLflow, use **Experiments** to compare runs and their parameters, metrics and artifacts. Use the **Models** registry to see registered versions and their stages.

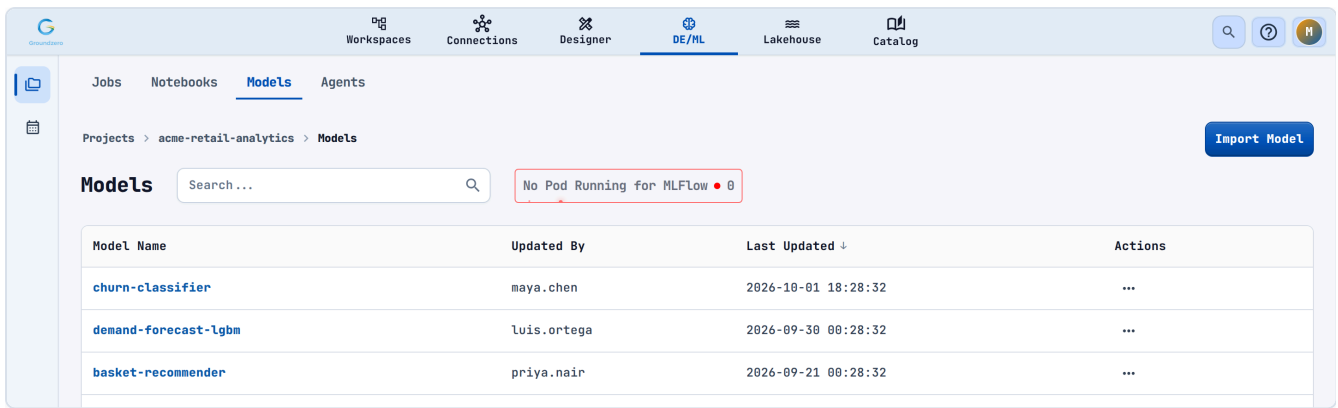


Figure 43. The Models tab. The red status box says No Pod Running for MLFlow until you open the MLflow screen.

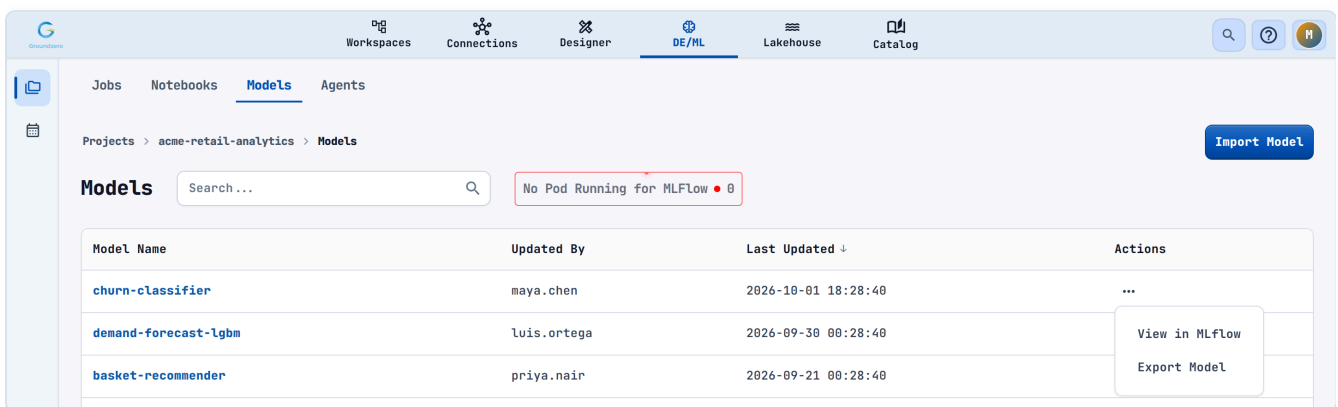


Figure 44. The row menu of a model.

Column	Meaning
Model Name	The name of the model
Updated By	The user who last changed it
Last Updated	The time of the last change, in your local time
Actions	View in MLflow and Export Model

The status box beside the search field shows whether MLflow is running for you. It turns green when it is. Each user has one MLflow, and Groundzero reuses it if it is already running.

When you are done, select **Stop Running Pod** and confirm. You see *Running MLFlow pod stopped successfully*.

7.3 Share a model

7.3.1 Publish a model

1. Open the row menu and select **Export Model**.
2. If the name is new, you see *name exported successfully*.
3. If a model with that name is already published, the **Overwrite Export Model** dialog asks you to confirm.

Publishing shares the model, together with the runtime it needs, with your site.

7.3.2 Import a published model

1. On the Models tab, select **Import Model**.
2. In **Select Model**, choose a published model.
3. Type an **Experiment Name** for your copy.
4. Type **Import Model As (must be unique)**. The dialog says *Model Name is available* in green or *Model Name already taken* in red.
5. Select **Import**.

Figure 45. Import Model. The unique-name check runs as you type.

What happens next: You see *name is getting imported*. Groundzero registers the model in your MLflow, using the library it was built with (`sklearn` , `tensorflow` or `pytorch`).

7.4 Use a model

- In a job or dataset editor, open **Models** in the left drawer and pick the model. It is loaded before your code runs.
- In a notebook, load it with MLflow in your own code.

7.5 Troubleshooting

Message	Cause	Fix
Error occurred while fetching running MLFlow pods	The check for a running MLflow did not work	Refresh the tab
Error occurred while starting MLFlow	MLflow could not start	Try again, and check that compute is available
Error occurred while exporting model	Publishing failed	Try again
Model Name already taken	The import name exists	Choose another name

Administrators manage published models in the admin console.

8 Connecting from MCP clients

Groundzero can work with the coding assistants you already use. After this chapter your assistant can list your projects, write and run jobs, open notebooks, query datasets and look at MLflow experiments for you.

You sign in with your own Groundzero account, so the assistant can do only what you can do. This works with Codex, Claude Code, Cursor and VS Code with Copilot agent mode, or any client that supports HTTP and OAuth sign-in.

8.1 Connect a client

1. Select the help icon (question mark) in the top bar.
2. Choose **Connect MCP**.
3. Pick your client in **Choose your client** and follow the numbered steps. Each step has a copy button.

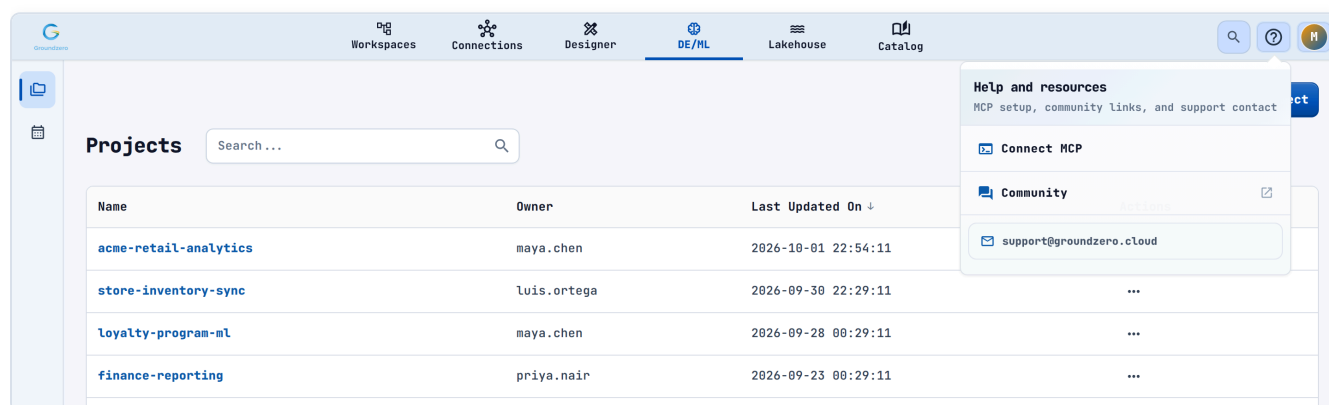


Figure 46. The Help and resources menu has Connect MCP, Community and the support address.

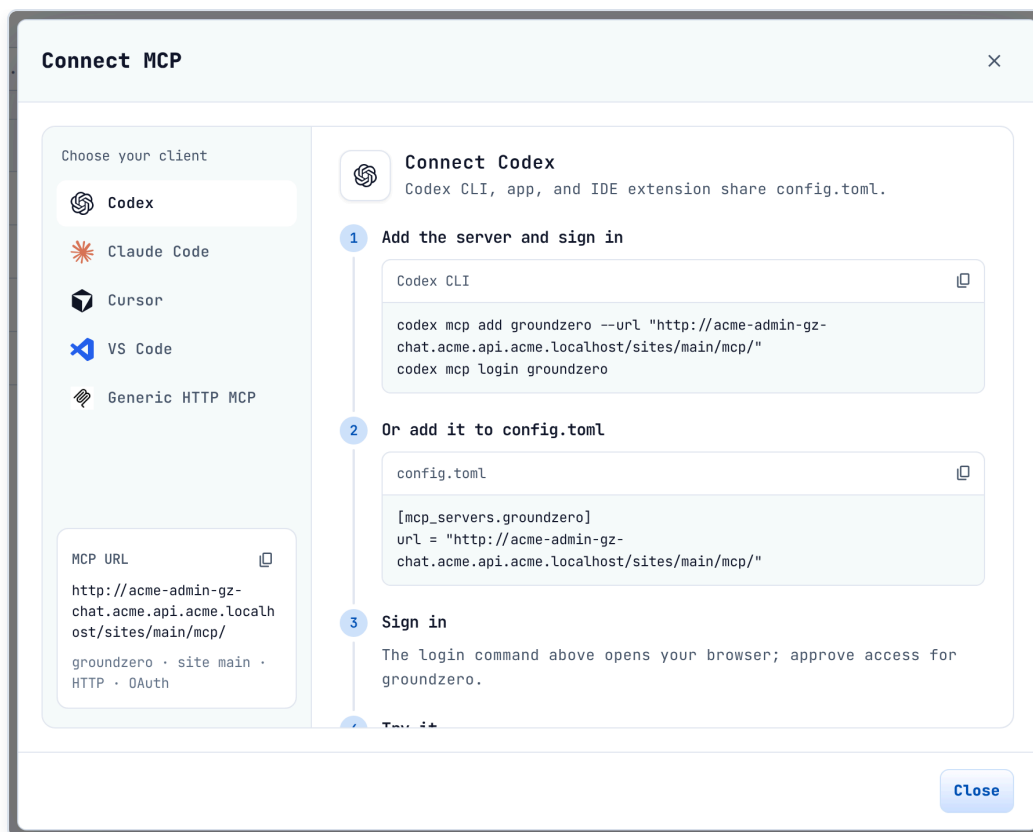


Figure 47. The guide shows the MCP URL for your site and the steps for each client.

Client	What you do
Codex	Run <code>codex mcp add groundzero --url "<URL>"</code> , then <code>codex mcp login groundzero</code> . Or add the server to <code>config.toml</code>
Claude Code	Run <code>claude mcp add --transport http groundzero "<URL>"</code> , then run <code>/mcp</code> inside Claude Code to sign in. Or share a <code>.mcp.json</code>
Cursor	Add the server to <code>.cursor/mcp.json</code>
VS Code	Add it to <code>.vscode/mcp.json</code> for GitHub Copilot agent mode
Generic HTTP MCP	Transport Streamable HTTP, the URL shown, authentication OAuth, scopes <code>openid email profile</code>

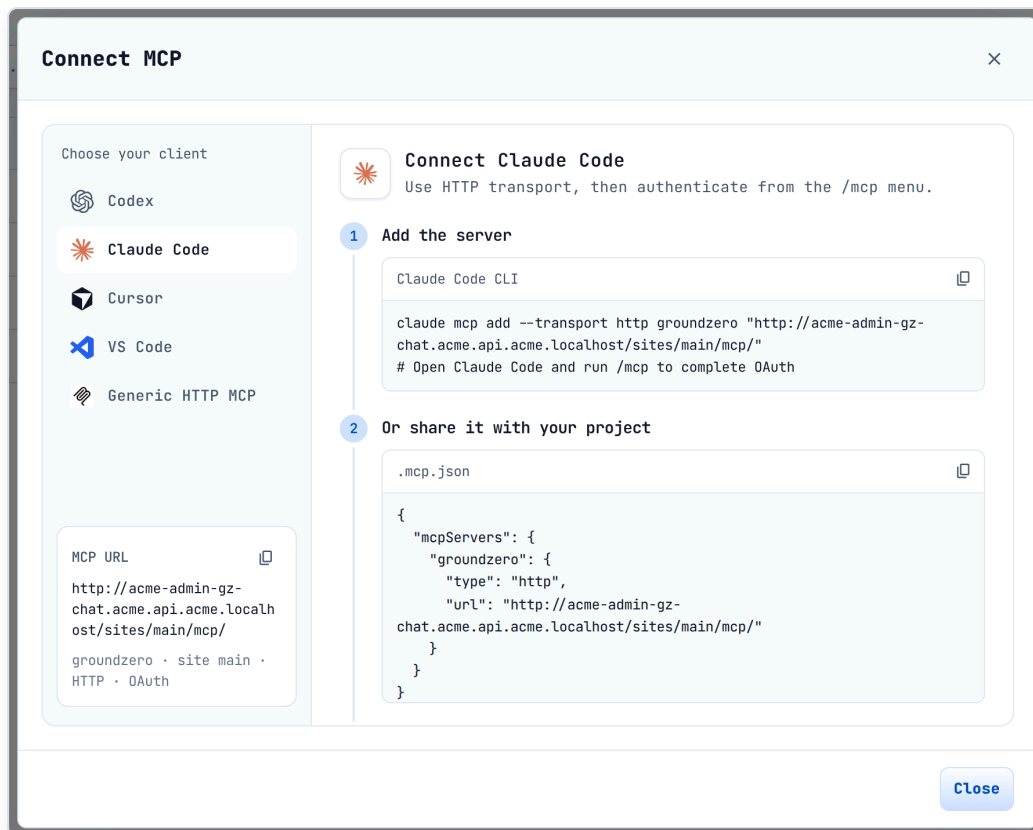


Figure 48. Steps for Claude Code. Replace the sample address with the one the guide shows for your site.

What happens next: Your browser asks you to approve access for the server named `groundzero`. Approve it, then try the first prompt the guide suggests: *List the datasets I can access in Groundzero.*

NOTE

The address in the screenshot is an example. The **MCP URL** box in your console shows the address of your site. Never share it together with a token.

8.2 What your assistant can do

Area	Examples
Projects and jobs	List projects and jobs, create them, save job code, run a job and read its logs
Connector syntax	Look up the right way to read from or write to a connection type
Notebooks	List, create, rename, delete, start and stop notebooks, write and run code, publish and import
Machine learning	List models, import published models, open MLflow, and list experiments, runs and registered models
Datasets and runtimes	List and read datasets, create and save them, list available runtimes

Ask your assistant to list the Groundzero tools it can use, or to run several independent requests at once (up to 24).

8.2.1 Read and write access

Actions that create, change, run, start or stop something, such as creating a job, saving job code, running a job, starting a notebook, running notebook code or publishing a notebook, need write access. Read-only actions, such as listing jobs, need only read access.

TIP

Ask your assistant to look up connector syntax before it writes job code. Do not ask it to batch requests that depend on each other's results.

8.3 Troubleshooting

Symptom	Fix
The client cannot sign in	Check that it supports OAuth over HTTP, and that the address ends with <code>/mcp/</code>
A tool returns <i>You do not have ... permission</i>	Ask an administrator for the permission on that resource
<i>MCP write scope is required for tool: name</i>	The client was approved for read only. Sign in again and approve write access

9 Glossary

Term	Meaning
Project	A container for jobs, notebooks and models. Has a name and an owner
Job	Python code that reads, transforms and writes data. Runs on a runtime, on demand or on a schedule
Sub-job	A job from the same project that runs together with a main job
Task	A scheduled run of a job in the Scheduler
Run	One execution of a job. Its status is Running, Success, Failed or aborted
Notebook	An interactive document of code and Markdown cells that runs on a kernel
Session	A running notebook. You start and stop it
Kernel	The Python process in a session that runs your cells
Checkpoint	A restore point of a notebook that you can go back to
Dataset	A saved SQL or code query whose result is shown in pages and used by charts and dashboards
Standard runtime	The dataset option that runs one SQL query directly on a source, with no compute to size
Runtime (RTE)	The environment that runs your code, such as a Spark or DuckDB runtime with its libraries
Compute	The memory and processor size given to a runtime
Minimum Workers	The number of Spark workers to keep. Spark runtimes only
GPU Size	The number of GPUs for a runtime
Custom RTE	A runtime an administrator builds from a base runtime and a list of extra packages
Connection	A saved link to an outside system. Code uses it by name and never holds credentials
System variable	A placeholder such as <code>\${JOBNAME}</code> that Groundzero fills in before your code runs
Published (exported) item	A copy of a job, notebook or model shared with everyone on the site
MLflow	The experiment tracking and model registry behind the Models tab
MCP	Model Context Protocol, the standard through which assistants work with Groundzero