ETL with Meltano + Singer in the LLM Era

By Pat Nadolny

About me

Current:

- Senior Software Engineer @ Arch (formerly Meltano)

Previous:

- Senior Data Engineer @ Meltano
- Data Engineer @ Walmart Ecomm’s Bonobos
- Data Analyst @ Deloitte Consulting

What I'll talk about today

1. Singer Spec
2. Meltano Projects and Open Source Ecosystem
3. Data Engineering for LLMs
4. Q&A

What is the Singer Spec?

Singer Spec

- Created by **Stitch** in 2016
- Spec for data interchange for ELT
  - JSON over **Unix pipe** via standard out
  - State bookmarks
  - JSON schema messages
  - Metadata
  - Logging, Versioning, Metrics, more
- Taps = Extractors = Readers
- Targets = Loaders = Writers
- Standalone GitHub repos

```
{"type": "SCHEMA", "stream": "users", "key_properties": ["id"], "schema": {"require
{"type": "RECORD", "stream": "users", "record": {"id": 1, "name": "Chris"}}
{"type": "RECORD", "stream": "users", "record": {"id": 2, "name": "Mike"}}
{"type": "SCHEMA", "stream": "locations", "key_properties": ["id"], "schema": {"req
{"type": "RECORD", "stream": "locations", "record": {"id": 1, "name": "Philadelphia
{"type": "STATE", "value": {"users": 2, "locations": 1}}
```

```
{
  "type": "RECORD",
  "stream": "tools",
  "time_extracted": "2021-11-20T16:45:33.000Z",
  "record": {
    "id": 1,
    "name": "Meltano",
    "active": true,
    "updated_at": "2021-10-20T16:45:33.000Z"
  }
}
```

```
{
    "properties": {
      "id": {
        "type": "integer"
      },
      "name": {
        "type": "string"
      },
      "active": {
        "type": "boolean"
      },
      "updated_at": {
        "type": "string",
        "format": "date-time"
      }
    }
```

Singer Benefits

- Well defined
- **Robust features** baked in: incremental, metrics, schema validation, etc.
- **Interchangeable** connectors
- **Flexible** - new features (e.g. Batch messages), fork and customize, etc.
- Mostly **Python** (not required) the language of choice for data teams
- **Large ecosystem** of existing implementations

Singer Challenges

- Challenging to **orchestrate**
- **Inconsistent** implementations
- Sometimes **hard to discover** taps split across github repos
- **Metadata and docs** are sometimes lacking
- Sustainable open source **contributing** and maintaining is difficult

What is Meltano?

The Meltano story

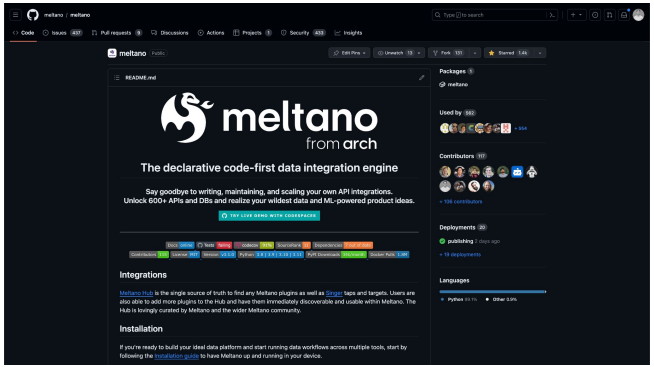
- 2018 - Built by GitLab's Data Team
- Insight: data tools were **lacking software engineering features**
  - Git backed
  - Everything as code, code reviews
  - Testing, Isolated Envs, etc.
- 2020 - Ultimately value found in **orchestrating** Singer ELT
- Embrace Singer and level up the community
- **Suite of products** to solve Singer challenges
- 2021 - Spun out of GitLab as a standalone company

Meltano Core

- Solves: Singer **orchestration**
- Engineering practices
  - CLI first
  - Yaml based
  - Environments
- 4300+ Slack Members
- 1400 GH stars
- 150 GH contributors
- 1000's of projects

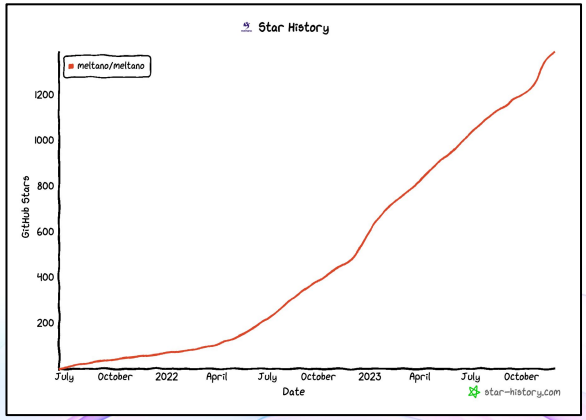
```
meltano add extractor tap-postgres
meltano config tap-postgres set --interactive
meltano test extractor tap-postgres


meltano add extractor tap-spreadsheets-anywhere
meltano config tap-spreadsheets-anywhere set --interactive
meltano test extractor tap-spreadsheets-anywhere
```



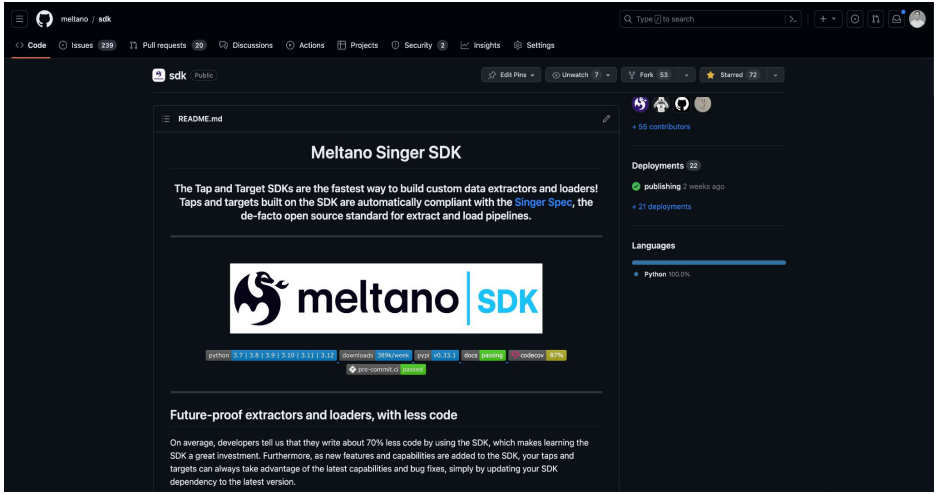
meltano.yml

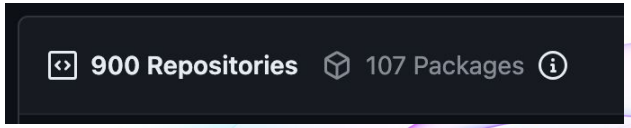
```yaml
plugins:
  extractors:
  - name: tap-shopify
  - name: tap-postgres--shopify-configs
    inherit_from: tap-postgres
    select:
      - shopify_configs.*
  loaders:
  - name: target-snowflake
schedules:
- name: sync-all-shopify
  interval: @hourly
  config_source:
    tap-shopify: tap-postgres--shopify-configs
  extractor: tap-shopify
  loader: target-snowflake
```



Meltano Singer SDK

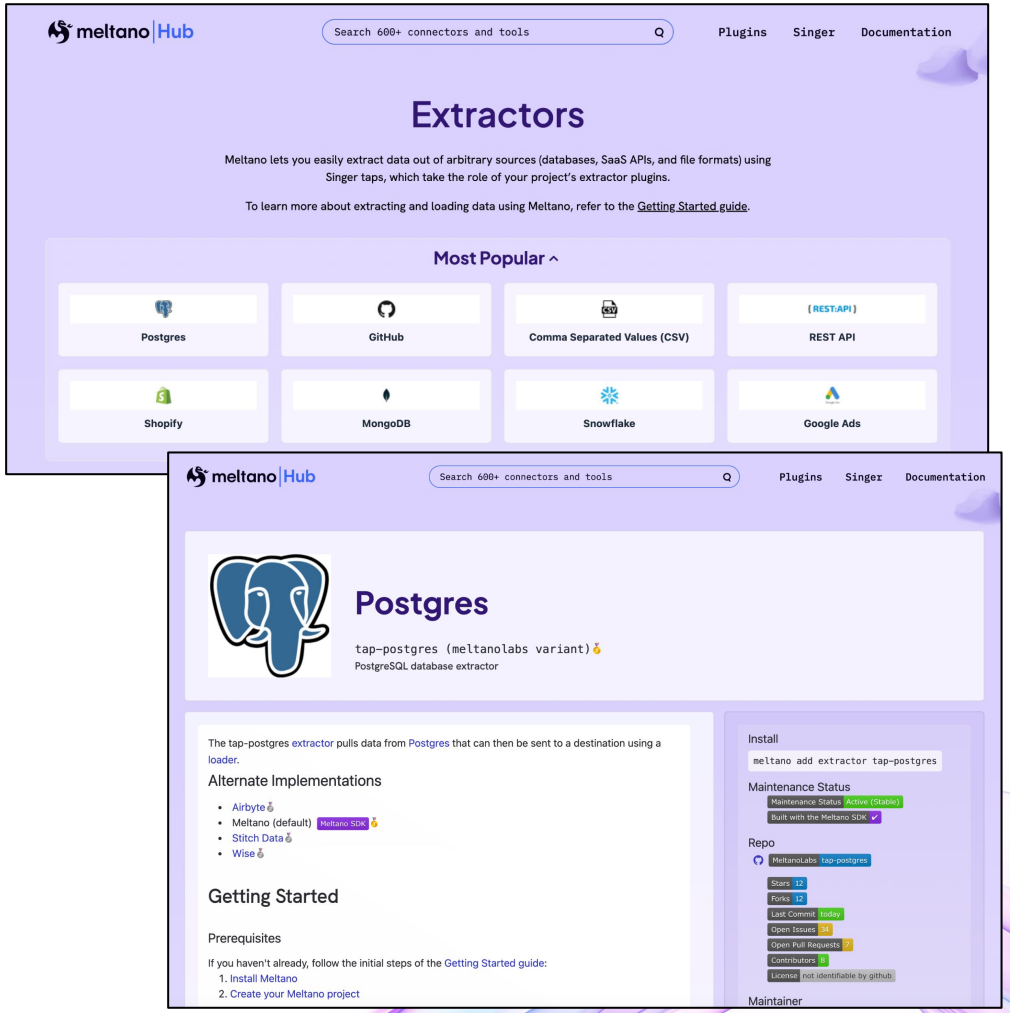
- Solves: Singer **inconsistent implementation**
- REST/GraphQL/SQL/ etc. helpers
- Parent child relationships
- Authentication helpers
- Rate limiting
- ~900 repos
- 70% less time





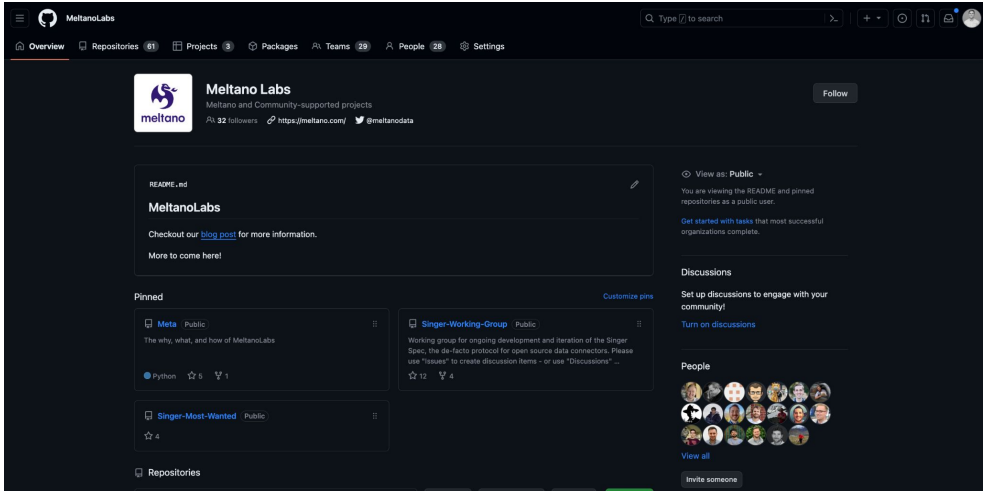
Meltano Hub

- Solves: Singer **discoverability**
- 600+ unique connectors listed
- Settings and documentation
- Usage stats
- API used by Meltano CLI



Meltano Labs

- Solves: Singer **contribution and maintenance**
- Ownership Models
  - **NEW: Community-Managed** Fork with Community Maintainers 🤝
  - Single Named Owner (singer-io, pipelinewise, etc.)
  - Vendor Self-Managed
  - Benevolent Community Member (pnadolny13 GitHub user)



Meltano x Singer

- ✅ Orchestrate - **Meltano Core**
- ✅ Implement - **Meltano Singer SDK**
- ✅ Discover - **Meltano Hub**
- ✅ Contribute and maintain - **Meltano Labs**

What is Arch?

(the company formerly known as Meltano)

arch

The bridge between your customers' data & your code

Stop wasting time on your own OAuth flows, API integrations, and data pipelines. Get back to shipping features with instant access to all your customers' data: raw, mapped, or embedded

LLM Apps Are Mostly Data Pipelines

Initial AI Excitement 📈🚀

Bloomberg

Technology + Work Shift

Microsoft Invests $10 Billion in ChatGPT Maker OpenAI

REUTERS

Technology

ChatGPT sets record for fastest-growing user base - analyst note

By Krystal Hu

February 2, 2023 10:33 AM EST · Updated 10 months ago

Falcon 180B

Hacker News new | past | comments | ask | show | jobs | submit

LlamaIndex raises $8.5M seed round, led by Greylock Partners (medium.com/llamaindex-blog)

33 points by freezed8 5 months ago | hide | past | favorite | 17 comments

Introducing Llama 2

The next generation of our open source large language model

Llama 2 is available for free for research and commercial use.

Download the Model

mosaicML

MPT-7B

A New Standard for Open-Source, Commercially Usable LLMs

a BigScience initiative

BLOOM

176B params · 59 languages · Open-access

BERT

MARKETS BUSINESS INVESTING TECH POLITICS CNBC TV INVESTING CLUB PRO NEW

Microsoft-backed OpenAI announces GPT-4 Turbo, its most powerful AI yet

PUBLISHED MON, NOV 6 2023-1:15 PM EST | UPDATED MON, NOV 6 2023-3:51 PM EST

TC

Join TechCrunch+

Login

Search

OpenAI launches an API for ChatGPT, plus dedicated capacity for enterprise customers

Kyle Wiggers @kyle_l_wiggers / 1:00 PM EST • March 1, 2023

Comment

Claude 2

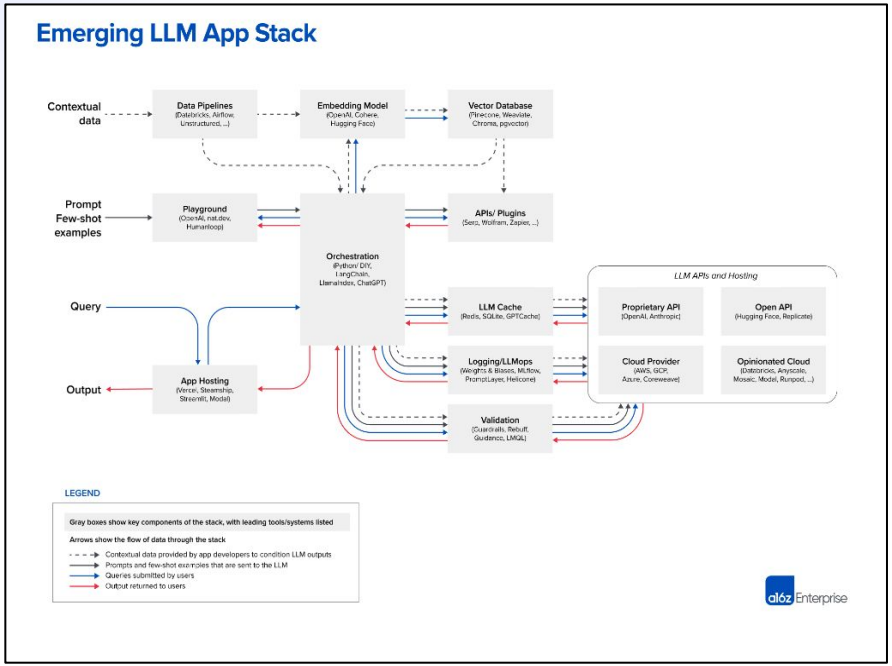
LANGCHAIN BLOG

Home By LangChain Release Notes GitHub Docs Case Studies Sign in Subscribe

Announcing our $10M seed round led by Benchmark

4 MIN READ APR 4, 2023

Taking a Step Back



Source: https://a16z.com/emerging-architectures-for-llm-applications/

"...this piece of the stack is relatively underdeveloped, though, and there's an opportunity for data-replication solutions purpose-built for LLM apps"

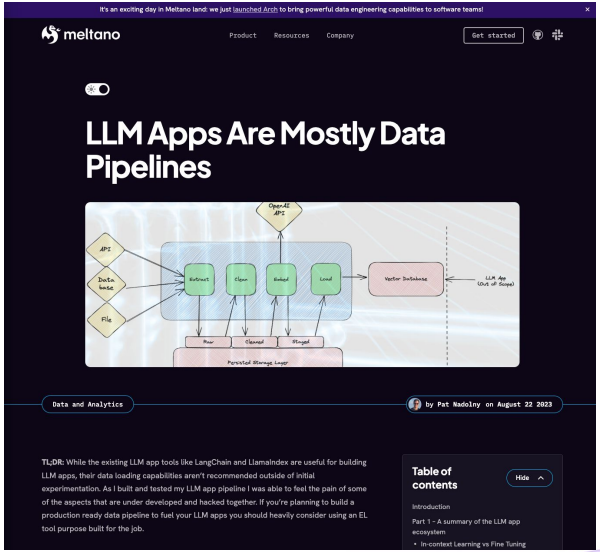
▲ 27 days ago | prev | next [–]

It is pointless - LlamaIndex and LangChain are re-inventing ETL - why use them when you have robust technology already?

▲ 26 days ago | parent | next [–]

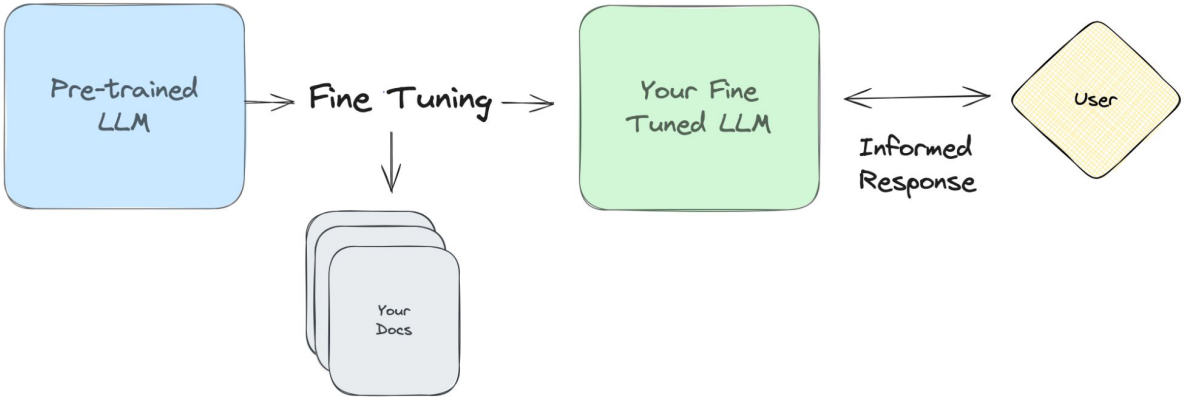
Why is this just not ETL, why do you need anything here? There is no new category or product needed here.

Findings 🔍


It's an exciting day in Meltano land: we just launched Arch to bring powerful data engineering capabilities to software teams!
meltano
Product
Resources
Company
Get started
LLM Apps Are Mostly Data Pipelines
Data and Analytics
by Pat Nadolny on August 22 2023
TL;DR: While the existing LLM app tools like LangChain and LlamaIndex are useful for building LLM apps, their data loading capabilities aren't recommended outside of initial experimentation. As I built and tested my LLM app pipeline I was able to feel the pain of some of the aspects that are under developed and hacked together. If you're planning to build a production ready data pipeline to fuel your LLM apps you should heavily consider using an EL tool purpose built for the job.
Table of contents
Hide
Introduction
Part 1 – A summary of the LLM app ecosystem
In-context Learning vs Fine Tuning

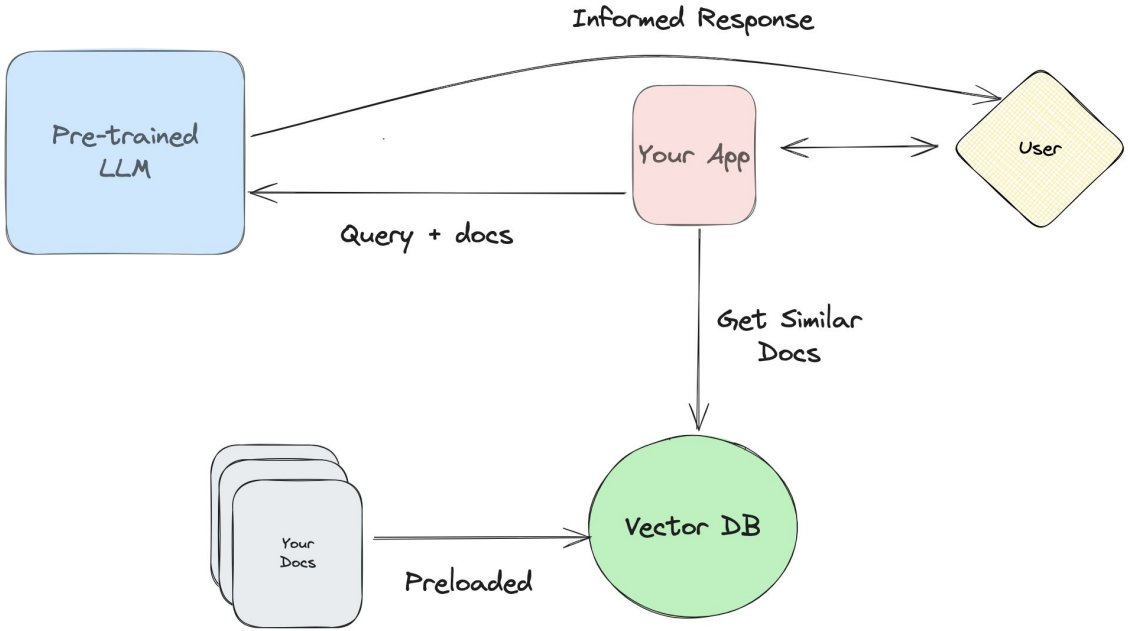
Fine-Tuning

- LLM doesn't have your domain specific knowledge
- LLM has shallow history "I'm sorry, I was trained on data from 2021…"
- New custom LLM trained on your data
- Expensive and challenging



In Context Learning

- Retrieval Augmented Generation (RAG)
- Vector database maintained with knowledge base of docs
- Prepends similar documents to original query for “context”
- Cheap and low effort



Takeaway 💡

- In context learning...
  - performs reasonably **well for most LLM use cases** as part of a RAG pipeline and **is the preferred approach**
  - leverages “off the shelf” tools like OpenAI’s API and Vector databases like Pinecone so a **small data team** can build an LLM app **without having to hire** specialized ML engineers

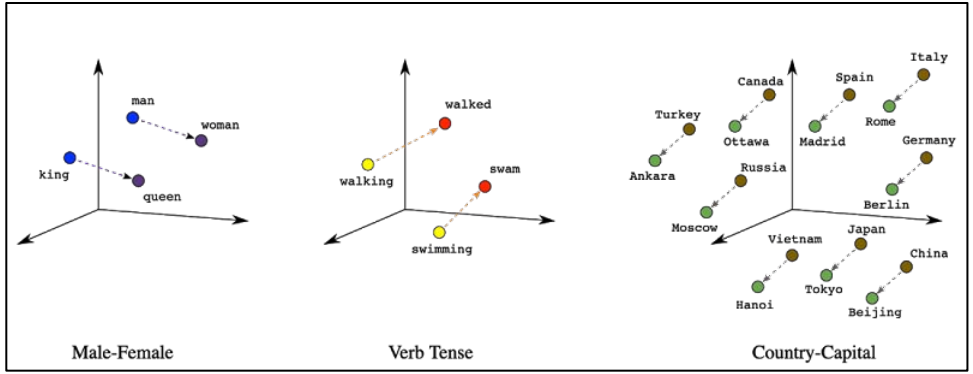
- Fine tuning...
  - performs better in **narrowly focused contexts** when the dataset is large and high quality
  - **requires more know-how** around getting your data to be properly weighted, i.e. not over or under indexing on your content
  - requires you to **host your own models and infrastructure** for serving it

Vector Databases and Embeddings

- Searchable text using semantic similarity vs keywords
  - Nearest neighbor search
  - Text with similar meaning
- Prompt context text
- Databases:
  - Purpose built for vector search
  - Vector search support

"Vector embeddings are a way to convert words and sentences and other data into numbers that capture their meaning and relationships"

source:https://weaviate.io/blog/vector-embeddings-explained



source:https://cloud.google.com/blog/topics/developers-practitioners/meet-ais-multitool-vector-embeddings

Pinecone

Weaviate

ClickHouse

pgvector / pgvector

LLM Tooling

- Langchain
  - App layer
  - Prompt chaining + memory
  - RAG vector DB retrieval
  - EL “data loaders”
- Llama Index
  - “A data framework for LLM-based applications to ingest, structure, and access private or domain-specific data”
  - LlamaHub - great extractors and tools but building an EL ecosystem from scratch

LLM Tooling – Takeaways 💡

- Great for app layer
- EL features aren't as robust as existing tools
- We should use great purpose built tools that already exist

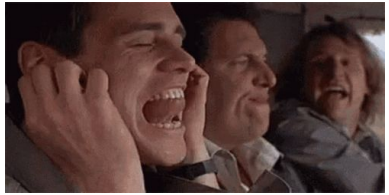
LlamaIndex

🦜🔗 LangChain

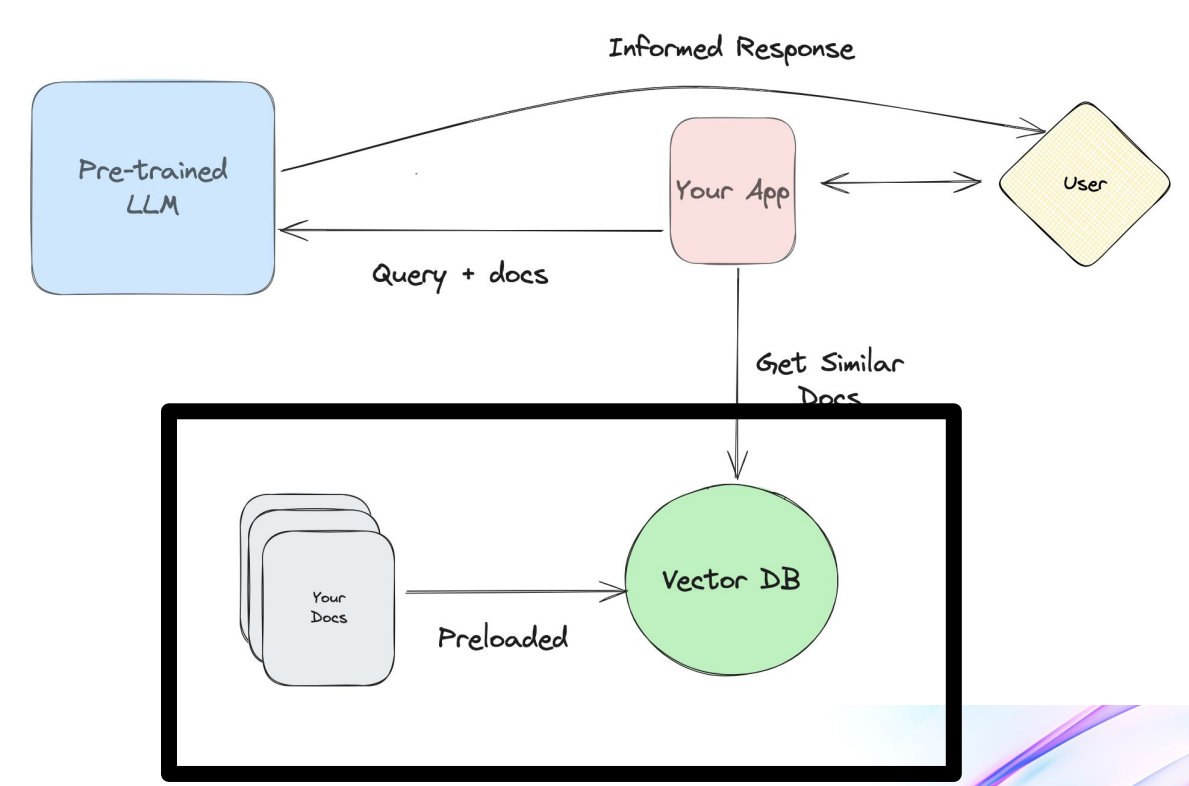
Component Parts

- Data Layer
  - Data movement
  - Enrichment
  - Storage

- App Layer
  - ~~Prompting~~
  - ~~Interfaces~~
  - ~~Retrieval~~



Component Parts



What can we apply from Data Engineering? 🛠️

DE Principles 📖📚

- Decouple, checkpoint, subtasks
- Process only new data
- Idempotency (run multiple times with no effect), deduplication built in
- Storage is cheap, keep raw data
- Extract once, transform multiple times
- Reproducibility
- Monitoring quality

Breaking it down

LLM App Pipelines

- **Data extraction** – e.g. pull message text from the slack API
- **Data cleansing** – e.g. remove certain characters, extra spaces, encoding, etc.
- **Data enrichment** – embedding
- **Data loading** – write to vector databases
- **Application UX** i.e. prompt chaining, retrieval, inference, memory, chat UI, etc.

Traditional ETL

- **Data extraction** – e.g. pull data from a variety of sources
- **Data enrichment and transformation** e.g. remove duplicates, add consistent names, aggregate complex data into consumable business metrics, etc.
- **Data loading** – write to a data warehouse
- **Data visualization** and consumption charts and dashboards that tell a story about the data

It’s just ETL again!!

ETL to ELT Learnings

- I'll spare the details
- Optimize the most expensive parts
- Extracting is slow, expensive, and once
- Transforming is fast, cheap, and frequent
- Skip the mistakes of ETL

Takeaway 💡 – Decouple Expensive Steps

- RAG
  - Extracting
  - Cleaning
  - Enriching (i.e. embedding)
  - Vector storage is too (but we'll skip that for now)
- Extraction decoupled from cleaning from embedding
- Re-embedding the shouldn't require re-extraction

Common EL Challenges

- Rate limited APIs and outages
- Pagination
- Metadata and logging
- Schema validation and data quality
- Personal Identifiable Information (PII) handling, obfuscation, removal, etc.
- Keeping incremental state between runs so you can pick up where you left off
- Schema change management
- Backfilling data

▲ 27 days ago | prev | next [–]

It is pointless - LlamaIndex and LangChain are re-inventing ETL - why use them when you have robust technology already?

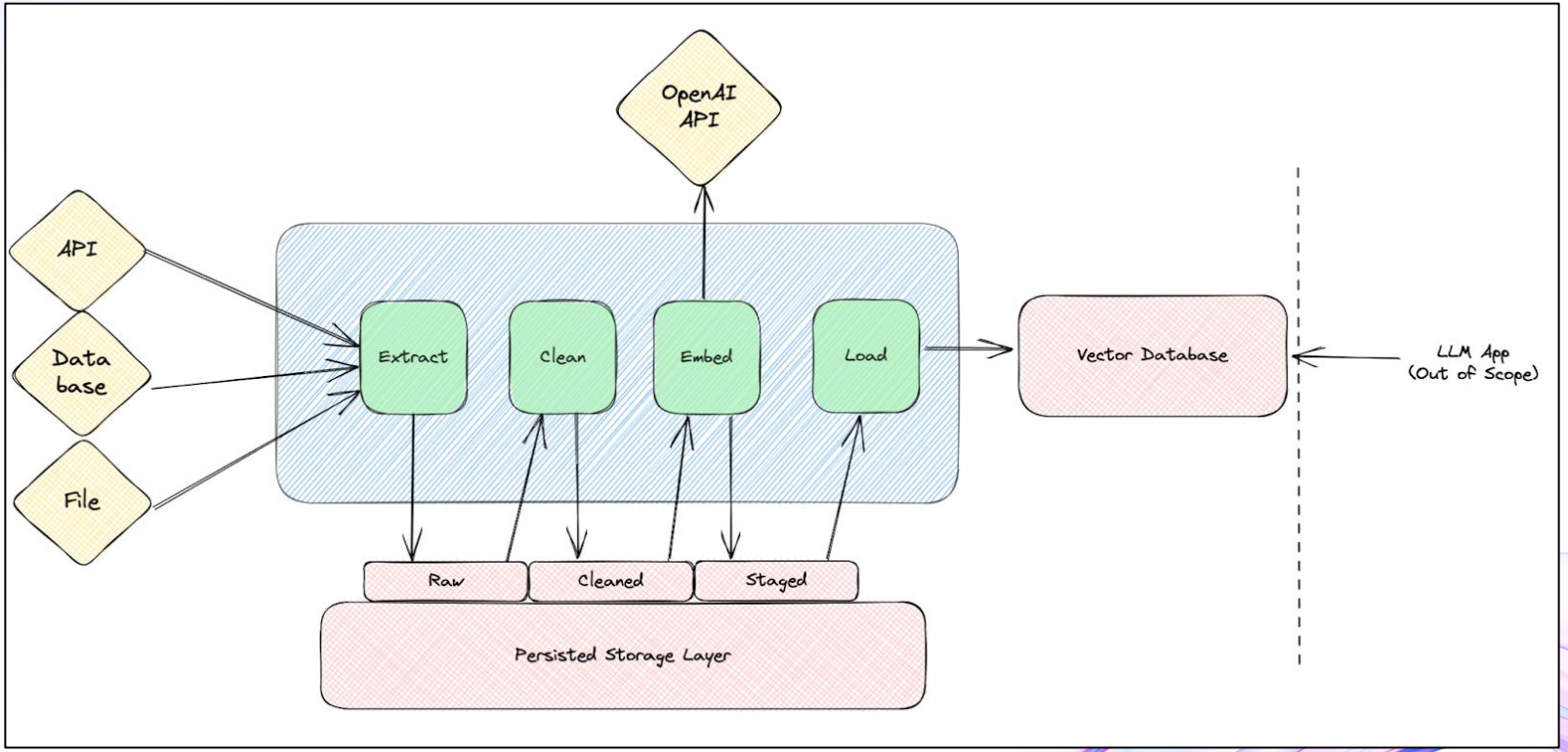
Takeaway 💡

Use an established framework…

meltano | SDK

🚀 🚀 🚀

✨Potential Design ✨



Implementing it with Meltano

Use Case

meltano | SDK

Search

DEVELOPER RESOURCES

- Getting Started
- In-depth Guides
- Python Tips
- Code Samples
- Command Line Samples
- Frequently Asked Questions

REFERENCE

- Classes Reference
- Singer Implementation Details
- JSON Schema Helpers
- Plugin Capabilities

ADVANCED CONCEPTS

- Incremental Replication
- Parent-Child Streams
- Stream Partitioning
- The Context Object
- Inline Stream Maps
- Batch Messages

Meltano Singer SDK

The Meltano Singer SDK for Taps and Targets is the fastest way to build custom data extractors and loaders! Taps and targets built on the SDK are automatically compliant with the Singer Spec, the de-facto open source standard for extract and load pipelines, and therefore Meltano.

If you're looking to add support to Meltano for a new data tool that would be listed on the Meltano Hub as a utility, check out the Meltano EDK (Extension Development Kit) instead.

Future-proof extractors and loaders, with less code

On average, developers tell us that they write about 70% less code by using the SDK, which makes learning the SDK a great investment. Furthermore, as new features and capabilities are added to the SDK, your taps and targets can always take advantage of the latest capabilities and bug fixes, simply by updating your SDK dependency to the latest version.

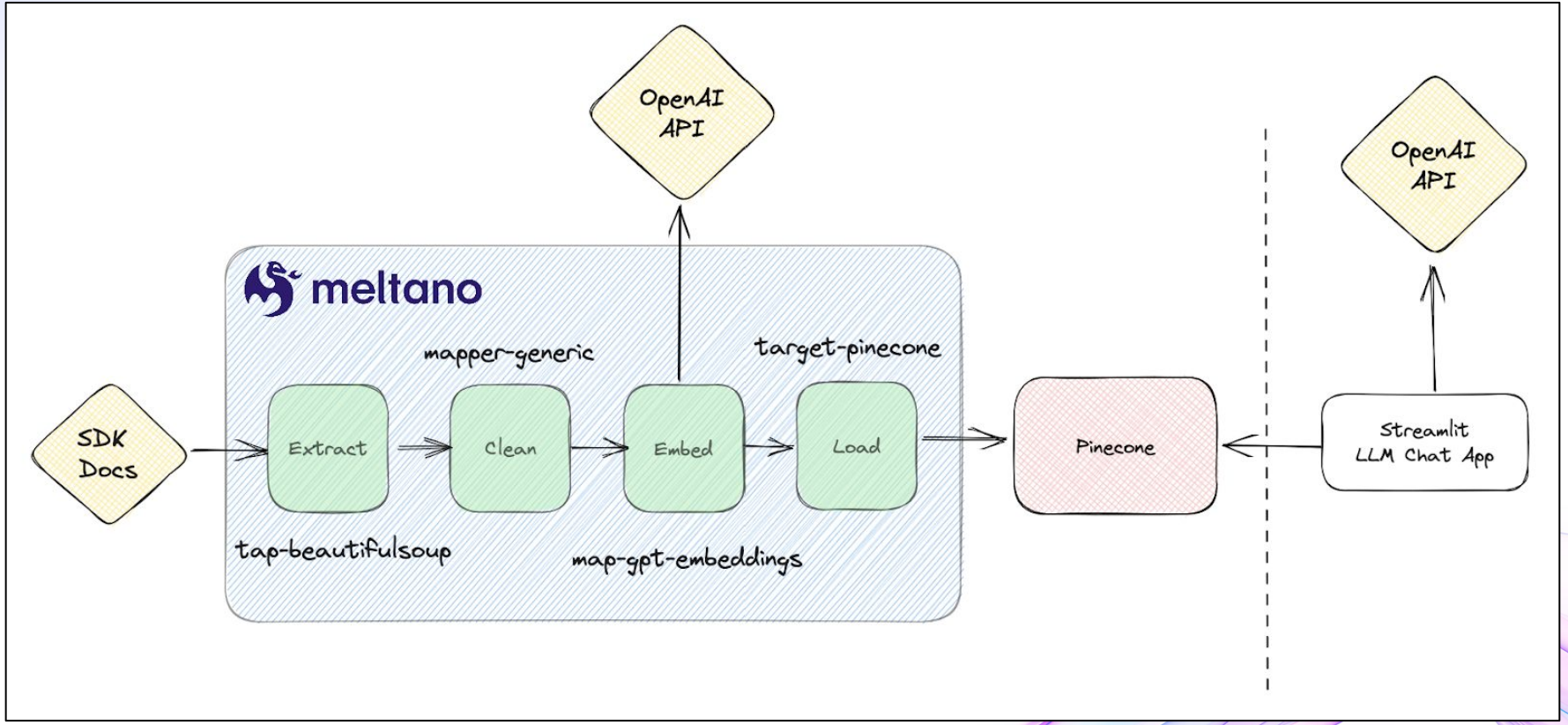
Built by Meltano and the Singer Community

The SDK is built with love by the Meltano core team and contributors, with contributions from developers across the Singer open source community. SDK-based plugins can easily be integrated with Meltano, but they can also be used in any data pipeline tool that supports the Singer Spec.

The SDK project is 100% open source, licensed under the Apache 2.0 permissive license. Please visit our code repo here in GitHub, where you can read the source code, log an issue or feature request, and contribute back to the SDK. We also regularly discuss SDK topics in our Meltano Slack community, within the #singer-tap-development and #singer-target-development Slack channels.

Index and Search

Implementation



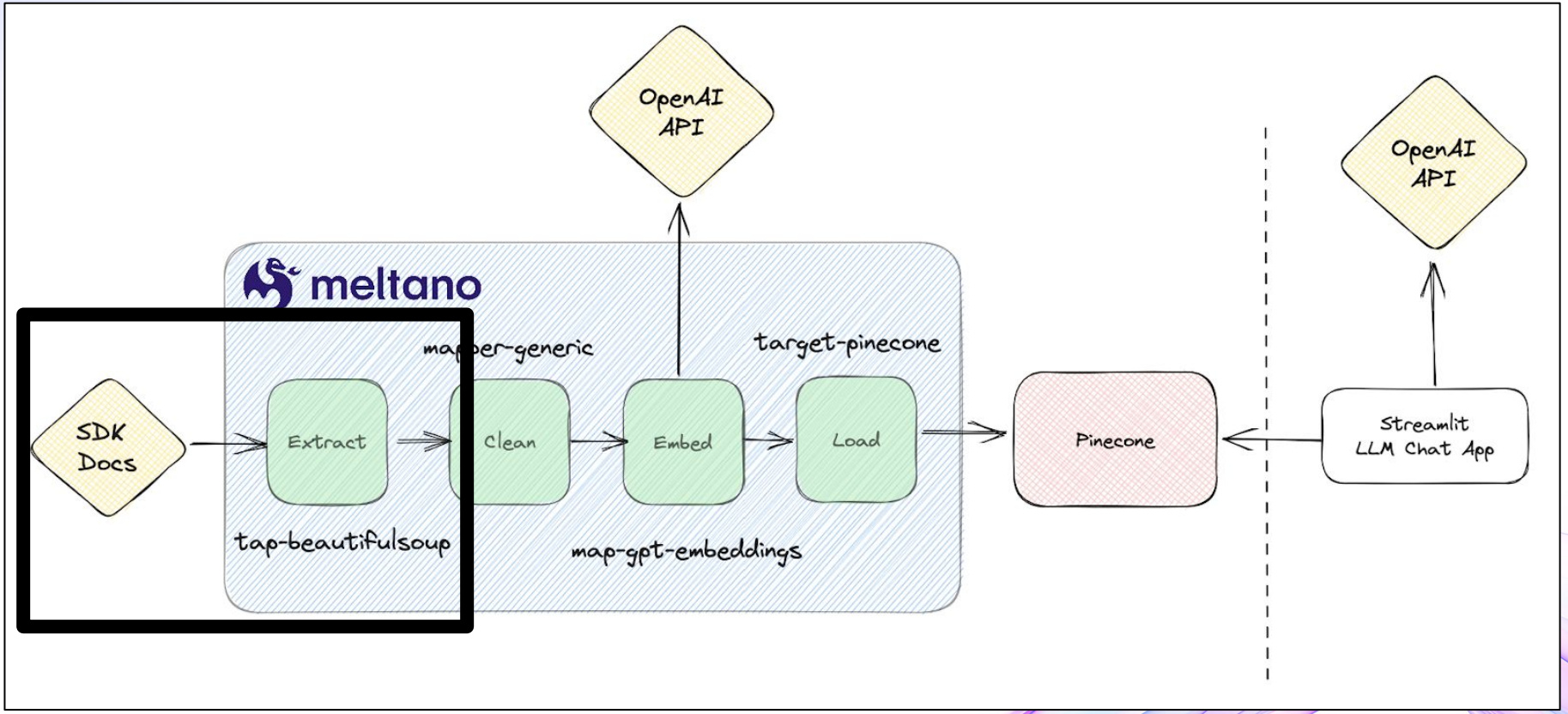
What do LLMs know about the Meltano Singer SDK? 🤔

Not much 😢


IS SLEEPING
AROUND
What you just said


MOTIVE
PAIR
Okay, a simple "wrong" would've done just fine.

Extract


OpenAI API
OpenAI API
meltano
mapper-generic
target-pinecone
SDK Docs
Extract
Clean
Embed
Load
Pinecone
Streamlit LLM Chat App
tap-beautifulsoup
map-gpt-embeddings

Extract

meltano.yml

```yaml
- name: tap-beautifulsoup
    variant: meltanolabs
    pip_url: git+https://github.com/MeltanoLabs/tap-beautifulsoup.git@v0.1.0
    config:
      source_name: sdk-docs
      site_url: https://sdk.meltano.com/en/latest/
      output_folder: output
      parser: html.parser
      download_recursively: true
      find_all_kwargs:
        attrs:
          role: main
```

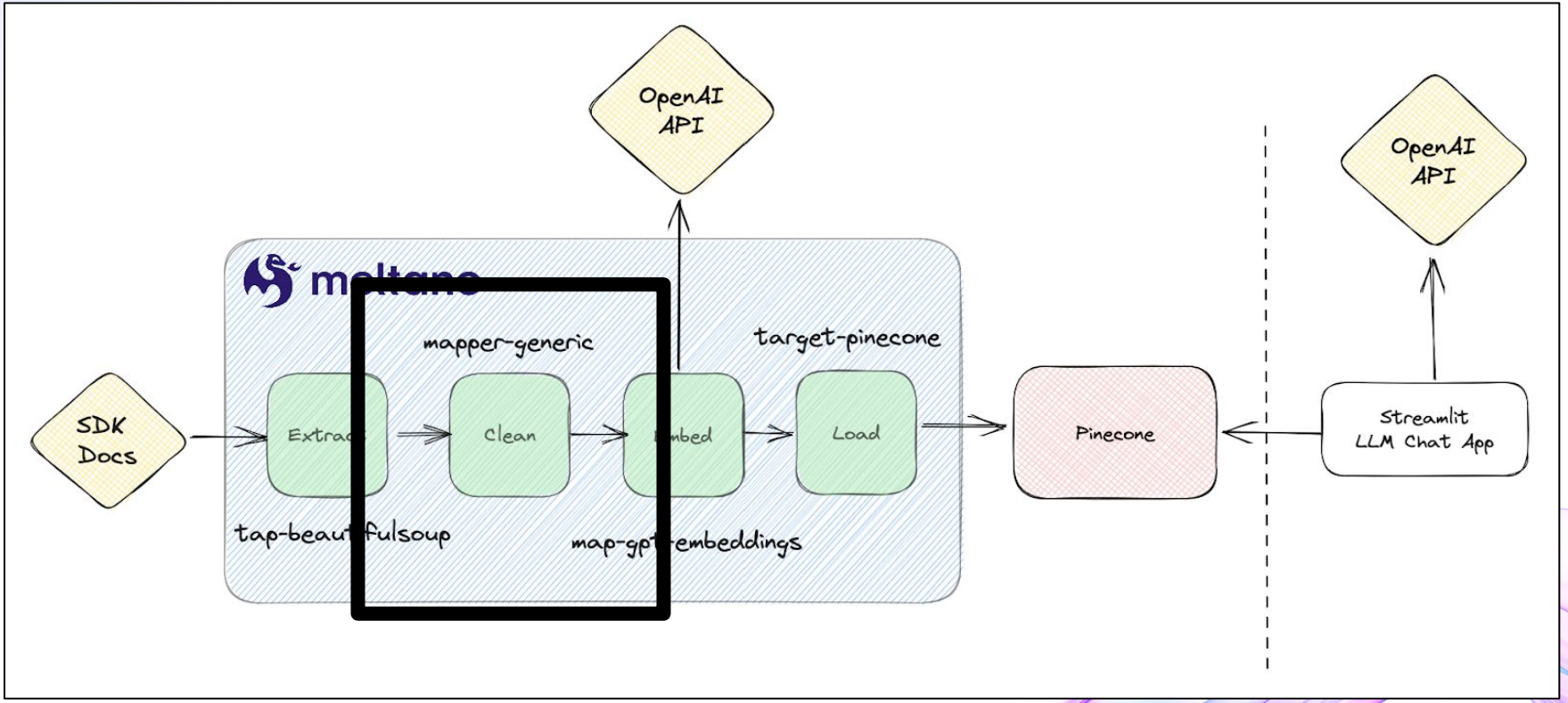
log

```
2023-08-17T15:28:57.408509Z [info     ] Environment 'dev' is active

2023-08-17 11:28:58,886 | INFO     | tap-beautifulsoup    | Beginning full_table
sync of 'page_content'...
{"type": "SCHEMA", "stream": "page_content", "schema": {"properties": {"source":
{"type": ["string", "null"]}, "page_content": {"description": "The page content.",
"type": ["string", "null"]}, "metadata": {"properties": {"source": {"type":
["string", "null"]}}, "type": ["object", "null"]}}, "type": "object"},
"key_properties": []}

{"type": "RECORD", "stream": "page_content", "record": {"source":
output/sdk.meltano.com/en/latest/typing.html", "page_content": "JSON Schema
elpers#\nClasses and functions to streamline…..[Trimmed Content]", "metadata":
{"source": "output/sdk.meltano.com/en/latest/typing.html"}}, "time_extracted":
"2023-08-17T15:29:38.975515+00:00"}
```

Clean


OpenAI API
OpenAI API
meltano
mapper-generic
target-pinecone
SDK Docs
Extract
Clean
Embed
Load
Pinecone
Streamlit LLM Chat App
tap-beautifulsoup
map-gpt-embeddings

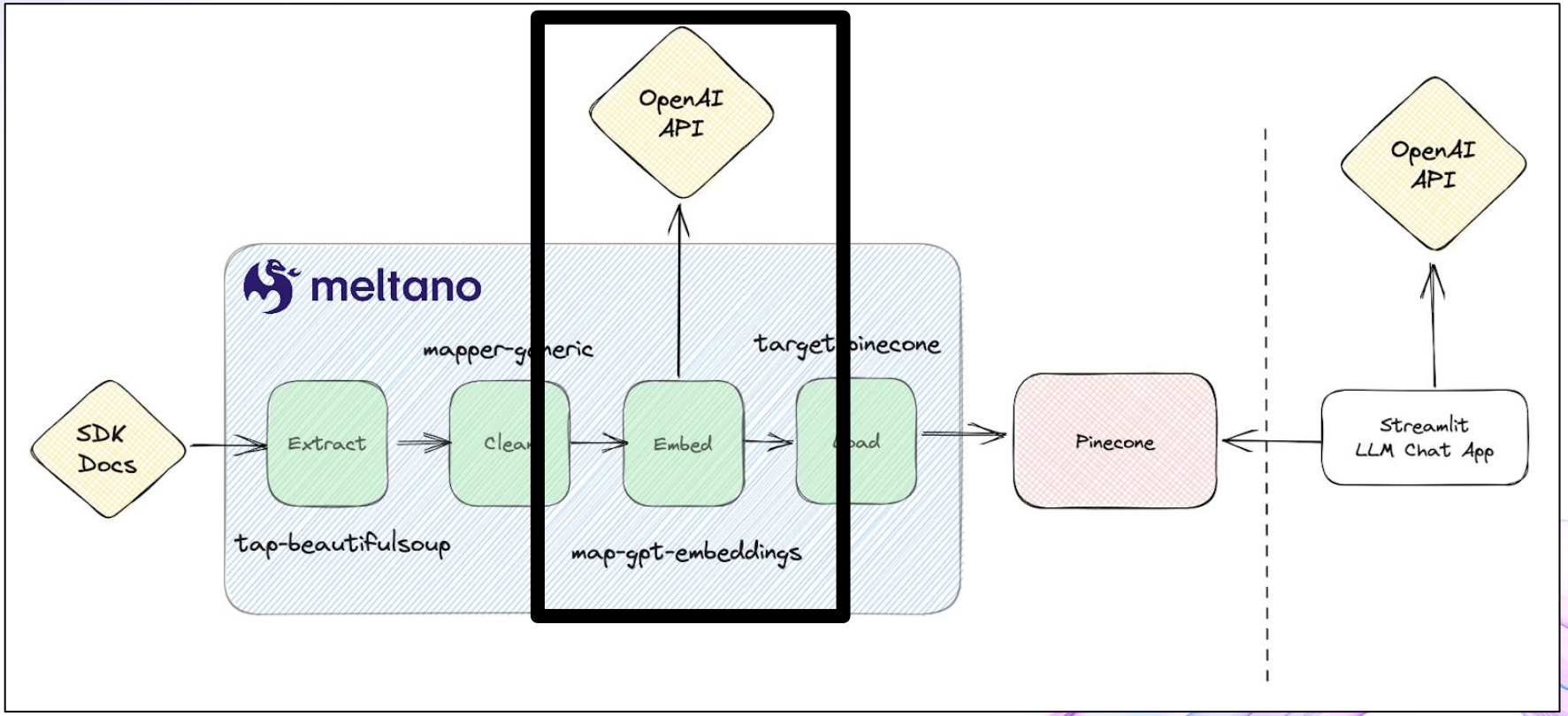
Clean

- Parsing
- Removing special characters
- Use external packages
- File formats (PDF, docs, etc.)

clean_text.py

```python
def map_record_message(self, message_dict: dict) -> t.Iterable[Message]:
    page_content = message_dict["record"]["page_content"]
    text_nl = " ".join(page_content.split("\n"))
    text_spaces = " ".join(text_nl.split())
    message_dict["record"]["page_content"] = text_spaces
    return message_dict
```

Embedding


OpenAI
API
meltano
mapper-generic
target-pinecone
SDK
Docs
Extract
Clean
Embed
Load
Pinecone
OpenAI
API
Streamlit
LLM Chat App
tap-beautifulsoup
map-gpt-embeddings

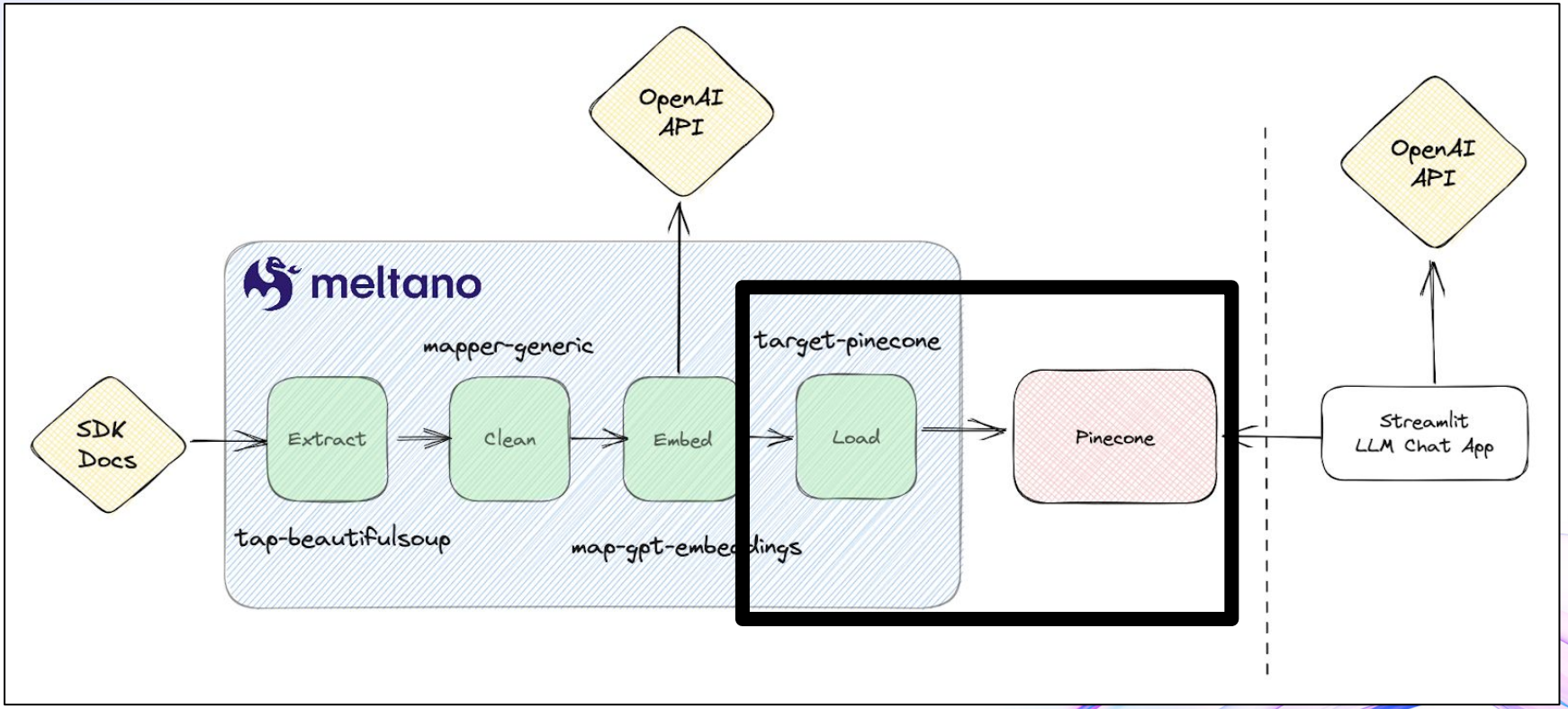
Mapper Embedding

- Chunks out text blobs
- Uses OpenAI API to embed
- Adds embedding to stream data
- Tap + Mapper combo
  - Re: Common EL Challenges 💡
  - Rate limiting, Auth, Retry, ...

meltano.yml

```yaml
- name: map-gpt-embeddings
  variant: meltanolabs
  pip_url: git+https://github.com/MeltanoLabs/map-gpt-embeddings.git
  mappings:
  - name: add-embeddings
    config:
      document_text_property: page_content
      document_metadata_property: metadata
```

Load


OpenAI API
OpenAI API
meltano
mapper-generic
target-pinecone
SDK Docs
Extract
Clean
Embed
Load
Pinecone
Streamlit LLM Chat App
tap-beautifulsoup
map-gpt-embeddings

Load

meltano.yml

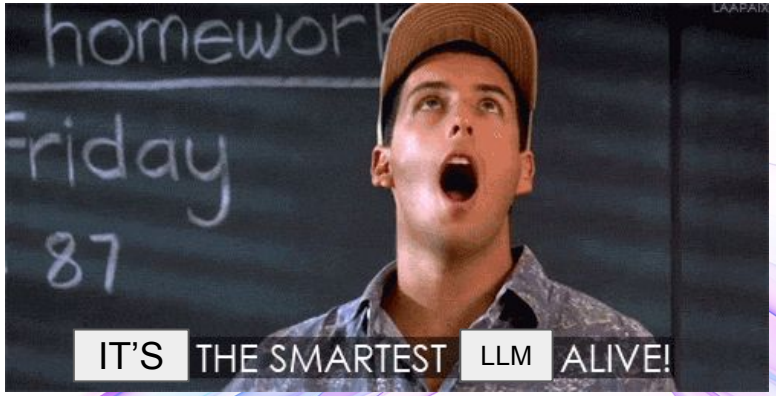
```yaml
- name: target-pinecone
  variant: meltanolabs
  config:
    index_name: target-pinecone-index
    environment: asia-southeast1-gcp-free
    document_text_property: page_content
    embeddings_property: embeddings
    metadata_property: metadata
    pinecone_metadata_text_key: text
    load_method: overwrite
```

- Embeddings
- Metadata
- Document text

Pinecone

🔄 Let’s try again..

Correct ✅



POC Next Steps

- Checkpointing each step
- Incremental upserts
- More target and model support
- Open AI improvements i.e. 16k context window

Takeaways 💡

- Singer Spec is alive and well
- Meltano solves Singer challenges
- Demystified the LLM app space
- Consider ***existing robust EL tools*** vs new built in EL features of LLM tools
- Applying DE principles

Join the Community

- Check out the blog and GitHub repo
- Join Meltano community
- Reach out if you're interested in Arch

Q&A