MEP 3 -- Merge PyMilvus and PyMilvus-ORM repository

Current state: Accepted

ISSUE: #597 Why we separate pymilvus and pymilvus-orm

PRs:

Keywords: sdk, pymilvus, pymilvus-orm

Released:

Summary

Currently, we have 2 python SDKs for Milvus, PyMilvus and ORM (short for PyMilvus-ORM). Both of them have a unique repository on GitHub and a unique package on PYPI.

This proposal is about

- Merging 2 repositories of PyMilvus and ORMdeciding which repository to keep.
- Deciding which set of APIs to keep and which package name to keep, `pymilvus` or `pymilvus-orm`
- The details on how to merge these 2 repositories.

After the tech meeting, we reached a consensus on:

- Keeping which repo: PyMilvus
- Keeping which set of API:
  - step1: both, and we mark PyMilvus API as `deprecated`
  - step2: pymilvus-orm
- Keeping which package name: pymilvus

Motivation

1. **The release is complicated**: ORM requires PyMilvus, thus we have to release PyMilvus first and then release ORM.
2. **Features and bug fixes are done only if both repositories are updated:** A bug fix on PyMilvus needs an update on ORM.
3. **Complexity on maintaining:** We have to maintain 2 repositories, 2 sets of CI pipelines, 2 GitHub actions.

Design Details

A. Which repository to keep?

GitHub repo(2021.7.15)	PyMilvus	PyMilvus-ORM
Stars	264	9
Forks	110	23
Issues(not closed)	25	6
Contributors	24	18
Used by(repositories)	106	5
Used by(packages)	21	1

Obviously (of course) PyMilvus repository is more valuable.

Plan A1 (Recommended): Keep the PyMilvus repository.

> **Pros:** The PyMilvus repository is more valuable.
>
> **Cons:** Not see any.

Plan A2: Keep the PyMilvus-ORM repository.

> **Pros:** Not see any.
>
> **Cons:** Lose all the stars and forks of the PyMilvus repository.

B. Which set of APIs to keep and which package name to keep?

Plan B1: Keep PyMilvus APIs and the ORM APIs

Pros:

1 More APIs for users to choose from.

2 Easier to merge 2 repositories

Cons:

1 APIs have duplicate functionality.

2 Complexity on maintaining new features, debugging, bug fixes, and CI pipeline is not reduced.

3 ORM's APIs depend on PyMilvus's API.

4 No much difference to two repositories, except one package less.

Package Name: In this case, I prefer `pymilvus`.

1 It's more like an "enhanced" pymilvus.

Plan B2 (Recommended): Keep ORM APIs and remove PyMilvus APIs

Pros:

1 Reduce the complexity of maintaining the repository.

2 Removing PyMilvus APIs means the 2 repositories' codes can combine deeper, reducing unnecessary function calling and object transfer.

Cons:

1 Merging is more complicated and needs more time.

Package Name: In this case, one of `pymilvus-orm`. `pymilvus2`, `pymilvus`.

1 We keep ORM APIs, `pymilvus-orm` is more suitable to the APIs

2 Milvus 1.x users won't be confused, Milvus 2.0.0RC users won't be confused.

Plan B3 (Not Recommended): Keep PyMilvus APIs and remove ORM APIs

Cons: All the efforts on ORM are wasted.

C. How to merge 2 repositories?

Three steps for plan A1

Step 1: Prepare locally, write a fully functional script.

During step 1, feel free to make any changes on PyMilvus or PyMilvus-ORM repositories.

Basically, this's what the script will do:

a. Tidy commits in ORM

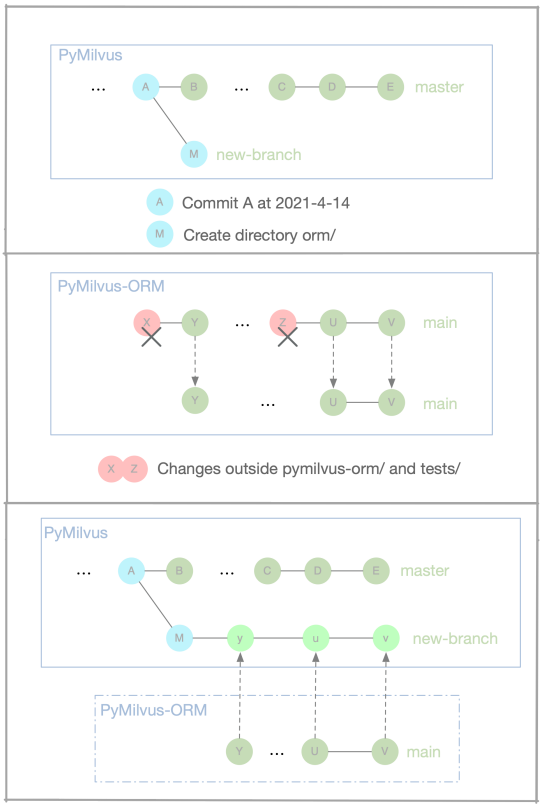
b. Check out a new branch of PyMilvus after one commit in 2021.4.14, and create a directory `orm/`.

c. Remove commits that are not in `pymilvus-orm` and `tests` of ORM repo.

d. Commit each valid commit of ORM with `author`, `author_date`, and `commiter_date` into orm/ directory.

e. PyMilvus rebase current `master` branch.

f. Make 2 APIs available.



Step 2: Merge

During step 2, no updates are allowed to both PyMilvus and ORM repositories.

After step 2, ORM repository is deprecated. There will be 2 sets of APIs in PyMilvus temporarily. **Further updates are determined by the results of topic B.**

Step 3: Correct behaviour of *ci, docs, Github actions, examples, tests, and changelog.*

Compatibility, Deprecation, and Migration Plan

Test Plan

Rejected Alternatives

References