MILVUS GUIDE

Table contents.

1. Install **1**

1.1 Requirements GPU-enable 1

1.2 Milvus server. 1

Pull milvus image 2

Download configuration file 2

Start milvus server 2

1.3 Milvus client with python. 2

Install pymilvus 2

Connect to milvus server 3

2. API reference **3**

Create collection 3

Create partition 3

Insert entities 4

Search Entities by Vector Similarity 4

Search Entities filtered by fields 5

Delete entities by id 6

Delete a partition 6

Delete a collection 6

1. Install

1.1 Requirements GPU-enable

- Docker 19.03 or higher
- Nvidia driver 418 or higher
- Nvidia container toolkit.

1.2 Milvus server.

- Pull milvus image

```
$ sudo docker pull milvusdb/milvus:0.11.0-gpu-d101620-4c44c0
```

- Download configuration file

```
$ mkdir -p /home/$USER/milvus/conf
$ cd /home/$USER/milvus/conf
$ wget https://raw.githubusercontent.com/milvus-io/milvus/0.11.0/core/conf/demo/milvus.yaml
```

After you downloaded the configuration file, you must set `enable` to `true` in `gpu` section of milvus.yaml.

- Start milvus server

```
$ sudo docker run -d --name milvus_gpu_0.11.0 --gpus all \
-p 19530:19530 \
-p 19121:19121 \
-v /home/$USER/milvus/db:/var/lib/milvus/db \
-v /home/$USER/milvus/conf:/var/lib/milvus/conf \
-v /home/$USER/milvus/logs:/var/lib/milvus/logs \
-v /home/$USER/milvus/wal:/var/lib/milvus/wal \
milvusdb/milvus:0.11.0-gpu-d101620-4c44c0
```

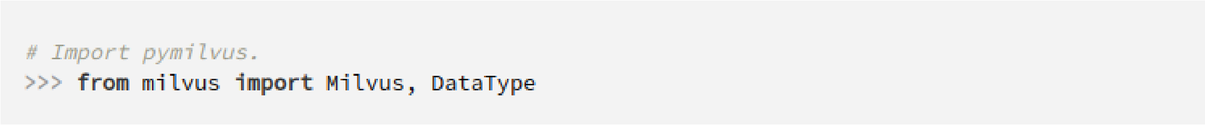
REF: https://www.milvus.io/docs/v0.11.0/milvus_docker-gpu.md

1.3 Milvus client with python.

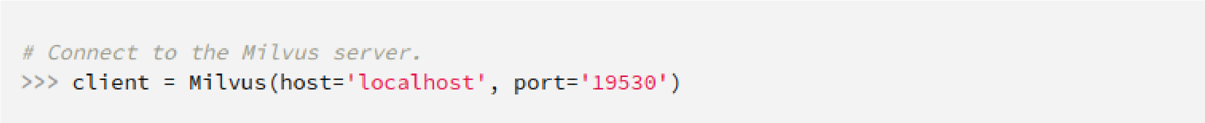
- Install pymilvus

```
# Install Milvus Python SDK
$ pip3 install pymilvus==0.3.0
```

- Connect to milvus server



```
# Import pymilvus.
>>> from milvus import Milvus, DataType
```

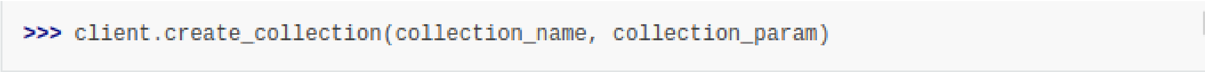


```
# Connect to the Milvus server.
>>> client = Milvus(host='localhost', port='19530')
```

REF: https://www.milvus.io/docs/v0.11.0/example_code.md
https://www.milvus.io/docs/v0.11.0/connect_milvus.md#Python

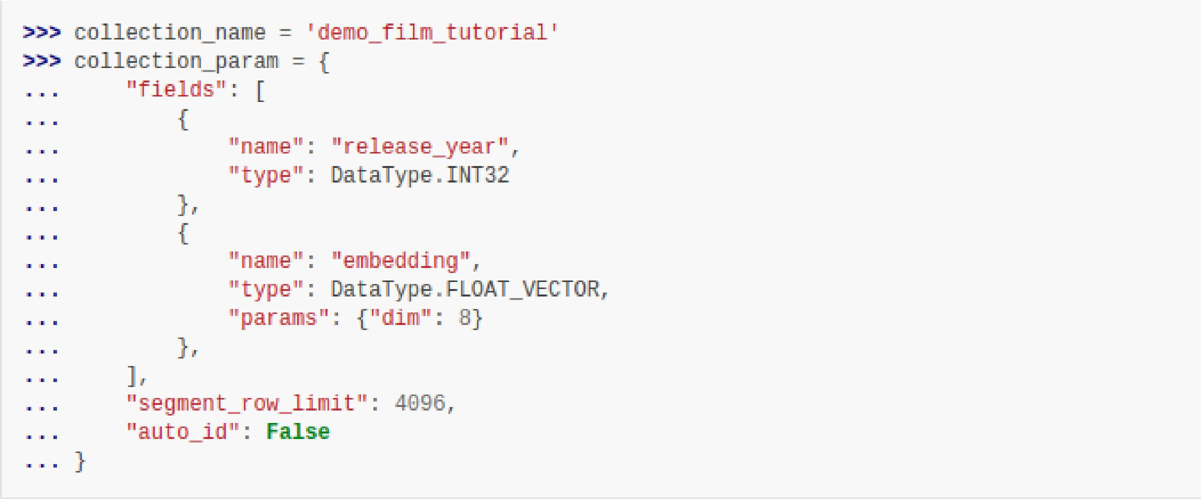
2. API reference

- Create collection



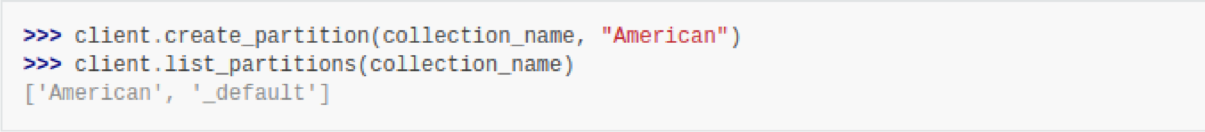
```
>>> client.create_collection(collection_name, collection_param)
```

Example :



```
>>> collection_name = 'demo_film_tutorial'
>>> collection_param = {
...     "fields": [
...         {
...             "name": "release_year",
...             "type": DataType.INT32
...         },
...         {
...             "name": "embedding",
...             "type": DataType.FLOAT_VECTOR,
...             "params": {"dim": 8}
...         },
...     ],
...     "segment_row_limit": 4096,
...     "auto_id": False
... }
```

- Create partition



```
>>> client.create_partition(collection_name, "American")
>>> client.list_partitions(collection_name)
['American', '_default']
```

+ Insert entities

```
>>> client.insert(collection_name, hybrid_entities, ids, partition_tag="American")
[1, 2, 3]
```

Example:

```
>>> import random
>>> The_Lord_of_the_Rings = [
...     {
...         "id": 1,
...         "title": "The_Fellowship_of_the_Ring",
...         "release_year": 2001,
...         "embedding": [random.random() for _ in range(8)]
...     },
...     {
...         "id": 2,
...         "title": "The_Two_Towers",
...         "release_year": 2002,
...         "embedding": [random.random() for _ in range(8)]
...     },
...     {
...         "id": 3,
...         "title": "The_Return_of_the_King",
...         "release_year": 2003,
...         "embedding": [random.random() for _ in range(8)]
...     }
... ]
```

```
>>> ids = [k.get("id") for k in The_Lord_of_the_Rings]
>>> release_years = [k.get("release_year") for k in The_Lord_of_the_Rings]
>>> embeddings = [k.get("embedding") for k in The_Lord_of_the_Rings]
```

```
>>> hybrid_entities = [
...     # Milvus doesn't support string type yet,
...     # so we cannot insert "title".
...     {
...         "name": "release_year",
...         "values": release_years,
...         "type": DataType.INT32
...     },
...     {
...         "name": "embedding",
...         "values": embeddings,
...         "type": DataType.FLOAT_VECTOR
...     },
... ]
```

+ Search Entities by Vector Similarity

```
>>> results = client.search(collection_name, dsl, fields=["release_year"])
```

Example:

```
>>> film_A = {
...     "title": "random_title",
...     "release_year": 2002,
...     "embedding": [random.random() for _ in range(8)]
... }
```

```
>>> dsl = {
...     "bool": {
...         "must": [
...             {
...                 "vector": {
...                     "embedding": {
...                         "topk": 2,
...                         "query": [film_A.get("embedding")],
...                         "metric_type": "L2"
...                     }
...                 }
...             }
...         ]
...     }
... }
```

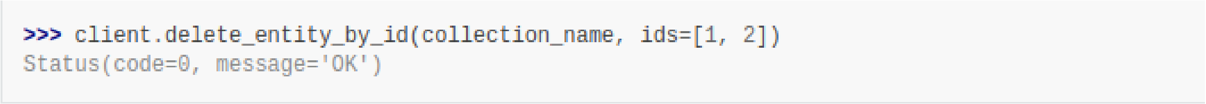
+ Search Entities filtered by fields

```
>>> results = client.search(collection_name, dsl_hybrid, fields=["release_year"])
>>> len(results[0])
1
```

Example:

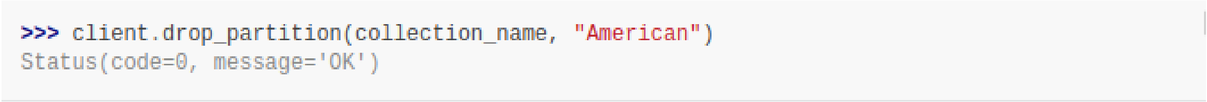
```
>>> dsl_hybrid = {
...     "bool": {
...         "must": [
...             {
...                 "term": {"release_year": [film_A.get("release_year")]}
...             },
...             {
...                 "vector": {
...                     "embedding": {
...                         "topk": 2,
...                         "query": [film_A.get("embedding")],
...                         "metric_type": "L2"
...                     }
...                 }
...             }
...         ]
...     }
... }
```

- Delete entities by id



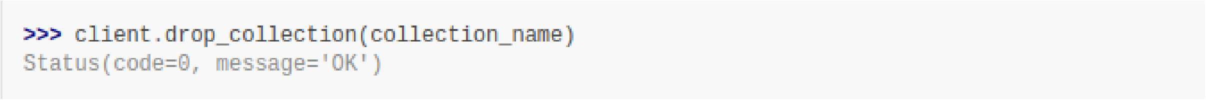
```
>>> client.delete_entity_by_id(collection_name, ids=[1, 2])
Status(code=0, message='OK')
```

- Delete a partition



```
>>> client.drop_partition(collection_name, "American")
Status(code=0, message='OK')
```

- Delete a collection



```
>>> client.drop_collection(collection_name)
Status(code=0, message='OK')
```

REF: https://pymilvus.readthedocs.io/en/0.3.0/tutorial.html