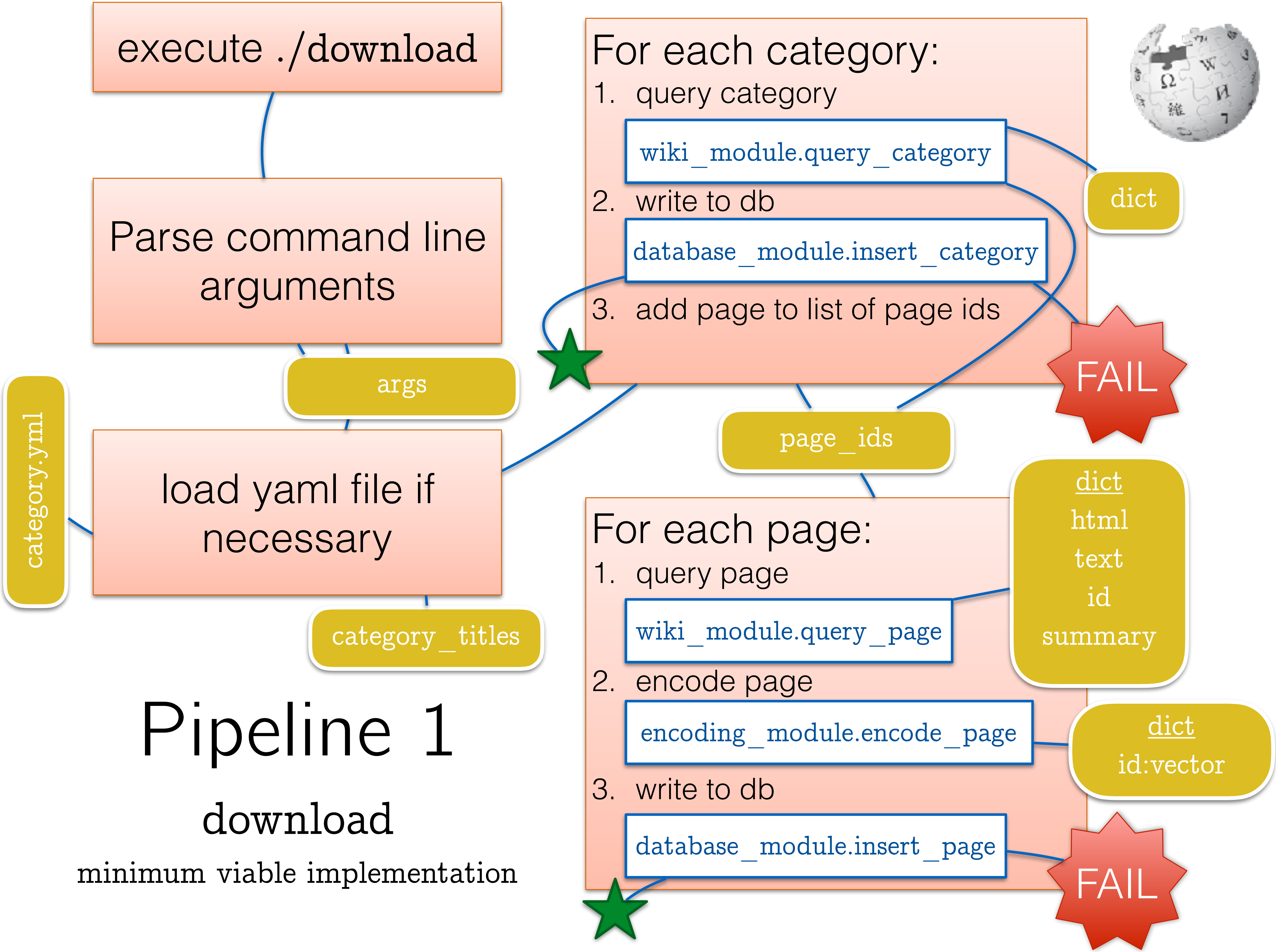
Pipeline 1

download

minimum viable implementation



execute ./download

Parse command line arguments

args

load yaml file if necessary

category.yml

category_titles

For each category:

1. query category

   wiki_module.query_category

2. write to db

   database_module.insert_category

3. add page to list of page ids

dict

FAIL

page_ids

For each page:

1. query page

   wiki_module.query_page

2. encode page

   encoding_module.encode_page

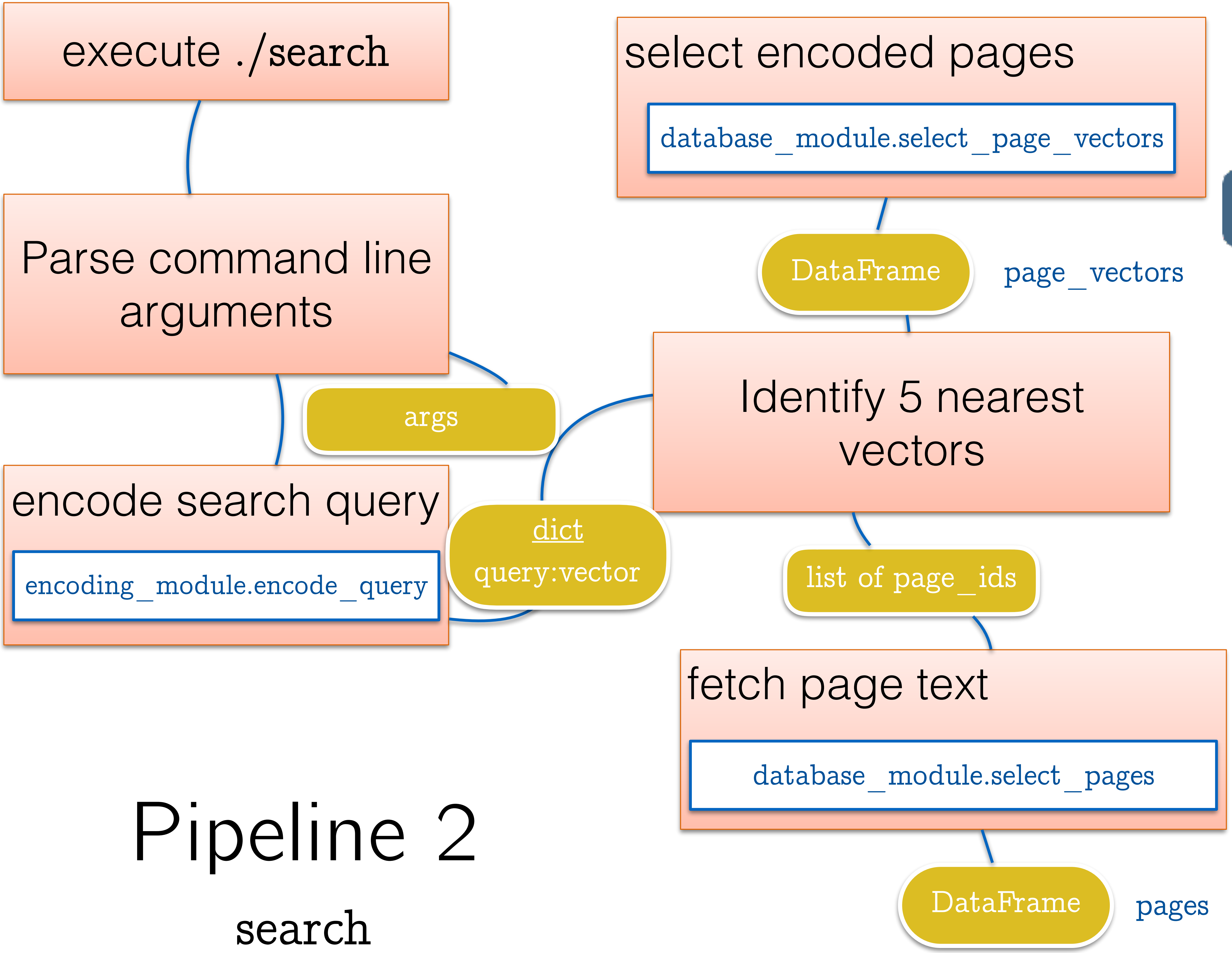
3. write to db

   database_module.insert_page

dict
html
text
id
summary

dict
id:vector

FAIL



execute ./search

Parse command line arguments

args

encode search query

encoding_module.encode_query

dict

query:vector

select encoded pages

database_module.select_page_vectors

DataFrame

page_vectors

Identify 5 nearest vectors

list of page_ids

fetch page text

database_module.select_pages

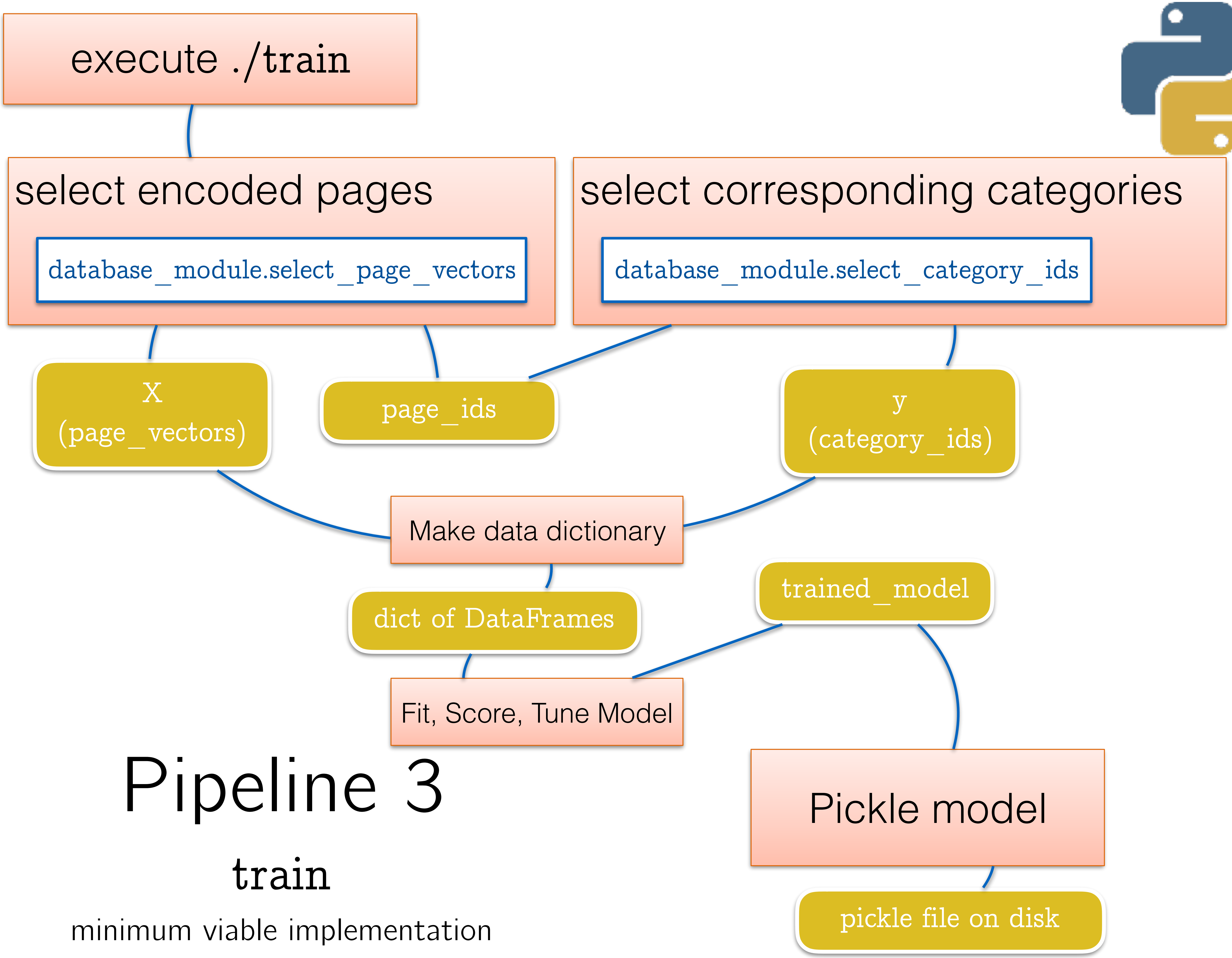
DataFrame

pages

Pipeline 2

search

minimum viable implementation



execute ./train

select encoded pages

database_module.select_page_vectors

select corresponding categories

database_module.select_category_ids

X
(page_vectors)

page_ids

y
(category_ids)

Make data dictionary

dict of DataFrames

trained_model

Fit, Score, Tune Model

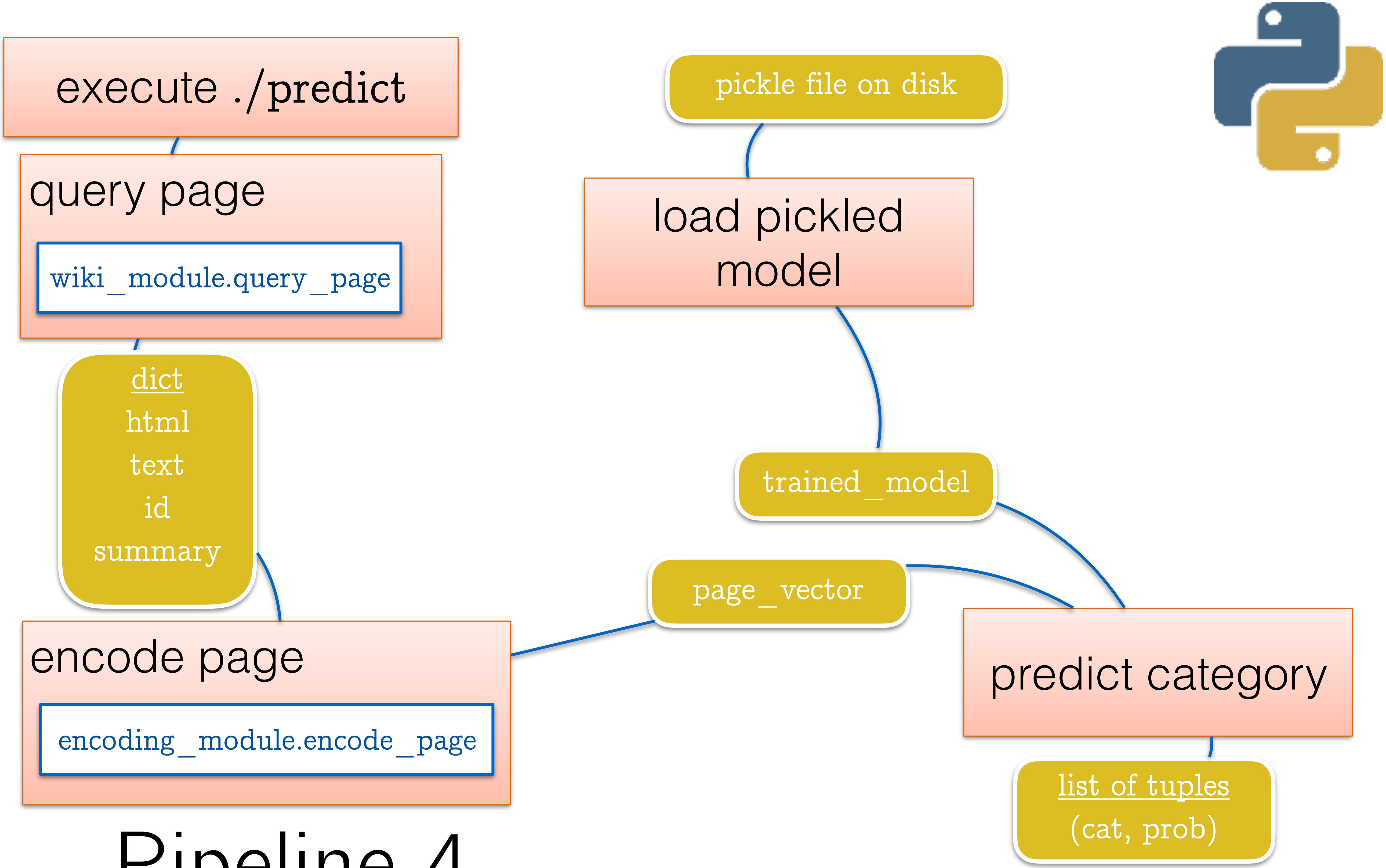
Pickle model

pickle file on disk

Pipeline 3

train

minimum viable implementation



execute ./predict

query page

wiki_module.query_page

dict
html
text
id
summary

encode page

encoding_module.encode_page

page_vector

pickle file on disk

load pickled model

trained_model

predict category

list of tuples
(cat, prob)

Pipeline 4

predict

minimum viable implementation