Create MTG Trading Card Database By API

Practical Exam

Goal

magicthegathering.io provides up-to-date information regarding all MTG trading cards available:

- https://docs.magicthegathering.io/
- E.g. https://api.magicthegathering.io/v1/cards/4711


```
{
  "card":{
    "name":"Manta Riders",
    "manaCost":"{U}",
    "cmc":1,
    "colors":[
      "Blue"
    ],
    "colorIdentity":[
      "U"
    ],
    "type":"Creature — Merfolk",
    "types":[
      "Creature"
    ],
    "subtypes":[
      "Merfolk"
    ],
    "rarity":"Common",
    "set":"TMP",
    "setName":"Tempest",
    "text":"{U}: Manta Riders gains flying until end of turn.",
    "flavor":"\"Water is firmament to the finned.\"\n—Oracle en-Vec",
    "artist":"Kaja Foglio",
    "power":"1",
    "toughness":"1",
    "layout":"normal",
    "multiverseid":4711,
    "imageUrl":"http://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=4711&type=
card",
  […]
```

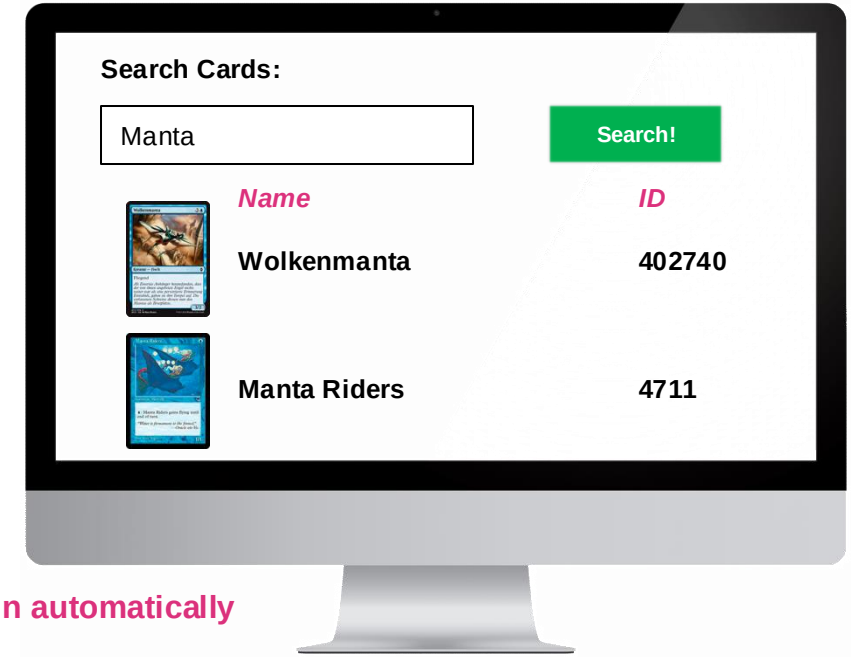
Goal

We want to make use of this data to build a searchable database of all MTG trading cards.

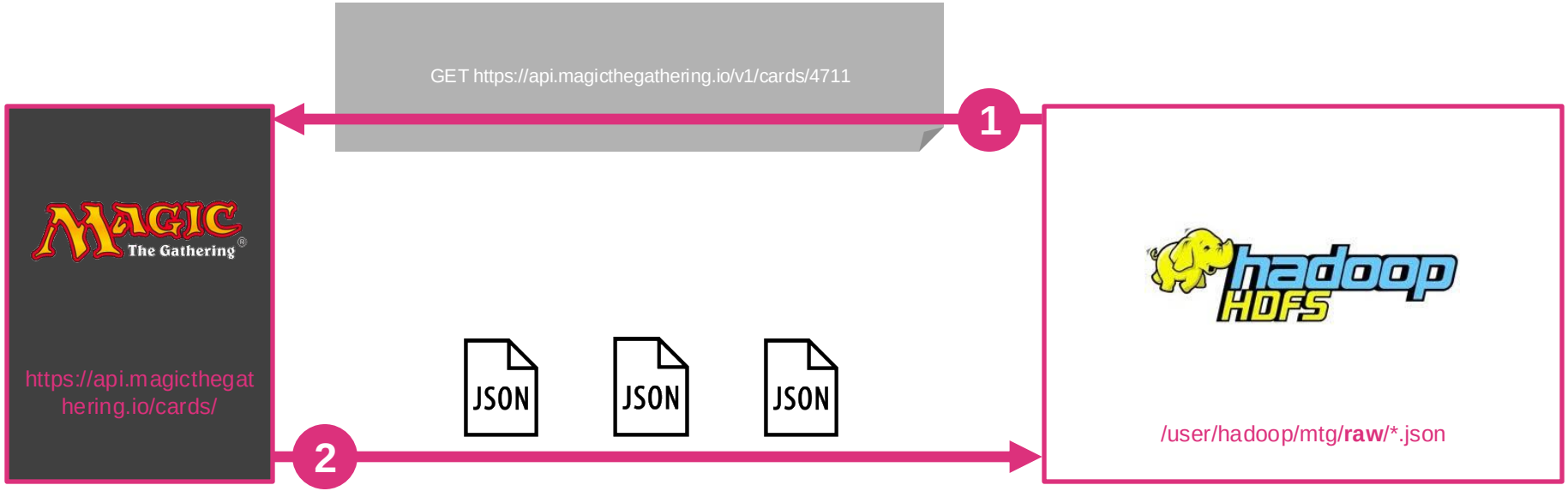
Workflow:

- **Gather data** from api.magicthegathering.io
- **Save raw data** (*JSON files*) to HDFS
- **Optimize**, **reduce** and **clean raw data** and save it to **final** directory on HDFS
- **Export** MTG data **to end-user database** (e.g. MySQL, MongoDB…)
- Provide a simple **HTML Frontend** which is able to:
  - read from end-user database
  - process user input (card name, text or artist)
  - **display search results**

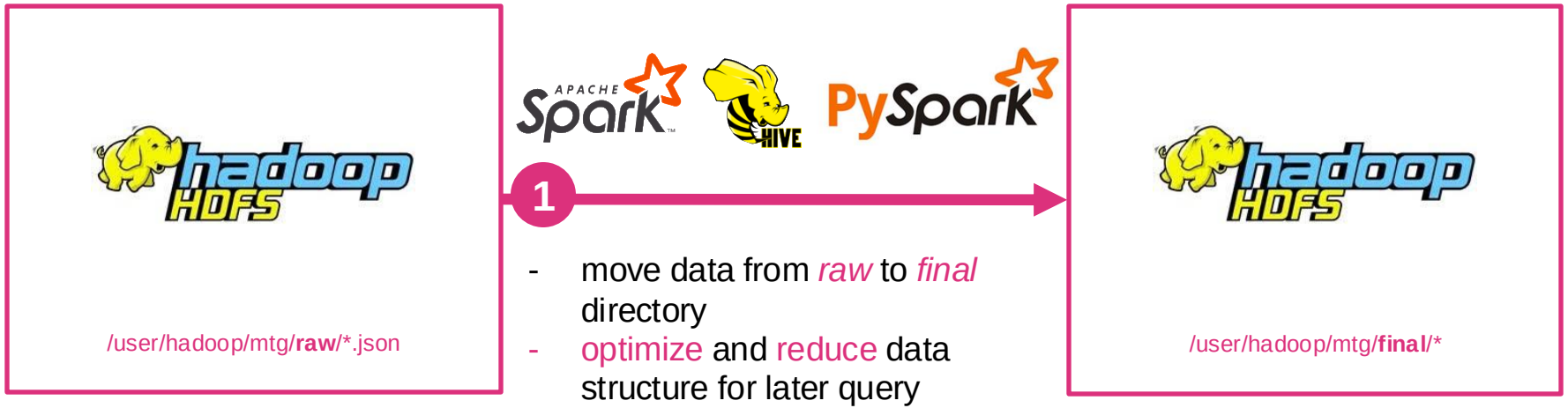
- The whole data workflow **must be implemented** within an ETL **workflow tool** (e.g. Pentaho Data Integration or Airflow) and **run automatically**



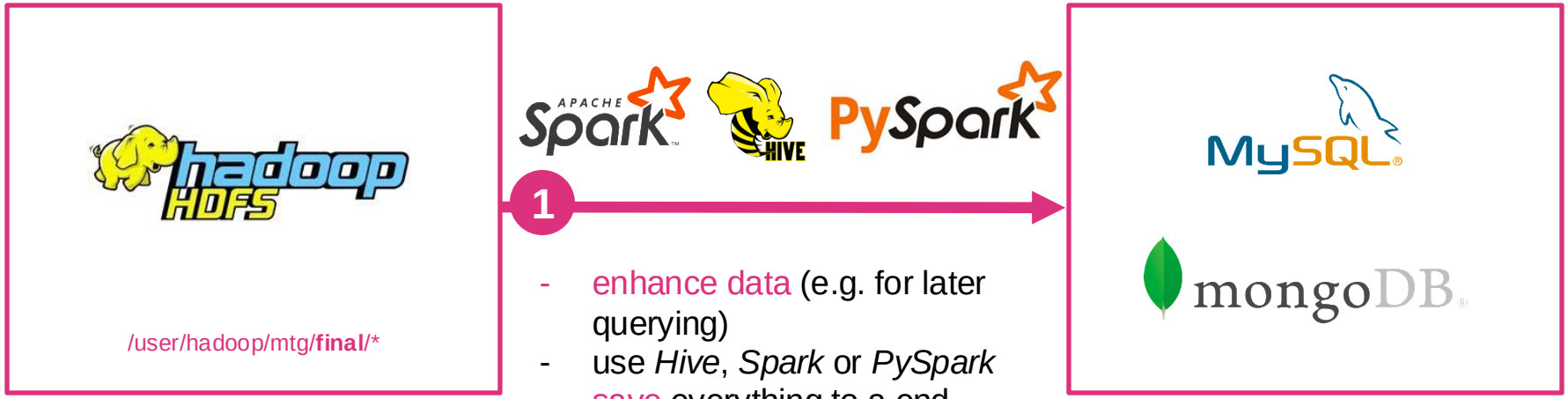
Dataflow: 1. Get MTG Data



Dataflow: 2. Raw To Final Transfer



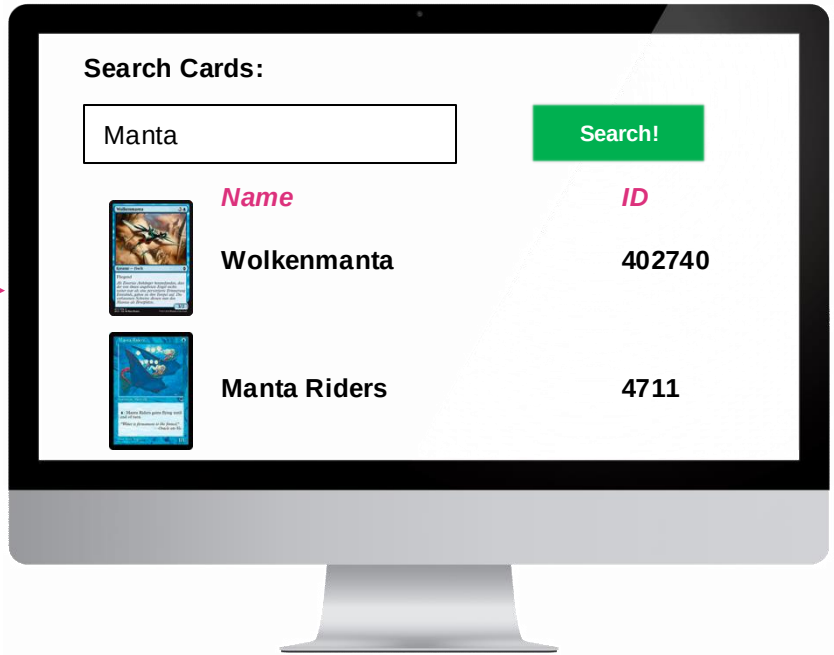
Dataflow: **3. Enhance Data And Save Results**



- enhance data (e.g. for later querying)
- use *Hive*, *Spark* or *PySpark*
- save everything to a end-user database (e.g. *MySQL*, *MongoDB*)

Dataflow: 4. Provide Simple Web Interface

1



- Provide a simple **HTML Frontend** which is able to:
  - read from end-user database
  - process user input (card name, text or artist)
  - **display search results**