

Datensammlung und Datenstruktur

Dokumentation Big Data Projekt

des Studienganges Informatik

an der Dualen Hochschule Baden-Württemberg Stuttgart

von

Luca Holder

22.11.2025

Bearbeitungszeitraum	04.11.2024-22.11.2024
Matrikelnummern, Kurs	5243642, TINF22C
Gutachter*in der Dualen Hochschule	Marcel Mittelstädt

Inhaltsverzeichnis

Inhaltsverzeichnis	I
1 Datensammlung	2
1.1 Datenstruktur	2
1.2 Datenabfrage	3
1.3 Automatisierung der Datensammlung	4
Literaturverzeichnis	5

1 Datensammlung

Dieses Kapitel erfüllt die erste Aufgabe, indem es die Daten sammelt und die Struktur sowie den Inhalt der Daten analysiert.

Die benötigten Daten lassen sich mit der API „Magic: The Gathering API“ abrufen, die Informationen über die Karten und Sets von dem Fantasy-Kartenspiel „Magic: The Gathering“. (vgl. [1])

1.1 Datenstruktur

Zum Verstehen der Daten wurde mit einem Python Skript eine Karte über die API abgerufen und ausgegeben. Das Skript ist in Anhang unter „A.1 Skript Beispielsabfrage einer Karte“ vorzufinden.

```
{
  "card": {
    "name": "Ankh of Mishra",
    "manaCost": "{2}",
    "cmc": 2.0,
    "type": "Artifact",
    "types": [
      "Artifact"
    ],
    "rarity": "Rare",
    "set": "LEA",
    "setName": "Limited Edition Alpha",
    "text": "Whenever a land enters the battlefield, Ankh of Mishra deals 2 damage to that land's controller.",
    "artist": "Amy Weber",
    "number": "230",
    "layout": "normal",
    "multiverseid": "1",
    "imageUrl": "http://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=1&type=card",
    "rulings": [
      {
        "date": "2004-10-04",
        "text": "It determines the land's controller at the time the ability resolves. If the land leaves the battlefield before the ability resolves, the land's last controller before it left is used."
      },
      {
        "date": "2004-10-04",
        "text": "This triggers on any land entering the battlefield. This includes playing a land or putting a land onto the battlefield using a spell or ability."
      }
    ],
    "printings": [
      "2ED",
      "30A",
      "3ED",
      "4BB",
      "4ED",
      "5ED",
      "6ED",
      "CED",
      "CEI",
      "FBB",
      "LEA",
      "LEB",
      "ME1",

```

```

    "SUM",
    "VMA"
  ],
  "originalText": "Ankh does 2 damage to anyone who puts a new land into play.",
  "originalType": "Continuous Artifact",
  "legalities": [
    {
      "format": "Commander",
      "legality": "Legal"
    },
    {
      "format": "Duel",
      "legality": "Legal"
    },
    {
      "format": "Legacy",
      "legality": "Legal"
    },
    {
      "format": "Oathbreaker",
      "legality": "Legal"
    },
    {
      "format": "Oldschool",
      "legality": "Legal"
    },
    {
      "format": "Predh",
      "legality": "Legal"
    },
    {
      "format": "Premodern",
      "legality": "Legal"
    },
    {
      "format": "Vintage",
      "legality": "Legal"
    }
  ],
  "id": "33b9ca30-0296-52b7-a8e2-7d4715404b0d"
}

```

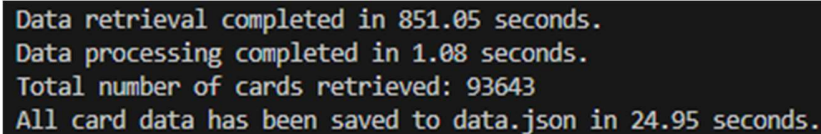
Abbildung 1: eigene Abbildung, Datenstruktur der Karten

Wie die Abbildung zeigt, liefert die API die Daten im JSON-Format. Sie liegen somit als Sem-strukturierte Daten vor, da das JSON-Format zwar Struktur bietet, nicht aber auf der Ebene von strukturierten Daten wie Tabellen. Dabei lassen sich die meisten Datensätze dennoch gut zu Tabellen formatieren. Z.B. kann jeder Karte ein Name und eine ID zugewiesen werden, dass sich gut in Tabellen darstellen lässt. Die Werte, die „rulings“, „printings“ und „legalities“ besitzen dabei mehrere Unterwerte.

1.2 Datenabfrage

Die Dokumentation der API weist auf SDKs hin, die die Datenabfragen vereinfachen sollen. Zudem ist die Datenabfrage über die API auf maximale 5000 Abfragen pro Stunde sowie auf maximal 100 Anfragen pro 100 Karten eingeschränkt wird. (vgl. [1])

Erstellt wird die Abfrage in Python, dabei wird das Python SDK verwendet, dass mit dem Befehl „pip install mtgsdk“ installiert werden kann.



```
Data retrieval completed in 851.05 seconds.  
Data processing completed in 1.08 seconds.  
Total number of cards retrieved: 93643  
All card data has been saved to data.json in 24.95 seconds.
```

Abbildung 2: Screenshot, Konsolenausgabe der Ausführungszeiten

Bei der Abfrage aller Karten (durchgeführt am 6. November 2024 auf der lokalen Maschine) betrug die Ausführungszeit für die Abfrage aller Karten 851,05 Sekunden, das ungefähr 14min und 19 Sekunden entspricht. Das Überführen der Daten in eine Liste dauerte 1,08 Sekunden, während das Speichern in einer JSON-Datei 24,95 Sekunden in Anspruch nahm. Mehrere Durchläufe zeigten, dass die Zeiten variieren, dennoch sollte der Prozess nach 15-20 Minuten durchgeführt sein. Die Datei hatte am Ende eine Größe von Size: 596 MB (625,281,818 bytes) und 93.643 Karten.

1.3 Automatisierung der Datensammlung

Die Datensammlung wird mit einem Python-Skript durchgeführt. Dieser Job läuft unter dem Namen collect_job_mtg und ruft alle Karten der API über die Python SDK auf und speichert diese als JSON-Datei ab. Anschließend wird über das Skript die Datei in den HDFS in das Verzeichnis „raw“ als JSON-Datei gespeichert, wie die Prüfungsleistung erfordert.

Literaturverzeichnis

- [1] „MTG API Docs“, Magic: The Gathering API Documentation. Zugegriffen: 5. November 2024. [Online]. Verfügbar unter: <https://docs.magicthegathering.io/>
- [2] T. S. Hukkeri, V. Kanoria, und J. Shetty, „A Study of Enterprise Data Lake Solutions“, Bd. 07, Nr. 05, 2020.
- [3] „docker-stacks/docs/using/running.md at main · jupyter/docker-stacks“, GitHub. Zugegriffen: 7. November 2024. [Online]. Verfügbar unter: <https://github.com/jupyter/docker-stacks/blob/main/docs/using/running.md>