Construção de dataset para histórico de *in-game currencies*

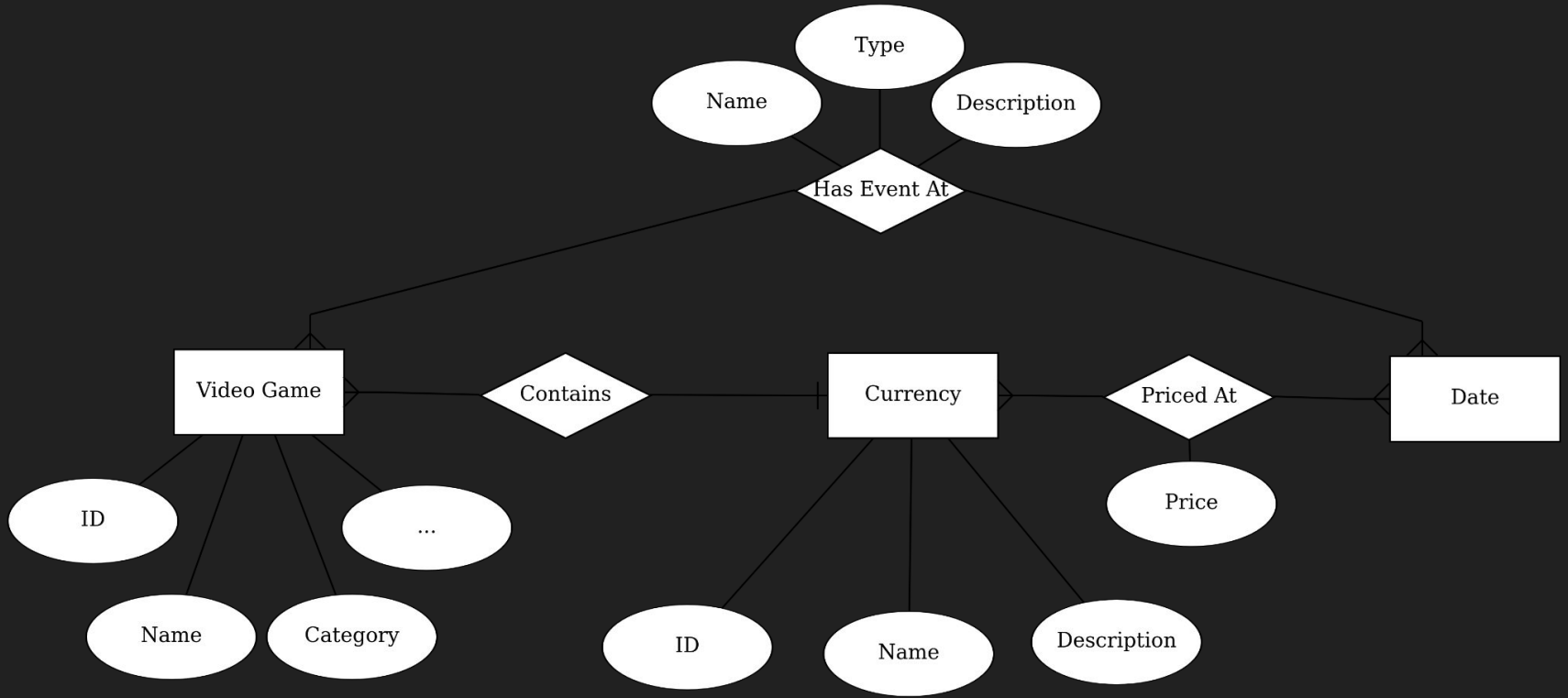
(Entrega Parcial)

Igor Gabriel Cavalcante de Carvalho Borges, RA 236773

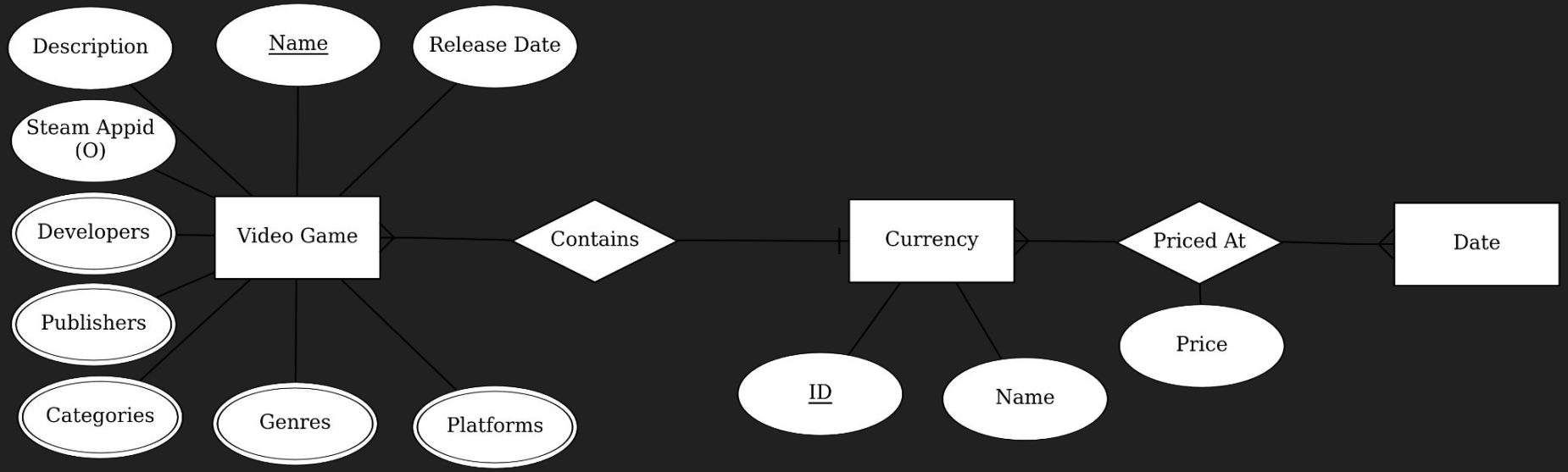
Resumo da Premissa

- *In-game currencies* são moedas virtuais bastante utilizadas dentro de jogos como meio de troca de bens virtuais, e que podem possuir valor em dinheiro real.
- De certa forma, elas são precursoras das atuais criptomoedas.
- Como existem a muito mais tempo, possuem uma quantidade maior de dados a serem explorados.
- Apesar disso, pesquisa no ramo é escassa e carece de fontes de dados compreensivas.
- **Proposta:** Um banco de dados contendo histórico de preços de moedas virtuais extraídos de diversas fontes, além de informações sobre seus respectivos jogos

Modelo Conceitual - Esboço Inicial



Modelo Conceitual - Esboço Atual



Modelos Lógicos

- Metadados de Video Games em formato JSON e Tabelas de Preços em formato CSV

```
game_info = {
    "name": "",
    "description": "",
    "steam": {"steam_game": False, "appid": 0},
    "developers": {},
    "publishers": {},
    "platforms": {},
    "categories": {},
    "genres": {},
    "release_date": "",
    "currencies": {}
}
```

Template utilizado para guardar as informações dos jogos (Modelo de documentos)

	date	price
0	2019-11-15	0.54512916
1	2019-11-16	0.54612894
2	2019-11-17	0.54773894
3	2019-11-18	0.53638545
4	2019-11-19	0.53478497
5	2019-11-20	0.52603063
6	2019-11-21	0.54710124
7	2019-11-22	0.56101775
8	2019-11-23	0.55097659
9	2019-11-24	0.54816689
10	2019-11-25	0.5280544
11	2019-11-26	0.52943581
12	2019-11-27	0.54993934
13	2019-11-28	0.55710786
14	2019-11-29	0.56868051
15	2019-11-30	0.5858149
16	2019-12-01	0.55394324
17	2019-12-02	0.54370801
18	2019-12-03	0.54212721
19	2019-12-04	0.55285276
20	2019-12-05	0.55023268

	date	price
0	2012-12-12	1.535
1	2012-12-13	1.786
2	2012-12-14	1.942
3	2012-12-15	2.027
4	2012-12-16	2.037
5	2012-12-17	2.065
6	2012-12-18	2.093
7	2012-12-19	2.091
8	2012-12-20	2.059
9	2012-12-21	1.905
10	2012-12-22	1.91
11	2012-12-23	1.925
12	2012-12-24	1.883
13	2012-12-25	2.043
14	2012-12-26	2.168
15	2012-12-27	2.161
16	2012-12-28	2.087
17	2012-12-29	2.135
18	2012-12-30	2.013
19	2012-12-31	1.931
20	2013-01-01	2.016

Tabelas de Preços (Modelo Relacional)

Fontes de Dados - Preços

backpack.tf

backpack.dt

Fontes de Dados - Metadados

Obtenção e Transformação de Dados

- Até o momento, somente por APIs
- Nenhum site necessitou de scraping*
- Porém, para resolver problemas de compatibilidade entre dados foram necessárias diversas conversões ao longo do caminho

Obtenção e Transformação de Dados

- Scripts para obter e converter automaticamente os dados

```python
import sys; sys.path.append("../util")
import requests
import json
from json_templates import game_info
from datetime import datetime
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

def get_steam_game_metadata(appid):
    url = f"https://store.steampowered.com/api/appdetails?appids={appid}&cc=US"
    req = requests.get(url).json()[str(appid)]
    if req["success"]:
        game = game_info.copy()
        data = req["data"]
        game["name"] = data["name"]
        game["description"] = data["short_description"]
        game["steam"]["steam_game"] = True
        game["steam"]["appid"] = data["steam_appid"]
        game["developers"] = data["developers"]
        game["publishers"] = data["publishers"]
        game["categories"] = data["categories"]
        game["platforms"] = data["platforms"]
        game["genres"] = data["genres"]
        game["release_date"] = datetime.strptime(data["release_date"]["date"], "%b %d, %Y").strftime("%Y-%m-%d")
        return game
    return None
```

Obtenção e Transformação de Dados

- Scripts para obter e converter automaticamente os dados

```
40 def price_history_json_to_csv(json_obj, filename):
41     df = price_history_json_to_dataframe(json_obj)
42     df.to_csv(filename)
43
44 def price_history_json_to_dataframe(json_obj):
45     x = []
46     y = []
47     prices = json_obj["prices"]
48     for price in prices:
49         price_date = datetime.strptime(price[0], "%b %d %Y %H: +0").date()
50         price_value = price[1]
51         x.append(price_date)
52         y.append(price_value)
53     return pd.DataFrame(np.array(list(zip(x, y))), columns=["date", "price"])
54
```

Obtenção e Transformação de Dados

- Scripts para obter e converter automaticamente os dados

```python
def get_binance_data(symbol, date, interval=intervals['daily']):
    base_url = "https://data.binance.vision"
    year = date.year
    month = date.month
    url = f"{base_url}/data/spot/monthly/klines/{symbol}/{interval}/{symbol}-{interval}-{year}-{month:02}.zip"

    req = requests.get(url, allow_redirects=True)

    with ZipFile(BytesIO(req.content)) as zf:
        zf.extractall("./extracted/")

    with open(f"./extracted/{symbol}-{interval}-{year}-{month:02}.csv", "r") as file:
        df = pd.read_csv(file, header=None, usecols=[0, 1], names=["date", "opening_price"])
        df["date"] = df["date"].apply(lambda x: datetime.utcfromtimestamp(x/1000))
        return df

def get_binance_data_range(symbol, start_date, end_date, interval=intervals['daily']):
    # TODO
    pass
```

Questões iniciais

- Como o comportamento dessas moedas virtuais se compara com o de moedas tradicionais?
- Similarmente, como seu comportamento se compara com o de criptomoedas?
- Como, e em que grau, elas são afetadas por eventos internos (como updates) e externos (como quedas no mercado)
- É possível prever seu comportamento com uso de algoritmos? (mais adequada para pesquisas de machine learning)

Questões baseadas em Consultas SQL

- Como conseguir um histórico combinado de preços?

```
MariaDB [testing]> SELECT O.Date Date, O.Price OSRS, M.Price MCSCK FROM OSRS_2 O, MCSCK_2 M WHERE O.Date=M.Date LIMIT 10;
+------------+------+-------+
| Date       | OSRS | MCSCK |
+------------+------+-------+
| 2019-11-15 | 0.55 |  2.47 |
| 2019-11-16 | 0.55 |  2.48 |
| 2019-11-17 | 0.55 |  2.48 |
| 2019-11-18 | 0.54 |  2.48 |
| 2019-11-19 | 0.53 |  2.40 |
| 2019-11-20 | 0.53 |  2.34 |
| 2019-11-21 | 0.55 |  2.36 |
| 2019-11-22 | 0.56 |  2.34 |
| 2019-11-23 | 0.55 |  2.35 |
| 2019-11-24 | 0.55 |  2.33 |
+------------+------+-------+
10 rows in set (0.007 sec)
```

Questões baseadas em Consultas SQL

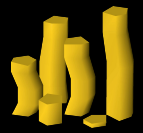
- Como conseguir o histórico da taxa de câmbio entre duas moedas?

```
MariaDB [testing]> SELECT O.Date Date, (O.Price/M.Price) OSRS_MCSCK FROM OSRS_2 O, MCSCK_2 M WHERE O.Date=M.Date LIMIT 10;
+------------+------------+
| Date       | OSRS_MCSCK |
+------------+------------+
| 2019-11-15 |   0.222672 |
| 2019-11-16 |   0.221774 |
| 2019-11-17 |   0.221774 |
| 2019-11-18 |   0.217742 |
| 2019-11-19 |   0.220833 |
| 2019-11-20 |   0.226496 |
| 2019-11-21 |   0.233051 |
| 2019-11-22 |   0.239316 |
| 2019-11-23 |   0.234043 |
| 2019-11-24 |   0.236052 |
+------------+------------+
10 rows in set (0.011 sec)
```

Questões baseadas em Consultas SQL

- Como é o comportamento médio de uma moeda X ao longo dos anos?

```
MariaDB [testing]> SELECT YEAR(O.Date) Year, AVG(O.Price) OSRS_AVG, MIN(O.Price) OSRS_MIN, MAX(O.Price) OSRS_MAX, STDDEV(O.Price) OSRS_STTDEV FROM OSRS_2 O GROUP BY YEAR(O.Date);
+------+----------+----------+----------+-------------+
| Year | OSRS_AVG | OSRS_MIN | OSRS_MAX | OSRS_STTDEV |
+------+----------+----------+----------+-------------+
| 2019 | 0.553617 |     0.53 |     0.62 |    0.018614 |
| 2020 | 0.624454 |     0.44 |     0.88 |    0.112294 |
| 2021 | 0.544593 |     0.48 |     0.63 |    0.028136 |
+------+----------+----------+----------+-------------+
3 rows in set (0.003 sec)
```



```
MariaDB [testing]> SELECT YEAR(M.Date) Year, AVG(M.Price) MCSCK_AVG, MIN(M.Price) MCSCK_MIN, MAX(M.Price) MCSCK_MAX, STDDEV(M.Price) MCSCK_STTDEV FROM MCSCK_2 M GROUP BY YEAR(M.Date);
+------+-----------+-----------+-----------+--------------+
| Year | MCSCK_AVG | MCSCK_MIN | MCSCK_MAX | MCSCK_STTDEV |
+------+-----------+-----------+-----------+--------------+
| 2012 |  1.991000 |      1.54 |      2.17 |     0.143035 |
| 2013 |  2.269507 |      1.82 |      2.47 |     0.139721 |
| 2014 |  2.397589 |      2.16 |      2.52 |     0.074864 |
| 2015 |  2.415315 |      2.16 |      2.52 |     0.069410 |
| 2016 |  2.409126 |      2.12 |      2.54 |     0.084539 |
| 2017 |  2.443123 |      2.28 |      2.52 |     0.039630 |
| 2018 |  2.398219 |      2.06 |      2.55 |     0.081391 |
| 2019 |  2.348986 |      2.19 |      2.53 |     0.064806 |
| 2020 |  2.347650 |      2.05 |      2.55 |     0.107982 |
| 2021 |  2.206186 |      1.97 |      2.54 |     0.121891 |
+------+-----------+-----------+-----------+--------------+
10 rows in set (0.008 sec)
```

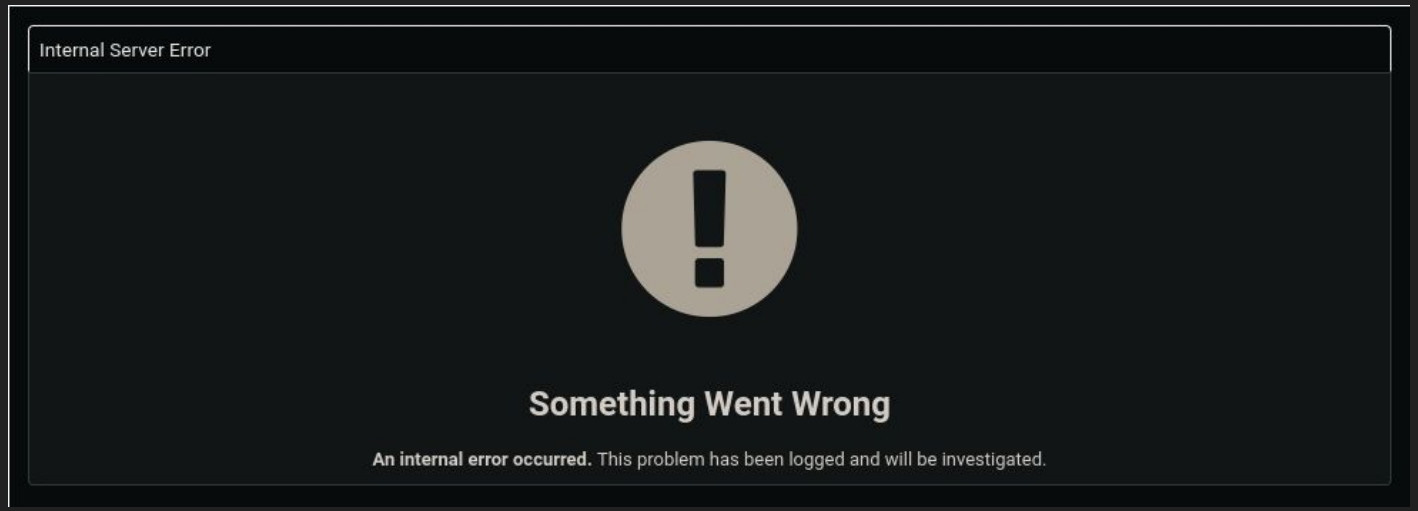
Questões baseadas em Consultas SQL

- Como a diferença entre moedas evolui ao longo dos anos?

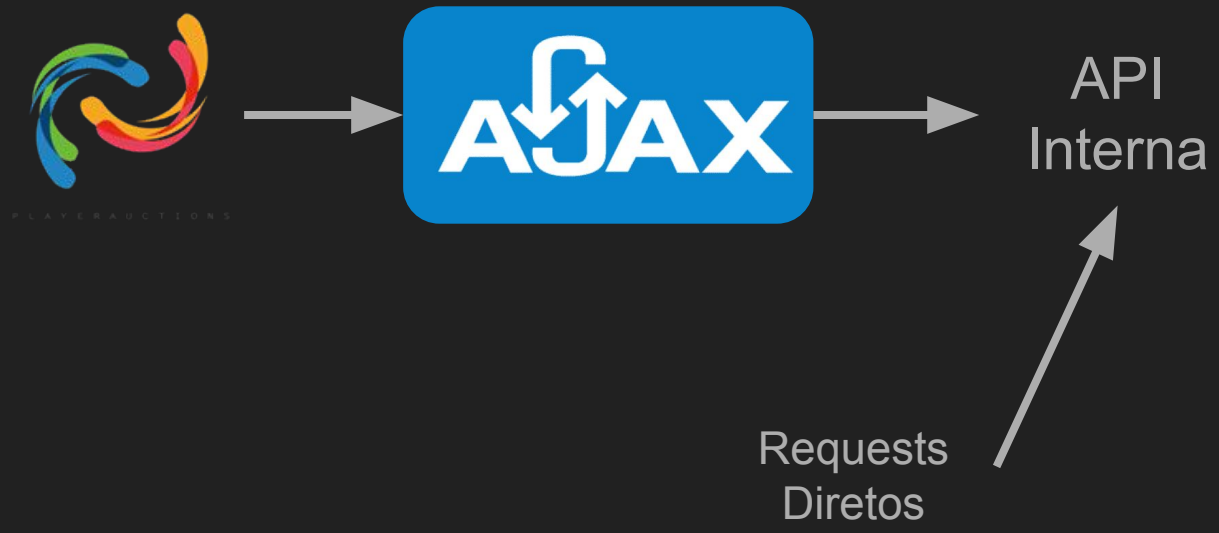
```
MariaDB [testing]> CREATE VIEW OSRS_MCSCK AS SELECT O.Date Date, (O.Price/M.Price) Price FROM OSRS_2 O, MCSCK_2 M WHERE O.Date=M.Date;
Query OK, 0 rows affected (0.010 sec)

MariaDB [testing]> SELECT YEAR(O.Date) Year, AVG(O.Price) AVG, MIN(O.Price) MIN, MAX(O.Price) MAX, STDDEV(O.Price) STTDEV FROM OSRS_MCSCK O GROUP BY YEAR(O.Date);
+------+--------------+----------+----------+--------------+
| Year | AVG          | MIN      | MAX      | STTDEV       |
+------+--------------+----------+----------+--------------+
| 2019 | 0.2370176170 | 0.217742 | 0.259912 | 0.0090106481 |
| 2020 | 0.2664729399 | 0.188841 | 0.397196 | 0.0506435960 |
| 2021 | 0.2411757623 | 0.194332 | 0.297030 | 0.0170052461 |
+------+--------------+----------+----------+--------------+
3 rows in set (0.027 sec)
```

Problemas - API Backpack.tf



Problemas - PlayerAuctions



Conclusão