

**Universidade Tecnológica Federal do Paraná - Campus Dois Vizinhos**

**Especialização em Ciência de Dados**

# **Projeto Integrador (PI)**

**Módulo 03**

**Dois Vizinhos - PR**

**2022**

## **1. IDENTIFICAÇÃO:**

**Título:** Análise de Dados de vendas de jogos eletrônicos com uso da linguagem de programação (Python), busca de similaridades em linguagem SQL e proposição do uso de sistemas NoSQL.

**Área:** Entretenimento

**Autores:** Wagner Moreno Schmitz, Matheus Vitoreli de Oliveira, Marcio Carvalho da Silva, Thiago Magalhães Amaral.

**Data:** 18/12/2022

## **2. INTRODUÇÃO:**

O comportamento da sociedade em relação às suas práticas de consumo tem sido foco de estudos no campo da Ciência de Dados com análises mais apuradas a fim de identificar padrões que possam ser utilizados para gerar competitividade nas empresas. Nos últimos anos, tem-se observado, no mundo todo, um crescimento exponencial em setores ligados ao entretenimento como o de jogos eletrônicos. Acredita-se que esse crescimento foi impulsionado também pela pandemia e pela necessidade de manter as pessoas por longos períodos em estado de isolamento social.

A era virtual tem chegado com grande força e ganhando espaço no cotidiano da sociedade, com representatividade também em diversos setores que abrangem desde áreas ligadas a indústrias, em seus diferentes ramos, como também no setor de serviços. Essa tendência de consumo virtual tem despertado no setor de games uma disputa por oferecer jogos eletrônicos que possibilitem uma experiência mais próxima possível da realidade.

Nesse sentido, torna-se interessante avaliar qual tem sido a preferência do consumidor, numa perspectiva global e regional, pelas principais empresas que despontam no mercado de jogos eletrônicos.

Ademais, espera-se a partir dessa análise proposta, avaliar o ranking dos jogos mais vendidos e sua participação de vendas no mundo e por regiões, entendendo ser um estudo que demonstra um potencial para contribuir junto às empresas produtoras de jogos na definição de estratégias para aumentar suas metas de vendas.

Nesse cenário, a escolha do dataset base deste estudo consiste nos registros fictícios de vendas de jogos eletrônicos obtidos a partir do ambiente de compartilhamento de dados

Kaggle. Tal arquivo intitulado “Video Games Sales (vgsales)” é composto pelos seguintes atributos:

- Rank - Classificação das vendas gerais
- Nome - O nome do jogo
- Plataforma - Plataforma de lançamento dos jogos (ou seja, PC, PS4, etc.)
- Ano - Ano de lançamento do jogo
- Gênero - Gênero do jogo
- Editora - Editora do jogo
- NA\_Sales - Vendas na América do Norte (em milhões)
- EU\_Sales - Vendas na Europa (em milhões)
- JP\_Sales - Vendas no Japão (em milhões)
- Other\_Sales - Vendas no resto do mundo (em milhões)
- Global\_Sales - Total de vendas mundiais.

Fonte: <https://www.kaggle.com/datasets/gregorut/videogamesales>

A partir desse conjunto de dados, foi possível atender aos requisitos do Projeto Integrador distribuídos pelas disciplinas do terceiro módulo da especialização em Ciência de Dados da UTFPr (Campus Dois Vizinhos), a saber:

a) **Recuperação da Informação Baseada em Conteúdos (RIBC):** que pretende apresentar o vetor de características o qual permitirá fazer consultas por similaridade entre alguns jogos que mantém características de vendas semelhantes entre si;

b) **Linguagens de Programação para Ciência de Dados (LPCD):** que pretende apresentar manipulações com uso da linguagem Python e algumas bibliotecas mais tradicionais, a fim de desenvolver consultas ao dataset com geração de gráficos que permitam montar um dashboard de apoio à decisão, e, por fim;

c) **Introdução ao Big Data (IBD):** que pretende especificar uma proposta de utilização de um sistema NoSQL considerando as características dos dados do dataset adotado no estudo, com destaque às motivações dessa escolha e a ideia de como se dá essa transição de um modelo relacional para um modelo NoSQL.

### **3. OBJETIVOS**

#### **3.1. OBJETIVO GERAL**

Analisar a representatividade dos principais games vendidos mundialmente, considerando um determinado período de análise, utilizando-se de técnicas de programação com linguagem Python, busca de similaridades em linguagem SQL, além de sugerir uma aplicação dessa base em sistema NoSQL com base em Dataset de vendas de jogos eletrônicos.

#### **3.2 OBJETIVOS ESPECÍFICOS**

- Aplicar a linguagem de programação Python para desenvolvimento do DW no PostgreSQL e inserção de dados do dataset (vendas de games) com uso das bibliotecas Pandas, Matplotlib, Psycopg2, entre outras;
- Aplicar os métodos de consultas por similaridade bem como a demonstração dos vetores característicos adotados com base no *dataset*, descrever as funções utilizadas nas consultas Range Query e KNN;
- Descrever e comparar a escolha de um banco de dados NoSQL a um sistema gerenciador de banco de dados relacional e suas vantagens na aplicação e desenvolvimento de um DW alinhado ao *dataset* .

#### 4. RECUPERAÇÃO DE INFORMAÇÃO BASEADA EM CONTEÚDO

Visando a aplicação da recuperação de informações baseada em conteúdo criou-se a hipótese de um banco de dados que utiliza o número de vendas de cada jogo para calcular sua proximidade.

##### **4.1. ARMAZENAMENTO: Descrever como armazenar as imagens (ou outros dados complexos) no data warehouse projetado e implementado**

Para receber os dados da base de dados vgsales.csv um database foi criado no pgAdmin4 através das funções CREATE EXTENSION CUBE e CREATE TABLE jogos especificadas abaixo:

*create extension cube*

*create table jogos(  
    id integer PRIMARY KEY,  
    features float[]);*

Caso o database utilizado no desenvolvimento do projeto trabalhasse com imagens, uma possibilidade seria utilizar o código dessas imagens em JSON e cadastrar dentro do banco de dados através de vetores de características.

Como a base utilizada não trabalha com imagens, foram criados vetores de características considerando os atributos numéricos referente ao número total de vendas em milhões por região.

##### **4.2 ARMAZENAR VETORES: Como armazenar os vetores de características no data warehouse mantido no PostgreSQL**

Para armazenar os vetores características no data warehouse mantido no PostgreSQL, foram inseridos 16.600 jogos com o vetor de característica associado às vendas (NA\_Sales, EU\_Sales, JP\_Sales, Other\_Sales, Global\_Sales), ou seja, o vetor de característica tinha 5 características composta das vendas por região e globalmente, com o total de 83.000 elementos. Foi usada a seguinte função para inserir todos os vetores no data warehouse:

```
insert into jogos values (1, '{4149,2902,377,846,8274}');
```

Outra possibilidade de inserção, seria gerar um CSV com todos os vetores de características e importar esse CSV para o Pgadmin, porém foi optado pela inserção via código.

#### **4.3 CONSULTAS POR SIMILARIDADE: Criar consultas por similaridade range query e kNN (aceita-se o valor fixo para k, ex: os 5 mais similares) sobre o data warehouse que mantém os vetores de características.**

Para as consultas por similaridades, escolhemos o jogo Super Mario Bros, muito popular entre os jovens e adultos, para verificar quais outros 5 jogos são similares a ele em termos de vendas. Fizemos a comparação usando a função l2 e a range query.

##### **4.3.1 Método KNN**

Para a implantação do KNN, criamos a função l2, a qual calcula a distância euclidiana:

```
create or replace function l2(elem1 float[], elem2 float[]) returns float as $$
```

```
declare
```

```
size integer;
```

```
somat float;
```

```
begin
```

```
select cardinality(features) into size from jogos limit 1;
```

```
somat := 0;
```

```
for i in 1..size loop
```

```
    somat := somat + (ABS(elem1[i] - elem2[i])*ABS(elem1[i] - elem2[i]));
```

```
end loop;
```

```
return sqrt(somat);
```

```
end $$
```



```
language plpgsql;
```

Em seguida, estruturamos o comando abaixo para selecionar os 5 mais similares ao jogo do Super Mario Bros (desconsiderando ele mesmo) em termos de vendas nas cinco posições do vetor característica.

```
select id, l2(features,array[2908,358,681,77,4024]) as distance
from jogos
order by l2(features,array[2908,358,681,77,4024]) LIMIT 6;
```

Também foi utilizada a função cube para ordenar as features e chegar ao mesmo resultado:

```
select id, cube(features) <-> cube(array[2908,358,681,77,4024]) as distance
from jogos
order by cube(features) <-> cube(array[2908,358,681,77,4024]);
```

	 id [PK] integer	 distance double precision
1	2	0
2	10	1408.4771918636097
3	3	1722.2987545719238
4	8	1946.40001027538
5	9	1998.776125532822
6	5	2088.719224788243

O que resultou em:

- 1º lugar, Duck Hunt (1408.4771918636097);
- 2º lugar, Mario Kart Wii (1722.2987545719238);
- 3º lugar, Wii Play (1946.40001027538);
- 4º lugar, New Super Mario Bros. Wii (1998.776125532822);
- 5º lugar, Pokemon Red/Pokemon Blue (2088.719224788243);

#### 4.3.2 Método RangeQuery

Para a implantação criamos a função RangeQuery da seguinte forma:

```
create or replace function RangeQuery(qc float[], radius float)
returns table (id integer, distance float) as $$
begin
return query
select jogos.id, l2(features,qc) as distance
```

```
from jogos  
where l2(features,qc) <= radius;  
end $$  
language plpgsql;
```

Foi escolhido um raio grande para selecionar todos os elementos e chegar ao mesmo resultado:

```
select * from RangeQuery(array[2908,358,681,77,4024], 9000) LIMIT 20;
```

O que resultou em:

- 1º lugar, Duck Hunt (1408.4771918636097);
- 2º lugar, Mario Kart Wii (1722.2987545719238);
- 3º lugar, Wii Play (1946.40001027538);
- 4º lugar, New Super Mario Bros. Wii (1998.776125532822);
- 5º lugar, Pokemon Red/Pokemon Blue (2088.719224788243);



## 5. LINGUAGENS DE PROGRAMAÇÃO PARA CIÊNCIA DE DADOS

Com o objetivo de gerar gráficos que possam ser incorporados em um dashboard relacionado ao número de vendas através dos anos, criou-se um banco de dados chamado vendas-jogos. Esse banco de dados recebe as informações de um dataframe que por sua vez importa as informações de uma base de dados .csv chamada vgsales.csv.

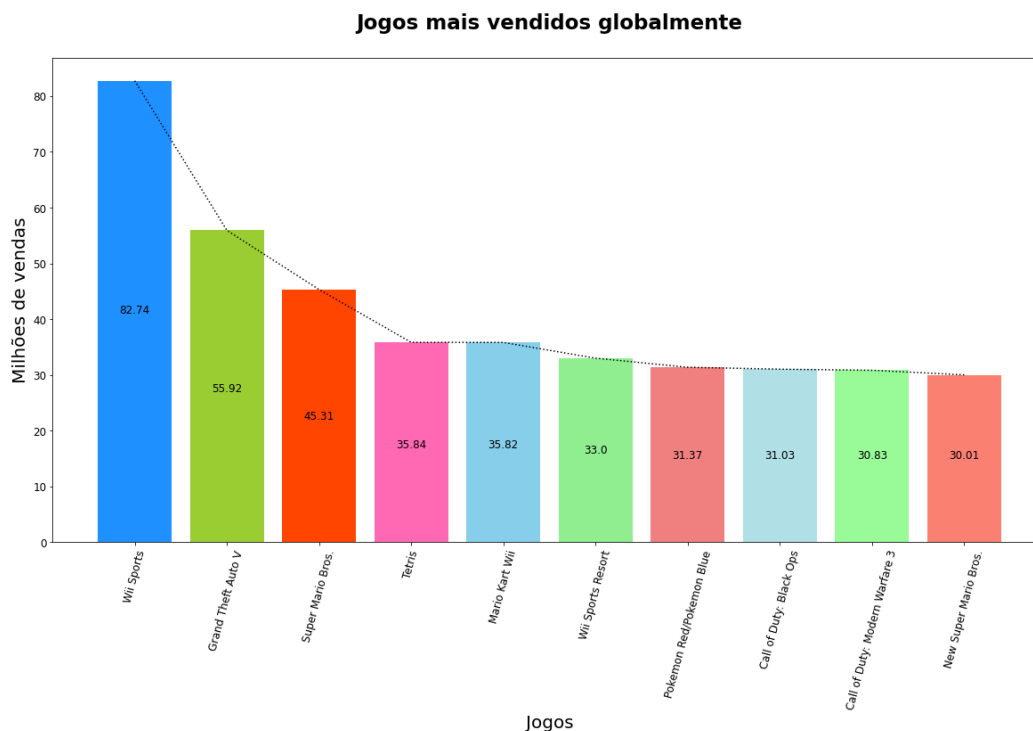
### 5.1. CONSULTAS AO DATA WAREHOUSE COM USO DO PYTHON

Após o cadastro dos dados foram gerados cinco gráficos que detalham os diversos indicadores de vendas de jogos, abaixo é possível verificar cada um deles.

#### 5.1.1 GRÁFICO 01 - Jogos mais vendidos globalmente

Neste gráfico são selecionados os dez jogos de videogames mais vendidos globalmente. Essa seleção foi feita pelo agrupamento do número total de vendas globais e ordenados pelos jogos com maior número de vendas.

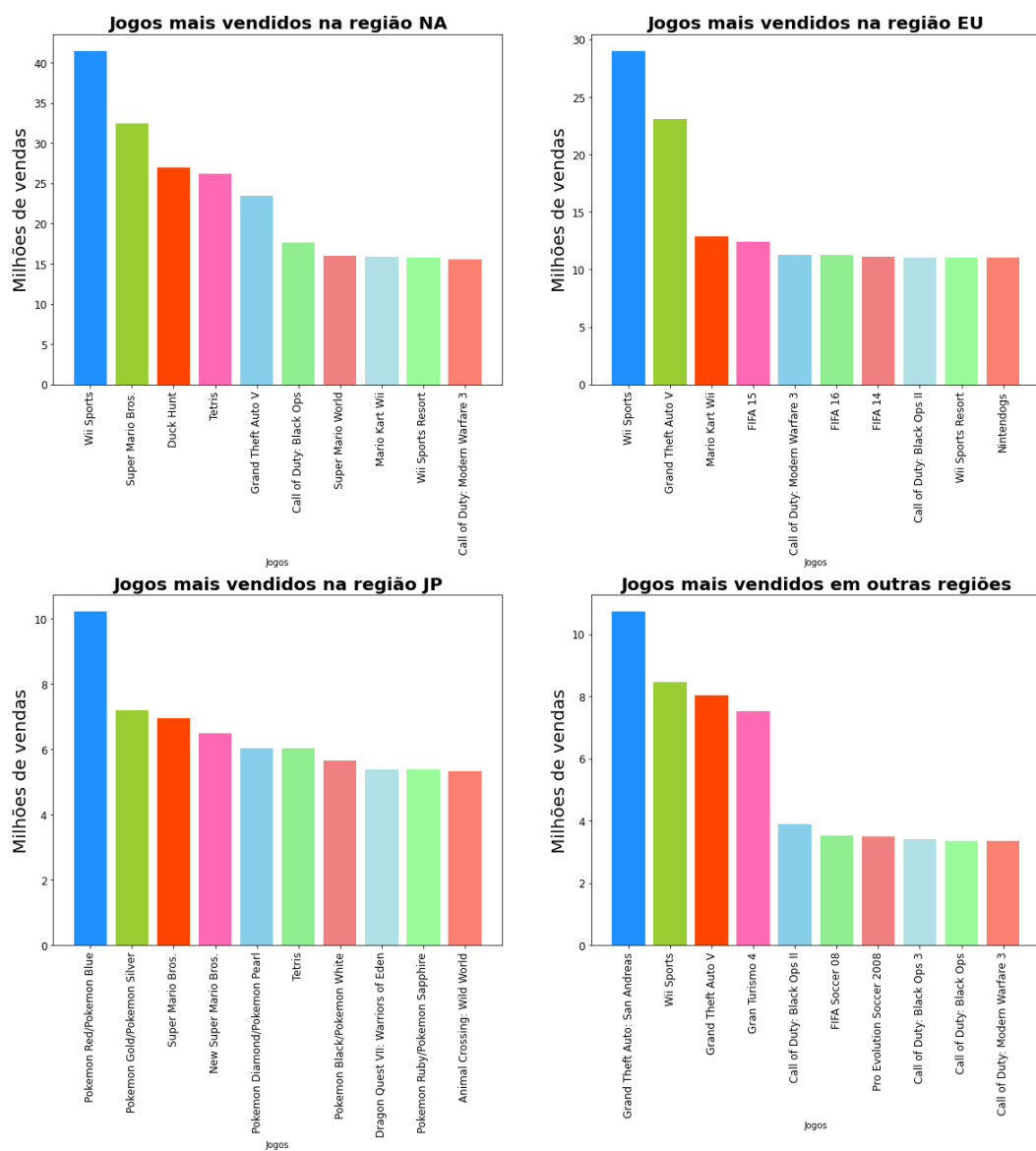
A geração do gráfico foi feita com o formato de gráfico de barras ordenado pelo maior número de vendas.



### 5.1.2 GRÁFICO 02 - Jogos mais vendidos por cada região

Assim como ocorrido no gráfico 01, este gráfico apresenta os dez jogos de videogames mais vendidos agrupados pelo número total de vendas. Entretanto nessa plotagem o gráfico se divide em quatro com as informações relacionadas a cada uma das regiões disponíveis na base de dados.

A geração do gráfico foi feita com o formato de gráfico de barras ordenado pelo maior número de vendas.

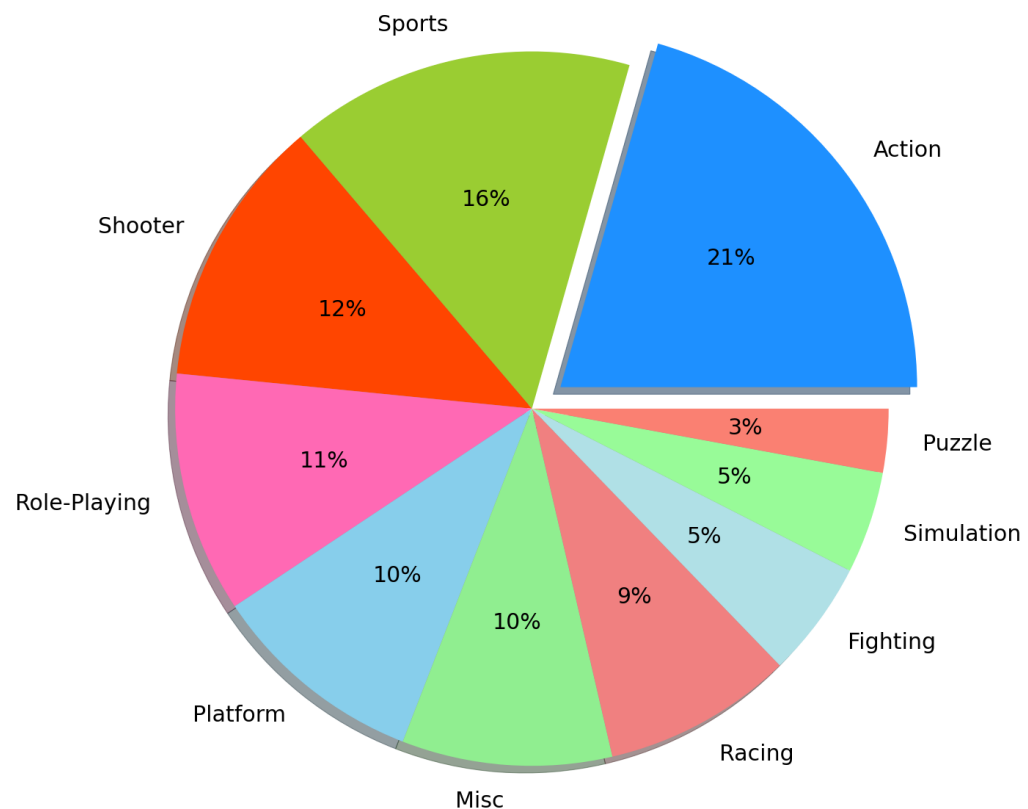


### 5.1.3 GRÁFICO 03 - Porcentagem de vendas por gênero

Para compreender qual gênero é mais popular entre os jogadores de videogame, foi necessário agrupar o número total de vendas globalmente por cada gênero disponível no dataframe.

Com essas informações ocorreu a construção de um gráfico de pizza que apresenta a porcentagem de vendas de jogos por gênero destacando do gráfico o gênero com maior interesse do público consumidor.

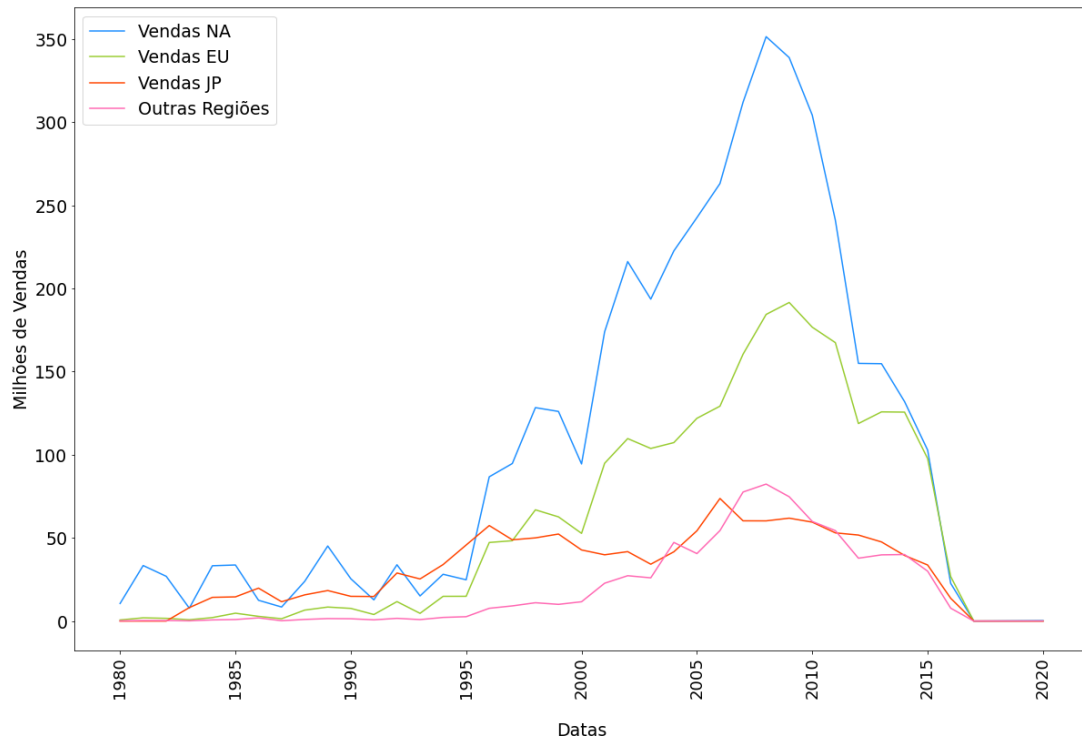
**Porcentagem de vendas por gênero**



### 5.1.4 GRÁFICO 04 - Evolução de vendas por região através dos anos

Com o intuito de analisar a evolução de vendas de jogos de videogame através dos anos em cada região, criou-se dataframes para cada região com o agrupamento do número de vendas de cada ano e gerado um gráfico de linhas com a evolução de cada região, sendo elas: vendas NA, vendas EU, vendas JP, Outras regiões.

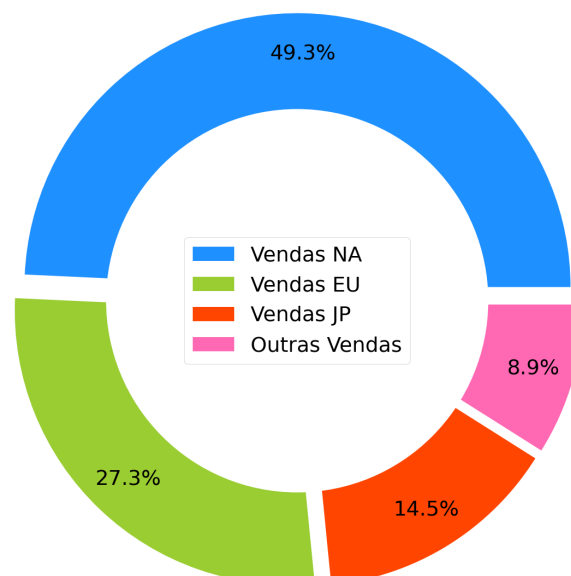
Evolução de vendas por região através dos anos



#### 5.1.5 GRÁFICO 04 - Evolução de vendas por região através dos anos

Para finalizar a análise foi criado um gráfico de rosca (donut) que apresenta a porcentagem de vendas globais por região. Para gerar o gráfico, utilizou-se duas listas, vendas\_valor contendo o número total de vendas em cada região e vendas\_titulo com o nome de cada uma delas. Elas foram utilizadas para gerar o gráfico e legendá-lo.

Regiões com mais vendas



## 6. INTRODUÇÃO AO BIG DATA

O tratamento e manipulação de dados é um processo que exige suporte em sistemas, metodologias e ferramentas que auxiliem o Cientista de Dados nas análises de negócios das empresas, garantindo maior agilidade nos retornos oriundos de consultas em bancos de dados, por vezes, gigantescos cuja capacidade de armazenamento vai da ordem de *tera* a *petabytes*, além de poderem ser distribuídos em inúmeros servidores, possibilitando ganhos superiores de desempenho e escalabilidade, tanto vertical como horizontal, mediante o particionamento desses dados (*sharding*), retornando informações com maior efetividade no menor tempo possível.

Na Era Digital, o volume desses recursos de dados impulsionados, principalmente, pelas inúmeras plataformas digitais de captação de informações em tempo real, tem sido o propulsor da área de estudos, conhecida popularmente por *Big Data*.

Para garantir esse suporte de armazenamento de dados estruturados e/ou não estruturados foram desenvolvidos inúmeros sistemas gerenciadores de banco de dados, sendo alguns deles mais conhecidos por apoiarem a estrutura baseada no modelo relacional que adotam a linguagem SQL.

No entanto, pesquisas vêm se desenvolvendo focando sistemas que auxiliem também a manipulação de dados do tipo não estruturados (complexos) e, dessa forma, surgem os sistemas gerenciadores de banco de dados do tipo NoSQL, os quais possuem características que se adequam a diferentes tipos de projetos e modelos de negócio adotados pelas empresas considerando não somente o tipo do dado, como seu volume e outras características de como são coletados e escritos.

Nesses termos, no decorrer da disciplina de Introdução ao Big Data, onde foram apresentados uma série de sistemas NoSQL com base em suas características e adequações aos diferentes modelos de negócio das organizações, destaca-se nas seções subsequentes a escolha do tipo mais adequado ao dataset estruturado na proposta deste estudo.

### 6.1 PROPOSIÇÃO DE UM SISTEMA NOSQL PARA O CASO EM ANÁLISE

#### Pergunta Norteadora: 1. Qual é o NoSQL escolhido e por que da sua escolha?

Em vista dessa condição de processar dados em grandes volumes e considerando a possibilidade do *dataset* escolhido expandir a quantidade de registros, optou-se pelo uso de um modelo NoSQL que melhor se adapta a problemática, tendo como escolha o **MongoDB** que é um banco de dados orientado a documentos com a escalabilidade e flexibilidade junto com a consulta e indexação.

O modelo documental do MongoDB é simples para os desenvolvedores aprendem a utilizarem, ele armazena dados em documentos do tipo JSON (BSON) flexíveis o que significa que os

campos podem variar de documento para documento e a estrutura de dados pode ser alterada ao longo do tempo, tornando os dados fáceis de serem trabalhados, bem como as consultas ad hoc, indexação e agregação em tempo real podem fornecer maneiras poderosas de acessar e análise de dados.

### **6.1.1 Modelo de Dados adotado na proposta NoSQL**

**Pergunta Norteadora: 2. Qual é o modelo de dados do NoSQL escolhido?**

O modelo escolhido foi o de documentos. A modelagem dos dados no MongoDB deve considerar o uso dos dados pela aplicação (consultas, atualizações e processamento) considerando as regras de negócio da organização que deseja consumir esses dados. Por se tratar de um sistema de armazenamento de dados não estruturados sua modelagem é mais fácil de se ajustar conforme haja necessidade de inclusão de novas informações nos documentos da coleção.

No caso do *dataset* proposto, a aplicação que define a coleta dos dados de vendas de jogos eletrônicos priorizou o aspecto da simplicidade considerando apenas uma coleção que armazena os registros das vendas globais e de algumas regiões do mundo. Há que se considerar que, no caso de se perceber a necessidade de incluir outras informações, por exemplo, no atributo “**Vendas do resto do mundo (Other\_Sales)**”, passando a especificar outras regiões que não haviam sido consideradas na versão inicial de desenvolvimento do banco, isso não representará grandes dificuldades, pois essa é uma das vantagens dos modelos NoSQL que permite maior flexibilidade a mudanças na estrutura das coleções sem comprometer os registros feitos anteriormente à inclusão de outros atributos.

### **6.1.2 Motivações pela escolha do NoSQL em detrimento ao PostgreSQL**

**Pergunta Norteadora: 3. Por que seria interessante migrar do PostgreSQL ao NoSQL escolhido? Detalhar a motivação.**

O NoSQL MongoDB é um banco de dados não relacional (NoSQL) que surgiu na primeira década do ano 2000, baseado nas experiências de construção em grande escala, alta disponibilidade e robustez. Tal modelo é completamente distinto do modelo relacional PostgreSQL, possuindo muitos diferenciais, como melhor desempenho, custos reduzidos, dados aprimorados, velocidade para o mercado, flexibilidade a mudanças com inserção de dados distintos, além de ter seu código aberto e ser gratuito (MongoDB, 2020).

Essa classe de banco de dados usa o formato de arquivos do tipo JSON (BSON), que pode ser estendido, dado que os documentos têm sua estrutura feita em coleções, não tendo a exigência de

que os documentos armazenados nele tenham todos o mesmo campo, como por exemplo um console e um jogo que podem ser armazenados em uma coleção, mesmo que não haja um sentido aparente (SILVA, 2013). Assim, o MongoDB é bastante útil em casos onde se queira armazenar documentos dentro de um campo de dados flexível.

Outra vantagem do MongoDB é que, enquanto o PostgreSQL usa a função GROUP\_BY para processar e executar consultas agregadas, o MongoDB normalmente usa *pipelines* de agregação para processar suas consultas obtendo maior ganho de performance.

Ante o exposto, faz-se mais conveniente o emprego do banco de dados NOSQL MongoDB por suas características citadas anteriormente em confronto ao PostgreSQL (relacional). A propriedade chave, motivação para essa escolha, que diferencia o MongoDB do PostgreSQL é sua abordagem para armazenar seus dados. Já que não é relacional, o MongoDB usa coleções ao invés de tabelas. Uma chave estrangeira é simplesmente um conjunto de atributos em uma tabela que se refere à chave primária de outra tabela e lida com cargas de trabalho transacionais, operacionais e analíticas em escala.

## **6.2 IMPLEMENTAÇÃO DAS TABELAS DIMENSÃO E FATO NO NOSQL**

No MongoDB não há o conceito de chave estrangeira como acontece nos sistemas do modelo relacional, cuja função é fazer a relação entre tabelas dimensão e fatos o que chamamos de integridade referencial.

Podemos desenvolver uma modelagem orientada a documentos, os quais podem ter sub-documentos e arrays embutidos. Com MongoDB podemos armazenar os dados sem um esquema pré-definido, com maior flexibilidade, diferente dos bancos SQL. Onde devemos primeiro declarar a estrutura da tabela para depois podermos inserir dados, onde pode-se criar relações referenciais ou normalizadas, documentos de uma coleção podem ser referenciados em outras.

Assim como os documentos embutidos ou não normalizados onde todas as referências englobadas dentro de um único documento, que está incorporando as relação dentro de um documento, tal relação é vista como um subdocumento, pois os dados são embutidos em arrays ou campos do documento (USP, 2017).

## **6.3 SINTAXE DE CONSULTAS DO NOSQL PROPOSTO**

Após concluída a instalação do MongoDB, bem como feitas as importações das bibliotecas Python (pymongo e json) e ainda a conexão padrão ao localhost: 27017,

parte-se para a criação do banco de dados e a coleção considerando os atributos do dataset utilizado “vgsales”.

Os procedimentos contendo a sintaxe adotada no MongoDB, seguem conforme as etapas a seguir:

- **PASSO 1: Criando Banco de Dados**

*# O banco de dados no MongoDB é criado com o comando use*

```
use vendas-jogos  
switched to db vendas-jogos
```

**Observação:** No MongoDB o banco de dados só é criado de fato após a inclusão dos primeiros documentos ou coleção (seguindo a linguagem técnica adotada nesse tipo de sistema NoSQL)

- **PASSO 2: Criando uma coleção chamada vendas**

*# Uma coleção seria o equivalente a uma tabela em sistemas relacionais*

```
db.createCollection("vendas")
```

- **PASSO 3: Inserindo o registro de vendas de um jogo na coleção vendas**

*# Para exemplificar a sintaxe de inserção, foi considerado um jogo apenas do dataset*

```
db.vendas.insert({  
    'Name': "Wii Sports"  
    'Platform': "Wii"  
    'Year': "2006"  
    'Genre': "Sports"  
    'Publisher': "Nintendo"  
    'NA_Sales': "41.49"  
    'EU_Sales': "29.02"  
    'JP_Sales': "3.77"  
    'Other_Sales': "8.46"  
    'Global_Sales': "82.74"  
})
```

**Observação:** No caso de inserir os dados do dataset completo, isto é a importação do arquivo JSON na coleção vendas, o comando seria o seguinte:

*#abrindo o arquivo vgsales.json*

```
with open('vgsales.json') as file:  
    arquivo = json.load(file)
```

*#inserindo múltiplos registros dentro da coleção vendas*

```
db.vendas.insert_many(arquivo)
```



- **PASSO 4: Realizando consultas na coleção vendas**

*#contando quantos documentos foram criados*

```
db.vendas.count_documents({})
```

16600

*#Uma query que conte a quantidade de jogos de aventura*

```
db.vendas.count_documents({'genre': 'adventure'})
```

*#Uma query que retorne todos os jogos da marca Nintendo*

```
list(db.vendas.find({'Publisher': 'Nintendo'}))
```

*#Query para retornar a quantidade de documentos que contenham o jogo Pokemon Gold/Pokemon Silver*

```
db.vendas.count_documents({'Name': 'Pokemon Gold/Pokemon Silver'})
```

## 7. CONCLUSÃO

Conclui-se que este trabalho mostrou na prática como trabalhar com as ferramentas de Recuperação da Informação Baseada em Conteúdos (RIBC), Linguagens de Programação para Ciência de Dados (LPCD) e Introdução ao Big Data (IBD). Apresentamos como o vetor de características poderia fazer consultas por similaridade entre alguns jogos que mantêm características de vendas semelhantes entre si; assim como mostramos manipulações com uso da linguagem Python e algumas bibliotecas mais tradicionais, a fim de desenvolver consultas ao dataset com geração de gráficos que permitam montar um dashboard de apoio à decisão; e, por fim, especificamos uma proposta de utilização de um sistema NoSQL considerando as características dos dados do dataset adotado no estudo, com destaque às motivações dessa escolha e a ideia de como se dá essa transição de um modelo relacional para um modelo NoSQL.

O Projeto Integrador mostrou como as disciplinas podem ser aplicadas ao mesmo dataset e nos instigou a trabalhar em equipe mesmo distantes geograficamente. A visão prática de cada conteúdo é importante para a fixação dos conteúdos teóricos e para o desenvolvimento de nossas competências técnicas como futuros cientistas de dados.

## **7. REFERÊNCIAS BIBLIOGRÁFICAS**

Milani, André. PostgreSQL guia do programador. Rio de Janeiro, Editora Novatec, 2008.

Silva, Fabio; como surgiu o MongoDB. Dezembro, 2013. Disponível em:  
<<https://fabiosilva.com.br/2014/08/07/como-surgiu-o-mongodb/>>. Acesso em: 15/12/2022.

Castro, João; Ciferri Cristina: MongoDB. Disponível em:  
<<http://wiki.icmc.usp.br/images/1/18/SSC0542012018MongoDB.pdf>>. Acesso em  
16/12/2022