Тема 23. Визуализация данных в Big Data проектах.

Цель занятия:

Изучить различные подходы к интеграции MongoDB с приложениями.

Учебные вопросы:

1. Введение в визуализацию данных.
2. Обзор библиотек для визуализации данных в Python.
3. Визуализации данных в PySpark с использованием библиотек Matplotlib и Seaborn.

1. Введение в визуализацию данных.

Визуализация данных играет важную роль в проектах Big Data, поскольку помогает анализировать и интерпретировать большие объемы данных.

Эффективная визуализация позволяет быстро выявлять закономерности, аномалии и инсайты.

Зачем нужна визуализация данных в Big Data?

- Упрощение анализа: Помогает упростить сложные данные и делает их более понятными для аудитории.
- Выявление закономерностей: Позволяет быстро обнаружить тренды, корреляции и аномалии.
- Улучшение принятия решений: Визуализированные данные облегчают процесс принятия решений на основе анализа.
- Коммуникация результатов: Эффективная визуализация помогает донести идеи до заинтересованных сторон.

Основные методы визуализации данных:

Графики и диаграммы:

- Линейные графики: Для отображения изменений во времени.
- Столбчатые диаграммы: Для сравнения значений между различными категориями.
- Круговые диаграммы: Для отображения долей.
- Гистограммы: Для распределения числовых данных.
- Скаттер-графики: Для изучения зависимостей между двумя переменными.

Тепловые карты: Для визуализации данных в виде матрицы, где цвет представляет значения.

Геоинформационные карты: Для визуализации пространственных данных

Применение в Big Data проектах:

- Анализ логов: Визуализация логов и метрик для выявления проблем в системах.
- Анализ пользовательского поведения: Визуализация данных о пользователях для улучшения продуктов.
- Финансовые отчеты: Визуализация финансовых данных для анализа и прогнозирования.
- Научные исследования: Визуализация сложных наборов данных для исследовательских проектов.

2. Обзор библиотек для визуализации данных в Python.

Matplotlib — одна из самых популярных библиотек для визуализации данных в Python. Она предоставляет множество возможностей для создания различных типов графиков и визуализаций, от простых до сложных.

1. Основные характеристики Matplotlib:

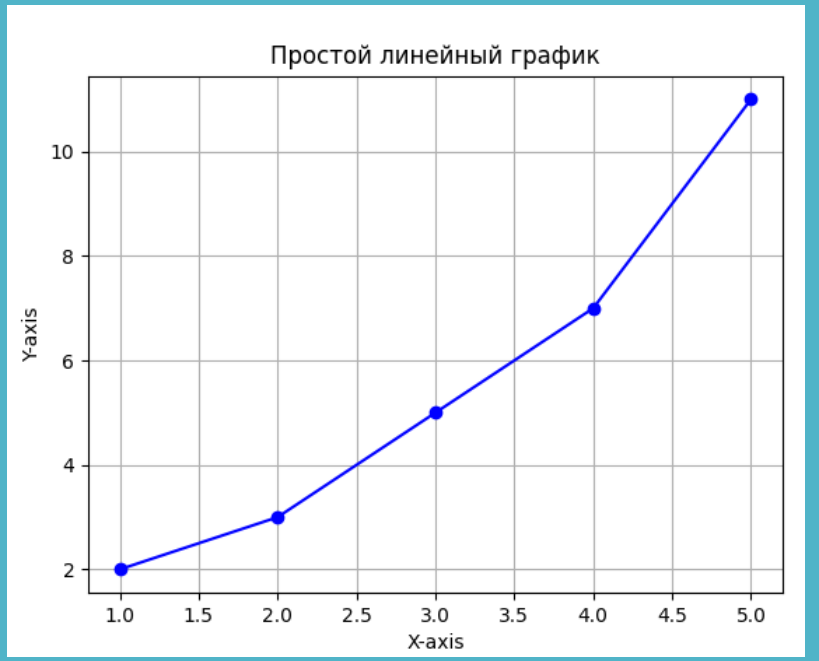
- Гибкость: Позволяет создавать практически любые типы графиков, включая линейные графики, гистограммы, диаграммы рассеяния, тепловые карты и многое другое.
- Настраиваемость: Обширные возможности настройки графиков, включая цвета, стили линий, аннотации, легенды и заголовки.
- Интерактивность: Поддерживает интерактивные графики с использованием ipython и других инструментов, что позволяет пользователям взаимодействовать с визуализацией.
- Экспорт: Возможность сохранения графиков в различных форматах, таких как PNG, PDF, SVG и другие.

Основные компоненты Matplotlib

- Figure: Основной контейнер для всех графиков. Можно рассматривать как холст для рисования.
- Axes: Подграфики внутри фигуры. Каждый график (например, линейный, столбчатый) находится на своих осях.
- Artist: Все, что отображается на графике, включая линии, текст, метки, легенды и т.д.

Вот простой пример, который показывает, как создать линейный график с помощью Matplotlib.

```python
import matplotlib.pyplot as plt

# Данные для графика
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]

# Создание графика
plt.plot(x, y, marker='o', linestyle='-', color='b')

# Настройка заголовка и меток осей
plt.title('Простой линейный график')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')

# Отображение графика
plt.grid(True)  # Включение сетки
plt.show()      # Показать график
```



Виды графиков

Matplotlib поддерживает множество типов графиков:

- Линейные графики: Для отображения данных, изменяющихся во времени.
- Столбчатые графики: Для сравнения значений между разными категориями.
- Гистограммы: Для отображения распределения данных.
- Диаграммы рассеяния (scatter plots): Для изучения взаимосвязи между двумя переменными.
- Тепловые карты: Для визуализации двумерных данных.

Настройка графиков

Matplotlib позволяет настраивать графики с помощью различных параметров:

- Цвета и стили: Вы можете изменять цвет и стиль линий, добавлять маркеры и изменять толщину линий.
- Подписи: Добавление заголовков, меток осей и аннотаций.
- Легенды: Добавление легенд для обозначения различных линий и категорий.

Заключение

Matplotlib — это мощный и гибкий инструмент для визуализации данных в Python. Он предоставляет множество возможностей для создания различных графиков, их настройки и экспорта. Благодаря своей популярности и большому сообществу разработчиков, Matplotlib является отличным выбором для визуализации данных как в небольших проектах, так и в крупных аналитических задачах

Seaborn — это мощная библиотека для визуализации данных, построенная на основе Matplotlib. Она предназначена для упрощения создания сложных визуализаций и обеспечивает высококачественные графики с минимальными усилиями.

Seaborn особенно полезен для анализа статистических данных.

Основные характеристики Seaborn

- Статистическая визуализация: Seaborn предоставляет готовые функции для создания графиков, которые хорошо подходят для статистического анализа.
- Эстетика: Библиотека использует приятные по умолчанию цветовые палитры и стили, что позволяет создавать красивые графики с минимальной настройкой.
- Работа с Pandas: Seaborn хорошо интегрируется с библиотекой Pandas, что упрощает работу с DataFrame.
- Упрощение сложных графиков: Seaborn предоставляет функции для создания сложных визуализаций, таких как парные графики и тепловые карты, с минимальным количеством кода.

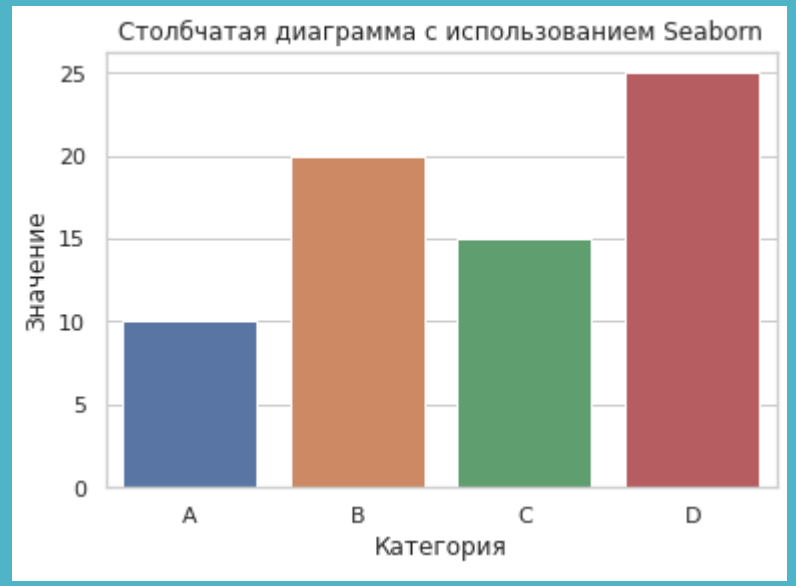
Основные компоненты Seaborn

Seaborn основан на Matplotlib, но добавляет некоторые удобные функции:

- Функции для создания графиков: Seaborn предоставляет высокоуровневые функции, которые автоматически управляют настройкой осей и форматированием.
- Темы и палитры: Упрощает изменение стиля графиков и выбор цветовых схем.

Пример, показывающий, как создать столбчатую диаграмму с использованием Seaborn.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Пример данных
data = {
    'Категория': ['A', 'B', 'C', 'D'],
    'Значение': [10, 20, 15, 25]
}

# Создание DataFrame
import pandas as pd
df = pd.DataFrame(data)

# Создание столбчатой диаграммы
sns.barplot(x='Категория', y='Значение', data=df)

# Настройка заголовка
plt.title('Столбчатая диаграмма с использованием Seaborn')

# Отображение графика
plt.show()
```



Виды графиков в Seaborn

Seaborn поддерживает множество типов графиков, включая:

- Столбчатые графики (bar plots): Для визуализации категориальных данных.
- Диаграммы рассеяния (scatter plots): Для визуализации взаимосвязей между двумя переменными.
- Тепловые карты (heatmaps): Для отображения матриц значений.
- Кросс-графики (pair plots): Для визуализации взаимосвязей между всеми парами переменных в наборе данных.
- Линейные графики (line plots): Для отображения изменений во времени.

Настройка графиков

Seaborn предлагает множество параметров для настройки графиков:

- Палитры цветов: Легко настраивайте цветовые схемы с помощью sns.set_palette().
- Темы: Изменение темы графиков с помощью sns.set_style(), например, на "darkgrid" или "whitegrid".
- Анотации: Легкое добавление аннотаций к графикам.

Заключение

Seaborn — это мощный инструмент для визуализации данных в Python, который делает процесс создания высококачественных графиков более удобным и интуитивно понятным. Благодаря своей интеграции с Pandas и простоте в использовании, Seaborn становится отличным выбором для анализа статистических данных и создания информативных визуализаций в проектах Big Data.

3. Визуализации данных в PySpark с использованием библиотек Matplotlib и Seaborn.

Основные шаги для визуализации данных с использованием Matplotlib и Seaborn в PySpark:

- Настройка окружения PySpark: Убедитесь, что у вас установлен PySpark и нужные библиотеки (Matplotlib и Seaborn).
- Подключение к кластеру Spark: Создайте объект SparkSession.
- Загрузка данных: Используйте методы PySpark для загрузки данных из файловых систем (CSV, JSON и т.д.).
- Обработка данных: Выполните необходимые трансформации и агрегации с использованием методов PySpark.
- Преобразование данных для визуализации: Переведите данные в Pandas DataFrame для визуализации.
- Создание графиков: Используйте Matplotlib и Seaborn для создания визуализаций.

Настройка окружения PySpark

In [1]:
```
!pip install pyspark pandas matplotlib seaborn
```

```
Requirement already satisfied: pyspark in /usr/local/spark-2
Requirement already satisfied: pandas in /opt/conda/lib/pyth
Requirement already satisfied: matplotlib in /opt/conda/lib/
Requirement already satisfied: seaborn in /opt/conda/lib/pyt
Requirement already satisfied: py4j==0.10.7 in /opt/conda/li
Requirement already satisfied: python-dateutil>=2.6.1 in /op
Requirement already satisfied: numpy>=1.13.3 in /opt/conda/l
Requirement already satisfied: pytz>=2017.2 in /opt/conda/li
Requirement already satisfied: cycler>=0.10 in /opt/conda/li
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.
lib) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/con
Requirement already satisfied: scipy>=1.0.1 in /opt/conda/li
Requirement already satisfied: six>=1.5 in /opt/conda/lib/py
0)
```

In [6]:
```
from pyspark.sql import SparkSession
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Подключение к кластеру Spark

```
In [13]: # Создание SparkSession
         spark = SparkSession.builder \
             .master("spark://spark-master:7077") \
             .appName("WordCount") \
             .getOrCreate()
```

Загрузка данных

In [18]:
```
# Чтение CSV файла
df = spark.read.csv("hdfs://namenode:8020/airports.dat", header=False, inferSchema=True)
```

In [19]:
```
# Присвоение имен колонкам
df = df.withColumnRenamed("_c0", "id") \
       .withColumnRenamed("_c1", "name") \
       .withColumnRenamed("_c2", "city") \
       .withColumnRenamed("_c3", "country") \
       .withColumnRenamed("_c4", "iata_code") \
       .withColumnRenamed("_c5", "icao_code") \
       .withColumnRenamed("_c6", "latitude") \
       .withColumnRenamed("_c7", "longitude") \
       .withColumnRenamed("_c8", "altitude") \
       .withColumnRenamed("_c9", "timezone") \
       .withColumnRenamed("_c10", "type") \
       .withColumnRenamed("_c11", "region") \
       .withColumnRenamed("_c12", "object_type") \
       .withColumnRenamed("_c13", "source")
```

Обработка данных

In [35]:
```
# Подсчет количества аэропортов по странам
country_counts_spark = df.groupBy("country").count()
```

In [36]:
```
from pyspark.sql.functions import col
```

In [41]:
```
# Получение 5 стран с самым большим количеством аэропортов
top_countries = country_counts_spark.orderBy(col("count").desc()).limit(5)

# Вывод результата
top_countries.show()
```

```
+-------------+-----+
|      country|count|
+-------------+-----+
|United States| 1512|
|       Canada|  430|
|    Australia|  334|
|       Russia|  264|
|       Brazil|  264|
+-------------+-----+
```

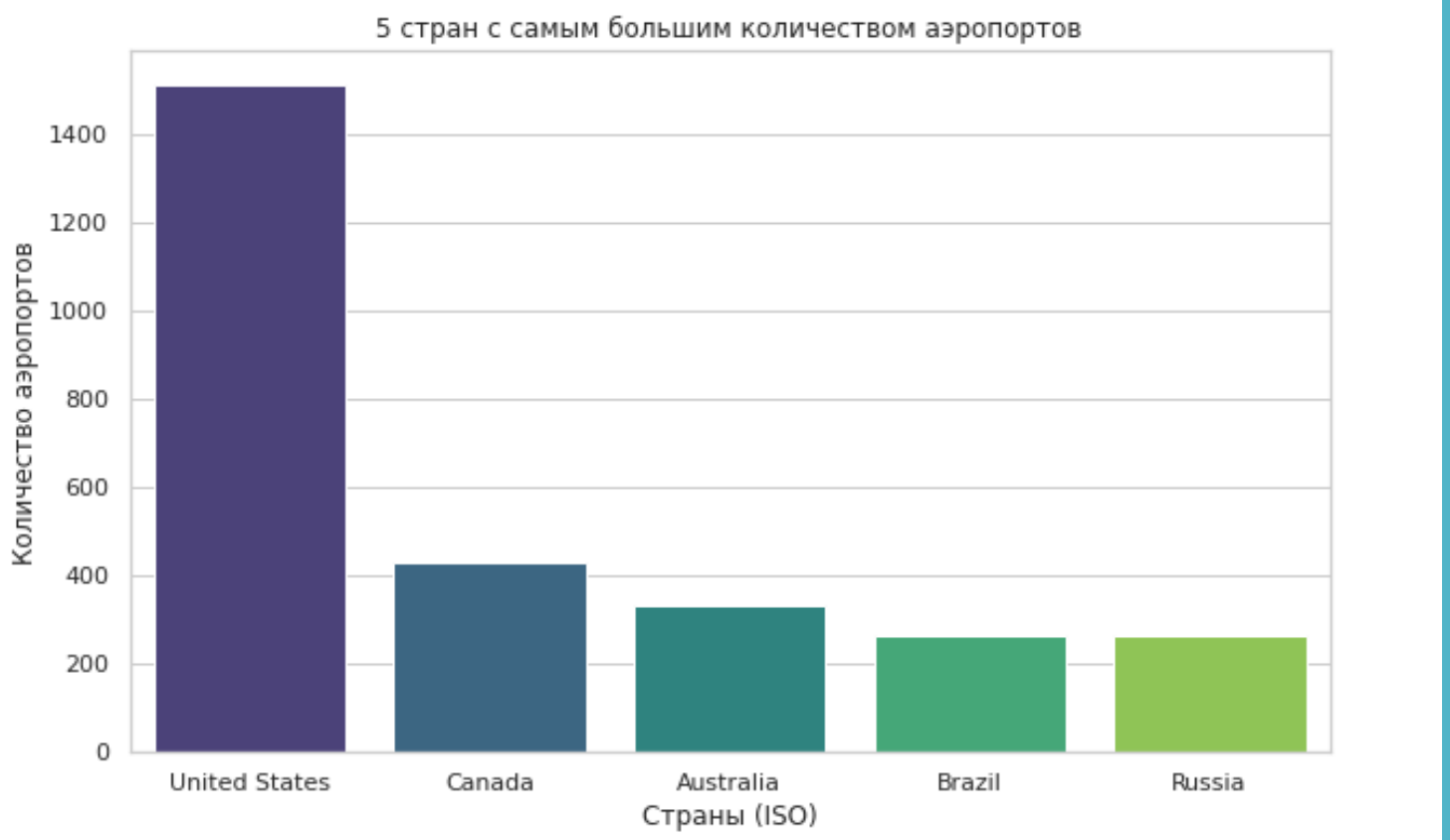
Преобразование данных для визуализации

In [38]:
```
# Преобразование в Pandas DataFrame
top_countries_pandas = top_countries_spark.toPandas()

# Вывод результата
print(top_countries_pandas)
```

```
          country  count
0  United States   1512
1         Canada    430
2      Australia    334
3         Brazil    264
4         Russia    264
```

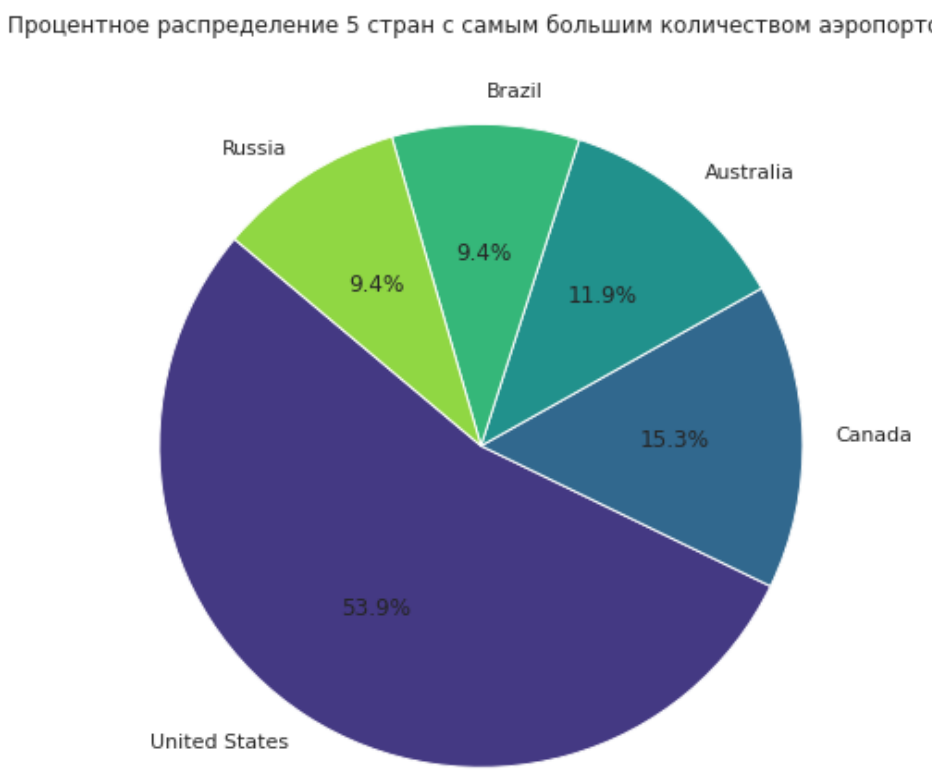
Создание графиков

In [42]:
```
# Визуализация с помощью Seaborn
plt.figure(figsize=(10, 6))
sns.barplot(x='country', y='count', data=top_countries_pandas, palette='viridis')
plt.title('5 стран с самым большим количеством аэропортов')
plt.xlabel('Страны (ISO)')
plt.ylabel('Количество аэропортов')
plt.show()
```



Создание графиков

In [30]:
```
# Визуализация с помощью Matplotlib: круговая диаграмма
plt.figure(figsize=(8, 8))
plt.pie(
    top_countries_pandas['count'],
    labels=top_countries_pandas['country'],
    autopct='%1.1f%%',
    startangle=140,
    colors=sns.color_palette("viridis", len(top_countries_pandas))
)
plt.title('Процентное распределение 5 стран с самым большим количеством аэропортов')
plt.show()
```



Домашнее задание:

1. Повторить материал лекции.

Список литературы:

1. В. Ю. Кара-ушанов SQL — язык реляционных баз данных
2. А. Б. ГРАДУСОВ. Введение в технологию баз данных
3. А.Мотеев. Уроки MySQL

Материалы лекций:

https://github.com/ShViktor72/Education

Обратная связь:

colledge20education23@gmail.com