**Bài thực hành số 6.**

**Chuẩn bị dữ liệu**

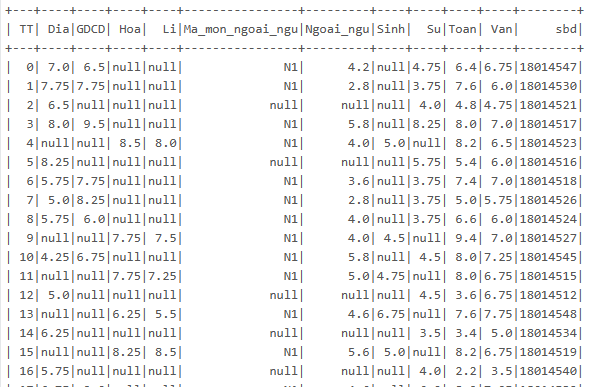
**Link bài là trên databrick:** [Bài làm](https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/345358637122346/4304257940740995/5017236833160125/latest.html)

**Bài 1. Khám phá điểm thi**

**1.1 Tạo DataFrame từ dữ liệu điểm thi THPT Quốc gia 2020.**

df = spark.read.csv('/FileStore/KTGK/diemthi2020.csv', header=True)

df.show()



**1.2 Thống kê sơ bộ về điểm thi các môn.**

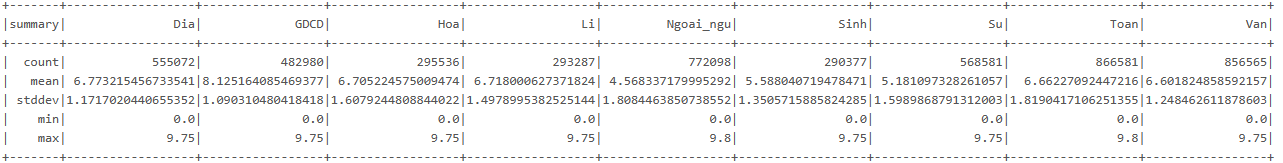
# Lấy ra danh sách cột các môn học và bỏ đi cột Ma\_mon\_ngoai\_ngu

subject\_columns = df.columns[1:-1]

subject\_columns = subject\_columns[:4] + subject\_columns[5:]

desc = df.describe(subject\_columns)

desc.show()

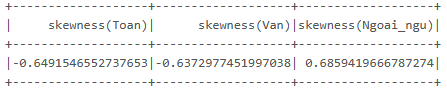


**1.3 Tính độ lệch, độ nhọn của phân bố điểm thi 3 môn: Toán, Văn, Ngoại ngữ.**

Độ Lệch

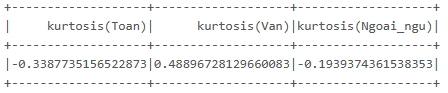
from pyspark.sql.functions import skewness, kurtosis

df.select(skewness('Toan'), skewness('Van'), skewness('Ngoai\_ngu')).show()



Độ Nhọn

df.select(kurtosis('Toan'), kurtosis('Van'), kurtosis('Ngoai\_ngu')).show()



**1.4 Vẽ biểu đồ tần suất của từng điểm thi với các mức cách nhau 0.5 điểm theo 3 cách:**

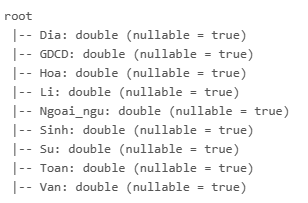
Tạo một dataframe với các cột ‘subject\_column’

df\_subject = df.select(subject\_columns)

for col in df\_subject.columns:

    df\_subject = df\_subject.withColumn(col, df\_subject[col].cast(DoubleType()))

df\_subject.printSchema()



Thay thế giá trị null thành giá trị trung bình cho tất cả các môn học trước khi vẽ biểu đồ

# Tính giá trị trung bình của từng môn học và lưu vào mean\_vals

mean\_vals = df.select([F.mean(col).alias(col) for col in subject\_columns]).collect()[0]

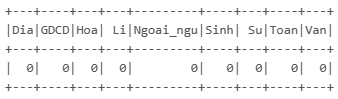
# Thay thế giá trị null bằng giá trị trung bình

for col in subject\_columns:

    df\_subject = df\_subject.withColumn(col, F.coalesce(df\_subject[col], F.lit(mean\_vals[col])))

#Kiểm tra số giá trị null của các cột

df\_subject.select([F.sum(F.col(col).isNull().cast("int")).alias(col) for col in df\_subject.columns]).show()



a) Cách 1: Gộp dữ liệu trên workers rồi thống kê từng nhóm ở Drive

import matplotlib.pyplot as plt

import numpy as np

# Tạo một lưới biểu đồ với 3 hàng và 3 cột để vẽ 9 biểu đồ cho 9 môn học

fig, axes = plt.subplots(3, 3, figsize=(30, 20))

# Lặp qua từng môn học và vẽ biểu đồ cho mỗi môn

for i, col in enumerate(df\_subject.columns):

    hist\_worker = df\_subject.select(col).rdd.flatMap(lambda row: row).histogram(20)

    data = {'bins': hist\_worker[0][:-1],

            'freq': hist\_worker[1]}

    # Tính toán vị trí của biểu đồ trong lưới 3x3

    row\_position = i // 3

    col\_position = i % 3

    # Vẽ biểu đồ tại vị trí đã tính toán

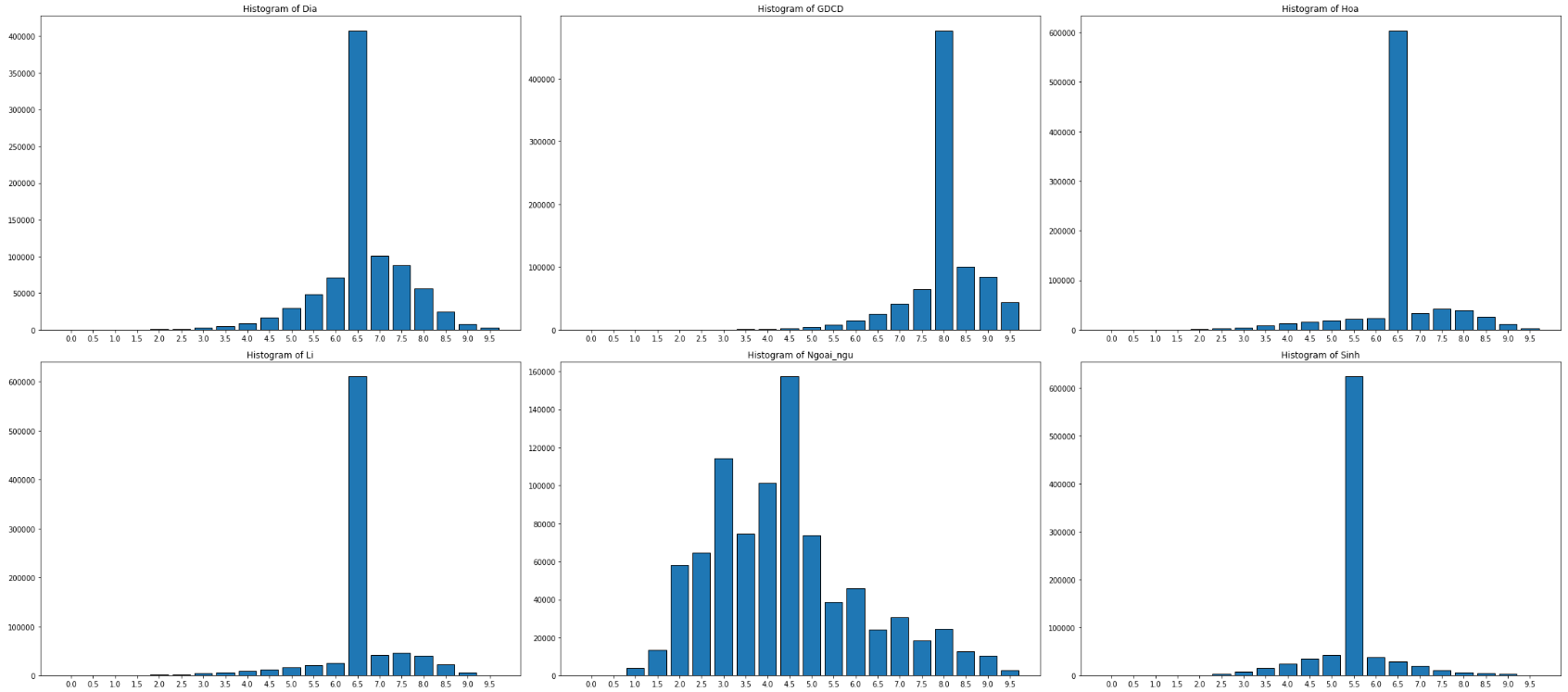
    axes[row\_position, col\_position].bar(data['bins'], data['freq'], width=0.4, edgecolor='black')

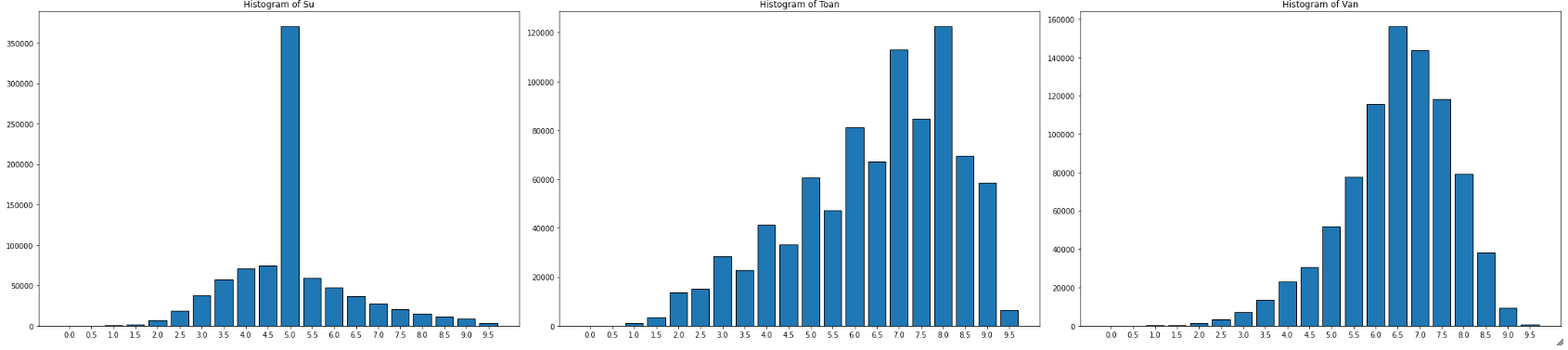
    axes[row\_position, col\_position].set\_xticks(data['bins'])

    axes[row\_position, col\_position].set\_title(f'Histogram of {col}')

plt.tight\_layout()

plt.show()





b) Cách 2: Lấy toàn bộ dữ liệu về Drive để vẽ biểu đồ tần suất

fig, axes = plt.subplots(3, 3, figsize=(30, 20))

for i, col in enumerate(df\_subject.columns):

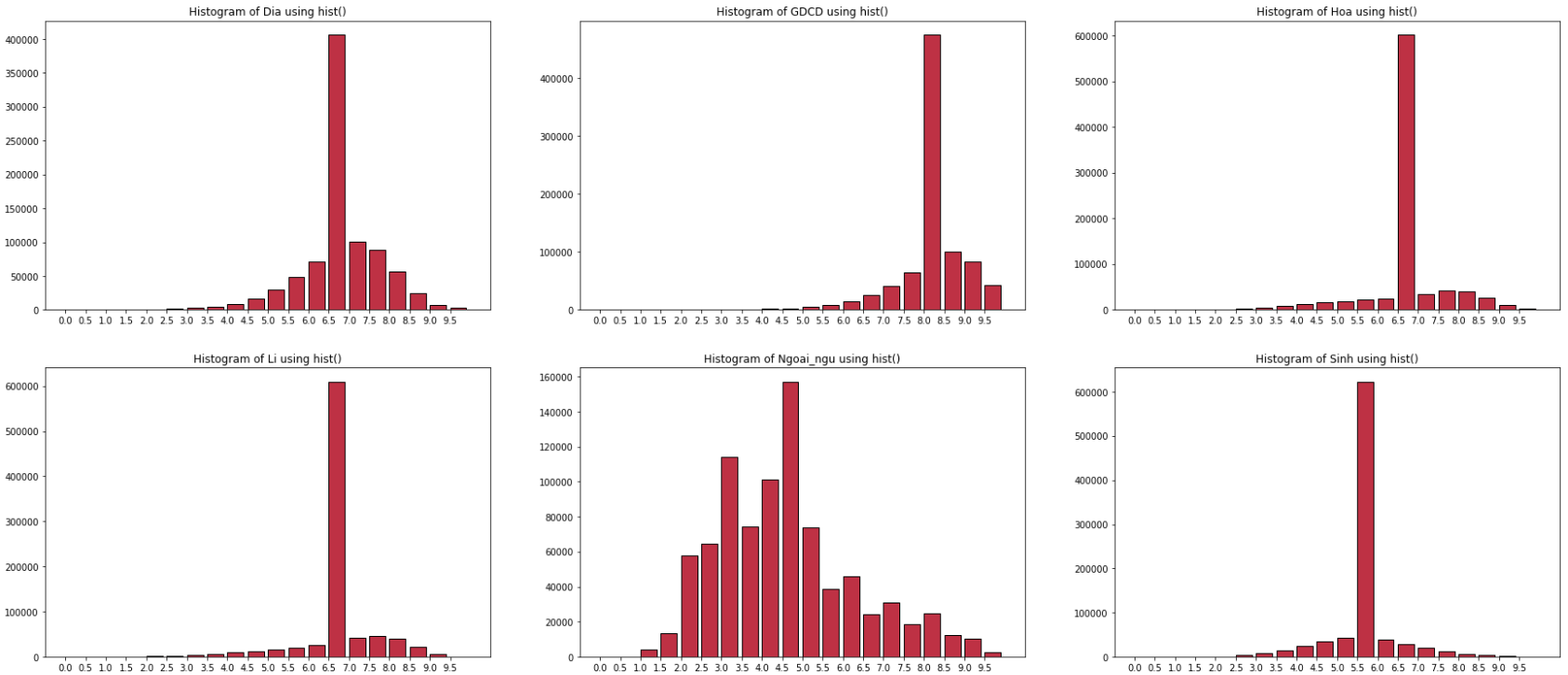
    data\_driver = {'obs': df\_subject.select(col).rdd.flatMap(lambda row: row).collect()}

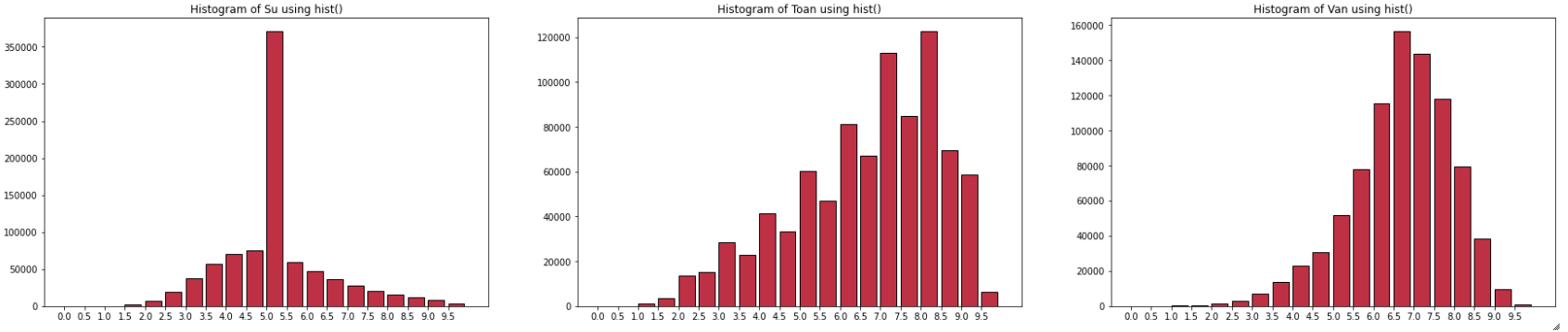
    row\_position = i // 3

    col\_position = i % 3

    axes[row\_position, col\_position].hist(data\_driver['obs'], bins=20, width=0.4, color='#BE3144', edgecolor='black')

    axes[row\_position, col\_position].set\_title(f'Histogram of {col} using hist()')





c) Cách 3: Lấy mẫu 2% theo điểm thi rồi thống kê và vẽ biểu đồ.

from pyspark.sql.functions import col, round

fig, axes = plt.subplots(3, 3, figsize=(30, 20))

for i, sub\_col in enumerate(df\_subject.columns):

    # Làm tròn các giá trị trong cột đến 0.5

    df\_subject\_rounded = df\_subject.withColumn(sub\_col, round(col(sub\_col) \* 2) / 2)

    # Lấy mẫu 2% của từng mức điểm và lưu vào dict fractions

    fractions = {value: 0.02 for value in df\_subject\_rounded.select(sub\_col).distinct().rdd.flatMap(lambda x: x).collect()}

    # Lấy mẫu dựa trên tỷ lệ đã tính toán

    data\_sample = df\_subject\_rounded.sampleBy(sub\_col, fractions).select(sub\_col)

    data\_values = data\_sample.rdd.flatMap(lambda x: x).collect()

    # Vẽ biểu đồ histogram

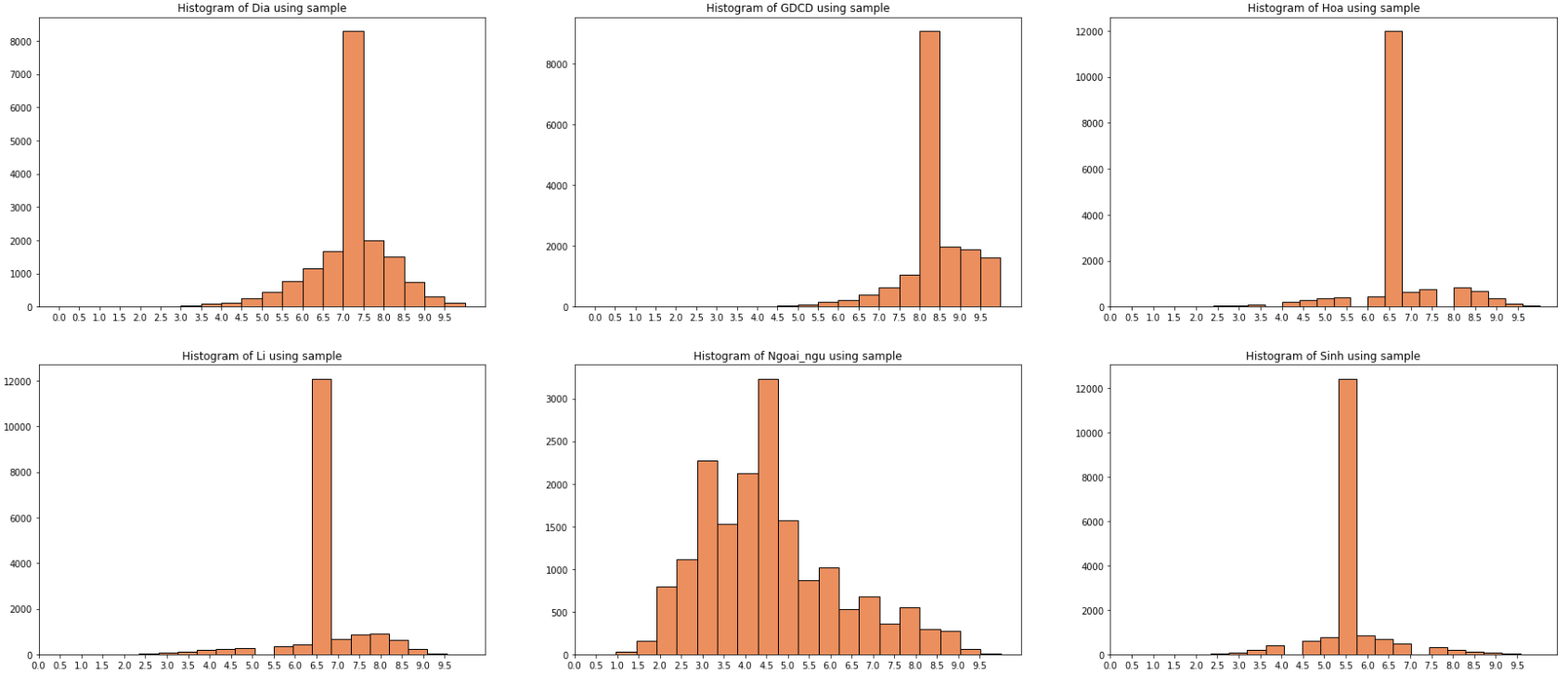
    row\_position = i // 3

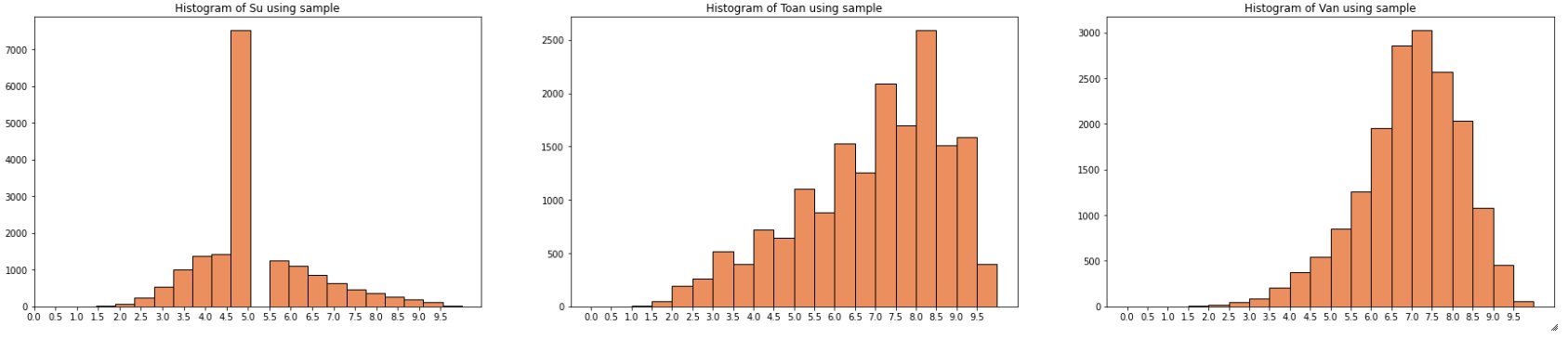
    col\_position = i % 3

    axes[row\_position, col\_position].hist(data\_values, bins=20, color='#EC8F5E', edgecolor='black')

    axes[row\_position, col\_position].set\_xticks(data['bins'])

    axes[row\_position, col\_position].set\_title(f'Histogram of {sub\_col} using sample')





=> So sánh các biểu đồ về mức độ tương đồng, thời gian xử lý, với 3 cách làm trên:

- Về độ tương đồng của biểu đồ là như nhau. Riêng cách 3, cách làm lấy mẫu, tại làm tròn điểm đến 0.5, cách làm tròn chưa khớp hoàn toàn với 2 cách trước nên giá trị bị lệch.

- Về thời gian thực thi theo thứ tự từ nhanh đến chậm: Cách 3 (1.20 phút) 🡪 Cách 2 (1.38 phút)🡪 Cách 1. (2.58 phút).

**1.5 Liệt kê SBD và điểm của những trường hợp outlier cho điểm thi từng môn.**

bounds = {}

for col in subject\_columns:

    quantiles = df\_subject.approxQuantile(col, [0.25, 0.75], 0.05)

    IQR = quantiles[1] - quantiles[0]

    bounds[col] = [

        quantiles[0] - 1.5 \* IQR,

        quantiles[1] + 1.5 \* IQR

    ]



outliers = df.select(\*['sbd']+[((df[c] < bounds[c][0]) | (df[c] >

bounds[c][1])).alias(c + '\_o') for c in subject\_columns])

outliers.show()

df\_outliers = df.join(outliers, on='sbd')

# Tạo một dict rỗng

list\_df\_outliers = {}

# Lặp qua từng môn và lọc lấy giá trị outlier và lưu vào dataframe, sau đấy thêm dataframe và dict list\_df\_outliers

for col in subject\_columns:

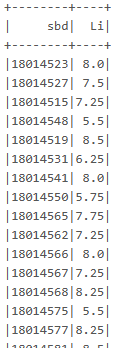
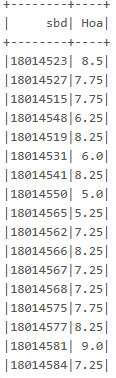
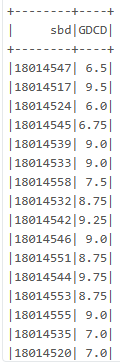
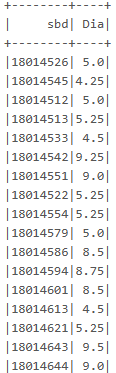
    df\_col\_outliers = df\_outliers.filter(f'{col}\_o').select('sbd', col)

    list\_df\_outliers[f'df\_{col}'] = df\_col\_outliers

for key, value in list\_df\_outliers.items():

    print(f'{key}:')

    value.show()

…

**1.6 Vẽ biểu đồ hộp (Boxplot) cho điểm thi các môn.**

import seaborn as sns

# Chuyển df\_subject về pandas

df\_subject\_pandas = df\_subject.toPandas()

plt.figure(figsize=(25, 15))

sns.boxplot(data=df\_subject\_pandas, showfliers=False)

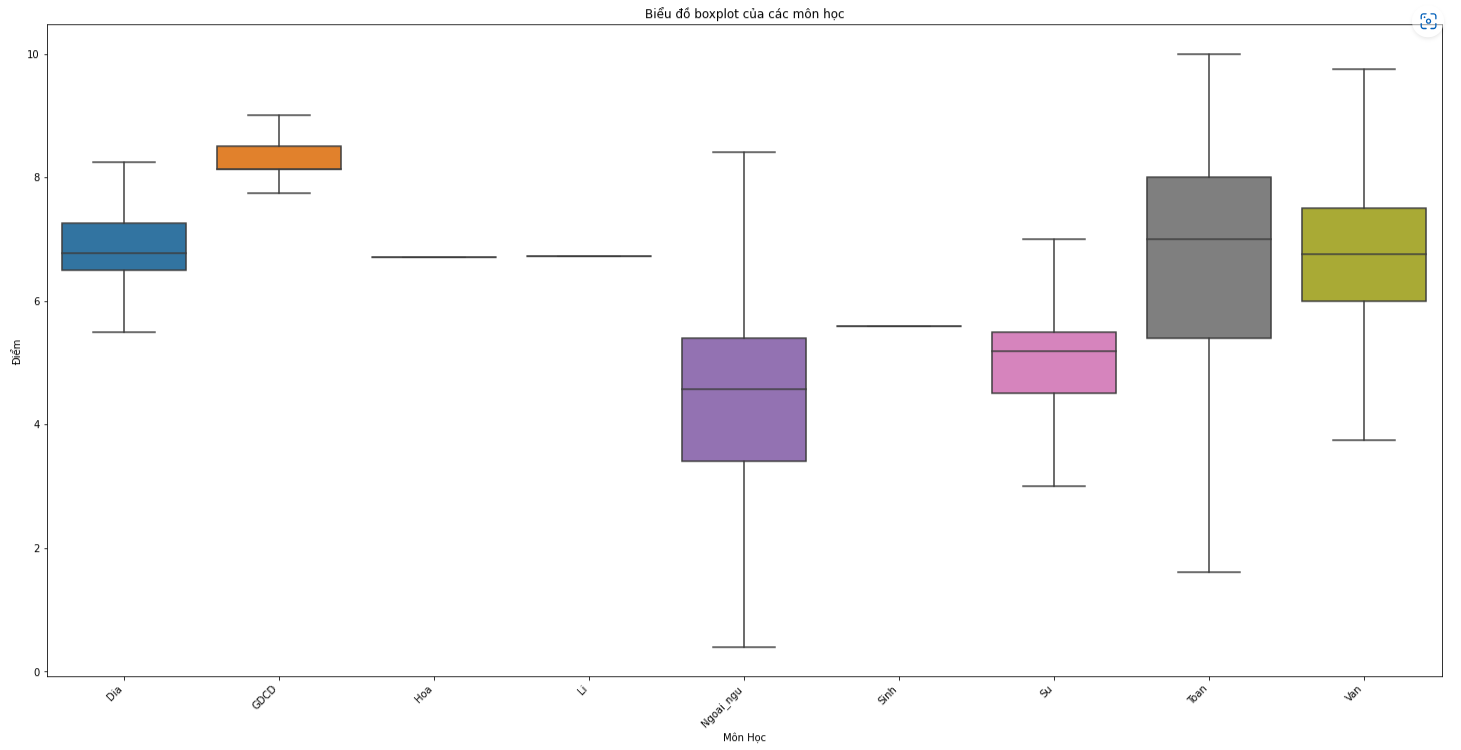
plt.title('Biểu đồ boxplot của các môn học')

plt.xlabel('Môn Học')

plt.ylabel('Điểm')

plt.xticks(rotation=45, ha='right')

plt.show()



**1.7 Tính độ tương quan điểm thi từng cặp môn.**

n\_subjects = len(subject\_columns)

corr = []

for i in range(0, n\_subjects):

    temp = [None] \* i

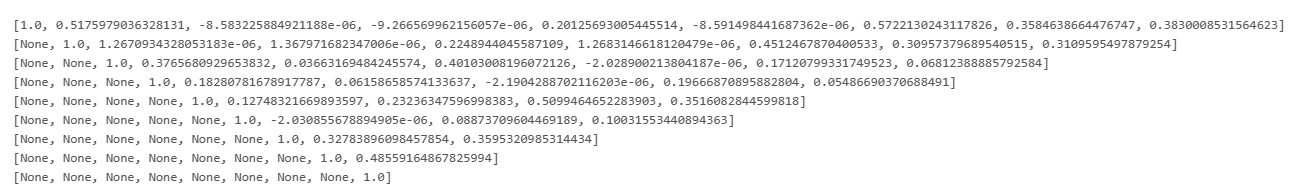
    for j in range(i, n\_subjects):

        temp.append(df\_subject.corr(subject\_columns[i], subject\_columns[j]))

    corr.append(temp)

for i in corr:

    print(i)



**1.8 Vẽ biểu đồ phân tán thể hiện phân bố điểm tương quan giữa hai môn Toán**

**và Văn.**

plt.figure(figsize=(12, 12))

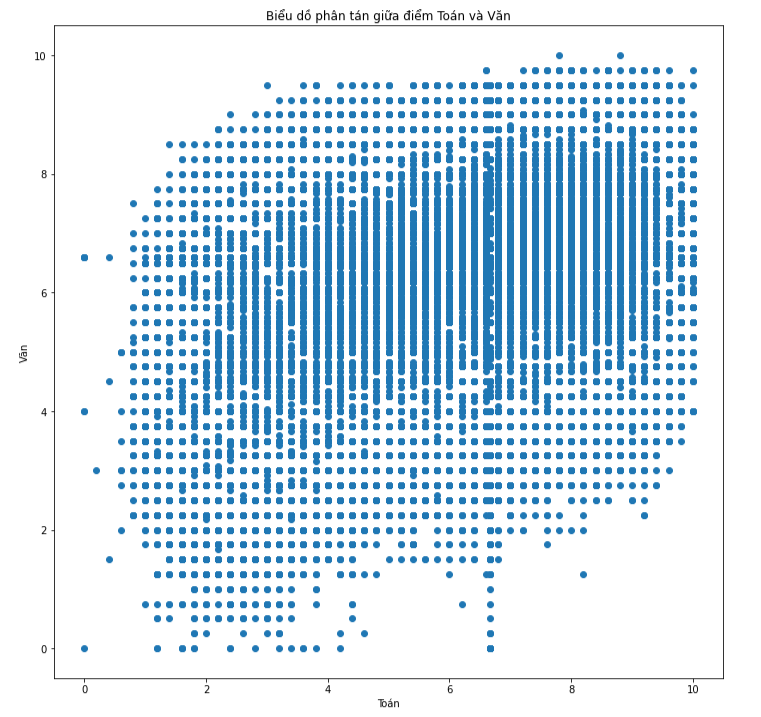
plt.scatter(df\_subject\_pandas['Toan'], df\_subject\_pandas['Van'])

plt.title('Biểu dồ phân tán giữa điểm Toán và Văn')

plt.xlabel('Toán')

plt.ylabel('Văn')

plt.show()

****

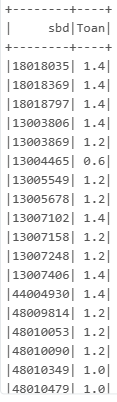
**Bài 2. Khám phá tương quan điểm giữa các tỉnh**

from pyspark.sql.functions import col, collect\_list, size

# Lấy Dataframe df\_Toan từ chứa các outlier điểm của môn toán trong danh sách list\_df\_outlies trong bài trên

df\_toan\_outlier = list\_df\_outliers['df\_Toan']

df\_toan\_outlier.show(20)

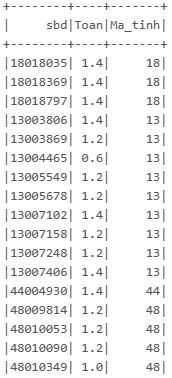
****

# Tách mã tỉnh từ 2 kí tự đầu tiên trong sbd vào cột mới Ma\_tinh

df\_toan\_outlier = df\_toan\_outlier.withColumn('Ma\_tinh', col('sbd').substr(1, 2))

df\_toan\_outlier = df\_toan\_outlier.withColumn('Toan', df\_toan\_outlier['Toan'].cast(DoubleType()))

df\_toan\_outlier.show()

****

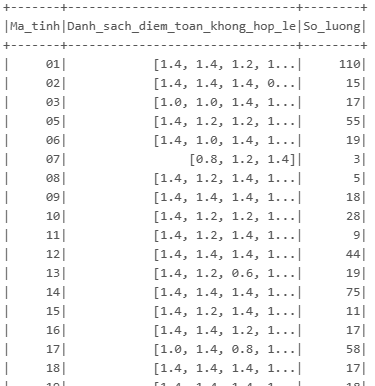
# Lọc điểm không hợp lệ theo mã tỉnh

df\_toan\_outlier\_by\_ma\_tinh = df\_toan\_outlier.groupBy('Ma\_tinh')\

                                .agg(collect\_list('Toan').alias('Danh\_sach\_diem\_toan\_khong\_hop\_le'), size(collect\_list('Toan')).alias('So\_luong'))\

                                .orderBy('Ma\_tinh')

df\_toan\_outlier\_by\_ma\_tinh.show()



**2.2 Dùng biểu đồ hộp để minh họa phân bố dữ liệu điểm môn toán của 10 tỉnh.**

from pyspark.sql.functions import mean

# Tạo dataframe chứa điểm toán. sdb và một số tính toán cần thiết trước khi vẽ biểu đồ

df\_toan = df.select('sbd', 'Toan').withColumn('Toan', df['Toan'].cast(DoubleType()))

mean\_toan = df\_toan.select(mean(col('Toan'))).collect()[0][0]

df\_toan.withColumn('Toan', F.when(col('Toan').isNotNull(), col('Toan')).otherwise(mean\_toan))

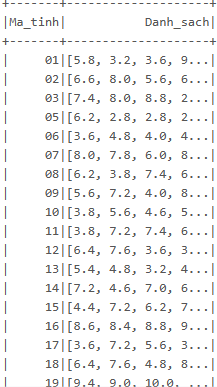
df\_toan = df\_toan.withColumn('Ma\_tinh', df['sbd'].substr(1, 2))\

        .groupBy('Ma\_tinh')\

        .agg(collect\_list('Toan').alias('Danh\_sach'))\

        .orderBy('Ma\_tinh')

df\_toan.show()



df\_toan\_pandas = df\_toan.limit(10).toPandas()

plt.figure(figsize=(25, 12))

plt.boxplot(df\_toan\_pandas['Danh\_sach'], patch\_artist=True)

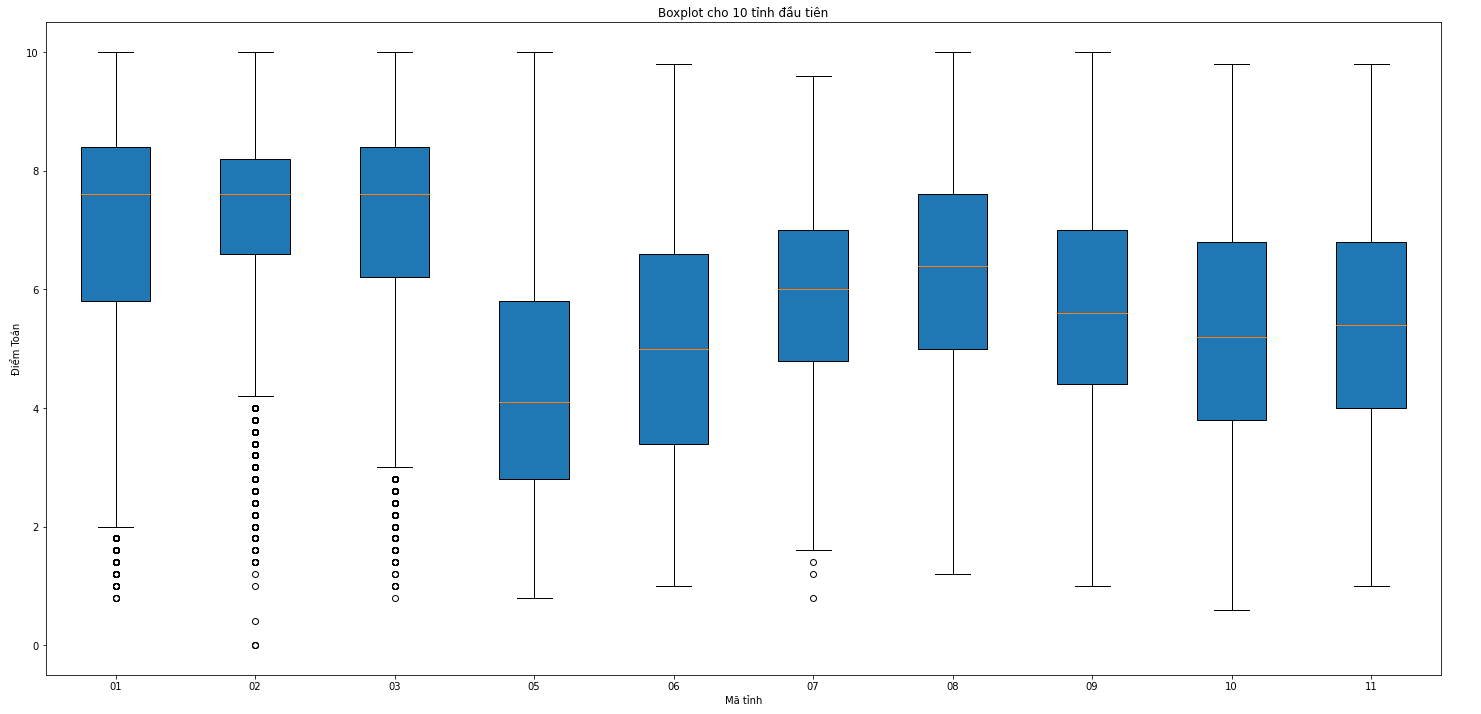
plt.xticks(range(1, len(df\_toan\_pandas['Ma\_tinh']) + 1), df\_toan\_pandas['Ma\_tinh'])

plt.title('Boxplot cho 10 tỉnh đầu tiên')

plt.xlabel('Mã tỉnh')

plt.ylabel('Điểm Toán')

plt.show()



**2.3. Tổng quát, tìm những điểm ngoại lệ của thí sinh từng tỉnh ở tất cả các môn.**

from functools import reduce

from pyspark.sql.functions import array\_remove

# sử dụng lại list\_df\_outliers chứa điểm ngoại lai của các môn ở câu trước, thực hiện join tất cả các dataframe thành một dataframe chứa sbd, điểm ngoại lai của các môn học

# sử dụng thư viện reduce để thực hiện join từng cặp dataframe

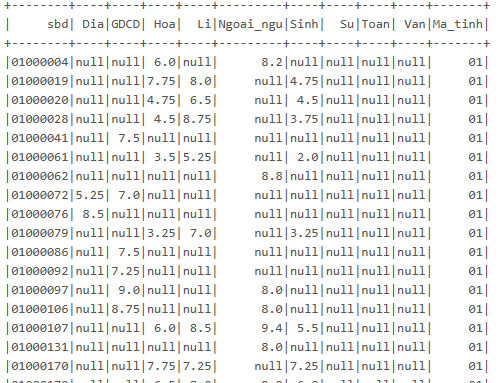
def join\_dataframes(df1, df2):

    return df1.join(df2, on='sbd', how='full')

df\_outliers\_diem = reduce(join\_dataframes, list\_df\_outliers.values())

df\_outliers\_diem = df\_outliers\_diem.withColumn('Ma\_tinh', df\_outliers\_diem['sbd'].substr(1, 2))

df\_outliers\_diem.show()

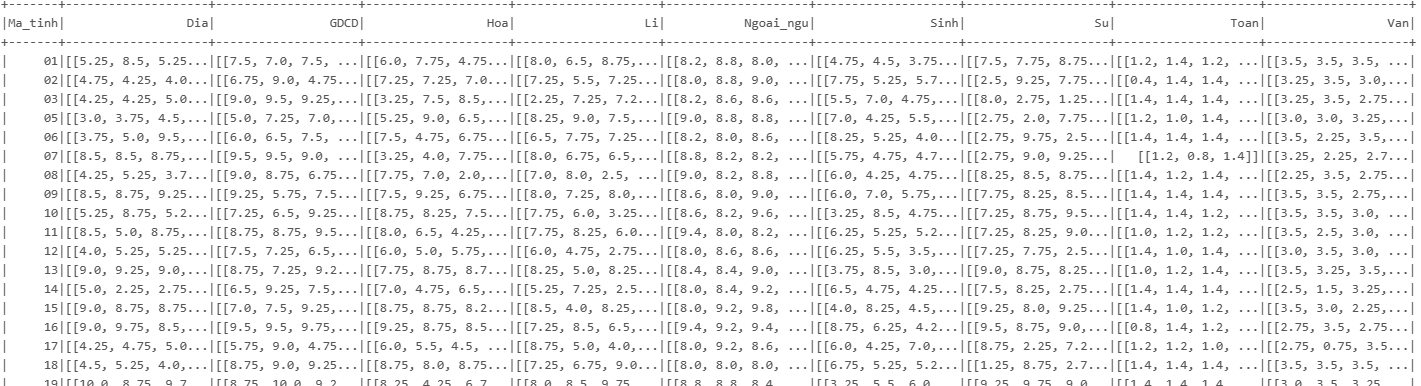
****

df\_outliers\_diem =

df\_outliers\_diem.groupBy('Ma\_tinh').agg(\*[collect\_list(col(c)).alias(c) for c in

subject\_columns])

df\_outliers\_diem.orderBy('Ma\_tinh').show()

****

**Bài 3. Thống kê dữ liệu theo khối**

**3.1 Vẽ biểu đồ tần suất phân bố điểm từng khối A, B, C, D với 30 mức điểm từ 0 đến 30.**

Tính điểm khối A:

df\_diem\_khoi\_a = df.select('sbd', 'Toan', 'Li', 'Hoa')

df\_diem\_khoi\_a = df\_diem\_khoi\_a.filter(df\_diem\_khoi\_a["Toan"].isNotNull() &

                                        df\_diem\_khoi\_a["Li"].isNotNull() &

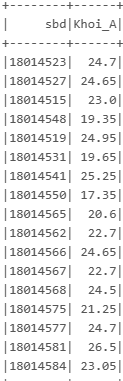
                                        df\_diem\_khoi\_a["Hoa"].isNotNull())

df\_khoi\_A = df\_diem\_khoi\_a.select('sbd', 'Toan', 'Li', 'Hoa')\

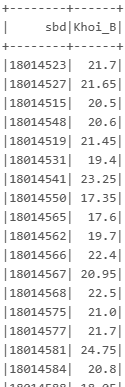
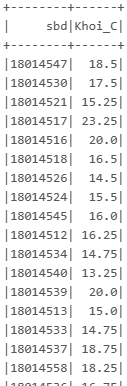
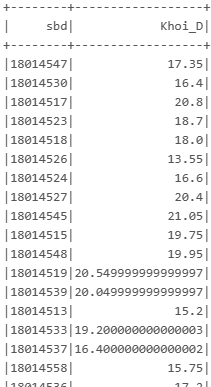
    .withColumn('Khoi\_A', df\_diem\_khoi\_a['Toan'] + df\_diem\_khoi\_a['Li'] + df\_diem\_khoi\_a['Hoa'])

df\_khoi\_A = df\_khoi\_A.drop(\*('Toan', 'Li', 'Hoa'))

df\_khoi\_A.show()



Tương tự với các khối điểm còn lại

Làm tròn điểm

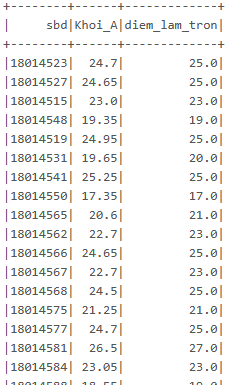
df\_khoi\_A\_round = df\_khoi\_A.withColumn("diem\_lam\_tron", round(df\_khoi\_A["Khoi\_A"], 0))

df\_khoi\_B\_round = df\_khoi\_B.withColumn("diem\_lam\_tron", round(df\_khoi\_B["Khoi\_B"], 0))

df\_khoi\_C\_round = df\_khoi\_C.withColumn("diem\_lam\_tron", round(df\_khoi\_C["Khoi\_C"], 0))

df\_khoi\_D\_round = df\_khoi\_D.withColumn("diem\_lam\_tron", round(df\_khoi\_D["Khoi\_D"], 0))

df\_khoi\_A\_round.show()



Vẽ biểu đồ

fig, axes = plt.subplots(2, 2, figsize=(30, 20))

diem\_khoi\_a = df\_khoi\_A\_round.select('diem\_lam\_tron').rdd.flatMap(lambda x: x).collect()

diem\_khoi\_b = df\_khoi\_B\_round.select('diem\_lam\_tron').rdd.flatMap(lambda x: x).collect()

diem\_khoi\_c = df\_khoi\_C\_round.select('diem\_lam\_tron').rdd.flatMap(lambda x: x).collect()

diem\_khoi\_d = df\_khoi\_D\_round.select('diem\_lam\_tron').rdd.flatMap(lambda x: x).collect()

diem\_khoi\_a\_list = []

for i in range(31):

    diem\_khoi\_a\_list.append(diem\_khoi\_a.count(i))

diem\_khoi\_b\_list = []

for i in range(31):

    diem\_khoi\_b\_list.append(diem\_khoi\_b.count(i))

diem\_khoi\_c\_list = []

for i in range(31):

    diem\_khoi\_c\_list.append(diem\_khoi\_c.count(i))

diem\_khoi\_d\_list = []

for i in range(31):

    diem\_khoi\_d\_list.append(diem\_khoi\_d.count(i))

axes[0, 0].bar([i for i in range(31)], diem\_khoi\_a\_list, color='#EC8F5E', edgecolor='black')

axes[0, 0].set\_title("Điểm Khối A")

axes[0, 1].bar([i for i in range(31)], diem\_khoi\_b\_list, color='#BE3144', edgecolor='black')

axes[0, 1].set\_title("Điểm Khối B")

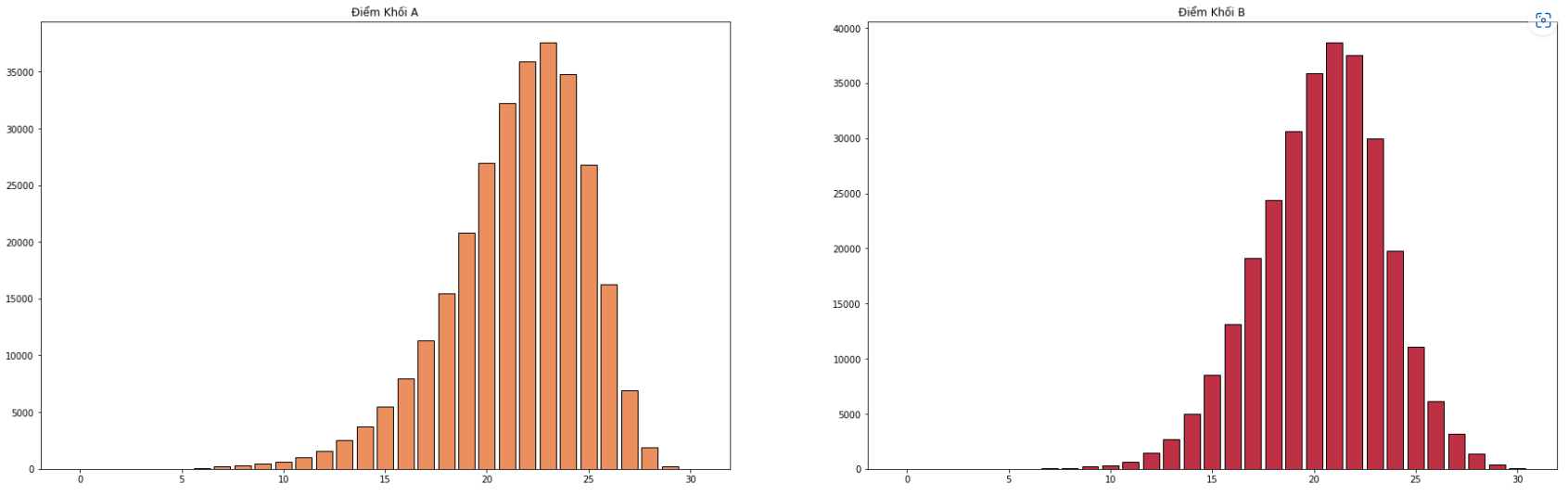
axes[1, 0].bar([i for i in range(31)], diem\_khoi\_c\_list, color='#508D69', edgecolor='black')

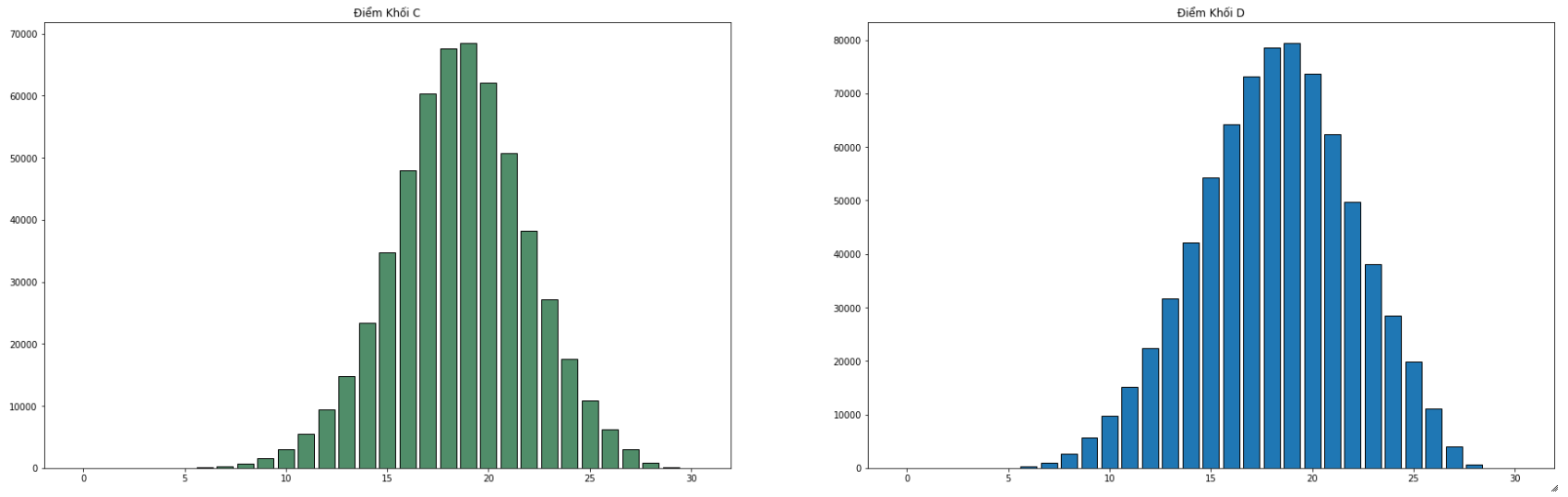
axes[1, 0].set\_title("Điểm Khối C")

axes[1, 1].bar([i for i in range(31)], diem\_khoi\_d\_list, edgecolor='black')

axes[1, 1].set\_title("Điểm Khối D")

plt.show()





**3.2 Tính độ lệch, độ nhọn của phân bố điểm thi các khối thi.**

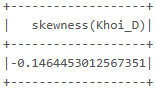
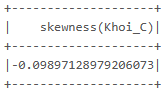
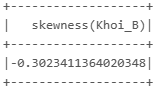
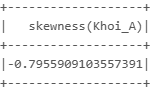
Độ lệch

df\_khoi\_A.select(skewness('Khoi\_A')).show()

df\_khoi\_B.select(skewness('Khoi\_B')).show()

df\_khoi\_C.select(skewness('Khoi\_C')).show()

df\_khoi\_D.select(skewness('Khoi\_D')).show()



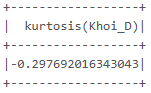
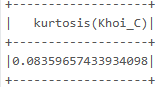
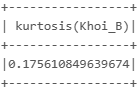
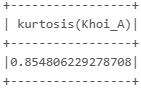
Độ nhọn

df\_khoi\_A.select(kurtosis('Khoi\_A')).show()

df\_khoi\_B.select(kurtosis('Khoi\_B')).show()

df\_khoi\_C.select(kurtosis('Khoi\_C')).show()

df\_khoi\_D.select(kurtosis('Khoi\_D')).show()



**3.3 Vẽ biểu đồ tròn biểu diễn tỷ lệ thí sinh khối A, B, C, D cho những thí sinh từ 15 điểm trở lên.**

# Tính số lượng thí sinh thi từ 15 điểm trờ lên của từng khối

count\_diem\_a\_15 = df\_khoi\_A.filter(col('Khoi\_A') >= 15).count()

count\_diem\_b\_15 = df\_khoi\_B.filter(col('Khoi\_B') >= 15).count()

count\_diem\_c\_15 = df\_khoi\_C.filter(col('Khoi\_C') >= 15).count()

count\_diem\_d\_15 = df\_khoi\_D.filter(col('Khoi\_D') >= 15).count()

print(count\_diem\_a\_15)

****

# Tính tỉ lệ số lượng thí sinh trên 15 điểm của từng khối trên tổng số lượng thí sinh trên 15 điểm

của cả 4 khối

tong\_so\_luong = count\_diem\_a\_15 + count\_diem\_b\_15 + count\_diem\_c\_15 + count\_diem\_d\_15

ti\_le\_a = count\_diem\_a\_15 / tong\_so\_luong

ti\_le\_b = count\_diem\_b\_15 / tong\_so\_luong

ti\_le\_c = count\_diem\_c\_15 / tong\_so\_luong

ti\_le\_d = count\_diem\_d\_15 / tong\_so\_luong

print(ti\_le\_a)

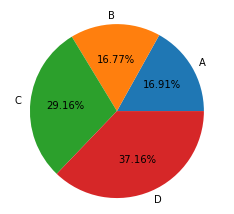
****

ti\_le\_thi\_sinh = [ti\_le\_a, ti\_le\_b, ti\_le\_c, ti\_le\_d]

# Vẽ biểu đồ tròn

plt.pie(ti\_le\_thi\_sinh, labels=["A", "B", "C", "D"], autopct="%.2f%%")

plt.show()

****