Start coding or generate with AI.

Double-click (or enter) to edit

```
# # Internship Task 1: Big Data Analysis using PySpark
# This notebook analyzes a large fraud detection dataset using PySpark to demonstrate scalability.
```

```
# 🔧 Step 1: Install Java (Spark needs Java runtime)
!apt-get install openjdk-11-jdk-headless -qq > /dev/null

# 🔧 Step 2: Download Spark 3.5.1 (you can change version if needed)
!wget -q https://archive.apache.org/dist/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz

# 🔧 Step 3: Extract Spark package
!tar -xzf spark-3.5.1-bin-hadoop3.tgz

# 🔧 Step 4: Install findspark to connect Python with Spark
!pip install -q findspark
```

```
import os
import findspark

# Set environment paths for Java and Spark
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-11-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.5.1-bin-hadoop3"

# Initialize findspark
findspark.init()
```

Double-click (or enter) to edit

```
from google.colab import files
uploaded = files.upload()
```

Choose Files fraud_detec…dataset.csv

- **fraud_detection_bank_dataset.csv**(text/csv) - 5010367 bytes, last modified: 8/8/2021 - 100% done

Saving fraud_detection_bank_dataset.csv to fraud_detection_bank_dataset.csv

```
from pyspark.sql import SparkSession

# Start or get the SparkSession
spark = SparkSession.builder \
    .appName("My Fraud Detection Analysis") \
    .getOrCreate()
```

```
df = spark.read.csv("fraud_detection_bank_dataset.csv", header=True, inferSchema=True)
df.printSchema()
df.show(5)
```

What can I help you build?

```
 |-- col_84: integer (nullable = true)
 |-- col_85: integer (nullable = true)
 |-- col_86: integer (nullable = true)
 |-- col_87: integer (nullable = true)
 |-- col_88: integer (nullable = true)
 |-- col_89: integer (nullable = true)
 |-- col_90: integer (nullable = true)
 |-- col_91: integer (nullable = true)
 |-- col_92: integer (nullable = true)
 |-- col_93: integer (nullable = true)
 |-- col_94: integer (nullable = true)
 |-- col_95: integer (nullable = true)
 |-- col_96: integer (nullable = true)
 |-- col_97: integer (nullable = true)
 |-- col_98: integer (nullable = true)
 |-- col_99: integer (nullable = true)
 |-- col_100: integer (nullable = true)
 |-- col_101: integer (nullable = true)
 |-- col_102: integer (nullable = true)
 |-- col_103: integer (nullable = true)
 |-- col_104: integer (nullable = true)
 |-- col_105: integer (nullable = true)
 |-- col_106: integer (nullable = true)
 |-- col_107: integer (nullable = true)
 |-- col_108: integer (nullable = true)
 |-- col_109: integer (nullable = true)
 |-- col_110: integer (nullable = true)
 |-- col_111: integer (nullable = true)
 |-- targets: integer (nullable = true)

+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+------+------+------+------+------+------+------+------+------
|_c0|col_0|col_1|col_2|col_3|col_4|col_5|col_6|col_7|col_8|col_9|col_10|col_11|col_12|col_13|col_14|col_15|col_16|col_17|col_18|col_19
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+------+------+------+------+------+------+------+------+------
|  0|    9| 1354|    0|   18|    0|    1|    7|    9|    0|    0|     0|     0|     0|     0|     1|     0|     1|     0|     0|     0
|  1|    0|  239|    0|    1|    0|    1|    0|    0|    0|    0|     0|     0|     0|     0|     0|     0|     1|     0|     0|     0
|  2|    0|  260|    0|    4|    0|    3|    6|    0|    0|    0|     0|     0|     0|     0|     1|     1|     0|     0|     0|     0
|  3|   17|  682|    0|    1|    0|    0|    8|   17|    0|    0|     0|     0|     0|     0|     0|     0|     1|     0|     0|     0
|  4|    1|  540|    0|    2|    0|    1|    7|    1|    0|    0|     0|     0|     0|     0|     1|     0|     1|     0|     0|     0
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+------+------+------+------+------+------+------+------+------
only showing top 5 rows
```

Generate print hello world using rot13 Close

```
print("Row count:", df.count())
print("Column count:", len(df.columns))
```

```
Row count: 20468
Column count: 114
```

```
df.describe().show()
```

```
+-------+-----------------+------------------+------------------+-------------------+-----------------+-------------------+-------------
|summary|              _c0|             col_0|             col_1|              col_2|            col_3|              col_4|
+-------+-----------------+------------------+------------------+-------------------+-----------------+-------------------+-------------
|  count|            20468|             20468|             20468|              20468|            20468|              20468|
|   mean|          10233.5|3.2262556185264804|294.79304279851476|0.42002149697088137|2.329343365253078|0.08359390267735001|0.93985733828
| stddev|5908.746990691005| 20.56430794451052| 717.5419842156925|  7.367275061894912|10.06851230174707| 0.8405365704812978| 4.2228960613
|    min|                0|                 0|                 0|                  0|                0|                  0|
|    max|            20467|              2301|             37808|                904|              772|                 54|
+-------+-----------------+------------------+------------------+-------------------+-----------------+-------------------+-------------
```

```
from pyspark.sql.functions import sum

# Get columns where all values = 0
zero_cols = []
for col_name in df.columns:
    if df.select(sum(col_name)).first()[0] == 0:
        zero_cols.append(col_name)

# Drop them
df_clean = df.drop(*zero_cols)
print("Dropped columns:", zero_cols)
```

```
Dropped columns: ['col_8', 'col_9', 'col_10', 'col_11', 'col_12', 'col_18', 'col_19', 'col_20', 'col_21', 'col_35', 'col_51', 'col_52',
```

```
df_clean.columns
```

```
 'col_50',
 'col_54',
 'col_55',
 'col_56',
 'col_57',
 'col_58',
 'col_59',
 'col_60',
 'col_61',
 'col_62',
 'col_63',
 'col_64',
 'col_65',
 'col_66',
 'col_67',
 'col_68',
 'col_69',
 'col_72',
 'col_73',
 'col_74',
 'col_75',
 'col_76',
 'col_77',
 'col_78',
 'col_79',
 'col_80',
 'col_81',
 'col_82',
 'col_83',
 'col_84',
 'col_85',
 'col_86',
 'col_87',
 'col_88',
 'col_89',
 'col_90',
 'col_91',
 'col_92',
 'col_93',
 'col_94',
 'col_95',
 'col_96',
 'col_97',
 'col_98',
 'col_99',
 'col_100',
 'col_101',
 'col_102',
 'col_103',
 'col_104',
 'col_105',
 'col_106',
 'col_107',
 'col_108',
 'col_109',
 'col_110',
 'col_111',
 'targets']
```

```
new_names = [
    "amount", "oldbalanceOrg", "newbalanceOrig", "type",  # first few known fields
    # Add generic names for the rest
] + [f"feature_{i}" for i in range(len(df_clean.columns) - 5)] + ["isFraud"]  # assuming last column is the target

df_clean = df_clean.toDF(*new_names)
```

```
df_clean.show(5)
```

```
+------+-------------+--------------+----+---------+---------+---------+---------+---------+---------+---------+---------+---------+----
|amount|oldbalanceOrg|newbalanceOrig|type|feature_0|feature_1|feature_2|feature_3|feature_4|feature_5|feature_6|feature_7|feature_8|feat
+------+-------------+--------------+----+---------+---------+---------+---------+---------+---------+---------+---------+---------+----
|     0|            9|          1354|   0|       18|        0|        1|        7|        9|        0|        1|        0|        1|
|     1|            0|           239|   0|        1|        0|        1|        0|        0|        0|        0|        0|        1|
|     2|            0|          260|   0|        4|        0|        3|        6|        0|        0|        1|        1|        0|
|     3|           17|          682|   0|        1|        0|        0|        8|       17|        0|        0|        0|        1|
|     4|            1|          540|   0|        2|        0|        1|        7|        1|        0|        1|        0|        1|
+------+-------------+-------------+----+---------+---------+---------+---------+---------+---------+---------+---------+---------+----
only showing top 5 rows
```

```
print("Rows:", df_clean.count())
print("Columns:", len(df_clean.columns))
```

```
Rows: 20468
Columns: 99
```

```
df_clean.printSchema()
```

```
 |-- feature_38: integer (nullable = true)
 |-- feature_39: integer (nullable = true)
 |-- feature_40: integer (nullable = true)
 |-- feature_41: integer (nullable = true)
 |-- feature_42: integer (nullable = true)
 |-- feature_43: integer (nullable = true)
 |-- feature_44: integer (nullable = true)
 |-- feature_45: integer (nullable = true)
 |-- feature_46: integer (nullable = true)
 |-- feature_47: integer (nullable = true)
 |-- feature_48: integer (nullable = true)
 |-- feature_49: integer (nullable = true)
 |-- feature_50: integer (nullable = true)
 |-- feature_51: double (nullable = true)
 |-- feature_52: integer (nullable = true)
 |-- feature_53: integer (nullable = true)
 |-- feature_54: integer (nullable = true)
 |-- feature_55: integer (nullable = true)
 |-- feature_56: integer (nullable = true)
 |-- feature_57: integer (nullable = true)
 |-- feature_58: integer (nullable = true)
 |-- feature_59: integer (nullable = true)
 |-- feature_60: integer (nullable = true)
 |-- feature_61: integer (nullable = true)
 |-- feature_62: integer (nullable = true)
 |-- feature_63: integer (nullable = true)
 |-- feature_64: integer (nullable = true)
 |-- feature_65: integer (nullable = true)
 |-- feature_66: integer (nullable = true)
 |-- feature_67: integer (nullable = true)
 |-- feature_68: integer (nullable = true)
 |-- feature_69: integer (nullable = true)
 |-- feature_70: integer (nullable = true)
 |-- feature_71: integer (nullable = true)
 |-- feature_72: integer (nullable = true)
 |-- feature_73: integer (nullable = true)
 |-- feature_74: integer (nullable = true)
 |-- feature_75: integer (nullable = true)
 |-- feature_76: integer (nullable = true)
 |-- feature_77: integer (nullable = true)
 |-- feature_78: integer (nullable = true)
 |-- feature_79: integer (nullable = true)
 |-- feature_80: integer (nullable = true)
 |-- feature_81: integer (nullable = true)
 |-- feature_82: integer (nullable = true)
 |-- feature_83: integer (nullable = true)
 |-- feature_84: integer (nullable = true)
 |-- feature_85: integer (nullable = true)
 |-- feature_86: integer (nullable = true)
 |-- feature_87: integer (nullable = true)
 |-- feature_88: integer (nullable = true)
 |-- feature_89: integer (nullable = true)
 |-- feature_90: integer (nullable = true)
 |-- feature_91: integer (nullable = true)
 |-- feature_92: integer (nullable = true)
 |-- feature_93: integer (nullable = true)
 |-- isFraud: integer (nullable = true)
```

```
df_clean.groupBy("isFraud").count().show()  #count farud transaction
```

```
+-------+-----+
|isFraud|count|
+-------+-----+
|      1| 5438|
|      0|15030|
+-------+-----+
```

```
df_clean.filter("isFraud == 1").groupBy().avg("amount").show() # avg transation amount for fraud
```

```
+-----------+
|avg(amount)|
+-----------+
|     2718.5|
+-----------+
```

```
df_clean.groupBy("type", "isFraud").count().orderBy("type").show() # fraud count by transaction
```

```
+----+-------+-----+
|type|isFraud|count|
+----+-------+-----+
|   0|      0|12826|
|   0|      1| 5276|
|   1|      0| 1119|
|   1|      1|  102|
|   2|      1|   25|
|   2|      0|  611|
|   3|      1|    8|
|   3|      0|  170|
|   4|      0|   66|
|   4|      1|   15|
|   5|      1|    4|
|   5|      0|   40|
|   6|      0|   40|
|   6|      1|    2|
|   7|      1|    1|
|   7|      0|   26|
|   8|      0|   15|
|   9|      0|   17|
|  10|      0|    9|
|  11|      0|    8|
+----+-------+-----+
only showing top 20 rows
```

Double-click (or enter) to edit

```
# Insights:
# - Transaction Type 0 had the highest volume and most frauds.
# - Fraud was present across nearly all types.
# - Type 1 had a significant fraud rate.
# - 15 columns were dropped due to containing only 0s.

# This shows the dataset had redundant information, and PySpark allowed fast filtering and analysis even for large structured data.
```