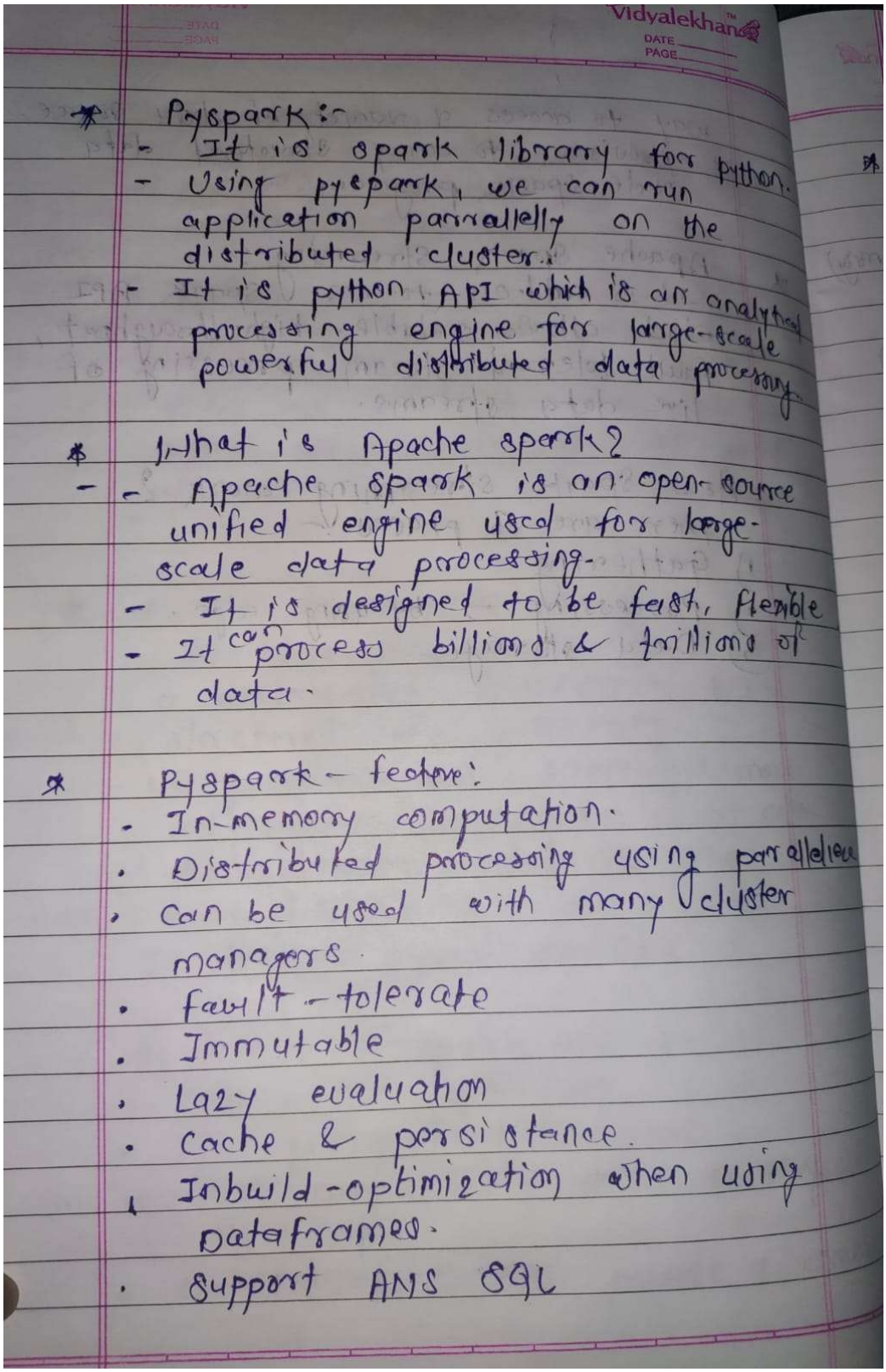
Name: Krushnakumar Patle

Email: krishnapatle128@gmail.com

Batch: Data Engineering Batch-1



* Pyspark :-
- It is spark library for python.
- Using pyspark, we can run application parrallelly on the distributed cluster.
- It is python API which is an analytical processing engine for large-scale powerful distributed data processing.

* 1. What is Apache spark?
- Apache spark is an open-source unified engine used for large-scale data processing.
- It is designed to be fast, flexible
- It can process billions & trillions of data.

* Pyspark - feature:
- In-memory computation.
- Distributed processing using parallelize
- Can be used with many cluster managers
- Fault-tolerate
- Immutable
- Lazy evaluation
- Cache & persistance.
- Inbuild-optimization when using dataframes.
- Support ANS SQL

Advantages of Pyspark:-

- Pyspark is a general-purpose, in-memory distributed processing engine that allows you to process data efficiency in a distributed fashion.
- It is faster ~~than~~ 100x than traditional system.
- It get great benefit for data ingestion pipeline.
- Using it we can process data from Hadoop, AWS, & many more.

Apache Kafka:-

It is an open-source distributed streaming system used for stream processing, real-time data pipeline, and data integration at scale.

Pyspark version:-

Supported - Python 3.8, Java- 8, 11, 13, 17 & latest version are depreciated.
Scala- 2.12 & 2.13
R- 3.5

* Pyspark module:-
- Pyspark RDD (pyspark.RDD)
- Pyspark Dataframe & SQL
- Pyspark streaming (pyspark.streaming)
- Pyspark MLib (pyspark.ml, .mllib)
- Pyspark Graphframes (Graphframes)
- Pyspark Resource (It's new in 3.0)

*

In [1]:
```
import pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("practice").getOrCreate()

spark
```

Out[1]:

SparkSession - in-memory

SparkContext

Spark UI

Version
`v3.5.0`

Master
`local[*]`

AppName
`practice`

In [2]:
```
import pandas as pd
log_data = pd.read_csv("Marks_data.csv")
print(log_data)
```

```
   Name  M1 Score  M2 Score  age
0  Alex      62.0      80.0   20
1  Brad      45.0      56.0   19
2  Joey      85.0      98.0   21
3   NaN      54.0      79.0   20
4  abhi       NaN       NaN   20
```

In [3]:
```
df = spark.read.csv("Marks_data.csv")
df
```

Out[3]:
```
DataFrame[_c0: string, _c1: string, _c2: string, _c3: string]
```

In [4]:
```
df.show()
```

```
+----+--------+--------+---+
| _c0|     _c1|     _c2|_c3|
+----+--------+--------+---+
|Name|M1 Score|M2 Score|age|
|Alex|      62|      80| 20|
|Brad|      45|      56| 19|
|Joey|      85|      98| 21|
|NULL|      54|      79| 20|
|abhi|    NULL|    NULL| 20|
+----+--------+--------+---+
```

In [5]:
```
dataList = [("Java", 20000), ("Python", 100000), ("Scala", 3000)]
rdd=spark.sparkContext.parallelize(dataList)
rdd.collect()
```

Out[5]:
```
[('Java', 20000), ('Python', 100000), ('Scala', 3000)]
```