Product table:

```
+------------+--------------+------------+
| product_id | product_name | unit_price |
+------------+--------------+------------+
| 1          | S8           | 1000       |
| 2          | G4           | 800        |
| 3          | iPhone       | 1400       |
+------------+--------------+------------+
```

Sales table:

```
+-----------+------------+----------+------------+----------+-------+
| seller_id | product_id | buyer_id | sale_date  | quantity | price |
+-----------+------------+----------+------------+----------+-------+
| 1         | 1          | 1        | 2019-01-21 | 2        | 2000  |
| 1         | 2          | 2        | 2019-02-17 | 1        | 800   |
| 2         | 1          | 3        | 2019-06-02 | 1        | 800   |
| 3         | 3          | 3        | 2019-05-13 | 2        | 2800  |
+-----------+------------+----------+------------+----------+-------+
```

I phone Sales Analysis:

Install and configure PySpark, Hive, and Hadoop.
Set up Hive tables: Partitioned table for sales data (in Parquet format).
Non-partitioned table for product data.
Verify the ability to read and write data into Hive tables.

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.3.1/hadoop-3.3.1.tar.gz
tar -xvzf hadoop-3.3.1.tar.gz
mv hadoop-3.3.1 /usr/local/hadoop

export HADOOP_HOME=/usr/local/hadoop
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

pip install pyspark

spark.sql.catalogImplementation=hive
spark.sql.warehouse.dir=/user/hive/warehouse
spark.hadoop.hive.metastore.uris=thrift://localhost:9083

import pyspark
```

```
spark =
pyspark.sql.SparkSession.builder.appName("HiveIntegration").enableHiveSupport().getOrCreate
()
print(spark.version)

CREATE TABLE IF NOT EXISTS sales_hive_table (
    seller_id INT,
    product_id INT,
    buyer_id INT,
    quantity INT,
    price INT
)
PARTITIONED BY (sale_date DATE)
STORED AS PARQUET;

ALTER TABLE sales_hive_table ADD PARTITION (sale_date='2019-01-21') LOCATION '/
user/hive/warehouse/sales_hive_table/sale_date=2019-01-21';
ALTER TABLE sales_hive_table ADD PARTITION (sale_date='2019-02-17') LOCATION '/
user/hive/warehouse/sales_hive_table/sale_date=2019-02-17';


CREATE TABLE IF NOT EXISTS product_hive_table (
    product_id INT,
    product_name STRING,
    unit_price INT
)
STORED AS PARQUET;


from pyspark.sql import SparkSession

# Initialize Spark session with Hive support
spark = SparkSession.builder \
    .appName("HiveIntegration") \
    .enableHiveSupport() \
    .getOrCreate()

# Reading the Product table
product_df = spark.sql("SELECT * FROM product_hive_table")
product_df.show()

# Reading the Sales table (with partitioning)
sales_df = spark.sql("SELECT * FROM sales_hive_table WHERE sale_date = '2019-01-21'")
sales_df.show()
```

```
# Sample data for products
product_data = [(1, 'S8', 1000), (2, 'G4', 800), (3, 'iPhone', 1400)]
columns = ['product_id', 'product_name', 'unit_price']

product_df = spark.createDataFrame(product_data, columns)

# Write to the non-partitioned product table
product_df.write.mode("append").insertInto("product_hive_table")

# Sample data for sales
sales_data = [(1, 1, 1, 2, 2000, '2019-01-21'), (1, 2, 2, 1, 800, '2019-02-17')]
sales_columns = ['seller_id', 'product_id', 'buyer_id', 'quantity', 'price', 'sale_date']

sales_df = spark.createDataFrame(sales_data, sales_columns)

# Write to the partitioned sales table
sales_df.write.mode("append").partitionBy("sale_date").format("parquet").saveAsTable("sales_hive_table")
```

```
SELECT * FROM product_hive_table;
SELECT * FROM sales_hive_table WHERE sale_date='2019-01-21';
```