**SUMMARY-DAY12**

**Name:Tejaswini Gokanakonda**

## **Roll no**:**DE142**

## **Date:21-11-2024**

### **Key Sections**

#### **1.Setup and Initialization**

* Imported libraries: pyspark.sql for Spark DataFrames.
* Created a Spark session:
* spark = SparkSession.builder.appName('pyspark - example join').getOrCreate()

### **2.Data Creation**

* Sample data initialized as a list of tuples with fields: Name, DOB, Gender, and Salary.
* Created a DataFrame with column names:
* df = spark.createDataFrame(data=data, schema=columns)

### **3.Renaming Columns**

* Renamed columns using:
* df = df.withColumnRenamed("DOB", "date\_of\_birth")
* df = df.withColumnRenamed("Name", "person\_name")

### **4.Selecting Specific Columns**

1. **Using selectExpr:**
2. data = df.selectExpr("Gender as category", "DOB", "Name as name", "salary")
3. **Using col() for aliasing:**
4. from pyspark.sql.functions import col
5. data = df.select(col("Name"), col("DOB"), col("Gender"), col("salary").alias("Amount"))
6. Displayed DataFrames using .show().

### **5.File Handling in PySpark**

* **File Location:** /FileStore/tables/simple\_zipcodes-1.csv
* **File Type:** CSV

#### **6.Reading CSV Data**

* Loaded CSV into DataFrame:
* df = spark.read.format("csv") \
* .option("inferSchema", "false") \
* .option("header", "false") \
* .option("sep", ",") \
* .load(file\_location)
* **Options Used:**
  + inferSchema: Ensures schema inference is disabled.
  + header: Specifies if the first row contains headers.
  + sep: Defines the delimiter.

#### **7.SQL Table Creation**

* Created a temporary SQL table:
* df.createOrReplaceTempView("tempdata")

#### **8.SQL Queries**

1. **Display all data:**
2. spark.sql("SELECT \* FROM tempdata").show()
3. **Select specific columns:**
4. spark.sql("SELECT \_c0, \_c1 FROM tempdata").show(5)
5. **Filter rows:**
6. spark.sql("SELECT \* FROM tempdata WHERE \_c4='AZ'").show(5)

#### **9.Handling Headers**

* Modified header option for column names:
* df = spark.read.format("csv") \
* .option("inferSchema", "false") \
* .option("header", "true") \
* .option("sep", ",") \
* .load(file\_location)
* Renamed DataFrame to customer:
* df.createOrReplaceTempView("customer")

### **10.Join Operations**

#### **Data Creation**

1. **Employee Data:**
   * Attributes: emp\_id, name, superior\_emp\_id, year\_joined, emp\_dept\_id, gender, salary.
   * Loaded into a DataFrame:
   * empDF = spark.createDataFrame(data=emp, schema=empColumns)
   * empDF.show()
2. **Department Data:**
   * Attributes: dept\_name, dept\_id.
   * Loaded into a DataFrame:
   * deptDF = spark.createDataFrame(data=dept, schema=deptColumns)
   * deptDF.show()

#### **11.Types of Joins**

1. **Inner Join:**
2. empDF.join(deptDF, empDF.emp\_dept\_id == deptDF.dept\_id, "inner").show()
3. **Outer Join:**
4. empDF.join(deptDF, empDF.emp\_dept\_id == deptDF.dept\_id, "outer").show()
5. **Full Join:**
6. empDF.join(deptDF, empDF.emp\_dept\_id == deptDF.dept\_id, "full").show()
7. **Left Join:**
8. empDF.join(deptDF, empDF.emp\_dept\_id == deptDF.dept\_id, "left").show()
9. **Right Join:**
10. empDF.join(deptDF, empDF.emp\_dept\_id == deptDF.dept\_id, "right").show()

### **12.Techniques Covered**

1. **Column Renaming**
   * withColumnRenamed for better clarity.
2. **Column Selection**
   * selectExpr for SQL-like aliasing.
   * col() for structured column access.
3. **CSV File Handling**
   * Options for schema inference, headers, and delimiters.
4. **SQL Operations**
   * Temporary view creation and queries.
5. **Join Operations**
   * Practical use of join types: inner, outer, full, left, and right.