Data Engineering 101

Data Cleaning using PySpark

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

dropDuplicates()

Removes duplicate rows from the DataFrame based on specified columns

```
df = df.dropDuplicates(['id', 'name'])
```

Removes duplicate rows from the DataFrame, keeping only unique combinations of 'id' and 'name'

Shwetank Singh
GritSetGrow - GSGLearn.com

dropna()

Removes rows with null values in specified columns

df = df.dropna(subset=['important_column'])

Removes any row where the 'important_column' contains a null value

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

fillna()

Replaces null values with specified values

```
df = df.fillna({'age': 0, 'salary': 50000})
```

Replaces null values in the 'age' column with 0 and in the 'salary' column with 50000

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

replace()

Replaces specified values with new values

df = df.replace({'status': {'old': 'legacy', 'new': 'current'}})

Replaces 'old' with 'legacy' and 'new' with 'current' in the 'status' column

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

cast()

Changes the data type of a column

```
df = df.withColumn('salary',
df['salary'].cast('double'))
```

Converts the 'salary' column to double data type

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

withColumn()

Adds a new column or replaces an existing one

df = df.withColumn('full_name', F.concat(df['first_name'], F.lit(' '), df['last_name']))

Creates a new 'full_name' column by concatenating 'first_name', a space, and 'last_name'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

drop()

Removes specified columns from the DataFrame

```
df = df.drop('unnecessary_column')
```

Removes the column named 'unnecessary_column' from the DataFrame

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

rename()

Renames columns in the DataFrame

df = df.withColumnRenamed('old_name', 'new_name')

Renames the column 'old_name' to 'new_name'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

trim()

Removes leading and trailing whitespace from string columns

```
df = df.withColumn('name', F.trim(df['name']))
```

Removes leading and trailing spaces from the 'name' column

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

regexp_replace()

Replaces substrings matching a regular expression

```
df = df.withColumn('phone', F.regexp_replace(df['phone'], r'[^\d]', ''))
```

Removes all non-digit characters from the 'phone' column

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

filter()

Filters rows based on a condition

```
df = df.filter(df['age'] > 18)
```

Keeps only the rows where the 'age' is greater than 18

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

where()

Another method to filter rows based on a condition

```
df = df.where(df['status'] == 'active')
```

Keeps only the rows where the 'status' is 'active'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

distinct()

Removes duplicate rows from the DataFrame

df = df.select('category').distinct()

Returns a DataFrame with unique 'category' values

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

unionByName()

Combines two DataFrames, aligning them by column names

df_combined = df1.unionByName(df2, allowMissingColumns=True)

Combines df1 and df2, matching columns by name and allowing missing columns

Shwetank Singh
GritSetGrow - GSGLearn.com

join()

Combines two DataFrames based on a key

df_joined = df1.join(df2, on='id', how='left')

Performs a left join of df1 and df2 on the 'id' column

Shwetank Singh
GritSetGrow - GSGLearn.com

groupBy()

Groups the DataFrame by specified columns

```
df_grouped = df.groupBy('department').agg(F.avg('salary') \
.alias('avg_salary'))
```

Groups by 'department' and calculates the average salary for each group

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

agg()

Performs aggregations on grouped data

df_summary = df.agg(F.min('age'), F.max('age'), F.avg('salary'))

Calculates the minimum and maximum age, and average salary across the entire DataFrame

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

pivot()

Pivots a DataFrame from long to wide format

df_pivoted = df.groupBy('date').pivot('category').sum('amount')

Creates a pivot table with 'date' as rows, 'category' as columns, and sum of 'amount' as values

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

explode()

Splits array columns into multiple rows

```
df = df.withColumn('item', F.explode(df['items_list']))
```

Creates a new row for each element in the 'items_list' array

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

coalesce()

Returns the first non-null value in a list of columns

```
df = df.withColumn('best_contact',
F.coalesce(df['email'], df['phone'], df['address']))
```

Creates a 'best_contact' column with the first non-null value from email, phone, or address

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

when()

Conditional operations in DataFrame

```
df = df.withColumn('age_group', F.when(df['age'] < 18, 'minor').otherwise('adult'))
```

Creates an 'age_group' column, assigning 'minor' if age < 18, otherwise 'adult'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

split()

Splits a string column into an array

df = df.withColumn('first_name', F.split(df['full_name'], ' ').getItem(0))

Splits 'full_name' by space and takes the first item as 'first_name'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

substring()

Extracts a substring from a string column

```
df = df.withColumn('year', F.substring(df['date'], 1, 4))
```

Extracts the first 4 characters from the 'date' column as 'year'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

to_date()

Converts a string to a date type

df = df.withColumn('date', F.to_date(df['string_date'], 'yyyy-MM-dd'))

Converts 'string_date' to a date type using the specified format

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

isNull()

Checks if a column value is null

```
df = df.filter(df['important_column'].isNull())
```

Keeps only the rows where 'important_column' is null

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

isNotNull()

Checks if a column value is not null

```
df = df.filter(df['important_column'].isNotNull())
```

Keeps only the rows where 'important_column' is not null

Shwetank Singh
GritSetGrow - GSGLearn.com

isin()

Checks if a column value is in a list of values

```
df = df.filter(df['category'].isin(['A', 'B', 'C']))
```

Keeps only the rows where 'category' is either 'A', 'B', or 'C'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

between()

Filters for values in a specified range

df = df.filter(df['age'].between(18, 65))

Keeps only the rows where 'age' is between 18 and 65 (inclusive)

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

rdd.map()

Applies a function to each row of the DataFrame

```
df = df.rdd.map(lambda x: (x['id'], x['name'].upper())).toDF(['id', 'name'])
```

Converts 'name' to uppercase for each row and creates a new DataFrame

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

withColumnRenamed()

Renames a column

df = df.withColumnRenamed('old_column_name', 'new_column_name')

Renames the column 'old_column_name' to 'new_column_name'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

na.fill()

Fills null values with specified values

```
df = df.na.fill({'age': 0, 'name': 'Unknown'})
```

Replaces null values in 'age' with 0 and in 'name' with 'Unknown'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

na.replace()

Replaces specified values with null

```
df = df.na.replace(['N/A', 'NA'], None)
```

Replaces 'N/A' and 'NA' values with null across all columns

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

approx_count_distinct()

Estimates the number of distinct items in a column

```
df = df.agg(F.approx_count_distinct('user_id').alias('distinct_users'))
```

Estimates the count of distinct 'user_id' values

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

collect_list()

Collects values from a column into a list

```
df = df.groupBy('category').agg(F.collect_list('item').alias('items'))
```

Groups by 'category' and collects all 'item' values into a list

Shwetank Singh
GritSetGrow - GSGLearn.com

array_contains()

Checks if an array column contains a value

```
df = df.filter(F.array_contains(df['tags'], 'important'))
```

Keeps only rows where the 'tags' array contains 'important'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

to_timestamp()

Converts a string to a timestamp

```
df = df.withColumn('timestamp',
F.to_timestamp(df['string_timestamp'], 'yyyy-MM-dd
HH:mm:ss'))
```

Converts 'string_timestamp' to a timestamp using the specified format

Shwetank Singh
GritSetGrow - GSGLearn.com

unix_timestamp()

Converts a date/timestamp to Unix timestamp

```
df = df.withColumn('unix_time', F.unix_timestamp(df['date']))
```

Converts 'date' column to Unix timestamp (seconds since 1970-01-01)

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

from_unixtime()

Converts Unix timestamp to a readable date string

df = df.withColumn('readable_time', F.from_unixtime(df['unix_time']))

Converts 'unix_time' to a readable date string

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

datediff()

Calculates the difference between two dates

df = df.withColumn('days_diff', F.datediff(df['end_date'], df['start_date']))

Calculates the number of days between 'start_date' and 'end_date'

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

months_between()

Calculates the number of months between two dates

df = df.withColumn('months_employed', F.months_between(F.current_date(), df['hire_date']))

Calculates the number of months between 'hire_date' and the current date

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

round()

Rounds a numeric column to specified decimal places

```
df = df.withColumn('rounded_salary', F.round(df['salary'], 2))
```

Rounds the 'salary' column to 2 decimal places

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

upper()

Converts a string column to uppercase

```
df = df.withColumn('uppercase_name', F.upper(df['name']))
```

Converts the 'name' column to uppercase

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

lower()

Converts a string column to lowercase

```
df = df.withColumn('lowercase_email', F.lower(df['email']))
```

Converts the 'email' column to lowercase

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

concat_ws()

Concatenates multiple columns with a separator

df = df.withColumn('full_address', F.concat_ws(', ', df['street'], df['city'], df['country']))

Concatenates 'street', 'city', and 'country' with comma separator

Shwetank Singh
GritSetGrow - GSGLearn.com

create_map()

Creates a map column from key-value pairs

```
df = df.withColumn('name_map',
F.create_map(F.lit('first'), df['first_name'], F.lit('last'),
df['last_name']))
```

Creates a map column with 'first' and 'last' as keys, and corresponding name columns as values

Shwetank Singh
GritSetGrow - GSGLearn.com

size()

Returns the size of an array or map column

```
df = df.withColumn('list_size', F.size(df['item_list']))
```

Adds a column with the size of the 'item_list' array

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

first()

Returns the first value in a group

```
df = df.groupBy('category').agg(F.first('item').alias('first_item'))
```

Groups by 'category' and gets the first 'item' in each group

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

last()

Returns the last value in a group

```
df = df.groupBy('category').agg(F.last('item').alias('last_item'))
```

Groups by 'category' and gets the last 'item' in each group

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

regexp_extract()

Extracts a pattern from a string column

```
df = df.withColumn('zip_code',
F.regexp_extract(df['address'], r'\d{5}', 0))
```

Extracts a 5-digit zip code from the 'address' column

Shwetank Singh
GritSetGrow - GSGLearn.com

Data Engineering 101: Data Cleaning using PySpark

window()

Defines a window for window functions

*window_spec = Window.partitionBy('department').orderBy(F.desc('salary'))
df = df.withColumn('salary_rank', F.rank().over(window_spec))*

Creates a window spec partitioned by 'department' and ordered by descending 'salary', then ranks salaries within each department

Shwetank Singh
GritSetGrow - GSGLearn.com

Thank you