**UDAF:**

**UDAF stands for User-Defined Aggregate Function. In the context of big data processing and Spark, it allows users to define their own custom aggregate functions for operations that aren't covered by the built-in aggregate functions provided by Spark.**

**Key Points About UDAFs:**

1. **Custom Aggregations: UDAFs let you define how to combine data across multiple rows into a single value, enabling more complex aggregations than the built-in functions.**
2. **Scala/Python Support: You can implement UDAFs in both Scala and Python, depending on your language preference.**
3. **Spark SQL: Once defined, these functions can be used within Spark SQL queries, providing a seamless way to perform custom aggregations during data analysis.**

**Why Use UDAFs?**

* **Flexibility: They provide flexibility to perform complex aggregations tailored to specific requirements.**
* **Reusability: Once defined, they can be reused across different Spark jobs and queries, ensuring consistency and saving development time.**
* **Performance: Custom aggregate functions can optimize performance for specific use cases by reducing the number of operations needed during the aggregation.**

**UDAF (User-Defined Aggregate Function)**

1. **Purpose: UDAFs are used to define custom aggregation operations that combine values from multiple rows into a single output.**
2. **Input/Output: They take a set of values from multiple rows and return a single aggregated result.**
3. **Example Use Case: Calculating a custom aggregate like product, finding a custom statistic, etc.**