# Apache Spark Application Performance Tuning (SPAT)

**ID** SPAT **Preis** US$ 2'355.– (exkl. MwSt.) **Dauer** 3 Tage

## Zielgruppe

This course is designed for software developers, engineers, and data scientists who have experience developing Spark applications and want to learn how to improve the performance of their code. This is not an introduction to Spark.

## Voraussetzungen

Spark examples and hands-on exercises are presented in Python and the ability to program in this language is required. Basic familiarity with the Linux command line is assumed. Basic knowledge of SQL is helpful.

## Kursziele

Students who successfully complete this course will be able to:

- Understand Apache Spark's architecture, job execution, and how techniques such as lazy execution and pipelining can improve runtime performance
- Evaluate the performance characteristics of core data structures such as RDD and DataFrames
- Select the file formats that will provide the best performance for your application
- Identify and resolve performance problems caused by data skew
- Use partitioning, bucketing, and join optimizations to improve SparkSQL performance
- Understand the performance overhead of Python-based RDDs, DataFrames, and user-defined functions
- Take advantage of caching for better application performance
- Understand how the Catalyst and Tungsten optimizers work
- Understand how Workload XM can help troubleshoot and proactively monitor Spark applications performance
- Learn about the new features in Spark 3.0 and specifically how the Adaptive Query Execution engine improves performance

## Kursinhalt

### Spark Architecture

- RDDs
- DataFrames and Datasets
- Lazy Evaluation
- Pipelining

### Data Sources and Formats

- Available Formats Overview
- Impact on Performance
- The Small Files Problem

### Inferring Schemas

- The Cost of Inference
- Mitigating Tactics

### Dealing With Skewed Data

- Recognizing Skew
- Mitigating Tactics

### Catalyst and Tungsten Overview

- Catalyst Overview
- Tungsten Overview
- Mitigating Spark Shuffles
- Denormalization
- Broadcast Joins
- Map-Side Operations
- Sort Merge Joins

### Partitioned and Bucketed Tables

- Partitioned Tables
- Bucketed Tables
- Impact on Performance

### Improving Join Performance

- Skewed Joins
- Bucketed Joins
- Incremental Joins

### Pyspark Overhead and UDFs

- Pyspark Overhead
- Scalar UDFs
- Vector UDFs using Apache Arrow
- Scala UDFs
- Caching Data for Reuse
- Caching Options
- Impact on Performance
- Caching Pitfalls

**Workload XM (WXM) Introduction**

- WXM Overview
- WXM for Spark Developers

**What's New in Spark 3.0?**

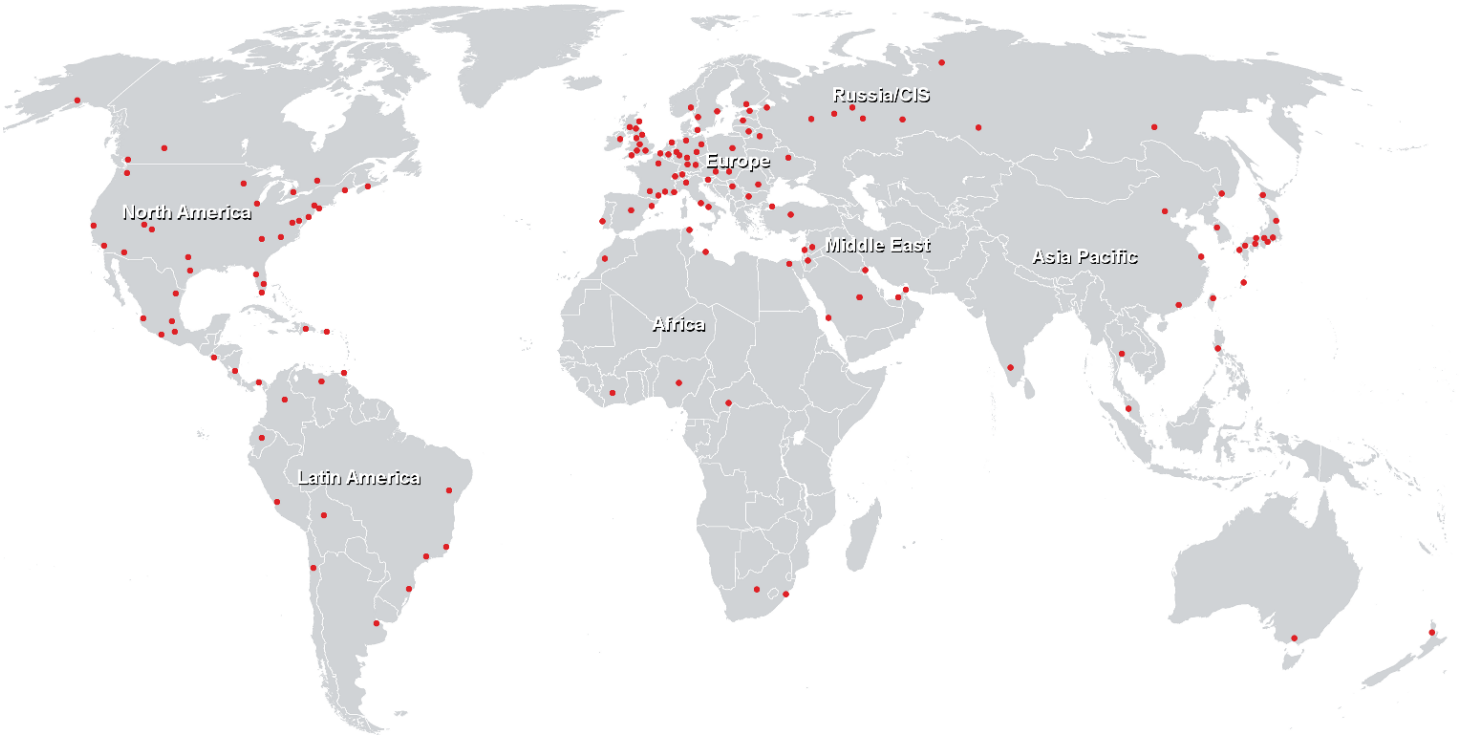
- Adaptive Number of Shuffle Partitions
- Skew Joins
- Convert Sort Merge Joins to Broadcast Joins
- Dynamic Partition Pruning
- Dynamic Coalesce Shuffle Partitions

**Appendix A: Partition Processing**

**Appendix B: Broadcasting**

**Appendix C: Scheduling**

**Weltweite Trainingscenter**




**Fast Lane Institute for Knowledge Transfer GmbH**

Husacherstrasse 3
CH-8304 Wallisellen
Tel. +41 44 832 50 80

**info@flane.ch, https://www.flane.ch**