HEXAWARE

# SAS to Pyspark

HEXAWARE

## Top of mind for customers using SAS

- High licensing & maintenance costs , with **70%+ enterprises cite cost as a key driver** for SAS modernization
- Batch-heavy processing limits real-time analytics with over **70% of organizations** to implement real-time data processing by 2026
- Vendor lock-in restricting flexibility, with organizations citing that **50% reduction in TCO** is achievable by **moving away from SAS**
- Limited scalability for growing data volumes, experiencing up to **30% slower job execution times** compared to cloud-native orchestrators

## SAS market insights in 2026

**Gartner**
Data & Analytics Trends 2026

Enterprises are migrating to **open, cloud-native ecosystems to improve scalability and reduce costs.**

**FORRESTER**
Modern Data Platforms Report

Organizations investing in Python & Spark-based analytics are **enabling faster model development, real-time insights, and AI adoption.**

**ISG**
Provider Lens 2026

Enterprises are prioritizing migration from legacy analytics stacks like **SAS to modern data lakehouse and Spark platforms for performance and agility**

Future proof your business by moving from **SAS to Pyspark**

HEXAWARE

## Challenges with SAS

- High lisencing, maintenance costs
- Limited scalability
- Batch-heavy, slow processing cycles
- Vendor lock-in
- Difficult integration with cloud & AI ecosystems

**Modernize with purpose**

## Our Recommendations

- Modernize analytics to distributed PySpark architecture
- Re-engineer SAS programs into scalable Spark pipelines
- Build cloud-native data processing for speed and scale
- Enable advanced analytics, ML & AI readiness

## Why Moving from SAS to PySpark is a Smart Business Strategy?

### Advantages of Pyspark

- Open-source ecosystem with zero license cost
- Scale up & down with convenience
- Native integration with cloud platforms & data lakes
- Faster insights through parallel processing
- Future-ready foundation for AI/ML use cases

### Business Value Created

- Significant cost optimization
- Cloud-scale analytics agility
- Faster decision-making
- Build a culture for AI innovation
- Future-ready data platform

HEXAWARE

## Our Goals

Modernize legacy SAS analytics to a distributed PySpark platform to achieve **40–60% reduction in analytics TCO**, **deliver 3–5x performance improvement** on large data workloads, and **enable ~50% faster time-to-market** through scalable, cloud-ready data processing.

## Values Created

- Modern, distributed analytics platform built for scale
- Faster insights through parallel PySpark processing
- Lower operational and maintenance overhead
- Improved agility for advanced analytics and AI adoption
- Strong foundation for long-term cost and performance efficiency

## Key Solution Highlights

**Intelligent Code Assessment**
Automated SAS code analysis quickly identifies migration scope and complexity, reducing assessment effort by up to 60%.

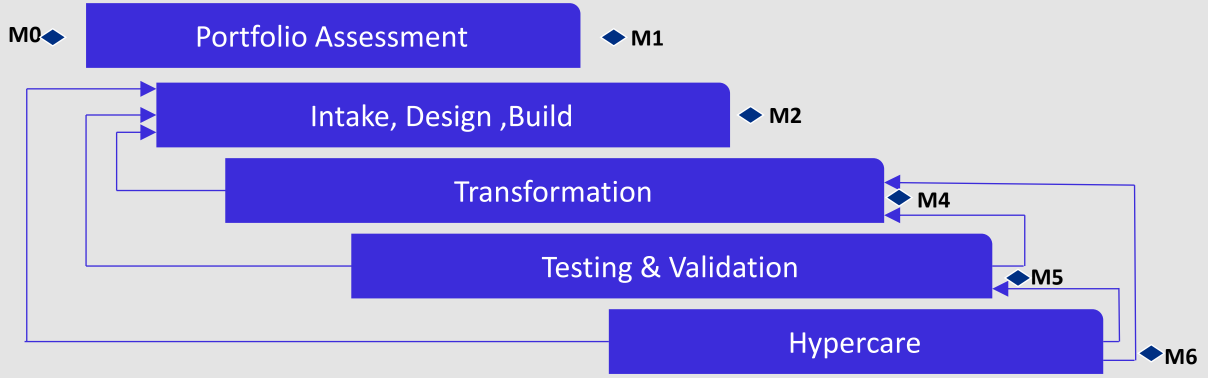
**Accelerated SAS to PySpark Conversion (Amaze -enabled)**
Amaze deliver up to 70% out-of-the-box conversion, transforming 100K+ lines of code in minutes

**Scalable PySpark Analytics Architecture**
SAS workloads are re-engineered into distributed PySpark pipelines, achieving 3–5x performance improvement with elastic scalability.

**Assured Quality & Validation (AMaZE-assisted)**
Automated code and data validation ensures output parity, enabling faster, low-risk go-live.

## The transformational Journey with Hexaware

Powered by amaze®



HEXAWARE

# SAS to PY Solution Overview

| Features | Use Cases | Tool Stack | Limitations | Features in upcoming version |
|---|---|---|---|---|
| **Automated SAS Assessment**<br>Identifies complexity, fitgap analysis & code comprehension for the SAS scripts<br><br>**AI-Driven Code Transformation**<br>Converts SAS procedures, macros, and SQL to PySpark using Gen AI and LLMs.<br><br>**Reusable Python Libraries**<br>Common functions and macro equivalents built into modular, reusable components.<br><br>**Automated Validation & Reconciliation**<br>Automated code, data, and SQL validation with reconciliation script generation.<br><br>**Seamless Integration & DevOps Automation**<br>CI/CD enabled deployment into existing workflows | - **Legacy Modernization**: Migrate SAS workloads to scalable, cloud-native PySpark<br><br>- **Cost Optimization**: Reduce licensing and infrastructure costs<br><br>- **Data Platform Unification**: Standardize analytics across open-source and cloud ecosystems<br><br>- **Accelerated Time-to-Insight**: Enable faster data processing and reporting<br><br>- **AI-Driven Transformation**: Leverage Gen AI for intelligent code conversion and reuse | - **AI/ML**: OpenAI LLMs. GPT 4.0 Turbo, Fine tuned prompts<br><br>- **Data Engineering**: PySpark, Spark SQL, Python (pandas, reusable libraries)<br><br>- **Validation & DevOps**: CI/CD (Jenkins, GitHub Actions)<br><br>- **Monitoring & Visualization**: logging frameworks, Assessment and TCO calculator tool (report based) | - **Complex Macros:** Limited automation for highly customized SAS macros<br><br>- **Statistical Fidelity**: Domain-specific tuning required for advanced procedures<br><br>- **Dependency Mapping**: Manual intervention may be needed for tightly coupled systems<br><br>- **Performance Tuning**: Post-conversion optimization required for large-scale workloads | Data Lineage<br><br>**Will be based on client requirement |

HEXAWARE

How does AMAZE Help?

### Assessment

- Analysis of the SAS code with:
- Complexity Analysis
- Fitgap analysis
- Code Comprehension

### Pre-processing

- **Code Profilers –** analyzes the performance, identifying bottlenecks
- Parsing
- Chunking

### Automated Conversion

- **LLMs**
  - GPT 4.0 Turbo
  - Code Llama
  - Mistral
  - Gemini (Bard AI)

### Validation

- Generated code validated
- Data to data validation
- Go Live

## Benefits

**100,000**
lines of code in 10 minutes

Upto **70%**
Conversion Out-of-the-box

**50%**
Faster time-to-market

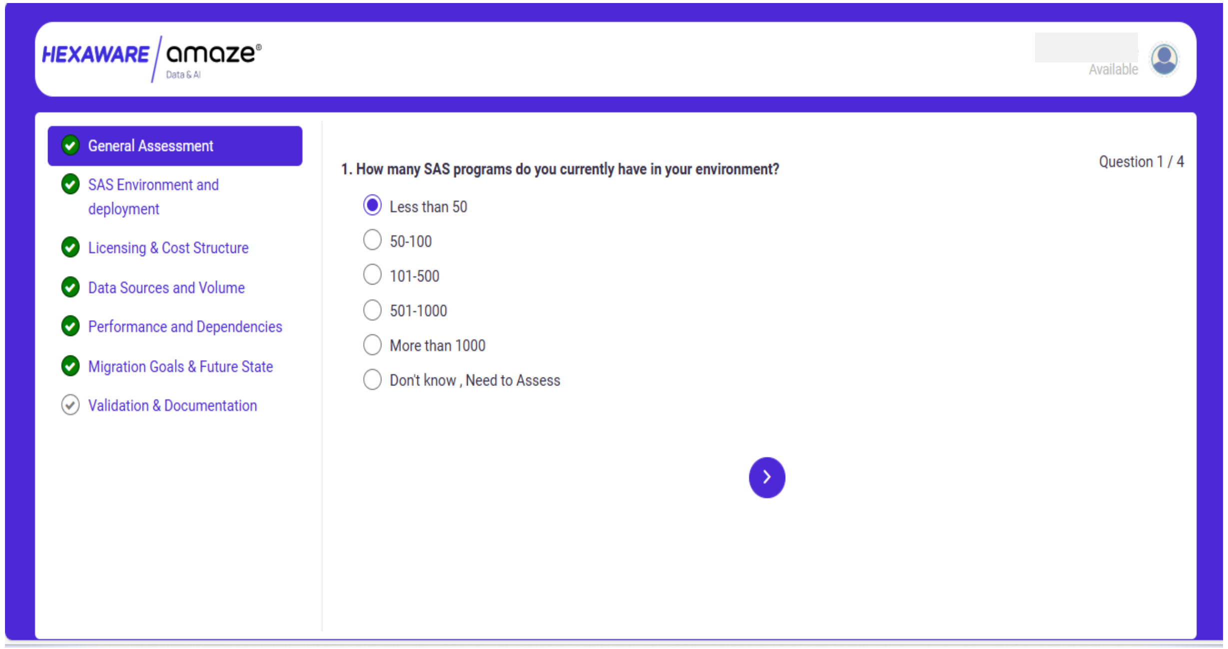
**3x to 5x**
Faster conversion rates

| Enhanced Scalability | Reduced Development Time | Minimum Maintenance Cost | Reduced technical debt | Improved code quality |
|---|---|---|---|---|

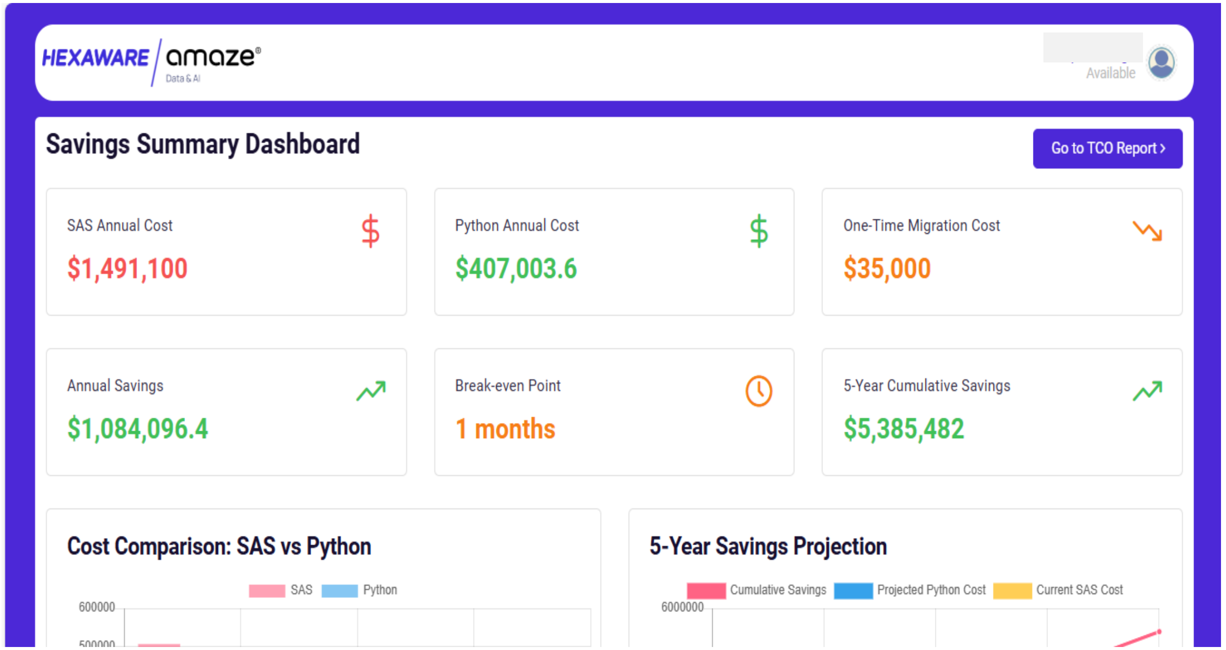
SAS to Py TCO Calculator Snapshots

[SAS to Py TCO Calculator]

SAS to Py TCO Calculator Assessment Questionnaire




SAS to Py TCO calculator Assessment generated Report



# Why choose Hexaware?

## Upto 70% automation with **Amaze®**

Automated LLM based Conversions

## **End-to-End** SAS & Pyspark Mastery

Accelerated development cycles, and reduced operational complexity

## **Proven case studies** in both regulated and non-regulated industries

Our expertise ensures compliance, security, and innovation tailored to your industry needs.

## Approx. **1000** SAS experts

## **11000+** certified cloud and analytics experts

Hybrid Skillsets Across Job scheduling Automation & Microservices

## Dedicated **Centers of Excellence** for various hyperscalers

Accelerated cloud adoption and maximum value from hyperscaler investments.

## **Automated Data validation**

accelerate decision-making and maintain high-quality data standards essential for reliable analytics and compliance

HEXAWARE

# SAS to Python with AI-Driven Automation

A leading financial institution faced rising costs and performance limitations with its legacy SAS platform, hindering its ability to scale and innovate.

**Solution**

Hexaware implemented a comprehensive automation solution, leveraging Amaze®, our Gen AI-powered code conversion engine.

- **Automated Code Conversion**: Amaze® accurately and efficiently converted SAS code to PySpark, minimizing manual effort and reducing the risk of errors.
- **Quality Assurance**: Built-in quality checks ensured the accuracy and reliability of the converted code, maintaining data integrity.
- **Performance Optimization**: Amaze® optimized the converted PySpark code for performance, leveraging the capabilities of the Spark engine.

**Benefits**

- **Reduced Costs** by eliminating SAS licensing fees and leveraging cost-effective cloud infrastructure.
- **Improved Performance:** Accelerated data processing by 30% through PySpark's distributed architecture.
- **Enhanced Scalability:** Successfully handled a 20% increase in data volume without performance degradation.
- **Accelerated Time to Market:** Amaze's automated conversion process and integration with Azure allowed client to rapidly deploy new data-driven applications, accelerating time to market.

Financial Services

HEXAWARE

# Streamlining Data Processing: An Insurer's Transition from SAS to PySpark

Our client was using SAS for data processing and analytics, but it was costly and difficult to scale while also limiting integration with modern platforms and reducing efficiency.

**Solution**

We leveraged our Gen AI powered conversion engine to automate the conversion of their existing SAS scripts to PySpark.

- **Automated Code Conversion:** Amaze® accurately and efficiently converted SAS code to PySpark, minimizing manual effort and reducing the risk of errors.
- **Quality Assurance**: Built-in quality checks ensured the accuracy and reliability of the converted code, maintaining data integrity.
- **Performance Optimization**: Amaze® optimized the converted PySpark code for performance, leveraging the capabilities of the Spark engine.

**Benefits**

- 30% reduction in processing costs
- Improved data processing speed and efficiency
- Enhanced scalability and flexibility

Insurance

You are Modernizing with Hexaware

## Supercharge with Amaze

Activate Amaze automation within your environment to fast-track your transformation

**Next Steps**

### Client

1. Shout out for discovery
2. Continuous collaboration

### Hexaware

1. Assessment of SAS scripts
2. Pre-processing
3. Automated conversion of 50 scripts
4. Validation

**Contact Hexaware for a personalized consultation and roadmap to your cloud-native future.**

**Email :**
**Website**
**Ph:**

# Technical Workflow

# SAS to Python: The Right Move for Our Clients Modernisation Journey

HEXAWARE

## Why shift from SAS to Python?

### Challenges with SAS

- High License Cost
- Limited Scalability
- Lack of integrated insights to cloud ecosystem

### Advantages with Python

- Pay-as-you-use – Python enables clients to avoid expensive SAS licenses
- Scale up & down with convenience - adjusting resources and payment options
- Simplify access to advanced AI –Seamless integration of powerful ML/AI tools with cloud
- Ease of hiring skilled resource - especially in APAC and EMEA

### How Python ensures future-proof enterprises

Seamless Integration with cloud-based services and applications

Multi-cloud Flexibility allowing easy integration across different cloud platforms

Building a Culture for AI Innovation enabling robust AI analytics strategy, providing better access to advanced AI

Quicker Access to Insights with unified access and analysing data more quickly

## Assessment Highlights

Enable predictable, risk-free SAS modernization through an AI-driven, deep-code assessment that eliminates ambiguity before conversion.

### 1 Intelligent Source Ingestion

- Automated intake of SAS programs
- Macro expansion & dependency resolution
- Normalization of legacy constructs
- Creation of executable logic graph

**A unified, machine-readable SAS workload blueprint**

### 2 Complexity Analysis

**Evaluates**

- Structural depth of data steps & procedures
- Macro-driven dynamism & conditional logic
- Nested transformations & execution paths
- External dependencies

**Output**

- Objective Low / Medium / High complexity scoring
- Conversion risk indicators
- Redesign-prone components flagged early

**Accurate effort estimation & migration predictability**

### 3 Fit-Gap Analysis

- Maps SAS constructs to PySpark-native equivalents
- Identifies:
- Directly convertible logic
- Partial mappings requiring refactoring
- Functional gaps needing custom engineering

- Conversion coverage %
- Gap classification (Replace / Refactor / Rebuild)
- Recommended PySpark libraries & patterns

**Zero-surprise conversion strategy**

### 4 Code Comprehension Engine

- Deconstructs SAS logic into human-readable insights
- Traces data lineage (source → transformation → target)
- Summarizes business rules independent of syntax

- Program intent documentation
- SME-independent understanding
- Developer-ready logic clarity

**A unified, machine-readable SAS workload blueprint**

## Benefits

**70-80%**
Reduction in manual assessment effort

**100%**
Workload transparency before conversion

**40-50%**
Reduction in downstream rework

# Validation Highlights

Deliver enterprise-grade trust in converted workloads through automated, hash-based data validation—ensuring PySpark outputs are functionally and numerically equivalent to SAS.

### 1 Dataset Standardization

- Ingest SAS & PySpark outputs
- Normalize data types, formats, null handling
- Align encodings & precision

### 2 Intelligent Key Resolution

- Uses existing primary keys where available
- Auto-generates surrogate keys where missing
- Ensures row-level alignment across datasets

### 3 Hash-Based Row Validation (Core Engine)

**Validation Logic**

- Generate deterministic hash keys per record
- Hash includes all relevant column values
- Compare SAS vs PySpark hashes across datasets

**Why Hashing?**

- Scales to millions of rows
- Eliminates column-order sensitivity
- Delivers high-performance validation

### 4 Multi-Dimensional Validation, Anomaly detection

- Record count reconciliation
- Row-level match & mismatch detection
- Column-wise deviation analysis
- Missing / extra record identification
- Pinpoints data mismatches & transformation issues
- Flags precision loss, truncation, or logic deviations
- Computes % completeness & accuracy score

## Deliverables

- Match / mismatch summary reports
- Row/ Column-wise deviation analysis
- Exception datasets for remediation
- Validation confidence scorecard

## Benefits

**99.9%**
Data Accuracy Assurance

**100%**
Row Level Validation Coverage

**60-70%**
Faster Validation Cycles

HEXAWARE

# Amaze Vs Bluewhale

Comparative view of the tools across different tracks

| # | Tracks | Sub Tracks | **Amaze** [LLM Driven] | **Bluewhale** [Rule Based] |
|---|---|---|---|---|
| **1** | **Assessment** | Complexity Analyzer | Yes | Yes |
| | | FitGap Analyzer | Yes | No |
| **2** | **Conversion** | Macro & Control Flow Handling | Yes | No |
| | | Dataset Management & Set Operation Conversion | Yes | Yes |
| | | Logging & Output Message Translation | Yes | Yes |
| | | Procedure (PROC) Conversion | Yes | Yes |
| | | Code Optimization & Modularization | Yes | No |
| **3** | **Validation** | Dataset to Dataset | Yes | Yes |
| | | Dataset to Report | No | No |
| | | Report to Dataset | No | No |
| | | Report to Report | No | No |
| **4** | **Lineage** | Lineage Tracer | No | Yes |
| | | File Level | No | Yes |
| | | Project Level | No | Yes |
| **5** | **Automation Intelligence** | Automation Coverage % | **90%** | **50%** |
| **6** | **Business Case and TCO Calculator** | | Yes | No |

HEXAWARE

# SAS to Py Case Studies

## A Leading Financial Institution

**Challenge:** The client's SAS-based analytics was costly, hard to scale, and lacked modern platform integration.

**Solution:**

1. Gen-AI based automated SAS to Py conversion using Amaze®
2. Quality assurance with built in quality checks for accuracy and reliability of the converted code.

**Benefits:**

- Cost Reduction
- Enhanced Scalability
- Accelerated data processing

**Accelerated time to market**

## Australian Insurance solution provider

**Challenge:** The client lacked streamlined data processing, the SAS based analytics was costly, hard to scale, and lacked modern platform integration, reducing efficiency.

**Solution:**

1. Amaze® leveraged Gen AI powered conversion to automate SAS to PySpark conversion
2. Quality checks done to maintain data integrity, accuracy and reliability.

**Benefits:**

- Improved Performance
- Better Integration
- Lower Operational cost

**30% reduction in cost**

## Oldest Australian Bank

**Challenge:** SAS based analytics was a challenge for client with its high licensing cost, poor scalability, non-compatibility with modern banking platforms and reduced efficiency.

**Solution**:

1. Amaze® leveraged Gen AI powered conversion to automate SAS to PySpark conversion
    - Modular PySpark Code – for efficient, scalable conversion
    - Production-Ready PySpark code aligned with dependencies.

**Benefits:**

- Faster Processing
- Enhanced flexibility
- Reduced licensing cost

**30% accelerated data processing**

HEXAWARE

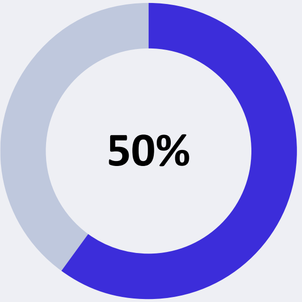
## We help mitigate your pain points during migrations while driving business value

### Migration Pain Points

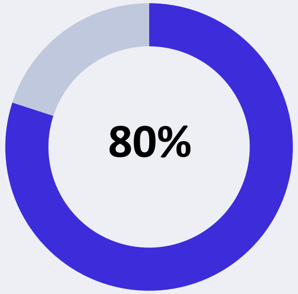
- Lack of specialized expertise in both SAS and Python
- Functional differences between original and converted code
- Inaccurate code conversion
- Difficulty in validating conversion accuracy
- Project delays and cost overruns due to conversion complexity

### Solutions we offer

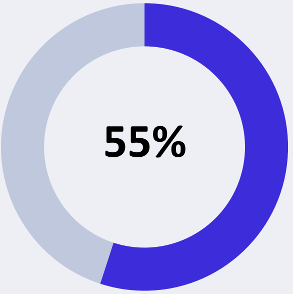
- Automated LLM based Conversions
- 90% accurate conversions
- Resources with Deep Technical expertise
- Business logic retention during conversion
- Built in data quality checks
- Automated Data validation




**Faster** Project Timelines and time-to-value




**Efforts reduced** with converting complex logics and formulas with LLM based conversions




**Improved ROI** with reduced SAS cost licenses and automated migration

# SAS transformation Journey

HEXAWARE

| | Discovery & Assessment | Intake | Transformation | Validation & Rollout |
|---|---|---|---|---|
| **Activities** | • Intelligent ingestion of SAS programs, macros, and dependencies<br>• Automated complexity and fit-gap analysis across data steps & procedures<br>• Identify direct, partial, and custom conversion candidates<br>• Define target PySpark architecture and migration strategy | • Map SAS constructs to PySpark-native equivalents<br>• Define data pipeline, orchestration, and security architecture<br>• Finalize scope based on conversion feasibility and risks<br>• Establish validation strategy upfront | • Amaze-enabled automated SAS to PySpark conversion<br>• Re-engineering of complex logic, macros, and edge cases<br>• Performance tuning and Spark optimization<br>• Pipeline integration and environment setup | • Dataset standardization and key resolution<br>• Hash-based row-level data validation at scale<br>• SIT, UAT, and anomaly detection<br>• Deployment support and production readiness |
| **Deliverables** | • SAS workload blueprint (machine-readable)<br>• Complexity & fit-gap assessment report<br>• Effort estimation and migration roadmap<br>• Target-state PySpark architecture | • SAS-to-PySpark mapping document<br>• Finalized target-state design<br>• Delivery plan & refined SOW | • Converted and optimized PySpark code<br>• Integrated pipelines with functional test cases<br>• Conversion coverage and optimization report | • Match / mismatch validation reports<br>• Row & column-level deviation analysis<br>• Exception datasets for remediation<br>• Knowledge transition & runbook |