# 11 Databricks Workflow & Automation

## Databricks Workflow & Automation: Jobs, Scheduling, Monitoring, Alerts & Workflow Automation

### Basic Level Questions (1-10)

**1. What are Databricks Jobs and how do they differ from interactive notebooks?** *Focus on: Production workloads, automation, resource management, reliability*

**2. What are the different types of tasks you can create in a Databricks Job?** *Focus on: Notebook tasks, JAR tasks, Python wheel tasks, SQL tasks, pipeline tasks*

**3. How do you schedule a Databricks Job to run at specific intervals?** *Focus on: Cron expressions, time zones, recurring schedules, trigger types*

**4. What is the Databricks Workflows feature and what problems does it solve?** *Focus on: Multi-task orchestration, dependency management, parallel execution, error handling*

**5. How do you monitor the execution status of Databricks Jobs?** *Focus on: Job runs page, execution history, logs, status indicators*

**6. What are the different cluster types available for running Databricks Jobs?** *Focus on: Job clusters, all-purpose clusters, pools, auto-scaling*

**7. How do you pass parameters to a Databricks Job at runtime?** *Focus on: Job parameters, notebook widgets, environment variables, command-line arguments*

**8. What is the difference between a job cluster and an all-purpose cluster?** *Focus on: Cost optimization, isolation, performance, lifecycle management*

**9. How do you set up basic email notifications for job failures in Databricks?** *Focus on: Notification settings, email alerts, webhook integration, escalation*

**10. What are some common reasons why Databricks Jobs might fail?** *Focus on: Resource issues, code errors, data problems, cluster startup failures*

### Intermediate Level Questions (11-20)

**11. How would you design a multi-task workflow with dependencies between different data processing stages?** *Focus on: Task dependencies, conditional execution, error propagation, parallel vs sequential execution*

**12. Explain how to implement retry logic and error handling in Databricks Workflows.** *Focus on: Retry policies, exponential backoff, circuit breakers, dead letter queues*

**13. How do you implement dynamic parameter passing between tasks in a workflow?** *Focus on: Task values, parameter inheritance, dynamic configuration, state management*

**14. What are the best practices for managing secrets and credentials in Databricks Jobs?** *Focus on: Secret scopes, Azure Key Vault integration, environment variables, security*

**15. How would you implement a data quality checkpoint system within a Databricks Workflow?** *Focus on: Validation tasks, conditional branching, data quality metrics, failure handling*

**16. Explain how to set up comprehensive monitoring for production Databricks Workflows.** *Focus on: Metrics collection, custom logging, observability, dashboard creation*

**17. How do you handle resource optimization for workflows with varying compute requirements?** *Focus on: Dynamic cluster sizing, job pools, spot instances, cost optimization*

**18. What strategies would you use to implement blue-green deployments for Databricks Jobs?** *Focus on: Version management, deployment strategies, rollback procedures, testing*

**19. How do you implement workflow orchestration for real-time and batch processing combined?** *Focus on: Hybrid architectures, streaming integration, scheduling coordination, data consistency*

**20. Explain how to set up custom alerting based on job performance metrics and SLA requirements.** *Focus on: Custom metrics, threshold-based alerts, escalation procedures, integration with external systems*

## Advanced/Difficult Level Questions (21-30)

**21. Design a comprehensive CI/CD pipeline for Databricks Workflows that includes testing, deployment, and rollback capabilities.** *Focus on: Infrastructure as code, automated testing, deployment automation, version control integration*

**22. How would you implement a sophisticated workflow orchestration system that handles complex data lineage and dependency management across 100+ jobs?** *Focus on: Dependency resolution, DAG optimization, circular dependency detection, scalability*

**23. Design a multi-tenant workflow automation system in Databricks with proper isolation and resource management.** *Focus on: Workspace isolation, resource quotas, access control, cost allocation, governance*

**24. How would you implement intelligent job scheduling that adapts to data availability and system load?** *Focus on: Event-driven scheduling, load balancing, resource optimization, predictive scheduling*

**25. Explain how you would build a workflow recovery system that can automatically handle partial failures and resume from checkpoints.** *Focus on: State management, checkpoint strategies, idempotency, partial execution recovery*

**26. Design a comprehensive monitoring and observability solution for a complex Databricks environment with hundreds of workflows.** *Focus on: Metrics aggregation, distributed tracing, log correlation, anomaly detection, predictive monitoring*

**27. How would you implement a workflow optimization system that automatically tunes cluster configurations based on historical performance data?** *Focus on: Machine learning for optimization, performance profiling, automated tuning, cost-performance balance*

**28. Design a disaster recovery strategy for critical Databricks Workflows with RPO/RTO requirements.** *Focus on: Cross-region replication, backup strategies, failover procedures, data consistency*

**29. How would you implement a governance framework for workflow automation that ensures compliance and auditability?** *Focus on: Policy enforcement, audit trails, compliance reporting, access controls, data governance*

**30. Design a self-healing workflow system that can automatically detect, diagnose, and remediate common failure patterns.** *Focus on: Failure pattern recognition, automated remediation, machine learning for anomaly detection, self-optimization*

---

## Practical Workflow Scenarios

### Real-World Implementation Challenges

**Scenario A:** You need to orchestrate a complex ETL pipeline that processes data from 10 different sources with varying SLAs and dependencies. How would you design this in Databricks Workflows?

**Scenario B:** A critical workflow is experiencing intermittent failures with no clear pattern. Walk through your troubleshooting and resolution approach.

**Scenario C:** You need to implement a workflow that processes both streaming and batch data with strict consistency requirements. How would you architect this?

**Scenario D:** Design a workflow system that can handle sudden spikes in data volume while maintaining cost efficiency.

---

## Advanced Configuration & Integration

### Enterprise Integration Patterns

**External System Integration:**

- REST API integration patterns
- Message queue integration (Kafka, Service Bus)
- Database connectivity and transaction management
- File system and cloud storage integration

**Advanced Scheduling:**

- Event-driven triggering mechanisms
- Cross-workflow dependencies
- Time-based and data-based triggers
- Complex conditional execution logic

**Security & Governance:**

- Role-based access control implementation
- Audit logging and compliance reporting
- Secret management and rotation
- Network security considerations

## Performance & Optimization

### Workflow Performance Tuning

**Resource Optimization:**

- Cluster sizing strategies for different workload types
- Spot instance utilization in production workflows
- Memory and CPU optimization techniques
- Storage optimization for intermediate data

**Execution Optimization:**

- Parallel task execution strategies
- Bottleneck identification and resolution
- Cache utilization in multi-task workflows
- Data locality optimization

**Cost Management:**

- Cost monitoring and attribution
- Resource usage optimization
- Automatic scaling policies
- Budget controls and alerts