1. What is Azure Databricks, and how does it differ from Apache Spark?

Azure Databricks is unified, and collaborative Apache Spark-based big data analytics service designed for data science and data engineering.

Apache Spark is an open-source in-memory distributed computing engine to process big data.

Azure Databricks provides a managed and optimized Spark environment in the cloud, it's built on Apache Spark and Jupyter Notebooks.

2. Explain the architecture of Azure Databricks.

Azure Databricks operates out of a control plane and a data plane.

The control plane includes the backend services that Azure Databricks manages in its own Azure account. Notebook commands and many other workspace configurations are stored in the control plane and encrypted at rest.

The data plane, and is where your data resides. This is also where data is processed. Use Azure Databricks connectors to connect clusters to external data sources outside of your Azure account to ingest data, or for storage.

3. How do you create an Azure Databricks workspace, and what are its components?

Search Azure Databricks > Create > Enter Details > Review + Create > Launch Workspace

- Notebooks
- Data
- Clusters
- Jobs
- Models

- Workspace -> To launch and work in Notebooks
- Compute -> To create and manage Spark clusters to run notebooks
- Data -> Store, ingest data
- Workflows -> Schedule and Orchestrate execution of Notebooks

4. What are the different cluster modes in Azure Databricks, and when would you use each one?

Cluster Modes

Standard	Single Node clusters	High Concurrency clusters
At least 1 worker node	Only Driver Node	
No Isolation Shared clusters		provide fine-grained sharing

Standard Cluster

Purpose: designed for data exploration, analysis, and development tasks.

Usage: Data scientists, analysts, and engineers typically use interactive clusters for ad-hoc querying, data exploration, and experimenting with code.

Behavior: Interactive clusters are automatically terminated after a period of inactivity to save costs. They're optimized for interactive tasks and prioritize responsiveness.

Job Cluster (Batch Cluster):

Purpose: Job clusters are meant for running batch workloads, such as scheduled jobs and ETL pipelines.

Usage: These clusters are used when you want to execute a specific job or process on a regular basis, often scheduled through Databricks Jobs or external orchestration tools.

Behavior: Unlike interactive clusters, job clusters are typically long-running and are not automatically terminated after a period of inactivity. They're optimized for executing batch processes efficiently.

High Concurrency Cluster: Shared

Purpose: High concurrency clusters are designed to support multiple users running queries and tasks concurrently without resource contention.

Usage: These clusters are suitable for scenarios where multiple users or teams need to share the same cluster resources for running their workloads simultaneously.

Behavior: High concurrency clusters allocate resources dynamically to users and tasks based on demand. They are optimized for efficient resource allocation and fair sharing among users.

Ref: Configure clusters - Azure Databricks | Microsoft Learn

Azure Databricks supports three cluster modes: Standard, High Concurrency, and Single Node. The default cluster mode is Standard

5. How can you optimize the performance of an Azure Databricks cluster?

- Right-sizing the cluster by selecting the **appropriate instance types** and cluster configurations based on workload requirements.
- Enabling **autoscaling** to dynamically adjust the cluster size based on the workload.
- Leveraging **cluster termination** and auto-restart policies to optimize costs.
- Utilizing **caching**, broadcast variables, and data shuffling optimizations.
- Tuning **Spark configurations** to optimize memory usage, parallelism, and resource allocation.

6. What is the purpose of the Databricks Job and how do you create one?

To schedule and orchestrate data processing workflows.

Databricks Jobs is the fully managed orchestrator for all your data, analytics, and AI.

Go to Databricks Workspace -> Workflows -> Create a Job

Ref: How to Save Time and Money on Data and ML Workflows With “Repair and Rerun” - The Databricks Blog

7. How do you schedule and monitor Databricks notebooks?

Scheduling:

Notebooks should be associated in a Job or Pipeline to be scheduled.

Open Notebook > click file > click Schedule > Create a new job if not exists > Add Schedule

Go to Workflows > Create a Job with this Notebook > Schedule the job.

Monitoring:

Go to workflows > click on the job > Monitoring details for the job.

8. Explain the process of connecting Azure Databricks with Azure Data Lake Storage Gen2.

Session Scoped Authentication - Connecting to DL with credentials used in Notebook session.
spark.config.set(<token>, <access key>)

Custer Scoped Authentication

AAD pass-through

Using Access Keys

SAS token

Service Principal

Cluster Scoped Authentication

Azure Active Directory (AAD) pass-through authentication

9. How can you use Azure Databricks to process streaming data?

Databricks recommends using Auto Loader to ingest supported file types from cloud object storage into Delta Lake. For ETL pipelines, Databricks recommends using Delta Live Tables (which uses Delta tables and Structured Streaming). You can also configure incremental ETL workloads by streaming to and from Delta Lake tables.

Ingest data from a streaming service such as EventHub, IoT hub, Apache Kafka and store in streaming dataframe. Perform data transformations on that dataframe and write to sink.

Ref: Streaming on Azure Databricks - Azure Databricks | Microsoft Learn

10. What are Delta tables in Azure Databricks, and why are they useful?

Delta lake is a open-source Data Lakehouse, which combines the functionality of Data Lake and Data Warehouse. Delta Lake is the data storage layer for Databricks. A table created in Delta Lake is called a Delta table.

A Delta table is essentially a type of table that is stored as Parquet files with added transactional capabilities, making it a powerful and versatile data storage format within the Delta Lake ecosystem.

Capabilities - Versioning, Transactional ca[abilities - ACID transactions, Time Travel, schema evolution, data consistency, easily integration with Streaming data pipelines.

11. How do you secure data in Azure Databricks? What security features does it provide?

Data Encryption:

- Customer managed keys
- Encrypt queries, history and results in Control Plane
- Double encryption for DBFS - enable encryption at the Azure Storage infrastructure level.
- Manage workspace settings - restrict notebook downloads etc.

Ref: Data security and encryption - Azure Databricks | Microsoft Learn

Azure Databricks Security Best Practices

12. How can you automate the deployment of Azure Databricks resources using infrastructure-as-code tools?

You can use an infrastructure-as-code (IaC) approach to programmatically provision Azure Databricks infrastructure and resources such as workspaces, clusters, cluster policies, pools, jobs, groups, permissions, secrets, tokens, and users by using the Databricks Terraform provider.

Efficient Databricks Deployment Automation with Terraform | Databricks Blog

13. What are the different data ingestion methods supported by Azure Databricks?

1. Delta Lakes
2. Batch Ingestion - ADLS, Blob
3. Structured Streaming
4. External Databases and Warehouses
5. Other streaming - Event hub, IoT hub, Delta Live Tables
6. API and REST endpoints

14. How do you handle data partitioning and bucketing in Azure Databricks?

Partitioning - Dividing the data into smaller manageable subsets based on a column This will greatly improve query performance as the engine can skip over irrelevant rows based on column values. Ex: Partitioning by date, region.

Create Partitions in Delta tables, then query the partitioned data.

df.write.partitionBy("date").format("delta").saveAsTable("sales")

SELECT * FROM sales WHERE date = '2023-08-01'

Bucketing - Organizing/clustering data in partitions into small, equal size buckets based on one or more columns. This is done to improve join and aggregation performance.

Create bucketed delta tables, then query the bucketed data

df.write.bucketBy(8, "user_id").sortBy("timestamp").format("delta").saveAsTable("user_data")

SELECT COUNT(*) FROM user_data WHERE user_id = 123

15. What are the best practices for managing and monitoring Azure Databricks workloads?

Managing Workloads:

- Resource Allocation: Choose appropriate cluster sizes and configurations based on your workload requirements.Utilize autoscaling to automatically adjust cluster size based on workload demands.
- Cluster Management: Avoid creating too many clusters; reuse clusters when possible to reduce overhead. automatically terminate idle clusters.
- Library Management: only install the libraries that are required for your workloads. Version libraries to ensure consistency and avoid compatibility issues.
- Workspace Organization: Use folders and notebooks to logically organize your code, data, and documentation. Define naming conventions for notebooks, clusters, and other resources to ensure consistency.
- Data Lake Management: Use structured storage formats like Parquet and Delta Lake to optimize query performance. Organize data into meaningful directories and partitions to improve data retrieval efficiency.
- Secrets and Credentials: Utilize Databricks secrets for storing sensitive information securely. Avoid hardcoding credentials; retrieve them from secrets during runtime.

Monitoring Workloads:

- Cluster Monitoring: Monitor cluster metrics like CPU usage, memory usage, and disk utilization to ensure efficient resource utilization. Set up alerts for abnormal behavior, such as sudden spikes in resource usage.
- Job Monitoring: Monitor job executions to track their progress and performance. Set up alerts for job failures or long execution times.
- Notebook Execution Monitoring: Monitor notebook execution durations to identify slow or resource-intensive processes. Monitor the progress of notebooks and visualize intermediate results to identify bottlenecks.

- Data and Storage Monitoring: Monitor the growth of your data lake and storage usage to prevent capacity issues. Keep an eye on the data partitioning and bucketing strategies for optimal performance.
- Audit Logs and Security Monitoring: Monitor audit logs to track user activities and ensure compliance. Set up alerts for unauthorized access or suspicious activities.
- Automated Alerts and Notifications: Set up automated alerts for critical events using Azure Monitor or other monitoring tools. Configure notifications to be sent to relevant stakeholders for immediate action.
- Optimize Cost Monitoring: Monitor resource consumption and spending patterns to identify opportunities for cost optimization. Use Azure Cost Management and Azure Advisor for cost-related insights.

Ref: Best practices for operational excellence - Azure Databricks | Microsoft Learn