AWS Data Engineering Services - Quick Reference Cheat Sheet

Data Ingestion Services

Amazon Kinesis Family

Service	Use Case	Key Features	Limits
Kinesis Data Streams	Real-time streaming	• Custom consumer apps • Replay capability • Low latency (< 1 second)	• 1MB record limit • 1000 records/sec per shard
Kinesis Data Firehose	Near real-time delivery	• Serverless • Built-in transformation • Direct S3/Redshift delivery	• 60 second buffer minimum • No replay capability
Kinesis Analytics	Real-time analytics	• SQL on streaming data • Anomaly detection • Time-windowed queries	• SQL-based processing only

AWS Glue

Component	Purpose	Key Points
Glue Crawlers	Schema discovery	• Auto-detect schema changes • Populate Data Catalog • Schedule-based or on-demand
Glue ETL Jobs	Data transformation	• Serverless Spark • Auto-scaling • Built-in retry logic
Glue Data Catalog	Metadata repository	• Hive-compatible metastore • Integration with Athena/EMR • Schema versioning
Glue DataBrew	Visual data preparation	• No-code transformations • Data profiling • 250+ built-in transformations

Data Storage Services

Amazon S3 Storage Classes

Storage Class	Use Case	Retrieval Time	Cost
Standard	Frequently accessed	Immediate	Highest storage cost
Standard-IA	Infrequently accessed	Immediate	Lower storage, retrieval fee
One Zone-IA	Non-critical, infrequent	Immediate	20% less than Standard-IA
Glacier Instant	Archive with instant access	Immediate	Lower storage cost
Glacier Flexible	Archive data	1-12 hours	Very low storage cost
Glacier Deep Archive	Long-term archive	12-48 hours	Lowest storage cost

Amazon Redshift

Feature	Description	Best Practice
Distribution Keys	How data is distributed	• Use JOIN columns • Avoid high cardinality • Consider EVEN for small tables
Sort Keys	Physical data ordering	• Use WHERE clause columns • Consider compound vs interleaved • Limit to 3-4 columns
Compression	Reduce storage/IO	• Use ANALYZE COMPRESSION • Different encoding per column • Automatic for new tables
Workload Management	Query prioritization	• Separate queues by workload • Set memory allocation • Use concurrency scaling

DynamoDB

Concept	Description	Guidelines
Partition Key	Primary hash key	• High cardinality • Uniform access pattern • Avoid hot partitions
Sort Key	Range key for sorting	• Enable range queries • Model 1:N relationships • Support query patterns
GSI/LSI	Secondary indexes	• GSI: Different partition key • LSI: Same partition key • Max 20 GSI per table
Capacity Modes	Billing model	• On-Demand: Unpredictable • Provisioned: Predictable + cheaper

Data Processing Services

Amazon EMR

Component	Purpose	Key Points
Master Node	Cluster management	• Manages cluster • NameNode for HDFS • Single point of failure
Core Nodes	Data storage + processing	• Run DataNode + TaskTracker • HDFS storage • Can be removed with care
Task Nodes	Processing only	• No HDFS storage • Spot instances recommended • Safe to terminate

AWS Lambda

Aspect	Specification	Considerations
Runtime	15 minutes max	• Use Step Functions for longer workflows • Consider EMR for heavy processing
Memory	128MB - 10GB	• CPU scales with memory • Optimize for cost vs performance
Triggers	Event-driven	• S3 events, Kinesis, DynamoDB Streams • EventBridge for schedules
Concurrency	1000 default	• Can request increases • Consider reserved concurrency

Analytics Services

Amazon Athena

Feature	Description	Optimization Tips
Serverless SQL	Query S3 data directly	• Use columnar formats (Parquet) • Partition data by query patterns • Compress data (GZIP, Snappy)
Query Engines	Presto/Trino based	• Use appropriate data types • Avoid SELECT * queries • Use LIMIT for exploration
Workgroups	Query organization	• Set data limits • Control costs • Separate environments

Amazon QuickSight

Component	Purpose	Key Features
SPICE	In-memory engine	• Fast query performance • Automatic data refresh • 10GB per dataset
Data Sources	Input connections	• 30+ native connectors • Direct query vs SPICE • Row-level security
Dashboards	Visualization	• Interactive dashboards • Mobile responsive • Embedded analytics

Security Services

AWS IAM for Data Engineering

Policy Type	Use Case	Example
Identity-based	User/role permissions	Glue job execution role
Resource-based	Cross-account access	S3 bucket policy
Session policies	Temporary restrictions	Federated access limits
Permissions boundaries	Maximum permissions	Developer sandbox limits

AWS KMS

Key Type	Management	Use Case
AWS Managed	AWS controls	• Default encryption • Service-specific keys
Customer Managed	You control	• Custom key policies • Cross-account access • Key rotation control
Customer Provided	You provide	• Full control • Higher complexity • Import your own keys

Monitoring & Governance

CloudWatch for Data Pipelines

Metric Category	Examples	Alerting Strategy
Glue Jobs	• Job duration • Success/failure rate • DPU hours	• Set SLA-based alarms • Monitor cost metrics
Kinesis	• IncomingRecords • WriteProvisionedThroughputExceeded	• Shard-level monitoring • Auto-scaling triggers
Redshift	• CPU utilization • Disk space • Query performance	• Performance alerts • Storage warnings

AWS Lake Formation

Feature	Purpose	Best Practice
Data Permissions	Fine-grained access control	• Column-level permissions • Row-level security • Tag-based policies
Data Discovery	Catalog population	• Automatic crawling • ML-powered classification • PII detection with Macie
Data Sharing	Cross-account access	• Resource sharing • Query federation • Audit trails

Common Architecture Patterns

Lambda Architecture

```
Batch Layer: S3 → Glue/EMR → Redshift (historical data)
Speed Layer: Kinesis → Lambda → DynamoDB (real-time)
Serving Layer: Athena/QuickSight (unified view)
```

Kappa Architecture

```
Stream Processing: Kinesis → Kinesis Analytics → Output
Everything is treated as a stream, including batch data
```

Data Lake Pattern

```
Landing Zone (S3 Raw) → Processing (Glue/EMR) →
Curated Zone (S3 Processed) → Analytics (Athena/Redshift)
```

Performance Optimization Quick Tips

S3 Optimization

- Use prefixes to avoid hot spots
- Multipart upload for files > 100MB
- S3 Transfer Acceleration for global access
- CloudFront for frequently accessed data

Redshift Optimization

- VACUUM regularly to reclaim space
- ANALYZE to update table statistics
- Use COPY command for bulk loads
- Monitor query performance with system tables

Glue Optimization

- Use bookmark for incremental processing
- Optimize for fewer, larger files
- Use pushdown predicates

- Consider Glue streaming for low latency

Remember: The exam tests your ability to choose the right service for the right use case. Focus on understanding trade-offs between services!