AWS Data Analytics Pipeline and Services Report

## 1. AWS Regions, Availability Zones, and Edge Locations

Regions: Physical geographic locations with multiple data centers.  
Availability Zones (AZs): Individual data centers within a region, designed for fault isolation.  
Edge Locations: CDN endpoints used by services like CloudFront to cache content closer to users.  
  
Importance: Low latency and high availability for analytics applications.

## 2. AWS CLI: List All Regions

Command:  
aws ec2 describe-regions --query "Regions[\*].RegionName" --output table  
  
Sample Output:  
| RegionName |  
|-------------|  
| us-east-1 |  
| eu-west-1 |

## 3. Create IAM User with Least Privilege for S3

Policy JSON:  
{  
 "Version": "2012-10-17",  
 "Statement": [  
 {  
 "Effect": "Allow",  
 "Action": ["s3:ListBucket", "s3:GetObject", "s3:PutObject"],  
 "Resource": ["arn:aws:s3:::your-bucket-name", "arn:aws:s3:::your-bucket-name/\*"]  
 }  
 ]  
}

## 4. Amazon S3 Storage Classes Comparison

| Storage Class | Use Case |  
|------------------------|-------------------------------------------|  
| S3 Standard | Frequent access, real-time analytics |  
| S3 Intelligent-Tiering | Unpredictable access, cost optimization |  
| S3 Glacier | Archival storage, long-term backups |

## 5. Create S3 Bucket & Upload with Versioning

Commands:  
aws s3api create-bucket --bucket my-data-bucket --region us-east-1  
aws s3api put-bucket-versioning --bucket my-data-bucket --versioning-configuration Status=Enabled  
aws s3 cp v1.csv s3://my-data-bucket/file.csv  
aws s3 cp v2.csv s3://my-data-bucket/file.csv

## 6. S3 Lifecycle Policy (Glacier after 30 days, Delete after 90)

Policy JSON:  
{  
 "Rules": [  
 {  
 "ID": "MoveToGlacierAndDelete",  
 "Prefix": "",  
 "Status": "Enabled",  
 "Transitions": [  
 {  
 "Days": 30,  
 "StorageClass": "GLACIER"  
 }  
 ],  
 "Expiration": {  
 "Days": 90  
 }  
 }  
 ]  
}

## 7. RDS vs DynamoDB vs Redshift

| Service | Best For | Use Case Example |  
|-----------|----------------------------------|-------------------------------|  
| RDS | Relational OLTP data | Transaction systems |  
| DynamoDB | NoSQL, key-value access | Session or metadata store |  
| Redshift | Columnar OLAP, analytics | BI dashboards, analytics |

## 8. DynamoDB Table & Lambda Triggered by S3 Upload

Lambda Function:  
import boto3  
  
def lambda\_handler(event, context):  
 file = event['Records'][0]['s3']['object']['key']  
 dynamodb = boto3.client('dynamodb')  
 dynamodb.put\_item(TableName='UploadEvents', Item={'FileName': {'S': file}})

## 9. What is Serverless Computing (AWS Lambda)

Pros: Auto-scaling, Pay-per-use, No server management  
Cons: Cold starts, Timeout limits, Stateless execution

## 10. Lambda Logging File Metadata to CloudWatch

def lambda\_handler(event, context):  
 for record in event['Records']:  
 file = record['s3']['object']  
 print(f"Uploaded: {file['key']}, Size: {file['size']}, Time: {record['eventTime']}")

## 11. AWS Glue Job: CSV to Parquet

from awsglue.context import GlueContext  
from pyspark.context import SparkContext  
sc = SparkContext()  
glueContext = GlueContext(sc)  
df = glueContext.create\_dynamic\_frame.from\_catalog(database='mydb', table\_name='mytable')  
glueContext.write\_dynamic\_frame.from\_options(frame=df, connection\_type='s3', format='parquet', connection\_options={'path': 's3://my-output-bucket/'})

## 12. Kinesis Services Comparison

| Service | Functionality | Example Use Case |  
|------------------------|----------------------------|--------------------------------|  
| Kinesis Data Streams | Real-time processing | Clickstream analytics |  
| Kinesis Firehose | Stream to S3/Redshift | Streaming logs to storage |  
| Kinesis Data Analytics | SQL-based stream analysis | Real-time data transformations|

## 13. Columnar Storage in Redshift

Benefits: Faster reads, compression, efficient aggregations for analytics workloads

## 14. COPY Command in Redshift

CREATE TABLE sales\_data (id INT, amount FLOAT, date DATE);  
COPY sales\_data FROM 's3://my-bucket/sales.csv' CREDENTIALS 'aws\_iam\_role=role-arn' CSV;

## 15. Glue Catalog & Athena (Schema on Read)

Glue Data Catalog manages table metadata.  
Athena uses schema-on-read to query data directly without prior transformation.

## 16. Athena Table from Glue & Query

SELECT \* FROM glue\_catalog.mytable WHERE amount > 1000;

## 17. Amazon QuickSight (SPICE & Embedded Dashboards)

SPICE: In-memory engine for fast performance  
Embedded Dashboards: Integrate into applications

## 18. QuickSight Dashboard

Components: Filter (e.g., Region = 'India'), Calculated field (e.g., Profit = Revenue - Cost)

## 19. CloudWatch vs CloudTrail

| Service | Function |  
|-------------|---------------------------------|  
| CloudWatch | Metrics, Logs, Alarms |  
| CloudTrail | Tracks API actions |

## 20. End-to-End Data Analytics Pipeline

S3 → Lambda → Glue → Athena → QuickSight  
  
Justification:  
- S3 for storage  
- Lambda for serverless ingestion  
- Glue for transformation  
- Athena for querying  
- QuickSight for visualization