# 1. Code

## 1.1 Cloud

* **AWS Glue:**
  + **ETL jobs, crawlers, and data catalogs.**
  + **Integration with S3, Redshift, and RDS.**
  + **Glue Studio, Glue DataBrew, and Glue Elastic Views.**
  + **Practice: Build a serverless ETL pipeline using Glue.**
* **Databricks:**
  + **Databricks topics**

## 1.2 Python-Spark

* **PySpark:**
  + **DataFrame API, RDDs, and Catalyst Optimizer.**
  + **Advanced: Broadcast variables, accumulators, and UDFs.**
  + **Practice: Optimize a PySpark job for skewed data.**
* **Delta Lake:**
  + **ACID transactions, schema evolution, and time travel.**
  + **Integration with Databricks and AWS S3.**
  + **Practice: Build a Lakehouse architecture using Delta Lake.**
* **Airflow:**
  + **DAGs, operators, sensors, and XComs.**
  + **Integration with AWS (e.g., S3, Redshift).**
  + **Practice: Orchestrate a multi-step ETL pipeline with Airflow.**

## 1.3 SQL

* **Advanced Queries:**
  + **Window functions, recursive CTEs, and pivot/unpivot.**
  + **Query optimization: indexing, partitioning, and execution plans.**
  + **Practice: Solve complex SQL problems on LeetCode.**
* **Optimization:**
  + **Partitioning, clustering, and materialized views.**
  + **Cost optimization in Redshift/BigQuery/Snowflake.**
  + **Practice: Optimize a slow-running query in Redshift.**

# 2. Core Concepts

## 2.1 Data Governance

* **Data Quality:**
  + **Validation rules, Great Expectations, and anomaly detection.**
  + **Practice: Implement data quality checks in a pipeline.**
* **Security:**
  + **IAM roles, encryption (KMS, TLS), and RBAC.**
  + **Practice: Set up column-level masking in Snowflake.**
* **Compliance:**
  + **GDPR, CCPA, and HIPAA compliance.**
  + **Practice: Design a GDPR-compliant data deletion pipeline.**

## 2.3 Data Modelling

* **Schemas:**
  + **Star, Snowflake, and Data Vault 2.0.**
  + **Slowly Changing Dimensions (SCD Types 1-6).**
  + **Practice: Migrate a 3NF schema to Data Vault 2.0.**
* **Existing Files:**
  + **Retain your current documentation for reference.**

## 2.4 Distributed Systems

* **CAP Theorem:**
  + **Tradeoffs between consistency, availability, and partition tolerance.**
  + **Practice: Design a distributed key-value store.**
* **Fault Tolerance:**
  + **Retries, idempotency, and circuit breakers.**
  + **Practice: Implement a fault-tolerant Kafka consumer.**

# 3. Data Architect

## 3.1 Data Mesh

* **Domain-Driven Design:**
  + **Decentralized data ownership and domain-oriented data products.**
  + **Practice: Design a data mesh for a multinational bank.**

## 3.2 Lakehouse

* **Delta Lake + Databricks:**
  + **Unified batch and streaming, time travel, and schema enforcement.**
  + **Practice: Build a petabyte-scale Lakehouse.**

## 3.3 MLOps

* **Model Lifecycle:**
  + **Feature stores (Feast, Tecton), model versioning, and A/B testing.**
  + **Practice: Deploy a fraud detection model with MLflow.**

# 4. System Design

## 4.1 Case Studies

* **Electoral Bonds:**
  + **Analyze and document the data pipeline for electoral bonds.**
* **Real-Time Fraud Detection:**
  + **Design a system using Kafka, Flink, and Redis.**
  + **Practice: Implement a real-time anomaly detection pipeline.**

## 4.2 Scalability

* **Partitioning and Sharding:**
  + **Hash vs. range partitioning, consistent hashing.**
  + **Practice: Optimize a Spark job for skewed data.**

## 4.3 Cost Optimization

* **Cloud Cost Levers:**
  + **Spot instances, reserved capacity, and storage tiering.**
  + **Practice: Reduce a $10k/month pipeline cost by 50%.**

# 5. Emerging Trends

## 5.1 AI-Native Systems

* **Vector Databases:**
  + **Weaviate, Pinecone, and Milvus.**
  + **Practice: Build a semantic search engine.**

## 5.2 Edge Computing

* **IoT Data Pipelines:**
  + **AWS IoT Greengrass and Azure Edge Zones.**
  + **Practice: Design a real-time anomaly detection system for IoT sensors.**

# 6. Practice

**6.1 SQL**

* **Solve advanced SQL problems on LeetCode and StrataScratch.**

**6.2 Spark**

* **Optimize PySpark jobs for performance and cost.**

# 7. Docs

## 7.1 Interview Prep

* **Behavioral:**
  + **STAR method, resume review, and project stories.**
* **Checklists:**
  + **Technical, system design, and behavioral checklists.**

**1. Core Data Engineering Concepts**

**Topics to Cover:**

* **Data Modeling:**
  + **Relational vs. non-relational data models.**
  + **Star schema, snowflake schema, and data vault modeling.**
  + **Normalization and denormalization.**
* **Data Warehousing:**
  + **Concepts of data lakes, data warehouses, and lakehouses.**
  + **ETL vs. ELT pipelines.**
  + **Data partitioning, indexing, and optimization.**
* **Data Governance:**
  + **Data quality, metadata management, and data lineage.**
  + **GDPR, CCPA, and other compliance frameworks.**
* **Data Pipeline Design:**
  + **Batch vs. stream processing.**
  + **Orchestration tools (e.g., Apache Airflow, AWS Step Functions).**
  + **Fault tolerance and idempotency in pipelines.**

**Advanced Topics:**

* **Data mesh architecture.**
* **Real-time data processing with change data capture (CDC).**
* **Data observability and monitoring.**

**Tools:**

* **Apache Spark, Apache Kafka, Apache Airflow.**
* **Databricks Delta Lake, AWS Glue, AWS Lake Formation.**

**Practical Exercises:**

* **Design a data model for an e-commerce platform.**
* **Build an ETL pipeline using AWS Glue and Databricks.**
* **Implement a CDC pipeline using Kafka and Spark Streaming.**

**2. Databricks Expertise**

**Topics to Cover:**

* **Databricks Platform:**
  + **Workspace, clusters, and notebooks.**
  + **Databricks Runtime and ML Runtime.**
* **Delta Lake:**
  + **ACID transactions, schema enforcement, and time travel.**
  + **Optimizations like Z-ordering and data skipping.**
* **Spark on Databricks:**
  + **Spark SQL, DataFrames, and Datasets.**
  + **Performance tuning (partitioning, caching, broadcast joins).**
* **Databricks Workflows:**
  + **Job scheduling and orchestration.**
  + **Integration with CI/CD pipelines.**

**Advanced Topics:**

* **Delta Live Tables for declarative pipeline development.**
* **Photon engine for accelerated query performance.**
* **Unity Catalog for data governance.**

**Tools:**

* **Databricks CLI, REST API, and SDKs.**
* **Databricks Connect for local development.**

**Practical Exercises:**

* **Build a Delta Lake for a streaming data source.**
* **Optimize a Spark job on Databricks using partitioning and caching.**
* **Create a Databricks workflow to orchestrate a multi-step pipeline.**

**3. AWS Cloud Ecosystem**

**Topics to Cover:**

* **AWS Core Services:**
  + **EC2, S3, IAM, VPC, and CloudWatch.**
* **Data Storage:**
  + **S3, Redshift, RDS, DynamoDB, and Aurora.**
* **Data Processing:**
  + **AWS Glue, EMR, Lambda, and Kinesis.**
* **Orchestration:**
  + **Step Functions, MWAA (Managed Workflows for Apache Airflow).**
* **Serverless Architectures:**
  + **AWS Lambda, API Gateway, and EventBridge.**

**Advanced Topics:**

* **Multi-account architecture and data sharing.**
* **Cost optimization strategies for data pipelines.**
* **Zero-ETL integration between AWS services.**

**Tools:**

* **AWS CLI, SDKs, and CloudFormation.**
* **AWS Data Pipeline, Glue Studio, and Redshift Spectrum.**

**Practical Exercises:**

* **Set up a serverless data pipeline using Lambda and S3.**
* **Build a real-time analytics dashboard using Kinesis and Redshift.**
* **Optimize a Glue ETL job for cost and performance.**

**4. Big Data Technologies**

**Topics to Cover:**

* **Distributed Computing:**
  + **Hadoop ecosystem (HDFS, YARN, MapReduce).**
  + **Apache Spark architecture and execution model.**
* **Stream Processing:**
  + **Apache Kafka, Apache Flink, and AWS Kinesis.**
  + **Windowing, watermarking, and state management.**
* **NoSQL Databases:**
  + **Cassandra, MongoDB, and DynamoDB.**
  + **CAP theorem and consistency models.**

**Advanced Topics:**

* **Event-driven architectures and event sourcing.**
* **Graph processing with Apache Giraph or Neo4j.**
* **Data compression and serialization formats (Avro, Parquet, ORC).**

**Tools:**

* **Apache Spark, Kafka, and Flink.**
* **AWS EMR, MSK (Managed Streaming for Kafka).**

**Practical Exercises:**

* **Process a large dataset using Spark on EMR.**
* **Build a real-time fraud detection system using Kafka and Flink.**
* **Compare performance of Parquet vs. ORC for a given dataset.**

**5. Data Architecture and Design**

**Topics to Cover:**

* **Architectural Patterns:**
  + **Lambda architecture, Kappa architecture.**
  + **Microservices vs. monolithic data architectures.**
* **Scalability and Performance:**
  + **Horizontal vs. vertical scaling.**
  + **Sharding, replication, and caching strategies.**
* **Security:**
  + **Encryption (at rest and in transit).**
  + **IAM roles, policies, and cross-account access.**

**Advanced Topics:**

* **Federated learning and edge computing for data pipelines.**
* **Multi-cloud architectures and hybrid cloud setups.**
* **AI/ML integration in data pipelines.**

**Tools:**

* **Terraform for infrastructure as code (IaC).**
* **AWS Well-Architected Framework.**

**Practical Exercises:**

* **Design a multi-region data architecture for high availability.**
* **Implement a data pipeline with end-to-end encryption.**
* **Use Terraform to provision a data lake on AWS.**

**6. Industry Trends for 2025**

**Emerging Trends:**

* **AI/ML Integration:**
  + **MLOps and data pipelines for model training and deployment.**
  + **Feature stores and model monitoring.**
* **Data Mesh:**
  + **Decentralized data ownership and domain-driven design.**
* **Sustainability:**
  + **Green data engineering and energy-efficient computing.**
* **Quantum Computing:**
  + **Early-stage applications in data processing.**

**Tools to Watch:**

* **Databricks MLflow, AWS SageMaker.**
* **Snowflake, dbt (data build tool).**

**Practical Exercises:**

* **Build a feature store using Databricks Feature Store.**
* **Implement a data mesh architecture for a mock organization.**
* **Explore quantum computing frameworks like Qiskit.**

**7. Documentation Structure**

**Categories:**

1. **Concept Notes:**
   * **Detailed explanations of core concepts.**
   * **Diagrams and flowcharts for architectures.**
2. **Code Snippets:**
   * **Reusable code for common tasks (e.g., Spark transformations, Glue jobs).**
3. **Project Documentation:**
   * **End-to-end documentation for each project (problem statement, solution, challenges, and results).**
4. **Cheat Sheets:**
   * **Quick references for AWS CLI, Spark SQL, and Databricks APIs.**
5. **Interview Prep:**
   * **Common interview questions and answers.**
   * **Case studies and system design scenarios.**

**Tools for Documentation:**

* **Markdown for lightweight documentation.**
* **Notion or Confluence for organizing notes.**
* **GitHub for version control and sharing code.**

**8. Timeline and Milestones**

**Phase 1: Foundation (Months 1-3)**

* **Focus on core data engineering concepts and AWS basics.**
* **Complete 2-3 small projects (e.g., ETL pipeline, data lake setup).**

**Phase 2: Intermediate (Months 4-6)**

* **Dive into Databricks, Spark, and advanced AWS services.**
* **Build 2-3 medium-complexity projects (e.g., real-time analytics, Delta Lake).**

**Phase 3: Advanced (Months 7-9)**

* **Explore data architecture, industry trends, and emerging tools.**
* **Complete 1-2 large-scale projects (e.g., data mesh, AI/ML pipeline).**

**Phase 4: Interview Prep (Months 10-12)**

* **Revise concepts, practice coding, and mock interviews.**
* **Document all projects and create a portfolio.**

**9. Additional Resources**

* **Books:**
  + **"Designing Data-Intensive Applications" by Martin Kleppmann.**
  + **"Spark: The Definitive Guide" by Bill Chambers and Matei Zaharia.**
* **Courses:**
  + **Databricks Academy (official training).**
  + **AWS Certified Data Analytics – Specialty.**
* **Communities:**
  + **Databricks Community, AWS Developer Forums, and Stack Overflow.**

**Data Modeling Architecture Roadmap**

**1. Understanding Core Concepts**

* **What is Data Modeling?**
  + Definition, purpose, and importance.
  + Types of data models: Conceptual, Logical, Physical.
* **Database Fundamentals**
  + SQL vs. NoSQL databases.
  + ACID properties, BASE properties (for NoSQL).
  + Normalization vs. denormalization.
* **Data Warehousing Basics**
  + What is a data warehouse?
  + ETL vs. ELT.
  + Differences between OLTP vs. OLAP systems.

**2. Schema Design & Modeling Techniques**

* **Star Schema**
  + Fact and dimension tables, benefits, and when to use.
* **Snowflake Schema**
  + Normalized dimensions, advantages over Star Schema.
* **Galaxy Schema (Fact Constellation)**
  + When multiple fact tables share dimensions.
* **Data Vault 2.0**
  + Hubs, Links, and Satellites for scalability and auditability.
  + Performance optimization and real-world use cases.
* **Anchor Modeling**
  + Sixth Normal Form (6NF) and its benefits.
  + Temporal data modeling.

**3. Slowly Changing Dimensions (SCD)**

* **Types of SCD**
  + Type 1: Overwrite.
  + Type 2: Historical tracking using new rows.
  + Type 3: Maintaining limited history with new columns.
  + Types 4-6: Hybrid approaches for handling complex scenarios.
* **Implementation Strategies**
  + Implementing SCD Type 2 in a data warehouse using SQL/dbt.

**4. Evolutionary Architecture & Schema Evolution**

* **Handling Schema Changes Over Time**
  + Adding/removing columns, handling breaking changes.
  + Backward compatibility and versioning strategies.
* **Real-Time Schema Evolution**
  + Designing flexible schemas that adapt to data changes.

**5. Performance Optimization & Advanced Architectures**

* **Indexing & Partitioning**
  + Optimizing query performance using indexes.
  + Table partitioning for scalable data processing.
* **Materialized Views**
  + When to use materialized views for performance gains.
* **Hybrid Schema Design**
  + Combining Star, Snowflake, and Galaxy schemas in a single architecture.

**6. Dynamic Data Warehousing & Event-Driven Design**

* **Batch vs. Real-Time Data Processing**
  + Architecting for hybrid data processing.
* **Event-Driven Data Warehousing**
  + Using event-driven architectures (Kafka, Kinesis) for streaming data.

**7. Data Governance & Metadata Management**

* **Data Lineage & Governance**
  + Ensuring traceability and compliance.
* **Apache Atlas for Metadata Management**
  + Building a metadata catalog for enterprise data.

**8. Tools & Technologies**

* **dbt (Data Build Tool)**
  + SQL transformations, modularization, macros, and hooks.
* **ERwin**
  + Creating ER diagrams, forward/reverse engineering.
* **Dataform**
  + Managing transformations and version control for data pipelines.
* **Liquibase & Flyway**
  + Tools for automated schema migrations.

**9. Real-World Projects & Hands-On Practice**

* Build an **e-commerce data model** (Star & Snowflake schemas).
* Implement **SCD Type 2** in a real-world data warehouse.
* Design a **Data Vault 2.0 model** for customer data integration.
* Create a **real-time data pipeline** using schema evolution techniques.
* Implement **data governance** using Apache Atlas.