Database Course Documentation

---

1. 📊 Flat File Systems vs. Relational Databases

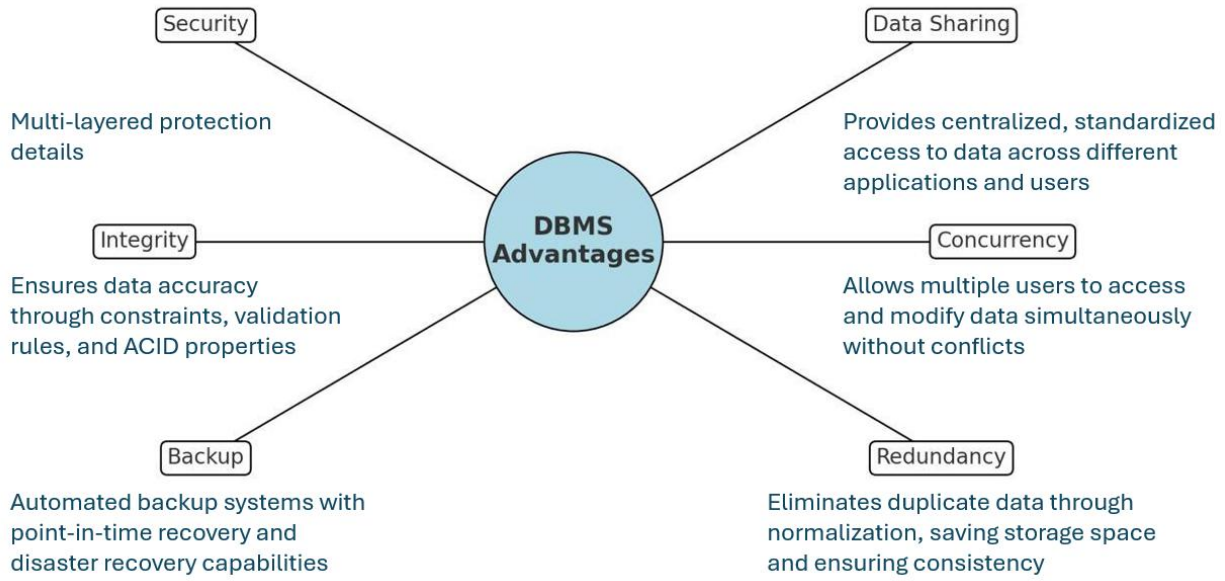
Feature	Flat File Systems	Relational Databases
Structure	Simple text files (CSV, TSV)	Organized in tables with rows and columns
Data Redundancy	High (duplicate entries common)	Low (data normalization reduces repetition)
Relationships	None or hardcoded manually	Support via foreign keys and joins
Example Usage	Small config files, spreadsheets	Web apps, CRMs, finance systems
Drawbacks	Prone to errors, no data integrity checks	Complex setup but better in the long run

💡 Diagram: Conceptual View

```
Flat File System        Relational Database
[Name, Age, City]       +--------------------------+
[Ali, 22, Muscat]       |   Students Table         |
[Ali, 22, Muscat]       +--------------------------+
                        | ID | Name | Age | City   |
                        |----|------|-----|--------|
                        | 1  | Ali  | 22  | Muscat |
```

---

2. DBMS Advantages Mind Map



---

3. Roles in a Database System

Role	Responsibility
System Analyst	Gathers user requirements, plans DB structure based on needs
Database Designer	Designs the schema, tables, and relationships
Database Developer	Writes SQL queries, procedures, triggers
DBA (Administrator)	Maintains the DB, backups, performance tuning
Application Developer	Connects the DB to the frontend/backend systems
BI Developer	Builds dashboards, performs reporting and analytics

---

4. Types of Databases

A. Relational vs Non-Relational

Type	Description	Examples
Relational	Tables with rows/columns (SQL)	MySQL, PostgreSQL
Non-Relational	Key-Value, Documents, Graphs (NoSQL)	MongoDB, Cassandra

B. Centralized vs Distributed vs Cloud

- **Centralized**: All data in one place (e.g. legacy apps)
- **Distributed**: Data is split across servers (e.g. Cassandra)
- **Cloud**: Hosted and scalable services (e.g. Firebase, AWS RDS)

Use Cases:

- Relational: Banking systems
- Non-Relational: Social media feeds
- Cloud DB: E-commerce platforms with high traffic

---

5. Cloud Storage and Databases

What is Cloud Storage?

Cloud storage is a service model that stores data on remote servers accessed via the internet, rather than on local hardware. It provides the foundation for cloud-based database services.

How It Supports Databases:

- Elastic scaling
- Global accessibility
- Backup and recovery built-in

Relationship Between Cloud Storage and Databases

Traditional Setup:

[Application] → [Local Database] → [Local Storage]

Cloud Setup:

[Application] → [Cloud Database Service] → [Cloud Storage Infrastructure]

↓

[Automatic Scaling & Management]

Advantages:

Advantage	Description	Example
Cost Efficiency	Pay-as-you-use model, no upfront hardware costs	AWS RDS pricing based on usage
Scalability	Automatic scaling based on demand	Azure SQL auto-scaling
Maintenance	Automated updates, patches, and maintenance	Google Cloud SQL automated backups

Advantage	Description	Example
🌍 **Global Access**	Access from anywhere with internet	Distributed team collaboration
🛡️ **Security**	Enterprise-grade security managed by providers	AWS encryption and compliance
📈 **High Availability**	Built-in redundancy and failover	Multi-region deployment

❌ Disadvantages:

Challenge	Impact	Mitigation Strategy
🌐 **Internet Dependency**	Service unavailable without connection	Implement offline capabilities
🔒 **Security Concerns**	Data stored on third-party servers	Use encryption and compliance frameworks
💸 **Cost Unpredictability**	Costs can escalate with high usage	Implement monitoring and alerts
🏃 **Vendor Lock-in**	Difficult to migrate between providers	Use multi-cloud strategies
⚡ **Latency Issues**	Network delays affect performance	Choose geographically close data centers
📋 **Compliance**	Regulatory requirements may restrict cloud usage	Verify provider compliance certifications

---

👨‍💻 Author: [Arooba AlFarsi]

This report was prepared for Codeline's Database Day 1 Task
