Database Search and Reporting Task - Full Report

**1. Comparison: Flat File Systems vs Relational Databases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Aspect** | **Flat File Systems** | **Relational Databases** |  |  |
| **Structure** | Simple file storage (CSV, TXT) | Structured tables with keys and constraints |  |  |
| **Data Redundancy** | High redundancy, no relational control | Low redundancy via normalization |  |  |
| **Relationships** | No relationships | Supports relations via foreign keys |  |  |
| **Example Usage** | Simple record keeping, logs | Enterprise applications, CRM, ERP |  |  |
| **Drawbacks** | Poor scalability, no concurrency | Initial setup complexity, needs expertise |  |  |

**2. DBMS Advantages Mind Map**



**3. Roles in a Database System**

System Analyst: Gathers business requirements and defines system needs.

Database Designer: Creates logical and physical data models.

Database Developer: Writes queries, stored procedures, and scripts.

DBA (Admin): Manages access, backups, and performance.

Application Developer: Integrates database with applications.

BI Developer: Creates reports and dashboards using data.

**4. Additional Research Topics**

**Types of Databases**

- Relational: Structured, table-based (e.g., MySQL).

- Non-Relational: Flexible schema (e.g., MongoDB).

- Centralized: Data in one location.

- Distributed: Data spread across multiple sites.

- Cloud: Hosted and scalable (e.g., Amazon RDS).

**Cloud Storage and Databases**

Cloud storage allows hosting and remote access to databases.

Advantages: Scalability, backup, no hardware required.

Disadvantages: Security risks, dependency on internet.

**Database Engines and Languages**

- SQL Server (T-SQL), Oracle (PL/SQL), PostgreSQL (ANSI SQL)

Some SQL standards work across engines, but many dialects differ.

**Database Transfer Between Engines**

Yes, but challenges include syntax differences, data types, and triggers.

Careful planning and testing are required.

**Logical vs Physical Schema**

- Logical Schema: Abstract design (e.g., tables, fields).

- Physical Schema: Actual implementation (e.g., indexes, file storage).

Understanding both helps optimize performance and design.