Crash Course Notes: Relational Database Management System

1. Basic Concepts

- Database Management System (DBMS): A software for storing, retrieving, and managing data efficiently.
  - Advantages over File-Based System: Eliminates redundancy, ensures consistency, supports multiple users.
  - Logical DBMS Architecture:
    - Three-Level Architecture:
      1. External Level: User-specific views.
      2. Conceptual Level: Logical structure of the database.
      3. Internal Level: Physical storage structure.

Diagram: Three-Level Architecture External View | Conceptual Schema | Internal Schema

3. Database Integrity and Normalization

- Normalization: Process to minimize redundancy and dependency.
  - Forms:
    - 1NF: Eliminate repeating groups.
    - 2NF: No partial dependency.
    - 3NF: No transitive dependency.
- Example:

Non-Normalized Table:

ID	Name	Course1	Course2
1	Alice	Math	Science

1NF:

| ID | Name | Course |

|----|-------|---------|

| 1 | Alice | Math |

| 1 | Alice | Science |

4. File Organization

- Storage Techniques:
  - Heap Files: Unordered records.
  - Sequential Files: Sorted by key.
  - Indexed Files: Use indices for faster search.

5. Structured Query Language (SQL)

- Categories:
  - DDL (Data Definition Language): CREATE, ALTER, DROP.
  - DML (Data Manipulation Language): SELECT, INSERT, UPDATE, DELETE.
  - DCL (Data Control Language): GRANT, REVOKE.
  - TCL (Transaction Control Language): COMMIT, ROLLBACK.
- Example Query:

```
SELECT Name, Age
FROM Students
WHERE Age > 18
ORDER BY Name;
```

6. Transaction and Concurrency Management

- Transactions: Ensure database consistency through ACID properties:
  - Atomicity: Complete or none.
  - Consistency: Valid state.

  - Isolation: Independent transactions.

  - Durability: Permanent changes.

- Concurrency Control:

  - Locking Protocols: Ensure transaction isolation.

  - Deadlock Prevention: Avoid circular wait.

7. PL/SQL

- Features: Extends SQL with procedural capabilities.

  - Variables, Loops, Conditional Statements.

  - Triggers: Automatically executes SQL code in response to events.

- Example Trigger:

```
CREATE TRIGGER update_time
BEFORE UPDATE ON Students
FOR EACH ROW
BEGIN
  :NEW.updated_at := SYSDATE;
END;
```