**🗃️ Database Management Systems (DBMS) – Detailed Notes**

**1. 📊 ER Diagrams & Normalization**

**🔹 ER Diagrams (Entity-Relationship Model)**

An ER Diagram is a **visual representation** of the data and its relationships in a system. It helps in designing the **logical structure** of databases.

**🧱 Key Components:**

* **Entity**: A real-world object. Example: Student, Teacher
* **Attributes**: Properties of an entity. Example: Student(Name, RollNo)
* **Primary Key**: Unique identifier of each entity instance
* **Relationship**: Association among entities. Example: Student ENROLLS in Course

**🧠 Types of Attributes:**

* **Simple vs Composite**: Name (simple), FullName = First + Last (composite)
* **Single-valued vs Multi-valued**: Email (single), Phone numbers (multi)
* **Derived**: Age derived from DOB

**🧬 Cardinality:**

* One-to-One, One-to-Many, Many-to-Many

**🔍 Normalization**

Normalization is a process of organizing data to eliminate redundancy and ensure **data integrity**.

**✅ Normal Forms:**

|  |  |  |
| --- | --- | --- |
| Normal Form | Condition | Goal |
| 1NF | Atomic (indivisible) values only | Remove repeating groups |
| 2NF | 1NF + No partial dependency | Remove dependency on part of primary key |
| 3NF | 2NF + No transitive dependency | Remove dependency via non-prime attributes |
| BCNF | Every determinant is a candidate key | Stronger form of 3NF |

**2. 📋 SQL Commands**

**🔹 Structured Query Language (SQL)**

Used to interact with and manipulate relational databases.

**📁 Categories of SQL Commands:**

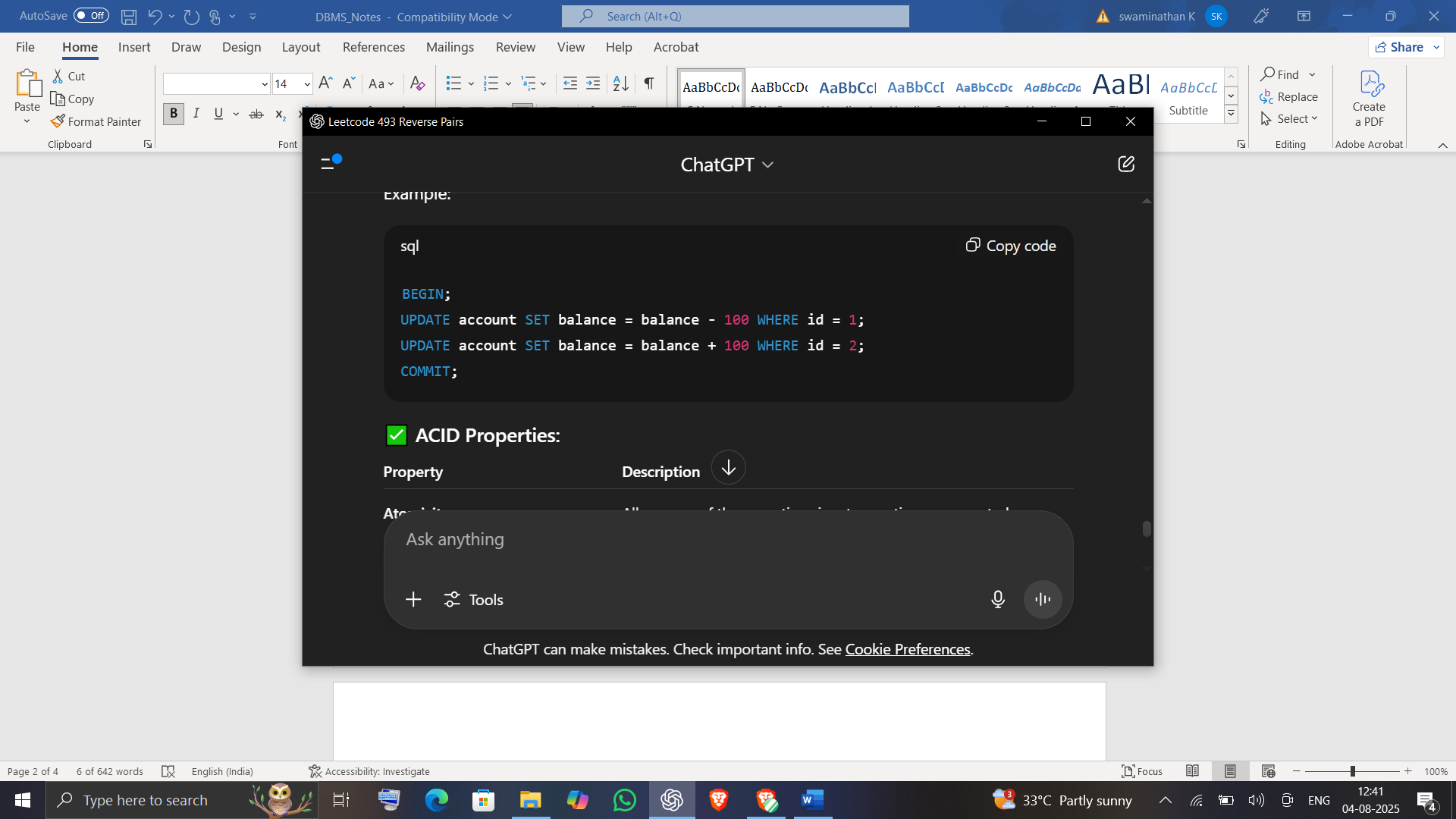
|  |  |  |
| --- | --- | --- |
| Type | Commands | Description |
| DDL | CREATE, ALTER, DROP | Define or modify table structure |
| DML | SELECT, INSERT, UPDATE, DELETE | Manipulate table data |
| TCL | COMMIT, ROLLBACK, SAVEPOINT | Control transactions |
| DCL | GRANT, REVOKE | Manage access permissions |

**3. 🔄 Transactions & ACID Properties**

**🔹 Transaction:**

A transaction is a **group of SQL operations** performed as a **single logical unit of work**.

Example:



**✅ ACID Properties:**

|  |  |
| --- | --- |
| Property | Description |
| Atomicity | All or none of the operations in a transaction are executed |
| Consistency | Brings the database from one valid state to another |
| Isolation | Transactions do not interfere with each other |
| Durability | Once committed, the result remains even after failure |

**4. 🔗 Joins in SQL**

**Joins** combine rows from two or more tables based on a related column.

|  |  |  |
| --- | --- | --- |
| Join Type | Description | Example |
| INNER JOIN | Returns records that have **matching values in both tables** | SELECT \* FROM A INNER JOIN B ON A.id = B.id |
| LEFT JOIN | Returns **all records from the left** table and **matched ones from the right** table | SELECT \* FROM A LEFT JOIN B ON A.id = B.id |
| RIGHT JOIN | Returns **all records from the right** table and **matched ones from the left** table | SELECT \* FROM A RIGHT JOIN B ON A.id = B.id |
| FULL JOIN | Returns **all records from both tables**; unmatched rows contain **NULLs** | SELECT \* FROM A FULL OUTER JOIN B ON A.id = B.id |

**5. 🔐 Concurrency Control**

**Concurrency Control** ensures correctness of database when **multiple transactions** are executed simultaneously.

**🔹 Key Problems:**

* **Lost Update**
* **Temporary Update (Dirty Read)**
* **Inconsistent Analysis (Non-repeatable Reads)**

**🔒 Techniques:**

* **Lock-Based Protocols**:
  + Shared Lock (S): multiple read
  + Exclusive Lock (X): write only
  + **Two-Phase Locking (2PL)**: Growing + Shrinking phase
* **Timestamp-Based Protocols**:
  + Assigns timestamp to transactions and uses to maintain order
* **Optimistic Concurrency Control**:
  + No locks used; validate at commit time

**6. 🚀 Indexing & Query Optimization**

**🔹 Indexing**

An index is a data structure that improves **data retrieval speed** at the cost of additional writes and storage.

|  |  |
| --- | --- |
| Index Type | Description |
| Single-level | Simple list-like structure |
| Multi-level | Uses tree-like structures (e.g., B+ Trees) |
| Clustered Index | Alters table order based on key |
| Non-Clustered Index | Maintains a separate structure with pointers |

**🔍 Query Optimization**

The process of selecting the **most efficient query plan**:

* Use **EXPLAIN** to view plan
* Minimize **joins** and use **indexes**

**💬 Common DBMS Interview Questions**

1. What is the difference between 1NF, 2NF, 3NF, and BCNF?
2. Explain ACID properties with an example.
3. How do INNER JOIN and LEFT JOIN differ?
4. What is a foreign key and how does it enforce referential integrity?
5. Explain two-phase locking protocol.
6. What is indexing and why is it used?
7. Compare pessimistic and optimistic concurrency control.