1. Database-management system (DBMS): A collection of interrelated data and a set of programs to access those data.
2. Database: A collection of interrelated data containing information relevant to an enterprise.
3. Database-system applications: The various uses of database systems in areas like commerce, social media, and science.
4. Online transaction processing: Database applications focused on processing many concurrent transactions.
5. Data analytics: The process of analyzing large datasets to draw conclusions.
6. File-processing systems: An older approach where each application manages its own isolated data files.
7. Data inconsistency: A situation where different files contain conflicting information about the same entity.
8. Consistency constraints: Rules that ensure the data in the database adheres to certain conditions.
9. Data abstraction: The process of hiding the complex details of data storage from users.
10. Physical level: The lowest level of abstraction describing how data are actually stored.
11. Logical level: The next-higher level describing what data are stored and the relationships among them.
12. View level: The highest level describing only part of the entire database for specific users.
13. Instance: The collection of information stored in the database at a particular moment.
14. Schema: The overall design of the database.
15. Physical schema: Describes the physical level of the database.
16. Logical schema: Describes the logical structure of the database.
17. Subschema: A view of the database schema tailored to a particular group of users.
18. Physical data independence: The ability to modify the physical schema without changing the logical schema.
19. Data models: Conceptual tools for describing data, relationships, semantics, and constraints.
20. Entity-relationship model: A data model using entities and relationships among them for database design.
21. Relational data model: A data model that uses a collection of tables to represent data and relationships.
22. Semi-structured data model: A data model that allows data items of the same type to have different sets of attributes.
23. Object-based data model: A category of data models that includes the object-oriented and object-relational models.
24. Database languages: Languages used to interact with a database system.
25. Data-definition language (DDL): A language to specify the database schema.
26. Data-manipulation language (DML): A language that enables users to access or manipulate data.
27. Procedural DML: A DML where the user specifies what data are needed and how to get them.
28. Declarative DML (nonprocedural DML): A DML where the user specifies what data are needed without specifying how to get them.
29. Query language: The part of a DML that involves information retrieval.
30. Domain Constraints: Restrictions on the values that attributes can take.
31. Referential Integrity: A constraint ensuring that foreign key values refer to existing primary key values in another table.
32. Authorization: The granting of rights to users to access and manipulate data.
33. Read authorization: Permission to view data.
34. Insert authorization: Permission to add new data.
35. Update authorization: Permission to modify existing data.
36. Delete authorization: Permission to remove data.
37. Metadata: Data about data, stored in the data dictionary or catalog.
38. Application program: A software program that interacts with a database.
39. Database design: The process of determining the structure and content of a database.
40. Conceptual design: The stage of design that defines the data requirements of the users.
41. Normalization: A set of techniques to design relational database schemas to reduce redundancy and improve data integrity.
42. Specification of functional requirements: Describing the operations or transactions that will be performed on the data.
43. Physical-design phase: The stage where physical storage features of the database are specified.
44. Database Engine: The core software components of a DBMS responsible for managing the database.
45. Storage manager: The component that provides the interface between low-level data and application programs.
46. Authorization and integrity manager: Enforces security and data integrity rules.
47. Transaction manager: Ensures database consistency and atomicity during transactions.
48. File manager: Manages the physical storage of data on disk.
49. Buffer manager: Manages the transfer of data between disk and main memory.
50. Data files: The files where the actual database data are stored.
51. Data dictionary: A repository of metadata about the database.
52. Indices: Data structures that support efficient retrieval of records.
53. Query processor: The component that compiles and executes DDL and DML statements.
54. DDL interpreter: Processes DDL statements.
55. DML compiler: Translates DML queries into low-level instructions.
56. Query optimization: The process of choosing the most efficient way to execute a query.
57. Query evaluation engine: Executes the low-level instructions generated by the DML compiler.
58. Transactions: A sequence of operations performed as a single logical unit of work.
59. Atomicity: The requirement that a transaction be performed in its entirety or not at all.
60. Consistency: The requirement that a transaction preserves the integrity of the database.
61. Durability: The requirement that the changes made by a committed transaction persist.
62. Recovery manager: Ensures atomicity and durability by restoring the database to a consistent state after failures.
63. Failure recovery: The process of restoring the database after a system failure.
64. Concurrency-control manager: Manages simultaneous execution of transactions to avoid conflicts.
65. Database Architecture: The structure of a database system.
66. Centralized: A database system where all components are located on a single computer.
67. Parallel: A database system that uses multiple processors to improve performance.
68. Distributed: A database system where the database is spread across multiple computers.
69. Database Application Architecture: The structure of applications that interact with a database.
70. Two-tier: An architecture where the client application directly communicates with the database server.
71. Three-tier: An architecture with a middle tier (application server) between the client and the database server.
72. Application server: A middle-tier component that handles business logic and data access.
73. Relation: In the relational model, a table with rows (tuples) and columns (attributes).
74. Tuple: A row in a relation, representing a relationship among a set of values.
75. Attribute: A column in a relation, with a unique name.
76. Relation instance: A specific set of tuples in a relation at a given point in time.
77. Domain: The set of permissible values for an attribute.
78. Atomic domain: A domain whose elements are considered to be indivisible units.
79. Null value: A special value indicating an absent or unknown value.
80. Database schema (relational): The logical design of the relational database, including relation schemas.
81. Database instance (relational): A snapshot of the data in the relations at a given instant.
82. Relation schema: The structure of a relation, including the attribute names and their domains.
83. Keys: Attributes or sets of attributes that uniquely identify tuples in a relation.
84. Superkey: A set of attributes whose values uniquely determine a tuple.
85. Candidate key: A minimal superkey (no proper subset is also a superkey).
86. Primary key: A candidate key chosen to be the principal means of identifying tuples.
87. Primary key constraints: Rules that enforce the uniqueness and non-nullness of primary key values.
88. Foreign-key constraint: A rule that ensures values in a foreign key column match values in a primary key column of another table.
89. Referencing relation: The relation that contains the foreign key.
90. Referenced relation: The relation that contains the primary key referred to by the foreign key.
91. Referential integrity constraint: Ensures that references between tables remain valid.
92. Schema diagram: A pictorial depiction of the database schema.
93. Query language types: Categories of languages used to request information.
94. Imperative: A language where the user specifies the sequence of operations.
95. Functional: A language based on applying functions to data.
96. Declarative: A language where the user specifies what data is needed.
97. Relational algebra: A formal query language based on a set of operations on relations.
98. Relational-algebra expression: A sequence of relational algebra operations.
99. Relational-algebra operations: Fundamental operations used to manipulate relations.
100. Select (σ): Retrieves tuples that satisfy a given predicate.
101. Project (Π): Selects specific attributes from a relation.
102. Cartesian product (×): Combines all tuples of two relations.
103. Join (⋈): Combines related tuples from multiple relations based on a condition.
104. Union (∪): Combines tuples from two relations, removing duplicates.
105. Set difference (minus −): Returns tuples in the first relation but not in the second.
106. Set intersection (∩): Returns tuples common to both relations.
107. Assignment (←): Assigns the result of an operation to a relation variable.
108. Rename (ρ): Changes the name of a relation or its attributes.
109. SQL: Structured Query Language, the most widely used relational query language.
110. SQL query structure: The basic components of an SQL query (select, from, where).
111. select clause: Specifies the attributes to be retrieved.
112. from clause: Specifies the tables to be queried.
113. where clause: Specifies the conditions that tuples must satisfy.
114. Multiset relational algebra: An extension of relational algebra that allows duplicate tuples.
115. as clause: Used to rename attributes or relations in SQL.
116. order by clause: Used to sort the results of an SQL query.
117. Table alias: A temporary name given to a table in a query.
118. Correlation name (correlation variable, tuple variable): A temporary name for a tuple in a query, often used in subqueries.
119. Set operations (SQL): Operations like union, intersect, and except to combine the results of queries.
120. union: Combines result sets, removing duplicates (unless union all is used).
121. intersect: Returns common rows between result sets.
122. except: Returns rows in the first result set but not in the second.
123. Aggregate functions (SQL): Functions like avg, min, max, sum, and count that operate on sets of values.
124. avg: Calculates the average value.
125. min: Finds the minimum value.
126. max: Finds the maximum value.
127. sum: Calculates the sum of values.
128. count: Counts the number of values or rows.
129. group by clause: Groups rows with the same values in specified columns.
130. having clause: Filters groups based on a specified condition.
131. Nested subqueries: Queries embedded within another SQL query.
132. Set comparisons (SQL): Operators like in, not in, some, all, exists, unique used to compare sets.
133. lateral clause: Allows a subquery to refer to attributes of preceding tables in the from clause.
134. with clause: Defines common table expressions (CTEs) for use within a query.
135. Scalar subquery: A subquery that returns a single value.
136. Database modification (SQL): Statements to change the data in the database.
137. Delete: Removes tuples from a relation.
138. Insert: Adds new tuples to a relation.
139. Update: Modifies existing tuples in a relation.
140. View (SQL): A virtual table based on the result of a query.
141. View definition: The SQL statement that defines the view.
142. View update: The ability to modify the underlying base tables through a view.
143. Transactions (SQL): A sequence of SQL statements treated as a single unit.
144. Commit work: Makes the changes of the current transaction permanent.
145. Rollback work: Undoes the changes of the current transaction.
146. Atomic transaction: A transaction that is treated as an indivisible unit (all or nothing).
147. Constraints (SQL): Rules to maintain data integrity.
148. Integrity constraints: Rules that ensure data in the database satisfies certain conditions.
149. Unique constraint: Ensures that values in a specified column or set of columns are unique.
150. Check clause: Defines a condition that each tuple must satisfy.
151. Referential integrity (SQL): Ensures foreign key values match primary key values.
152. Cascading deletes: Specifies that when a referenced row is deleted, corresponding rows in the referencing table should also be deleted.
153. Cascading updates: Specifies that when a referenced primary key value is updated, corresponding foreign key values should also be updated.
154. Assertions: Declarative expressions that state conditions that must always be true.
155. Data types (SQL): The types of values that attributes can hold.
156. Date and time types: Data types for storing dates and times.
157. Default values: Values automatically assigned to an attribute when no value is specified during insertion.
158. Large objects (LOBs): Data types for storing large unstructured data.
159. clob (character large object): For storing large text data.
160. blob (binary large object): For storing large binary data.
161. User-defined types: Custom data types created by users.
162. Distinct types: User-defined types that are distinct from their underlying base type.
163. Domains (SQL): Named sets of allowed values for attributes.
164. Type conversions: Changing the data type of a value.
165. Catalogs (SQL): Collections of schemas.
166. Schemas (SQL): Named collections of database objects, such as tables and views.
167. Indices (SQL): Data structures to speed up data retrieval.
168. Privileges (SQL): Authorizations granted to users.
169. Types of privileges: Specific permissions like select, insert, update, delete.
170. JDBC: Java Database Connectivity, a Java API for connecting to databases.
171. ODBC: Open Database Connectivity, a standard API for accessing databases from various applications.
172. Embedded SQL: SQL statements embedded within a host programming language.
173. Embedded database: A database system integrated directly into an application.
174. Stored procedures and functions: Blocks of SQL code that can be stored and executed on the database server.
175. Table functions: Functions that return a table as a result.
176. Parameterized views: Views that can accept parameters.
177. Persistent Storage Module (PSM): An SQL standard for procedural extensions.
178. Exception conditions: Errors or special conditions that can occur during program execution.
179. Handlers: Code blocks that respond to exception conditions.
180. External language routines: Procedures or functions written in languages other than SQL that can be called from within SQL.
181. Sandbox: A secure environment for executing external language routines.
182. Trigger: A database object that automatically executes a predefined SQL code in response to certain events.
183. Transitive closure: Finding all reachable nodes in a graph, often applied to hierarchical data.
184. Hierarchies: Data structures representing parent-child relationships.
185. Create temporary table: A SQL statement to create a table that exists only for the duration of the current session.
186. Base query: The initial query in a recursive query.
187. Recursive query: A query that refers to itself to process hierarchical or recursive data.
188. Fixed point: The result of a recursive query when no new tuples are generated.
189. Monotonic: A property of recursive rules where adding more input data does not remove any existing output data.
190. Windowing: SQL features that allow calculations across a set of table rows that are related to the current row.
191. Ranking functions: SQL functions that assign ranks to rows within a partition of a result set.
192. Cross-tabulation: A way to summarize data in a table format.
193. Transaction Concept: The understanding of a transaction as a logical unit of work.
194. ACID properties: The fundamental properties of transactions: Atomicity, Consistency, Isolation, Durability.
195. Isolation (transaction property): Ensures that concurrent transactions appear to execute in isolation.
196. Volatile storage: Computer memory that loses its content when power is lost.
197. Partially committed: A transaction state after the final operation but before commitment.
198. Failed (transaction state): A state indicating that normal transaction execution cannot proceed.
199. Aborted (transaction state): A state after a transaction has been rolled back.
200. Locking (concurrency control): A mechanism to control access to data items by concurrent transactions.
201. Data normalization (topic): The process of organizing data to reduce redundancy and improve data integrity.
202. Denormalization: The process of adding controlled redundancy to improve query performance.
203. Information system: An organizational system comprising computers, storage, software, and databases.
204. Requirements specification and analysis: The initial phase of database design to understand user needs.
205. Conceptual design (information system): A high-level design representing the data and their relationships.
206. Logical design (information system): Mapping the conceptual design to the data model of the chosen DBMS.
207. Physical design (information system): Specifying the storage and access methods for the database.
208. Data catalog: Another term for the database catalog or data dictionary.
209. Program-data independence: The separation of data descriptions from the application programs that use the data.
210. User view: A subset of the database tailored to a specific user or group of users.
211. DBA (Database Administrator): The person responsible for managing and maintaining the database system.
212. End user: A person who uses the database to perform tasks.
213. Canned transaction: A predefined sequence of database operations for specific tasks.
214. Deductive database system: A database system that can infer new information using deduction rules.
215. Persistent object: An object that continues to exist even after the program that created it terminates.
216. Transaction-processing application: An application designed to handle many concurrent transactions.
217. Online analytical processing: Another term for data analytics.
218. Big data storage systems (NOSQL systems): Database systems designed to manage large volumes of non-traditional data.
219. Self-describing data: Data that includes the data item names and values together.
220. Data normalization (definition from Chapter 1): Designing a database to store each logical data item only once.
221. Deduction rules: Rules defined in a deductive database system for inferring new information.
222. Active database systems: Database systems that can automatically initiate actions based on events and conditions.
223. Workers behind the scene: Personnel involved in the design, development, and operation of the DBMS software.
224. DBMS system designers and implementers: Individuals who design and implement the DBMS software.
225. Naive end users: Users who interact with the database through pre-built applications.
226. Casual users: Users who occasionally query the database.
227. Sophisticated users: Users who are familiar with the DBMS and can formulate complex queries.
228. Standalone users: Users who maintain personal databases using ready-made software.
229. Data independence: The capacity to make changes to one level of the database system without affecting other levels (includes physical and logical data independence).
230. Program-operation independence: The ability to change how operations are implemented without affecting application programs.
231. Data model inherent rules: Rules specific to a particular data model (e.g., a relationship must involve at least two entities in the E-R model).
232. Triggers (definition from Chapter 1): Rules associated with tables that are activated by updates to the table.
233. Stored procedures (definition from Chapter 1): More involved procedures to enforce rules that become part of the database definition.
234. Active rules: Rules in active database systems that can automatically initiate actions.
235. Online transaction processing (OLTP): Systems primarily focused on transaction processing.
236. Data analytics (definition from Chapter 1): Applications focused on analyzing data for business intelligence.
237. NoSQL systems (definition from Chapter 1): Lightweight data management systems that often do not support SQL or strict consistency.
238. Eventual consistency: A consistency model where distributed copies of data may be temporarily inconsistent but will eventually converge.
239. Data warehousing: The collection and management of data from various sources to provide business insights.
240. Data mining: The process of discovering patterns and knowledge from large datasets.
241. Object-oriented model: A data model based on objects, classes, and inheritance.
242. Object-relational model: A database model that combines features of relational and object-oriented models.
243. Database language standards: Efforts to standardize database languages like SQL.
244. Data storage and definition language: A special type of DDL to specify storage structure and access methods.
245. Data dictionary: A special type of table containing metadata.
246. Data manipulation: Includes retrieval, insertion, deletion, and modification of data.
247. Query: A request for the retrieval of information from a database.
248. Query language (definition from Chapter 1): The portion of a DML that involves information retrieval.
249. Query processor components: Modules responsible for processing queries.
250. Query evaluation: The execution of low-level instructions to retrieve query results.
251. Query optimization: The process of finding the most efficient execution plan for a query.
252. Funds transfer (example transaction): An example illustrating the need for atomicity, consistency, and durability.
253. Concurrency control: Techniques to manage simultaneous access to the database by multiple users.
254. High availability: The ability of a system to remain operational despite failures.
255. Data-intensive applications: Applications that handle large amounts of data.
256. Relational Model (Chapter 2 definition): A data model based on a collection of tables (relations).
257. Relation Schema (Chapter 2 definition): Specifies the name of the relation and the attributes (columns).
258. Relation Instance (Chapter 2 definition): A specific occurrence of a relation, containing a set of tuples (rows).
259. Tuple (Chapter 2 definition): A sequence of values representing a single record in a relation.
260. Attribute (Chapter 2 definition): A named column in a relation.
261. Domain (Chapter 2 definition): The set of allowable values for each attribute.
262. Database Schema (Chapter 2 definition): The logical structure of the entire database, consisting of relation schemas.
263. Database Instance (Chapter 2 definition): The data stored in the database at a particular moment in time.
264. Primary Key (Chapter 2 definition): An attribute or set of attributes that uniquely identifies each tuple.
265. Foreign Key (Chapter 2 definition): An attribute in one relation that references a primary key in another relation.
266. Referential Integrity Constraint (Chapter 2 definition): Ensures that foreign key values match existing primary key values.
267. Query Language (Chapter 2 definition): A language for requesting information from the database.
268. Relational Algebra (Chapter 2 definition): A set of operations that take relations as input and produce relations as output.
269. Relation (Chapter 2 expanded definition): A mathematical relation, a subset of the Cartesian product of domains.
270. Tuple Relational Calculus: A declarative query language based on specifying conditions that tuples must satisfy.
271. Domain Relational Calculus: A declarative query language based on specifying conditions that attribute values must satisfy.
272. Schema Diagram (Chapter 2 definition): A visual representation of the database schema, showing relations, attributes, primary keys, and foreign keys.
273. SQL Data Definition (DDL) (Chapter 3 definition): SQL commands for defining, deleting, and modifying relation schemas.
274. SQL Data Manipulation (DML) (Chapter 3 definition): SQL commands for querying and modifying data.
275. Integrity (SQL definition): SQL commands for specifying integrity constraints.
276. View Definition (SQL definition): SQL commands for defining views.
277. Transaction Control (SQL definition): SQL commands for managing transactions (commit and rollback).
278. Embedded SQL and Dynamic SQL (SQL definition): Ways to embed SQL in programming languages.
279. Authorization (SQL definition): SQL commands for specifying access rights.
280. Basic Types (SQL): Built-in data types supported by SQL (e.g., char, varchar, numeric).
281. Null Value (SQL definition): A special value indicating an absent or unknown value.
282. Primary Key (SQL DDL): Specifies the primary key of a table.
283. Foreign Key (SQL DDL): Specifies a foreign key constraint referencing another table.
284. Drop Table: An SQL command to remove a relation.
285. Delete (SQL DML): An SQL command to remove tuples from a relation.
286. Basic Structure of SQL Queries (Chapter 3): Consists of select, from, and where clauses.
287. Distinct (SQL keyword): Used to eliminate duplicate rows in query results.
288. All (SQL keyword): Explicitly specifies that duplicates should not be removed (default).
289. Cartesian Product in SQL: Formed by listing multiple relations in the from clause.
290. Attribute Specification in Select Clause: Using attribute names or '\*' to select columns.
291. Order By Clause (Chapter 3 definition): Used to sort the output of a query.
292. Aggregate Functions (Chapter 3 expanded definition): Functions that summarize data (avg, min, max, sum, count).
293. Group By Clause (Chapter 3 definition): Groups rows with matching values in specified columns.
294. Having Clause (Chapter 3 definition): Filters groups based on a condition.
295. Insert (SQL DML) (Chapter 3 definition): An SQL command to add new tuples.
296. Update (SQL DML) (Chapter 3 definition): An SQL command to modify existing tuples.
297. Natural Join: A join operation that connects tables based on attributes with the same name.
298. Outer Join: Joins that include all tuples from one or both relations, even if there is no match.
299. View Definition (Chapter 4 definition): Creating a virtual table with a name based on a query.
300. Commit Work (Chapter 4 definition): SQL command to finalize a transaction.
301. Rollback Work (Chapter 4 definition): SQL command to undo the changes of a transaction.
302. Atomic Transaction (Chapter 4 definition): A transaction that is treated as an indivisible unit.
303. Constraints (Chapter 4 expanded definition): Rules to maintain data consistency and integrity.
304. Unique Constraint (Chapter 4 definition): Ensures that all values in a column are different.
305. Check Clause (Chapter 4 definition): Defines a condition that must be true for every tuple.
306. Referential Integrity (Chapter 4 definition): Ensures links between tables are valid.
307. Assertions (Chapter 4 definition): Conditions that the database must always satisfy.
308. Data Types (Chapter 4 expanded definition): Includes date, time, LOBs, user-defined types, distinct types, and domains.
309. Catalogs and Schemas (Chapter 4): Mechanisms for organizing database objects.
310. Indices (Chapter 4 expanded definition): Used for efficient query processing and constraint enforcement.
311. Privileges (Chapter 4 expanded definition): Permissions for accessing and manipulating database objects.
312. JDBC (Chapter 5 definition): A Java API for database access.
313. ODBC (Chapter 5 definition): A C API for database access.
314. Embedded SQL (Chapter 5 definition): Embedding SQL in programming languages.
315. Stored Procedures and Functions (Chapter 5 definition): Procedural SQL code stored in the database.
316. Triggers (Chapter 5 expanded definition): Automatic actions in response to database events.
317. Recursive Queries (Chapter 5): SQL queries that refer to themselves.
318. Metadata (Chapter 5 definition): Information about the database structure accessible through APIs like JDBC.
319. Updatable Result Sets: Result sets from queries that can be used to update the underlying data.
320. Embedded Database (Chapter 5 definition): A database system that is integrated into an application.
321. Table Functions (Chapter 5 definition): Functions that return tables.
322. Parameterized Views (Chapter 5 definition): Views that can accept parameters.
323. Persistent Storage Module (PSM) (Chapter 5 definition): SQL standard for procedural extensions.
324. Exception Conditions and Handlers (Chapter 5): Mechanisms for error handling in procedural SQL.
325. External Language Routines (Chapter 5 definition): Procedures/functions in other languages callable from SQL.
326. Sandbox (Chapter 5 definition): A secure execution environment.
327. Transitive Closure (Chapter 5 definition): Finding all reachable elements in a relationship.
328. Hierarchies (Chapter 5 definition): Data with parent-child relationships.
329. Create Temporary Table (Chapter 5 definition): Creating a temporary table.
330. Base Query (Chapter 5 definition): The non-recursive part of a recursive query.
331. Recursive Query (Chapter 5 expanded definition): A query that calls itself.
332. Fixed Point (Chapter 5 definition): The final result of a recursive query.
333. Monotonic (Chapter 5 definition): A property where more input doesn't remove output.
334. Windowing (Chapter 5 definition): SQL features for calculations across related rows.
335. Ranking Functions (Chapter 5 definition): SQL functions for assigning ranks.
336. Cross-tabulation (Chapter 5 definition): Summarizing data in a table format.
337. Business-logic layer: The layer providing a high-level view of data and actions.
338. Data-access layer: The layer providing the interface between business logic and the database.
339. SQL injection attack: A security vulnerability where malicious SQL code is inserted into queries.
340. Connection pool: A cache of database connections for reuse.
341. CRUD: Create, Read, Update, Delete, the basic operations on data.
342. REST: Representational State Transfer, an architectural style for web services.
343. Transaction (Chapter 17 definition): A unit of program execution that accesses and possibly updates data.
344. Atomicity (Chapter 17 definition): All or none of the transaction's operations are reflected in the database.
345. Consistency (Chapter 17 definition): A transaction executed in isolation preserves database consistency.
346. Isolation (Chapter 17 definition): Concurrent transactions appear to each other as if they are executed sequentially.
347. Durability (Chapter 17 definition): Changes made by a committed transaction are permanent.
348. Active (transaction state): The initial state of a transaction.
349. Partially Committed (Chapter 17 definition): After the last statement of a transaction has been executed.
350. Failed (Chapter 17 definition): A state where normal execution can no longer proceed.
351. Aborted (Chapter 17 definition): After a transaction has been rolled back.
352. Locking (Chapter 17 definition): A concurrency control technique where data items are locked before access.
353. Serializable schedules: Transaction execution schedules that are equivalent to some serial execution.
354. Volatile Storage (Chapter 17 definition): Storage that does not survive system crashes (e.g., main memory).