DBMS Multiple Choice Question Examination (Beginner to Intermediate)

Shuvam Banerji Seal
sbs22ms076@iiserkol.ac.in

May 28, 2025

Instructions

- This paper contains 120 multiple-choice questions.
- Complete the paper within 2 hours ie 1 min per question
- Each question has only one correct answer.
- For each correct answer, you will be awarded +4 marks.
- For each incorrect answer, 1 mark will be deducted (-1).
- Unattempted questions will receive 0 marks.
- Please read each question carefully before answering. Some choices may be intentionally confusing.

Syllabus Overview

The questions in this paper are based on the following DBMS topics:

- Intro to DBMS (File System vs DBMS, Architectures)
- Concept of Keys (Candidate, Super, Primary, Alternate, Foreign, Referential Integrity, Number of Super Keys)
- Intro to ER Model (Attributes, Relationships, Mappings)
- Relational Algebra (Selection, Projection, Cartesian Product, Union, Set-Difference, Rename, Intersection, Joins)
- Functional Dependency
- SQL (General, MySQL vs MariaDB, SQLite3 vs MySQL Server, Aliases, Alter/Update, Delete/Drop/Truncate, Constraints, Aggregate Functions, Group By, Having, Order By, Nested/Correlated Queries, WITH Clause, ANY/ALL, IN/NOT IN, EXISTS/NOT EXISTS, Set Operations)
- SQL using Python

A: Intro to DBMS & Basics

1. Which of the following is a primary disadvantage of a traditional file system when compared to a DBMS?
   (a) Faster data retrieval for simple files.
   (b) High data redundancy and inconsistency.
   (c) Built-in complex query capabilities.
   (d) Centralized control and maintenance.

2. In a 3-tier DBMS architecture, the tier responsible for managing the database and executing queries is typically the:
   (a) Presentation tier (Client)
   (b) Application tier (Business Logic)
   (c) Database tier (Data Server)
   (d) Network tier

3. The level of data abstraction that describes *how*

the data is actually stored is the:

(a) View level
(b) Conceptual level
(c) Physical level
(d) External level

4. Which of the following is NOT a typical responsibility of a Database Administrator (DBA)?

(a) Designing the logical and physical schema.
(b) Writing application programs that use the database.
(c) Granting user authority to access the database.
(d) Monitoring performance and tuning the database.

5. Data independence means:

(a) Applications are not dependent on the data.
(b) The database schema can be changed without affecting most application programs.
(c) Data is stored independently of the DBMS software.
(d) Users can access any data without restrictions.

6. A DBMS that primarily uses a client-server model with a dedicated database server process is characteristic of:

(a) Embedded DBMS like SQLite3.
(b) Traditional file systems.
(c) Server-based DBMS like MySQL Server or PostgreSQL.
(d) 1-tier architecture.

7. The 'Conceptual Schema' in a DBMS defines:

(a) The physical storage structures.
(b) The overall logical structure of the entire database for a community of users.
(c) How individual end-users perceive their portion of the database.
(d) The access methods for data retrieval.

8. Which problem is significantly reduced by using a DBMS compared to a file system for managing large datasets?

(a) Need for data backup.
(b) Complexity of data structures.
(c) Concurrent access anomalies.
(d) Hardware costs.

9. In a 2-tier architecture, the application logic typically resides:

(a) Entirely on the client side.
(b) Entirely on the server side.
(c) Split between client and server, or primarily on the client.
(d) On a dedicated middle-tier server.

10. The main purpose of data models in DBMS is:

(a) To specify the hardware requirements for the database.
(b) To provide a way to describe the design of a database at different levels of abstraction.
(c) To define the programming languages used to access the database.
(d) To enforce security protocols.

B: Concept of Keys

11. A superkey is an attribute or a set of attributes that:

(a) Is chosen by the DBA to uniquely identify tuples.
(b) Uniquely identifies each tuple in a relation.
(c) Is a minimal superkey.
(d) References a primary key in another table.

12. If R(A, B, C, D) is a relation schema and A, B is a candidate key, which of the following is always a superkey but not necessarily a candidate key?

(a) A
(b) C, D
(c) A, B, C
(d) B

13. A primary key is:

(a) Any attribute that contains unique values.
(b) A candidate key selected by the database designer to uniquely identify tuples.
(c) Always a single attribute.
(d) The same as a foreign key.

14. An alternate key is:

(a) A key that is not currently used.
(b) Any candidate key that is not selected as the primary key.
(c) A superkey that is not minimal.
(d) A key used for sorting data.

15. Consider a relation R(A, B, C) with candidate keys A and B, C. How many superkeys does R have? (Note: A superkey is any set of attributes that contains a candidate key).

(a) 2
(b) 3
(c) 4
(d) 5

16. A foreign key constraint helps to enforce:
    (a) Entity integrity.
    (b) Domain integrity.
    (c) Referential integrity.
    (d) User-defined integrity.

17. If a foreign key in Table_A references the primary key of Table_B, what happens if a row in Table_B corresponding to a foreign key value in Table_A is deleted, and the referential integrity constraint is 'ON DELETE SET NULL'?
    (a) The deletion in Table_B is disallowed.
    (b) The corresponding rows in Table_A are also deleted.
    (c) The foreign key values in the corresponding rows in Table_A are set to NULL.
    (d) The primary key in Table_B is set to NULL.

18. Consider a relation R(P, Q, R, S) with the only candidate key being P, Q. The total number of superkeys for R is:
    (a) 1
    (b) 2
    (c) 3
    (d) 4

19. Which statement is true about primary keys?
    (a) A primary key can contain NULL values.
    (b) A relation can have multiple primary keys.
    (c) A primary key must uniquely identify all tuples and cannot be NULL.
    (d) A primary key is always composed of a single attribute.

20. What is the main purpose of a candidate key?
    (a) To be a candidate for indexing.
    (b) To be a minimal set of attributes that uniquely identifies a tuple.
    (c) To link tables together.
    (d) To be an attribute that might become a primary key later.

21. If a relation has attributes X, Y, Z and X is a candidate key, and Y is also a candidate key, then:
    (a) X, Y must be a superkey but not necessarily a candidate key.
    (b) Z cannot be part of any other candidate key.
    (c) Both X and Y are minimal superkeys.
    (d) The relation must have at least one foreign key.

22. Referential integrity constraint 'ON UPDATE CASCADE' means:
    (a) If a referenced primary key value is updated, the update is blocked.
    (b) If a referenced primary key value is updated, all corresponding foreign key values are also updated.
    (c) If a foreign key value is updated, the corresponding primary key is also updated.
    (d) If a referenced primary key value is updated, corresponding foreign key values are set to their default.

23. A relation R(EmpID, Name, DeptID, ProjID) has candidate keys EmpID and Name, DeptID. Which is a valid choice for the primary key?
    (a) DeptID, ProjID
    (b) EmpID
    (c) Name
    (d) ProjID

24. Which of the following is NOT a property of a candidate key?
    (a) Uniqueness (no two tuples have the same value for the candidate key).
    (b) Minimality (no proper subset of the candidate key is also a unique identifier).
    (c) It must be a single attribute.
    (d) It can be chosen as a primary key.

25. Consider a relation with attributes A, B, C, D, E. If A,B and C,D are the only candidate keys, how many superkeys containing attribute E (but not A,B or C,D alone) exist?
    (a) 2 (e.g., A,B,E, C,D,E)
    (b) 4
    (c) 6
    (d) A relation R(X, Y, Z) has candidate keys X and Y. How many superkeys does R have?
        i. 2
        ii. 3
        iii. 4
        iv. 5
    (e) A relation R(A, B) has candidate keys A and B. How many superkeys does R have?
        i. 2
        ii. 3
        iii. 4
        iv. 1
    (f) If an attribute in a referencing relation can be NULL, what does this imply for the foreign key constraint?
        i. The foreign key constraint cannot be enforced.
        ii. It means that a referencing tuple may exist without a corresponding referenced tuple.
        iii. The referenced primary key must also allow NULLs (which is false).

iv. The ‘ON DELETE‘ rule must be ‘CASCADE‘.

(g) Entity Integrity constraint states that:
   i. No foreign key value can be NULL.
   ii. No primary key value can be NULL.
   iii. All attributes in a relation must have atomic values.
   iv. Every relation must have at least one foreign key.

(h) Consider a relation ‘STUDENT(StudentID, Name, AdvisorID)‘ where ‘AdvisorID‘ is a foreign key to ‘FACULTY(FacultyID, FName)‘. If ‘AdvisorID‘ in ‘STUDENT‘ is NULL, it means:
   i. The student must have an advisor whose ‘FacultyID‘ is NULL in ‘FACULTY‘.
   ii. The student currently does not have an advisor, or the advisor is unknown.
   iii. This is a violation of referential integrity.
   iv. The ‘FACULTY‘ table must be empty.

(i) A key that consists of more than one attribute is called a:
   i. Simple key
   ii. Composite key
   iii. Domain key
   iv. Partial key

(j) In a relational schema R(A, B, C, D), if A,B is a candidate key, and B,C is also a candidate key. Which of the following MUST be true?
   i. B itself is a candidate key.
   ii. A,C is a candidate key.
   iii. A and C must be functionally dependent on B.
   iv. A,B,C is a superkey.

C: ER Model

(k) An attribute that can be broken down into smaller logical parts is called a:
   i. Simple attribute
   ii. Multivalued attribute
   iii. Composite attribute
   iv. Derived attribute

(l) In an ER diagram, a rectangle represents:
   i. An attribute
   ii. An entity set
   iii. A relationship set
   iv. A constraint

(m) An attribute that can have multiple values for the same entity instance is a:
   i. Key attribute
   ii. Simple attribute
   iii. Multivalued attribute
   iv. Composite attribute

(n) A ’derived attribute’ is one whose value:
   i. Is stored directly in the database.
   ii. Can be computed from other attributes or related entity instances.
   iii. Uniquely identifies an entity instance.
   iv. Links two entity sets.

(o) In an ER diagram, a diamond shape represents:
   i. An entity set
   ii. An attribute of an entity
   iii. A relationship set
   iv. A weak entity set

(p) A one-to-many (1:N) relationship between entity set E1 and entity set E2 means:
   i. One entity in E1 can be related to at most one entity in E2, and one entity in E2 can be related to many in E1.
   ii. One entity in E1 can be related to many entities in E2, but one entity in E2 can be related to at most one entity in E1.
   iii. Each entity in E1 is related to exactly one entity in E2.
   iv. Many entities in E1 can be related to many entities in E2.

(q) ’Cardinality ratio’ of a binary relationship specifies:
   i. The number of attributes in the relationship.
   ii. The maximum number of relationship instances an entity can participate in.
   iii. Whether an entity’s existence depends on another entity.
   iv. The domain of the attributes in the relationship.

(r) A weak entity set is one that:
   i. Has no attributes.
   ii. Cannot be uniquely identified by its attributes alone and relies on an identifying relationship with an owner entity.
   iii. Participates in all relationships.
   iv. Has only multivalued attributes.

(s) In an ER diagram, an underlined attribute within an entity rectangle typically represents:
   i. A multivalued attribute.
   ii. A derived attribute.
   iii. A primary key attribute (or part of it).
   iv. A descriptive attribute.

(t) A many-to-many (M:N) relationship between entity sets E1 and E2 is typically implemented in a relational database by:
   i. Adding a foreign key from E1 to E2.
   ii. Adding a foreign key from E2 to E1.

iii. Creating a new junction/associative table that includes primary keys from both E1 and E2 as foreign keys.
iv. Duplicating attributes from E1 into E2.

(u) 'Participation constraint' (or 'existence dependency') specifies:
i. Whether an entity's participation in a relationship is mandatory (total) or optional (partial).
ii. The maximum number of entities that can participate in a relationship instance.
iii. The attributes of the relationship.
iv. The primary key of the participating entities.

(v) A relationship where an entity instance relates to other instances of the same entity set is called a:
i. Binary relationship
ii. Ternary relationship
iii. Recursive (or unary) relationship
iv. Identifying relationship

(w) If entity set A has a (1:1) relationship with entity set B, and participation of A is total while participation of B is partial, how is this typically mapped to relations?
i. Merge A and B into one table, with PK from A.
ii. Create two tables A and B. Put PK of B as FK in A. FK in A can be NULL.
iii. Create two tables A and B. Put PK of A as FK in B. FK in B can be NULL.
iv. Create two tables A and B. Put PK of A as FK in B. FK in B must be NOT NULL and UNIQUE. (This represents total participation of B side if FK is in B). Correct way: PK of B as FK in A (FK cannot be NULL). Or PK of A as FK in B (FK can be NULL). Since A is total, its instance must relate to B. If FK is in A (referencing B), then this FK cannot be NULL.

(x) A 'double-lined' rectangle in an ER diagram usually signifies:
i. A strong entity set.
ii. A weak entity set.
iii. An associative entity.
iv. A derived entity set.

(y) When mapping a 1:N relationship from E1 (1 side) to E2 (N side) to relational tables:
i. The primary key of E1 is included as a foreign key in the table for E2.
ii. The primary key of E2 is included as a foreign key in the table for E1.
iii. A new table is created with primary keys of both E1 and E2.
iv. Attributes of E1 are merged into E2.

D: Relational Algebra

(z) The SELECT operation (σ) in relational algebra is used to:
i. Select a subset of attributes (columns).
ii. Select a subset of tuples (rows) that satisfy a given condition.
iii. Combine tuples from two relations.
iv. Rename attributes or relations.

() The PROJECT operation (π) in relational algebra:
i. Always preserves the number of tuples.
ii. Selects specified attributes and automatically eliminates duplicate tuples from the result.
iii. Filters rows based on a condition.
iv. Requires the input relation to have at least two attributes.

() If relation R has n tuples and k attributes, and relation S has m tuples and l attributes, the Cartesian product R $\times$ S will have:
i. $n \times m$ tuples and $k \times l$ attributes.
ii. $n + m$ tuples and $k + l$ attributes.
iii. $n \times m$ tuples and $k + l$ attributes.
iv. $\max(n, m)$ tuples and $\max(k, l)$ attributes.

() For the UNION operation (R $\cup$ S) to be valid, R and S must be:
i. Disjoint (have no common tuples).
ii. Have the exact same set of attributes and attribute names.
iii. Union-compatible (same number of attributes, and corresponding attributes have compatible domains).
iv. Have at least one common attribute.

() The SET DIFFERENCE operation (R - S) results in a relation containing:
i. Tuples that are in S but not in R.
ii. Tuples that are in R but not in S.
iii. Tuples that are common to both R and S.
iv. All tuples from R and S, with duplicates removed from S.

() The RENAME operation (ρ) can be used to:
i. Change the data type of an attribute.
ii. Change the name of a relation or its attributes in the result of an expression.
iii. Delete attributes from a relation.
iv. Add new tuples to a relation.

() The INTERSECTION operation (R $\cap$ S) can be expressed using other fundamental relational algebra operations as:
i. R $\cup$ S

   ii. R - (R - S) or S - (S - R) or R - (R - S) which is equivalent to S ∩ R. Correct is R - (R - S).
   iii. R × S
   iv. $\sigma_{condition}$(R ∪ S)

() A NATURAL JOIN (R ⋈ S) between two relations R and S:
   i. Requires the user to specify the join condition explicitly.
   ii. Is equivalent to a Cartesian product followed by a selection based on equality of all common attributes.
   iii. Always results in more tuples than either R or S.
   iv. Can only be performed if R and S have no common attribute names.

() A Conditional Join (or Theta Join) R $\bowtie_\theta$ S is:
   i. A natural join where θ is implicitly equality on common attributes.
   ii. A Cartesian product R × S followed by a selection σ_θ(R × S).
   iii. A join that only works with numeric conditions.
   iv. A join that always removes common attributes.

() A LEFT OUTER JOIN (R S) produces a result that includes:
   i. Only matching tuples from R and S.
   ii. All tuples from R, and matching tuples from S (with NULLs for S attributes where no match).
   iii. All tuples from S, and matching tuples from R (with NULLs for R attributes where no match).
   iv. All tuples from both R and S, with NULLs where matches don't exist.

() Consider relations Student(SID, SName) and Enroll(SID, CID). The expression π_{SName}(Student ⋈ Enroll) will list:
   i. Names of all students.
   ii. Names of students who are enrolled in at least one course.
   iii. SIDs of students who are enrolled.
   iv. SNames and CIDs of enrolled students.

() If relation R has schema (A, B) and S has schema (B, C), then R ⋈ S (Natural Join) will have schema:
   i. (A, B, C)
   ii. (A, B, B, C)
   iii. (A, C)
   iv. (A, B) if B is the only common attribute.

() Which of the following is NOT a basic (fundamental) relational algebra operation? (Basic usually refers to select, project, union, set difference, Cartesian product, rename).
   i. Selection (σ)
   ii. Projection (π)
   iii. Natural Join (⋈)
   iv. Cartesian Product (×)

() The result of $\sigma_{A=10 \text{ AND } B='X'}(R)$ is:
   i. Tuples from R where A=10 OR B='X'.
   ii. Tuples from R where A=10 AND B='X'.
   iii. Attributes A and B from R.
   iv. Tuples from R where A is not 10.

() If R and S are union-compatible and S is a subset of R, then R - S will be:
   i. Empty.
   ii. Equal to R.
   iii. Equal to S.
   iv. Tuples in R that are not in S.

() A FULL OUTER JOIN (R *JOINER* S) ensures that:
   i. All tuples from R are included, S attributes are NULL if no match.
   ii. All tuples from S are included, R attributes are NULL if no match.
   iii. All tuples from both R and S are included, with NULLs used appropriately for non-matching parts.
   iv. Only tuples that have matches in both R and S are included.

() The operation $\pi_{A,B}(\sigma_{C>5}(R))$ first:
   i. Projects attributes A, B then selects tuples where C ¿ 5.
   ii. Selects tuples where C ¿ 5 then projects attributes A, B.
   iii. Performs a join then a projection.
   iv. Renames attributes C.

() If you want to find all employees who earn more than their managers, using Employee(EID, EName, Salary, MgrID) and assuming MgrID references EID, this would likely involve a:
   i. Union operation.
   ii. Self-join (or a join of the relation with a renamed version of itself).
   iii. Simple projection.
   iv. Cartesian product with an unrelated table.

() The DIVISION operation (R ÷ S) is used to find tuples in R that are associated with:
   i. At least one tuple in S.
   ii. No tuples in S.
   iii. All tuples in S (for a specific set of attributes).
   iv. Exactly one tuple in S.

() Renaming an attribute in a relational algebra expression using $\rho_{NewName \leftarrow OldName}(R)$ changes:

i. The stored data in the database.
ii. The attribute's name only for the scope of the current expression's result.
iii. The data type of the attribute.
iv. The number of tuples in the relation.

() If R = { (1,a), (2,b) } and S = { (a,x), (c,y) }. What is $\pi_{1,3}(R \bowtie_{R.2=S.1} S)$? (Assuming attributes are numbered 1,2 for R and 1,2 for S in the join condition context, and 1,2,3,4 for the Cartesian product result). Let R(A,B), S(C,D). R $\bowtie_{B=C}$ S. Cartesian product: (1,a,a,x), (1,a,c,y), (2,b,a,x), (2,b,c,y). Selection B=C: (1,a,a,x). Tuples: (1,a,a,x). Schema (A,B,C,D). $\pi_{A,D}$ of this result: (1,x).

i. { (1,x) }
ii. { (1,a), (2,b) }
iii. { (a,x) }
iv. { (1,a,x) }

() A RIGHT OUTER JOIN R S is equivalent to:

i. S R
ii. R $\bowtie$ S
iii. R $\times$ S
iv. S - R

() The expression $\sigma_P(\sigma_Q(R))$ is equivalent to:

i. $\sigma_{P \text{ OR } Q}(R)$
ii. $\sigma_{P \text{ AND } Q}(R)$
iii. $\sigma_P(R) \cup \sigma_Q(R)$
iv. $\pi_{P,Q}(R)$

() If $\pi_X(R)$ and $\pi_Y(R)$ are two projections from relation R, what can be said about $\pi_X(R) \cup \pi_Y(R)$?

i. It's valid only if X and Y are the same set of attributes.
ii. It's valid only if X and Y are union-compatible (same number and type of attributes).
iii. It will always produce R.
iv. It's equivalent to $\pi_{X \cup Y}(R)$.

() To find tuples that are in R or in S or in both (set union semantics), you use:

i. R $\cap$ S
ii. R $\cup$ S
iii. R $\times$ S
iv. R $\bowtie$ S

E: Functional Dependency

() If X $\rightarrow$ Y is a functional dependency in relation R, it means:

i. For any two tuples t1 and t2 in R, if t1[X] = t2[X], then t1[Y] = t2[Y].
ii. For any two tuples t1 and t2 in R, if t1[Y] = t2[Y], then t1[X] = t2[X].
iii. X and Y must be single attributes.
iv. Y functionally determines X.

() A functional dependency X $\rightarrow$ Y is trivial if:

i. X is a proper subset of Y.
ii. Y is a proper subset of X.
iii. Y is a subset of or equal to X (Y $\subseteq$ X).
iv. X and Y are disjoint.

() Armstrong's Axiom of Transitivity states that if X $\rightarrow$ Y and Y $\rightarrow$ Z hold, then:

i. X $\rightarrow$ Z holds.
ii. Z $\rightarrow$ X holds.
iii. Y $\rightarrow$ X holds.
iv. X $\rightarrow$ YZ holds.

() The closure of a set of attributes X, denoted as X^+, with respect to a set of FDs F, is:

i. The set of all attributes not functionally determined by X.
ii. The set of all attributes that are functionally determined by X using F.
iii. The set of all candidate keys containing X.
iv. The set of all FDs that can be inferred from X.

() If A,B $\rightarrow$ C,D and C $\rightarrow$ E are FDs, which of the following can be inferred by Armstrong's Axioms?

i. A,B $\rightarrow$ E (By Pseudotransitivity or Augmentation + Transitivity)
ii. C $\rightarrow$ A
iii. E $\rightarrow$ C
iv. D $\rightarrow$ A

F: SQL

() What is a primary difference between MySQL and MariaDB?

i. MySQL is open-source, MariaDB is proprietary.
ii. MariaDB was forked from MySQL and aims for drop-in compatibility with added features.
iii. MySQL supports SQL, MariaDB uses a different query language.
iv. MariaDB is an embedded database, MySQL is a server database.

() A key difference between SQLite3 and a MySQL server is:

i. SQLite3 is primarily client-server, MySQL is embedded.
ii. SQLite3 is serverless (embedded in applications), MySQL typically runs as a separate server process.
iii. SQLite3 does not support SQL transactions.
iv. MySQL is limited to single-user access.

() The SQL keyword 'AS' is used for:

i. Type casting.
ii. Creating an alias for a column or table.
iii. Asserting a condition.
iv. Joining tables.

() Which SQL statement is used to modify the structure of an existing table (e.g., add a column)?
i. ‘UPDATE TABLE‘
ii. ‘MODIFY TABLE‘
iii. ‘ALTER TABLE‘
iv. ‘CHANGE TABLE‘

() What is the difference between ‘DELETE‘ and ‘TRUNCATE‘ in SQL?
i. ‘DELETE‘ removes all rows, ‘TRUNCATE‘ removes specific rows based on a condition.
ii. ‘TRUNCATE‘ is slower but logs individual row deletions; ‘DELETE‘ is faster.
iii. ‘DELETE‘ can remove specific rows (with ‘WHERE‘) and can be rolled back (if in a transaction); ‘TRUNCATE‘ removes all rows, is faster, and typically cannot be easily rolled back.
iv. ‘TRUNCATE‘ removes the table structure, ‘DELETE‘ only removes data.

() Which SQL constraint ensures that all values in a column are different?
i. ‘NOT NULL‘
ii. ‘UNIQUE‘
iii. ‘CHECK‘
iv. ‘PRIMARY KEY‘ (PRIMARY KEY also implies UNIQUE and NOT NULL)

() Which aggregate function returns the number of non-NULL values in a specified column?
i. ‘SUM(column)‘
ii. ‘TOTAL(column)‘
iii. ‘COUNT(column)‘
iv. ‘AVG(column)‘

() The ‘GROUP BY‘ clause in SQL is used to:
i. Order the result set.
ii. Filter rows based on a condition.
iii. Arrange identical data into groups, often used with aggregate functions.
iv. Join multiple tables.

() The ‘HAVING‘ clause is used to:
i. Filter rows before grouping.
ii. Filter groups created by the ‘GROUP BY‘ clause based on aggregate function results.
iii. Specify join conditions.
iv. Sort the final result set.

() To sort the results of a SQL query in descending order of a column ‘Salary‘, you would use:
i. ‘ORDER BY Salary ASC‘
ii. ‘SORT BY Salary DESC‘
iii. ‘ORDER BY Salary DESC‘
iv. ‘GROUP BY Salary DESC‘

() A correlated subquery is one where:
i. The inner query is executed only once.
ii. The inner query’s execution depends on values from the outer query’s current row.
iii. The inner query and outer query operate on completely independent data.
iv. The subquery must return multiple columns.

() The ‘WITH‘ clause (Common Table Expression - CTE) in SQL is primarily used to:
i. Define constraints.
ii. Create temporary, named result sets that can be referenced within a single SQL statement.
iii. Update data in multiple tables simultaneously.
iv. Declare variables.

() The ‘ANY‘ operator in SQL, when used with a subquery (e.g., ‘¿ ANY (subquery)‘), returns true if the comparison is true for:
i. All values returned by the subquery.
ii. At least one of the values returned by the subquery.
iii. No values returned by the subquery.
iv. Exactly one value returned by the subquery that must not be NULL.

() The ‘IN‘ operator is logically equivalent to a series of:
i. ‘AND‘ conditions.
ii. ‘OR‘ conditions (e.g., ‘column = val1 OR column = val2 ...‘).
iii. ‘NOT‘ conditions.
iv. ‘LIKE‘ conditions.

() The ‘EXISTS‘ operator returns true if:
i. The subquery returns at least one row.
ii. The subquery returns NULL.
iii. The subquery returns no rows.
iv. The subquery returns an exact match for all columns in the outer query.

() Which SQL set operation returns all distinct rows selected by either query?
i. ‘INTERSECT‘
ii. ‘UNION‘
iii. ‘MINUS‘ (or ‘EXCEPT‘)
iv. ‘UNION ALL‘ (this includes duplicates)

() What does ‘DROP TABLE Students;‘ do?
i. Deletes all data from the ‘Students‘ table, but keeps the table structure.

ii. Deletes specific rows from ‘Students‘ based on a condition.
iii. Completely removes the ‘Students‘ table structure and all its data.
iv. Renames the ‘Students‘ table.

() The ‘CHECK‘ constraint is used to:

i. Ensure a column does not have NULL values.
ii. Define a condition that all values in a column (or row) must satisfy.
iii. Link two tables together.
iv. Uniquely identify each row.

() ‘SELECT AVG(Salary) FROM Employees WHERE Department = ’Sales’;‘ What could cause this query to return NULL or an error, assuming ‘Salary‘ is numeric?

i. If there are no employees in the ’Sales’ department and ‘AVG‘ on an empty set returns NULL.
ii. ‘AVG‘ cannot be used with a ‘WHERE‘ clause.
iii. If any ‘Salary‘ value in ’Sales’ is NULL, ‘AVG‘ always returns NULL. (AVG ignores NULLs unless all are NULL).
iv. If ‘Salary‘ has negative values.

() ‘UPDATE Employees SET Salary = Salary * 1.1 WHERE EmpID = 101;‘ This statement:

i. Changes the structure of the Employees table.
ii. Increases the salary of employee with EmpID 101 by 10%.
iii. Deletes employee 101.
iv. Inserts a new employee record with a calculated salary.

() To list all departments and the number of employees in each, you would use:

i. ‘SELECT Department, TOTAL(Employee) FROM Employees GROUP BY Department;‘
ii. ‘SELECT Department, COUNT(*) FROM Employees HAVING Department;‘
iii. ‘SELECT Department, COUNT(EmpID) FROM Employees GROUP BY Department;‘
iv. ‘SELECT Department, SUM(EmpID) FROM Employees ORDER BY Department;‘

() ‘SELECT Name FROM Students WHERE Age ¿ ALL (SELECT Age FROM Students WHERE Major = ’CS’);‘ This query finds students:

i. Whose age is greater than at least one CS student.
ii. Whose age is greater than the average age of CS students.
iii. Whose age is greater than the age of every CS student.
iv. Who are CS majors and are the oldest.

() The ‘NOT IN‘ operator with a subquery might produce unexpected results if the subquery:

i. Returns many rows.
ii. Returns duplicate values.
iii. Returns a NULL value. (If subquery returns NULL, ‘val NOT IN (..., NULL, ...)‘ might be false or unknown, not necessarily true)
iv. Is correlated.

() ‘SELECT T1.A, T2.B FROM Table1 T1 JOIN Table2 T2 ON T1.ID = T2.ID;‘ The ‘T1‘ and ‘T2‘ are:

i. Column names.
ii. Table aliases.
iii. Conditions.
iv. Function names.

() Which of the following is true about ‘UNION ALL‘ compared to ‘UNION‘?

i. ‘UNION ALL‘ removes duplicate rows; ‘UNION‘ does not.
ii. ‘UNION ALL‘ does not remove duplicate rows and is generally faster; ‘UNION‘ removes duplicates.
iii. ‘UNION ALL‘ requires tables to have different numbers of columns.
iv. ‘UNION ALL‘ can only be used with two tables; ‘UNION‘ can combine more.

() Consider tables ‘Orders(OrderID, CustomerID, OrderDate)‘ and ‘Customers(CustomerID, Name)‘. To find the name of customers who placed an order on ’2023-01-15’:

Listing 1: Query for Q101

```sql
SELECT C.Name
FROM Customers C JOIN Orders O
ON C.CustomerID = O.CustomerID
WHERE O.OrderDate = '2023-01-15';
```

Which choice best describes the above query?

i. It’s syntactically incorrect.
ii. It correctly retrieves the names of customers who ordered on that date.
iii. It will list all customers, and then filter by date.
iv. It should use a ‘LEFT JOIN‘ to be correct.

() ‘SELECT Department FROM Employees WHERE Salary ¿ 50000 EXCEPT SELECT Department FROM Employees WHERE ManagerID IS NULL;‘ This query attempts to find departments:

i. With employees earning ¿ 50000 AND who have managers.

ii. With employees earning ¿ 50000, excluding departments where at least one employee has no manager. (Slightly confusing wording) Correctly: Departments of employees earning ¿ 50000, but not if that department is also a department of an employee with no manager.
iii. Only departments where ALL employees earn ¿ 50000 and have managers.
iv. Only departments of employees who have no manager and earn ¿ 50000.

() What does ‘ALTER TABLE Products ADD CONSTRAINT chk_price CHECK (Price ¿ 0);‘ achieve?

i. Adds a new column ‘chk_price‘.
ii. Adds a check constraint ensuring ‘Price‘ is always positive.
iii. Creates a foreign key named ‘chk_price‘.
iv. Updates all prices to be positive.

() A subquery in the ‘SELECT‘ clause must:

i. Return multiple rows and multiple columns.
ii. Return a single value (scalar subquery).
iii. Always be correlated.
iv. Be used with an aggregate function.

() ‘DELETE FROM Orders;‘ vs ‘TRUNCATE TABLE Orders;‘ Assuming no triggers or complex foreign keys, which is generally true?

i. ‘DELETE‘ is faster.
ii. ‘TRUNCATE‘ uses more transaction log space.
iii. ‘TRUNCATE‘ is usually faster and uses less transaction log space.
iv. Both are identical in performance and logging.

() What is the purpose of ‘ORDER BY NULLS LAST‘ (or ‘NULLS FIRST‘) in some SQL dialects?

i. To exclude NULL values from the sort.
ii. To specify where NULL values should appear in the sorted result set.
iii. To convert NULLs to a specific value before sorting.
iv. It’s not valid SQL syntax.

() If a ‘GROUP BY‘ clause groups by ‘DeptID‘, and the ‘SELECT‘ list contains ‘MAX(Salary)‘, what does ‘MAX(Salary)‘ represent?

i. The overall maximum salary in the entire table.
ii. The maximum salary for each distinct ‘DeptID‘ group.
iii. A syntax error, as ‘Salary‘ is not in ‘GROUP BY‘. (This is false for aggregates)
iv. The sum of salaries for each department.

() ‘WITH RECURSIVE cte$_{n}ameAS(...)SELECT...FROM$

() Is defined multiple times.
() Can reference itself, typically for hierarchical or graph traversal queries.
() Is executed very slowly.
() Can only select from one base table.

26. ‘SELECT Name FROM Products WHERE Price = (SELECT MAX(Price) FROM Products);‘ This query is likely to:

(a) Be inefficient compared to using ‘ORDER BY Price DESC LIMIT 1‘.
(b) Return the name of all products that have the highest price.
(c) Only work if there’s exactly one product with the maximum price.
(d) Cause an error because a subquery in WHERE cannot use aggregates. (This is false)

27. The ‘ALL‘ operator when used as ‘value ¿ ALL (subquery)‘ will evaluate to true if ‘value‘ is greater than:

(a) Any single value returned by the subquery.
(b) The average of values returned by the subquery.
(c) Every value returned by the subquery (or if the subquery returns an empty set).
(d) The sum of values returned by the subquery.

G: SQL using Python

28. In Python’s ‘sqlite3‘ module, which object is primarily used to execute SQL queries?

(a) The ‘Connection‘ object.
(b) The ‘Cursor‘ object.
(c) The ‘Result‘ object.
(d) The ‘Database‘ object.

29. To prevent SQL injection when inserting user-provided data into a database using Python, one should:

(a) Manually concatenate strings to form the SQL query.
(b) Use parameterized queries (e.g., with ‘¿ or ‘
(c) Always encrypt the user data before insertion.
(d) Use ‘eval()‘ on the SQL string.

30. After executing a query like ‘SELECT * FROM users‘ with a Python DB-API cursor, which method would typically retrieve all resulting rows as a list of tuples?

(a) ‘cursor.fetchone()‘
(b) ‘cursor.fetchmany(0)‘

(c) ‘cursor.fetchall()‘
(d) ‘cursor.getresults()‘

31. What is the purpose of ‘connection.commit()‘ in Python’s DB-API when working with databases like SQLite3 or MySQL?
(a) To execute a query.
(b) To save changes made during the current transaction to the database.
(c) To close the database connection.
(d) To rollback the current transaction.

32. Consider the Python ‘sqlite3‘ code:

Listing 2: Python SQLite Code for Q115

```python
import sqlite3
conn = sqlite3.connect('example.db')
cursor = conn.cursor()
name = "O'Malley"
# Which is the safest way to insert name?
# Query Option X: cursor.execute(f"INSERT
    INTO users VALUES ('{name}')")
# Query Option Y: cursor.execute("INSERT
    INTO users VALUES (?)", (name,))
```

Which query option (X or Y) is safer against SQL injection?
(a) Option X is safer.
(b) Option Y is safer.
(c) Both are equally safe.
(d) Neither is safe; a different method is required.

33. If ‘cursor.execute(”SELECT id, name FROM employees”)‘ is run, and then ‘row = cursor.fetchone()‘, what will ‘row‘ typically be if a record is found?
(a) A list containing two elements, e.g., ‘[1, ’Alice’]‘.
(b) A tuple containing two elements, e.g., ‘(1, ’Alice’)‘.
(c) A dictionary, e.g., ‘’id’: 1, ’name’: ’Alice’‘. (Possible with row factories, but tuple is default)
(d) A single string with values concatenated.

34. When using Python to interact with a database, what is the general role of a ”connection string”?
(a) It’s the SQL query itself.
(b) It contains parameters needed to establish a connection to the database server (e.g., host, user, password, database name).
(c) It’s a Python string representing the table schema.
(d) It’s an error message returned by the database.

35. After executing an ‘INSERT‘, ‘UPDATE‘, or ‘DELETE‘ statement using ‘cursor.execute()‘, what attribute of the cursor often provides the number of rows affected (for some database drivers)?
(a) ‘cursor.affected$_rows$‘‘$cursor.rowcount$‘
(b) ‘cursor.num$_rows$‘‘$cursor.changes$‘

36) What does ‘connection.rollback()‘ typically do in a Python DB-API transaction?
(a) Saves all pending changes to the database.
(b) Closes the connection.
(c) Undoes all changes made since the last ‘commit()‘ or the start of the transaction.
(d) Re-executes the last query.

37. To execute multiple SQL statements provided in a single string with ‘sqlite3‘ in Python, one might use:
(a) cursor.execute_many(sql_string)
(b) cursor.executescript(sql_string)
(c) Looping cursor.execute() for each statement manually split.
(d) connection.run$_script(sql\_string)$

Answer Key

1.	1b	**25.**	25(e)ii	**49.**	25()iii	**73.**	25()i	**97.**	25()iii
2.	2c	**26.**	25(f)ii	**50.**	25()ii	**74.**	25()ii	**98.**	25()iii
3.	3c	**27.**	25(g)ii	**51.**	25()ii	**75.**	25()i	**99.**	25()ii
4.	4b	**28.**	25(h)ii	**52.**	25()ii	**76.**	25()ii	**100.**	25()ii
5.	5b	**29.**	25(i)ii	**53.**	25()ii	**77.**	25()ii	**101.**	25()ii
6.	6c	**30.**	25(j)iv	**54.**	25()ii	**78.**	25()ii	**102.**	25()ii
7.	7b	**31.**	25(k)iii	**55.**	25()ii	**79.**	25()iii	**103.**	25()ii
8.	8c	**32.**	25(l)ii	**56.**	25()ii	**80.**	25()iii	**104.**	25()ii
9.	9c	**33.**	25(m)iii	**57.**	25()i	**81.**	25()ii	**105.**	25()iii
10.	10b	**34.**	25(n)ii	**58.**	25()iii	**82.**	25()iii	**106.**	25()ii
11.	11b	**35.**	25(o)iii	**59.**	25()ii	**83.**	25()iii	**107.**	25()ii
12.	12c	**36.**	25(p)ii	**60.**	25()iv	**84.**	25()ii	**108.**	25
13.	13b	**37.**	25(q)ii	**61.**	25()iii	**85.**	25()iii	**109.**	26b
14.	14b	**38.**	25(r)ii	**62.**	25()ii	**86.**	25()ii	**110.**	27c
15.	15d	**39.**	25(s)iii	**63.**	25()ii	**87.**	25()ii	**111.**	28b
16.	16c	**40.**	25(t)iii	**64.**	25()iii	**88.**	25()ii	**112.**	**??**
17.	17c	**41.**	25(u)i	**65.**	25()ii	**89.**	25()ii	**113.**	30c
18.	18d	**42.**	25(v)iii	**66.**	25()i	**90.**	25()i	**114.**	31b
19.	19c	**43.**	25(w)iv	**67.**	25()i	**91.**	25()ii	**115.**	32b
20.	20b	**44.**	25(x)ii	**68.**	25()ii	**92.**	25()iii	**116.**	33b
21.	21c	**45.**	25(y)i	**69.**	25()ii	**93.**	25()ii	**117.**	34b
22.	22b	**46.**	25(z)ii	**70.**	25()ii	**94.**	25()i	**118.**	35b
23.	23b	**47.**	25()ii	**71.**	25()i	**95.**	25()ii	**119.**	36c
24.	24c	**48.**	25()iii	**72.**	25()iii	**96.**	25()iii	**120.**	37b