1. What is a programming language?
2. What are the types of programming languages?
3. What is the difference between a high-level and low-level programming language?
4. What is the difference between a compiled and interpreted language?
5. What is the role of a compiler and an interpreter?
6. What is the difference between static and dynamic typing?
7. What is the difference between strong and weak typing?
8. What is a variable?
9. What is a data type?
10. What is the difference between primitive and non-primitive data types?
11. What is the difference between a statement and an expression?
12. What is a control structure in programming?
13. What is a loop?
14. What is the difference between a while loop and a for loop?
15. What is a function?
16. What is the difference between a function declaration and a function expression?
17. What is recursion?
18. What is a class?
19. What is an object?
20. What is inheritance?
21. What is polymorphism?
22. What is encapsulation?
23. What is abstraction?
24. What is a constructor?
25. What is method overloading and method overriding?
26. What is an interface?
27. What is a package/module?
28. What is exception handling?

29. What is the difference between compile-time and runtime errors?
30. What is the difference between a stack and a heap?
31. What is memory management?
32. What is garbage collection?
33. What is multithreading?
34. What is the difference between parallelism and concurrency?
35. What is a race condition?
36. What is a deadlock?
37. What is a database?
38. What is SQL?
39. What is normalization and denormalization?
40. What is an API?
41. What is RESTful architecture?
42. What is the difference between REST and SOAP?
43. What is version control?
44. What is Git?
45. What is branching and merging in Git?
46. What is a code review?
47. What is debugging?
48. What is unit testing?
49. What is continuous integration?
50. What is the difference between agile and waterfall methodologies?

1. What is procedural programming?
2. What are the key features of procedural programming?
3. What is a procedure or function?
4. How do you define a function in procedural programming?
5. What is variable scope in procedural programming?
6. What is the difference between local and global variables?

7. How do you pass arguments to a function in procedural programming?
8. What is a return statement in procedural programming?
9. What is the difference between call by value and call by reference?
10. How do you handle errors in procedural programming?

1. What is object-oriented programming (OOP)?
2. What are the four pillars of OOP?
3. What is a class?
4. What is an object?
5. What is encapsulation in OOP?
6. What is inheritance in OOP?
7. What is polymorphism in OOP?
8. What is the difference between overloading and overriding?
9. What is a constructor in OOP?
10. What is a destructor in OOP?
11. What is an access modifier in OOP?
12. What is the difference between private, protected, and public access modifiers?
13. What is a static method or variable in OOP?
14. What is the difference between a class and an interface?
15. What is the purpose of abstract classes and methods?
16. How do you achieve data hiding in OOP?
17. What is the difference between composition and inheritance?
18. What is method chaining in OOP?
19. What is the role of the 'this' keyword in OOP?
20. How do you create objects in OOP?

Certainly! Here are some basic interview questions related to classes, objects, inheritance, polymorphism, and encapsulation in object-oriented programming:

Classes and Objects:

1. What is a class in object-oriented programming?
2. How do you define a class in programming?
3. What is an object and how is it different from a class?
4. How do you create an object in programming?
5. What is the constructor of a class and why is it used?
6. What is the destructor of a class and when is it called?
7. How do you define methods in a class?
8. What is the difference between an instance method and a static method?
9. What is the role of the 'this' keyword in a class?

Inheritance: 10. What is inheritance in object-oriented programming?

11. How do you achieve inheritance in programming languages that support it?
12. What is a base class and a derived class?
13. What is the difference between method overloading and method overriding?
14. How does inheritance promote code reusability?
15. What are the different types of inheritance (e.g., single, multiple, multilevel, hierarchical) and how are they implemented?

Polymorphism: 16. What is polymorphism in object-oriented programming?

17. How is polymorphism achieved in programming languages?
18. What is method overriding and how does it relate to polymorphism?
19. What is method overloading and how does it relate to polymorphism?
20. How does polymorphism improve code readability and maintainability?

Encapsulation: 21. What is encapsulation in object-oriented programming?

22. How does encapsulation help in achieving data hiding?
23. What are access modifiers and how are they related to encapsulation?
24. What is the difference between public, private, and protected access modifiers?
25. How do you create encapsulated classes in programming languages that support encapsulation?

Abstraction: 26. What is abstraction in object-oriented programming?

27. How does abstraction help in managing complexity in software systems?
28. What is the difference between abstraction and encapsulation?
29. How do you achieve abstraction in programming languages?
30. Can you provide an example of abstraction in a real-world scenario?

Ambiguity: 31. What is ambiguity in programming languages?

32. How is ambiguity resolved in programming languages?
33. What are the different types of ambiguities that can occur in code?
34. Can you provide an example of an ambiguous statement in a programming language?
35. How can ambiguity be minimized or avoided in programming?

1. What is a DBMS?
2. Why do we need a DBMS?
3. What are the advantages of using a DBMS?
4. What are the disadvantages of using a DBMS?
5. What are the different types of DBMS?

Relational Model: 6. What is a relation in a relational database?

7. What is a tuple and attribute in a relation?
8. What is a primary key?
9. What is a foreign key?
10. What is a candidate key?
11. What is a super key?
12. What is a composite key?

Normalization: 13. What is normalization?

14. Why do we normalize a database?

15. What are the different normal forms?
16. Explain 1NF, 2NF, 3NF, BCNF, and 4NF with examples.
17. What is denormalization?

SQL (Structured Query Language): 18. What is SQL?

19. What are the different types of SQL commands?
20. What is the difference between DDL, DML, and DCL commands?
21. What is a SELECT statement?
22. What is a WHERE clause in SQL?
23. What is a JOIN in SQL?
24. What are the different types of JOINs in SQL?
25. What is a subquery in SQL?

Indexes and Views: 26. What is an index in a database?

27. Why do we use indexes?
28. What is a view in a database?
29. Why do we use views?

Transactions and Concurrency Control: 30. What is a transaction in a database?

31. What are the properties of a transaction (ACID properties)?
32. What is concurrency control?
33. How do you ensure data consistency in a multi-user environment?

Database Design: 34. What is database design?

35. What are the different stages of database design?
36. What is an ER diagram?
37. What is the purpose of normalization in database design?

Database Security: 38. What is database security?

39. How do you ensure data security in a database?
40. What are the different types of database security threats?

Backup and Recovery: 41. What is database backup?

42. Why do we need to backup a database?
43. What is database recovery?
44. What are the different types of database recovery techniques?

Data Integrity: 45. What is data integrity?

46. How do you ensure data integrity in a database?
47. What are the different types of data integrity constraints?

Database Management System (DBMS) Concepts: 48. What is a database schema?

49. What is a database instance?
50. What is the difference between a database and a database management system (DBMS)?

I see, you're looking for questions specifically related to DDL, DML, and other database concepts without reference to specific database languages. Here's a revised list:

DDL (Data Definition Language):

1. What is DDL and how is it used in database management?
2. Can you provide examples of DDL statements?
3. How does DDL differ from DML?
4. What are the main components of a DDL statement?

5. How do you create a table using DDL?
6. What is the purpose of the ALTER TABLE statement in DDL?
7. How do you drop a table using DDL?

DML (Data Manipulation Language): 8. What is DML and what are its main functions?

9. Can you provide examples of DML statements?
10. How does DML differ from DDL?
11. What are the main components of a DML statement?
12. How do you insert data into a table using DML?
13. How do you update existing data in a table using DML?
14. How do you delete data from a table using DML?

Transactions: 15. What is a database transaction?

16. What is the purpose of using transactions in a database?
17. What are the properties of a transaction (ACID properties)?
18. How do you start and end a transaction in a database?

Data Integrity: 19. What is data integrity and why is it important in a database?

20. What are the different types of data integrity constraints?
21. How do you enforce data integrity in a database?

Views: 22. What is a view in a database?

23. How are views used in database management?
24. What are the advantages of using views in a database?
25. How do you create and modify views in a database?

1. **Atomicity:**
    - What is the principle of atomicity in database transactions?
    - How does the database ensure atomicity in a transaction?
    - Can you provide an example of a scenario where atomicity is crucial in a database transaction?
2. **Consistency:**
    - What does consistency mean in the context of ACID properties?
    - How does a database maintain consistency during and after a transaction?
    - Can you explain the role of constraints in ensuring consistency in a database?
3. **Isolation:**
    - What is the isolation property of database transactions?
    - How does isolation prevent concurrency issues in a database?
    - Can you explain the difference between isolation levels like Read Uncommitted, Read Committed, Repeatable Read, and Serializable?
4. **Durability:**
    - What is the durability property of database transactions?
    - How does a database ensure durability of committed transactions?
    - Can you explain the role of transaction logs in ensuring durability?
5. **ACID Compliance:**
    - Why is ACID compliance important in database systems?
    - How do modern database systems ensure ACID compliance?
    - Can you discuss any trade-offs or challenges in maintaining ACID properties in distributed databases?