Below is a comprehensive list of SQL query implementation questions grouped by topics, ensuring coverage of all mentioned topics. The common schema will be used throughout all questions.

Schema Creation

Let's define the schema that will be used for all queries:

1. **Authors**
   - `AuthorID` INT PRIMARY KEY
   - `FirstName` VARCHAR(50)
   - `LastName` VARCHAR(50)
   - `Birthdate` DATE
2. **Books**
   - `BookID` INT PRIMARY KEY
   - `Title` VARCHAR(100)
   - `Genre` VARCHAR(50)
   - `PublishedDate` DATE
   - `Price` DECIMAL(5,2)
   - `AuthorID` INT, FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID)
3. **Customers**
   - `CustomerID` INT PRIMARY KEY
   - `FirstName` VARCHAR(50)
   - `LastName` VARCHAR(50)
   - `Email` VARCHAR(100)
   - `Phone` VARCHAR(15)
   - `JoinDate` DATE
4. **Orders**
   - `OrderID` INT PRIMARY KEY
   - `OrderDate` DATE
   - `CustomerID` INT, FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
5. **OrderDetails**
   - `OrderDetailID` INT PRIMARY KEY
   - `OrderID` INT, FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
   - `BookID` INT, FOREIGN KEY (BookID) REFERENCES Books(BookID)
   - `Quantity` INT
   - `Price` DECIMAL(5,2)

Query Implementation Questions

Introduction to SQL

1. What is SQL and what are its main uses?

2. Explain the difference between DDL, DML, DQL, and DCL.

CREATE

3. Create the `Authors` table.
4. Create the `Books` table with a foreign key referencing `Authors`.
5. Create the `Customers` table.
6. Create the `Orders` table with a foreign key referencing `Customers`.
7. Create the `OrderDetails` table with foreign keys referencing `Orders` and `Books`.

ALTER

8. Alter the `Books` table to add a column `ISBN` (string).
9. Alter the `Customers` table to modify the `Phone` column to be of type `VARCHAR(15)`.

DROP

10. Drop the `ISBN` column from the `Books` table.
11. Drop the `OrderDetails` table.

SELECT

12. Retrieve all records from the `Authors` table.
13. Retrieve the `Title` and `Price` of all books from the `Books` table.
14. Retrieve all records from the `Customers` table.
15. Retrieve all records from the `Orders` table.
16. Retrieve all records from the `OrderDetails` table.

INSERT

17. Insert a new author into the `Authors` table.
18. Insert a new book into the `Books` table.
19. Insert a new customer into the `Customers` table.
20. Insert a new order into the `Orders` table.
21. Insert a new order detail into the `OrderDetails` table.

UPDATE

22. Update the price of a book with `BookID = 1` to 19.99.
23. Update the email of a customer with `CustomerID = 1`.
24. Update the quantity of a specific order detail.
25. Update the genre of a specific book.
26. Update the phone number of a specific customer.

DELETE

27. Delete a book from the `Books` table where `BookID = 5`.

28. Delete a customer from the `Customers` table where `CustomerID = 10`.
29. Delete an order from the `Orders` table where `OrderID = 2`.
30. Delete an order detail from the `OrderDetails` table where `OrderDetailID = 3`.
31. Delete an author from the `Authors` table where `AuthorID = 4`.

WHERE clause

32. Retrieve all books where the `Genre` is 'Science Fiction'.
33. Retrieve all authors born before '1970-01-01'.
34. Retrieve all customers who joined after '2020-01-01'.
35. Retrieve all orders placed on '2021-12-25'.

ORDER BY

36. Retrieve all customers ordered by their `JoinDate` in descending order.
37. Retrieve all books ordered by their `Price` in ascending order.
38. Retrieve all authors ordered by their `LastName` in alphabetical order.

GROUP BY

39. Retrieve the count of books in each genre.
40. Retrieve the average price of books for each author.
41. Retrieve the number of customers who joined each year.
42. Retrieve the total quantity of each book sold.

Functions

Date Functions

43. Retrieve the current date.
44. Retrieve all books published in the last year.
45. Retrieve the age of each author based on their `Birthdate`.

String Functions

46. Retrieve the first 5 characters of the `Title` for all books.
47. Convert the `LastName` of all authors to uppercase.
48. Concatenate `FirstName` and `LastName` of all customers.

Numeric Functions

49. Retrieve the square root of the `Price` for all books.
50. Retrieve the ceiling value of the `Price` for all books.
51. Retrieve the rounded value of the `Price` for all books.

Aggregate Functions

52. Retrieve the total number of books.

53. Retrieve the average price of all books.
54. Retrieve the maximum price of a book.
55. Retrieve the minimum price of a book.
56. Retrieve the total sales amount for each book.

IF and Case

57. Retrieve all books and display 'Expensive' if the price is greater than 20, else 'Affordable'.
58. Retrieve all orders and classify them as 'High Quantity' if the total quantity ordered is more than 10, else 'Low Quantity'.

JOINs (INNER, LEFT, RIGHT, FULL OUTER)

59. Retrieve all books along with their authors' names using an INNER JOIN.
60. Retrieve all customers and their orders using a LEFT JOIN.
61. Retrieve all orders and the customers who placed them using a RIGHT JOIN.
62. Retrieve all books and their orders using a FULL OUTER JOIN.

Subqueries

63. Retrieve all authors who have written more than 5 books.
64. Retrieve all customers who have placed more than 3 orders.
65. Retrieve the book with the highest price.
66. Retrieve the customer who has spent the most money.

Views

67. Create a view that shows the total sales amount for each book.
68. Create a view that shows the total quantity sold for each genre.

CTE

69. Create a CTE to find the top 3 most expensive books.
70. Create a CTE to find the authors who have written books in multiple genres.

Window Function

71. Retrieve the cumulative sales amount for each book.
72. Retrieve the rank of each book based on its price.

Constraints (PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL)

73. Add a UNIQUE constraint to the `Email` column in the `Customers` table.
74. Add a NOT NULL constraint to the `Title` column in the `Books` table.
75. Add a PRIMARY KEY constraint to the `OrderDetailID` column in the `OrderDetails` table.
76. Add a FOREIGN KEY constraint to the `BookID` column in the `OrderDetails` table referencing `Books`.

Data Types

77. Define the appropriate data type for the `Price` column in the `Books` table.
78. Define the appropriate data type for the `Email` column in the `Customers` table.

This list covers a wide range of SQL topics and ensures that students get hands-on practice with various SQL concepts using the common schema.