`FormatQuizToJson` method takes a `NpgsqlCommand` object as input and returns a formatted JSON representation of the quiz data.

Here's how it works:

1. It creates a `Dictionary<int, Dictionary<string, string>>` named `quizDict` to store the quiz data. The outer dictionary uses the question ID as the key, and the inner dictionary stores the question text and the corresponding options.

2. It reads the data from the database using the provided `NpgsqlCommand` object and `NpgsqlDataReader` to iterate over the rows.

3. For each row, it retrieves the question ID, question text, option name, and option text from the reader.

4. It checks if the question

ID already exists in the `quizDict` dictionary. If not, it creates a new entry in the dictionary with the question text and the option text as the initial key-value pairs.

5. If the question ID already exists, it adds the current option text to the existing entry in the dictionary.

6. After reading all the rows, it creates a list named `quizList` to store the quiz objects.

7. It iterates over the entries in the `quizDict` dictionary.

8. For each entry, it creates a dictionary named `optionsDict` to store the options for the current question.

9. It iterates over the options in the entry's value dictionary, excluding the "QuestionText" key, and adds the option name and text to the `optionsDict`.

10. It creates a quiz object using an anonymous type, including the question ID, question text, and options dictionary.

11. It adds the quiz object to the `quizList`.

12. Once all entries have been processed, it serializes the `quizList` to a JSON string using `JsonConvert.SerializeObject`. The `Formatting.Indented` option is used to include indentation for better readability.

13. Finally, it returns the JSON response string.

This method transforms the raw database query result into a structured and formatted JSON representation of the quiz data, making it easier to consume and process in client applications.