This work has been thoroughly reviewed by the reviewer. The authors addressed the challenges of cross-language schema linking, SQL parsing accuracy, and error propagation in Chinese Text-to-SQL tasks by proposing something called FGCSQL. However, there are still aspects that need further explanations. Please find the reviewer's major concerns as follows.

1. In Section 3, I see the authors claimed a three-stage pipeline improves Chinese Text-to-SQL tasks, but where is the theoretical justification for why this specific division (filtering, generating, correcting) is optimal? I would assume a comparison with alternative architectures, such as direct fine-tuning of LLMs without filtering, might be a good one.
2. Section 3, it mentions that the multi-stage pipeline mitigates error propagation, but it does not provide a detailed error breakdown. I didn't follow this part. Is there an ablation study analyzing how errors propagate across the three stages?
3. I see the paper claims that the filtering encoder improves schema linking, but I don't see a qualitative analysis of how well it selects relevant schema items.
4. Section3.2, I see the IECQN method restructures questions, but where is the analysis provided on whether this restructuring biases model outputs or limits its generalization to unseen formats?
5. In addition, the filtering encoder removes irrelevant schema items, but no ablation study investigates cases where important schema items are mistakenly filtered. Also, I see the method uses column data types for filtering, but how it handles ambiguous column names (e.g., "age" appearing in multiple tables)?
6. I see the generative LLM produces SQL, but how it handles multi-table joins?
7. Minor: why was the beam search size set to 16?
8. Section 5, I see it mainly compares FGCSQL with multi-stage models, but omits direct comparison with end-to-end approaches (such as Transformer-based direct SQL generation). I suggest the authors explain in the response letter of why ignoring this direct comparison?
9. I see the use of NatSQL as an intermediate representation, but why NatSQL is superior to other SQL simplification methods?