CHASE: A Large-Scale and Pragmatic Chinese Dataset for Cross-Database Context-Dependent Text-to-SQL

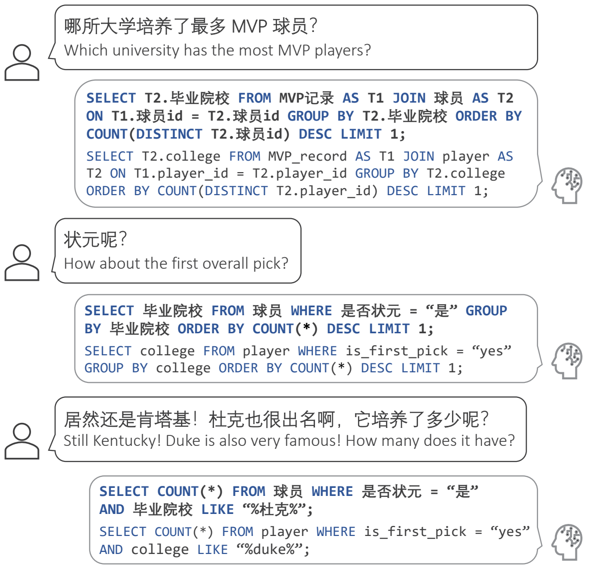
Jiaqi Guo[†][*] Ziliang Si[†] Yu Wang[†] Qian Liu[‡][*] Ming Fan[†]
Jian-Guang Lou[§] Zijiang Yang[†][¶] and Ting Liu[†]
[†]Xi'an Jiaotong University, Xi'an, China [‡]Beihang University, Beijing, China
[§]Microsoft Research, Beijing, China [¶]GuardStrike Inc.
{jasperguo2013,szl123,uyleewang}@stu.xjtu.edu.cn
qian.liu@buaa.edu.cn jlou@microsoft.com
yang@guardstrike.com {mingfan,tingliu}@mail.xjtu.edu.cn

Abstract

The cross-database context-dependent Text-to-SQL (XDTS) problem has attracted considerable attention in recent years due to its wide range of potential applications. However, we identify two biases in existing datasets for XDTS: (1) a high proportion of context-independent questions and (2) a high proportion of easy SQL queries. These biases conceal the major challenges in XDTS to some extent. In this work, we present CHASE, a large-scale and pragmatic Chinese dataset for XDTS. It consists of 5,459 coherent question sequences (17,940 questions with their SQL queries annotated) over 280 databases, in which only 35% of questions are context-independent, and 28% of SQL queries are easy. We experiment on CHASE with three state-of-the-art XDTS approaches. The best approach only achieves an exact match accuracy of 40% over all questions and 16% over all question sequences, indicating that CHASE highlights the challenging problems of XDTS. We believe that CHASE can provide fertile soil for addressing the problems.

1 Introduction

The problem of mapping a natural language utterance into an executable SQL query in the cross-database and context-dependent setting has attracted considerable attention due to its wide range of applications (Wang et al., 2020b; Zhong et al., 2020). This problem is notoriously challenging, due to the complex contextual dependencies among questions in a sequence. Consider the question sequence in Figure 1. In order to understand the last question, one needs to figure out the elliptical object of the verb "培养(have)" from the first two questions in the sequence, which is "状元球员(first pick player)". Questions like this are *context-dependent*, since they require resolutions of contextual dependencies such as *ellipsis* in this question. There are also *context-independent* questions that can be understood individually, such as the first question in Figure 1. For ease of reference, we refer to this cross-database context-dependent Text-to-SQL problem as **XDTS**. To study the challenges in XDTS, a continuous effort has been dedicated to constructing datasets, including SParC (Yu et al., 2019a) and CoSQL (Yu et al., 2019b).



哪所大学培养了最多 MVP 球员?
Which university has the most MVP players?

```
SELECT T2.毕业院校 FROM MVP记录 AS T1 JOIN 球员 AS T2
ON T1.球员id = T2.球员id GROUP BY T2.毕业院校 ORDER BY
COUNT(DISTINCT T2.球员id) DESC LIMIT 1;
SELECT T2.college FROM MVP_record AS T1 JOIN player AS
T2 ON T1.player_id = T2.player_id GROUP BY T2.college
ORDER BY COUNT(DISTINCT T2.player_id) DESC LIMIT 1;
```

状元呢?
How about the first overall pick?

```
SELECT 毕业院校 FROM 球员 WHERE 是否状元 = “是” GROUP
BY 毕业院校 ORDER BY COUNT(*) DESC LIMIT 1;
SELECT college FROM player WHERE is_first_pick = “yes”
GROUP BY college ORDER BY COUNT(*) DESC LIMIT 1;
```

居然还是肯塔基！杜克也很出名啊，它培养了多少呢?
Still Kentucky! Duke is also very famous! How many does it have?

```
SELECT COUNT(*) FROM 球员 WHERE 是否状元 = “是”
AND 毕业院校 LIKE “%杜克%”;
SELECT COUNT(*) FROM player WHERE is_first_pick = “yes”
AND college LIKE “%duke%”;
```

Figure 1: A question sequence from our CHASE dataset. Each question is annotated with its corresponding SQL query. The second and third questions are context-dependent, requiring resolutions of ellipsis.

However, through a careful analysis on existing datasets, we identify two biases in them and these biases conceal the major challenges in XDTS to some extent. First, there are only a limited number of context-dependent questions in existing datasets. Specifically, only 32% of questions in CoSQL are context-dependent, and only 66% of question sequences have context-dependent questions. SParC has more context-dependent ques-

[*] Work done during an internship at Microsoft Research.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2316–2331
August 1–6, 2021. ©2021 Association for Computational Linguistics

tions than CoSQL, but it still has 48% of context-independent questions. Such a limited number of context-dependent questions is unexpected, because prior work (Bertomeu et al., 2006) has shown that questions within a database dialogue are highly likely to be context-dependent, and how to effectively model the context to understand a context-dependent question is one of the major challenges in XDTS. Second, 40% of SQL queries in both SParC and CoSQL are particularly easy, involving at most one condition expression. This biased distribution of SQL queries is potentially caused by their construction methods. In fact, we find that SQL queries for question sequences created from scratch are much more challenging.

Upon identifying the limitations of existing datasets, we present **CHASE**, a large-scale and pragmatic Chinese dataset for XDTS. CHASE consists of 5,459 question sequences (17,940 questions with their SQL queries annotated) over 280 multi-table relational databases. Compared with SParC and CoSQL, the number of context-independent questions in CHASE is reduced from 48% and 68% to 35%, and the number of easy SQL queries is reduced from 40% and 41% to 28%. Moreover, CHASE has richer semantic annotations, including the contextual dependency and schema linking (Lei et al., 2020) of each question. CHASE is also the first Chinese dataset for XDTS.

CHASE is made up of two parts: CHASE-C and CHASE-T. In CHASE-C, we recruit 12 Chinese college students who are proficient in SQL to create question sequences from scratch and annotate corresponding SQL queries. To ensure the diversity and cohesion of question sequences, we propose an intent recommendation method. When a student is going to raise a question, an intent category is randomly sampled with the method, and the student is recommended to write the question and SQL query according to it. In CHASE-T, inspired by the construction of CSpider (Min et al., 2019), we translate all the questions, SQL queries, and databases in SParC from English to Chinese. We also try our best to mitigate the biases in SParC.

To understand the characteristics of CHASE, we conduct a detailed data analysis and experiment with three state-of-the-art (SOTA) XDTS approaches, namely, EditSQL (Zhang et al., 2019), IGSQL (Cai and Wan, 2020), and our extension of RAT-SQL (Wang et al., 2020a). The best approach only achieves an exact match accuracy of 40% over all questions and 16% over all question sequences, indicating that CHASE presents significant challenges for future research. The dataset, benchmark approaches, and our annotation tools are available at `https://xjtu-intsoft.github.io/chase`.

In summary, this paper makes the following main contributions:

- We identify two biases in existing datasets for XDTS: (1) a high proportion of context-independent questions and (2) a high proportion of easy SQL queries.
- We propose an intent recommendation method to guide the question sequence creation. The analysis on CHASE shows that our method is useful to enrich the diversity and cohesion of question sequences.
- CHASE, to the best of our knowledge, is the first large-scale and pragmatic Chinese dataset for XDTS. Experimental results on CHASE with three state-of-the-art approaches show that there is still a long way to solve the challenging problems of XDTS.

2 Study of Existing Datasets

In this section, we first formally define the problem of XDTS and its evaluation metrics. Then, we present our study to understand the limitations and biases of existing datasets in *Contextual Dependency* and *SQL Hardness Distribution*.

2.1 Definition of XDTS

Let $\mathcal{Q}^i = \langle q^i_1, \cdots, q^i_n \rangle$ and $\mathcal{Y}^i = \langle y^i_1, \cdots, y^i_n \rangle$ denote a question sequence and its SQL queries, where q^i_j is the j-th question in $\mathcal{Q}^i$ and y^i_j is the corresponding SQL query for q^i_j. Given a database $\mathcal{DB}^i$, a question q^i_j, and the question's *context* $\langle q^i_1, \cdots, q^i_{j-1} \rangle$, the goal of XDTS is to generate the SQL query y^i_j for q^i_j. An XDTS dataset is a set of question sequences $\{\mathcal{Q}^i, \mathcal{Y}^i, \mathcal{DB}^i\}^N_{i=1}$.

Two metrics are widely used to evaluate the prediction accuracy for XDTS: *Question Match* and *Interaction Match*. Question Match is 1 when the predicted SQL query of q^i_j matches y^i_j.[1] Interaction Match is 1 when all predicted SQL queries of $\mathcal{Q}^i$ match $\mathcal{Y}^i$.

[1] Following (Yu et al., 2018), we decompose a predicted query into different clauses, such as SELECT, WHERE, and compute scores for each clause using set matching separately.

2.2 Study Setup

Dataset There are two datasets for studying XDTS, all of which are English corpora.

(1) SParC (Yu et al., 2019b) SParC is the first dataset for XDTS. It is constructed upon the Spider dataset (Yu et al., 2018). Given a pair of question and SQL query chosen from Spider, an annotator was asked to write a sequence of questions to achieve the gold specified in the chosen pair.

(2) CoSQL (Yu et al., 2019a) CoSQL is a corpus for task-oriented dialogue. It uses SQL queries for dialogue state tracking. Hence, it is also used to study XDTS. Question sequences in CoSQL were collected under the Wizard-of-Oz setup (Kelley, 1984). An annotator was assigned a pair of question and SQL query chosen from Spider, and she was asked to raise interrelated questions towards the goal specified in the pair. Another annotator wrote the SQL query for the question if it was answerable.

Benchmark Approach We consider three SOTA approaches as our benchmark approaches to understand the characteristics of existing datasets: EditSQL (Zhang et al., 2019), IGSQL (Cai and Wan, 2020), and RAT-Con. RAT-Con is our extension of RAT-SQL (Wang et al., 2020a), which is the SOTA approach for the context-independent Text-to-SQL problem. Appendix A.1 provides the details of our extension. All of the three approaches utilize BERT (Devlin et al., 2019) for encodings.

2.3 Contextual Dependency

Prior work (Bertomeu et al., 2006) on database question answering dialogues reveals that questions within a dialogue tend to be context-dependent, i.e., the meaning of a question cannot be understood without its context. The last two questions in Figure 1 are typical context-dependent questions, requiring resolutions of ellipsis. In fact, how to effectively model the context to understand a context-dependent question is one of the major challenges in XDTS (Liu et al., 2020). Hence, we study this characteristic of existing datasets to understand how pragmatic and challenging they are.

To measure the contextual dependency of an XDTS dataset, we manually classify all the questions in its development set into context-dependent and context-independent. If a question is context-dependent, we further label whether it has *coreference* or *ellipsis*, which are two frequently observed linguistic phenomena in dialogues (Androutsopoulos et al., 1995). Note that a question can have both coreference and ellipsis. Each question is first classified by one author of this paper, and then cross-checked and corrected by another.

Dataset	Context Independent	Context Dependent		
		Overall	Coreference	Ellipsis
SParC	47.5%	52.5%	36.6%	20.9%
CoSQL	**68.2%**	31.8%	18.1%	4.9%
Chase	35.3%	64.7%	36.2%	29.0%
Chase-C	28.8%	**71.2%**	**40.3%**	**31.4%**
Chase-T	42.2%	57.8%	33.1%	26.4%

Table 1: Measurement of Contextual dependency. 8.8% of context-dependent questions in CoSQL do not have coreference or ellipsis phenomena.

Dataset	Approach	Question Match (%)			Interaction Match (%)
		Overall	Indep.	Dep.	
SParC	EditSQL	47.1	58.3	37.0	29.4
	IGSQL	49.5	59.4	40.7	30.1
	RAT-Con	**60.1**	**67.4**	**53.5**	**38.6**
CoSQL	EditSQL	39.9	47.1	24.5	12.3
	IGSQL	42.6	50.1	26.4	14.7
	RAT-Con	**50.8**	**57.1**	**37.5**	**20.1**

Table 2: Experimental results on the development set of SParC and CoSQL. ‘Indep.’ and ‘Dep.’ are short for ‘context-independent’ and ‘context-dependent’.

As shown in Table 1, there are only a limited number of context-dependent questions in existing datasets. Specifically, only 32% of questions in CoSQL are context-dependent, and the remaining 68% questions can be understood without the context. Among the 293 question sequences in the development set of CoSQL, 34% of them do not have any context-dependent question. Table 15 in Appendix provides a set of CoSQL question sequences and our classification results. Compared with CoSQL, SParC has more context-dependent questions and more questions that require resolutions of coreference and ellipsis. Nevertheless, 48% of its questions are still context-independent.

Table 2 shows the Question Match (QM) and Interaction Match (IM) of our benchmark approaches on SParC and CoSQL. The QM on context-dependent questions is substantially lower than that on context-independent ones, showing that it is challenging for SOTA approaches to generate SQL queries for context-dependent questions. In view of this challenge and the limited number of context-dependent questions in existing datasets, it is necessary to construct a more pragmatic dataset, involving more context-dependent questions, for studying XDTS.

Dataset	Easy	Medium	Hard	Extra Hard
SParC	40.1%	36.7%	12.1%	11.1%
CoSQL	**41.4%**	31.8%	16.2%	10.5%
CHASE	27.7%	**37.5%**	18.8%	16.0%
CHASE-C	18.6%	37.3%	**24.4%**	**19.7%**
CHASE-T	37.4%	37.8%	12.8%	12.0%

Table 3: SQL hardness distribution.

2.4 SQL Hardness Distribution

SQL hardness is defined as a four-level complexity for SQL queries: *easy*, *medium*, *hard*, and *extra hard*, according to the number of components, selections, and conditions in a SQL query (Yu et al., 2018). The more components a SQL query has, the more complex it is. Intuitively, the more hard and extra hard SQL queries a dataset has, the more challenging the dataset is.

Table 3 presents the SQL hardness distribution in the development set of SParC and CoSQL. We can observe a biased distribution in both datasets, i.e., more than 40% of SQL queries are easy. This biased distribution is potentially caused by their construction methods. Take SParC as an example. A question sequence is constructed by decomposing a complex SQL query into multiple thematically related ones. Although this method is cost-effective, there is little chance that a SQL query is more complicated than the one that it is decomposed from. As we will show in Section 4.3, the SQL hardness distribution of question sequences created from scratch differs a lot from those created via decomposition.

3 Dataset Construction

Given the limitations of existing datasets, we present **CHASE**, a large-scale and pragmatic Chinese dataset for XDTS. Unlike the construction of SParC and CoSQL, we do not specify a final goal for each question sequence. Instead, we motivate our annotators to raise diverse and coherent questions via an intent recommendation method. Based on this method, we collect a set of relational databases, and we recruit annotators to create question sequences from scratch and annotate corresponding SQL queries. Data collected in this way are referred as **CHASE-C**.

Besides, inspired by the construction of CSpider (Min et al., 2019) and Vietnamese Spider (Tuan Nguyen et al., 2020), we translate all the questions, SQL queries, and databases in SParC from English to Chinese. During translation, we also try out best to mitigate the biases in SParC. Data collected with this method are referred as **CHASE-T**. CHASE is make up of both CHASE-C and CHASE-T.

Since all existing datasets for XDTS are constructed for English, prior work on this problem primarily focuses on English, leaving other languages underexplored. To enrich the language diversity, in this paper, we construct CHASE for Chinese, and we leave the support of more languages as our important future work.

3.1 Intent Recommendation

In XDTS, the intent of a question q_j^i is fully reflected by its SQL query y_j^i. Hence, by defining a rich set of relations between y_{j-1}^i and y_j^i, we can derive diverse y_j^i based on y_{j-1}^i. Consequently, we can motivate annotators to raise questions with diverse intents. We define four basic intent categories of relations between y_{j-1}^i and y_j^i:

(1) Same Instances. y_j^i focuses on the other properties of the instances queried in y_{j-1}^i, e.g., by replacing columns in the `SELECT` clause of y_{j-1}^i.

(2) Different Instances of the Same Entity. y_j^i queries the same type of entity and properties as in y_{j-1}^i, but it focuses on different instances, e.g., by adding an extra condition in the `WHERE` clause.

(3) Different Entity. y_j^i queries a different type of entity than y_{j-1}^i, e.g., by altering the tables in the `FROM` clause of y_{j-1}^i.

(4) Display. y_j^i alters the way to display the information queried in y_{j-1}^i, e.g., by adding an `ORDER BY` clause or `DISTINCT` in the `SELECT` clause.

We define 16 relations in these four categories, and we also allow combinations of them. Due to the limit of space, we only present 8 relations with their examples in Table 4. Complete relations are available in Table 12 of Appendix.

When an annotator is going to raise a follow-up question, one of the five intent categories in Table 4 will be randomly selected. The annotator is then recommended to choose a relation belonging to the selected category and raise the question according to the relation. Also, the annotator is allowed to change the intent category when it is not applicable or she has a better choice. With this intent recommendation method, follow-up questions will be closely related to their previous questions and present rich intent diversity.

Category	Relation	Example	
		Precedent SQL Query y^i_{j-1}	Current SQL Query y^i_j
Same Instances	R1. Add Property	*select name from student;*	*select name,* ***age*** *from student;*
	R2. Add Group	*select count(*) from student;*	*select* ***country****, count(*) from student* ***group by country****;*
Different Instances of the Same Entity	R3. Subset	*select name from student;*	*select name from student where* ***country = "US"****;*
	R4. Overlap	*select name from student join student_course where course_name = "Python";*	*select name from student join student_course where course_name =* ***"C++"****;*
Different Entity	R5. Change Entity	*select name from student;*	*select* ***course_name*** *from* ***course****;*
Display	R6. Add Order	*select country, count(*) from student group by country;*	*select country, count(*) from student group by country* ***order by count(*)****;*
	R7. Distinct	*select country from student;*	*select* ***distinct*** *country from student;*
Combination	R8. Add Property (R1) & Subset (R3)	*select name from student;*	*select name,* ***age*** *from student where* ***country = "US"****;*

Table 4: A subset of relations between precedent SQL query y^i_{j-1} and current SQL query y^i_j.

3.2 Construction of CHASE-C

Data in CHASE-C are collected in three stages: (1) database collection; (2) question sequence creation; and (3) data review.

3.2.1 Database Collection

We collect 120 Chinese multi-table relational databases from the DuSQL dataset (Wang et al., 2020c). There are 200 databases and 813 tables in DuSQL, but most of the tables are crawled from encyclopedias and forums. Hence, there are a lot of missing entries and noises (e.g., duplicated or conflicted columns, tables in a database describing unrelated topics, and missing foreign key constraints). To obtain high-quality databases, we manually revise all the databases, dropping those without related tables, resolving duplicated or conflicted columns, and complementing missing entries. As a result, we collect 120 high-quality databases, covering 60 different domains such as Sport, Education, and Entertainment.

3.2.2 Question Sequence Creation

We recruit 12 Chinese college students that are skilled at SQL to create question sequences for databases from scratch. They are also asked to write the SQL query for each question. When a student starts a question sequence creation session, she is shown all the contents from a database, and she can get familiar with the database by executing arbitrary SQL queries. Once she gets ready, she will receive a specification of the minimum number of questions in the sequence.[2] She can raise the first question with her interests. Take the creation of question sequence in Figure 1 as an example. The student asks the first question "哪所大学培养了最多MVP球员？" and writes its corresponding SQL query. The execution results of the SQL query will be shown to the student, helping her raise the follow-up question. After that, she receives the intent category *Different Instances of the Same Entity*, which is randomly sampled by our annotation tool.[3] She chooses the *Overlap* relation in this category and raises the second question "状元呢？". This creation session continues until the minimum number of questions is reached.

To help study the characteristics of questions and address the schema linking challenge (Guo et al., 2019b; Lei et al., 2020) in Text-to-SQL, we also ask the students to label each question's contextual dependency as in Section 2.3 and the linking between database schema items (tables and columns in databases) and their mentions in questions.

3.2.3 Data Review

To ensure the data quality, we conduct two rounds of data review. First, when a student creates her first 20 question sequences, we carefully review all the annotations to check whether the questions in each sequence are thematically related and whether the semantics of SQL queries match their questions. If not, we run a new round of training for the student. Through this round of review, we can resolve misunderstandings of annotations as early as possible. After the finish of the question sequence creation stage, we review all the question sequences like in the first round, and we ask the students to modify their annotations if there are any problems.

[2] Following (Yu et al., 2019b), the minimum number of questions in a sequence ranges from 3 to 5.

[3] Appendix A.2 provides an introduction of our annotation tool for question sequence creation.

Dataset	Language	# DB	# Table	# Seq.	# Pair	# Avg. Turn	# Avg. Qlen	Contextual Dependency	Schema Linking
ATIS	English	1	27	1,658	11,653	**7.0**	10.2	✗	✗
SParC	English	200	1,020	4,298	12,726	3.0	8.1	✗	✗
CoSQL	English	200	1,020	3,007	15,598	5.2	11.2	✗	✗
CHASE	Chinese	**280**	**1,280**	**5,459**	**17,940**	3.3	13.0	✓	✓
CHASE-C	Chinese	120	462	2,003	7,694	3.8	**14.3**	✓	✓
CHASE-T	Chinese	160	818	3,456	10,246	3.0	12.1	✓	✓

Table 5: Statistics of CHASE and existing datasets for the context-dependent Text-to-SQL problem.

3.3 Construction of CHASE-T

The original SParC dataset consists of 4,298 question sequences and 200 databases, but only 3,456 and 160 of them are publicly available for training and development. Hence, we could only translate those to construct CHASE-T.

The translation work is performed by 11 college students, 10 of whom also participate in the question sequence creation stage of CHASE-C. Each database and all its question sequences are translated by one student. The student also needs to label each question's contextual dependency and the linking between schema items and their mentions in the translated questions. We encourage the student to translate a question based on its semantics to obtain the most natural question in Chinese.

To mitigate the biases in SParC, we ask our students to modify those context-independent or thematically unrelated questions and SQL queries to make the question sequences more coherent and natural. Our intent recommendation method is also applied to guide the modification. To ensure the data quality, we also run a two-round data review as in Section 3.2.3.

During the construction of CHASE-T, we identified and fixed 150 incorrect SQL queries in SParC.[4] Also, we modified 1,470 SQL queries to make the question sequences in CHASE-T more coherent.

4 Data Statistics and Analysis

We compute the statistics of CHASE and conduct a thorough analysis to understand its three characteristics: contextual dependency, SQL hardness distribution, and mention of database schema items.

4.1 Data Statistics

Table 5 summarizes the statistics of CHASE. CHASE has 5,459 questions sequences (17,940 questions with their corresponding SQL queries annotated) over 280 databases. CHASE-C contributes 37% question sequences and 43% question-SQL pairs; CHASE-T takes the rest part. CHASE is the largest dataset for XDTS to date, consisting of the most question sequences, SQL queries, and databases. CHASE also has rich semantic annotations, including contextual dependency and schema linking, which can inspire innovations to address challenges in XDTS. Table 16 in Appendix provides a list of question sequences in CHASE.

Dataset	Split	# DB	# Seq.	# Pair
CHASE	Train	200	3,949	12,914
	Dev	40	755	2,494
	Test	40	755	2,532
CHASE-C	Train	80	1,377	5,141
	Dev	20	333	1,291
	Test	20	333	1,262
CHASE-T	Train	140	3,034	9,043
	Dev	20	422	1,203
	Test	-	-	-

Table 6: Dataset split statistics.

Data Split According to the cross-database setting of XDTS, we split CHASE such that a database appears in only one of the train, development, and test set. To understand the characteristics of the data collected in CHASE-C and CHASE-T, we also split them accordingly. Since CHASE-T is constructed from SParC, we follow the train and development split of the original SParC dataset. Table 6 shows the data split statistics.

4.2 Contextual Dependency

Table 1 presents the contextual dependency characteristic of CHASE. The numbers are computed on the development set in consistency with our study setup in Section 2.3. The number of context-dependent questions in CHASE (65%) is substantially larger than existing datasets. Also, CHASE has more questions that require resolutions of coref-

[4] We have emailed the authors of SParC to apply our patch to fix the incorrect SQL queries.

Dataset	Exact String Match	Fuzzy String Match	Semantic Match
CHASE	48.2%	40.2%	11.6%
CHASE-C	41.2%	44.8%	14.0%
CHASE-T	53.7%	37.0%	9.9%

Table 7: Mention of database schema items.

Match	Schema Item	Question
Fuzzy String	歌名 song_name	这首歌曲的名字是? What is the name of this song?
Fuzzy String	售价 selling_price	均价是多少? What is the average price?
Semantic	售价 selling_price	哪些音箱比这便宜? Which speakers are cheaper than this?
Semantic	成立时间 founding_date	哪只球队历史最悠久? Which team has the longest history?

Table 8: Examples of fuzzy string match and semantic match. Each item's mention is highlighted.

erence and ellipsis. From this point of view, CHASE is a better testbed for XDTS. When it comes to CHASE-C and CHASE-T, 71% of questions in CHASE-C are context-dependent, showing that question sequences collected with our method have richer contextual dependencies than those collected via decomposition. Compared with SParC, the number of context-dependent questions in CHASE-T increases from 53% to 58% through our effort.

4.3 SQL Hardness Distribution

Table 3 shows the SQL hardness distribution of CHASE. SQL queries in different hardness levels are more evenly distributed in CHASE, and only 28% of them are easy. By comparing CHASE-C with existing datasets, we can observe a remarkable difference between their hardness distributions. Specifically, the number of easy queries (19%) in CHASE-C is less than that of hard (24%) and extra hard (20%) queries, indicating that question sequences created from scratch with our method are much more challenging. In terms of CHASE-T, the number of easy queries decreases from 40% to 37% through our effort, compared with SParC.

4.4 Mention of Database Schema Items

To understand how database schema items (tables and columns) are mentioned in questions, for each item annotated in the schema linking, we examine whether or not it can exactly match its mention in the question (Suhr et al., 2020). As shown in Table 7, among the 26,464 items annotated in the schema linking of CHASE, 48% of them are exactly mentioned in questions (*Exact String Match*), and 40% of them have at least one token that appears in their mentions (*Fuzzy String Match*). The remaining 12% items cannot be matched with their mentions via any string-match based methods (*Semantic Match*). Table 8 presents four typical examples for fuzzy string match and semantic match.

Compared with CHASE-T, whose data are constructed from SParC, CHASE-C has more items in the fuzzy string match and semantic match groups, implying that CHASE-C is more challenging and its mentions of schema items are more diverse.

5 Experiments

To understand the performance of the SOTA approaches on CHASE, CHASE-C, and CHASE-T, we experiment with the three approaches introduced in Section 2.2. Appendix A.3 provides the details of our adaptations for Chinese inputs and the experimental setup.

5.1 Experimental Results

Table 9 presents the experimental results, from which we make four main observations.

First, the performance of the SOTA approaches on CHASE is far from satisfactory. The best approach on CHASE, IGSQL, only achieves 40.4% Question Match (QM), which is significantly lower than the SOTA QM on SParC (60.1%) and CoSQL (50.8%). In terms of Interaction Match (IM), the best approach on CHASE only achieves 15.6%, lagging behind the SOTA IM on SParC (38.1%) and CoSQL (20.1%) by a large margin.[5] These results show that CHASE presents significant challenges for future research on XDTS.

Second, the performance of the SOTA approaches on CHASE-C is lower than that on CHASE-T. Specifically, IGSQL can achieve 43.3% QM and 26.3% IM on CHASE-T, but only 32.6% QM and 9.3% IM on CHASE-C. It shows that question sequences created from scratch with our method is much more challenging, which is consistent with our analysis in Section 4.

Third, the performance of the SOTA approaches on CHASE-T is lower than that on SParC. There are two reasons for the degradation. First, during the construction of CHASE-T, we try our best to mitigate the two biases found in Section 2,

[5] CoSQL has more questions in a question sequence (5.2) than SParC (3.0) and CHASE (3.3) on average.

Approach	CHASE				CHASE-C				CHASE-T		SParC		CoSQL	
	Dev		Test		Dev		Test		Dev		Dev		Dev	
	QM	IM	QM	IM	QM	IM	QM	IM	QM	IM	QM	IM	QM	IM
EditSQL	37.7	17.4	37.8	14.7	**33.6**	8.4	**32.6**	8.7	41.6	21.6	47.1	29.4	39.9	12.3
IGSQL	**41.4**	**20.0**	**40.4**	**15.6**	31.4	**10.8**	**32.6**	**9.3**	43.3	**26.3**	49.5	30.1	42.6	14.7
RAT-CON	35.1	14.6	32.5	9.8	24.6	5.4	23.9	4.5	**43.7**	21.6	**60.1**	**38.6**	**50.8**	**20.1**

Table 9: Question Match (QM) and Interaction Match (IM) of the three benchmark approaches.

Dataset	Contextual Dependency				SQL Hardness				Question Position				
	Indep.	Dep.	Coref.	Ellipsis	Easy	Medium	Hard	Extra Hard	1	2	3	4	>=5
CHASE	56.3	33.3	33.1	33.2	65.6	41.1	27.3	16.6	59.1	42.3	29.4	24.5	20.5
CHASE-C	45.4	25.7	25.8	25.2	52.3	36.6	23.5	11.4	48.6	34.2	22.5	19.0	17.1
CHASE-T	56.2	33.9	35.1	31.1	66.2	36.9	24.0	12.5	58.8	40.3	32.6	17.0	0.0

Table 10: Question Match of IGSQL on the development sets. 'Coref.' is short for 'Coreference'.

q_1	哪所大学培养了最多MVP球员？ (Which university has the most MVP players?)
y_1	select t2.college from MVP_Record as t1 join player as t2 group by t2.college order by count(distinct t2.player_id) desc limit 1
$\hat{y}_1$	select college from player group by college order by count(*) desc limit 1
q_2	状元呢？ (How about the first overall pick?)
y_2	select college from player where is_first_pick = "yes" group by college order by count(*) desc limit 1
$\hat{y}_2$	select is_first_pick from player group by college order by count(*) desc limit 1
q_3	居然还是肯塔基！杜克也非常出名啊，它培养了多少呢？ (Still Kentucky! Duke is also very famous! How many does it have?)
y_3	select count(*) from player where is_first_pick = "yes" and college like "%duke%"
$\hat{y}_3$	select count(*) from player where college like "%duke%"

Table 11: Predictions $\hat{y}_j$ of IGSQL for the question sequence in Figure 1. SQL queries are translated to English.

which makes CHASE-T more pragmatic and challenging than SParC. Second, existing approaches for XDTS are tuned for English only, and some components of these approaches cannot process Chinese inputs as well as English inputs.

Finally, although RAT-CON achieves the SOTA performance on SParC and CoSQL, it lags behind EditSQL and IGSQL by a large margin on CHASE and CHASE-C. Through a careful examination, we find that RAT-SQL (Wang et al., 2020a), the model that RAT-CON builds upon, adopts a string-match based method to find the linking between database schema items and their mentions in questions. However, this string-match based method struggles when many schema items are not exactly mentioned in questions. Also, this method struggles in Chinese probably because it is only tuned for English. The annotations of schema linking in CHASE can provide a great opportunity for future research to tackle this problem.

5.2 Fine-Grained Analysis

Table 10 shows the QM of IGSQL on the development set of CHASE, stratified by contextual dependency, SQL hardness, and question position.[6] We can observe a remarkable discrepancy between QM on context-independent and context-dependent questions. To tackle this problem, more advanced context modeling methods are needed. Our annotations of contextual dependency in CHASE can enable a fine-grained analysis on XDTS approaches, and they potentially can be used to address this problem. Besides, we observe that the QM of IGSQL on medium, hard, and extra hard queries of CHASE is higher than that of CHASE-C and CHASE-T, implying that more training samples for these complex queries can improve an approach's performance on them. A similar observation can be obtained in the question position. The QM of IGSQL on questions in turn 4 and >=5 is higher than that of CHASE-C and CHASE-T.

5.3 Case Study

Table 11 shows the predictions of IGSQL for the question sequence shown in Figure 1. q_1 queries the players that have won MVP, but IGSQL misses the "MVP_Record" table, probably because the

[6] Table 13 and 14 in Appendix present the detailed experimental results of EditSQL and RAT-CON.

FROM clause of SQL is synthesized based on the other predicted clauses. q_2 requires a resolution of ellipsis. It queries the college with the most first pick players, but IGSQL fails to resolve the ellipsis and predicts the wrong column in the SELECT clause. The last question omits the object "first pick players" of the verb "have", but the approach cannot fully resolve it and misses the first pick constraint in the WHERE clause.

6 Related Work

Dataset XDTS is a sub-task of *context-dependent semantic parsing* (CDSP) (Suhr et al., 2018; Guo et al., 2019a; Li et al., 2020). Many datasets have been constructed for CDSP. They can be categorized into two groups according to their annotations.

(1) Denotation Utterances in this group of datasets are only labelled with their denotations, i.e., the execution results of logical forms. SEQUENTIALQA (Iyyer et al., 2017), SCONE (Long et al., 2016), and CSQA (Saha et al., 2018) are representative datasets in this group. SEQUENTIALQA was constructed by decomposing some complicated questions from WikiTableQuestions (Pasupat and Liang, 2015) into sequences of simple questions. A question sequence in SCONE was collected by randomly generating a sequence of world states and asking annotators to write an utterance between each pair of successive states. CSQA was constructed by collecting a large number of individual questions and converting them into question sequences via a set of manually crafted templates.

(2) Logical Form Utterances in this group are labelled with their logical forms. Except for SParC and CoSQL, ATIS (Hemphill et al., 1990; Dahl et al., 1994) and TEMPSTRUCTURE (Chen and Bunescu, 2019) also fall into this group. ATIS was constructed under the Wizard-of-Oz (WOZ) setup. An annotator raised a question, and another annotator wrote the corresponding SQL query. Unlike datasets for XDTS, ATIS only focuses on the flight planning domain, which limits the possible SQL logic it contains. TEMPSTRUCTURE was also constructed under the WOZ setup, but it synthesized many artificial question sequences with templates to enlarge the dataset.

CHASE belongs to the group of logical form. To the best of our knowledge, it is the largest dataset with logical forms annotated for CDSP. Also, CHASE is the first Chinese dataset for CDSP.

Approach A lot of approaches have been proposed to address XDTS (Zhang et al., 2019; Cai and Wan, 2020; Zhong et al., 2020; Hui et al., 2021; Yu et al., 2021). Zhang et al. (2019) proposed EditSQL, which generates a SQL query by editing the query generated for previous turns. EditSQL also uses an interaction-level encoder (Suhr et al., 2018) to model the interactions between the current question and previous questions. IGSQL (Cai and Wan, 2020) improves over EditSQL by introducing a graph encoder to model database schema items together with historically mentioned items. Hui et al. (2021) jointly modeled a question sequence, schema items, and their interactions via a dynamic graph and a graph encoder. They also proposed a re-ranking module to improve the generation accuracy. Liu et al. (2020) systematically compared different context modeling methods on SParC and CoSQL. They found that concatenating all questions as inputs rivals or even outperforms more complicated context modeling methods. This finding also motivates us to implement the strong benchmark approach, RAT-CON.

7 Conclusion and Future Work

This work presents CHASE, to date the largest dataset for XDTS, consisting of 5,459 question sequences over 280 databases. Each question in CHASE has rich semantic annotations, including its SQL query, contextual dependency, and schema linking. Experimental results show that CHASE highlights the challenging problems of XDTS and there is a long way for us to achieve real Text-to-SQL demands of users. Currently, CHASE is constructed for Chinese. We plan to support more languages in the future. Besides, we plan to explore the ways to utilize the rich semantic annotations in CHASE to address the challenges in XDTS.

Acknowledgments

We thank Wuxia Jin and the anonymous reviewers for their helpful discussion and detailed comments. We thank Weixu Zhang, Jiawei Lin, Xiaotong Zheng, Nan Hu, Tingting Zhang, Zekun Qi, Chengzu Li, Junjie Tao, Jinghan He, and Yu Ma for participating in the construction of CHASE. Ming Fan was partially supported by NSFC (61902306), China Postdoctoral Science Foundation (2019TQ0251, 2020M673439), Youth Talent Support Plan of Xi'an Association for Science and Technology (095920201303).

Ethical Considerations

This work presents CHASE, a free and open dataset for the research community to study the cross-database context-dependent Text-to-SQL problem (XDTS). Data in CHASE are collected from two sources. First, we collect 120 databases from the DuSQL (Wang et al., 2020c) dataset, a free and open dataset for the Chinese Text-to-SQL problem. To collect question sequences on these 120 databases, we recruit 12 Chinese college students (5 females and 7 males). Each student is paid 10 yuan ($1.6 USD) for creating each question sequence. This compensation is determined according to prior work on similar dataset construction (Yu et al., 2019a). Since all question sequences are collected against open-access databases, there is no privacy issue. Second, to enlarge our dataset, we translate all the data, including questions, SQL queries, and databases, from English to Chinese in SParC (Yu et al., 2019b). SParC is a free and open English dataset for XDTS. 11 college students (5 females and 6 males) are recruited to perform the translation, each of whom is paid 2 yuan ($0.3 USD) for translating each question. The details of our data collection and characteristics are introduced in Section 3 and 4.

References

I. Androutsopoulos, G.D. Ritchie, and P. Thanisch. 1995. Natural language interfaces to databases – an introduction. *Natural Language Engineering*, 1(1):29–81.

Núria Bertomeu, Hans Uszkoreit, Anette Frank, Hans-Ulrich Krieger, and Brigitte Jörg. 2006. Contextual phenomena and thematic relations in database QA dialogues: results from a Wizard-of-Oz experiment. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006*, pages 1–8, New York, NY, USA. Association for Computational Linguistics.

Yitao Cai and Xiaojun Wan. 2020. IGSQL: Database schema interaction graph based neural model for context-dependent text-to-SQL generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6903–6912, Online. Association for Computational Linguistics.

Charles Chen and Razvan Bunescu. 2019. Context dependent semantic parsing over temporally structured data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3576–3585, Minneapolis, Minnesota. Association for Computational Linguistics.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2019a. Coupling retrieval and meta-learning for context-dependent semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 855–866, Florence, Italy. Association for Computational Linguistics.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019b. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990.*

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021. Dynamic hybrid relation exploration network for cross-domain context-dependent semantic parsing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):13116–13124.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831, Vancouver, Canada. Association for Computational Linguistics.

J. F. Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Trans. Inf. Syst.*, 2(1):26–41.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, Online. Association for Computational Linguistics.

Zhuang Li, Lizhen Qu, and Gholamreza Haffari. 2020. Context dependent semantic parsing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2509–2521, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Jian-Guang Lou, Bin Zhou, and Dongmei Zhang. 2020. How far are we from effective context modeling? an exploratory study on semantic parsing in context. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3580–3586. International Joint Conferences on Artificial Intelligence Organization. Main track.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1456–1465, Berlin, Germany. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Qingkai Min, Yuefeng Shi, and Yue Zhang. 2019. A pilot study for Chinese SQL semantic parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3652–3658, Hong Kong, China. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 705–713. AAAI Press.

Torsten Scholak, Raymond Li, Dzmitry Bahdanau, Harm de Vries, and Chris Pal. 2021. DuoRAT: Towards simpler text-to-SQL models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1313–1321, Online. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Alane Suhr, Ming-Wei Chang, Peter Shaw, and Kenton Lee. 2020. Exploring unexplored generalization challenges for cross-database semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8372–8388, Online. Association for Computational Linguistics.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2238–2249, New Orleans, Louisiana. Association for Computational Linguistics.

Anh Tuan Nguyen, Mai Hoang Dao, and Dat Quoc Nguyen. 2020. A pilot study of text-to-SQL semantic parsing for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4079–4085, Online. Association for Computational Linguistics.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Huajie Wang, Mei Li, and Lei Chen. 2020b. PG-GSQL: Pointer-generator network with guide decoding for cross-domain context-dependent text-to-SQL generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 370–380, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020c.

DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935, Online. Association for Computational Linguistics.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021. {SC}ore: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019. Editing-based SQL query generation for cross-domain context-dependent questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5338–5349, Hong Kong, China. Association for Computational Linguistics.

Victor Zhong, Mike Lewis, Sida I. Wang, and Luke Zettlemoyer. 2020. Grounded adaptation for zero-shot executable semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6869–6882, Online. Association for Computational Linguistics.

A Appendix

A.1 RAT-Con

RAT-Con is our extension of RAT-SQL (Wang et al., 2020a), the SOTA approach for the context-independent Text-to-SQL problem. Given a question q and a database $\mathcal{DB}$, RAT-SQL first links the database schema items with their mentions in questions via a string-match based method. Then, the linking results are jointly encoded with q and $\mathcal{DB}$ using a relation-aware self-attention transformer (Shaw et al., 2018). To generate a SQL query y, RAT-SQL adopts a grammar-based decoder (Yin and Neubig, 2017).

To extend RAT-SQL to the context-dependent setting, we use the simple concatenation context modeling method, which has shown to be competitive with other more complex context modeling methods (Liu et al., 2020). Specifically, to generate SQL query y_j^i for q_j^i, we concatenate all its prior questions $\langle q_1^i, \cdots, q_{j-1}^i \rangle$ with a special symbol `[SEP]`: $\langle q_j^i, \texttt{[SEP]}, q_{j-1}^i, \cdots, \texttt{[SEP]}, q_1^i \rangle$. The other components of RAT-SQL remain the same. Figure 2 shows the architecture of RAT-Con with an illustrative example.

We implement RAT-Con on the codebase of DuoRAT (Scholak et al., 2021). We use the default hyper-parameters in DuoRAT except for the batch size, which is altered to 24.

A.2 Annotation Tool for Chase-C

Figure 3 shows the user interface of our annotation tool for collecting question sequences in Chase-C. When an annotator is going to raise a follow-up question, an intent category is randomly sampled from one of the five categories in Table 4. The chosen category is highlighted in the row “意图” of the left panel. The annotator is recommended to raise a question that meets one of the relations in the category. After raising the question, the annotator is asked to label the contextual dependency and the corresponding SQL query of the question. The SQL query should be executable in the SQLite database engine. The execution results are shown to the annotator. Besides, we extract all the tables, columns, and values in the query, and we ask the annotator to link them to their mentions in the question. The linked characters are highlighted in the row “Tokens” of the left panel.

A.3 Experimental Details

To study existing datasets for XDTS, we need to get the predictions on the development sets from the benchmark approaches. The predictions of EditSQL are released with its source code. Hence, we directly use them for analysis. As for IGSQL, we train it with the default hyper-parameters specified in its source code, but we cannot reproduce the numbers reported in its paper. Nevertheless, IGSQL still outperforms EditSQL in both SParC and CoSQL. In terms of RAT-Con, we train it from scratch. All our experiments were conducted on TITAN RTX with 24GB memory.

A.3.1 Adaptation to Chinese Inputs

Since all the benchmark approaches use BERT for encodings and Chase is constructed for Chinese, we replace BERT with Chinese-BERT.[7] During the adaptations of EditSQL and IGSQL, we identified and fixed 3 bugs in their pre-process and post-process procedures. The string-match based schema linking method in RAT-SQL utilizes the Stanford CoreNLP Toolkit (Manning et al., 2014) to tokenize a question, and the method performs string matches between the resulting words and schema items. To adapt this method to Chinese, we try to use the Chinese package of CoreNLP to tokenize questions. However, we find that doing so fails to link a lot of schema items. Consider the question “这首歌曲的名字是？” and the column “歌名” which is an abbreviation for “歌曲的名字”. The question is tokenized by CoreNLP into 〈 这, 首, 歌曲, 的, 名字, 是, ？ 〉. None of the resulting words can be matched with “歌名”. Consequently, the method cannot link the column to the question. To solve this problem, we simply tokenize a Chinese question character by character. In this way, the character ‘歌’ and ‘名’ can be partly matched to “歌名”. Although this solution would introduce a lot of noises, our experimental results show that this solution outperforms the one using CoreNLP. It would be very useful to explore the ways to conduct schema linking in Chinese.

[7] https://github.com/google-research/bert

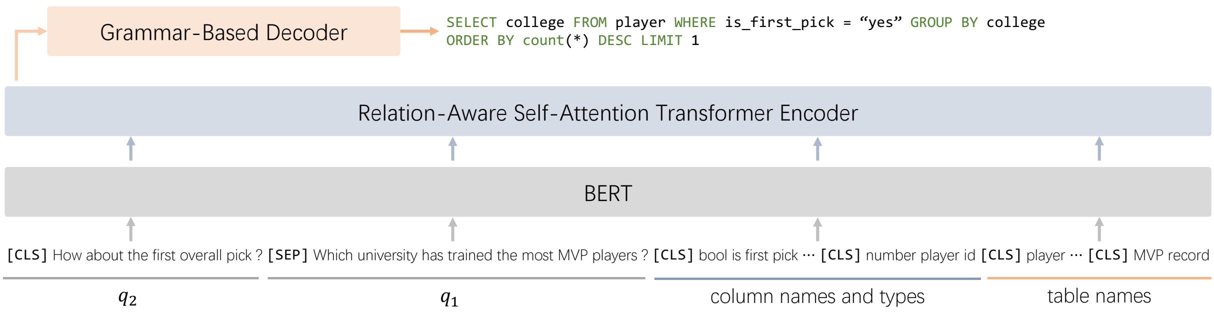
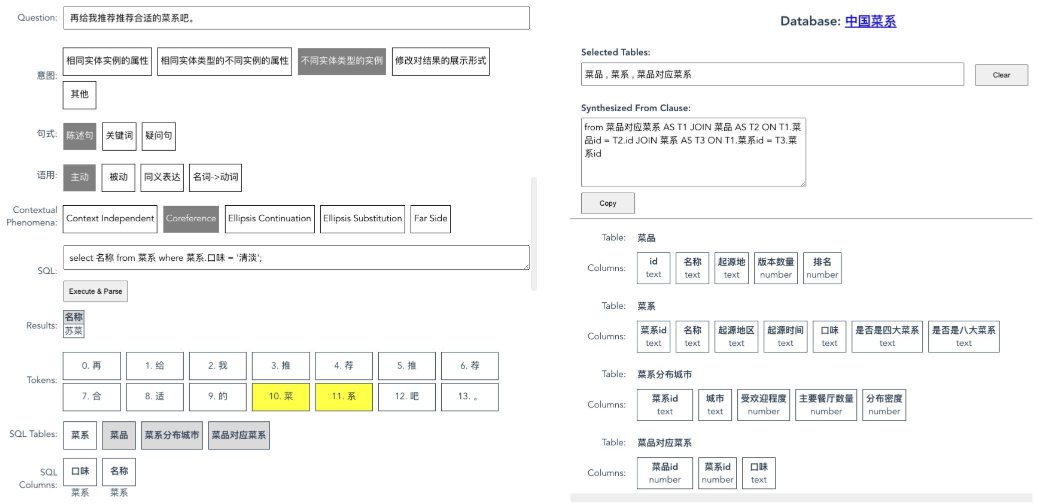

Grammar-Based Decoder
SELECT college FROM player WHERE is_first_pick = “yes” GROUP BY college
ORDER BY count(*) DESC LIMIT 1
Relation-Aware Self-Attention Transformer Encoder
BERT
[CLS] How about the first overall pick ? [SEP] Which university has trained the most MVP players ? [CLS] bool is first pick ⋯ [CLS] number player id [CLS] player ⋯ [CLS] MVP record
q_2
q_1
column names and types
table names

Figure 2: The architecture of RAT-CON.



Question: 再给我推荐推荐合适的菜系吧。

意图: 相同实体实例的属性 | 相同实体类型的不同实例的属性 | 不同实体类型的实例 | 修改对结果的展示形式 | 其他

句式: 陈述句 | 关键词 | 疑问句

语用: 主动 | 被动 | 同义表达 | 名词->动词

Contextual Phenomena: Context Independent | Coreference | Ellipsis Continuation | Ellipsis Substitution | Far Side

SQL: select 名称 from 菜系 where 菜系.口味 = '清淡';

Execute & Parse

Results:

名称
苏菜

Tokens: 0. 再 | 1. 给 | 2. 我 | 3. 推 | 4. 荐 | 5. 推 | 6. 荐 | 7. 合 | 8. 适 | 9. 的 | 10. 菜 | 11. 系 | 12. 吧 | 13. 。

SQL Tables: 菜系 | 菜品 | 菜系分布城市 | 菜品对应菜系

SQL Columns: 口味 (菜系) | 名称 (菜系)

Database: 中国菜系

Selected Tables: 菜品，菜系，菜品对应菜系

Clear

Synthesized From Clause:

from 菜品对应菜系 AS T1 JOIN 菜品 AS T2 ON T1.菜品id = T2.id JOIN 菜系 AS T3 ON T1.菜系id = T3.菜系id

Copy

Table: 菜品

Columns: id text | 名称 text | 起源地 text | 版本数量 number | 排名 number

Table: 菜系

Columns: 菜系id text | 名称 text | 起源地区 text | 起源时间 text | 口味 text | 是否是四大菜系 text | 是否是八大菜系 text

Table: 菜系分布城市

Columns: 菜系id text | 城市 text | 受欢迎程度 number | 主要餐厅数量 number | 分布密度 number

Table: 菜品对应菜系

Columns: 菜品id number | 菜系id number | 口味 text

Figure 3: The user interface of our annotation tool.

Category		Relation	Example	
			Precedent SQL Query y^i_{j-1}	Current SQL Query y^i_j
Same Instances	R1	Add Property	*select name from student;*	*select name, **age** from student;*
	R2	Remove Property	*select name, age from student;*	*select **name** from student;*
	R3	Replace Property	*select name from student;*	*select **country** from student;*
	R4	Add Group	*select count(*) from student;*	*select **country**, count(*) from student **group by country**;*
	R5	Add Aggregation	*select name from student;*	*select **count(*)** from student*
	R6	Alter Aggregation	*select max(age) from student;*	*select **avg**(age) from student;*
	R7	Delete Aggregation	*select count(*) from student;*	*select **name** from student;*
Different Instances of the Same Entity	R8	Subset	*select name from student;*	*select name from student **where country = “US”**;*
	R9	Superset	*select name from student where country = “US”;*	*select name from student where country = “US” **or country = “China”**;*
	R10	Disjoint	*select name from student where country = “US”;*	*select name from student where country = **“China”**;*
	R11	Complement	*select name from student where country = “US”;*	*select name from student where country **!=** “US”;*
	R12	Overlap	*select name from student join student_course where course_name = “Python”;*	*select name from student join student_course where course_name = **“C++”**;*
Different Entity	R13	Change Entity	*select name from student;*	*select **course_name** from **course**;*
Display	R14	Add Order	*select country, count(*) from student group by country;*	*select country, count(*) from student group by country **order by count(*)**;*
	R15	Alter Order	*select country, count(*) from student group by country order by count(*) asc;*	*select country, count(*) from student group by country order by count(*) **desc**;*
	R16	Distinct	*select country from student;*	*select **distinct** country from student;*

Table 12: All the 16 relations in the four basic intent categories presented in Section 3. Except for the relations in the *Different Entity* category, all the others can be combined.

Dataset	Approach	Contextual Dependency				SQL Hardness				Question Position				
		Indep.	Dep.	Coref.	Ellipsis	Easy	Medium	Hard	Extra Hard	1	2	3	4	>=5
CHASE	EditSQL	52.8	29.5	29.4	29.2	63.2	37.0	22.0	13.8	55.9	40.4	25.7	16.4	12.0
	IGSQL	**56.3**	**33.3**	**33.1**	**33.2**	**65.6**	**41.1**	**27.3**	**16.6**	**59.1**	**42.3**	**29.4**	**24.5**	**20.5**
	RAT-CON	49.3	27.3	27.1	28.1	60.9	35.4	16.4	11.6	51.1	37.6	24.9	16.4	7.2
CHASE-C	EditSQL	**51.1**	**26.6**	**27.1**	**25.4**	**58.5**	**36.8**	**24.4**	**15.4**	**55.3**	**37.2**	**22.8**	17.6	15.9
	IGSQL	45.4	25.7	25.8	25.2	52.3	36.6	23.5	11.4	48.6	34.2	22.5	**19.0**	**17.1**
	RAT-CON	35.8	20.0	19.8	20.2	46.5	29.5	13.3	8.3	38.7	29.1	18.3	11.4	7.3
CHASE-T	EditSQL	55.8	30.2	29.4	31.1	65.7	34.7	20.8	14.6	58.5	37.7	25.6	20.5	0.0
	IGSQL	56.2	33.9	**35.1**	31.1	66.2	36.9	**24.0**	12.5	58.8	**40.3**	32.6	17.0	0.0
	RAT-CON	**56.4**	**34.5**	33.2	**36.8**	**68.0**	**37.4**	18.2	**15.3**	**59.7**	38.4	**33.7**	**23.9**	0.0

Table 13: Fine-grained experimental results on the development set of CHASE, CHASE-C, and CHASE-T. 'Indep.' and 'Dep.' are short for 'context-independent' and 'context-dependent'. 'Coref.' indicates 'Coreference'.

Dataset	Approach	Contextual Dependency				SQL Hardness				Question Position				
		Indep.	Dep.	Coref.	Ellipsis	Easy	Medium	Hard	Extra Hard	1	2	3	4	>=5
CHASE	EditSQL	54.1	28.0	30.1	24.5	63.0	34.3	26.1	10.8	58.8	36.6	25.0	20.6	18.4
	IGSQL	**55.7**	**31.2**	**35.1**	**25.8**	**65.9**	**38.5**	**27.1**	10.2	**61.2**	**40.0**	**26.7**	**23.7**	**21.1**
	RAT-CON	45.7	24.5	25.0	23.8	56.0	29.1	18.4	**11.7**	51.4	31.8	20.2	18.6	10.5
CHASE-C	EditSQL	**49.3**	25.3	28.6	20.2	59.1	33.6	**25.6**	8.0	**52.6**	32.7	**22.8**	19.8	**18.3**
	IGSQL	46.7	**26.5**	**29.8**	**21.6**	**61.2**	**33.8**	22.9	**9.0**	50.2	**34.5**	22.5	**21.9**	**18.3**
	RAT-CON	34.7	19.2	21.5	15.6	48.3	24.9	14.3	5.2	38.1	26.7	17.4	11.5	8.5

Table 14: Fine-grained experimental results on the test set of CHASE and CHASE-C.

#	Question & SQL Query	Contextual Dependency
Question Sequence $\mathcal{Q}^1$		
q_1^1 y_1^1	How many templates are there? *select count(*) from templates*	Independent
q_2^1 y_2^1	What is the date effective of template 1? *select date_effective_from, date_effective_to from templates where template id = 1*	Independent
q_3^1 y_3^1	What is the template type code template id 4? *select template_type_code from templates where template_id = 4*	Independent
q_4^1 y_4^1	What is the version number of template id 0? *select version_number from templates where template_id = 0*	Independent
Question Sequence $\mathcal{Q}^2$		
q_1^2 y_1^2	What is the first name of player id 2000001? *select first_name from player where player_id = 2000001*	Independent
q_2^2 y_2^2	What is the birth date for Martina? There are a lot of Martina. Do you mean the Marina with id 200001? Martina with id 2000001 *select birth_date from player where player_id = 2000001*	Dependent (Other)
q_3^2 y_3^2	What is the country code for player id 2000003? *select country_code from player where player_id = 2000003*	Independent
Question Sequence $\mathcal{Q}^3$		
q_1^3 y_1^3	What unique cities are in Asian countries? select distinct t3.name from country as t1 join countrylanguage as t2 join city as t3 where t1.continent = "Asia"	Independent
q_2^3 y_2^3	Which of those cities have a population over 200,000? select distinct t3.name from country as t1 join countrylanguage as t2 join city as t3 where t1.continent = "Asia" and t3.population >200000	Dependent (Coreference)
q_3^3 y_3^3	What is the average population of all cities in China? select avg(t3.population) from country as t1 join countrylanguage as t2 join city as t3 where t1.name = "China"	Independent
q_4^3 y_4^3	What is the average population of all cities that speak the Dutch language? select avg(t3.population) from country as t1 join countrylanguage as t2 join city as t3 where t2.language = "Dutch"	Independent

Table 15: Question sequence examples in CoSQL. Since CoSQL is a task-oriented dialogue corpus, it has some questions involving *clarification*, e.g., the second question q_2^2 in Question Sequence $\mathcal{Q}^2$. We also consider these questions as context-dependent. Among the 1,007 questions in the development set of CoSQL, 95 of them involve clarifications. SParC and CHASE do not have this kind of questions.

#	Question & SQL Query	Contextual Dependency
Question Sequence $\mathcal{Q}^1$		
q_1^1 y_1^1	哪个专业可以本硕博连读？(Which major has a continuous academic program?) select 专业名称 from 专业 where 学科类型 = “本硕博” (select name from major where enrollment_mode = “continuous academic program”)	Independent
q_2^1 y_2^1	要读多少年？(How many years does one have to study?) select 学制 from 专业 where 学科类型 = “本硕博” (select duration from major where enrollment_mode = “continuous academic program”)	Dependent (Ellipsis)
q_3^1 y_3^1	该专业的招考类型 (enrollment type of this major) select t2.招考类型 from 专业 as t1 join 清华大学招生计划 as t2 on t1.id = t2.专业id where t1.学科类型 = “本硕博” (select t2.enrollment_type from major as t1 join enrollment_plan as t2 on t1.id = t2.major_id where t1.enrollment_mode = “continuous academic program”)	Dependent (Coreference)
q_4^1 y_4^1	除了这个专业，我还能报考哪些专业的专项计划？(Among the majors in special plan, which can I study except this major?) select t1.专业名称 from 专业 as t1 join 清华大学招生计划 as t2 on t1.id = t2.专业id where t2.招考类型 = “专项” except select 专业名称 from 专业 where 学科类型 = “本硕博” (select t1.name from major as t1 join enrollment_plan as t2 on t1.id = t2.major_id where t2.enrollment_type = “special plan” except select name from major where enrollment_type = “continuous academic program”)	Dependent (Coreference)
Question Sequence $\mathcal{Q}^2$		
q_1^2 y_1^2	我要寄一个快递到浙江。哪家公司的价格会最便宜？ (I’m going to send an express package to Zhejiang. Which company offers the lowest price?) select t2.公司名 from 快递费 as t1 join 快递公司 as t2 on t1.快递公司id = t2.公司id join 省份 as t3 on t1.区域 = t3.省id where t3.省名 = “浙江” order by t1.每公斤价格 asc limit 1 (select t2.name from express_cost as t1 join express_company as t2 on t1.express_company_id = t2.company_id join province as t3 on t1.region = t3.province_id where t3.province_name = “Zhejiang” order by t1.price_per_kg asc limit 1)	Independent
q_2^2 y_2^2	至少需要多少公斤才能寄？(How many kilograms at least?) select t1.起步公斤数 from 快递费 as t1 join 快递公司 as t2 on t1.快递公司id = t2.公司id join 省份 as t3 on t1.区域 = t3.省id where t3.省名 = “浙江” order by t1.每公斤价格 asc limit 1 (select t1.starting_kgs from express_cost as t1 join express_company as t2 on t1.express_company_id = t2.company_id join province as t3 on t1.region = t3.province_id where t3.province_name = “Zhejiang” order by t1.price_per_kg asc limit 1)	Dependent (Ellipsis)
q_3^2 y_3^2	有起步价格吗？如果有的话请告诉我。(Please tell me the starting price, if any.) select t1.起步价格 from 快递费 as t1 join 快递公司 as t2 on t1.快递公司id = t2.公司id join 省份 as t3 on t1.区域 = t3.省id where t3.省名 = “浙江” order by t1.每公斤价格 asc limit 1 (select t1.starting_price from express_cost as t1 join express_company as t2 on t1.express_company_id = t2.company_id join province as t3 on t1.region = t3.province_id where t3.province_name = “Zhejiang” order by t1.price_per_kg asc limit 1)	Dependent (Ellipsis)
q_4^2 y_4^2	那我需要认真考虑了。它有多少网点？(I’d like to think it over, how many branches does it have?) select t2.网点数量 from 快递费 as t1 join 快递公司 as t2 on t1.快递公司id = t2.公司id join 省份 as t3 on t1.区域 = t3.省id where t3.省名 = “浙江” order by t1.每公斤价格 asc limit 1 (select t2.branch_number from express_cost as t1 join express_company as t2 on t1.express_company_id = t2.company_id join province as t3 on t1.region = t3.province_id where t3.province_name = “Zhejiang” order by t1.price_per_kg asc limit 1)	Dependent (Coreference)
Question Sequence $\mathcal{Q}^3$		
q_1^3 y_1^3	哪些水果适合在秋季种植？(What fruit is suitable for autumn planting?) select 名称 from 水果 where 适合季节 = “秋季” (select name from fruit where suitable_season = “autumn”)	Independent
q_2^3 y_2^3	有哪些省份种植这些水果？(Which province is the fruit planted in?) select t3.名称 from 种植水果 as t1 join 水果 as t2 on t1.水果id = t2.id join 省份 as t3 on t1.省份id = t3.id where t2.适合季节 = “秋季” (select t3.name from fruit_planting as t1 join fruit as t2 on t1.fruit_id = t2.id join province as t3 on t1.province_id = t3.id where t2.suitable_season = “autumn”)	Dependent (Coreference)
q_3^3 y_3^3	去除重复的 (Remove the duplicated!) select distinct t3.名称 from 种植水果 as t1 join 水果 as t2 on t1.水果id = t2.id join 省份 as t3 on t1.省份id = t3.id where t2.适合季节 = “秋季” (select distinct t3.name from fruit_planting as t1 join fruit as t2 on t1.fruit_id = t2.id join province as t3 on t1.province_id = t3.id where t2.suitable_season = “autumn”)	Dependent (Ellipsis)
q_4^3 y_4^3	这些地方分别种植多少种水果？(How many kinds of fruit are planted in these provinces respectively?) select t2.名称, count(*) from 种植水果 as t1 join 省份 as t2 on t1.省份id = t2.id where t2.名称 in (select distinct t5.名称 from 种植水果 as t3 join 水果 as t4 on t3.水果id = t4.id join 省份 as t5 on t3.省份id = t5.id where t4.适合季节 = “秋季”) group by t2.名称 (select t2.name, count(*) from fruit_planting as t1 join province as t2 on t1.province_id = t2.id where t2.name in (select distinct t5.name from fruit_planting as t3 join fruit as t4 on t3.fruit_id = t4.id join province as t5 on t3.province_id = t5.id where t4.suitable_season = “autumn”) group by t2.name)	Dependent (Coreference)
Question Sequence $\mathcal{Q}^4$		
q_1^4 y_1^4	微软赞助了哪些大会？(What AI summits did Microsoft sponsor?) select t2.名称 from 峰会赞助公司 as t1 join 峰会 as t2 on t1.峰会id = t2.峰会id where t1.公司 = “微软集团” (select t2.name from sponsor_company as t1 join summit as t2 on t1.summit_id = t2.summit_id where t1.company = “Microsoft”)	Independent
q_2^4 y_2^4	该会是谁主办的？(What’s the organizer of this summit?) select t2.主办单位 from 峰会赞助公司 as t1 join 峰会 as t2 on t1.峰会id = t2.峰会id where t1.公司 = “微软集团” (select t2.organizer from sponsor_company as t1 join summit as t2 on t1.summit_id = t2.summit_id where t1.company = “Microsoft”)	Dependent (Coreference)
q_3^4 y_3^4	有多少嘉宾参与？(How many honoured guests attended?) select count(*) from 嘉宾参与峰会 as t1 join 嘉宾 as t2 on t1.嘉宾id = t2.嘉宾编号 join 峰会 as t3 on t1.峰会id = t3.峰会id join 峰会赞助公司 as t4 on t3.峰会id = t4.峰会id where t4.公司 = “微软集团” (select count(*) from guests_of_summits as t1 join guests as t2 on t1.guest_id = t2.guest_id join summit as t3 on t1.summit_id = t3.summit_id join sponsor_company as t4 on t3.summit_id = t4.summit_id where t4.company = “Microsoft”)	Dependent (Ellipsis)

Table 16: Question sequence examples in CHASE.