A Datasheet for Datasets

We follow instructions provided by `Datasheet for Datasets` to answer the important questions considering this dataset.

A.1 Motivation

For what purpose was the dataset created? The advancement of Large Language Models (LLMs) has raised concerns regarding whether state-of-the-art LLMs, such as ChatGPT and Codex, can replace human effort in real-world text-to-SQL tasks involving large database values. That is because their exceptional performance on previous academic tasks like SPIDER impresses researchers. However, we observe that current cross-domain text-to-SQL benchmarks only focus on the database schema, which lack of full attention on values, resulting in a gap between academic and real-world applications. To address this issue, we introduce BIRD, the largest cross-domain text-to-SQL benchmark highlighting extensive and realistic databases for community development. Additionally, we hope to observe the performance gap between LLMs and humans. Our experimental results indicate that, as of now, LLMs are still unable to replace human effort. As far as we know, BIRD is the first text-to-SQL benchmark to collect human performance.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)? Please refer to the author list for details. Our research team involves Star Lab at The University of Hong Kong, Alibaba DAMO Academy Conversational AI (ConAI) Team, the Department of Computer Science at the University of Illinois Urbana-Champaign, the Department of EECS at Massachusetts Institute of Technology, the School of Data Science at The Chinese University of Hong Kong (Shenzhen), and Database Group of Tsinghua University.

Who funded the creation of the dataset? This dataset is fully funded by the Alibaba DAMO Academy ConAI team. We spent 97,654 USD for presenting this data. The budget includes 10% for recruiting competent research interns, 80% for developing the benchmark, and 10% for refining and implementing the benchmark.

A.2 Composition

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? BIRD contains natural language questions, external knowledge evidence sentences, processed large databases, database description files (csv), and SQL queries.

How many instances are there in total (of each type, if appropriate)? BIRD contains 12,751 natural language questions, 12,751 external knowledge evidence sentences, 95 processed large databases, 95 folders of database description CSV files, and 12,751 ground truth SQL queries.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? In BIRD, we divide it into three sets: training, development, and testing. Training and development sets are public while testing data set is hidden for the fair evaluation of all text-to-SQL challengers. This could witness the real development of text-to-SQLs in the LLM era.

Is there a label or target associated with each instance? In BIRD, we provide two labels for each question instance: SQLs (the target of input) and external knowledge evidence (expert annotated evidence for each expected SQLs).

Is any information missing from individual instances? No.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? No.

Are there recommended data splits? Our data consists of 9,428 instances for the training set, 1,534 instances for the development set, and 1,789 instances for the concealed test set. The training and development sets are derived from public databases, while the test set databases are curated and designed by our specialized team. We do this because some researchers express concerns that the remarkable performance of LLMs in text-to-SQL tasks may not be attributed to an improvement in capabilities, but rather to the exposure of data and database values to the LLMs during the pre-training phase. To address these concerns, we opt to self-design new databases in testing using actual tabular data, thereby ensuring that LLMs do not preview the databases.

Are there any errors, sources of noise, or redundancies in the dataset? As stated in the main content, our double-blind annotation procedure is both expensive and rigorous, ensuring data quality. However, it is virtually impossible for any dataset, especially complex ones, to be entirely free of errors. Our team is committed to enhancing the data even after this paper is accepted, thereby contributing to the text-to-SQL community. In addition, we encourage users to provide feedback and report errors on our data website, allowing us to rectify and enhance the dataset.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? Yes, all databases in training and development are collected under appropriate licenses. Please see Section 3.2 for more details

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals' non-public communications)? No.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? No.

Does the dataset identify any subpopulations (e.g., by age, gender)? Some questions mention ages and genders, but they are just used to detect the capability of models on text-to-SQLs. No bias or other opinions are involved.

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? No. All databases are collected from open-sourced platforms, and any sensitive data has already been processed before.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals race or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? No, this is a QA-based text-to-SQL dataset, we don't require models to deliver any opinions on results. And also we don't present any bias or opinions in the dataset.

A.3 Collection Process

How was the data associated with each instance acquired? Section 3 and Appendix B.2 introduce this in detail.

What mechanisms or procedures were used to collect the data (e.g., hardware apparatuses or sensors, manual human curation, software programs, software APIs)? Section 3 and Appendix B.2 introduce this in detail. Our crowdworkers use Alibaba internal labeling software to annotate the data and examine the results.

If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)? No.

Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)? Four PhD students and two MS students are involved in the creation of database description files. Two independent teams of crowdworkers are recruited to annotate questions, and SQLs. The question annotators are composed of 11 English native speakers and SQL annotators are comprised of database engineers and DB students. The total consumption is 97,654 USD.

Over what timeframe was the data collected? From Sep. 2022 to Mar. 2023.

Were any ethical review processes conducted (e.g., by an institutional review board)? Yes, we take such issues very seriously. During the review process, we found that certain questions related to politics or inappropriate language. We have addressed these concerns by modifying the content and providing a serious warning to the annotators responsible for such instances.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)? Section 3 and Appendix B.2 introduce this in detail.

Were the individuals in question notified about the data collection? Yes.

Did the individuals in question consent to the collection and use of their data? Sure, we recruited them and paid the satisfying salaries.

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? No.

Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis) been conducted? Yes, we did a very comprehensive analysis including error analysis, and efficiency analysis, in the experiments of the paper and Appendix.

A.4 Preprocessing/cleaning/labeling

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? Yes, we provide token list for each question and SQLs from NLTK for users.

Was the “raw” data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? No. Is the software that was used to preprocess/clean/label the data available? Yes, https://www.nltk.org/

A.5 Uses

Has the dataset been used for any tasks already? No.

Is there a repository that links to any or all papers or systems that use the dataset? No.

What (other) tasks could the dataset be used for? Sure, our databases and analysis-style questions are most valuable, so they could be beneficial to DB-based code generation, data science analysis, etc.

Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses? No.

Are there tasks for which the dataset should not be used? No.

A.6 Distribution

Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created? No.

How will the dataset will be distributed (e.g., tarball on the website, API, GitHub)? All source codings, and datasets could be found on our leaderboard website: https://bird-bench.github.io/. And we provide fast download links for the convenience of researchers who want to use our big data. Furthermore, the code repository can be found in https://github.com/AlibabaResearch/DAMO-ConvAI/tree/main/bird

When will the dataset be distributed? Now.

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)? Given the database size of BIRD is the largest until now, we are afraid that abusing ample database values may lead to inappropriate commercial use. Therefore, we claim that this dataset should be distributed under CC BY-NC 4.0.

Have any third parties imposed IP-based or other restrictions on the data associated with the instances? No.

Do any export controls or other regulatory restrictions apply to the dataset or to individual instances? No.

A.7 Maintenance

Who will be supporting/hosting/maintaining the dataset? HKU STAR LAB and Alibaba DAMO Academy

How can the owner/curator/manager of the dataset be contacted (e.g., email address)? Contact bird.bench23@gmail.com or the corresponding authors or co-first authors in the author list.

Is there an erratum? No.

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? Yes, we will keep polishing and optimizing our data periodically.

If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., was the individuals in question were told that their data would be retained for a fixed period of time and then deleted)? No.

Will older versions of the dataset continue to be supported/hosted/maintained? No. The most updated version will be more reliable.

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? Yes, but they should contact the authors first.

B Appendix

B.1 Text-to-SQL Difficulty

In order to help researchers deeply analyze model performance in various text-to-SQL case levels, we class all examples as `simple` (30%), `moderate` (60%), and `challenging` (10%). Previous work, such as SPIDER, computed difficulty mainly based on SQL complexity. However, we find that additional factors, such as question comprehension, schema linking, and external knowledge reasoning, also influence model and human performance. Therefore, each SQL annotator is required to evaluate examples based on these factors, and experts conclude the ratings to divide examples into the three aforementioned difficulty levels. This approach offers a more extensive difficulty analysis for text-to-SQL tasks. And the performance of ChatGPT on three different difficulty levels is shown in Table B.1.

MODEL	DEV SET				TEST SET			
	simple	moderate	challenging	total	simple	moderate	challenging	total
(EX) ChatGPT	31.08	13.29	12.08	24.05	35.41	19.46	12.28	26.77
(EX) ChatGPT + KG	**45.44**	**26.14**	**19.01**	**37.22**	**49.21**	**31.89**	**20.70**	**39.30**
(VES) ChatGPT	36.20	15.43	14.42	27.97	50.09	24.71	15.39	36.68
(VES) ChatGPT + KG	**54.71**	**28.16**	**22.80**	**43.81**	**65.06**	**41.21**	**25.81**	**51.40**

Table 4: The Execution Accuracy (EX) and Valid Efficiency Score (VES) are presented for both the ChatGPT model and its version with grounding (KG) for external knowledge evidence, taking into consideration development and testing datasets.

B.2 Annotation Entrance

Question Annotation Entrance. We hire a group of native speakers of English with degrees above the bachelor's level and database-related knowledge to ask a variety of natural language questions regarding the values of databases. To fulfill this objective, we have adopted the following procedure: (1). ER diagrams and database description files are documented to assist the annotators in understanding the databases; (2). we present the annotators with three databases from different domains and require them to generate 10 questions for each database; (3). these questions are then assessed by 3 text-to-SQL experts applying predefined rules. Those questions earning at least two votes are marked as valid. Only annotators capable of generating more than 8 valid questions per database are preserved. As a result, 11 native speakers contribute questions to BIRD.

SQL Annotation Entrance. With the purpose of enhancing the quality of our SQL queries, we assemble a team of skilled data engineers and database students. The team undergoes rigorous testing through the text-to-SQL evaluation process, which assesses their capability of generating SQL queries for a variety of questions facing different domains of databases. Each annotator is asked to answer 10 questions, and only those who score at least 9 out of 10 will be qualified to annotate SQL queries for BIRD.

B.3 Question Distribution

Figure 6 contains the detailed question types and their examples.

B.4 Experiment Details

FT-based Models. T5 is a strong and versatile pre-trained language model (PLM) for the text-to-text generation that has achieved state-of-the-art performance in a variety of semantic parsing tasks, including text-to-SQL. We concatenate the question with serialized database schema as input [38, 48, 39]. And SQL can be fetched in an end-to-end fashion by easily fine-tuning. While seq2AST-based methods [41, 3] are also effective in text-to-SQL, actually their grammar rules utilized during decoding are constrained on specific datasets [24]. We implement our codes mainly based on the

Question Type	Sub Type	Question / SQL	Percentage
Fundamental Type	Match-based	How many gas stations in CZE has Premium gas? `SELECT COUNT(GasStationID) FROM gasstations WHERE Country = 'CZE' AND Segment = 'Premium'`	83.9 %
	Ranking	What are the titles of the top 5 posts with the highest popularity? `SELECT Title FROM posts ORDER BY ViewCount DESC LIMIT 5`	20.3 %
	Comparison	How many color cards with no borders have been ranked higher than 12000 on EDHRec? `SELECT COUNT(id) FROM cards WHERE edhrecRank > 12000 AND borderColor = 'borderless'`	16.7 %
	Counting	How many of the members' hometowns are from Maryland state? `SELECT COUNT(T2.member_id) FROM zip_code AS T1 INNER JOIN member AS T2 ON T1.zip_code = T2.zip WHERE T1.state = 'Maryland'`	30.4 %
	Aggregation	What is the average height of the superheroes from Marvel Comics? `SELECT AVG(T1.height_cm) FROM superhero AS T1 INNER JOIN publisher AS T2 ON T1.publisher_id = T2.id WHERE T2.publisher_name = 'Marvel Comics'`	15.7 %
Reasoning Type	Domain Knowledge	Name the ID and age of patient with two or more laboratory examinations which show their hematoclit level exceeded the normal range. `SELECT T1.ID, STRFTIME('%Y', CURRENT_TIMESTAMP) - STRFTIME('%Y', T1.Birthday) FROM Patient AS T1 INNER JOIN Laboratory AS T2 ON T1.ID = T2.ID WHERE T1.ID IN ( SELECT ID FROM Laboratory WHERE HCT > 52 GROUP BY ID HAVING COUNT(ID) >= 2 )`	23.6 %
	Numeric Computation	Among the posts with a score of over 20, what is the percentage of them being owned by an elder user? `SELECT CAST(SUM(IIF(T2.Age > 65, 1, 0)) AS REAL) * 100 / count(T1.Id) FROM posts AS T1 INNER JOIN users AS T2 ON T1.OwnerUserId = T2.Id WHERE T1.Score > 20`	24.5 %
	Synonym	How many clients opened their accounts in Jesenik branch were women ? (female) `SELECT COUNT(T1.client_id) FROM client AS T1 INNER JOIN district AS T2 ON T1.district_id = T2.district_id WHERE T1.gender = 'F' AND T2.A2 = 'Jesenik'`	7.2 %
	Value Illustration	Among the weekly issuance accounts, how many have a loan of under 200000? `SELECT COUNT(T1.account_id) FROM loan AS T1 INNER JOIN account AS T2 ON T1.account_id = T2.account_id WHERE T2.frequency = 'POPLATEK TYDNE' AND T1.amount < 200000`	70.1 %

Figure 6: Questions in the BIRD contain two main categories. The `Fundamental Type` of questions are comparable to other text-to-SQL benchmarks. The `Reasoning Type` of questions requires external knowledge grounding to answer.

hugging-face transformers library [**]. We set the max input length as 1024, the generation max length as 512, and the batch size as 32. We also adopt Adafactor as our primary optimizer with a linear decayed learning rate of 5e-5. All experiments are conducted on one NVIDIA Tesla A100 80GB, which is available for most research centers. We set the random seed as 1 for all runs of FT-based models since 1 is an optimal seed proven by previous SOTA models [26, 48].

[**] https://huggingface.co/

ICL-based Models. Codex (`code-davinci-002`[††‡‡]) and ChatGPT (`gpt-3.5-turbo`) are popular and powerful large-scale pre-trained language models (LLMs) for code generation driven by ICL. It can produce multiple types of codes, including SQL, from human instructions without additional training. We employ programming-based prompts, as described in [37], to collect results by calling the API. Also, we choose Azure OpenAI API to align the codes with other variants of LLMs. Given that models are not allowed to unseen databases and ground-truth SQLs in the evaluation set, a zero-shot generation strategy is the most appropriate. Moreover, to investigate the impact of multi-step reasoning of LLMs on BIRD, we implement the Chain-Of-Thought (COT) technique [47] by easily adding the prompt sentence `"Let's think step by step."` before the generation of SQLs [20]. However, we find out the output of ChatGPT is too uncertain with many unexpected explanations, thus we provide a 1-shot pseudo example for ChatGPT to learn the procedure of thinking and output format. The detailed prompt design is shown in Figure 7. In order to minimize the randomness of results, we set the temperature as 0 to ensure reproduction.

Knowledge Fusion. In the baseline implementation, we naively concatenate the knowledge evidence sentences with questions and database schemas, but we can observe a significant improvement by this easy method. A more complicated and effective strategy of knowledge grounding for ChatGPT and T5 would be an important future topic. The knowledge evidence sentences are concluded to the external knowledge provided by annotators as described in Section 3.3.

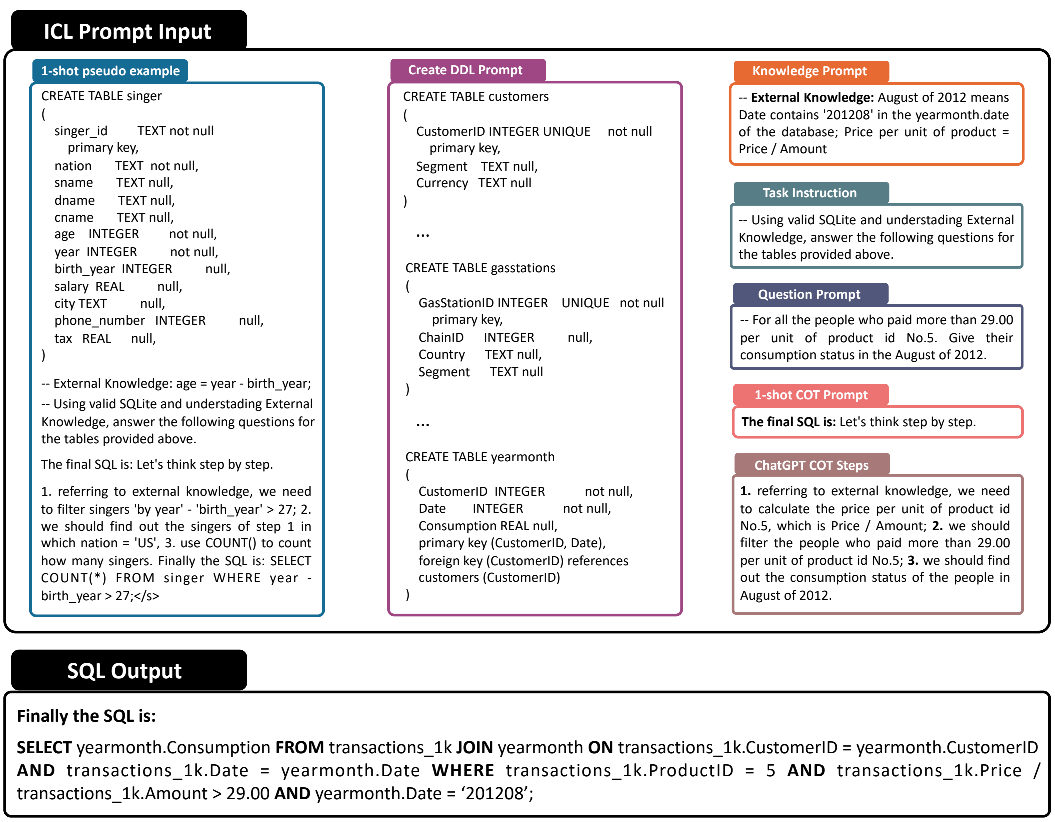

Figure 7: The detailed prompt design for implementation of ChatGPT + KG + COT.

[††] https://openai.com/blog/openai-api
[‡‡] It's still valid at the time of writing.

B.5 Efficiency Analysis Details

Two strategies for performing text-to-efficient-SQL are presented in Figure 8. Examples show that both two-stage optimization and embodied databases can help semantic parsings generate more efficient SQLs.

Query Rewriting

Ex1.1 Question:
List out the age of users who located in Vienna, Austria obtained the badge?

ChatGPT SQL:

```
SELECT Age FROM users WHERE Location = 'Vienna, Austria‘ AND Id IN (SELECT UserId FROM
badges)
```

Optimized SQL: (time-saving percentage: **99.92%**)

```
SELECT u.Age FROM users AS u INNER JOIN badges AS b ON u.Id = b.UserId WHERE u.Location
= 'Vienna, Austria’
```

Take Away:
By applying a **JOIN** operation instead of a subquery with **IN** can improve efficiency, as the database may execute the **JOIN** and filtering processes concurrently in just one operation **without** the need to store the intermediate results to filter primary query.

Ex1.2 Question:
How many of the members' hometowns are from Maryland state?

ChatGPT SQL:

```
SELECT COUNT(*) FROM member INNER JOIN zip_code ON member.zip = zip_code.zip_code WHERE
zip_code.state = 'Maryland’
```

Optimized SQL: (time-saving percentage: **67.93%**)

```
SELECT COUNT(member.member_id) FROM member INNER JOIN zip_code ON member.zip =
zip_code.zip_code WHERE zip_code.state = 'Maryland’
```

Take Away:
Utilizing the **COUNT** function on a **NOT-NULL** column, as opposed to **COUNT(*)**, can increase time efficiency. This rewritten SQL enables the database to count **NOT-NULL** values within a single column, rather than compute all rows including those with **NULL** values. Usually, the primary key column is selected as this **NOT-NULL** column.

Ex1.3 Question:
Who is the owner of the account with the largest loan amount?

ChatGPT SQL:

```
SELECT c.client_id FROM client c INNER JOIN disp d ON c.client_id = d.client_id INNER
JOIN loan l ON d.account_id = l.account_id ORDER BY l.amount DESC LIMIT 1
```

Optimized SQL: (time-saving percentage: **62.39%**)

```
SELECT c.client_id FROM client c INNER JOIN disp d ON c.client_id = d.client_id
INNER JOIN loan l ON d.account_id = l.account_id WHERE l.amount = ( SELECT MAX(amount)
FROM loan)
```

Take Away:
In an unindexed environment, employing the **MAX** function can potentially yield faster results since it avoids the need for **sorting**, which could run against a large table.

Adding Indexes to Database

Ex2.1 Question:
How many accounts are there in the district of \"Pisek\"?

ChatGPT SQL:

```
SELECT COUNT(*) FROM account a INNER JOIN district d ON a.district_id = d.district_id
WHERE d.A2 = ‘Pisek’
```

Added Indexes: (time-saving percentage: **87.27%**)

```
CREATE INDEX account_district_id_index ON account(district_id);
CREATE UNIQUE INDEX district_district_id_uindex ON district(district_id);
```

Take Away:
Adding **indexes** into a database can significantly increase the speed of SQL queries because it creates a data structure that enables the database engine to quickly locate rows that match specific criteria instead of **scanning** the entire table.

Figure 8: Two possible solutions and explanations to improve efficiency are presented. The first batch of examples shows how to optimize SQL efficiency by rewriting SQL based on rules. The last example is to show that adding indexes to databases can also improve SQL efficiency without rewriting them.

B.6 Error Analysis Details

Figure 9 presents a detailed analysis of errors made by ChatGPT.

Wrong Schema Linking (41.6%)

Question:
List the names of schools with more than 30 differences in enrollments between K-12 and ages 5-17. Please also give the full street address of the schools.
Evidence:
Difference in enrollment = `Enrollment (K-12)` - `Enrollment (Ages 5-17)`
Ground Truth:
```
SELECT T1.School, T1.StreetAbr FROM
schools AS T1 INNER JOIN frpm AS T2 ON
T1.CDSCode = T2.CDSCode WHERE
T2.`Enrollment (K-12)` -
T2.`Enrollment (Ages 5-17)` > 30
```
ChatGPT SQL:
```
SELECT s.School, s.Street, s.City,
s.Zip FROM schools s JOIN frpm f ON
s.CDSCode = f.CDSCode WHERE
f.`Enrollment (K-12)` - f.`Enrollment
(Ages 5-17)` > 30
```

Question:
List out the accounts who have the earliest trading date in 1995.
Evidence:
False
Ground Truth:
```
SELECT account_id FROM trans WHERE
STRFTIME('%Y', date) = '1995' ORDER
BY date ASC LIMIT 1
```
ChatGPT SQL:
```
SELECT account_id FROM account WHERE
date = (SELECT MIN(date) FROM
account WHERE date LIKE '1995%')
```

Misunderstanding Knowledge Evidence (17.6%)

Question:
On average, how much did the Student_Club spend on food for the typical event in the past?
Evidence:
DIVIDE(SUM(spent), COUNT(spent)) where category = 'Food'; 'event in the past' means event_status = 'Closed'
Ground Truth:
```
SELECT SUM(spent) / COUNT(spent) FROM
budget WHERE category = 'Food' AND
event_status = 'Closed'
```
ChatGPT SQL:
```
SELECT DIVIDE(SUM(spent), COUNT(spent))
FROM budget WHERE category = 'Food' AND
event_status = 'Closed'
```

Question :
Name elementary schools in Riverside which the average of average math score for SAT is grater than 400, what is the funding type of these schools?
Evidence:
Average of average math = sum(average math scores) / count(schools)
Ground Truth:
```
SELECT T1.sname, T2.`Charter Funding
Type` FROM satscores AS T1 INNER JOIN
frpm ··· HAVING CAST(SUM(T1.AvgScrMath)
AS REAL) / COUNT(T1.cds) > 400
```
ChatGPT SQL:
```
SELECT s.School, s.FundingType FROM
schools s ··· AND sc.rtype = 'E' AND
sc.AvgScrMath > 400 GROUP BY s.School,
s.FundingType
```

Misunderstanding Database Content (40.8%)

Question:
What is the average math score of the school with the lowest average score for all subjects, and in which county is it located?
Evidence:
Average score for all subjects can be computed by AvgScrMath + AvgScrRead + AvgScrWrite
Ground Truth:
```
SELECT T1.AvgScrMath, T2.County FROM
satscores AS T1 INNER JOIN schools AS
T2 ON T1.cds = T2.CDSCode WHERE
T1.AvgScrMath IS NOT NULL ORDER BY
T1.AvgScrMath + T1.AvgScrRead +
T1.AvgScrWrite ASC LIMIT 1
```
ChatGPT SQL:
```
SELECT cname, AvgScrMath FROM satscores
JOIN schools ON satscores.cds =
schools.CDSCode WHERE satscores.rtype =
'S' ORDER BY (AvgScrMath + AvgScrRead +
AvgScrWrite) ASC LIMIT 1
```

Question:
What is the average lap time in milliseconds of all the lap records set on the various circuits in Italy?
Evidence:
average = AVG(milliseconds)
Ground Truth:
```
SELECT CAST(SUM(T1.milliseconds) AS
REAL) / COUNT(T1.lap) FROM lapTimes AS
T1 INNER JOIN races AS T2 on T1.raceId
= T2.raceId INNER JOIN circuits AS T3
on T2.circuitId = T3.circuitId WHERE
T3.country = 'Italy'
```
ChatGPT SQL:
```
SELECT AVG(milliseconds) FROM
lap_records WHERE circuit IN ('Monza',
'Imola', 'Mugello', 'Misano')
```

Syntax Error (3.0%)

Question:
How many meeting events were held in 2020?
Evidence:
meeting events refers to type = 'Meeting'; held in 2020 refers to YEAR(event_date) = 2020
Ground Truth:
```
SELECT COUNT(event_id) FROM event
WHERE type = 'Meeting' AND
STRFTIME('%Y', COLUMN) = '2020'
```
ChatGPT SQL:
```
SELECT COUNT(*) FROM event WHERE type
= 'Meeting' AND YEAR(event_date) =
2020
```

Question:
How much more was customer 7 consuming in April 2013 than customer 5?
Evidence:
April 2013 refers to '201304' in the yearmonth.date
Ground Truth:
```
SELECT SUM(IIF(CustomerID = 7,
Consumption, 0)) - SUM(IIF(CustomerID =
5, Consumption, 0)) FROM yearmonth
WHERE Date = '201304'
```
ChatGPT SQL:
```
SELECT7, (SELECT Consumption FROM
yearmonth WHERE CustomerID = 7 AND Date
= 201304) - (SELECT Consumption FROM
yearmonth WHERE CustomerID = 5 AND Date
= 201304)
```

Figure 9: 4 major types of error cases are presented. Some cases are shortcuts for better presentation.

B.7 Evaluation Details

During double-blind annotation in BIRD, we encountered numerous ambiguous issues that led to mismatches, predominantly due to unclear user intents. The most serious ambiguity is the use of "`DISTINCT`". Some annotators believe it should present only unique values, such as names, and cities, while others argue that it should be used only when questions explicitly mention "`different`" or "`distinctive`". Therefore, we use HashSet rather than List to compare final results since HashSet disregards row order and automatically filters repetitive rows to reduce this ambiguity. However, this may result in false positives for questions utilizing "ORDER BY." We identify three "ORDER BY" usage scenarios in BIRD: **1) Rank-based questions** (e.g., "`Show me the top 5 students according to their math scores`"): The order is less important as long as the results contain the correct students. **2) Superlative questions**: (e.g., "`List the longest river in the USA`"): The answer typically contains only one item (or tied results), so the impact is minimal. **3) Questions requiring a specific order** (e.g., "`Show me the top five students based on their math scores in descending order`"): This scenario explicitly requires correct ordering and may lead to false positives. However, such instances are uncommon, accounting for less than 1% of BIRD.

B.8 Human Performance Collection

The procedure of collecting human performance is still rigorous. During the annotation, all data is divided into 10 batches for better management and error tracks by experts. The first 8 batches of data is the final training data and dev data for public use, and the remaining 2 batches of data are used for testing. We consider the annotation of the first 8 batches of data as a learning process for SQL annotators since their erroneous SQLs could be fixed by experts and learn how to generate good-quality SQLs for this task. Then their first scores on an examination, conducted by testing set from the final two batches can be viewed as human performance since we don't interrupt and assist them during the examination and all errors are preserved. After testing, we proceed with the following double-blind SQL annotation procedures as Section 3.4 to correct SQLs for these data by a discussion with experts. And SQLs after the second round of double-blind annotation are collected as ground truth.

B.9 Distribution of Open-source Databases

The databases in BIRD are all in accordance with one of following licenses:

Public Domain Public Domain Mark
A public domain license refers to a legal designation that allows intellectual property, such as creative works or inventions, to be freely used, shared, and built upon by anyone without restrictions. When a work is in the public domain, it is no longer protected by copyright, patent, or trademark laws.

CC-BY Creative Commons Attribution 4.0 International
This license is one of the open Creative Commons licenses and allows users to share and adapt the dataset so long as they give credit to the creator.

CC-BY-SA Creative Commons Attribution-ShareAlike 4.0 International
This license is one of the open Creative Commons licenses and allows users to share and adapt the dataset so long as they give credit to the creator and distribute any additions, transformations, or changes to the dataset under this license.

GPL General Public License
The GPL was created by the Free Software Foundation (FSF) and is also known as the GNU GPL, as it is used by the GNU Project. And it allows users to use, study, share, and modify the software under certain terms and conditions.

CPOL Code Project Open License
It is a software license that is often used for articles, tutorials, and sample code shared on The Code Project website. The CPOL is intended to be a more permissive license, allowing developers to use, modify, and distribute the software without many of the restrictions imposed by other licenses like the GPL.

CC0 Creative Commons Zero
It is a public domain dedication tool created by Creative Commons. It allows creators to waive all their copyright and related rights in a work, effectively placing it in the public domain. This means that anyone can freely use, share, modify, and build upon the work without seeking permission or providing attribution to the original creator.

B.10 Keyword Statistic

We have conducted a comprehensive analysis of the keywords employed in the BIRD dataset and visualize the results in the form of a nice-looking word cloud, which can be found in Figure 10. We further classify keywords into 7 following categories:

Main Body Keywords • `SELECT` • `FROM` • `WHERE` • `AND` • `OR` • `NOT` • `IN` • `EXISTS` • `IS` • `NULL` • `IIF` • `CASE` • `CASE WHEN`.

Join Keywords • `INNER JOIN` • `LEFT JOIN` • `ON` • `AS`.

Clause Keywords • `BETWEEN` • `LIKE` • `LIMIT` • `ORDER BY` • `ASC` • `DESC` • `GROUP BY` • `HAVING` • `UNION` • `ALL` • `EXCEPT` • `PARTITION BY`.

Aggregation Keywords • `AVG` • `COUNT` • `MAX` • `MIN` • `ROUND` • `SUM`.

Scalar Keywords • `ABS` • `LENGTH` • `STRFTIME` • `JULIADAY` • `NOW` • `CAST` • `SUBSTR` • `INSTR`.

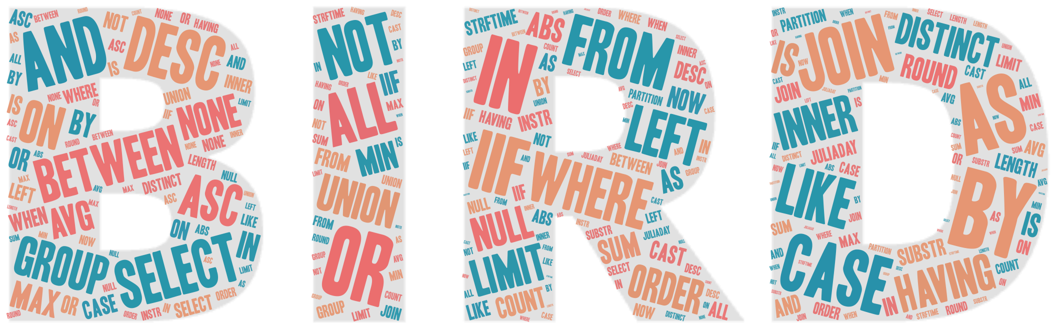

Figure 10: Keyword cloud presentation for SQLs in BIRD.

Comparison Keywords • = • > • < • >= • <= • != .

Computing Keywords • − • + • ⋆ • / .

B.11 Study about Text-to-SQL Models

The fundamental principle of a cross-domain text-to-SQL parser involves the construction of an encoder to learn representations of questions and schemas, followed by a decoder to generate SQLs [35]. For example, IRNET [11] designs an encoder consisting of attention-based Bi-LSTM for learning question and schema representations, and a decoder to predict SQLs based on the encoded intermediate representations. RATSQL [41], SDSQL [16], LGESQL [3], and S^2SQL [17], Proton [42] enhance the representation learning of natural language questions and database schema via relational graph neural network. R^2SQL [15], SCoRE [53], and STAR [2] enhance contextual learning for conversational text-to-SQL tasks. Later, sequence-to-sequence pre-trained language models (PLMs) such as T5 [36] become popular in text-to-SQL tasks due to their portability and capability of generation across different datasets. These models achieve impressive results by fine-tuning with minimal effort. Furthermore, RASAT [34] enhances T5's structural information encoding via schema alignment into the encoder, while Graphix [26] equips T5 with multi-hop reasoning to achieve state-of-the-art results on complicated cross-domain text-to-SQL tasks. In recent years, LLMs such as ChatGPT [31], Palm [7], OPT [55], have attracted considerable attention due to their powerful zero-shot reasoning and domain generalization capabilities. ChatGPT can perform exceptionally well on semantic parsing tasks, including text-to-SQL tasks, with minimal input data. In fact, in the BIRD project, ChatGPT even performs more impressively than initially expected.

Study about SQL Efficiency Efficient execution of SQL queries on big databases has been a significant topic in both academia and industries. Many techniques are proposed to improve SQL query efficiency, by index selection [21], SQL optimization [25, 60], etc. SQL optimiztion is a common method for enhancing the efficiency of SQL queries. Several SQL optimization algorithms [27, 29, 45], such as rule-based optimization and cost-based optimization, are proven effective. Rule-based optimization employs a set of principles to transform the SQL query into a form that can be executed more efficiently. On the other hand, cost-based optimization estimates the execution cost of various query plans and selects the one with the lowest cost by analyzing the statistic distribution of database values. Similar to the NLP community, there are also recent works utilizing artificial intelligence for query optimization such as [60]. Index prediction is another important technique for improving SQL execution efficiency. Researchers propose many algorithms of index prediction [59] based on various optimization criteria, such as minimizing SQL execution time, and maximizing index utilization. In this work, we provide VES to measure the efficiency of text-to-SQL generators to encourage them to generate accurate and fast SQLs for users.

B.12 Limitation and Future work

Despite the high quality of SQL annotation produced by double-blind annotation, the procedure is resource-intensive. Future research could explore a human-computer interaction (HCI) based approach, incorporating advanced AI systems such as GPT-4 for taking parts of annotation duties, to maintain data quality while reducing human effort. In addition, SQLite was chosen as the primary SQL codebase for previous text-to-SQL benchmarks and this study since it's friendly to users. While it presents difficulties in fetching Query Execution Plans (QEP) for precise efficiency computation and adapting to different SQL syntaxes. Future work will include PostgreSQL and MySQL versions of BIRD to resolve these limitations and provide a more robust research environment for both NLP and DB experts.