

## A.8 Interactive Correction Cases

Before	After	Reason
<b>Q:</b> What is the phone number of the school that has the highest number of test takers with an SAT score of over 1500?		
<b>Before:</b> SELECT schools.Phone FROM satscores JOIN schools ON satscores.cds = schools.CDSCode WHERE satscores.NumGE1500 > 1500 GROUP BY schools.School ORDER BY COUNT(*) DESC LIMIT 1;	<b>After:</b> SELECT schools.Phone FROM satscores JOIN schools ON satscores.cds = schools.CDSCode ORDER BY satscores.NumGE1500 DESC LIMIT 1;	The WHERE clause is checking if the number of test takers with an SAT score over 1500 is greater than 1500, which is not what the question is asking. The question is asking for the school with the highest number of test takers who scored over 1500. Additionally, the GROUP BY clause is grouping by the school name, which may not be unique. It would be better to group by the school code (CDSCode) which is unique for each school.
<b>Q:</b> List the top five schools, by descending order, from the highest to the lowest, the most number of Enrollment (Ages 5-17). Please give their NCES school identification number.		
<b>Before:</b> SELECT T1."School Name", T2.NCESSchool FROM frpm AS T1 JOIN schools AS T2 ON T1.CDSCode = T2.CDSCode ORDER BY T1."Enrollment (Ages 5-17)" DESC LIMIT 5;	<b>After:</b> SELECT T2.NCESSchool FROM frpm AS T1 JOIN schools AS T2 ON T1.CDSCode = T2.CDSCode ORDER BY T1."Enrollment (Ages 5-17)" DESC LIMIT 5;	The question asks for the top five schools with the most number of Enrollment (Ages 5-17), and their NCES school identification number. However, the SQL query provided returns the school name, which is not required by the question.
<b>Q:</b> find the package option of the tv channel that do not have any cartoon directed by Ben Jones.		
<b>Before:</b> SELECT TV_Channel.Package_Option FROM TV_Channel LEFT JOIN Cartoon ON TV_Channel.id = Cartoon.Channel WHERE Cartoon.Directed_by IS NOT 'Ben Jones';	<b>After:</b> SELECT TV_Channel.Package_Option FROM TV_Channel WHERE TV_Channel.id NOT IN ( SELECT Cartoon.Channel FROM Cartoon WHERE Cartoon.Directed_by = 'Ben Jones');	The current query is selecting all channels where the director is not Ben Jones, but it should exclude channels that have any cartoons directed by Ben Jones. This can be achieved by using a subquery to find all channels that have cartoons directed by Ben Jones, and then excluding these channels in the main query. Here is the corrected SQL:
<b>Q:</b> List the earnings of poker players in descending order.		
<b>Before:</b> SELECT * FROM poker_player ORDER BY Earnings DESC;	<b>After:</b> SELECT Earnings FROM poker_player ORDER BY Earnings DESC;	The question only asks for the earnings of poker players. Therefore, we should only select the 'Earnings' column to make the result more concise and relevant to the question.
<b>Q:</b> How many countries speak both English and Dutch?		
<b>Before:</b> SELECT COUNT(DISTINCT CountryCode) FROM countrylanguage WHERE Language = 'English' AND CountryCode IN ( SELECT CountryCode FROM countrylanguage WHERE Language = 'Dutch')	<b>After:</b> SELECT COUNT(*) FROM ( SELECT CountryCode FROM countrylanguage WHERE Language = 'English' INTERSECT SELECT CountryCode FROM countrylanguage WHERE Language = 'Dutch')	The SQL query provided uses the 'IN' clause, which selects countries that speak either English or Dutch. However, the question asks for countries speaking both languages. Using IN results in a union of sets (either English or Dutch), not an intersection (both English and Dutch). This could be done with two separate queries joined by an 'INTERSECT' clause.

Table 8

## A.9 Prompt Examples

Describe how you understand the question based on the evidence, and help me write an SQL to answer the question.

### EVIDENCE: {evidence}

### USER\_QUESTION: {question}

### RELATED SQL:

{related\_sql}

### DATABASE STRUCTURE:

{schema}

Prompt 1: CoT-SQL-Writer

Write an to answer the question.

Program of Thoughts (PoT) is a variant of Chain of Thought (CoT), pre-generating Python code to assist in the creation of SQL. Please apply PoT (and PoT only) before generating an SQL.

In your python code, `Table %s` is stored in `db\_dict['%s']`, `db\_dict` is of type dict[pandas.DataFrame].

### RELATED SQL:

{related\_sqls}

### DATABASE STRUCTURE:

{schema}

### EXAMPLES:

QUESTION: What is %s in the earliest year and what year was it?

SQL:

```
earliest_year = db_dict[%s]['Year'].min()
```

```
year_filtered_data = step1_result[step1_result['Year'] ==  
earliest_year]
```

```
result = year_filtered_data[[%s, 'Year']]
```

```
```sql
```

```
SELECT T1.%s, T2.Year FROM %s AS T1 JOIN %s AS T2 ON T1.Id = T2.Id  
WHERE T2.Year = (SELECT min(YEAR) FROM %s);
```

```
```
```

QUESTION: Show names for all %s except for %s having a %s in year 2023.

SQL:

```
%s_2023 = db_dict['%s'][db_dict['%s']['year'] == '2023']
```

```
result = db_dict[%s][~db_dict[%s][%s].isin(%ss_2023[%s])]
```

```
```sql
```

```
SELECT name FROM %s EXCEPT SELECT T2.name FROM %s AS T1 WHERE T1.  
year = 2023
```

```
```
```

QUESTION: Find the %s that %s is A and B?

SQL:

```

condition_a_data = db_dict[%s][db_dict['Cartoon'][%s] == 'A']
condition_b_data = db_dict[%s][db_dict['Cartoon'][%s] == 'B']
result = pd.merge(condition_a_data, condition_b_data, how='inner')
```sql
SELECT T1.%s FROM %s AS T1 WHERE %s = 'A'
INTERSECT
SELECT T1.%s FROM %s AS T1 WHERE %s = 'B'
```

### EVIDENCE: {evidence}
### USER_QUESTION: {question}
### SQL:

```

#### Prompt 2: PoT-SQL-Writer

You are the manager of a Database project. You are going to invite {n} experts to review an SQL query.  
Who would you invite?

considering:  
(1) the domain of this database;  
(2) the structure of this SQL.  
Please write your invitation as a JSON format dictionary, Enclose the JSON within ```json...```.

### DATABASE STRUCTURE:

{schema}

### QUESTION: {question}

### SQL:

{pred\_sql}

### EXAMPLES:

```

```json
{
  "Reviewer PVsg": "Data Analyst in automotive industry",
  "Reviewer 2KtR": "Senior Database Engineer specialized in writing
    various clauses",
  "Reviewer LmN3": "Senior Database Engineer specialized in writing
    filtering conditions"
}
```

```

### INVITATION:

#### Prompt 3: Invitation