Reproduction of Sphinteract on KaggleDBQA (Ambiguity Filtered)

This notebook reproduces the three methods described in the paper "Sphinteract: ...":

1. **Baseline (Method 1)**: Zero-shot generation with Self-Correction (Fix Invalid).
2. **Sphinteract (Method 2)**: Interactive framework with Clarification Questions (SRA) and Feedback.
3. **Break No Ambiguity (Method 3)**: Similar to Method 2 but with an Early Stopping mechanism.

We use the **KaggleDBQA** dataset.

Modification: Instead of random sampling, we filter for 20 "Ambiguous" samples using GPT-3.5-Turbo.

Requirements

- `openai`
- `pandas`
- `sqlite3`
- `numpy`
- `python-dotenv`
- `tiktoken`
- `xxhash`

A `.env` file with `OPENAI_API_KEY` and `OPENAI_BASE_URL` (optional) is needed.

In [31]:
```
import nbformat as nbf
import os
import pandas as pd
import sqlite3
import numpy as np
import time
import xxhash
import json
```

```python
import pickle
import sys
from openai import OpenAI
from dotenv import load_dotenv
from multiprocessing import Process, Queue
from langchain_community.vectorstores import Chroma
from langchain_community.embeddings import OpenAIEmbeddings
import matplotlib.pyplot as plt
import seaborn as sns
import concurrent.futures
import re

# Add current directory and parent directory to path to allow importing local modules
sys.path.append(os.getcwd())
sys.path.append(os.path.abspath(os.path.join(os.getcwd(), '..')))
from reproduction_utils import execute_query_worker

# Load environment variables
load_dotenv(override=True)

api_key = os.getenv("OPENAI_API_KEY")
base_url = os.getenv("OPENAI_BASE_URL")

if not api_key:
    raise ValueError("Please set OPENAI_API_KEY in your .env file.")

client = OpenAI(
    api_key=api_key,
    base_url=base_url if base_url else "https://api.openai.com/v1"
)

# Initialize Vectorstore for Few-Shot
try:
    embeddings = OpenAIEmbeddings(openai_api_key=api_key)
    if os.path.exists("./userstudy_chroma"):
        vectorstore = Chroma(persist_directory="./userstudy_chroma", embedding_function=embeddings)
        print("Vectorstore loaded successfully.")
    else:
        print("Warning: userstudy_chroma directory not found. Few-shot retrieval may fail.")
        vectorstore = None
except Exception as e:
```

```
    print(f"Error loading vectorstore: {e}")
    vectorstore = None

print("Environment setup complete.")
```

```
Vectorstore loaded successfully.
Environment setup complete.
```

Helper Functions

In [32]:
```
# Helper Functions
```

In [33]:
```
def clean_query(sql_query):
    # Remove markdown code blocks
    pattern = r"```sql(.*?)```"
    match = re.search(pattern, sql_query, re.DOTALL | re.IGNORECASE)
    if match:
        sql_query = match.group(1)
    else:
        sql_query = sql_query.replace("```sql", '').replace("```", '')

    sql_query = sql_query.replace(';', '')
    sql_query = sql_query.replace('"""', '')

    # Find the start of the SQL statement (SELECT or WITH)
    match_select = re.search(r'\bSELECT\b', sql_query, re.IGNORECASE)
    match_with = re.search(r'\bWITH\b', sql_query, re.IGNORECASE)

    start_index = -1
    if match_with and match_select:
        start_index = min(match_with.start(), match_select.start())
    elif match_with:
        start_index = match_with.start()
    elif match_select:
        start_index = match_select.start()

    if start_index != -1:
        sql_query = sql_query[start_index:]
    else:
```

```
        # If no SELECT/WITH found, assume it's a completion and prepend SELECT
        # But ensure we don't double prepend if the user output " * FROM ..."
        if 'FROM' in sql_query.upper():
             sql_query = 'SELECT ' + sql_query
        else:
             # Fallback: just prepend SELECT
             sql_query = 'SELECT ' + sql_query

    return sql_query.strip()

def generate_db_schema(db_path):
    conn = sqlite3.connect(db_path)
    cursor = conn.cursor()
    cursor.execute("SELECT name FROM sqlite_master WHERE type='table'")
    tables = cursor.fetchall()
    schemas = []
    for table in tables:
        if table[0] == 'sqlite_sequence':
            continue
        cursor.execute(f"SELECT sql FROM sqlite_master WHERE type='table' AND name='{table[0]}'")
        create_prompt = cursor.fetchone()[0]
        schemas.append(create_prompt)
    conn.close()
    return "\n\n".join(schemas)

def evalfunc(sql_source, sql_target, db_path):
    if not os.path.isfile(db_path):
        return False, [FileNotFoundError(f"Database not found: {db_path}")]

    timeout = 30 # seconds
    output = Queue()
    p = Process(target=execute_query_worker, args=(db_path, sql_source, output))
    p.start()

    try:
        source_results = output.get(timeout=timeout)
        p.join()
        if isinstance(source_results, Exception):
            return False, [source_results]
    except Exception as e:
        p.terminate()
```

```
        return False, [e]

    # Execute Gold SQL (Target) - assumed safe and fast
    try:
        conn = sqlite3.connect(db_path)
        cursor = conn.cursor()
        target_results = cursor.execute(sql_target).fetchall()
        conn.close()
    except Exception as e:
        return False, [e]

    # Compare results
    if len(source_results) != len(target_results):
        return False, []

    # Heuristic comparison (order-independent if no ORDER BY, else strict)
    if 'ORDER BY' in sql_target.upper():
        return source_results == target_results, []
    else:
        # Sort both by a stable key (str representation)
        s_sorted = sorted(list(source_results), key=lambda x: str(x))
        t_sorted = sorted(list(target_results), key=lambda x: str(x))
        return s_sorted == t_sorted, []

def LLM_generation(prompt, model='gpt-3.5-turbo', temperature=0.0, retries=3, retry_delay=1.5, log_each_retry=False
    last_err = None
    models_to_try = [model]
    if fallback_models is None:
        fallback_models = [os.getenv("AMBIGUITY_MODEL", "gpt-4o-mini"), "gpt-4o"]
    for m in fallback_models:
        if m not in models_to_try:
            models_to_try.append(m)
    for try_model in models_to_try:
        for attempt in range(retries):
            try:
                response = client.chat.completions.create(
                    model=try_model,
                    messages=[{"role": "user", "content": prompt}],
                    temperature=temperature,
                    max_tokens=4096
                )
```

```
                return response.choices[0].message.content.strip(), 0.0
            except Exception as e:
                last_err = e
                err_str = str(e)
                l = err_str.lower()
                if "502" in err_str or "bad gateway" in l:
                    err_label = "502 Bad Gateway"
                elif "429" in err_str or "too many requests" in l or "rate limit" in l:
                    err_label = "rate_limit"
                elif "timeout" in l or "timed out" in l:
                    err_label = "timeout"
                elif "connection reset" in l or "reset by peer" in l:
                    err_label = "conn_reset"
                elif "ssl" in l:
                    err_label = "ssl_error"
                elif "unauthorized" in l or "401" in err_str:
                    err_label = "unauthorized"
                elif "model" in l and "not found" in l:
                    err_label = "model_not_found"
                else:
                    err_label = "error"
                if log_each_retry:
                    print(f"LLM error ({err_label}), retry {attempt+1}/{retries}")
                time.sleep(retry_delay * (1.5 ** attempt))
    if last_err is not None:
        err_str = str(last_err)
        l = err_str.lower()
        if "502" in err_str or "bad gateway" in l:
            err_label = "502 Bad Gateway"
        elif "429" in err_str or "too many requests" in l or "rate limit" in l:
            err_label = "rate_limit"
        elif "timeout" in l or "timed out" in l:
            err_label = "timeout"
        elif "connection reset" in l or "reset by peer" in l:
            err_label = "conn_reset"
        elif "ssl" in l:
            err_label = "ssl_error"
        elif "unauthorized" in l or "401" in err_str:
            err_label = "unauthorized"
        elif "model" in l and "not found" in l:
            err_label = "model_not_found"
```

```
        else:
            err_label = "error"
        print(f"LLM error ({err_label}); giving up")
    return "SELECT * FROM error", 0.0
```

Prompts

In [34]:
```
# Prompts (Aligned with FewShotAmbSQL.ipynb)
```

In [35]:
```
SRA = """/* Ask the user a new multiple choice clarification question to help you find the correct SQL answer for t
{question}
/* Given the following database schema: */
{schema}
/* And the following incorrect sql answers: */
{sqls}
/* And the following previous clarification questions and user replies: */
{cqs}

/* Consider the following ambiguity categories:
    - AmbQuestion: Is the question itself ambiguous?
    - AmbTableColumn: Is there ambiguity in mapping the entities from the QUESTION to tables and columns in the DAT
    - AmbOutput: What fields and how many fields should be included in the output table?
    - AmbValue: What predicate value should be used to filter results?
*/

/* The clarification question should be easy to understand for people with no coding experience. */

/* Let's think step by step to generate the helpful multiple choice clarification question.
1. Summarize the clear information based on previous clarification questions and incorrect queries.
2. Evaluate whether AmbQuestion, AmbTableColumn, AmbOutput, and AmbValue remain in formulating an SQL query, consid
3. Ask a new multiple-choice question to address the remaining ambiguities and assist in identifying the correct SQ
4. Prioritize granularity alignment and valid join keys; avoid suggesting joins across incompatible levels (e.g., d
*/
"""

SRA_ES = """/* Ask the user a new multiple choice clarification question to help you find the correct SQL answer fo
{question}
/* Given the following database schema: */
{schema}
```

```
/* And the following incorrect sql answers: */
{sqls}
/* And the following previous clarification questions and user replies: */
{cqs}

/* Consider the following ambiguity categories:
    - AmbQuestion: Is the question itself ambiguous?
    - AmbTableColumn: Is there ambiguity in mapping the entities from the QUESTION to tables and columns in the DAT
    - AmbOutput: What fields and how many fields should be included in the output table?
    - AmbValue: What predicate value should be used to filter results?
*/

/* The clarification question should be easy to understand for people with no coding experience. */

/* Let's think step by step to generate the helpful multiple choice clarification question.
1. Summarize the clear information based on previous clarification questions and incorrect queries.
2. Evaluate whether AmbQuestion, AmbTableColumn, AmbOutput, and AmbValue remain in formulating an SQL query, consid
3. If no remaining ambiguities are identified, then output "NO AMBIGUITY".
   Else, ask a new multiple-choice question to address the remaining ambiguities and assist in identifying the corr
4. Prioritize granularity alignment and valid join keys; avoid suggesting joins across incompatible levels (e.g., d
*/
"""

sql_generation_v2 = """/* Given the following database schema: */
{schema}
/* And the following incorrect sql answers: */
{sqls}
/* And the following user replies to help you write the correct sql query: */
{cqas}

{metadata}
/* Answer the following with no explanation: {question} */
/* Output ONLY SQL wrapped in a markdown block: ```sql */
"""

fix_invalid_v1 = """/* Given the following database schema: */
{schema}
/* And the following inexecutable sql query */
{invalidSQL}
/* And the following exception message */
{ex}
```

```
/* Fix the exception and write a new executable SQL query with no explanation */
/* Output ONLY SQL wrapped in a markdown block: ```sql */
"""

sql_generation_selfdebug = """/* Given the following database schema: */
{schema}
/* And the following incorrect sql answers: */
{sqls}

{metadata}
/* Answer the following with no explanation: {question} */
/* Output ONLY SQL wrapped in a markdown block: ```sql */
"""

def build_metadata_constraints(nlq, schema):
    s_lower = schema.lower()
    n_lower = nlq.lower()
    cons = []
    cons.append("Constraints: SQL must be executable on the given schema and use a single consistent granularity ac
    cons.append("When tables differ in granularity, aggregate to a common key before joining; do not join district/
    cons.append("Use only valid join keys present in the schema with matching types; prefer exact equality joins.")
    cons.append("If feedback conflicts with these constraints, follow the constraints.")
    if ("finrev_fed_17" in s_lower) and ("ndecorexcel_math_grade8" in s_lower):
        cons.append("For FINREV_FED_17 with NDECoreExcel_Math_Grade8, compute metrics at state+year granularity: ag
    if ("resultsdata15" in s_lower) and ("lod" in s_lower or "limit of detection" in n_lower):
        cons.append("For 'easiest to be tested' tasks, use the pesticide with lowest average LOD; if LOD is unavail
    meta = "\n".join([f"/* {c} */" for c in cons])
    return meta

cq_prefix_v1 = '''/* some examples are provided */
/* example question: */
Which artist/group is most productive?
/* example previous clarification questions and user replies: */
clarification questions: How to rank artist/group productivity? a) rank by the number of records produced, b) rank
user: b) rank by the total number of downloads```
/* example reasoning and remaining ambiguity type*/
It is clear that the SQL answer should use ORDER BY and LIMIT 1 based on the sum of total downloads. However, it is
/* example clarification question */
mul_choice_cq = "Which columns represent the 'artist/group' information? a) the artist column only, b) the groupNam
```

```
/* example question: */
Which Premier League matches ended in a draw in 2016?
/* example previous clarification questions and user replies: */
clarification questions: Is the year '2016' referring to? a) season is 2016, b) season is either 2015/2016 or 2016/
user: a) season is 2016,
clarification questions: How to find the 'Premier league'? a) consider all leagues, b) consider only the league wit
user: b) consider only the league with name 'Premier League'
/* example reasoning and remaining ambiguity type*/
It is clear that the SQL answer to this question needs to contain a WHERE clause for three conditions: 'Premier Lea
/* example clarification question */
mul_choice_cq = “What fields represent the target 'matches'? a) all fields from football data table, b) the `league

/* example question: */
Which type of crime has the highest rate of ‘Investigation complete’?
/* example previous clarification questions and user replies: */
No previous clarification questions.
/* example reasoning and remaining ambiguity type*/
It is clear that the SQL answer to this question needs to contain a WHERE clause to find crimes that have 'Investig
/* example clarification question */
mul_choice_cq = “What information should be used to find 'Investigation complete'? a) see if outcome contains the p

/* example question: */
For award winners, which position has the most hall of fame players?
/* example previous clarification questions and user replies: */
clarification questions: How should the 'position' for players be identified? a) by the `award_id` column, b) by th
user: c)  by the `note` column
/* example reasoning and remaining ambiguity type*/
It is clear that the answer should use the `note` column for player ‘positions’. However, it is unclear what fields
/* example clarification question */
mul_choice_cq = “What fields should be contained in the output table? a) one field: the position, b) two fields: th

/* example question: */
How many Wisconsin school districts receive federal funding?
/* example previous clarification questions and user replies: */
clarification question: How to determine if a district has received federal funding? a) based on the t_fed_rev is l
user: c) every school in `FINREV_FED_17` table has received federal funding.
/* example reasoning and remaining ambiguity type*/
It is clear that every school in `FINREV_FED_17` table have received federal funding. However, it is unclear if the
/* example clarification question */
mul_choice_cq = “Is 'Wisconsin school districts' referring to? a) all school districts in the state Wisconsin, b) s
```

```
/* example question: */
How many 2-year public schools are there in "California"?
/* example previous clarification questions and user replies: */
clarification question: Which column(s) should be used to find ‘2-year public schools’? a) `level` column, b) `cont
user: c) use both `level` and `control` columns to find ‘2-year public schools’ information.
/* example reasoning and remaining ambiguity type*/
It is clear that the correct SQL answer should have a WHERE clause with filters based on the `level` and `control`
/* example clarification question */
mul_choice_cq = “What predicate values should be used for the `level` and `control` columns to find  ‘2-year public

/* example question: */
Calculate the total beat of the crimes reported in a community area in the central side with a population of 50,000
/* example previous clarification questions and user replies: */
clarification question: What column and predicate value should be used to determine ‘central side’? a) Column `side
user: b) Column `side` in table `Community_Area` with value ‘Central’
/* example reasoning and remaining ambiguity type*/
It is clear that the output table should contain a single number and use the predicate ‘Central’ in `Community_Area
/* example clarification question */
mul_choice_cq = “Which column is related to ‘total beat’? a) `Crime`.`beat`, b) `Crime`.`report_no`, c) other (plea

/* example question: */
Of all the nonessential genes that are not of the motorprotein class and whose phenotype is cell cycle defects, how
/* example previous clarification questions and user replies: */
No previous clarification questions.
/* example reasoning and remaining ambiguity type*/
It is clear that ‘phenotype’ is referring to the `Phenotype` column, ‘motorprotein class’ is referring to the `clas
/* example clarification question */
mul_choice_cq = “What fields should be included in the output table? a) One column for the number of genes b) Two c
\n\n
'''

feedback_v2 = """/* Given the following Natural Language Question: */
{nlq}
/* And the following Gold Query: */
{query}
/* Answer the following multiple choice clarification question truthfully based on the Gold Query: */
{question}

/* Follow these steps:
1. Identify which portion of the Gold Query answers the clarification question.
2. Evaluate the correctness of each multiple choice answer based only on the Gold Query.
```

```
3. If none of the choices are correct or you select "other (please specify)", provide a short answer for the clarif
4. Output the final answer in the format: answer_to_cq = "".

Let's proceed step by step. */
/* Only use information from the Gold Query; do not guess.
   Prefer answers that maintain consistent granularity and valid join semantics. */
"""

feedback_prefix_v1='''/* some examples are provided */
/* example question: */
How many acres burned in fires in California each year between 2000 and 2005?
/* example gold sql query*/
SELECT\n  SUM(FIRE_SIZE),\n  FIRE_YEAR\nFROM Fires\nWHERE\n  State = "CA" AND FIRE_YEAR BETWEEN 2000 AND 2005\nGROU
/* example clarification question*/
What information should the output table contain? a) two columns: the total acres burned and the year, b) one colum
/* example reasoning */
Output table is determined by the SELECT clause in the gold sql query. The gold query uses 'SELECT  SUM(FIRE_SIZE),
/* example  answer*/
answer_to_cq = "a) two columns: the total acres burned and the year"

/* example question: */
What was the most common cause of fire between 2000 and 2005?
/* example gold sql query*/
SELECT\n  STAT_CAUSE_DESCR\nFROM Fires\nWHERE\n  FIRE_YEAR BETWEEN 2000 AND 2005\nGROUP BY\n  STAT_CAUSE_DESCR\nORD
/* example clarification question*/
Which information should be used to represent the 'cause of fire'? a) the code that represents the cause, b) the de
/* example reasoning */
The clarification question is asking for which column should be used to represent the cause of fire. The gold query
/* example  answer*/
answer_to_cq = "b) the description of the cause"

/* example question: */
Whose CDs sells best?
/* example gold sql query*/
SELECT\n  artist\nFROM torrents\nGROUP BY\n  artist\nORDER BY\n  SUM(totalSnatched) DESC\nLIMIT 1;
/* example clarification question*/
Which column should be used to identify music related to 'CD'? a) groupName, b) tag, c) releaseType, d) other (plea
/* example reasoning */
The gold query does not use a WHERE clause to filter the CDs. Hence, the CD information is not contained in the tag
/* example  answer*/
answer_to_cq = "d) Consider all music; No filter on 'CD' "
```

```
/* example question: */
How many people wrote comments for the question "Any additional notes or comments."? */
/* example gold sql query*/
SELECT COUNT(T1.UserID) FROM Answer AS T1 INNER JOIN Question AS T2 ON T1.QuestionID = T2.questionid WHERE T2.quest
/* example clarification question*/
How to determine if a user has provided comments? a) no check needed, b) see if `AnswerText` column has empty strin
/* example reasoning */
In the gold SQL query, it checks “T1.AnswerText IS NOT NULL”. Hence, choice a and b are both wrong.
/* example  answer*/
answer_to_cq = "c) ‘wrote comments’ imply `AnswerText` is not a NULL value".

/* example question: */
Calculate the difference between the number of customers and the number of subscribers who did the trip in June 201
/* example gold sql query*/
SELECT SUM(IIF(subscription_type = 'Subscriber', 1, 0)) - SUM(IIF(subscription_type = 'Customer', 1, 0)) FROM trip
/* example clarification question*/
What predicate value should be used to determine a trip in June 2013? a) start_data > 06/2013, b) start_data = ‘Jun
/* example reasoning */
The gold sql query uses start_date LIKE '6/%/2013%' to find trips in June 2013.
/* example  answer*/
answer_to_cq = "c) start_date LIKE '6/%/2013%'"


/* example question: */
Identify the players who weigh 120 kg.
/* example gold sql query*/
SELECT T2.PlayerName FROM weight_info AS T1 INNER JOIN PlayerInfo AS T2 ON T1.weight_id = T2.weight WHERE T1.weight
/* example clarification question*/
What fields should be contained in the output? a) one column of player name, b) one column of player id, c) two col
/* example reasoning */
The gold query selects ‘SELECT T2.PlayerName’. Hence, a is correct.
/* example  answer*/
answer_to_cq = "a) one column of player name"

/* example question: */
How many reviews are created for the podcast "Scaling Global" under?
/* example gold sql query*/
SELECT COUNT(T2.content) FROM podcasts AS T1 INNER JOIN reviews AS T2 ON T2.podcast_id = T1.podcast_id WHERE T1.tit
/* example clarification question*/
Which column represents the reviews? a) `podcast` column, b) `content` column, c) other (please specify).
```

```
/* example reasoning */
The gold query uses “COUNT(T2.content)” to determine the number of reviews. Hence, b is correct in which the `conte
/* example  answer*/
answer_to_cq = "b) `content` column"
\n\n
'''

selfdebug_examples_prefix = '''/* Given the following incorrect sql asnwers: */
SELECT creation, COUNT(*) FROM department GROUP BY creation ORDER BY
COUNT(*) DESC LIMIT 1
/* Answer the following with no explanation: In which year were most departments established? */
SELECT creation FROM department GROUP BY creation ORDER BY COUNT(*) DESC LIMIT 1
-------
/* Given the following incorrect sql asnwers: */
SELECT customers.customer_name FROM customers JOIN orders ON customers.customer_id = orders.customer_id WHERE order
/* Answer the following with no explanation: Which customers have both "On Road" and "Shipped" as order status? Lis
SELECT customers.customer_name FROM customers JOIN orders ON customers.customer_id = orders.customer_id WHERE order
-------
/* Given the following incorrect sql asnwers: */
SELECT COUNT(status) FROM city
/* How many different statuses do cities have? */
SELECT COUNT(DISTINCT status) FROM city
-------'''
selfdebug_examples = selfdebug_examples_prefix.split('-------')

# Generate few-shot strings for self-debug (Method 1 & Refinement)
selfdebug_few_shot = []
for i in range(1, 4): # We only need up to 3 for this reproduction
    prefix = []
    for j in range(i):
        prefix.append(selfdebug_examples[j])
    selfdebug_few_shot.append('\n'.join(prefix))

fewshot_prefix = "/* some examples are provided */\n"
```

In [36]:
```
# Data Loading
```

In [37]:
```
dataset_path = 'kaggle_dataset.csv'
if not os.path.exists(dataset_path):
    raise FileNotFoundError("kaggle_dataset.csv not found.")
```

```
df = pd.read_csv(dataset_path)
print(f"Loaded {len(df)} records.")

# Prepare Schema Cache
db_schema_cache = {}
def get_schema(db_name):
    if db_name not in db_schema_cache:
        db_path = f'./databases/{db_name}/{db_name}.sqlite'
        if not os.path.exists(db_path):
             db_path = f'./databases/{db_name}.sqlite'

        if os.path.exists(db_path):
             db_schema_cache[db_name] = generate_db_schema(db_path)
        else:
             db_schema_cache[db_name] = ""
    return db_schema_cache[db_name]

def get_db_path(db_name):
    paths = [
        f'./databases/{db_name}/{db_name}.sqlite',
        f'./databases/{db_name}.sqlite'
    ]
    for p in paths:
        if os.path.exists(p):
            return p
    return None

def get_few_shot_examples(target_nlq, n_shots=3):
    if not vectorstore or n_shots <= 0:
        return ""
    try:
        docs = vectorstore.similarity_search(target_nlq, k=n_shots)
        examples = []
        for doc in docs:
            nl = doc.metadata.get('nl', '')
            gold = doc.metadata.get('gold', '')
            if nl and gold:
                examples.append(f"/* Example */\n/* Question: {nl} */\n/* SQL: */\n{gold}")

        if examples:
```

```
            return "/* some examples are provided */\n" + "\n\n".join(examples) + "\n\n"
        return ""
    except Exception as e:
        print(f"Error retrieving examples: {e}")
        return ""

def get_feedback_few_shot_examples(target_nlq, n_shots=3):
    if not vectorstore or n_shots <= 0:
        return ""
    try:
        # Just use NLQ to find similar examples, as per notebook logic which uses vectorstore_feedback
        # Note: In notebook, vectorstore_feedback is created from examples that have feedback.
        # Here we assume 'vectorstore' has feedback metadata if available.
        docs = vectorstore.similarity_search(target_nlq, k=n_shots)
        examples = []
        for doc in docs:
            nl = doc.metadata.get('nl', '')
            gold = doc.metadata.get('gold', '')
            fb = doc.metadata.get('feedback', '')

            if nl and gold and fb:
                # Format matches notebook: \nExample Question: {nl}\nExample Feedback:{feedback}\nExample Answer: {
                examples.append(f"\nExample Question: {nl}\nExample Feedback:{fb}\nExample Answer: {gold}")

        if examples:
            return "/* some examples are provided */\n" + "".join(examples) + "\n\n"
        return ""
    except Exception as e:
        print(f"Error retrieving feedback examples: {e}")
        return ""
```

```
Loaded 272 records.
```

Method 1: Baseline with Self-Correction (Aligned)

In [38]:
```
# Method 1: Baseline with Self-Correction (Aligned)
```

In [39]:
```
def run_m1_sample(args):
    idx, row, model, max_rounds, n_shots = args
```

```
    nlq = row['nl']
    gold_sql = row['sql']
    db_name = row['target_db'] if 'target_db' in row else row['db_id']
    schema = get_schema(db_name)
    db_path = get_db_path(db_name)

    if not db_path: return None

    print(f"Processing [{idx}]: {nlq}")

    # Initial Generation
    initial_prompt = ""
    meta = build_metadata_constraints(nlq, schema)
    if n_shots > 0:
        examples_str = get_few_shot_examples(nlq, n_shots)
        initial_prompt = f"{examples_str}/* Given the following database schema: */\n{schema}\n{meta}\n/* Answer th
    else:
        initial_prompt = f"/* Given the following database schema: */\n{schema}\n{meta}\n/* Answer the following wi

    sql, _ = LLM_generation(initial_prompt, model=model)
    sql = clean_query(sql)

    # Initial Execution & Fix
    is_correct, errors = evalfunc(sql, gold_sql, db_path)
    syntax_fix = False
    if not is_correct and errors:
            print(f"  [{idx}] ❌ Execution Error (Initial): {errors[0]}")
            invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
            sql, _ = LLM_generation(invalid_prompt, model=model)
            sql = clean_query(sql)
            print(f"  [{idx}] 🔧 Fixed SQL (Initial): {sql}")
            is_correct, errors = evalfunc(sql, gold_sql, db_path)
            if is_correct:
                syntax_fix = True

    sqls_history = [sql]

    if is_correct:
        print(f"  [{idx}] ✅ Initial SQL Correct.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': 0, 'is_correct': True, 'syntax_fix': syntax_fix}
```

```
    print(f"  [{idx}] ❌ Initial SQL Incorrect. Starting Self-Correction...")
    success = False

    for round_i in range(max_rounds):
        sqls_str = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        prompt = sql_generation_selfdebug.format(schema=schema, sqls=sqls_str, question=nlq, metadata="")

        # Few-Shot for Self-Debug
        if n_shots > 0 and len(selfdebug_few_shot) >= 1:
            idx_shot = min(n_shots, len(selfdebug_few_shot)) - 1
            if idx_shot < 0: idx_shot = 0
            prompt = fewshot_prefix + selfdebug_few_shot[idx_shot] + prompt

        sql, _ = LLM_generation(prompt, model=model)
        sql = clean_query(sql)
        sqls_history.append(sql)

        # Verify
        is_correct, errors = evalfunc(sql, gold_sql, db_path)

            # If error, fix it immediately
        if not is_correct and errors:
            print(f"    [{idx}] ❌ Execution Error: {errors[0]}")
            invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
            fixed_sql, _ = LLM_generation(invalid_prompt, model=model)
            fixed_sql = clean_query(fixed_sql)
            print(f"    [{idx}] 🔧 Fixed SQL: {fixed_sql}")

            sqls_history.pop()
            sqls_history.append(fixed_sql)
            sql = fixed_sql

            is_correct, errors = evalfunc(sql, gold_sql, db_path)
            if is_correct:
                syntax_fix = True

        if is_correct:
            print(f"    [{idx}] ✅ Success!")
            success = True
            return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': round_i+1, 'is_correct': True, 'syntax_fix':
```

```
    if not success:
        print(f"    [{idx}] ❌ Failed after max rounds.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': max_rounds, 'is_correct': False, 'syntax_fix': F

def run_simple_feedback_experiment(samples, df_full, model='gpt-3.5-turbo', max_rounds=6, n_shots=0):
    print(f"--- Running Method 1: Baseline (Self-Correction) on {len(samples)} samples (n_shots={n_shots}) in Paral
    results = []

    args_list = []
    for idx, row in samples.iterrows():
        args_list.append((idx, row, model, max_rounds, n_shots))

    with concurrent.futures.ThreadPoolExecutor(max_workers=20) as executor:
        futures = {executor.submit(run_m1_sample, args): args[0] for args in args_list}

        for future in concurrent.futures.as_completed(futures):
            try:
                res = future.result()
                if res:
                    results.append(res)
            except Exception as e:
                print(f"Error in parallel execution: {e}")

    return pd.DataFrame(results)
```

Method 2: Sphinteract (Aligned)

In [40]:
```
# Method 2: Sphinteract (Aligned)
```

In [41]:
```
def run_m2_sample(args):
    idx, row, model, max_rounds, n_shots = args
    nlq = row['nl']
    gold_sql = row['sql']
    db_name = row['target_db'] if 'target_db' in row else row['db_id']
    schema = get_schema(db_name)
    db_path = get_db_path(db_name)

    if not db_path: return None
```

```python
    print(f"Processing [{idx}]: {nlq}")

    # Initial Generation
    initial_prompt = ""
    meta = build_metadata_constraints(nlq, schema)
    if n_shots > 0:
        examples_str = get_few_shot_examples(nlq, n_shots)
        initial_prompt = f"{examples_str}/* Given the following database schema: */\n{schema}\n{meta}\n/* Answer th
    else:
        initial_prompt = f"/* Given the following database schema: */\n{schema}\n{meta}\n/* Answer the following wi

    sql, _ = LLM_generation(initial_prompt, model=model)
    sql = clean_query(sql)

    # Initial Fix Invalid
    is_correct, errors = evalfunc(sql, gold_sql, db_path)
    syntax_fix = False
    if not is_correct and errors:
            print(f"  [{idx}] ❌ Execution Error (Initial): {errors[0]}")
            invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
            sql, _ = LLM_generation(invalid_prompt, model=model)
            sql = clean_query(sql)
            print(f"  [{idx}] 🔧 Fixed SQL (Initial): {sql}")
            is_correct, errors = evalfunc(sql, gold_sql, db_path)
            if is_correct:
                syntax_fix = True

    if is_correct:
        print(f"  [{idx}] ✅ Initial SQL Correct.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': 0, 'is_correct': True, 'syntax_fix': syntax_fix}

    print(f"  [{idx}] ❌ Initial SQL Incorrect. Starting Sphinteract...")

    sqls_history = [sql]
    cqas_history = []

    success = False

    for round_i in range(max_rounds):
        # 1. Generate Clarification Question (SRA)
        cqas_str = ""
```

```
        for i in range(0, len(cqas_history), 2):
            if i+1 < len(cqas_history):
                cqas_str += f"multiple choice clarification question: {cqas_history[i]}\n"
                cqas_str += f"user: {cqas_history[i+1]}\n"
        if not cqas_str:
            cqas_str = "no previous clarification question.\n"

        sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        cq_prompt = SRA.format(schema=schema, question=nlq, sqls=sqls_unique, cqs=cqas_str)
        cq_prompt = cq_prefix_v1 + cq_prompt

        cq, _ = LLM_generation(cq_prompt, model=model)
        if "mul_choice_cq =" in cq:
            cq = cq.split("mul_choice_cq =")[-1].strip().strip('"')
        elif "mul_choice_cq=" in cq:
            cq = cq.split("mul_choice_cq=")[-1].strip().strip('"')
        elif len(cq.split('\n')) < 5:
            pass
        else:
            lines = cq.strip().split('\n')
            cq = lines[-1]

        # print(f"    [{idx}] ❓ CQ: {cq}")

        # 2. Simulate User Feedback
        feedback_prompt = feedback_v2.format(nlq=nlq, query=gold_sql, question=cq)
        feedback_prompt = feedback_prefix_v1 + feedback_prompt

        feedback, _ = LLM_generation(feedback_prompt, model="gpt-4o")
        if "answer_to_cq =" in feedback:
            feedback = feedback.split("answer_to_cq =")[-1].strip().strip('"')
        elif "answer_to_cq=" in feedback:
            feedback = feedback.split("answer_to_cq=")[-1].strip().strip('"')

        # print(f"    [{idx}] 🗣️ Feedback: {feedback}")

        cqas_history.append(cq)
        cqas_history.append(feedback)

        # 3. Generate New SQL
        cqas_block = ""
```

```
        for i in range(0, len(cqas_history), 2):
            if i+1 < len(cqas_history):
                cqas_block += f"multiple choice clarification question: {cqas_history[i]}\n"
                cqas_block += f"user: {cqas_history[i+1]}\n"
        if not cqas_block:
            cqas_block = "no previous clarification questions are asked.\n"

        sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        meta = build_metadata_constraints(nlq, schema)
        sql_prompt = sql_generation_v2.format(schema=schema, question=nlq, sqls=sqls_unique, cqas=cqas_block, metad

        if n_shots > 0:
            fs_ex = get_feedback_few_shot_examples(nlq, n_shots)
            sql_prompt = fs_ex + sql_prompt

        sql, _ = LLM_generation(sql_prompt, model=model)
        sql = clean_query(sql)
        # print(f"    [{idx}] 📝 New SQL: {sql}")
        sqls_history.append(sql)

        # 4. Verify & Fix if needed
        is_correct, errors = evalfunc(sql, gold_sql, db_path)

        if not is_correct and errors:
                # print(f"    [{idx}] ❌ Execution Error: {errors[0]}")
                invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0
                fixed_sql, _ = LLM_generation(invalid_prompt, model=model)
                fixed_sql = clean_query(fixed_sql)
                # print(f"    [{idx}] 🔧 Fixed SQL: {fixed_sql}")

                sqls_history.pop()
                sqls_history.append(fixed_sql)
                sql = fixed_sql

                is_correct, errors = evalfunc(sql, gold_sql, db_path)
                if is_correct:
                    syntax_fix = True

        if is_correct:
            print(f"    [{idx}] ✅ Success!")
            success = True
```

```
            return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': round_i+1, 'is_correct': True, 'syntax_fix':

    if not success:
        print(f"    [{idx}] ❌ Failed after max rounds.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': max_rounds, 'is_correct': False, 'syntax_fix': F

def run_sphinteract_experiment_seq(samples, df_full, model='gpt-3.5-turbo', max_rounds=6, n_shots=0):
    print(f"--- Running Method 2: Sphinteract on {len(samples)} samples (n_shots={n_shots}) ---")
    results = []

    for idx, row in samples.iterrows():
        nlq = row['nl']
        gold_sql = row['sql']
        db_name = row['target_db'] if 'target_db' in row else row['db_id']
        schema = get_schema(db_name)
        db_path = get_db_path(db_name)

        if not db_path: continue

        print(f"\\n{'='*60}")
        print(f"Processing [{idx}]: {nlq}")
        print(f"{'='*60}")

        # Initial Generation
        meta = build_metadata_constraints(nlq, schema)
        if n_shots > 0:
            examples_str = get_few_shot_examples(nlq, n_shots)
            initial_prompt = f"{examples_str}/* Given the following database schema: */\n{schema}\n{meta}\n/* Answe
        else:
            initial_prompt = f"/* Given the following database schema: */\n{schema}\n{meta}\n/* Answer the followin

        sql, _ = LLM_generation(initial_prompt, model=model)
        sql = clean_query(sql)

        # Initial Fix Invalid
        is_correct, errors = evalfunc(sql, gold_sql, db_path)
        syntax_fix = False
        if not is_correct and errors:
             print(f"  ❌ Execution Error (Initial): {errors[0]}")
             invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
             sql, _ = LLM_generation(invalid_prompt, model=model)
```

```
            sql = clean_query(sql)
            print(f"  🔧 Fixed SQL (Initial): {sql}")
            is_correct, errors = evalfunc(sql, gold_sql, db_path)
            if is_correct:
                syntax_fix = True

        sqls_history = [sql]
        cqas_history = [] # Stores alternating CQ and User Answer

        if is_correct:
            print("  ✅ Initial SQL Correct.")
            results.append({'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': 0, 'is_correct': True, 'syntax_fix':
            continue

        print(f"  ❌ Initial SQL Incorrect. Starting Interaction...")
        success = False

        for round_i in range(max_rounds):
            print(f"\\n  --- Round {round_i+1} ---")

            # 1. Generate Clarification Question (SRA)
            # Construct cqas string
            cqas_str = ""
            for i in range(0, len(cqas_history), 2):
                if i+1 < len(cqas_history):
                    cqas_str += f"multiple choice clarification question: {cqas_history[i]}\n"
                    cqas_str += f"user: {cqas_history[i+1]}\n"
            if not cqas_str:
                cqas_str = "no previous clarification question.\n"

            sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
            cq_prompt = SRA.format(schema=schema, question=nlq, sqls=sqls_unique, cqs=cqas_str)
            cq_prompt = cq_prefix_v1 + cq_prompt

            cq, _ = LLM_generation(cq_prompt, model=model)
            if "mul_choice_cq =" in cq:
                cq = cq.split("mul_choice_cq =")[-1].strip().strip('"')
            elif "mul_choice_cq=" in cq:
                cq = cq.split("mul_choice_cq=")[-1].strip().strip('"')
            # Fallback: if no specific format found, use the whole response if it's short, or last line
            elif len(cq.split('\n')) < 5:
```

```
            pass
        else:
            # Try to extract the last sentence or question
            lines = cq.strip().split('\n')
            cq = lines[-1]

        print(f"    ❓ CQ: {cq}")

        # 2. Simulate User Feedback
        feedback_prompt = feedback_v2.format(nlq=nlq, query=gold_sql, question=cq)
        feedback_prompt = feedback_prefix_v1 + feedback_prompt

        feedback, _ = LLM_generation(feedback_prompt, model="gpt-4o")
        if "answer_to_cq =" in feedback:
            feedback = feedback.split("answer_to_cq =")[-1].strip().strip('"')
        elif "answer_to_cq=" in feedback:
            feedback = feedback.split("answer_to_cq=")[-1].strip().strip('"')

        print(f"    🗣️ Feedback: {feedback}")

        cqas_history.append(cq)
        cqas_history.append(feedback)

        # 3. Generate New SQL
        # Re-construct cqas string for SQL prompt (format slightly different in notebook?)
        # Notebook: "multiple choice clarification question: ...\nuser: ...\n"
        # It seems consistent.
        cqas_block = ""
        for i in range(0, len(cqas_history), 2):
            if i+1 < len(cqas_history):
                cqas_block += f"multiple choice clarification question: {cqas_history[i]}\n"
                cqas_block += f"user: {cqas_history[i+1]}\n"
        if not cqas_block:
            cqas_block = "no previous clarification questions are asked.\n"

        sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        meta = build_metadata_constraints(nlq, schema)
        sql_prompt = sql_generation_v2.format(schema=schema, question=nlq, sqls=sqls_unique, cqas=cqas_block, m

        # Few-Shot for Interaction
        if n_shots > 0:
```

```
                fs_ex = get_feedback_few_shot_examples(nlq, n_shots)
                sql_prompt = fs_ex + sql_prompt

            sql, _ = LLM_generation(sql_prompt, model=model)
            sql = clean_query(sql)
            print(f"    📝 New SQL: {sql}")
            sqls_history.append(sql)

            # 4. Verify & Fix if needed
            is_correct, errors = evalfunc(sql, gold_sql, db_path)

            if not is_correct and errors:
                print(f"    ❌ Execution Error: {errors[0]}")
                invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[
                fixed_sql, _ = LLM_generation(invalid_prompt, model=model)
                fixed_sql = clean_query(fixed_sql)
                print(f"    🔧 Fixed SQL: {fixed_sql}")

                # Replace invalid SQL with fixed SQL in history
                sqls_history.pop()
                sqls_history.append(fixed_sql)
                sql = fixed_sql

                is_correct, errors = evalfunc(sql, gold_sql, db_path)
                if is_correct:
                    syntax_fix = True

            if is_correct:
                print("    ✅ Success!")
                success = True
                results.append({'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': round_i+1, 'is_correct': True, '
                break
            else:
                print("    ❌ Still Incorrect.")

        if not success:
            print("    ❌ Failed after max rounds.")
            results.append({'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': max_rounds, 'is_correct': False, 'sy

    return pd.DataFrame(results)
```

```python
def run_sphinteract_experiment(samples, df_full, model='gpt-3.5-turbo', max_rounds=6, n_shots=0):
    print(f"--- Running Method 2: Sphinteract on {len(samples)} samples (n_shots={n_shots}) in Parallel ---")
    results = []

    args_list = []
    for idx, row in samples.iterrows():
        args_list.append((idx, row, model, max_rounds, n_shots))

    with concurrent.futures.ThreadPoolExecutor(max_workers=20) as executor:
        futures = {executor.submit(run_m2_sample, args): args[0] for args in args_list}

        for future in concurrent.futures.as_completed(futures):
            try:
                res = future.result()
                if res:
                    results.append(res)
            except Exception as e:
                print(f"Error in parallel execution: {e}")

    return pd.DataFrame(results)
```

Method 3: Break No Ambiguity (Aligned)

In [42]:
```python
# Method 3: Break No Ambiguity (Aligned)
```

In [43]:
```python
def run_m3_sample(args):
    idx, row, model, max_rounds, n_shots = args
    nlq = row['nl']
    gold_sql = row['sql']
    db_name = row['target_db'] if 'target_db' in row else row['db_id']
    schema = get_schema(db_name)
    db_path = get_db_path(db_name)

    if not db_path: return None

    print(f"Processing [{idx}]: {nlq}")

    # Initial Generation
    if n_shots > 0:
```

```
        examples_str = get_few_shot_examples(nlq, n_shots)
        initial_prompt = f"{examples_str}/* Given the following database schema: */\n{schema}\n/* Answer the follow
    else:
        initial_prompt = f"/* Given the following database schema: */\n{schema}\n/* Answer the following with no ex

    sql, _ = LLM_generation(initial_prompt, model=model)
    sql = clean_query(sql)

    # Initial Fix
    is_correct, errors = evalfunc(sql, gold_sql, db_path)
    syntax_fix = False
    if not is_correct and errors:
         print(f"  [{idx}] ❌ Execution Error (Initial): {errors[0]}")
         invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
         sql, _ = LLM_generation(invalid_prompt, model=model)
         sql = clean_query(sql)
         print(f"  [{idx}] 🔧 Fixed SQL (Initial): {sql}")
         is_correct, errors = evalfunc(sql, gold_sql, db_path)
         if is_correct:
             syntax_fix = True

    sqls_history = [sql]
    cqas_history = []

    if is_correct:
        print(f"  [{idx}] ✅ Initial SQL Correct.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': 0, 'is_correct': True, 'syntax_fix': syntax_fix}

    print(f"  [{idx}] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...")
    success = False

    for round_i in range(max_rounds):
        # 1. Generate CQ with Early Stopping (SRA_ES)
        cqas_str = ""
        for i in range(0, len(cqas_history), 2):
            if i+1 < len(cqas_history):
                cqas_str += f"multiple choice clarification question: {cqas_history[i]}\n"
                cqas_str += f"user: {cqas_history[i+1]}\n"
        if not cqas_str:
            cqas_str = "no previous clarification question.\n"
```

```
        sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        cq_prompt = SRA_ES.format(schema=schema, question=nlq, sqls=sqls_unique, cqs=cqas_str)
        cq_prompt = cq_prefix_v1 + cq_prompt

        cq, _ = LLM_generation(cq_prompt, model=model)

        # Check for Early Stopping
        if "NO AMBIGUITY" in cq:
            print(f"    [{idx}] 🛑 Detected NO AMBIGUITY. Stopping early.")
            break

        if "mul_choice_cq =" in cq:
            cq = cq.split("mul_choice_cq =")[-1].strip().strip('"')
        elif "mul_choice_cq=" in cq:
            cq = cq.split("mul_choice_cq=")[-1].strip().strip('"')
        elif len(cq.split('\n')) < 5:
            pass
        else:
            lines = cq.strip().split('\n')
            cq = lines[-1]

        # print(f"    [{idx}] ❓ CQ: {cq}")

        # 2. Feedback
        feedback_prompt = feedback_v2.format(nlq=nlq, query=gold_sql, question=cq)
        feedback_prompt = feedback_prefix_v1 + feedback_prompt
        feedback, _ = LLM_generation(feedback_prompt, model="gpt-4o")
        if "answer_to_cq =" in feedback:
            feedback = feedback.split("answer_to_cq =")[-1].strip().strip('"')
        elif "answer_to_cq=" in feedback:
            feedback = feedback.split("answer_to_cq=")[-1].strip().strip('"')

        # print(f"    [{idx}] 🗣️ Feedback: {feedback}")
        cqas_history.append(cq)
        cqas_history.append(feedback)

        # 3. New SQL
        cqas_block = ""
        for i in range(0, len(cqas_history), 2):
            if i+1 < len(cqas_history):
                cqas_block += f"multiple choice clarification question: {cqas_history[i]}\n"
```

```
                cqas_block += f"user: {cqas_history[i+1]}\n"
        if not cqas_block:
            cqas_block = "no previous clarification questions are asked.\n"

        sqls_unique = ";\n".join(sorted(list(set(sqls_history)), key=lambda x: sqls_history.index(x)))
        meta = build_metadata_constraints(nlq, schema)
        sql_prompt = sql_generation_v2.format(schema=schema, question=nlq, sqls=sqls_unique, cqas=cqas_block, metad

        if n_shots > 0:
            fs_ex = get_feedback_few_shot_examples(nlq, n_shots)
            sql_prompt = fs_ex + sql_prompt

        sql, _ = LLM_generation(sql_prompt, model=model)
        sql = clean_query(sql)
        # print(f"    [{idx}] 📝 New SQL: {sql}")
        sqls_history.append(sql)

        # 4. Verify & Fix
        is_correct, errors = evalfunc(sql, gold_sql, db_path)

        if not is_correct and errors:
             # print(f"    [{idx}] ❌ Execution Error: {errors[0]}")
             invalid_prompt = fix_invalid_v1.format(schema=schema, question=nlq, invalidSQL=sql, ex=str(errors[0]))
             fixed_sql, _ = LLM_generation(invalid_prompt, model=model)
             fixed_sql = clean_query(fixed_sql)
             # print(f"    [{idx}] 🔧 Fixed SQL: {fixed_sql}")

             sqls_history.pop()
             sqls_history.append(fixed_sql)
             sql = fixed_sql

             is_correct, errors = evalfunc(sql, gold_sql, db_path)
             if is_correct:
                 syntax_fix = True

        if is_correct:
            print(f"    [{idx}] ✅ Success!")
            success = True
            return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': round_i+1, 'is_correct': True, 'syntax_fix':

    if not success:
```

```
        print(f"    [{idx}] ❌ Failed after max rounds.")
        return {'id': idx, 'nlq': nlq, 'final_sql': sql, 'rounds': max_rounds, 'is_correct': False, 'syntax_fix': F

def run_break_no_ambiguity_experiment(samples, df_full, model='gpt-3.5-turbo', max_rounds=6, n_shots=0):
    print(f"--- Running Method 3: Break No Ambiguity on {len(samples)} samples (n_shots={n_shots}) in Parallel ---"
    results = []

    args_list = []
    for idx, row in samples.iterrows():
        args_list.append((idx, row, model, max_rounds, n_shots))

    with concurrent.futures.ThreadPoolExecutor(max_workers=20) as executor:
        futures = {executor.submit(run_m3_sample, args): args[0] for args in args_list}

        for future in concurrent.futures.as_completed(futures):
            try:
                res = future.result()
                if res:
                    results.append(res)
            except Exception as e:
                print(f"Error in parallel execution: {e}")

    return pd.DataFrame(results)
```

Visualization

In [44]:
```
# Visualization & Main
```

Execution

In [45]:
```
def visualize_results(res_m1_zero, res_m1_few, res_m2_zero, res_m2_few, res_m3_zero, res_m3_few, test_subset=None):
    print("\n--- Visualization of Results ---")

    # 1. Construct unified data structure & Save to JSON
    data_map = {
        'M1_Zero': res_m1_zero, 'M1_Few': res_m1_few,
        'M2_Zero': res_m2_zero, 'M2_Few': res_m2_few,
```

```
    'M3_Zero': res_m3_zero, 'M3_Few': res_m3_few
}

json_data = []
for key, df in data_map.items():
    if df.empty: continue
    method, mode = key.split('_')
    records = df.to_dict(orient='records')
    for r in records:
        r['Method'] = method
        r['Mode'] = mode
        # Calculate status for stacked bar
        if r['is_correct']:
            if r.get('syntax_fix', False):
                r['Status'] = 'Syntax Fix Correct'
            elif r.get('rounds', 0) == 0:
                r['Status'] = 'Initial Correct'
            else:
                r['Status'] = 'Interactive Correct'
        else:
            r['Status'] = 'Incorrect'
    json_data.extend(records)

json_path = 'experiment_results.json'
with open(json_path, 'w') as f:
    json.dump(json_data, f, indent=2, default=str)
print(f"Results saved to {json_path}")

# 2. Read from JSON for plotting (as requested)
print(f"Reading results from {json_path} for plotting...")
with open(json_path, 'r') as f:
    loaded_data = json.load(f)

full_df = pd.DataFrame(loaded_data)

if full_df.empty:
    print("No data to visualize.")
    return

# Set Academic Style
sns.set_theme(style="white")
```

```python
sns.set_context("paper", font_scale=1.2)

# Define Palettes: M1=Blue, M2=Orange, M3=Green
palette_dict = {'M1': '#4E79A7', 'M2': '#F28E2B', 'M3': '#59A14F'}

# --- Figure 1: 2x2 Grid (Accuracy & Rounds) ---
fig1, axes1 = plt.subplots(2, 2, figsize=(14, 10))
fig1.suptitle('Performance Overview', fontsize=16, fontweight='bold')

modes = ['Zero', 'Few']
agg_df = full_df.groupby(['Method', 'Mode']).agg(
    Accuracy=('is_correct', 'mean'),
    Avg_Rounds=('rounds', 'mean'),
    SyntaxFixCount=('syntax_fix', 'sum')
).reset_index()
print("\nAggregated Performance Table (includes SyntaxFixCount):")
print(agg_df.to_string(index=False))

for i, mode in enumerate(modes):
    # Top Row: Accuracy
    ax_acc = axes1[0, i]
    subset = agg_df[agg_df['Mode'] == mode]
    if not subset.empty:
        sns.barplot(data=subset, x='Method', y='Accuracy', palette=palette_dict, ax=ax_acc)
        ax_acc.set_title(f'Accuracy ({mode}-shot)')
        ax_acc.set_ylim(0, 1.1)
        ax_acc.set_ylabel('Success Rate' if i==0 else '')
        for p in ax_acc.patches:
            ax_acc.annotate(f'{p.get_height():.2f}', (p.get_x() + p.get_width() / 2., p.get_height()),
                            ha='center', va='center', xytext=(0, 5), textcoords='offset points', fontweight='bo

    # Bottom Row: Rounds
    ax_rds = axes1[1, i]
    if not subset.empty:
        sns.barplot(data=subset, x='Method', y='Avg_Rounds', palette=palette_dict, ax=ax_rds)
        ax_rds.set_title(f'Avg Interaction Rounds ({mode}-shot)')
        ax_rds.set_ylabel('Rounds' if i==0 else '')
        # Add values on top
        max_h = 0
        for p in ax_rds.patches:
            h = p.get_height()
```

```
            if h > max_h: max_h = h
            ax_rds.annotate(f'{h:.2f}', (p.get_x() + p.get_width() / 2., h),
                            ha='center', va='bottom', fontsize=10, fontweight='bold')
        ax_rds.set_ylim(0, max_h * 1.2 if max_h > 0 else 1)

plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()

# --- Figure 2: Stacked Bar Chart (Correctness Breakdown) ---
breakdown_list = []
for (method, mode), group in full_df.groupby(['Method', 'Mode']):
    total = len(group)
    initial = len(group[group['Status'] == 'Initial Correct'])
    interactive = len(group[group['Status'] == 'Interactive Correct'])
    syntaxfix = len(group[group['Status'] == 'Syntax Fix Correct'])
    incorrect = len(group[group['Status'] == 'Incorrect'])

    breakdown_list.append({'Method': method, 'Mode': mode, 'Type': 'Initial Correct', 'Prop': initial/total})
    breakdown_list.append({'Method': method, 'Mode': mode, 'Type': 'Interactive Correct', 'Prop': interactive/t
    breakdown_list.append({'Method': method, 'Mode': mode, 'Type': 'Syntax Fix Correct', 'Prop': syntaxfix/tota
    breakdown_list.append({'Method': method, 'Mode': mode, 'Type': 'Incorrect', 'Prop': incorrect/total})

breakdown_df = pd.DataFrame(breakdown_list)
status_palette = {'Initial Correct': '#1f77b4', 'Interactive Correct': '#2ca02c', 'Syntax Fix Correct': '#17bec

if not breakdown_df.empty:
    fig2, axes2 = plt.subplots(1, 2, figsize=(14, 6))
    fig2.suptitle('Correctness Breakdown', fontsize=16, fontweight='bold')

    for i, mode in enumerate(modes):
        ax = axes2[i]
        subset = breakdown_df[breakdown_df['Mode'] == mode]
        if not subset.empty:
            pivot_df = subset.pivot(index='Method', columns='Type', values='Prop').fillna(0)
            # Ensure all columns exist
            for col in ['Initial Correct', 'Interactive Correct', 'Syntax Fix Correct', 'Incorrect']:
                if col not in pivot_df.columns: pivot_df[col] = 0
            pivot_df = pivot_df[['Initial Correct', 'Interactive Correct', 'Syntax Fix Correct', 'Incorrect']]

            pivot_df.plot(kind='bar', stacked=True, color=[status_palette[c] for c in pivot_df.columns], ax=ax)
            ax.set_title(f'Breakdown ({mode}-shot)')
```

```
                ax.set_ylabel('Proportion' if i==0 else '')
                ax.set_ylim(0, 1.0)
                if i == 1:
                    ax.legend(title='Status', bbox_to_anchor=(1.05, 1), loc='upper left')
                else:
                    ax.get_legend().remove()

        plt.tight_layout(rect=[0, 0.03, 1, 0.95])
        plt.show()

    # --- Figure 3: Difficulty Distribution ---
    if test_subset is not None and 'difficulty' in test_subset.columns:
        plt.figure(figsize=(6, 6))
        diff_counts = test_subset['difficulty'].value_counts()
        colors_map = {'Hard': '#ff9999', 'Medium': '#ffff99', 'Simple': '#66b3ff'}
        pie_colors = [colors_map.get(l, '#cccccc') for l in diff_counts.index]

        plt.pie(diff_counts, labels=diff_counts.index, autopct='%1.1f%%', startangle=140, colors=pie_colors)
        plt.title('Test Sample Difficulty Distribution', fontsize=14, fontweight='bold')
        plt.tight_layout()
        plt.show()



def select_test_samples(df, simple_n=5, medium_n=5, hard_n=30):
    # This function is replaced by ambiguity filtering
    return select_ambiguous_samples(df, target_n=30)

def select_ambiguous_samples(df, target_n=30):
    print(f"--- Selecting {target_n} Ambiguous Samples using GPT-3.5-Turbo ---")

    if 'sql_tok_len' not in df.columns:
        print("Calculating SQL token lengths...")
        df['sql_tok_len'] = df['sql'].apply(lambda x: len(x.split()))

    df['len'] = df['sql_tok_len']
    df_sorted = df.sort_values('len')

    # Define thresholds for difficulty (based on full dataset)
    n = len(df)
    t1 = n // 3
```

```
    t2 = 2 * n // 3

    simple_pool = df_sorted.iloc[:t1]
    medium_pool = df_sorted.iloc[t1:t2]
    hard_pool = df_sorted.iloc[t2:]

    def get_difficulty(length):
        if length >= hard_pool['len'].min(): return 'Hard'
        elif length >= medium_pool['len'].min(): return 'Medium'
        else: return 'Simple'

    selected_indices = []

    # Shuffle df to get random candidates
    df_shuffled = df.sample(frac=1, random_state=42).reset_index(drop=True)

    def check_ambiguity_row(row_tuple):
        idx, row = row_tuple
        nlq = row['nl']
        db_name = row['target_db'] if 'target_db' in row else row['db_id']
        schema = get_schema(db_name)

        prompt = f"""/* Given the following database schema: */
{schema}
/* And the following Natural Language Question: */
{nlq}

/* Task: Determine if the question is ambiguous given the schema.
   Ambiguity can arise from:
   - AmbQuestion: The question phrasing is unclear.
   - AmbTableColumn: Unclear mapping to tables/columns.
   - AmbOutput: Unclear what columns to output.
   - AmbValue: Unclear predicate values.

   Answer "Yes" if the question is ambiguous, or "No" if it is clear.
   Provide a brief reason.
*/
Is the question ambiguous? Answer: """
        # print(f"Checking sample {idx}...")
        amb_model = os.getenv("AMBIGUITY_MODEL", "gpt-4o-mini")
        response, _ = LLM_generation(prompt, model=amb_model, retries=5, retry_delay=2.0, log_each_retry=False)
```

```
    resp = response.strip()
    if resp.startswith("SELECT * FROM error") or resp.lower().startswith("error") or "bad gateway" in resp.lowe
        return idx, None, nlq
    is_ambiguous = False
    u = resp.upper()
    if u.startswith("YES") or "YES" in u[:5]:
        is_ambiguous = True
    return idx, is_ambiguous, nlq

# Process in batches until target_n is reached
candidates = list(df_shuffled.iterrows())
total_candidates = len(candidates)
current_idx = 0
workers_env = os.getenv("AMBIGUITY_WORKERS", "8")
try:
    max_workers = max(1, int(workers_env))
except Exception:
    max_workers = 8
batch_size = max_workers

print(f"Searching for {target_n} ambiguous samples from {total_candidates} candidates...")

while len(selected_indices) < target_n and current_idx < total_candidates:
    batch_end = min(current_idx + batch_size, total_candidates)
    batch_candidates = candidates[current_idx:batch_end]

    print(f"Checking batch {current_idx}-{batch_end} (Found: {len(selected_indices)}/{target_n})...")

    with concurrent.futures.ThreadPoolExecutor(max_workers=max_workers) as executor:
        futures = {executor.submit(check_ambiguity_row, c): c[0] for c in batch_candidates}

        error_count = 0
        reached = False
        for future in concurrent.futures.as_completed(futures):
            try:
                idx, is_ambiguous, nlq = future.result()
                if is_ambiguous is True:
                    print(f"  [Ambiguous] {nlq}")
                    selected_indices.append(idx)
                    if len(selected_indices) >= target_n:
```

```
                        reached = True
                        print(f"Reached target {target_n}; early stop.")
                        break
                elif is_ambiguous is False:
                    print(f"  [Clear] {nlq}")
                else:
                    print(f"  [Skipped: LLM error] {nlq}")
                    error_count += 1
            except Exception as e:
                print("Error processing sample: LLM failure; skipped")
                error_count += 1
        if error_count > (batch_size * 0.3):
            print("High LLM error rate in batch; pausing 10s before next batch...")
            time.sleep(10)
        if reached:
            break

    current_idx += batch_size

selected_indices = selected_indices[:target_n]
if len(selected_indices) < target_n:
    need = target_n - len(selected_indices)
    selected_set = set(selected_indices)
    hard_fill = [i for i in hard_pool.index.tolist() if i not in selected_set]
    medium_fill = [i for i in medium_pool.index.tolist() if i not in selected_set]
    simple_fill = [i for i in simple_pool.index.tolist() if i not in selected_set]
    filler = []
    for i in hard_fill:
        if len(filler) >= need:
            break
        filler.append(i)
    if len(filler) < need:
        for i in medium_fill:
            if len(filler) >= need:
                break
            filler.append(i)
    if len(filler) < need:
        for i in simple_fill:
            if len(filler) >= need:
                break
            filler.append(i)
```

```
        selected_indices = selected_indices + filler[:need]

    if not selected_indices:
        print("Warning: No ambiguous samples found. Returning random samples.")
        result = df.sample(target_n)
    else:
        result = df_shuffled.loc[selected_indices].copy()

    # Assign difficulty labels
    result['difficulty'] = result['len'].apply(get_difficulty)

    print(f"Selected {len(result)} samples.")
    # Print difficulty distribution
    print(result['difficulty'].value_counts())

    # Plot difficulty distribution
    plt.figure(figsize=(6, 6))
    diff_counts = result['difficulty'].value_counts()
    colors_map = {'Hard': '#ff9999', 'Medium': '#ffff99', 'Simple': '#66b3ff'}
    pie_colors = [colors_map.get(l, '#cccccc') for l in diff_counts.index]

    plt.pie(diff_counts, labels=diff_counts.index, autopct='%1.1f%%', startangle=140, colors=pie_colors)
    plt.title('Ambiguous Sample Difficulty Distribution', fontsize=14, fontweight='bold')
    plt.tight_layout()
    plt.show()

    return result

def generate_notebook():
    print("\n--- Generating Notebook (reproduction_sphinteract_ambiguity_generated.ipynb) ---")
    nb = nbf.v4.new_notebook()

    text_intro = """# Reproduction of Sphinteract on KaggleDBQA (Ambiguity Filtered)

This notebook reproduces the three methods described in the paper "Sphinteract: ...":
1.  **Baseline (Method 1)**: Zero-shot generation with Self-Correction (Fix Invalid).
2.  **Sphinteract (Method 2)**: Interactive framework with Clarification Questions (SRA) and Feedback.
3.  **Break No Ambiguity (Method 3)**: Similar to Method 2 but with an Early Stopping mechanism.

We use the **KaggleDBQA** dataset.
```

```
**Modification**: Instead of random sampling, we filter for 20 "Ambiguous" samples using GPT-3.5-Turbo.

## Requirements
- `openai`
- `pandas`
- `sqlite3`
- `numpy`
- `python-dotenv`
- `tiktoken`
- `xxhash`

Ensure you have a `.env` file with `OPENAI_API_KEY` and `OPENAI_BASE_URL` (optional).
"""
    nb.cells.append(nbf.v4.new_markdown_cell(text_intro))

    # Read current file
    try:
        with open(__file__, 'r') as f:
            content = f.read()
    except NameError:
        # Fallback if __file__ is not defined (e.g. interactive)
        content = open('reproduce_sphinteract_ambiguity.py', 'r').read()

    # Split by separators
    separator = "# " + "-" * 78
    parts = content.split(separator)

    # Imports
    nb.cells.append(nbf.v4.new_code_cell(parts[0].strip()))

    for part in parts[1:]:
        if not part.strip(): continue

        lines = part.strip().split('\n')

        # Heuristic: If it's the main block, maybe mark it?
        if "if __name__ == \"__main__\":" in part:
             nb.cells.append(nbf.v4.new_markdown_cell("## Execution"))
        elif "Helper Functions" in lines[0]:
             nb.cells.append(nbf.v4.new_markdown_cell("## Helper Functions"))
        elif "Prompts" in lines[0]:
```

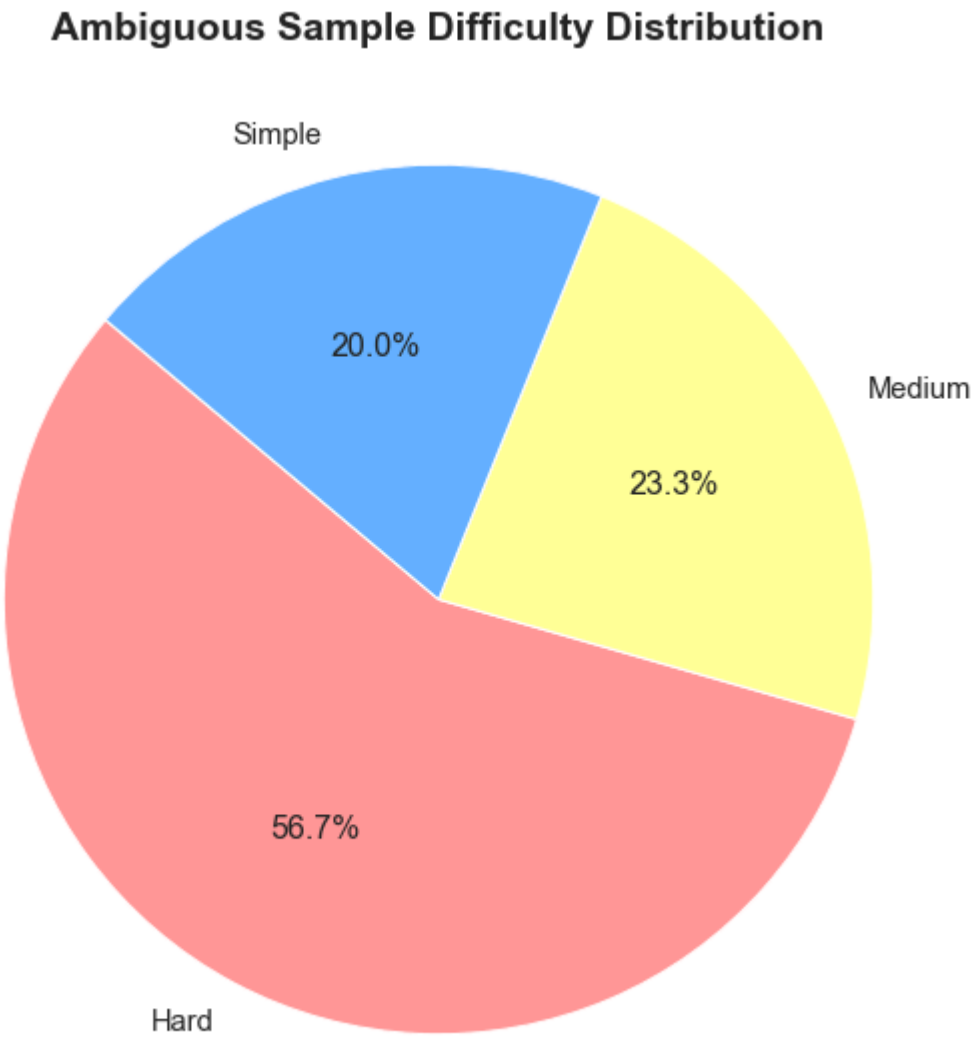
```
            nb.cells.append(nbf.v4.new_markdown_cell("## Prompts"))
        elif "Method" in lines[0]:
            nb.cells.append(nbf.v4.new_markdown_cell(f"## {lines[0].replace('# ', '').strip()}"))
        elif "Visualization" in lines[0]:
            nb.cells.append(nbf.v4.new_markdown_cell("## Visualization"))

        nb.cells.append(nbf.v4.new_code_cell(part.strip()))

    with open('reproduction_sphinteract_ambiguity_generated.ipynb', 'w') as f:
        nbf.write(nb, f)
    print("Notebook reproduction_sphinteract_ambiguity_generated.ipynb generated successfully.")

if __name__ == "__main__":
    # Generate the notebook representation of this script
    generate_notebook()

    # 1. Select Shared Test Samples
    # Modified sample selection as per user request: Simple=5, Medium=5, Hard=30
    test_subset = select_test_samples(df)

    if len(test_subset) == 0:
        print("Warning: No samples found. Using first 30 samples.")
        test_subset = df.head(30)

    # 2. Run Experiments
    print("\\n=== Method 1: Baseline (Zero-Shot) ===")
    res_m1_zero = run_simple_feedback_experiment(test_subset, df, n_shots=0)

    print("\\n=== Method 1: Baseline (Few-Shot) ===")
    res_m1_few = run_simple_feedback_experiment(test_subset, df, n_shots=3)

    print("\\n=== Method 2: Sphinteract (Zero-Shot) ===")
    res_m2_zero = run_sphinteract_experiment(test_subset, df, n_shots=0)
    print("\\n=== Method 2: Sphinteract (Few-Shot) ===")
    res_m2_few = run_sphinteract_experiment(test_subset, df, n_shots=3)

    print("\\n=== Method 3: Break No Ambiguity (Zero-Shot) ===")
    res_m3_zero = run_break_no_ambiguity_experiment(test_subset, df, n_shots=0)

    print("\\n=== Method 3: Break No Ambiguity (Few-Shot) ===")
    res_m3_few = run_break_no_ambiguity_experiment(test_subset, df, n_shots=3)
```

```
# 3. Visualize
visualize_results(res_m1_zero, res_m1_few, res_m2_zero, res_m2_few, res_m3_zero, res_m3_few, test_subset=test_s
```

```
--- Generating Notebook (reproduction_sphinteract_ambiguity_generated.ipynb) ---
Notebook reproduction_sphinteract_ambiguity_generated.ipynb generated successfully.
--- Selecting 30 Ambiguous Samples using GPT-3.5-Turbo ---
Searching for 30 ambiguous samples from 272 candidates...
Checking batch 0-8 (Found: 0/30)...
  [Ambiguous] How much did the federal government spend in No Child Left Behind funding in 2017?
  [Ambiguous] What is the average age of players from USA?
  [Clear] Average date of year that fire was discovered from 2000~2004?
  [Clear] Which matches has the highest draw opening so far?
  [Ambiguous] What are the most likely outcome of the police investigation if the crime happen on "street"?
  [Ambiguous] What is the average Title 1 fund in Virginia?
  [Clear] What was the most common cause of fire between 2000 and 2005?
  [Clear] Who is the highest paid player since 2010?
Checking batch 8-16 (Found: 4/30)...
  [Clear] How many total acres of land in Texas have seen a wildfire in the decade between 2000-2010?
  [Clear] Where is the first BWR type power plant built and located?
  [Ambiguous] Rank the country of product origins in terms of pesticide residues detection.
  [Clear] Which state has the most wildfires?
  [Clear] which foods are captured in the data set?
  [Ambiguous] What’s the year that have the largest acres in the fire area?
  [Ambiguous] which entry have been downloaded the least?
  [Clear] What's the unit of measure for sample 3879?
Checking batch 16-24 (Found: 7/30)...
  [Clear] Which country lead the total capacity of the power plants it held?
  [Clear] State with highest average math score
  [Clear] Where is the place with the largest number of sexual offenses crime events?
  [Ambiguous] What is the top league that pays the most to their players?
  [Clear] Which year has the most released song?
  [Ambiguous] When was the last instance of a violent or sexual offense in Manchester?
  [Clear] Which are the top 10 commodities that have the highest residue during 2015?
  [Ambiguous] Which school district receive the most of federal revenue through state in Wisconsin?
Checking batch 24-32 (Found: 10/30)...
  [Clear] How many crimes has been conducted?
  [Ambiguous] What are the top 100 single musics?
  [Clear] Which city the most players were born?
  [Ambiguous] What were the years when any special elections happened in hall of fame?
  [Clear] Which counties in Washington had fires in 2012?
  [Clear] On what type of land (public or private) do more wildfires occur?
  [Clear] How many crimes are still "Under investigation" to date?
  [Ambiguous] What are the birth places of players won on hall of fame since 1871?
Checking batch 32-40 (Found: 13/30)...
```

```
  [Clear] How many league division does football_data database has?
  [Clear] Which reactor type has the largest average capacity?
  [Ambiguous] Name the most popular and least popular releases of lasean camry?
  [Clear] What's the odds for draw on Bet365 for the game Swindon v.s. Millwall for 2016/2017 season?
  [Clear] How many number of units are there in sample 9628?
  [Ambiguous] Do other leagues have referee name records outside of Scotland and England?
  [Clear] what year was each specific entry released?
  [Clear] What is the average pay for players inducted into the hall of fame?
Checking batch 40-48 (Found: 15/30)...
  [Ambiguous] Which birth place has the most player awards?
  [Clear] What's the most common extraction method?
  [Ambiguous] How many matches did Pinnacle have betting records?
  [Ambiguous] Which neighborhood/area has the highest burglary rate?
  [Clear] For every award, who is the oldest winner?
  [Clear] Which Premier League matches ended in a draw in 2016?
  [Clear] Which states had the largest number of fires in 2001?
  [Clear] How many Wisconsin school districts receive federal funding?
Checking batch 48-56 (Found: 18/30)...
  [Ambiguous] how was a specific sample tested?
  [Ambiguous] What's the code for confirmation for the latest sample?
  [Clear] How many nuclear power plants are in preparation to be used in Japan?
  [Clear] What is the total area that has been burned until now?
  [Clear] Which type of crime happen the most in Salford?
  [Clear] Which country has only one nuclear power plants?
  [Ambiguous] For award winners, which position that has the most hall of fame players?
  [Ambiguous] Which is the most popular voting method for Hall of Fame in 2000?
Checking batch 56-64 (Found: 22/30)...
  [Ambiguous] Which leage has higher average salaries for player?
  [Clear] In 2014, how many wildfires were the result of mismanaged campfires?
  [Ambiguous] show all imported samples?
  [Clear] How many years of data are recorded in this database?
  [Clear] How many players were awarded more than ten times?
  [Clear] Which state has the highest average score in math exam?
  [Clear] Which year had the largest number of fires?
  [Ambiguous] For award winners, what's average weight for each position
Checking batch 64-72 (Found: 25/30)...
  [Clear] Which state has the most number of fires being recorded?
  [Ambiguous] Which country first started using nuclear power plant(s)?
  [Clear] What are the planed nuclear power plants and their located countries?
  [Clear] What is the main source of the information for this table?
  [Clear] If sample 6480 is imported, which country is it originally from?
```

```
  [Clear] What are name of top 10 artists or groups?
  [Ambiguous] Top 10 teams with the most hall of fame players
  [Clear] Find me top 10 albums ranked by their popularity.
Checking batch 72-80 (Found: 27/30)...
  [Clear] How many operating nuclear station in France?
  [Clear] What’s the most common cause of the fire (code) in the database?
  [Clear] How many kinds of nuclear reactor model in the world?
  [Clear] Who was responsible for the land of the biggest fire in Oregon in 2015?
  [Clear] Which year has the most wildfires?
  [Ambiguous] Which state spent the least revenue towards schools and whats the state average score
  [Ambiguous] What is the top 3 area of crime conducted?
  [Clear] Which kind of release type is the most popular?
Checking batch 80-88 (Found: 29/30)...
  [Ambiguous] What are the Pinnacle odds for Arsenal winning matches?
Reached target 30; early stop.
Selected 30 samples.
difficulty
Hard      17
Medium     7
Simple     6
Name: count, dtype: int64
```


Ambiguous Sample Difficulty Distribution
Simple
20.0%
Medium
23.3%
56.7%
Hard

```
\n=== Method 1: Baseline (Zero-Shot) ===
--- Running Method 1: Baseline (Self-Correction) on 30 samples (n_shots=0) in Parallel ---
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [5]: What is the average age of players from USA?
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [15]: What’s the year that have the largest acres in the fire area?
Processing [14]: which entry have been downloaded the least?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [53]: What's the code for confirmation for the latest sample?
  [29] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [38] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [41] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [15] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [20] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [31] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [14] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [26] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [37] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [53] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [22] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [6] ✅ Initial SQL Correct.
```

```
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
  [7] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [8] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [51] ❌ Execution Error (Initial): near "annotated": syntax error
  [17] ✅ Initial SQL Correct.
  [1] ✅ Initial SQL Correct.
  [40] ✅ Initial SQL Correct.
  [45] ✅ Initial SQL Correct.
  [5] ✅ Initial SQL Correct.
  [50] ✅ Initial SQL Correct.
  [55] ✅ Initial SQL Correct.
    [14] ✅ Success!
  [62] ✅ Initial SQL Correct.
  [70] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): WITH results AS (
  SELECT *
  FROM resultsdata15
  WHERE annotate = 'annotated' AND quantitate = 'quantitated'
)
SELECT *
FROM results
  [51] ✅ Initial SQL Correct.
    [20] ✅ Success!
  [57] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
  [60] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
  [79] ✅ Initial SQL Correct.
    [38] ❌ Failed after max rounds.
    [37] ❌ Failed after max rounds.
\n=== Method 1: Baseline (Few-Shot) ===
--- Running Method 1: Baseline (Self-Correction) on 30 samples (n_shots=3) in Parallel ---
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [5]: What is the average age of players from USA?
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [14]: which entry have been downloaded the least?
```

```
Processing [15]: What’s the year that have the largest acres in the fire area?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [53]: What's the code for confirmation for the latest sample?
  [26] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [6] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [1] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [15] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [53] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [20] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [5] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [29] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [31] ✅ Initial SQL Correct.
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
  [8] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [41] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [17] ✅ Initial SQL Correct.
  [45] ✅ Initial SQL Correct.
  [7] ✅ Initial SQL Correct.
  [14] ✅ Initial SQL Correct.
  [37] ✅ Initial SQL Correct.
  [40] ✅ Initial SQL Correct.
```

```
  [38] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [22] ✅ Initial SQL Correct.
  [51] ❌ Execution Error (Initial): no such column: specific_sample_pk
  [55] ✅ Initial SQL Correct.
  [60] ✅ Initial SQL Correct.
  [70] ✅ Initial SQL Correct.
  [62] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): SELECT
  confmethod,
  confmethod2,
  annotate,
  quantitate,
  mean,
  extract,
  determin
FROM resultsdata15
WHERE sample_pk = ?
  [51] ❌ Initial SQL Incorrect. Starting Self-Correction...
  [50] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
  [57] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
    [51] ❌ Execution Error: Incorrect number of bindings supplied. The current statement uses 1, and there are 0 su
pplied.
  [79] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
    [51] 🔧 Fixed SQL: SELECT
  confmethod,
  confmethod2,
  annotate,
  quantitate,
  mean,
  extract,
  determin
FROM resultsdata15
WHERE sample_pk = 1
    [51] ✅ Success!
    [20] ❌ Failed after max rounds.
    [38] ❌ Failed after max rounds.
\n=== Method 2: Sphinteract (Zero-Shot) ===
--- Running Method 2: Sphinteract on 30 samples (n_shots=0) in Parallel ---
```

```
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [5]: What is the average age of players from USA?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [14]: which entry have been downloaded the least?
Processing [15]: What’s the year that have the largest acres in the fire area?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [53]: What's the code for confirmation for the latest sample?
  [6] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [38] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [26] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [20] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [14] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [1] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [31] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [53] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [17] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [37] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [51] ❌ Execution Error (Initial): near "annotated": syntax error
  [22] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [29] ✅ Initial SQL Correct.
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
```

```
  [15] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [7] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [41] ✅ Initial SQL Correct.
  [45] ✅ Initial SQL Correct.
  [5] ❌ Execution Error (Initial): no such function: year
  [40] ✅ Initial SQL Correct.
  [8] ✅ Initial SQL Correct.
  [62] ✅ Initial SQL Correct.
  [60] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): WITH results AS (
  SELECT *
  FROM resultsdata15
  WHERE annotate = 'annotated' AND quantitate = 'quantitated'
)
SELECT *
FROM results
  [55] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
  [51] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
  [57] ✅ Initial SQL Correct.
  [70] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
  [79] ✅ Initial SQL Correct.
  [5] 🔧 Fixed SQL (Initial): SELECT AVG((strftime('%Y', 'now') - CAST(birth_year AS INTEGER))) AS average_age_usa_p
layers
FROM player
WHERE birth_country = 'USA' AND birth_year IS NOT NULL AND birth_year != ''
  [5] ✅ Initial SQL Correct.
  [50] ✅ Initial SQL Correct.
    [14] ✅ Success!
    [20] ✅ Success!
    [37] ✅ Success!
    [38] ✅ Success!
\n=== Method 2: Sphinteract (Few-Shot) ===
--- Running Method 2: Sphinteract on 30 samples (n_shots=3) in Parallel ---
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [5]: What is the average age of players from USA?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
```

```
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [15]: What’s the year that have the largest acres in the fire area?
Processing [14]: which entry have been downloaded the least?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [53]: What's the code for confirmation for the latest sample?
  [15] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [6] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [31] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [20] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [38] ❌ Initial SQL Incorrect. Starting Sphinteract...
  [37] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [26] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [41] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [29] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [53] ✅ Initial SQL Correct.
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
  [45] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [5] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [7] ✅ Initial SQL Correct.
  [14] ✅ Initial SQL Correct.
```

```
  [22] ✅ Initial SQL Correct.
  [1] ✅ Initial SQL Correct.
  [51] ❌ Execution Error (Initial): Incorrect number of bindings supplied. The current statement uses 1, and there
are 0 supplied.
  [40] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): SELECT
  lab,
  confmethod,
  confmethod2,
  annotate,
  quantitate,
  extract,
  determin
FROM resultsdata15
WHERE sample_pk = 1
  [51] ✅ Initial SQL Correct.  [50] ✅ Initial SQL Correct.

  [57] ✅ Initial SQL Correct.
  [8] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
  [17] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
  [60] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
  [79] ✅ Initial SQL Correct.
  [70] ✅ Initial SQL Correct.
  [55] ✅ Initial SQL Correct.
  [62] ✅ Initial SQL Correct.
    [38] ✅ Success!
    [20] ✅ Success!
\n=== Method 3: Break No Ambiguity (Zero-Shot) ===
--- Running Method 3: Break No Ambiguity on 30 samples (n_shots=0) in Parallel ---
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [5]: What is the average age of players from USA?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [15]: What's the year that have the largest acres in the fire area?
Processing [14]: which entry have been downloaded the least?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
```

```
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [53]: What's the code for confirmation for the latest sample?
  [6] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [41] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [38] ❌ Execution Error (Initial): incomplete input
  [45] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [15] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [8] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [20] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [14] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [26] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [17] ✅ Initial SQL Correct.
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
  [37] ❌ Execution Error (Initial): near "SELECT": syntax error
  [40] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [29] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [7] ✅ Initial SQL Correct.
  [5] ❌ Execution Error (Initial): no such function: year
  [1] ✅ Initial SQL Correct.
  [22] ✅ Initial SQL Correct.
  [51] ❌ Execution Error (Initial): Incorrect number of bindings supplied. The current statement uses 1, and there
are 0 supplied.
  [53] ✅ Initial SQL Correct.
```

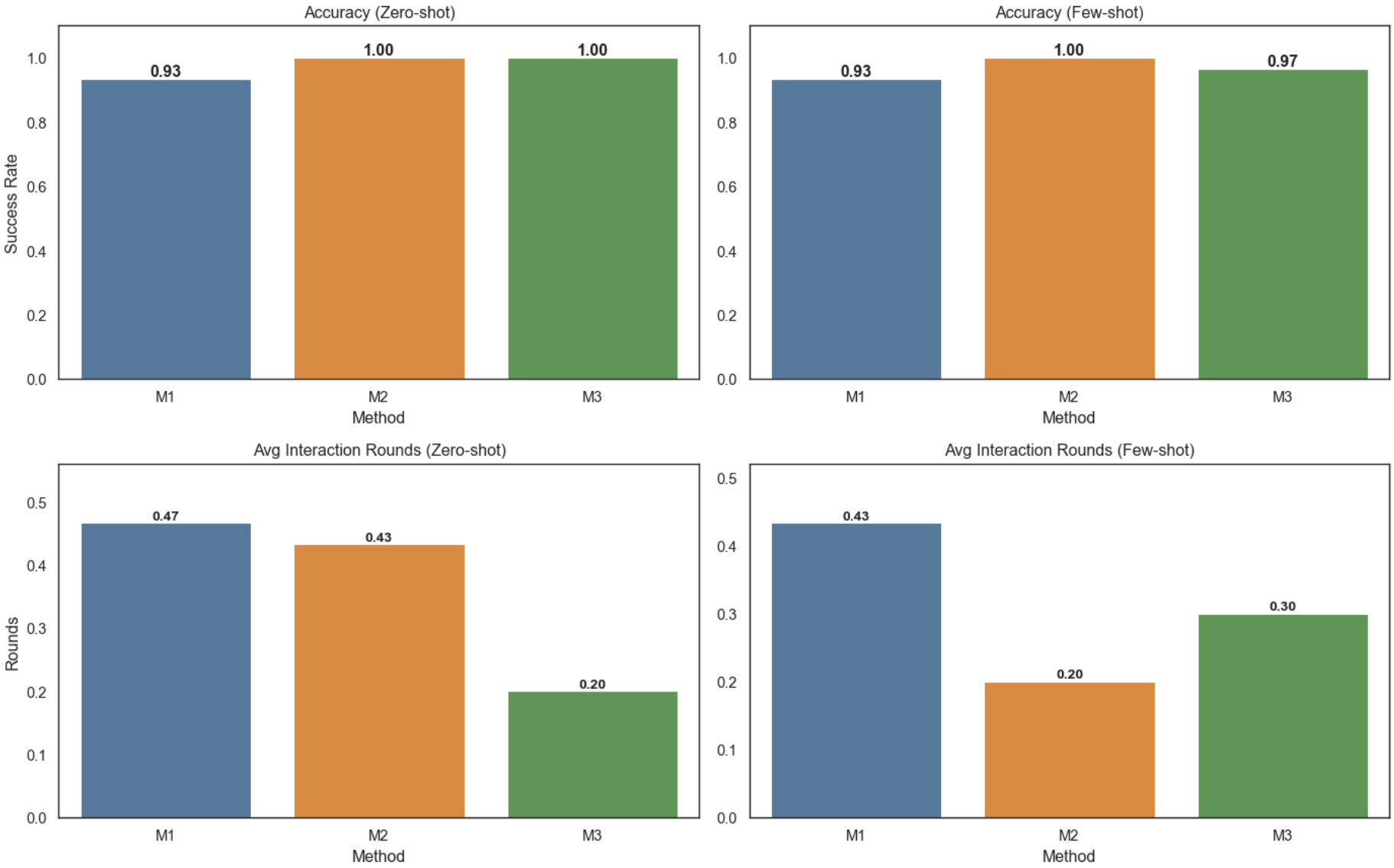
```
  [31] ✅ Initial SQL Correct.
  [50] ❌ Execution Error (Initial): no such column: p.position
  [60] ✅ Initial SQL Correct.
  [38] 🔧 Fixed SQL (Initial): SELECT *
FROM betfront
  [55] ✅ Initial SQL Correct.
  [38] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [62] ✅ Initial SQL Correct.
  [37] 🔧 Fixed SQL (Initial): SELECT groupName, MAX(totalSnatched) AS maxSnatched
FROM torrents
WHERE artist = 'Lasean Camry'

-- Least popular release of Lasean Camry
SELECT groupName, MIN(totalSnatched) AS minSnatched
FROM torrents
WHERE artist = 'Lasean Camry'
  [37] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [70] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
  [57] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): SELECT confmethod, confmethod2
FROM resultsdata15
WHERE sample_pk = 1
  [51] ✅ Initial SQL Correct.
  [5] 🔧 Fixed SQL (Initial): SELECT AVG((strftime('%Y', 'now') - CAST(birth_year AS INTEGER))) AS average_age_usa_p
layers
FROM player
WHERE birth_country = 'USA' AND birth_year IS NOT NULL AND birth_year != ''
  [5] ✅ Initial SQL Correct.
  [50] 🔧 Fixed SQL (Initial): SELECT p.birth_country, COUNT(DISTINCT h.player_id) AS hof_count
FROM player_award pa
JOIN hall_of_fame h ON pa.player_id = h.player_id
JOIN player p ON pa.player_id = p.player_id
GROUP BY p.birth_country
ORDER BY hof_count DESC
LIMIT 1
  [50] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
  [79] ✅ Initial SQL Correct.
    [20] ✅ Success!
```

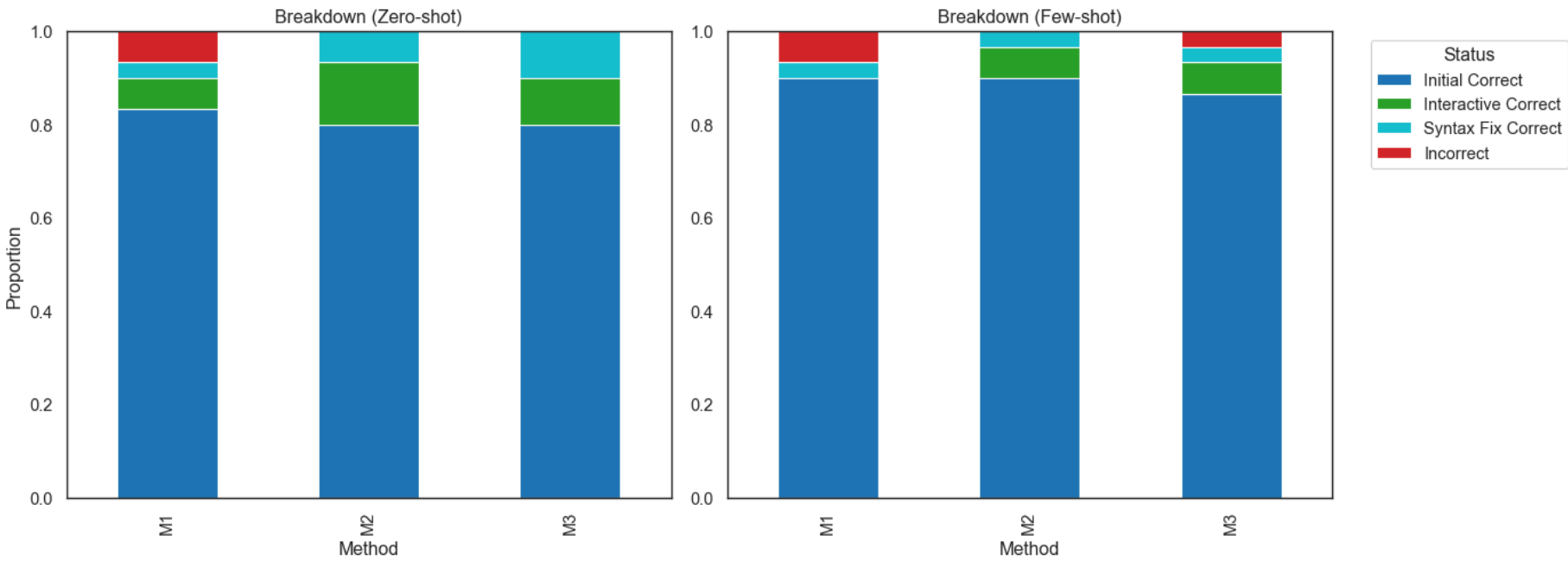
```
    [38] ✅ Success!
    [37] ✅ Success!
\n=== Method 3: Break No Ambiguity (Few-Shot) ===
--- Running Method 3: Break No Ambiguity on 30 samples (n_shots=3) in Parallel ---
Processing [6]: How much did the federal government spend in No Child Left Behind funding in 2017?
Processing [1]: What are the most likely outcome of the police investigation if the crime happen on "street"?
Processing [5]: What is the average age of players from USA?
Processing [8]: Rank the country of product origins in terms of pesticide residues detection.
Processing [7]: What is the average Title 1 fund in Virginia?
Processing [14]: which entry have been downloaded the least?
Processing [15]: What’s the year that have the largest acres in the fire area?
Processing [17]: What is the top league that pays the most to their players?
Processing [20]: When was the last instance of a violent or sexual offense in Manchester?
Processing [22]: Which school district receive the most of federal revenue through state in Wisconsin?
Processing [31]: What are the top 100 single musics?
Processing [29]: What are the birth places of players won on hall of fame since 1871?
Processing [26]: What were the years when any special elections happened in hall of fame?
Processing [37]: Name the most popular and least popular releases of lasean camry?
Processing [38]: Do other leagues have referee name records outside of Scotland and England?
Processing [40]: Which birth place has the most player awards?
Processing [41]: How many matches did Pinnacle have betting records?
Processing [45]: Which neighborhood/area has the highest burglary rate?
Processing [51]: how was a specific sample tested?
Processing [53]: What's the code for confirmation for the latest sample?
  [45] ✅ Initial SQL Correct.
Processing [50]: For award winners, which position that has the most hall of fame players?
  [51] ❌ Execution Error (Initial): Incorrect number of bindings supplied. The current statement uses 1, and there
are 0 supplied.
  [53] ✅ Initial SQL Correct.
Processing [55]: Which is the most popular voting method for Hall of Fame in 2000?
  [8] ✅ Initial SQL Correct.
Processing [57]: Which leage has higher average salaries for player?
  [22] ✅ Initial SQL Correct.
Processing [62]: show all imported samples?
  [41] ✅ Initial SQL Correct.
Processing [60]: For award winners, what's average weight for each position
  [26] ✅ Initial SQL Correct.
Processing [64]: Which country first started using nuclear power plant(s)?
  [1] ✅ Initial SQL Correct.
Processing [70]: Top 10 teams with the most hall of fame players
  [37] ✅ Initial SQL Correct.
```

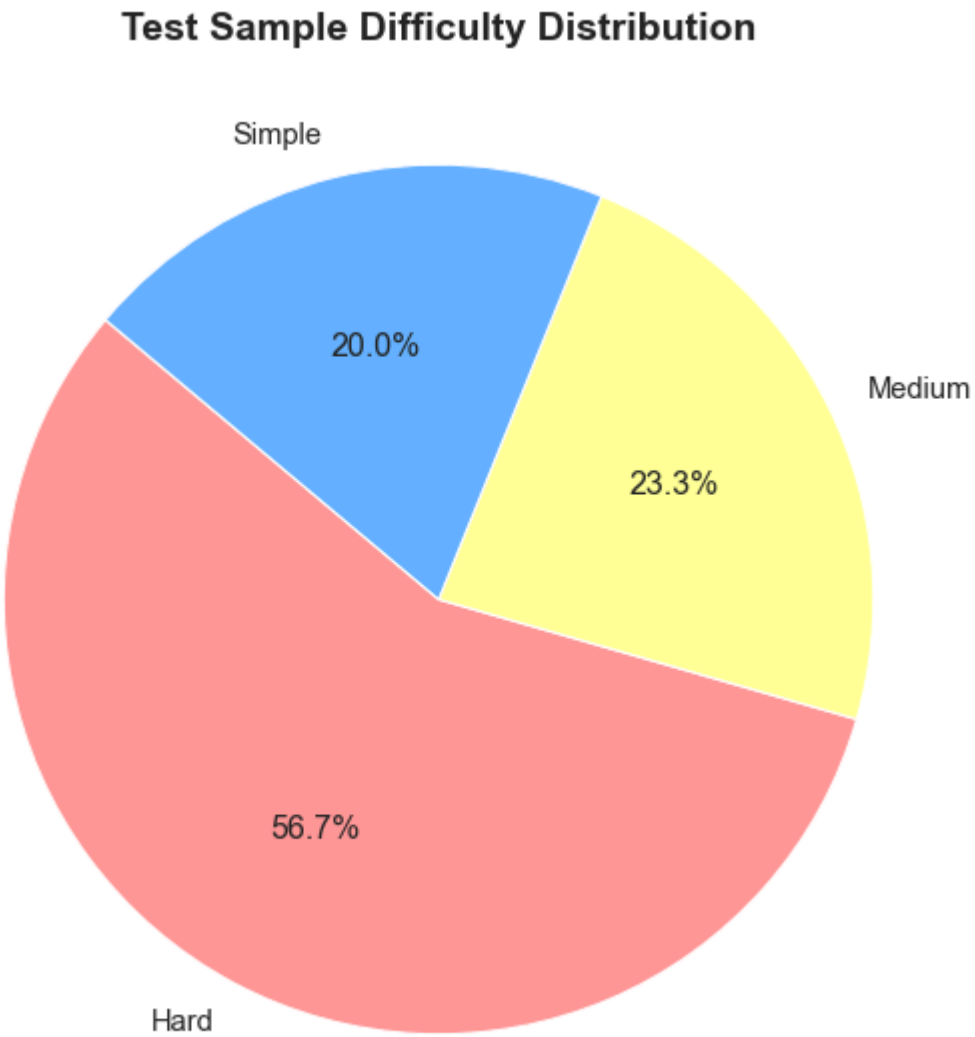
```
Processing [79]: Which state spent the least revenue towards schools and whats the state average score
  [29] ✅ Initial SQL Correct.
Processing [73]: What is the top 3 area of crime conducted?
  [20] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [6] ✅ Initial SQL Correct.
Processing [83]: What are the Pinnacle odds for Arsenal winning matches?
  [40] ✅ Initial SQL Correct.
  [17] ✅ Initial SQL Correct.
  [31] ✅ Initial SQL Correct.
  [5] ✅ Initial SQL Correct.
  [14] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [15] ✅ Initial SQL Correct.
  [51] 🔧 Fixed SQL (Initial): SELECT
  confmethod,
  confmethod2,
  extract,
  determin
FROM resultsdata15
WHERE sample_pk = 1
  [51] ✅ Initial SQL Correct.
  [38] ❌ Initial SQL Incorrect. Starting Interaction (CQs & ES)...
  [57] ✅ Initial SQL Correct.
  [7] ✅ Initial SQL Correct.
  [64] ✅ Initial SQL Correct.
  [55] ✅ Initial SQL Correct.
  [50] ✅ Initial SQL Correct.
  [79] ✅ Initial SQL Correct.
  [62] ✅ Initial SQL Correct.
  [70] ✅ Initial SQL Correct.
  [83] ✅ Initial SQL Correct.
  [60] ✅ Initial SQL Correct.
  [73] ✅ Initial SQL Correct.
    [14] ✅ Success!
    [20] ✅ Success!
    [38] ❌ Failed after max rounds.

--- Visualization of Results ---
Results saved to experiment_results.json
Reading results from experiment_results.json for plotting...

Aggregated Performance Table (includes SyntaxFixCount):
```

```
Method Mode  Accuracy  Avg_Rounds  SyntaxFixCount
    M1  Few  0.933333    0.433333               1
    M1 Zero  0.933333    0.466667               1
    M2  Few  1.000000    0.200000               1
    M2 Zero  1.000000    0.433333               2
    M3  Few  0.966667    0.300000               1
    M3 Zero  1.000000    0.200000               3
```

```
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/2805775401.py:72: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=subset, x='Method', y='Accuracy', palette=palette_dict, ax=ax_acc)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/2805775401.py:83: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=subset, x='Method', y='Avg_Rounds', palette=palette_dict, ax=ax_rds)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/2805775401.py:72: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=subset, x='Method', y='Accuracy', palette=palette_dict, ax=ax_acc)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/2805775401.py:83: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=subset, x='Method', y='Avg_Rounds', palette=palette_dict, ax=ax_rds)
```


Performance Overview
Accuracy (Zero-shot)
Success Rate
0.93
1.00
1.00
M1
M2
M3
Method
Accuracy (Few-shot)
0.93
1.00
0.97
M1
M2
M3
Method
Avg Interaction Rounds (Zero-shot)
Rounds
0.47
0.43
0.20
M1
M2
M3
Method
Avg Interaction Rounds (Few-shot)
0.43
0.20
0.30
M1
M2
M3
Method


Correctness Breakdown
Breakdown (Zero-shot)
Breakdown (Few-shot)
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
M1
M2
M3
Method
Status
Initial Correct
Interactive Correct
Syntax Fix Correct
Incorrect



Beatify Figures

In [62]:
```
def redraw_from_results(json_path='experiment_results.json', save_dir=None):
    import os, json, numpy as np, pandas as pd
```

```python
import matplotlib.pyplot as plt, seaborn as sns
from matplotlib import colors as mcolors

# 亮色生成：提高亮度，不改变透明度
def lighten(c, factor=0.40):
    r, g, b = mcolors.to_rgb(c)
    return (1 - (1 - r) * (1 - factor),
            1 - (1 - g) * (1 - factor),
            1 - (1 - b) * (1 - factor))

# 读取结果
if not os.path.exists(json_path):
    raise FileNotFoundError(json_path)
with open(json_path, 'r') as f:
    loaded = json.load(f)
df = pd.DataFrame(loaded)
if df.empty:
    raise ValueError('Empty results')

# 全局更大字体与符号尺寸
sns.set_theme(style='white')
sns.set_context('talk', font_scale=1.6)  # 比 paper 1.2 大很多
plt.rcParams.update({
    'figure.dpi': 160,
    'axes.titlesize': 18,
    'axes.titleweight': 'bold',
    'axes.labelsize': 14,
    'xtick.labelsize': 12,
    'ytick.labelsize': 12,
    'legend.title_fontsize': 12,
    'legend.fontsize': 12,
    'axes.linewidth': 1.2,
    'lines.linewidth': 2.0,
    'patch.linewidth': 0.8,
    'grid.linewidth': 0.8,
})

# 颜色：Zero-shot 基础色；Few-shot 亮色（透明度不变）
base_palette = {'M1': '#4E79A7', 'M2': '#F28E2B', 'M3': '#59A14F'}
few_palette = {k: lighten(v, 0.40) for k, v in base_palette.items()}
modes = ['Zero', 'Few']
```

```python
# 聚合指标
agg = df.groupby(['Method', 'Mode']).agg(
    Accuracy=('is_correct', 'mean'),
    Avg_Rounds=('rounds', 'mean')
).reset_index()

# 图 1：性能总览（普通柱状图），Few 使用亮色
fig1, axes1 = plt.subplots(2, 2, figsize=(15, 10))
fig1.suptitle('Performance Overview', fontsize=18, fontweight='bold')

for i, mode in enumerate(modes):
    sub = agg[agg['Mode'] == mode]
    pal = base_palette if mode == 'Zero' else few_palette

    # Accuracy
    ax_acc = axes1[0, i]
    if not sub.empty:
        sns.barplot(data=sub, x='Method', y='Accuracy', palette=pal, ax=ax_acc)
        ax_acc.set_title(f'Accuracy ({mode}-shot)')
        ax_acc.set_ylim(0, 1.1)
        ax_acc.set_ylabel('Success Rate' if i == 0 else '')
        for p in ax_acc.patches:
            h = p.get_height()
            ax_acc.annotate(f'{h:.2f}', (p.get_x() + p.get_width()/2., h),
                            ha='center', va='center', xytext=(0, 8), textcoords='offset points',
                            fontsize=12, fontweight='bold')

    # Avg Rounds
    ax_rds = axes1[1, i]
    if not sub.empty:
        sns.barplot(data=sub, x='Method', y='Avg_Rounds', palette=pal, ax=ax_rds)
        ax_rds.set_title(f'Avg Interaction Rounds ({mode}-shot)')
        ax_rds.set_ylabel('Rounds' if i == 0 else '')
        max_h = 0
        for p in ax_rds.patches:
            h = p.get_height()
            max_h = max(max_h, h)
            ax_rds.annotate(f'{h:.2f}', (p.get_x() + p.get_width()/2., h),
                            ha='center', va='bottom', fontsize=12, fontweight='bold')
        ax_rds.set_ylim(0, max_h * 1.25 if max_h > 0 else 1)
```

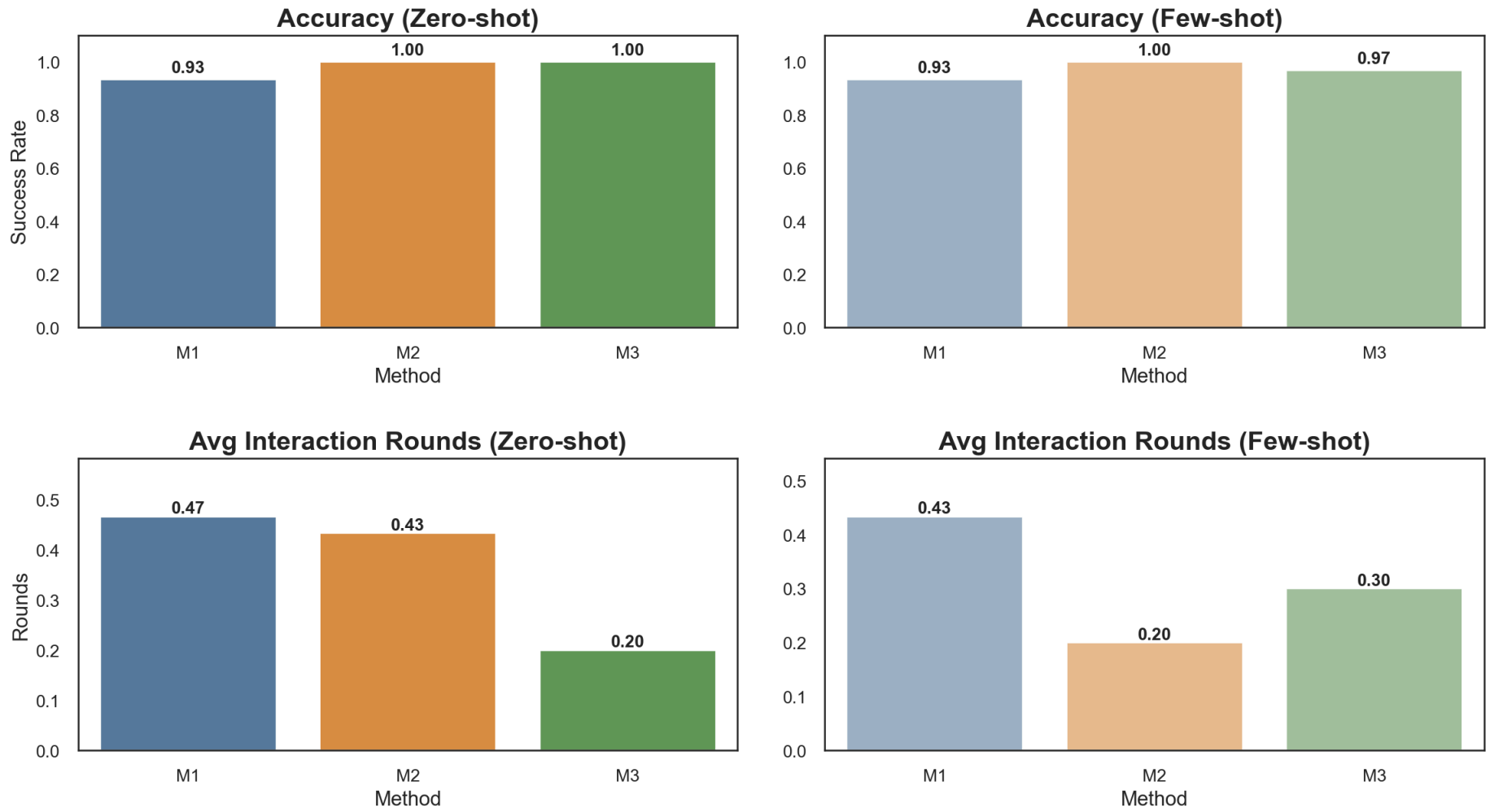
```
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
if save_dir:
    os.makedirs(save_dir, exist_ok=True)
    fig1.savefig(os.path.join(save_dir, 'performance_overview.png'), dpi=160)
plt.show()

# 图 2：正确性分解（累积柱状图，分段比例标签）
status_order = ['Initial Correct', 'Interactive Correct', 'Syntax Fix Correct', 'Incorrect']
status_palette = {'Initial Correct': '#1f77b4', 'Interactive Correct': '#2ca02c',
                  'Syntax Fix Correct': '#17becf', 'Incorrect': '#d62728'}

breakdown = []
for (m, md), grp in df.groupby(['Method', 'Mode']):
    t = len(grp)
    vals = {s: (len(grp[grp['Status'] == s]) / t if t else 0) for s in status_order}
    for s in status_order:
        breakdown.append({'Method': m, 'Mode': md, 'Type': s, 'Prop': vals[s]})
bdf = pd.DataFrame(breakdown)

fig2, axes2 = plt.subplots(1, 2, figsize=(15, 6))
fig2.suptitle('Correctness Breakdown', fontsize=18, fontweight='bold')

methods = ['M1', 'M2', 'M3']
x = np.arange(len(methods))
width = 0.6

for i, mode in enumerate(modes):
    ax = axes2[i]
    sub = bdf[bdf['Mode'] == mode]
    if sub.empty:
        continue
    pivot = sub.pivot(index='Method', columns='Type', values='Prop').reindex(
        index=methods, columns=status_order).fillna(0)

    bottoms = np.zeros(len(methods))
    bars_info = []
    for s in status_order:
        h = pivot[s].values
        b = ax.bar(x, h, width, bottom=bottoms, color=status_palette[s], label=s)
        bars_info.append((s, h, bottoms, b))
```

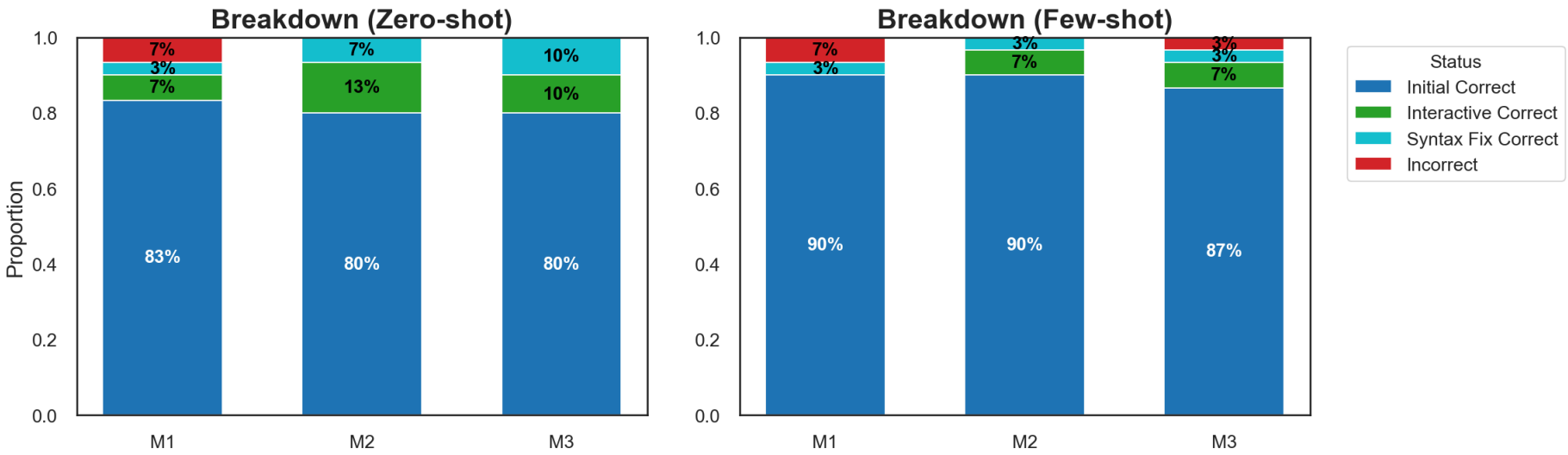
```
            bottoms = bottoms + h

        ax.set_xticks(x)
        ax.set_xticklabels(methods)
        ax.set_ylim(0, 1.0)
        ax.set_ylabel('Proportion' if i == 0 else '')
        ax.set_title(f'Breakdown ({mode}-shot)')

        # 每个分段的比例标签
        for s, h, btm, b in bars_info:
            for j in range(len(h)):
                val = h[j]
                if val > 0.02:
                    y = btm[j] + val / 2
                    lbl = f'{val:.0%}'
                    ax.text(x[j], y, lbl, ha='center', va='center',
                            fontsize=12, fontweight='bold',
                            color='#000000' if val < 0.25 else '#ffffff')

        if i == 1:
            ax.legend(title='Status', bbox_to_anchor=(1.05, 1), loc='upper left')
        else:
            leg = ax.get_legend()
            if leg is not None:
                leg.remove()

    plt.tight_layout(rect=[0, 0.03, 1, 0.95])
    if save_dir:
        os.makedirs(save_dir, exist_ok=True)
        fig2.savefig(os.path.join(save_dir, 'correctness_breakdown.png'), dpi=160)
    plt.show()
```

In [63]:

```
# 确认文件存在与记录数
import os, json
print(os.path.exists('experiment_results.json'))
with open('experiment_results.json') as f:
    print(len(json.load(f)))

# 直接重绘（并保存）
redraw_from_results('experiment_results.json', save_dir='./figs')
```

```
True
180
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/3131208275.py:62: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=sub, x='Method', y='Accuracy', palette=pal, ax=ax_acc)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/3131208275.py:75: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=sub, x='Method', y='Avg_Rounds', palette=pal, ax=ax_rds)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/3131208275.py:62: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=sub, x='Method', y='Accuracy', palette=pal, ax=ax_acc)
/var/folders/wk/v3jtqnys2ts0fyk3q7pq7qrw0000gn/T/ipykernel_72514/3131208275.py:75: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=sub, x='Method', y='Avg_Rounds', palette=pal, ax=ax_rds)
```


Performance Overview
Accuracy (Zero-shot)
Success Rate
0.0
0.2
0.4
0.6
0.8
1.0
0.93
1.00
1.00
M1
M2
M3
Method
Accuracy (Few-shot)
0.93
1.00
0.97
M1
M2
M3
Method
Avg Interaction Rounds (Zero-shot)
Rounds
0.0
0.1
0.2
0.3
0.4
0.5
0.47
0.43
0.20
M1
M2
M3
Method
Avg Interaction Rounds (Few-shot)
0.43
0.20
0.30
M1
M2
M3
Method


Correctness Breakdown
Breakdown (Zero-shot)
Breakdown (Few-shot)
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
M1
M2
M3
7%
3%
7%
83%
7%
13%
80%
7%
10%
10%
80%
7%
3%
90%
3%
7%
90%
3%
3%
7%
87%
Status
Initial Correct
Interactive Correct
Syntax Fix Correct
Incorrect