Results

Experiments with prompt

Experiments #	EM	True	False	Erro
(1) → 3-shot	236	123	123	18
(2) → 3-shot + context	253	107	131	9
(3) → 3-shot + context + table info	293	93	112	2
(4) → (3) + modified prompt	282	103	115	0

Experiments #	True (sum)	False (sum)	EX
(1)	359	141	47.2 %
(2)	360	140	50.6 %
(3)	386	114	**58.6 %**
(4)	385	115	56.4 %

(4) Experiments with number of shots

# shots	EM	True	False	Erro
0-shot	193	171	132	4
3-shot	282	103	115	0
5-shot	287	110	103	0
7-shot	292	105	103	0

# shots	True (sum)	False (sum)	EX
0-shot	364	136	38.6 %
3-shot	385	115	56.4 %
5-shot	397	103	57.4 %
7-shot	397	103	**58.4%**

Length difficulty Unigram Accuracy

# shots	Easy	Hard
3-shot	86.5 %	39.0 %
5-shot	**87.8** %	**46.0 %**
7-shot	**87.8** %	**46.0 %**

Nested difficulty Unigram Accuracy

# shots	Easy	Hard
3-shot	86.1 %	42.9 %
5-shot	**88.4 %**	45.7 %
7-shot	87.3 %	**49.5 %**

(5) Experiment → 5-shot with codes-15b

model	EM	True	False	Erro
7b	287	110	103	0
15b	306	98	96	0

model	True (sum)	False (sum)	EX
7b	397	103	57.4 %
15b	404	96	**61.2 %**

Length difficulty Unigram Accuracy

model	Easy	Hard
7b	87.8 %	**46.0 %**
15b	**90.0 %**	44.0 %

Nested difficulty Unigram Accuracy

model	Easy	Hard
7b	88.4 %	45.7 %
15b	**89.9 %**	**46.7 %**

(6) Experiment → SQLCoder-7b-2

	EX	Unigram
0-shot	27.4 %	59.0 %

Experiments details

Dataset has 100 real questions with corresponding SQL query and answers, and 4 more variants of each one of them, making up 500 questions in total.

Dataset contains `question, true_query, true_answer, generated_query, generated_answer`

I executed the generated query on the database and stored the answer as `generated_answer`

EM → if `true_answer` = `generated_answer`

Error → if the `generated_query` is syntactically wrong

True → if `true_answer` != `generated_answer` and 1-gram overlap ≥ 0.33

False → if `true_answer` != `generated_answer` and 1-gram overlap < 0.33

True (sum) → EM + True

False (sum) → Error + False

I divided the difficulty of the questions into Easy and Hard in 2 ways:

1. **Length difficulty** → according to the length of the `true_query` (100 Hard questions)
2. **Nested difficulty** → Hard if `true_query` requires nested Select statements to answer the question (105 Hard questions)

The 2 tables above show the percentage of correctly answered question.

The columns information passed in the prompt

```
ID -- Primary key
Owner_ID -- Unique ID of each owner of the property
Owner_First_Name -- First name of the owner of the property
Owner_Family_Name -- Family name of the owner of the property
Property_Type -- Specific type of the property given in Italian
Rent_Income -- Rent price of the property that the owner receives as income, given in Venice ancient g
Property_Location -- Ancient approximate toponym of the property given in Italian
```

Experiment 1 → 3-shot

Data: 500 questions

Prompt:

- 3 shot

```
"""table catastici , columns = [ catastici.Owner_First_Name ( text ) , catastici.Owner_Family_Name 
Owner_First_Name -- First name of the owner of the property ; Owner_Family_Name -- Family name of t
{question_1}
{sql_1}
{question_2}
{sql_2}
{question_3}
{sql_3}
{question}
"""
```

Issues:

- Multiple columns in `Count` → difficulty with counting owners
- Sometimes confuses feature names

Experiment 2 → 3-shot + context

Data: 500 questions

- I corrected quite many of the 100 original question, and regenerated the variants
- I also added the `Owner_ID` column

Prompt:

- 3 shot
- with context → If there is any entity name in the question, the we show which column it corresponds to as `context` , for example:
    - question: *How much rental income do properties typically generate in "al ponte di san provolo" on average?*
    - context: *“Property_Location” = “al ponte di san provolo”*

```
"""table catastici , columns = [ catastici.Owner_ID ( integer ) , catastici.Owner_First_Name ( text
Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First name of the owner of
{question_1}
{context_1}
```

```
{sql_1}
{question_2}
{context_2}
{sql_2}
{question_3}
{context_3}
{sql_3}
{question}
{context}
"""
```

Experiment 3 → 3-shot + context + table info

Same as experiment 3 but with the following prompt:

```
"""table catastici , columns = [ catastici.Owner_ID ( integer ) , catastici.Owner_First_Name ( text
Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First name of the owner of
{question_1}
{context_1}
{sql_1}
table catastici , columns = [ catastici.Owner_ID ( integer ) , catastici.Owner_First_Name ( text ) ,
Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First name of the owner of
{question_2}
{context_2}
{sql_2}
table catastici , columns = [ catastici.Owner_ID ( integer ) , catastici.Owner_First_Name ( text ) ,
Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First name of the owner of
{question_3}
{context_3}
{sql_3}
table catastici , columns = [ catastici.Owner_ID ( integer ) , catastici.Owner_First_Name ( text ) ,
Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First name of the owner of
{question}
{context}
"""
```

Experiment 4 → 3 (modified)

Data:

- I added the `ID` column → Primary (unique) key

Prompt:

- Modified the prompt to match how it is used in the paper

```
"""database schema :
table catastici , columns = [ catastici.ID ( integer ) , catastici.Owner_ID ( integer ) , catastici
columns info :
ID -- Primary key ; Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First
primary key :
catastici.ID
matched contents : {context_1}
{question_1}
{sql_1}

database schema :
table catastici , columns = [ catastici.ID ( integer ) , catastici.Owner_ID ( integer ) , catastici
columns info :
ID -- Primary key ; Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First
primary key :
catastici.ID
matched contents : {context_2}
{question_2}
```

```
{sql_2}

database schema :
table catastici , columns = [ catastici.ID ( integer ) , catastici.Owner_ID ( integer ) , catastici
columns info :
ID -- Primary key ; Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First n
primary key :
catastici.ID
matched contents : {context_3}
{question_3}
{sql_3}

database schema :
table catastici , columns = [ catastici.ID ( integer ) , catastici.Owner_ID ( integer ) , catastici
columns info :
ID -- Primary key ; Owner_ID -- Unique ID of each owner of the property; Owner_First_Name -- First n
primary key :
catastici.ID
matched contents : {context}
{question}
"""
```

Experiment 5 → 5-shot with codes-15b

- Same as experiment 4, but with
    - 5-shot
    - codes-15b model

Experiment 6 → sqlcoder-7b-2

- Zero shot experiment with sqlcoder-7b-2
- The prompt

```
"""### Task
Generate a SQL query to answer [QUESTION]{question}[/QUESTION]

### Instructions
- If you cannot answer the question with the available database schema, return 'I do not know'
- Search for exact match with "=", unless specified otherwise

### Database Schema
CREATE TABLE [catastici]
(
    [Property_ID] INTEGER PRIMARY KEY,
    [Owner_ID] INTEGER, -- Unique ID of each owner of the property
    [Owner_First_Name] VARCHAR(30), -- First name of the owner of the property
    [Owner_Family_Name] VARCHAR(30), -- Family name of the owner of the property
    [Property_Type] VARCHAR(100), -- Specific type of the property given in Italian
    [Rent_Income] INTEGER, -- Rent price of the property that the owner receives as income, given in
    [Property_Location] VARCHAR(100) -- Ancient approximate toponym of the property given in Italian
);

### Matched contents
{matched_contents}

### Answer
Given the database schema, here is the SQL query that answers [QUESTION]{question}[/QUESTION]
[SQL]
"""
```

Examples

Non-matching but TRUE queries

1. Can you calculate the cumulative rental income from "casa" property types?

True Answer: 189473

```
SELECT SUM("Rent_Income") as Total_Rent_Income
FROM catastici
WHERE "Property_Type" = 'casa';
```

Generated Answer: (‘casa', 189473)

```
SELECT "Property_Type", SUM("Rent_Income") AS Cu
FROM catastici
WHERE "Property_Type" = 'casa'
GROUP BY "Property_Type";
```

2. Who are the entities listed as owners of properties?

True Answer: List of people

```
SELECT DISTINCT "Owner_First_Name", "Owner_Famil
FROM catastici;
```

Generated Answer: Ordered List of people

```
SELECT "Owner_First_Name", "Owner_Family_Name"
FROM catastici
GROUP BY "Owner_First_Name", "Owner_Family_Name"
ORDER BY "Owner_First_Name", "Owner_Family_Name";
```

Non-matching and FALSE queries

1. In the dataset, is the property with the smallest amount of rent income attributed to "domenico" "michiel"?

True Answer:

```
SELECT "Owner_First_Name", "Owner_Family_Name", '
FROM catastici
WHERE "Owner_First_Name" = 'domenico' AND "Owner_
AND Rent_Income = ( SELECT MIN("Rent_Income")
FROM catastici
ORDER BY "Rent_Income" ASC
LIMIT 1);
```

Generated Answer: 6

```
SELECT MIN("Rent_Income")
FROM catastici
WHERE "Owner_First_Name" = 'domenico' AND "Owner_
```

2. How many types of properties are unique to a certain location?

True Answer: [('preallvo', 3), ('al capitello', 3), ('al capitello vicino la calle larga', 7), ('al gaffaro', 1), ('al paré di tolle', 8), ('al ponte', 1), ...]

```
SELECT "Property_Location", COUNT(DISTINCT "Prope
FROM catastici
GROUP BY "Property_Location";
```

Generated Answer: [('calle del forno', 72)]

```
SELECT "Property_Location", COUNT(DISTINCT "Prope
FROM catastici
GROUP BY "Property_Location"
ORDER BY num_property_types DESC
LIMIT 1;
```

Ambiguous queries

1. What's the interval of rental earnings found on "calle de franchi"?

True Answer: (8, 32)

```
SELECT MIN("Rent_Income"), MAX("Rent_Income")
FROM catastici
WHERE "Property_Location" = 'calle de franchi';
```

Generated Answer: 24

```
SELECT MAX("Rent_Income") - MIN("Rent_Income")
FROM catastici
WHERE "Property_Location" = 'calle de franchi';
```

2. How much income does "carlo" "gritti" generate from rent on his property?

True Answer: 512

Generated Answer: [13, 30, 40, 22, 16, 20, 70, 44, 185, 72]

```
SELECT SUM("Rent_Income")
FROM catastici
WHERE "Owner_First_Name" = 'carlo' AND "Owner_Fa
```

```
SELECT "Rent_Income"
FROM catastici
WHERE "Owner_First_Name" = 'carlo' AND "Owner_Fa
```

3. How much do the top 5 earners generate in total rental income?

True Answer: [(23954,)]

```
SELECT SUM("Rent_Income") AS Total_Rent_Income
FROM catastici
WHERE ("Owner_First_Name", "Owner_Family_Name") I
SELECT "Owner_First_Name", "Owner_Family_Name"
FROM catastici
ORDER BY "Rent_Income" DESC
LIMIT 5);
```

Generated Answer: [('antonio', 'danielli', 15004), ('antonio', 'grimani', 3966), ('michiel', 'morosini', 3808), ('domenico', 'michiel', 3263), ('nicolo', 'venier', 3222)]

```
SELECT "Owner_First_Name", "Owner_Family_Name", S
FROM catastici
GROUP BY "Owner_First_Name", "Owner_Family_Name"
ORDER BY "Total_Rent_Income" DESC
LIMIT 5;
```

Limitations

- May not understand the paraphrased questions, i.e. synonyms.
  - Examples
    - What is the number of households possessing assets across multiple type categories?
    - For each property location, what's the standard rental income?
- When the question refers to income and asks for some aggregation, it may wrongly do it on Rent_Income column.
  - Example
    - What is the total number of properties generating less than 30 ducati in rent? (does sum of rent instead of count)
- Limits the number of outputs, even when not asked specifically
  - Example
    - Which properties have a rent income higher than 50 ducati? (limits to 1)
- When asked for the count, it may return the list, without counting them
  - Example
    - What is the count of unique locations where properties are situated? (returns the list of unique locations)