

Text-to-SQL Project Using Gemini

Ethan Ondreicka | March 19th, 2024

Grading Methodology

0 - 100 scale

10 points for giving a relevant answer

20 points for giving back sql code

20 points for giving back sql code with no errors

50 points for giving the correct answer on first try

20 points for giving the correct answer on second try

10 points for giving the correct answer on ≥ 3 attempts

Background Information

Before I ask Gemini for any sql queries, I will first preface it for what I want to accomplish, and then feed it all of the relevant tables from the CAP Database



Can you give me some sql queries based on a table schema I provide



Absolutely! I can help you write SQL queries based on a table schema you provide. Here's what you can do:



This is my table schema



-- PostgreSQL create, load, and query script for CAP v008....



Absolutely, with this schema, I can help you write various SQL queries. Here are some examples based on your tables:

Question 1

After slightly rewording the
Question 1 prompt, Gemini
was able to solve it easily with
no errors

Grade: 100/100

No fixing required
Results are the same

Using the tables provided above, make a query to display the cities that makes the most different kinds of products. Use the rank() function.

Show drafts ▾ 🔊

Here's the SQL query to display the cities that make the most different kinds of products using the `rank()` function:

```
SQL

SELECT city, COUNT(DISTINCT prodId) AS num_distinct_products,
       RANK() OVER (ORDER BY COUNT(DISTINCT prodId) DESC) AS rank
FROM Products
GROUP BY city
ORDER BY rank;
```

Use code [with caution](#). 📄

Gemini's Results

	city text 🔒	num_distinct_products bigint 🔒	rank bigint 🔒
1	Newark	3	1
2	Dallas	2	2
3	Duluth	2	2
4	Beijing	1	4
5	Montreal	1	4

My Results

Data Output		Messages	Notifications
📄	📄	📄	📄
	city text 🔒	num_products bigint 🔒	city_rank bigint 🔒
1	Newark	3	1
2	Dallas	2	2
3	Duluth	2	2
4	Beijing	1	4
5	Montreal	1	4

Question 2

Just like question 1,


Gemini was easily able to

Give the correct query

Grade 100/100

No fixing needed

Results are the exact same


 Using the provided tables above, Display the names of products whose priceUSD is less than 1% of the average priceUSD in alphabetical order from A - Z

Show drafts ▾ 🔊


✦ Here's the SQL query to display product names with a priceUSD less than 1% of the average priceUSD, ordered alphabetically:

SQL


```
SELECT name
FROM Products
WHERE priceUSD < (
  SELECT AVG(priceUSD) * 0.01
  FROM Products
)
ORDER BY name ASC;
```

Use code [with caution](#). 

Gemini's Results

	name text	
1	Denis Wick Valve Oil	
2	Flux Capacitor	
3	HAL 9000 memory chip	
4	LCARS module	

My Results

	name text	
1	Denis Wick Valve Oil	
2	Flux Capacitor	
3	HAL 9000 memory chip	
4	LCARS module	


Question 3

Gemini struggled quite a bit with question 3. After

multiple tries and rewording,

Gemini was unable to give me

error-free code with the correct solution



Using the provided tables above, display the customer last name, product id ordered, and the totalUSD for all orders made in March of any year, sorted by totalUSD from low to high

Here's the SQL query to display customer last name, product ID ordered, and total USD for orders in March, sorted by totalUSD (low to high):

```
SQL

SELECT c.lastName, o.prodId, o.totalUSD
FROM Customers c
INNER JOIN Orders o ON c.pid = o.custId
WHERE EXTRACT(MONTH FROM o.dateOrdered) = 3
ORDER BY o.totalUSD ASC;
```

Use code [with caution](#).

Gemini's Error

Data Output	Messages	Notifications
ERROR: column cus.lastname does not exist LINE 1: SELECT cus.lastName, o.prodId, o.totalUSD ^		
SQL state: 42703 Character: 8		

Gemini's Error

ERROR: column cus.firstname does not exist LINE 1: SELECT cus.firstName, o.prodId, o.totalUSD ^		
SQL state: 42703 Character: 8		

Gemini's Wrong Result

	customer_id integer	prodid character	totalusd numeric (12,2)
1	1	p05	6550.98
2	1	p06	31236.48
3	5	p01	56671.55

Fixing Question 3

To fix the code, I had to change all references to the customer table and query the lastName from the People table instead

Grade: 50/100

Fixing required, but error free code was eventually given

Results are the exact same with different column names

```
SELECT p.lastName AS people_last_name,  
       o.prodId AS product_id_ordered,  
       SUM(o.totalUSD) AS totalUSD  
FROM Orders o  
JOIN People p ON o.custId = p.pid  
WHERE EXTRACT(MONTH FROM o.dateOrdered) = 3  
GROUP BY p.lastName, o.prodId  
ORDER BY totalUSD ASC;
```

My Results

	customer_last_name text	product_id_ordered character	totalusd numeric
1	Ferguson	p05	6550.98
2	Ferguson	p06	31236.48
3	Armstrong	p01	56671.55

Gemini's Results

	people_last_name text	product_id_ordered character	totalusd numeric
1	Ferguson	p05	6550.98
2	Ferguson	p06	31236.48
3	Armstrong	p01	56671.55

Question 4

Again, Gemini is unable to get the answer correct through the first prompt. It seems to be selecting the customer table instead of the people table.



Using the provided tables above, display the last name of all customers (in reverse alphabetical order), and their total ordered by customer and NOTHING more. Use coalesce to avoid NULL totals.



Sure, here's the SQL query to achieve what you described:

SQL

```
SELECT c.lastName, COALESCE(SUM(o.totalUSD), 0.00) AS total_ordered
FROM Customers c
LEFT JOIN Orders o ON c.pid = o.custId
GROUP BY c.lastName
ORDER BY c.lastName DESC;
```

Use code [with caution](#).



```
ERROR: column c.lastname does not exist
LINE 1: SELECT c.lastName, COALESCE(SUM(o.totalUSD), 0.00) AS total_...
              ^
```

SQL state: 42703

Character: 8

*for question 4, I decided to start a new chat with Gemini, just to keep it's memory fresh

Fixing Question 4

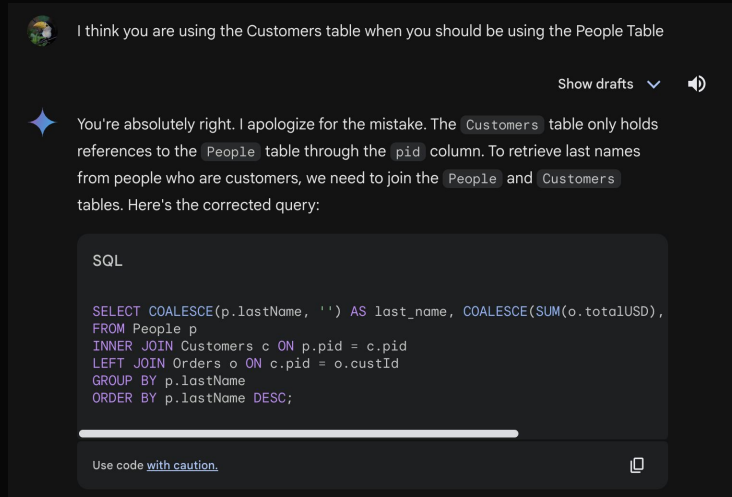
After some quick prompt engineering, Gemini was able to fix the problems and select the right table in the query.

Grade: 70/100

No fixing was required, but needed a little bit of prompt engineering

Results were a little bit different as Gemini's didn't include a lot of the orders where total_ordered = 0

*for question 4, I decided to start a new chat with Gemini, just to keep it's memory fresh



Gemini's Results

	last_name text	total_ordered numeric
1	Severinsen	142768.32
2	Robinson	0.00
3	Morrison	109053.62
4	Ferguson	146525.21
5	Armstrong	72442.75

My Results

	lastname text	total_ordered numeric
1	Skonberg	0
2	Severinsen	142768.32
3	Robinson	0
4	Morrison	109053.62
5	Helseth	0
6	Gillespie	0
7	Ferguson	146525.21
8	Davis	0
9	Armstrong	72442.75

Question 5

Gemini is still unable to get the answer correct through the first prompt. It seems to be selecting the customer table instead of the people table once more.



Using the tables provided above, display the names of all customers who bought products from agents based in Chilliwack, along with the names of the products they ordered, and the names of the agents who sold it to them



Here's the SQL query to find customers who bought products from Chilliwack agents:

SQL

```
SELECT
  cust.lastName AS customer_name,
  prod.name AS product_name,
  agent.lastName AS agent_name
FROM Customers cust
INNER JOIN Orders ord ON cust.pid = ord.custId
INNER JOIN Agents agent ON ord.agentId = agent.pid
INNER JOIN People agent_people ON agent.pid = agent_people.pid
INNER JOIN Products prod ON ord.prodId = prod.prodId
WHERE agent_people.homeCity = 'Chilliwack';
```

Use code [with caution](#).



```
ERROR: column cust.lastname does not exist
LINE 3:   cust.lastName AS customer_name,
          ^
```

```
SQL state: 42703
Character: 12
```

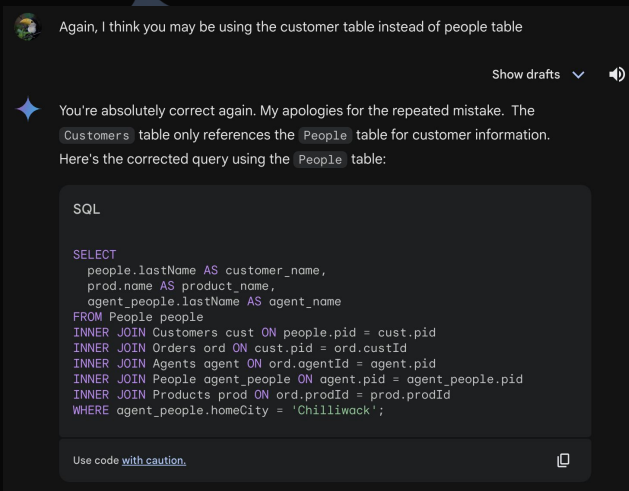
Fixing Question 5

After prompt engineering, Gemini was able to fix the selection problem.

Grade: 70/100

No fixing was required, but needed a little bit of prompt engineering

Results were pretty much the same except I had a column separating first and last name



Gemini's Results

	customer_name text	product_name text	agent_name text
1	Ferguson	Heisenberg Compensator	Skonberg
2	Ferguson	Universal Translator	Skonberg
3	Ferguson	Denis Wick Valve Oil	Skonberg
4	Armstrong	Heisenberg Compensator	Skonberg

My Results

	customer_name text	product_name text	agent_first_name text	agent_last_name text
1	Ferguson	Heisenberg Compensator	Bria	Skonberg
2	Ferguson	Universal Translator	Bria	Skonberg
3	Ferguson	Denis Wick Valve Oil	Bria	Skonberg
4	Armstrong	Heisenberg Compensator	Bria	Skonberg

Question 6

Again, Gemini is unable to get the answer correct through the first prompt. I am not quite sure where it went wrong. Probably selecting the wrong tables or doing the wrong math.

	ordernum [PK] integer	order_total_recorded numeric (12,2)	calculat numeric	
1	1018	8050.49		
2	1025	799.11	888.00	-88.89
3	1022	31236.48	39600.00	-8363.52
4	1016	60977.22	67760.00	-6782.78
5	1015	15771.20	16610.00	-838.80
6	1021	66086.33	67760.00	-1673.67
7	1024	56671.55	59628.80	-2957.25
8	1012	76096.81	78024.00	-1927.19
9	1020	585.18	600.00	-14.82
10	1017	25643.98	32510.00	-6866.02
11	1019	16249.28	20600.00	-4350.72
12	1011	58794.00	74536.00	-15742.00
13	1023	6550.98	8305.00	-1754.02
14	1026	47277.29	52536.16	-5258.87



using the table provided above. Write a query to check the accuracy of the totalUSD column in the orders table. This means calculating Orders.totalUSD from data in other tables and comparing those values to the values in Orders.totalUSD. Display all rows in Orders where Orders.totalUSD are incorrect, if any. If there are any ...



Here's the SQL query to identify discrepancies in the `totalUSD` column of the Orders table:

SQL

```
SELECT
  o.orderNum,
  o.totalUSD AS order_total_recorded,
  ROUND(SUM(p.priceUSD * o.quantityOrdered), 2) AS calculated_total,
  ROUND(o.totalUSD - ROUND(SUM(p.priceUSD * o.quantityOrdered), 2), 2) /
FROM Orders o
INNER JOIN Products p ON o.prodId = p.prodId
GROUP BY o.orderNum, o.totalUSD
HAVING ROUND(o.totalUSD - ROUND(SUM(p.priceUSD * o.quantityOrdered), 2)
```

Use code [with caution](#).



Fixing Question 6

After prompt engineering, Gemini was able to fix the selection problem.

Grade: 60/100

No fixing was required, but needed a few rounds of prompt engineering

Results were formatted differently but still the same

SQL

```
SELECT
  o.orderNum,
  o.totalUSD AS recorded_total,
  ROUND(SUM(p.priceUSD * o.quantityOrdered * (1 - (c.discountPct / 100)),
  ROUND(o.totalUSD - ROUND(SUM(p.priceUSD * o.quantityOrdered * (1 - (c
```

Use code [with caution](#).



Gemini's Results

	ordernum [PK] integer	recorded_total numeric (12,2)	calculated_total numeric	difference numeric
1	1024	56671.55	56617.55	54.00
2	1017	25643.98	25643.89	0.09

My Results

	ordernum integer	dateordered date	custid integer	agentid integer	prodid character	quantityordered integer	totalusd numeric (12,2)	pid integer	paymentterms text	discountpct numeric (5,2)
1	1017	2023-02-14	1	3	p03	500	25643.98	1	Net 30	21.12
2	1024	2023-03-15	5	2	p01	880	56671.55	5	In Advance	5.05

In conclusion...

I think that LLMs could be very useful for SQL queries. But at this point in time, it seems like once the queries start to get a little more complex, the LLM starts to mess up a little bit. So for now, human jobs are safe, but as LLMs advance, I can 100% see them taking over and doing all the queries instead of a human manually typing them in.

This worked much better than I expected and only produced very small errors that were easily fixed.