
- 출처: LangChain 공식 문서 또는 해당 교재명
- 원본 URL: https://smith.langchain.com/hub/teddynote/summary-stuff-documents

---

3. `구조화된 출력 체인` (`with_structered_output`)

1) `구조화된 출력을 사용하는 체인` *(with_structured_output)*

- **`환경설정`**

```
# API 키를 환경변수로 관리하기 위한 설정 파일
from dotenv import load_dotenv

# API 키 정보 로드
load_dotenv()                                    # True
```

- **`특정 주제`에 대한 `4지선다형 퀴즈`를 `생성`하는 `과정 구현하기`**
    - `Quiz 클래스` = 퀴즈의 `질문`, `난이도`, 그리고 `네 개의 선택지`를 정의함
    - `LLM` 인스턴스 = `gemini-2.5-flash-lite` → **`자연어 처리` 수행**
    - `ChatPromptTemplate` = `퀴즈 생성`을 위한 `대화형 프롬프트` 정의

```
from langchain.chains.openai_functions import create_structured_output_runnable
from langchain_google_genai import ChatGoogleGenerativeAI
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.pydantic_v1 import BaseModel, Field
from typing import List

# Quiz 클래스 정의하기
class Quiz(BaseModel):
    """4지선다형 퀴즈의 정보를 추출합니다"""

    question: str = Field(..., description="퀴즈의 질문")
    level: str = Field(
        ..., description="퀴즈의 난이도를 나타냅니다. (쉬움, 보통, 어려움)"
    )
    options: List[str] = Field(..., description="퀴즈의 4개의 선택지 입니다.")
```

- `Quiz` 클래스 정의하기 - (`3.9s`)

```
For example, replace imports like: `from langchain_core.pydantic_v1 import BaseM
with: `from pydantic import BaseModel`
or the v1 compatibility namespace if you are working in a code base that has no

exec(code_obj, self.user_global_ns, self.user_ns)
```

```
# LLM 생성하기

import os
from dotenv import load_dotenv

# API 키 확인
if not os.getenv("GOOGLE_API_KEY"):
    os.environ["GOOGLE_API_KEY"] = input("Enter your Google API key: ")

# LLM 초기화
gemini_lc = ChatGoogleGenerativeAI(
    model="gemini-2.5-flash-lite",
    temperature=0.1,
    max_output_tokens=4096,
)
```

- **`gemini-2.5-flash-lite` 설정하기**

```
E0000 00:00:1760245856.780887 4673979 alts_credentials.cc:93] ALTS creds ignore
```

```
# 프롬프트 생성하기
prompt = ChatPromptTemplate.from_messages(
    [
        (
            "system",
            "You're a world-famous quizzer and generates quizzes in structured fo
        ),
        (
            "human",
            "TOPIC 에 제시된 내용과 관련한 4지선다형 퀴즈를 출제해 주세요. 만약, 실제 출제된 기출문
            "단, 문제에 TOPIC 에 대한 내용이나 정보는 포함하지 마세요. \nTOPIC:\n{topic}",
        ),
        ("human", "Tip: Make sure to answer in the correct format"),
    ]
)
```

```
# 구조화된 출력을 위한 모델 생성

llm_with_structured_output = gemini_lc.with_structured_output(Quiz)
```

```
# 퀴즈 생성 체인 생성하기

chain = prompt | llm_with_structured_output
```

```
# 퀴즈 생성 요청해보기

generated_quiz = chain.invoke({"topic": "ADSP(데이터 분석 준전문가) 자격 시험"})
```

- 퀴즈 생성 요청 - (`1.2s`)
  - `print(generated_quiz)`

```
question='다음 중 데이터 분석의 주요 단계가 아닌 것은 무엇인가요?' level='쉬움' options=[
```

- **`생성된 퀴즈 출력해보기`**

```
# 생성된 퀴즈 출력하기

print(f"{generated_quiz.question} (난이도: {generated_quiz.level})\n")
for i, opt in enumerate(generated_quiz.options):
    print(f"{i+1}) {opt}")
```

- 퀴즈처럼 출력하기

```
다음 중 데이터 분석의 주요 단계가 아닌 것은 무엇인가요? (난이도: 쉬움)

1) 데이터 수집
2) 데이터 정제
3) 모델링
4) 하드웨어 구매
```

---

2) `Structured-Data-Chat`

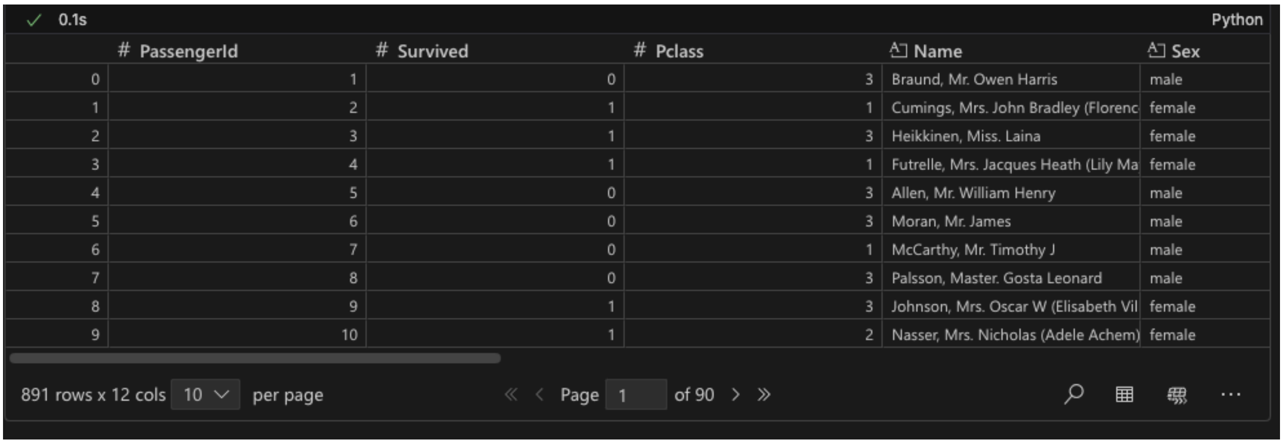
- **데이터 로드하기**

```
import pandas as pd
from langchain_core.prompts import ChatPromptTemplate
from langchain_experimental.tools import PythonAstREPLTool
```

```
df = pd.read_csv("../14_Chains/data/titanic.csv")    # 1.2s
```

```
# titanic.csv 파일에서 데이터를 읽어와 DataFrame으로 저장하기

tool = PythonAstREPLTool(locals={"df": df})
```

```
# PythonAstREPLTool을 사용하여 로컬 변수 'df'를 포함하는 환경 생성하기
tool.invoke("df")
```

- `tool.invoke("df")` 결과 - (`0.1s`)
  - 

    ✓ 0.1s Python

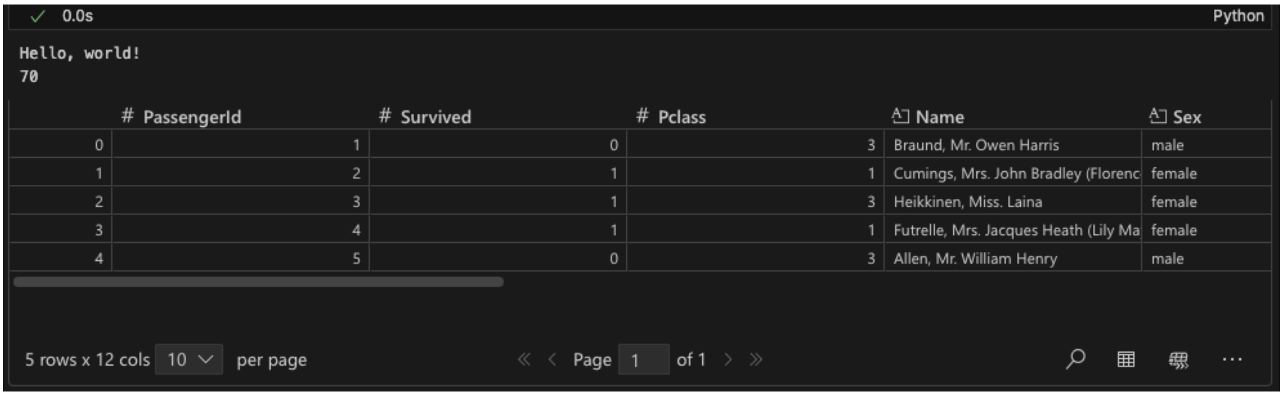
	PassengerId	Survived	Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florenc	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily Ma	female
4	5	0	3	Allen, Mr. William Henry	male
5	6	0	3	Moran, Mr. James	male
6	7	0	1	McCarthy, Mr. Timothy J	male
7	8	0	3	Palsson, Master. Gosta Leonard	male
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vil	female
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female

    891 rows x 12 cols 10 per page Page 1 of 90
  - `data` → `csv`로 저장함: [`titanic_data_tool_invoke.csv`]

```
hello = """
print("Hello, world!")

def add(a, b):
    return a + b

print(add(30, 40))

import pandas as pd

df = pd.read_csv("./data/titanic.csv")
df.head()
"""
```

```
tool = PythonAstREPLTool(locals={"df": df})

# PythonAstREPLTool 사용 → 로컬 변수 'df'를 포함하는 환경 생성하기
tool.invoke(hello)
```

- `tool.invoke(hello)` - (`0.0s`)
  - **`Hello, world!`**
  - **`70`**

    - 
      ✓ 0.0s Python

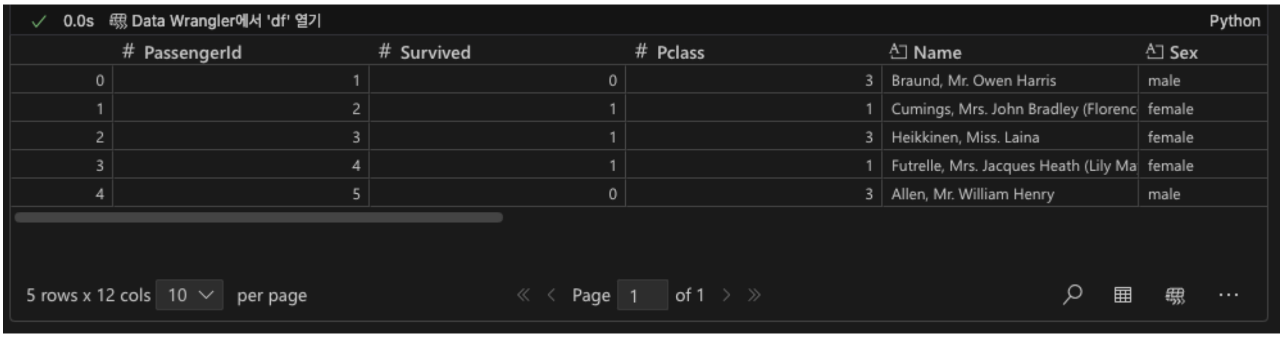
      ```
      Hello, world!
      70
      ```

	# PassengerId	# Survived	# Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florenc	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily Ma	female
4	5	0	3	Allen, Mr. William Henry	male

    - `data` → `csv`로 저장함: [`titanic_data_tool_invoke_hello.csv`](titanic_data_tool_invoke_hello.csv)

- **`pandas` → `csv` → `DataFrame`으로 로드하기**

```
import pandas as pd

# 데이터프레임으로 로드하기
df = pd.read_csv("../14_Chains/data/titanic.csv")

# 로드된 데이터프레임 일부 확인하기
df.head()
```

- `df.head()`
    - 
      ✓ 0.0s Data Wrangler에서 'df' 열기 Python

	# PassengerId	# Survived	# Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florenc	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily Ma	female
4	5	0	3	Allen, Mr. William Henry	male

- **`Pandas DataFrame Angent`**

```
from langchain.agents.agent_types import AgentType
from langchain_experimental.agents.agent_toolkits import create_pandas_dataframe_
from langchain_google_genai import ChatGoogleGenerativeAI
from langchain.agents.agent_types import AgentType
from langchain.callbacks.base import BaseCallbackHandler

# 스트리밍 콜백 클래스 정의
class StreamCallback(BaseCallbackHandler):
    def on_llm_new_token(self, token: str, **kwargs):
        print(token, end="", flush=True)                # 0.2s
```

```
# 에이전트 생성
agent = create_pandas_dataframe_agent(
```

```
    # 모델 정의
    ChatGoogleGenerativeAI(
        model="gemini-2.5-flash-lite",
        streaming=True,
        callbacks=[StreamCallback()],
    ),

    # 데이터프레임
    df,

    # 추론과정 출력하기
    verbose=True,  # 추론과정 출력

    # AgentType.ZERO_SHOT_REACT_DESCRIPTION
    agent_type=AgentType.ZERO_SHOT_REACT_DESCRIPTION,

    # 🚨 [해결 방안 추가]: 임의 코드 실행 위험을 감수하고 기능 사용을 허용
    allow_dangerous_code=True,
)
```

- `gemini-2.5-flash-lite`로 `agent` 생성하기

  ```
  E0000 00:00:1760247595.484815 4673979 alts_credentials.cc:93] ALTS creds ignore
  ```

- 추가: **`llow_dangerous_code=True`**

```
# 질의

agent.invoke({"input": "데이터의 행과 열의 갯수는 어떻게 돼?"})
```

- 과정 출력 - (*`verbose=True`*)

  ```
  **Entering new AgentExecutor chain...**

  Thought: The question asks for the number of rows and columns in the dataframe.

  Action: python_repl_ast

  Action Input: print(df.shape) *Thought: The question asks for the number of row

  *Action: python_repl_ast*

  *Action Input: print(df.shape)*

  `df.shape` attribute returns a tuple where the first element is the number of r
  ```

```
Final Answer: 데이터의 행은 891개이고 열은 12개입니다. **`df.shape` attribute returns a

**Final Answer: 데이터의 행은 891개이고 열은 12개입니다.**

**Finished chain.**
```

- 최종 결과 - (`1.6s`)

```
{'input': '데이터의 행과 열의 갯수는 어떻게 돼?', 'output': '데이터의 행은 891개이고 열은 12개
```

```
# 질의_2

agent.run("남자 승객의 생존율을 어떻게 돼? %로 알려줘")
```

- 질의_2
- 과정 출력 - (`verbose=True`)

```
/var/folders/h3/l7wnkv352kqftv0t8ctl2ld40000gn/T/ipykernel_90413/3819232585.py:
  agent.run("남자 승객의 생존율을 어떻게 돼? %로 알려줘")
```

```
**Entering new AgentExecutor chain...**
Thought: The question asks for the survival rate of male passengers as a percen
To answer this, I need to:
1. Filter the dataframe to include only male passengers.
2. Calculate the number of male passengers who survived.
3. Calculate the total number of male passengers.
4. Divide the number of survivors by the total number of male passengers and mu

I will start by filtering the dataframe for male passengers.
Then I will count the number of survivors within that subset.
Finally, I will calculate the percentage.

First, let's select only the male passengers.
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male']['Survived'].value_counts(normalize=
To answer this, I need to:
1. Filter the dataframe to include only male passengers.
2. Calculate the number of male passengers who survived.
3. Calculate the total number of male passengers.
4. Divide the number of survivors by the total number of male passengers and mu
```

```
I will start by filtering the dataframe for male passengers.
Then I will count the number of survivors within that subset.
Finally, I will calculate the percentage.

First, let's select only the male passengers.
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male']['Survived'].value_counts(normalize=
0    0.811092
1    0.188908
Name: proportion, dtype: float64
**The output shows the proportion of male passengers who did not survive (0) an
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male']['Survived'].mean() * 100) **The out
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male']['Survived'].mean() * 100) **18.8908
*The previous action directly calculated the mean of the 'Survived' column for
Final Answer: 남자 승객의 생존율은 약 18.89%입니다. **The previous action directly calc
**Final Answer: 남자 승객의 생존율은 약 18.89%입니다.**

**Finished chain.**
```

- 최종 결과 - (`2.7s`)

```
'남자 승객의 생존율은 약 18.89%입니다.'
```

```
# 질의_3

agent.run(
    "나이가 15세 이하인 승객중 1,2등급에 탑승한 남자 승객의 생존율은 어떻게 돼? %로 알려줘"
)
```

- 질의_3
- 과정 출력 (`verbose=True`)

```
**Entering new AgentExecutor chain...**
Thought: The user is asking for the survival rate of male passengers who are 15
To answer this, I need to:
1. Filter the dataframe to include only male passengers.
2. Filter the dataframe to include only passengers aged 15 or younger.
3. Filter the dataframe to include only passengers in Pclass 1 or 2.
4. Calculate the survival rate (mean of the 'Survived' column) for the filtered
5. Present the result as a percentage.
```

```
Let's start by filtering for male passengers.
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male'].head()) *Thought: The user is askin
To answer this, I need to:
1. Filter the dataframe to include only male passengers.
2. Filter the dataframe to include only passengers aged 15 or younger.
3. Filter the dataframe to include only passengers in Pclass 1 or 2.
4. Calculate the survival rate (mean of the 'Survived' column) for the filtered
5. Present the result as a percentage.

Let's start by filtering for male passengers.
Action: python_repl_ast
Action Input: print(df[df['Sex'] == 'male'].head())
PassengerId  Survived  Pclass                            Name   Sex   Age  \
0            1         0       3         Braund, Mr. Owen Harris  male  22.0
4            5         0       3        Allen, Mr. William Henry  male  35.0
5            6         0       3                Moran, Mr. James  male   NaN
6            7         0       1         McCarthy, Mr. Timothy J  male  54.0
7            8         0       3  Palsson, Master. Gosta Leonard  male   2.0

   SibSp  Parch     Ticket     Fare Cabin Embarked
0      1      0  A/5 21171   7.2500   NaN        S
4      0      0     373450   8.0500   NaN        S
5      0      0     330877   8.4583   NaN        Q
6      0      0      17463  51.8625   E46        S
7      3      1     349909  21.0750   NaN        S

**Now that I have filtered for male passengers, I need to apply the age and Pcl

Action: python_repl_ast
Action Input: print(df[(df['Sex'] == 'male') & (df['Age'] <= 15) & (df['Pclass']

Action: python_repl_ast
Action Input: print(df[(df['Sex'] == 'male') & (df['Age'] <= 15) & (df['Pclass']

PassengerId  Survived  Pclass                                 Name   Sex  \
78            79         1       2        Caldwell, Master. Alden Gates  male
183          184         1       2            Becker, Master. Richard F  male
193          194         1       2           Navratil, Master. Michel M  male
305          306         1       1       Allison, Master. Hudson Trevor  male
340          341         1       2       Navratil, Master. Edmond Roger  male
407          408         1       2       Richards, Master. William Rowe  male
445          446         1       1            Dodge, Master. Washington  male
549          550         1       2       Davies, Master. John Morgan Jr  male
755          756         1       2            Hamalainen, Master. Viljo  male
802          803         1       1  Carter, Master. William Thornton II  male
827          828         1       2                Mallet, Master. Andre  male
```

```
831          832         1       2      Richards, Master. George Sibley  male

      Age  SibSp  Parch           Ticket      Fare    Cabin Embarked
78    0.83      0      2           248738   29.0000      NaN        S
183   1.00      2      1           230136   39.0000       F4        S
193   3.00      1      1           230080   26.0000       F2        S
305   0.92      1      2           113781  151.5500  C22 C26        S
340   2.00      1      1           230080   26.0000       F2        S
407   3.00      1      1            29106   18.7500      NaN        S
445   4.00      0      2            33638   81.8583      A34        S
549   8.00      1      1       C.A. 33112   36.7500      NaN        S
755   0.67      1      1           250649   14.5000      NaN        S
802  11.00      1      2           113760  120.0000  B96 B98        S
827   1.00      0      2  S.C./PARIS 2079   37.0042      NaN        C
831   0.83      1      1            29106   18.7500      NaN        S

*I have successfully filtered the dataframe to include only male passengers age

Action: python_repl_ast

Action Input: print(df[(df['Sex'] == 'male') & (df['Age'] <= 15) & (df['Pclass'
Action: python_repl_ast
Action Input: print(df[(df['Sex'] == 'male') & (df['Age'] <= 15) & (df['Pclass'
**Final Answer: 100.0**
**The survival rate is 1.0, which means 100%.**
**Final Answer: 100.0**

**Finished chain.**
```

- 최종 결과 - (`3.7s`)

```
'100.0'
```

```
# 질의_4

agent.run(
    "Pclass 가 1등급인 승객 중에서 나이가 20세~30세 사이이고, 여성 승객의 생존율은 어떻게 돼? %로
)
```

- 질의_4
- 과정 출력 - (`verbose=True`)

```
**Entering new AgentExecutor chain...**
Thought: The user is asking for the survival rate of female passengers who are
```

```
I need to filter the dataframe based on these conditions and then calculate the
The survival rate can be calculated as the number of survivors divided by the t

Here's the plan:
1. Filter `df` for `Pclass == 1`.
2. From the result, filter for `Sex == 'female'`.
3. From the result, filter for `Age >= 20` and `Age <= 30`.
4. Calculate the mean of the 'Survived' column for the final filtered group. Th
5. Multiply by 100 to get the percentage.

Let's start by filtering for Pclass 1.
Then, I will filter for female passengers.
Then, I will filter for the age range.
Finally, I will calculate the survival rate.
Action: python_repl_ast
Action Input: print(df[(df['Pclass'] == 1) & (df['Sex'] == 'female') & (df['Age
I need to filter the dataframe based on these conditions and then calculate the
The survival rate can be calculated as the number of survivors divided by the t

Here's the plan:
6. Filter `df` for `Pclass == 1`.
7. From the result, filter for `Sex == 'female'`.
8. From the result, filter for `Age >= 20` and `Age <= 30`.
9. Calculate the mean of the 'Survived' column for the final filtered group. Th
10. Multiply by 100 to get the percentage.

Let's start by filtering for Pclass 1.
Then, I will filter for female passengers.
Then, I will filter for the age range.
Finally, I will calculate the survival rate.
Action: python_repl_ast
Action Input: print(df[(df['Pclass'] == 1) & (df['Sex'] == 'female') & (df['Age
*The observation shows the survival rate as a percentage. The question asks for
**Final Answer: 95.23809523809523%** **The observation shows the survival rate
**Final Answer: 95.23809523809523%**

**Finished chain.**
```

- 최종 결과 - (`2.5s`)

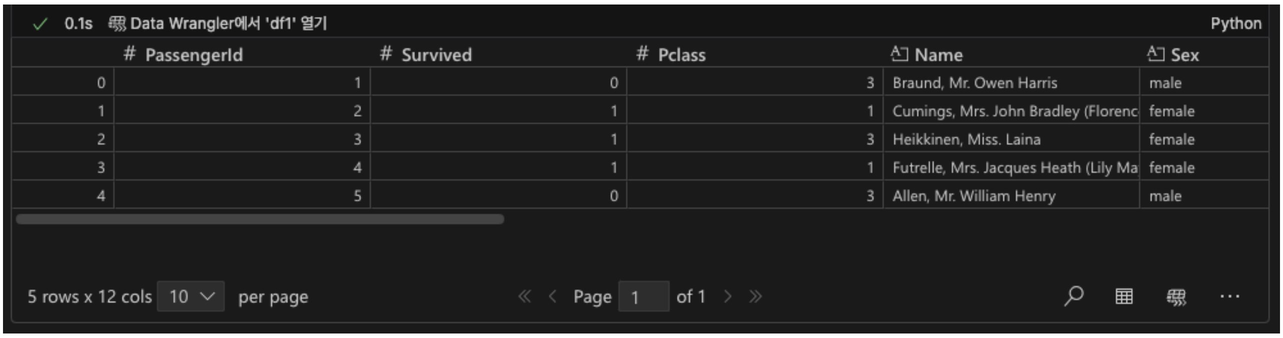
```
'95.23809523809523%'
```

---

3) `2개 이상의 DataFrame`

- **`2개 이상의 DataFrame에 기반한 LLM 기반 질의 가능`**
  - *2개 이상의 DataFrame 입력 시 `[ ]` 로 묶어주면 됨*

```
# 샘플 데이터프레임 생성해보기

df1 = df.copy()
df1 = df1.fillna(0)
df1.head()
```

- `df1.head()` - (`0.1s`)
  - 

✓ 0.1s Data Wrangler에서 'df1' 열기 Python

	# PassengerId	# Survived	# Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florenc	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily Ma	female
4	5	0	3	Allen, Mr. William Henry	male

5 rows x 12 cols 10 per page Page 1 of 1

```
# 에이전트 생성_2
agent = create_pandas_dataframe_agent(
    # 모델 정의
    ChatGoogleGenerativeAI(
        model="gemini-2.5-flash-lite",
        streaming=True,
        callbacks=[StreamCallback()],
    ),

    # 데이터프레임
    df,

    # 추론과정 출력하기
    verbose=True,  # 추론과정 출력

    # AgentType.ZERO_SHOT_REACT_DESCRIPTION
    agent_type=AgentType.ZERO_SHOT_REACT_DESCRIPTION,

    # 🚨 [해결 방안 추가]: 임의 코드 실행 위험을 감수하고 기능 사용을 허용
    allow_dangerous_code=True,
)
```

- `agent` 생성하기

```
E0000 00:00:1760249015.026490 4673979 alts_credentials.cc:93] ALTS creds ignore
```

```
# 질의해보기

agent.invoke({"input": "나이 컬럼의 나이의 평균차이는 어떻게 돼? %로 구해줘."})
```

- 과정 출력하기 - (`verbose=True`)

```


**Entering new AgentExecutor chain...**
Thought: The user is asking for the average difference in age, expressed as a p

First, I need to calculate the mean and median of the 'Age' column.
Then, I will calculate the absolute difference between the mean and median.
Finally, I will express this difference as a percentage of the mean age.
Action: python_repl_ast
Action Input: print(df['Age'].mean()) *Thought: The user is asking for the aver

First, I need to calculate the mean and median of the 'Age' column.
Then, I will calculate the absolute difference between the mean and median.
Finally, I will express this difference as a percentage of the mean age.
Action: python_repl_ast
Action Input: print(df['Age'].mean()) **29.69911764705882**
*I have calculated the mean of the 'Age' column. Now I need to calculate the me
Action: python_repl_ast
Action Input: print(df['Age'].median()) **I have calculated the mean of the 'Ag
Action: python_repl_ast
Action Input: print(df['Age'].median()) **28.0**
*I have the mean (29.699) and median (28.0) of the 'Age' column.*
Now I need to calculate the absolute difference between the mean and median, an

Absolute difference = |mean - median|
Percentage difference = (Absolute difference / mean) * 100
Action: python_repl_ast
Action Input: print(abs(29.69911764705882 - 28.0) / 29.69911764705882 * 100) **
Now I need to calculate the absolute difference between the mean and median, an

Absolute difference = |mean - median|
Percentage difference = (Absolute difference / mean) * 100
Action: python_repl_ast
Action Input: print(abs(29.69911764705882 - 28.0) / 29.69911764705882 * 100 **5
*I have calculated the mean age, the median age, and the percentage difference
**Final Answer: 나이 컬럼의 평균과 중앙값의 차이는 평균값의 약 5.72% 입니다.**
**I have calculated the mean age, the median age, and the percentage difference
**Final Answer: 나이 컬럼의 평균과 중앙값의 차이는 평균값의 약 5.72% 입니다.**

**Finished chain.**

```

- 최종 결과물 - (`3.9s`)

  ```
  {'input': '나이 컬럼의 나이의 평균차이는 어떻게 돼? %로 구해줘.',
  'output': '나이 컬럼의 평균과 중앙값의 차이는 평균값의 약 5.72% 입니다.'}
  ```

- next: ***`CH15 평가 (Evaluations)`***
