Add Memory to OpenAI Functions Agent

This notebook goes over how to add memory to OpenAI Functions agent.

```
from langchain import (
    LLMMathChain,
    OpenAI,
    SerpAPIWrapper,
    SQLDatabase,
    SQLDatabaseChain,
)
from langchain.agents import initialize_agent, Tool
from langchain.agents import AgentType
from langchain.chat_models import ChatOpenAI
```

API Reference:

- initialize_agent from `langchain.agents`
- Tool from `langchain.agents`
- AgentType from `langchain.agents`
- ChatOpenAI from `langchain.chat_models`

```
/Users/harrisonchase/.pyenv/versions/3.9.1/envs/langchain/lib/python3.9/site
packages/deeplake/util/check_latest_version.py:32: UserWarning: A newer
version of deeplake (3.6.4) is available. It's recommended that you update t
the latest version using `pip install -U deeplake`.
      warnings.warn(
```

```
llm = ChatOpenAI(temperature=0, model="gpt-3.5-turbo-0613")
search = SerpAPIWrapper()
llm_math_chain = LLMMathChain.from_llm(llm=llm, verbose=True)
db = SQLDatabase.from_uri("sqlite:///../../../../../notebooks/Chinook.db")
db_chain = SQLDatabaseChain.from_llm(llm, db, verbose=True)
tools = [
```

```
    Tool(
        name="Search",
        func=search.run,
        description="useful for when you need to answer questions about
current events. You should ask targeted questions",
    ),
    Tool(
        name="Calculator",
        func=llm_math_chain.run,
        description="useful for when you need to answer questions about
math",
    ),
    Tool(
        name="FooBar-DB",
        func=db_chain.run,
        description="useful for when you need to answer questions about
FooBar. Input should be in the form of a question containing full
context",
    ),
]
```

```
from langchain.prompts import MessagesPlaceholder
from langchain.memory import ConversationBufferMemory

agent_kwargs = {
    "extra_prompt_messages":
[MessagesPlaceholder(variable_name="memory")],
}
memory = ConversationBufferMemory(memory_key="memory",
return_messages=True)
```

API Reference:

- MessagesPlaceholder from `langchain.prompts`
- ConversationBufferMemory from `langchain.memory`

```
agent = initialize_agent(
    tools,
    llm,
    agent=AgentType.OPENAI_FUNCTIONS,
    verbose=True,
```

```
    agent_kwargs=agent_kwargs,
    memory=memory,
)
```

```
agent.run("hi")
```

```
> Entering new  chain...
Hello! How can I assist you today?

> Finished chain.
```

```
'Hello! How can I assist you today?'
```

```
agent.run("my name is bob")
```

```
> Entering new  chain...
Nice to meet you, Bob! How can I help you today?

> Finished chain.
```

```
'Nice to meet you, Bob! How can I help you today?'
```

```
agent.run("whats my name")
```

```
> Entering new  chain...
Your name is Bob.

> Finished chain.
```

```
'Your name is Bob.'
```