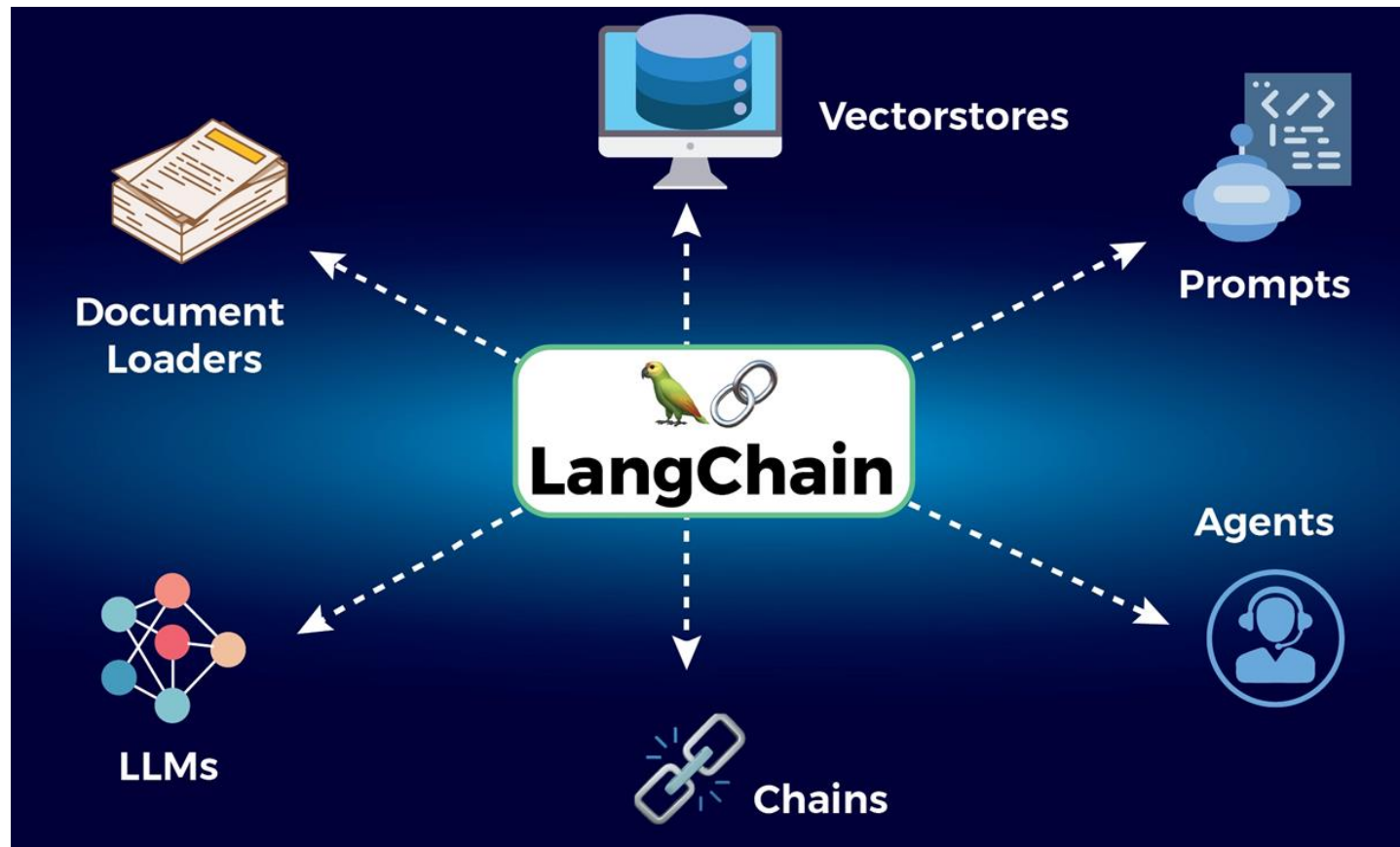
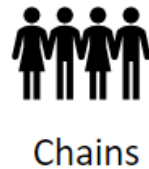


6. LangChain 모듈



LangChain 모듈

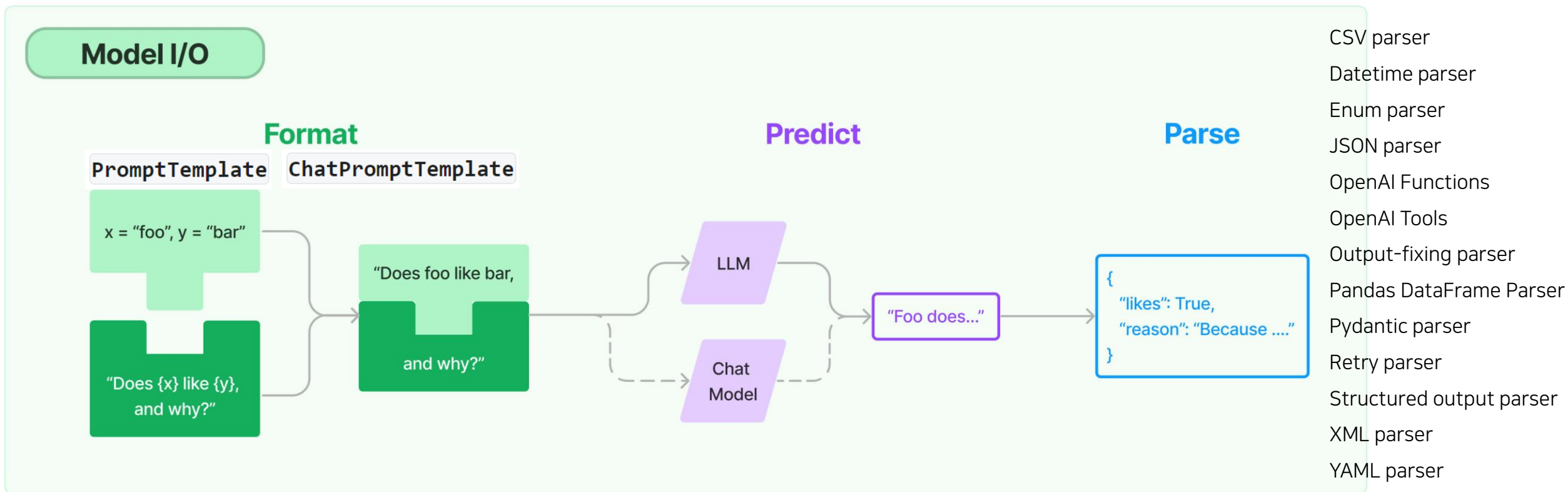


- Model I/O : 언어 모델과의 인터페이스
- Data Connection : 데이터와의 인터페이스
- Chains : 호출 시퀀스 구축
- Callbacks : 체인의 중간 단계를 기록 및 스트리밍
- Agents : 상위 지시문이 주어지면 체인이 사용할 도구(Tool)을 선택할 수 있도록 함
- Memory : 체인 실행 간에 애플리케이션 상태 유지

Model I/O

모든 언어모델 애플리케이션의 핵심 요소는 Model입니다.

Model I/O는 LangChain이 모든 언어모델과 인터페이스 할 수 있는 빌딩 블록을 제공합니다.



Model I/O

LangChain Data Ecosystem



Data Connectors (> 120 Integrations)

Unstructured

Structured

Public



arXiv



Datasets



OpenWeather

Proprietary

Personal / Company Data

.pdf, .txt, .json., .md, ...



Datastores

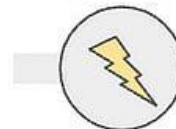


stripe

Vector Storage (> 35 Integrations)



Transformations



Embeddings

(> 25 Integrations)

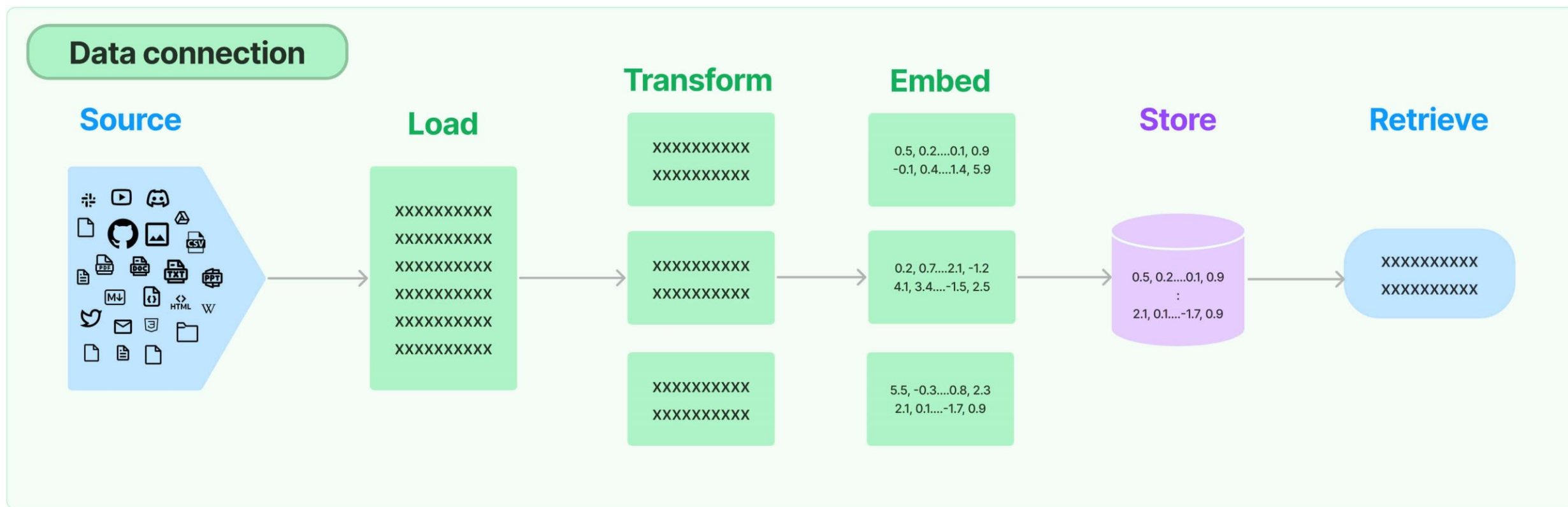
Model I/O 실습



LangChainModule_ModelIO.ipynb

Retrieval

많은 LLM 애플리케이션에는 모델 학습 세트의 일부가 아닌 사용자 데이터가 필요합니다.
이를 달성하는 주요 방법은 검색 증강 생성(RAG, Retrieval-Augmented Generation)을 사용하는 것입니다.
LangChain은 RAG 애플리케이션을 위한 모든 빌딩 블록을 제공합니다.



Retrieval

Document loader

- Document loader 는 다양한 소스에서 문서를 로드합니다.
- LangChain은 100가지가 넘는 다양한 문서 로더를 제공할 뿐만 아니라 AirByte 및 Unstructured와 같은 다른 주요 공급자와의 통합을 제공합니다.
- LangChain은 모든 유형의 위치(비공개 S3 버킷, 공개 웹사이트)에서 모든 유형의 문서(HTML, PDF, 코드)를 로드할 수 있는 통합 기능을 제공합니다.

Text Splitting

- 검색의 핵심은 문서에서 관련 부분만 가져오는 것입니다.
- 여기에는 검색을 위해 문서를 준비하기 위한 몇 가지 변환 단계가 포함됩니다.
- 여기서 가장 중요한 것 중 하나는 큰 문서를 작은 덩어리로 분할(또는 청크)하는 것입니다.
- LangChain은 이를 위한 몇 가지 변환 알고리즘과 특정 문서 유형(코드, 마크다운 등)에 최적화된 로직을 제공합니다.

Text embedding model

- 검색의 또 다른 핵심 부분은 문서에 임베딩을 만드는 것입니다.
- 임베딩은 텍스트의 의미론적 의미를 포착하여 유사한 텍스트를 빠르고 효율적으로 찾을 수 있게 해줍니다.

Retrieval

Vector store

- 벡터 스토어는 임베딩의 효율적인 저장과 검색을 지원합니다.
- LangChain은 오픈소스 로컬 스토어부터 클라우드 호스팅 벡터스토어까지 다양한 벡터 스토어와의 통합을 제공합니다.

Retriever

- 데이터베이스에 저장된 데이터를 검색 합니다.
- LangChain은 다양한 검색 알고리즘을 지원합니다.
- LangChain은 시맨틱 검색을 지원하며, 아래와 같은 성능 향상 알고리즘이 있습니다.
 - Parent Document Retriever: 상위 문서당 여러 개의 임베딩을 생성할 수 있어 작은 청크를 조회할 수 있지만 더 큰 컨텍스트를 반환할 수 있습니다.
 - Self Query Retriever: 사용자 질문에는 단순한 의미론이 아니라 메타데이터 필터로 가장 잘 표현될 수 있는 논리를 표현하는 참조가 포함되어 있는 경우가 많습니다. 자체 쿼리를 사용하면 쿼리에 존재하는 다른 메타데이터 필터로부터 쿼리의 의미론적 부분을 파싱할 수 있습니다.
 - Ensemble Retriever: 여러 다른 소스에서 또는 여러 다른 알고리즘을 사용하여 문서를 검색합니다.

Retrieval 실습

LangChain
소스 다운로드

<https://github.com/langchain-ai/langchain/tree/master>

실습 디렉토리에
복사

`.Wlangchain-masterWdocsWdocsWintegrationsWdocument_loadersWexample_data`



LangChainModule_Retrieval.ipynb

Agent

Agent의 핵심 아이디어는 언어 모델을 사용하여 수행할 일련의 작업을 선택하는 것입니다.
에이전트에서는 언어 모델이 추론 엔진으로 사용되어 어떤 작업을 어떤 순서로 수행할지 결정합니다.

Agent 타입

Agent Type	Intended Model Type	When to Use	API
OpenAI Tools	Chat	최신 OpenAI 모델(1106 이후)을 사용하는 경우	Ref
OpenAI Functions	Chat	OpenAI 모델 또는 함수 호출을 위해 미세 조정된 오픈 소스 모델을 사용 중이며 OpenAI와 동일한 함수 매개 변수를 노출하는 경우	Ref
XML	LLM	Anthropic 모델 또는 XML에 능숙한 다른 모델을 사용하는 경우	Ref
Structured Chat	Chat	여러 입력이 있는 도구를 지원해야 하는 경우	Ref
JSON Chat	Chat	JSON에 능숙한 모델을 사용하는 경우	Ref
ReAct	LLM	간단한 모델을 사용하는 경우	Ref
Self Ask With Search	LLM	간단한 모델을 사용 중이고 검색 도구가 하나만 있는 경우	Ref

Agents

LangSmith 셋업

<https://smith.langchain.com/>

- Agent는 사용자가 볼 수 있는 출력을 반환하기 전에 자체적으로 결정된 입력 종속적인 일련의 단계를 거칩니다.
- 따라서 이러한 시스템을 디버깅하는 것은 특히 까다롭고 관찰 가능성이 특히 중요합니다.
- LangSmith는 이러한 경우에 유용합니다.
- LangChain으로 빌드할 때, 모든 단계는 LangSmith에서 자동으로 추적됩니다.
- LangSmith를 설정하려면 다음 환경 변수를 설정하기만 하면 됩니다:

```
export LANGCHAIN_TRACING_V2="true"
```

```
export LANGCHAIN_API_KEY=" "
```



LangChainModule_Agent.ipynb

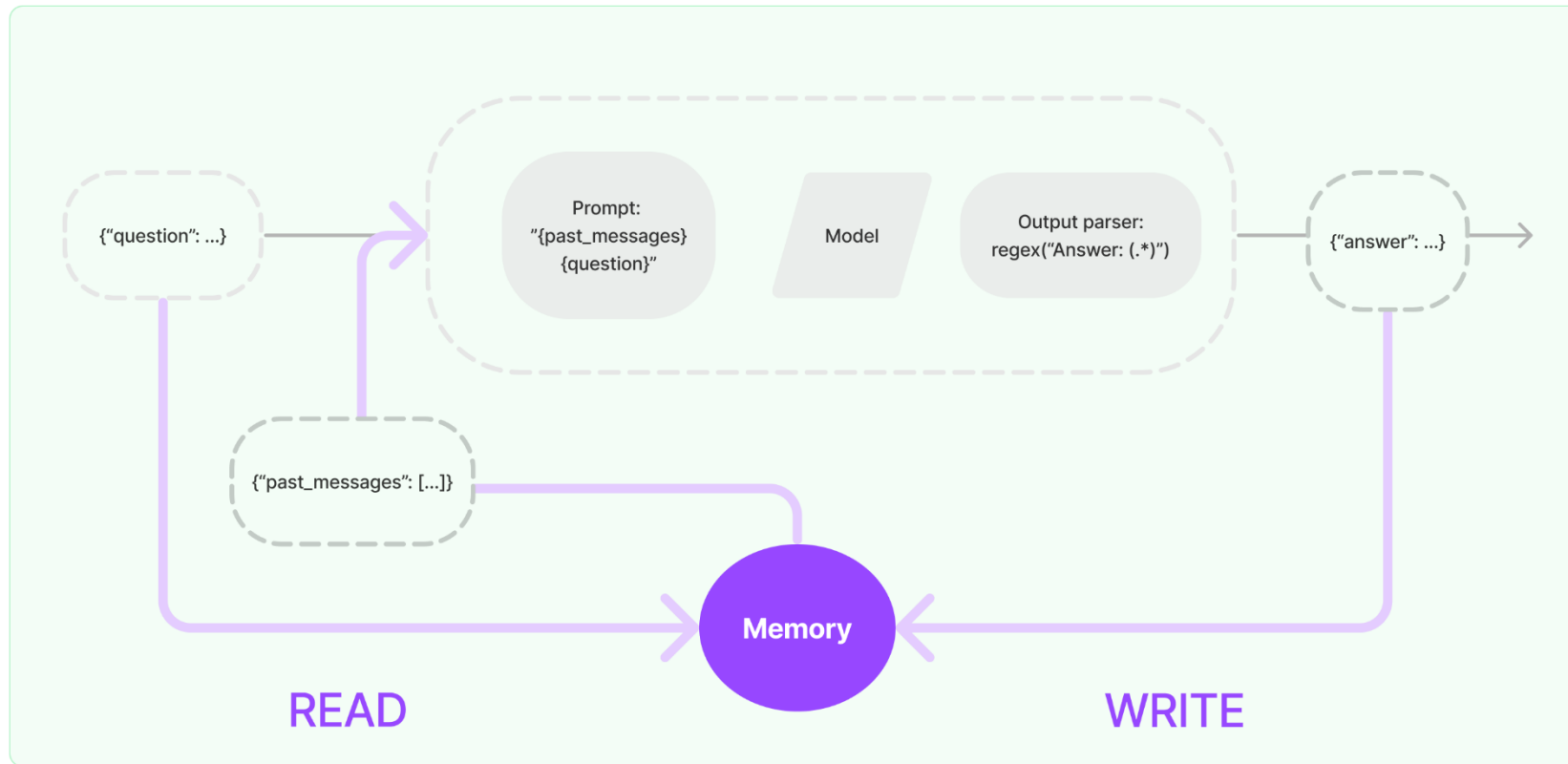
Chain

체인은 LLM, 도구 또는 데이터 전처리 단계에 대한 호출 시퀀스입니다. 기본적으로 지원되는 방법은 LCEL입니다. LCEL은 자신만의 체인을 구성하는 데 유용하지만, 기성품(off-the-shelf)으로 제공되는 체인을 사용하는 것도 좋습니다.

LCEL Chain	
Chain Constructor	When to Use
create_stuff_documents_chain	이 체인은 문서 목록을 가져와서 모두 프롬프트로 포맷한 다음 해당 프롬프트를 LLM에 전달합니다. 모든 문서를 전달하므로 사용 중인 LLM의 컨텍스트 창에 맞는지 확인해야 합니다.
create_openai_fn_runnable	OpenAI 함수 호출을 사용하여 선택적으로 출력 응답을 구조화하려는 경우. 호출을 위해 여러 함수를 전달할 수 있지만 반드시 호출할 필요는 없습니다.
create_structured_output_runnable	OpenAI 함수 호출을 사용하여 LLM이 특정 함수로 응답하도록 강제하려는 경우. 하나의 함수만 전달할 수 있으며, 체인은 항상 이 응답을 반환합니다.
load_query_constructor_runnable	쿼리를 생성하는 데 사용할 수 있습니다. 허용된 연산 목록을 지정해야 하며, 자연어 쿼리를 허용된 연산으로 변환하는 runnable을 반환합니다.
create_sql_query_chain	자연어로 SQL 데이터베이스에 대한 쿼리를 작성하려는 경우.
create_history_aware_retriever	이 체인은 대화 기록을 가져온 다음 이를 사용하여 검색 쿼리를 생성하고 이를 기본 리트리버에 전달합니다.
create_retrieval_chain	이 체인은 사용자 문의를 받아 리트리버로 전달하여 관련 문서를 가져옵니다. 그런 다음 해당 문서(및 원본 입력)는 LLM으로 전달되어 응답을 생성합니다.

Memory

LangChain Memory 는 대화 과정에서 발생하는 정보를 저장하고 관리하여, 보다 자연스럽게 지속적인 대화 경험을 제공할 수 있도록 합니다.



<https://python.langchain.com/docs/modules/memory/>



LangChainModule_Memory.ipynb

Callback

LangChain은 LLM 애플리케이션의 다양한 단계에 연결할 수 있는 콜백 시스템을 제공합니다. 이는 로깅, 모니터링, 스트리밍 및 기타 작업에 유용합니다. API 전체에서 사용할 수 있는 콜백 인수를 사용하여 이러한 이벤트를 구독할 수 있습니다.

```
class BaseCallbackHandler:
    """Base callback handler that can be used to handle callbacks from langchain."""

    def on_llm_start(
        self, serialized: Dict[str, Any], prompts: List[str], **kwargs: Any
    ) -> Any:
        """Run when LLM starts running."""

    def on_chat_model_start(
        self, serialized: Dict[str, Any], messages: List[List[BaseMessage]], **kwargs: Any
    ) -> Any:
        """Run when Chat Model starts running."""

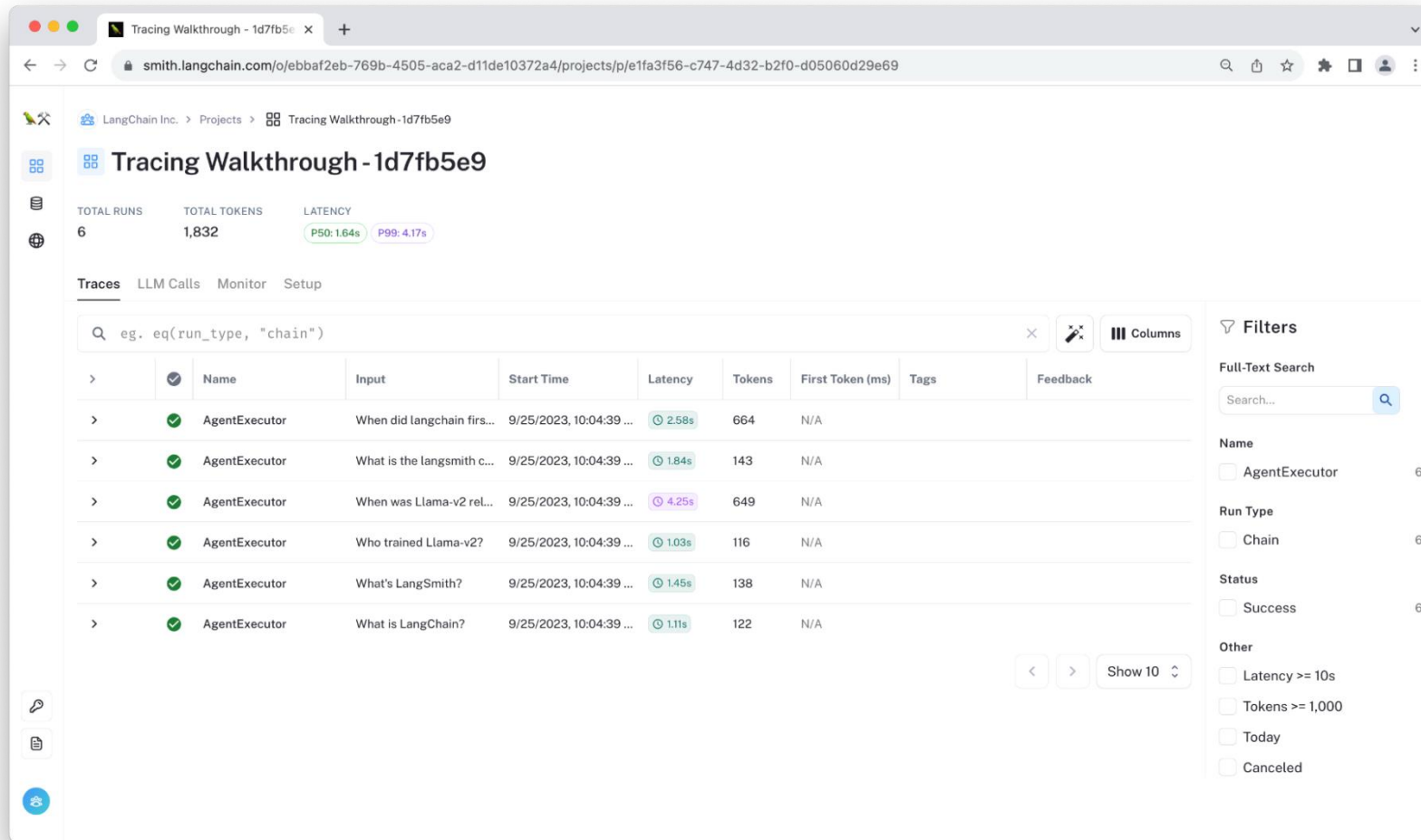
    def on_llm_new_token(self, token: str, **kwargs: Any) -> Any:
        """Run on new LLM token. Only available when streaming is enabled."""
```

<https://python.langchain.com/docs/modules/callbacks/>



LangSmith

LangSmith를 사용하면 LLM 애플리케이션을 쉽게 디버그, 테스트하고 지속적으로 개선할 수 있습니다.



The screenshot displays the LangSmith Tracing Walkthrough interface. At the top, the breadcrumb navigation shows 'LangChain Inc.' > 'Projects' > 'Tracing Walkthrough - 1d7fb5e9'. Below this, the title 'Tracing Walkthrough - 1d7fb5e9' is followed by summary statistics: 'TOTAL RUNS: 6', 'TOTAL TOKENS: 1,832', and 'LATENCY: P50: 1.64s, P99: 4.17s'. The 'Traces' tab is selected, showing a table of LLM calls. The table has columns for Name, Input, Start Time, Latency, Tokens, First Token (ms), Tags, and Feedback. A search bar at the top of the table contains the query 'eg. eq(run_type, "chain")'. The table lists six runs, all of which are successful (indicated by green checkmarks). The right sidebar contains a 'Filters' section with a 'Full-Text Search' bar and several filter categories: 'Name' (AgentExecutor, 6), 'Run Type' (Chain, 6), 'Status' (Success, 6), and 'Other' (Latency >= 10s, Tokens >= 1,000, Today, Canceled).

>	✓	Name	Input	Start Time	Latency	Tokens	First Token (ms)	Tags	Feedback
>	✓	AgentExecutor	When did langchain fir...	9/25/2023, 10:04:39 ...	2.58s	664	N/A		
>	✓	AgentExecutor	What is the langsmith c...	9/25/2023, 10:04:39 ...	1.84s	143	N/A		
>	✓	AgentExecutor	When was Llama-v2 rel...	9/25/2023, 10:04:39 ...	4.25s	649	N/A		
>	✓	AgentExecutor	Who trained Llama-v2?	9/25/2023, 10:04:39 ...	1.03s	116	N/A		
>	✓	AgentExecutor	What's LangSmith?	9/25/2023, 10:04:39 ...	1.45s	138	N/A		
>	✓	AgentExecutor	What is LangChain?	9/25/2023, 10:04:39 ...	1.11s	122	N/A		



walkthrough.ipynb

Thank you