Tableau DataFest

Tableau x LLM:
생성형 AI로 진화하는 데이터 인사이트

jonghoon(Jaden) Lee, STUG leader

jaden0715.lee@gmail.com

Tableau DataFest

이종훈

Full-stack engineer

Seoul Tableau User Group Leader (STUG)

- **‘22 DataFest Speaker (**반도체 공급망 대시보드**)**
- **‘24 1st offline meeting -** 식약처 데이터 활용 **pipeline** 구축 및 대시보드
- **‘25 1st offline meeting -** 경영전략 대시보드
- **‘25 Saleforces worldtour - Tableau community (on-premise tableau + LLM)**

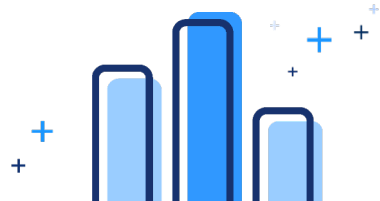
목표

Tableau DataFest

Tableau Server 환경에서도 **AI**와 자연어 질의 기반의 데이터 분석 가능성을 실현

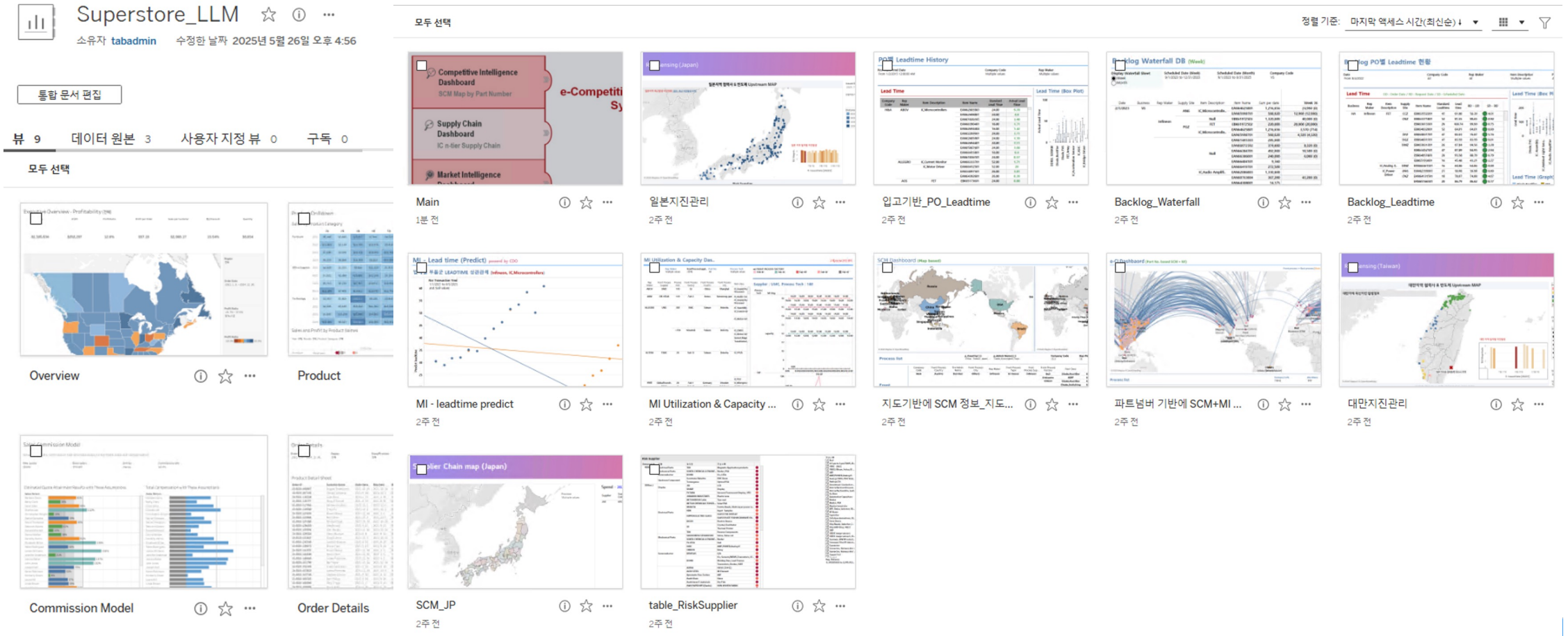
VizQL + VDS + MCP + RAG를 활용한 **Copilot** 구조를 쉽게 이해시키고, 구현 흐름

보안**(CORS,** 인증**),** 임베딩, 챗봇**, Agent** 구성 등 실무 중심



우리의 현실과 질문

대시보드는 많지만, 질문하기는 어렵죠?

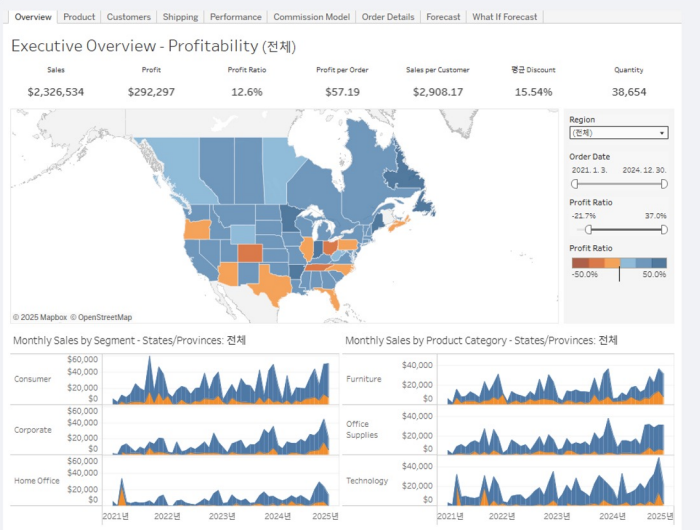


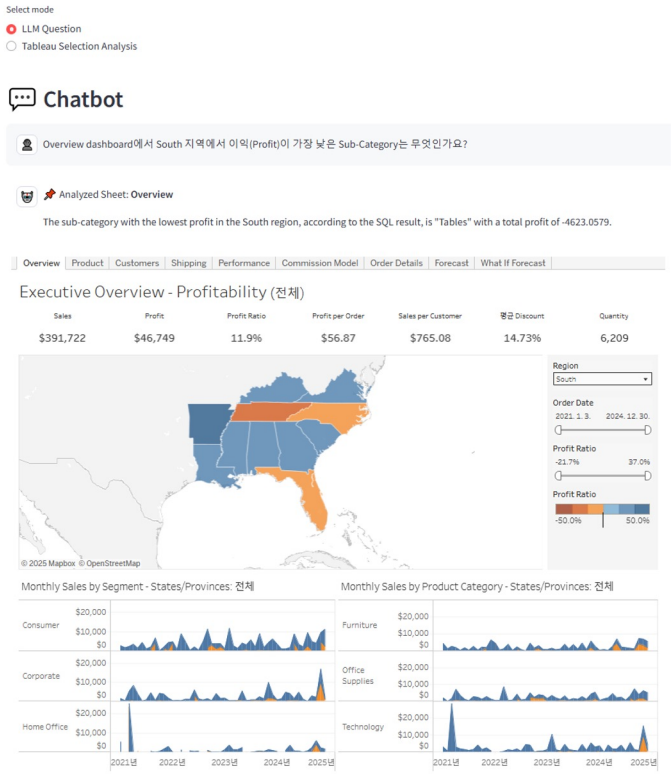
왜 **Tableau + LLM** 인가?

Tableau DataFest

"3월에 매출이 갑자기 줄어든 지역은 어디야?" 같은 질문이 실제로 가능한 시대

① 우리는 더 빠르고 자연스럽게 인사이트를 원합니다.
② 기존 분석 도구는 비대화적 과정이었습니다. 필터링, 클릭, 시트넘김 - 모두 대화는 아니었습니다.
③ LLM은 이런 장벽을 없애고, 자연어로 인사이트를 찾을 수 있게 합니다.





기존 BI vs Tableau+LLM: 진정한 변화

질문 예시: "3월에 매출이 줄어든 지역은?"

Tableau DataFest

기존 BI 방식

필요한 단계:

1. 판매 대시보드 시트 탐색 및 선택
2. 월별 필터에서 '3월' 선택
3. 지역별 필터 설정
4. 월간 비교 시트로 이동
5. 추가 필터링 및 정렬 설정

문제점:

- 복잡한 UI 작업이 필요
- 여러 대시보드 간 이동
- 반복적인 필터 설정

LLM + Tableau 방식

필요한 단계:

"3월에 매출이 줄어든 지역은?"

↓

만들어 놓은 시트 중에서 filter와 parameter를 통해 결과를 시각화

장점:

+ 자연어로 직관적 질의
+ 원하는 인사이트 즉시 확인
+ 기술적 지식 없이도 고급 분석 가능

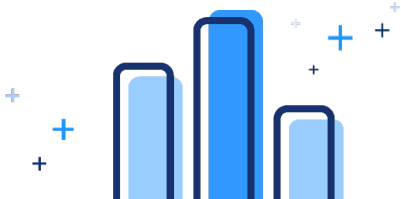


Tableau 환경의 현실과 제약

보안 그리고 on-premise tableau server 활용

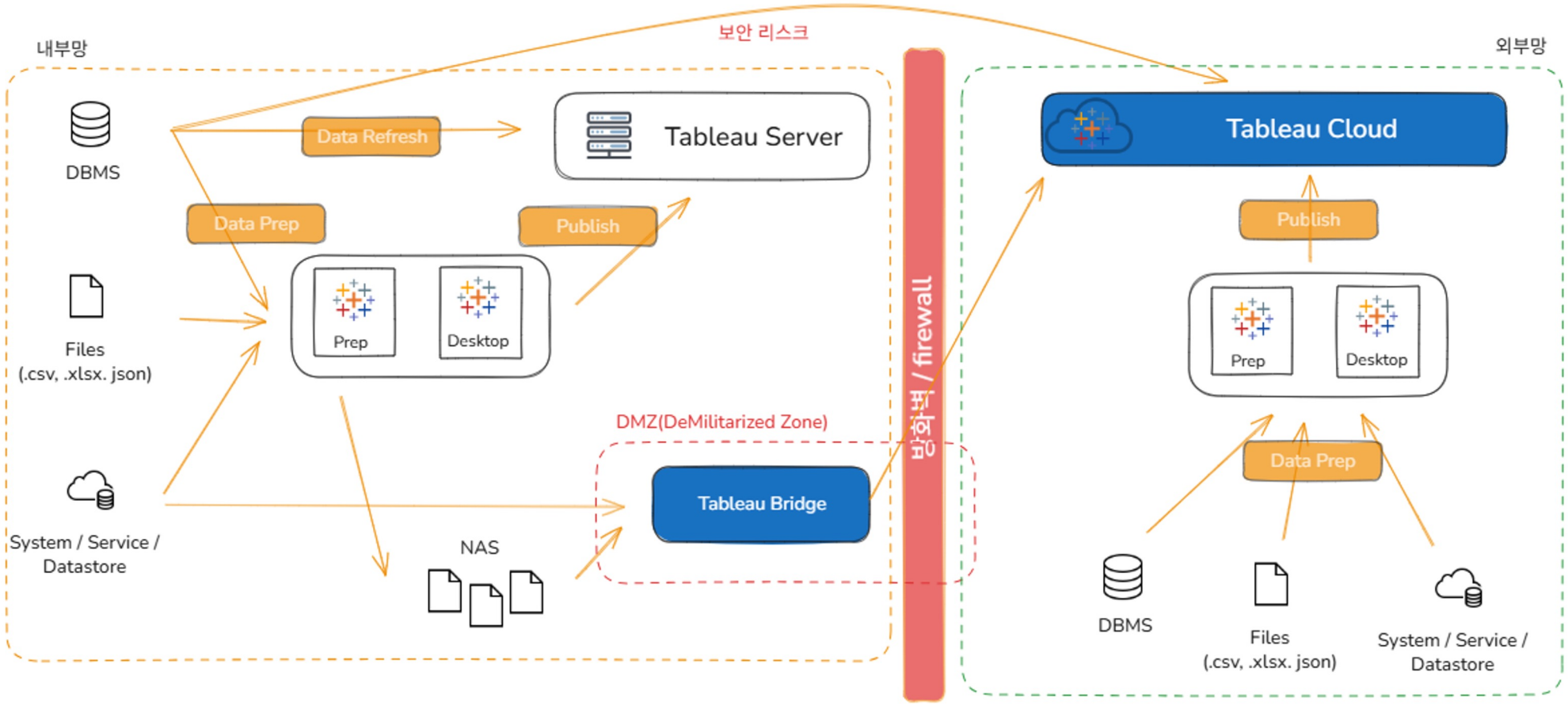


Tableau Cloud vs Tableau Server 주요 차이점

Tableau DataFest

Tableau Cloud (SaaS)

- ✓ 최신 AI 기능 즉시 사용 가능
- ✓ 자연어 질의 기본 탑재
- ✓ 자동 업데이트 및 유지관리
- ✓ Einstein Copilot 완전 지원

Tableau Server (온프레미스)

- ✓ 자체 네트워크 내 데이터 보안
- ✓ 사내 인증 시스템 연동
- ⚠ AI 기능 구현 위한 추가 구성 필요
- ⚠ 수동 업그레이드 및 관리 필요

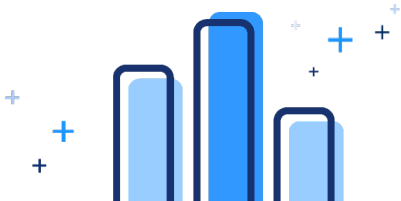


Tableau 온프레미스 환경의 LLM/AI 연동 솔루션

Tableau DataFest

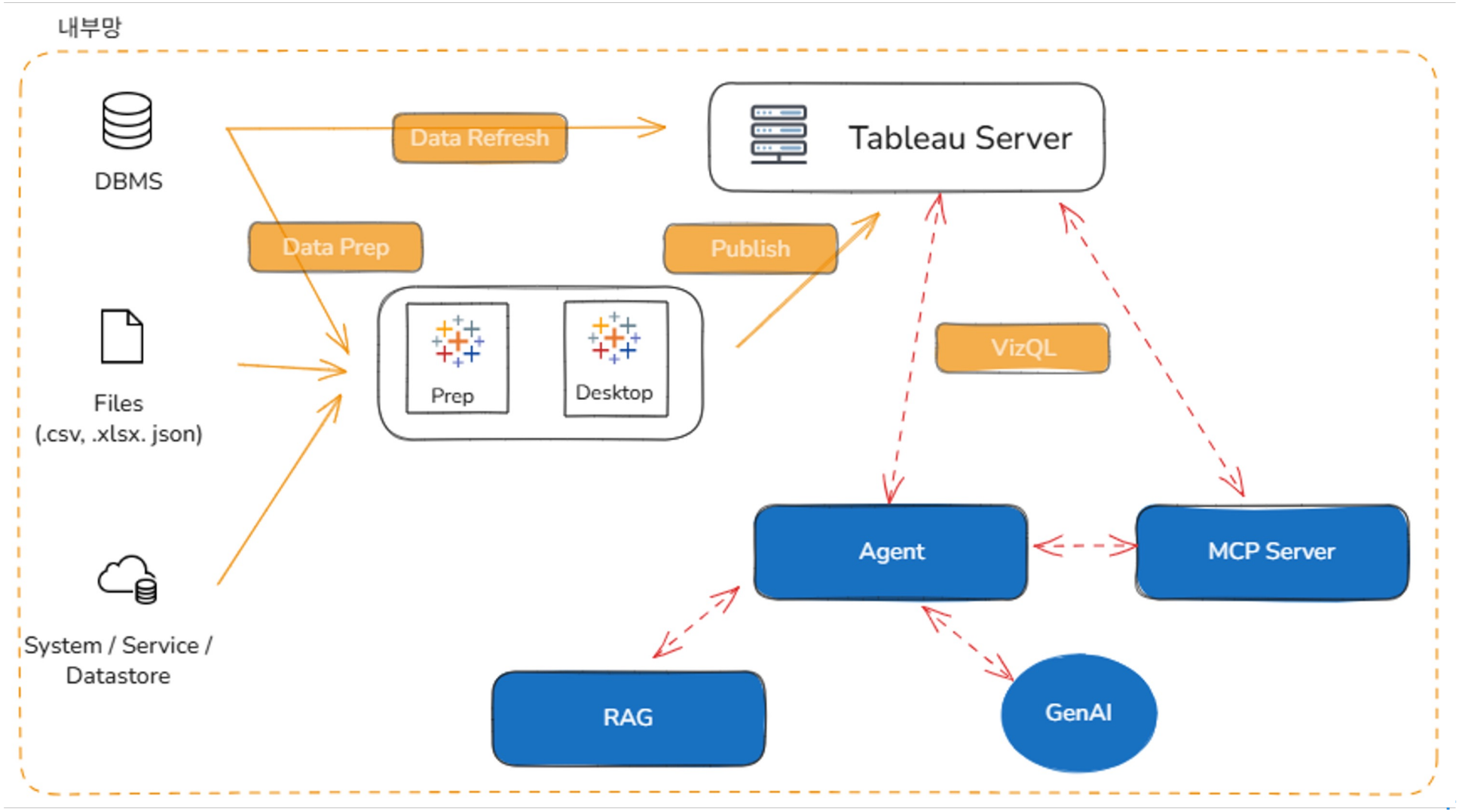


Tableau DataFest

VizQL & VizQL Data Service (VDS)

VizQL Architecture

클릭만으로 데이터를 분석하는 비밀

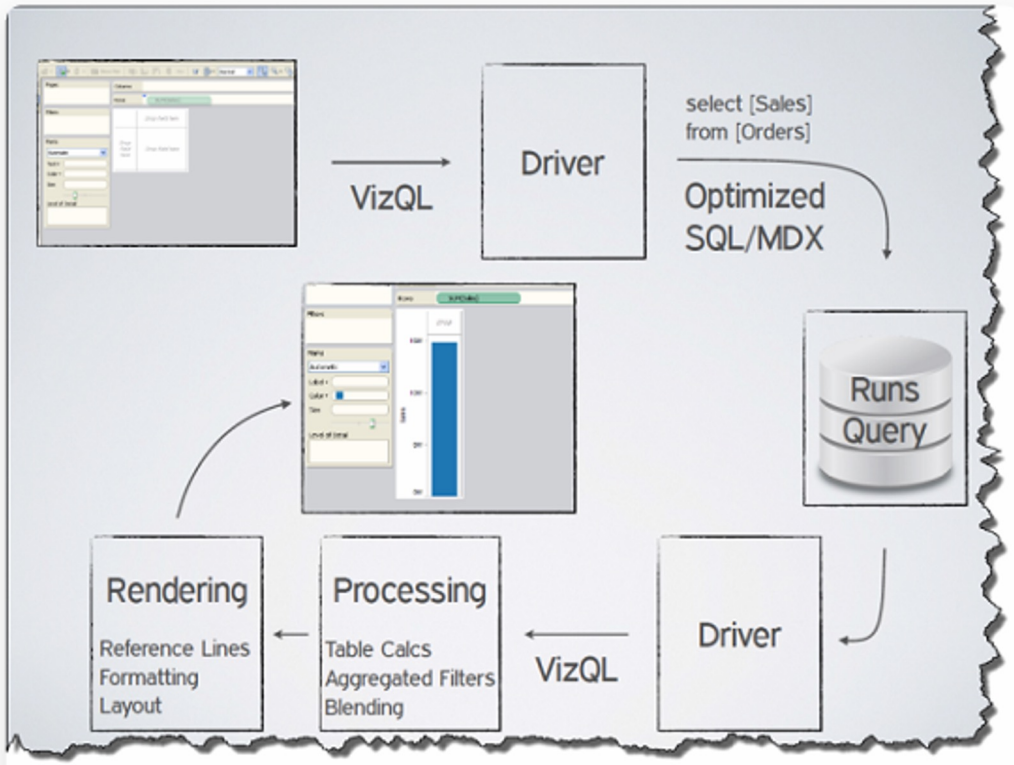
Tableau DataFest

VizQL의 핵심 특징

표준 데이터베이스 쿼리 언어 확장
SQL과 MDX의 기능을 확장하여 시각화 결과 지원

행과 열 구조 정의 구문
표와 시각적 결과의 구조를 명확히 기술

마크 타입과 시각적 속성 제어
사용자가 선택한 마크 타입과 시각적 속성 제어 가능



Tableau에서 간단한 드래그 앤 드롭 작업으로 복잡한 SQL 쿼리가 자동 생성됩니다

VizQL 작동 원리

Tableau DataFest

사용자 동작
드래그 앤 드롭

>

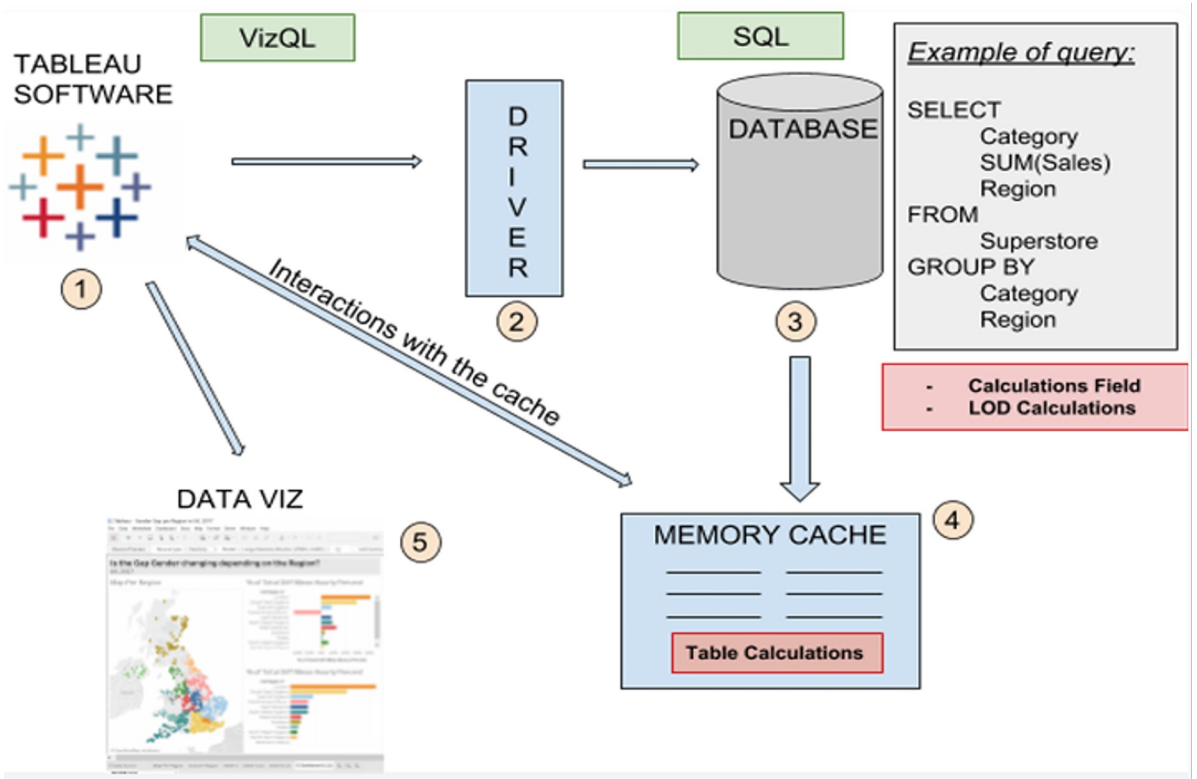
SQL 변환
쿼리 생성

>

시각화
결과 렌더링

VizQL Server의 역할:

- 사용자 작업을 SQL/MDX 쿼리로 해석
- WHERE, ORDER BY, GROUP BY 절 자동 생성
- 행/열 구조와 마크 타입 추상화



VizQL 아키텍처 다이어그램: 시각적 동작의 변환 흐름

VizQL 처리 과정:

1. 사용자 인터페이스에서 시각적 요소 조작
2. VizQL 언어로 사용자 작업 해석
3. 데이터베이스 질의 생성 및 실행
4. 결과 데이터 수신 및 시각적 렌더링
5. 사용자에게 시각화된 결과 제공

https://www.tableau.com/ko-kr/drive/what-is-vizql

전통 BI vs Tableau VizQL

Tableau DataFest

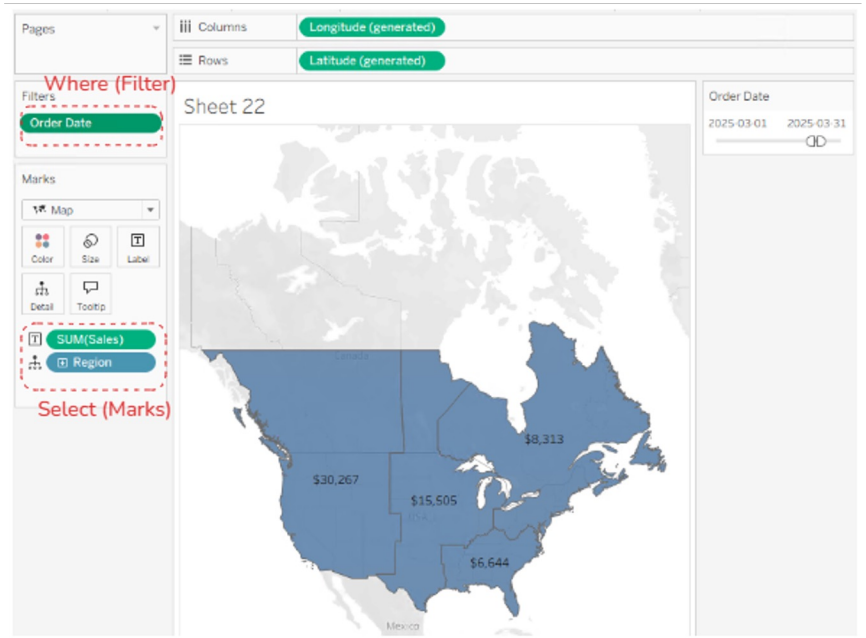
전통 BI

- 분석가는 직접 SQL 작성
- 쿼리 복잡 → 진입 장벽 높음
- 재사용·협업 어려움

Tableau VizQL

- Drag & Drop → 자동 SQL 변환
- SQL을 몰라도 분석 가능
- 시각적 인터페이스로 데이터와 대화

```sql
SELECT Region, SUM(Sales)
FROM Orders
WHERE OrderDate BETWEEN '2025-03-01' AND '2025-03-31'
GROUP BY Region;
```



마크 하나 = 데이터베이스 질의

Tableau DataFest

점(Point)

→ SELECT / GROUP BY

```
SELECT category, COUNT(*)
FROM sales
GROUP BY category
```

막대(Bar)

→ SUM / COUNT

```
SELECT region, SUM(sales)
FROM transactions
GROUP BY region
```

선(Line)

→ ORDER BY / 시계열

```
SELECT date, SUM(revenue)
FROM monthly_sales
ORDER BY date
```

색상/크기(Size/Color)

→ CASE WHEN / 조건 / 순위

```
SELECT product,
CASE WHEN profit > 1000 THEN 'High'
ELSE 'Low' END as profit_level
```

"Tableau의 시각적 요소는 단순 그림이 아니다. VizQL은 마크를 데이터베이스 질의로 번역한다."

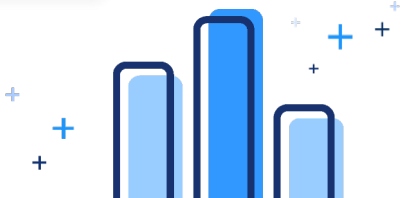


Tableau VizQL TSM 명령 설정

Tableau DataFest

>_ TSM 명령어 모음

현재 인스턴스 확인:	`tsm topology list-nodes -v`
인스턴스 수 설정:	`tsm topology set-process -n node1 -pr vizqlserver -c 1` `tsm pending-changes apply`
외부 접근 허용:	`tsm configuration set -k wgserver.trusted_hosts -v "111.222.33.44"` `tsm pending-changes apply`
VizPortal 확인:	`tsm configuration get -k vizportal.enabled` (False일 경우 대시보드 접속 불가 → 활성화 필요)

tsm pending-changes apply를 마지막에 한번만 적용하시면 됩니다.

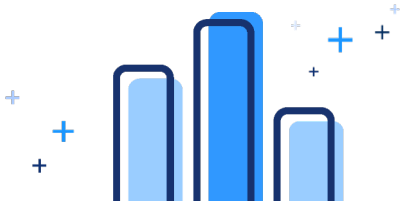


Tableau VizQL Data Service (VDS)

대시보드 없는 Tableau

Tableau DataFest

VDS의 주요 특징

- REST API 기반의 데이터 접근 인터페이스
- 퍼블리시된 데이터소스에 대해 필터링된 결과 반환
- 대시보드 없이도 활용 가능 → 백엔드 연동 최적화
- Python, JavaScript 등 다양한 언어와 연동 가능

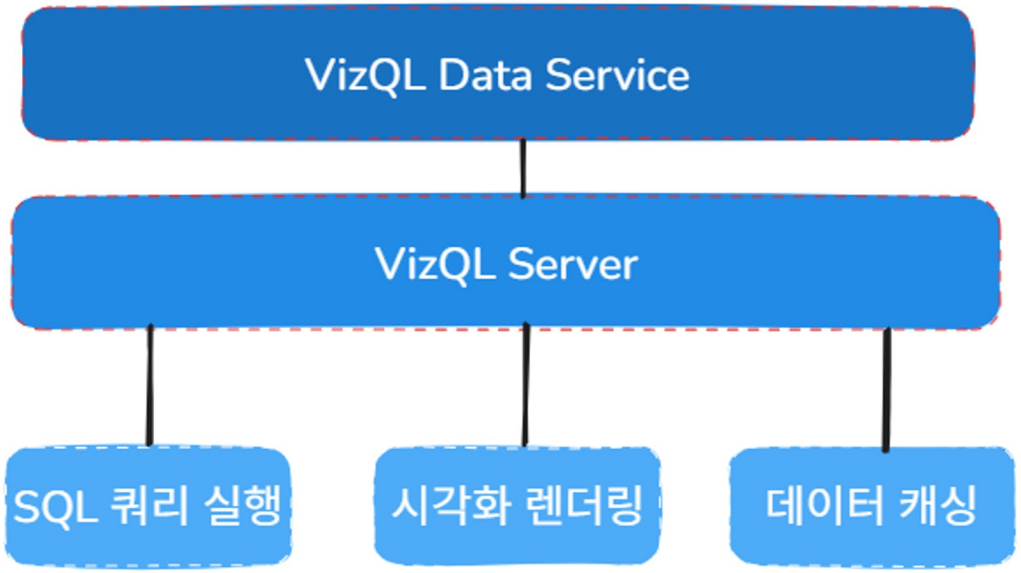


Tableau VDS 설정 방법

Tableau DataFest

VDS 활성화 및 설정

VDS 기능 활성화:

```
tsm configuration set -k
features.VizqlDataServiceDeployWithTsm -v true
tsm pending-changes apply
```

로그 레벨 확인 및 디버깅:

```
tsm configuration get -k vizqlserver.log.level
```

(기본은 'info', 필요시 'debug'로 설정 가능)

VDS 요청 시 중요 파라미터

- `dataSourceId`: 데이터 소스 식별자
- `fields`: 조회할 필드 목록
- `filters`: 데이터 필터링 조건
- `groupBy`: 집계 기준 설정
- `aggregates`: 집계 함수 지정

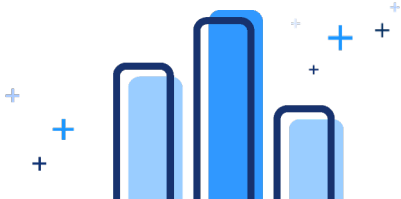


Tableau DataFest

Tableau MCP

Tableau MCP 기능

LLM과 Tableau를 잇는 브릿지

Tableau DataFest

Tableau를 AI와 연결해주는 Node.js 기반 미들웨어

- **MCP = Model Context Protocol**
  AI 모델과 애플리케이션 간 표준화된 통신 프로토콜
- **자연어 명령(JSON)을 Tableau API 호출로 변환**
  사용자 질문을 VizQL 및 기타 API 호출로 자동 변환
- **구성: MCP Server + Tools + LLM Agent**
  모듈식 아키텍처로 유연한 확장 가능
- **LangChain, OpenAI, Cohere 등과 연동 가능**
  다양한 LLM 및 AI 서비스와 쉽게 통합

LLM → MCP → Tableau 흐름 아키텍처

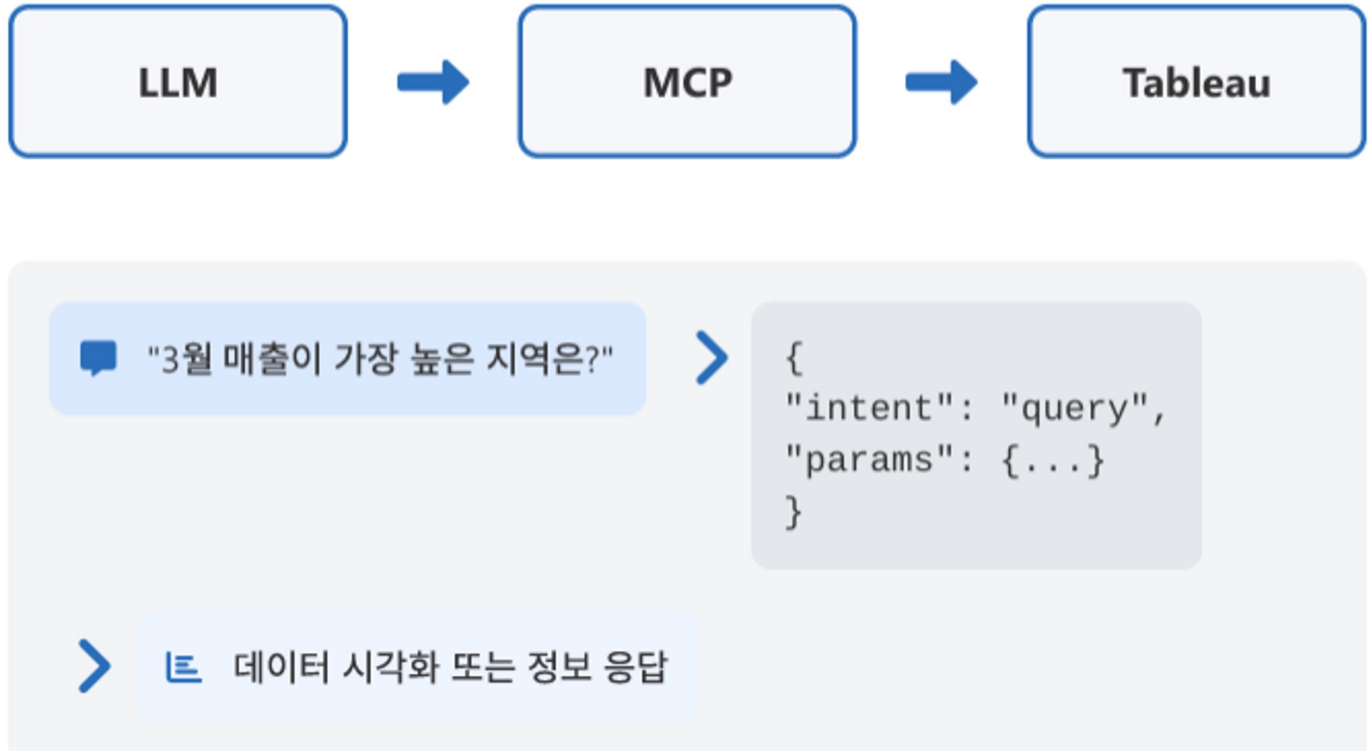


Tableau MCP 구성과 동작

https://github.com/tableau/tableau-mcp

Tableau DataFest

LLM의 요청을 Tableau가 이해할 수 있는 명령으로 변환

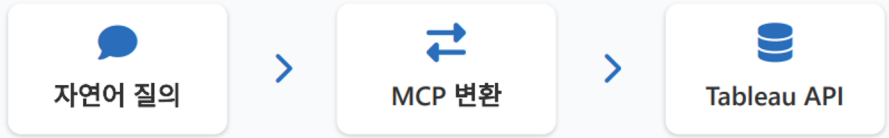
MCP Server
Node.js 기반 명령 처리 서버

Tools 폴더
queryDatasource, listFields 등 기능별 API 실행

Agent 폴더
LangChain 연동 가능

```
$ node dist/tools/query-datasource.js
```

MCP 명령 흐름도



MCP 폴더 구조

- tableau-mcp/
  - src/
    - tools/ - API 실행 도구
    - agents/ - LangChain 연동
    - server.js - MCP 서버 코어
  - package.json

실행 시나리오 예시

1. LLM에서 "3월 매출이 가장 높은 지역은?" 질문 입력
2. MCP가 JSON 명령으로 변환: {"action": "query", "filter": "date=March", "measure": "sales", "groupBy": "region"}
3. queryDatasource.js가 Tableau VizQL Data Service API 호출
4. 결과를 LLM에 반환하여 자연어 응답 생성

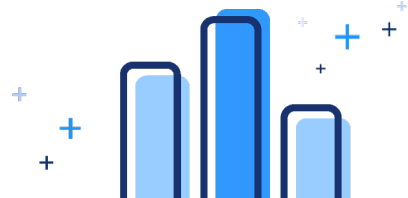


Tableau MCP 확장성

Tableau 환경에서의 활용 확대

Tableau DataFest

Tools 추가

Tableau API 활용 범위 확대

- 쿼리 실행 기능
- 메타데이터 조회 및 관리
- 데이터 추출 및 변환 작업
- 사용자 정의 분석 로직 적용

외부 시스템 연동

Tableau + 외부 시스템

- CRM 데이터 통합 및 분석
- ERP 시스템과 양방향 데이터 흐름
- 사내 데이터 API 연결 인터페이스
- 클라우드 서비스 연계 분석

LLM 교체 가능

다양한 언어 모델 적용 가능

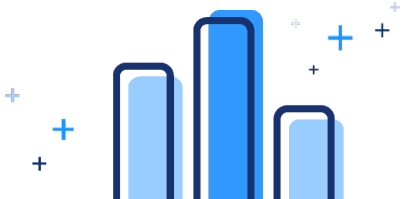
- OpenAI 모델 (GPT-3.5/4 등)
- Cohere 모델 활용
- 사내 자체 LLM 적용
- 최적 성능/비용 모델 선택 유연성

표준화 장점

일관된 패턴 적용

- 자연어 요청 → JSON Intent → Tableau 호출
- 확장 개발의 간소화 및 일관성
- API 변경에 대한 적응력 향상
- 안정적인 확장 생태계 구축

MCP 구조를 통해 Tableau 기능을 확장하고 다양한 시스템과 통합하여 데이터 활용 범위를 넓힐 수 있습니다.



MCP와 Tableau 보안 구조

Tableau DataFest

접속 보안

- Tableau 로그인/SSO 인증
- 인증 토큰 기반 접근 제어
- 세션 관리 및 접속 만료 시스템

권한 제어

- 프로젝트·워크북·데이터소스 단위 Role 관리
- 사용자/그룹 권한에 따라 쿼리 실행 제한
- 기능별 세분화된 접근 권한 설정

데이터 거버넌스

- Row-Level Security (RLS) 적용
- Data Source Filter 설정
- 민감 데이터 접근 제어 (예: 부서별/지역별 제한)

MCP 연동 장점

- LLM이 Tableau API를 호출해도 사용자 권한 범위 내에서만 응답
- 데이터 유출 위험 최소화
- 기존 보안 인프라 활용으로 안전한 AI 확장

"MCP는 Tableau 보안 프레임워크 위에서 동작한다.
LLM이 요청해도, 권한 없는 데이터는 절대 보여지지 않는다."

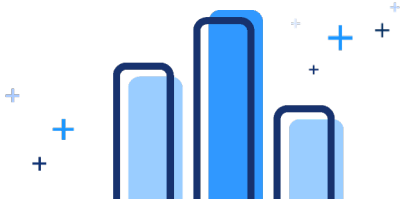


Tableau DataFest

Tableau VDS vs Tableau MCP

VDS vs MCP 차이

같은 기능(메타데이터 조회, 쿼리 실행)을 서로 다른 프로토콜로 노출

① end-point 다름

ⓐ VDS: /api/v1/vizql-data-service/...
ⓑ MCP: /tableau-mcp/

② 호출 방식 다름

ⓐ VDS: REST (method + path)
ⓑ MCP:
JSON-RPC (method=tools/call + params)

③ 공통점

ⓐ 모두 Tableau Server 엔진(VizQL/VDS)을 활용
ⓑ 쿼리 실행 & 메타데이터 조회 지원

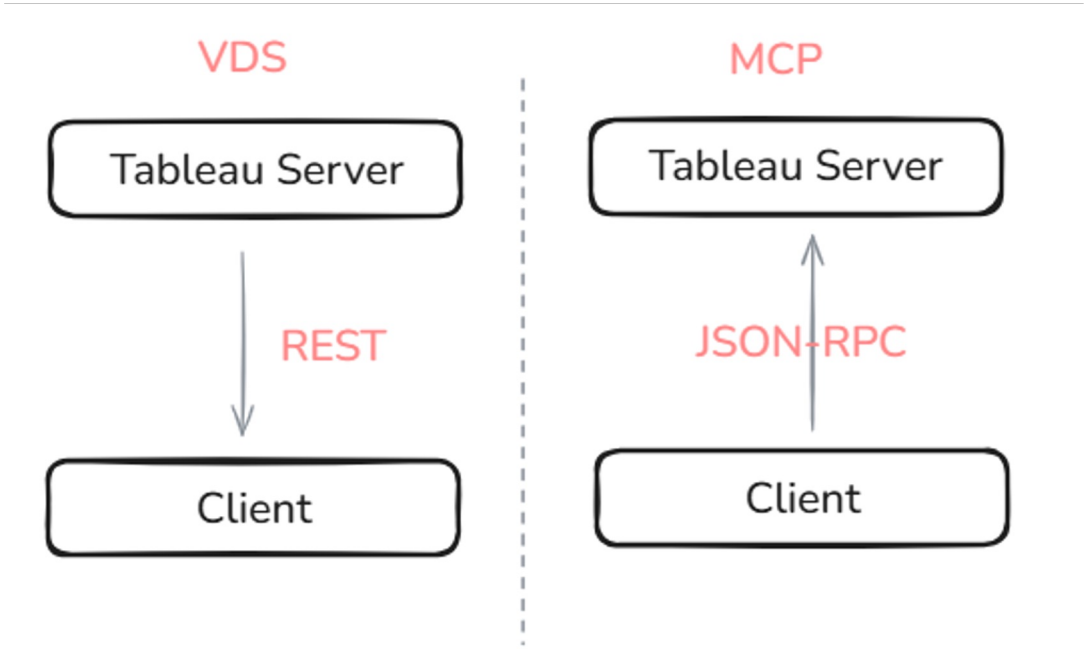


Tableau VDS & MCP 주요 Endpoint

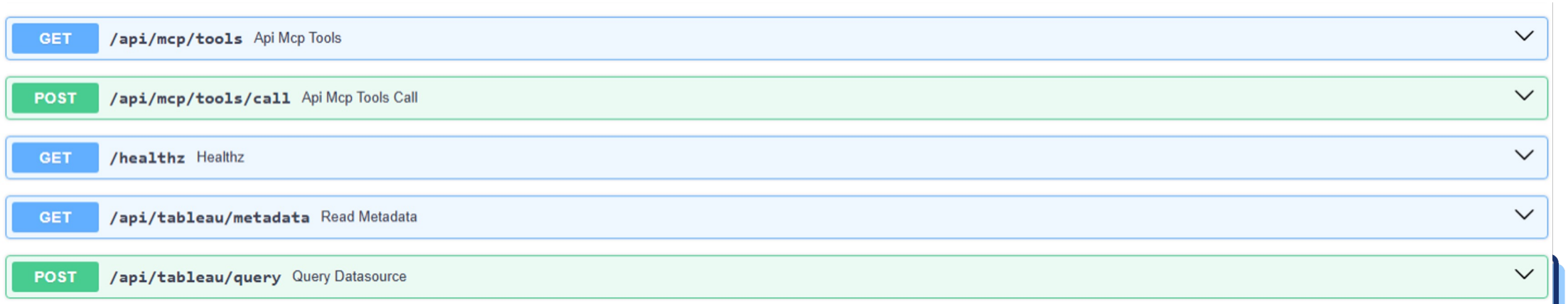
Tableau DataFest

① VDS 관련

ⓐ /api/tableau/query – 데이터소스 질의
ⓑ /api/tableau/metadata – 메타데이터 조회

② MCP 관련

ⓐ /api/mcp/tools – MCP 툴 목록
ⓑ /api/mcp/tools/call – MCP 툴 호출




FastAPI (Swagger)
VDS

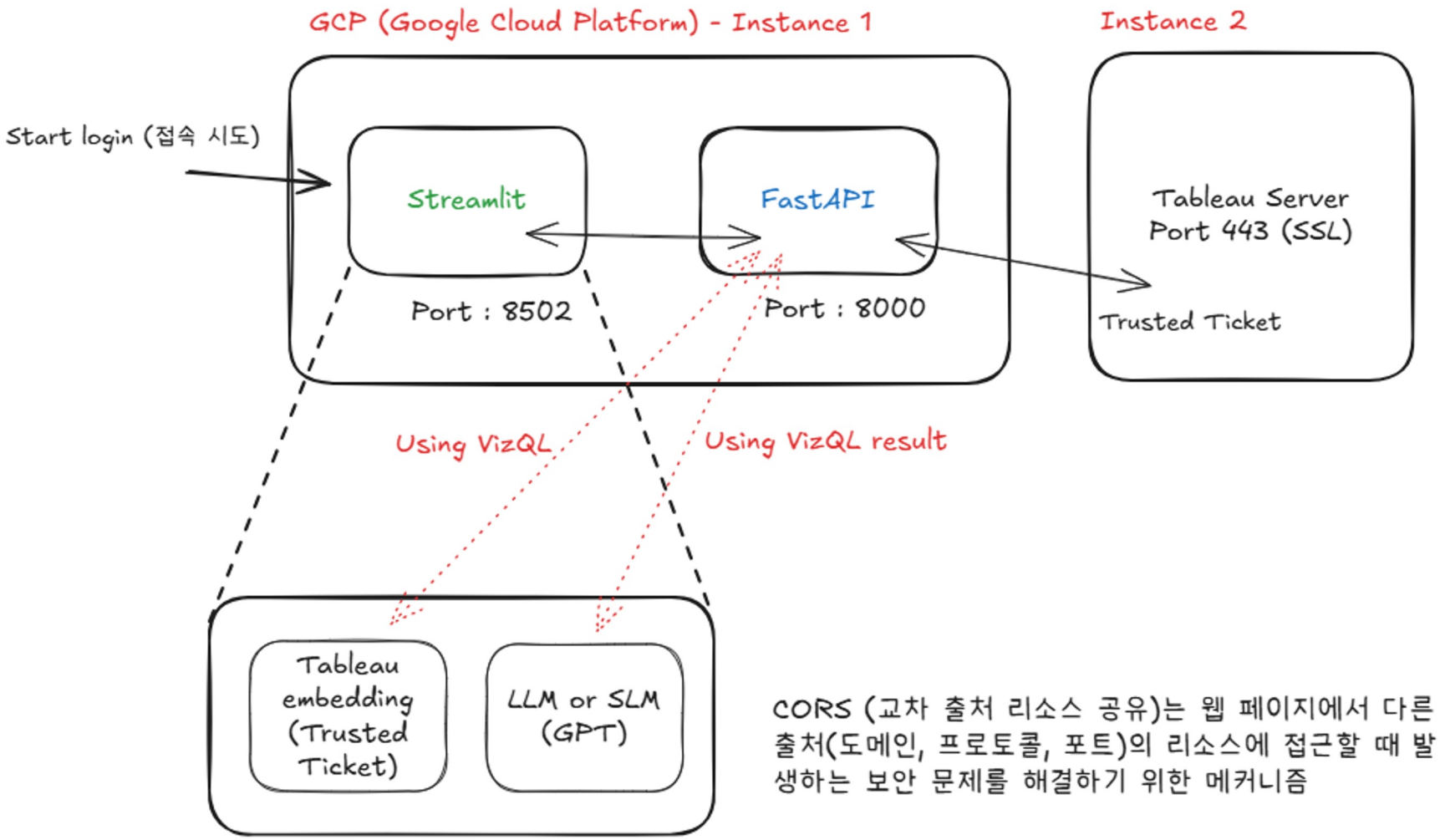
Tableau DataFest

Tableau + Agent

Demo #1

LLM + Tableau 전체 Architecture

SaaS 기반 이중 서버 구조와 시스템 통합



CORS (교차 출처 리소스 공유)는 웹 페이지에서 다른 출처(도메인, 프로토콜, 포트)의 리소스에 접근할 때 발생하는 보안 문제를 해결하기 위한 메커니즘

Trusted Ticket 기반 인증 이해하기

login

Tableau Server가 발급하는 일회성 티켓으로 안전하게 접근합니다.

- 외부 웹앱이나 Streamlit에서 Tableau에 접근할 때 Trusted Ticket 방식 사용
- Tableau Server는 등록된 IP에 대해서만 티켓 발급
- iframe + Trusted Ticket 조합은 임베디드 분석에 최적

Trusted Ticket 발급 흐름도

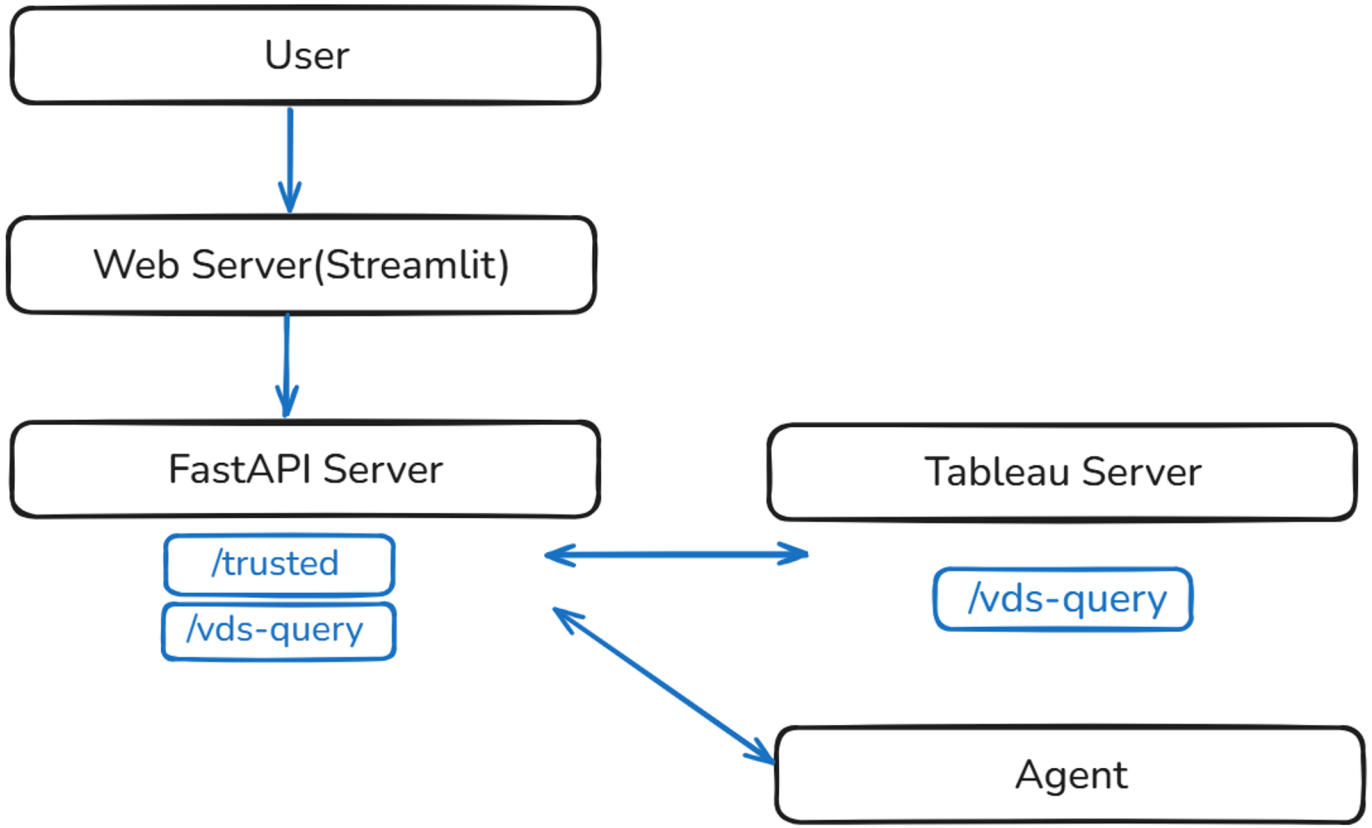
이점 및 주의사항

- 외부 포털에 Tableau 분석 임베딩 가능
- 사용자별 맞춤 대시보드 동적 렌더링
- 신뢰할 수 있는 IP 설정 필수 (wgserver.trusted_hosts)
- 티켓은 일회성, 즉시 사용 필요

Fast API 서버 구성

❷ Backend Server

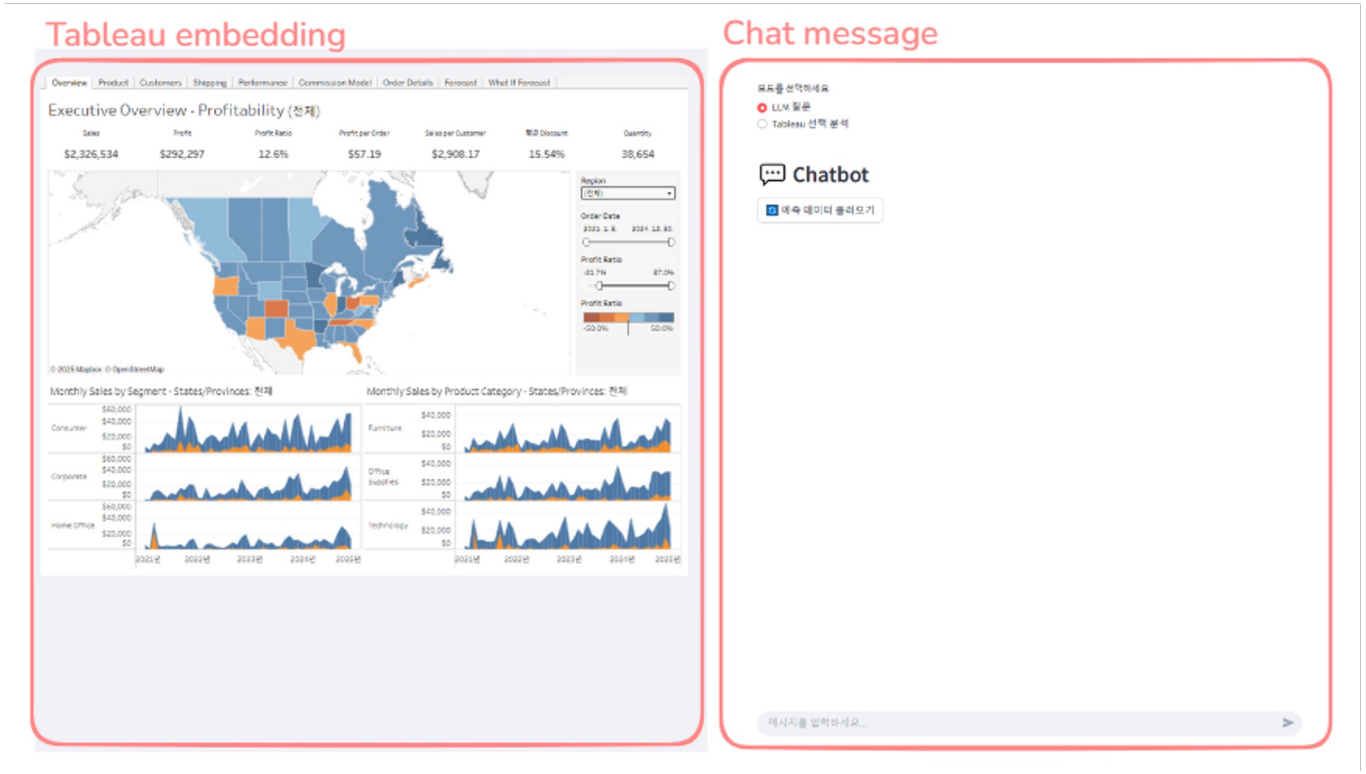
① 사용자 인증 요청
② Tableau Ticket 발급 요청
③ VizQL data service 요청
④ Agent



Streamlit 연동 구조

❸ UI interface

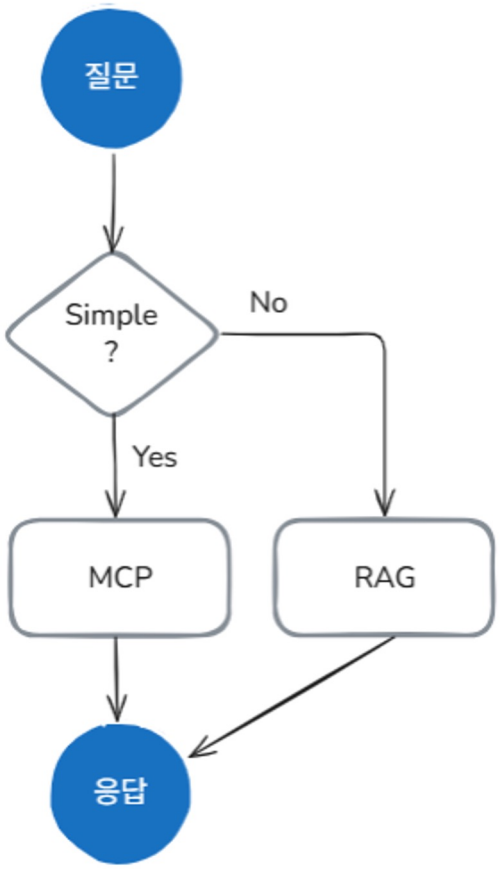
① Login
② FastAPI에 Trusted Ticket 요청
③ 발급된 티켓으로 iframe URL 구성
④ Tableau embedding
⑤ Chatbot message
⑥ FastAPI로 Chat message 전달



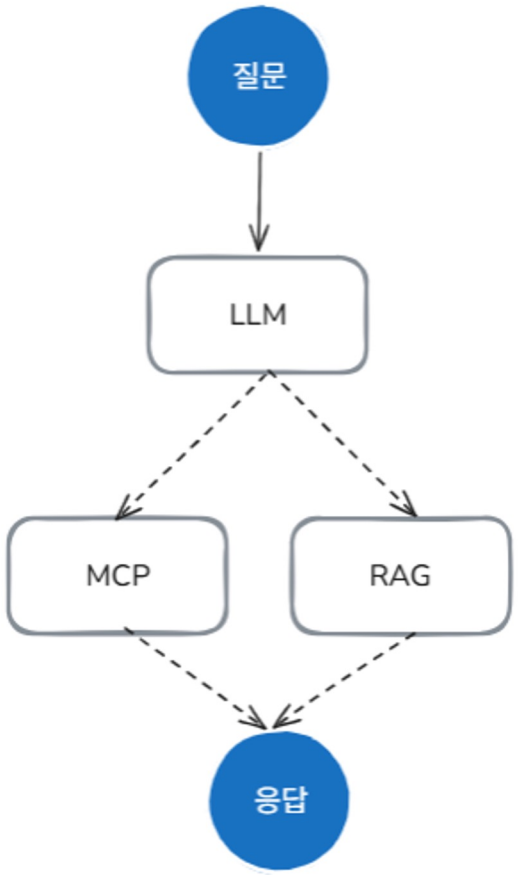
MCP + RAG

Agent

ⓐ MCP + RAG를 선택적으로 연결하는 Agent 구성



ⓑ Agentic Agent 구성 방식



Demo #2

CORS (Cross-Origin Resource Sharing) 회피 전략

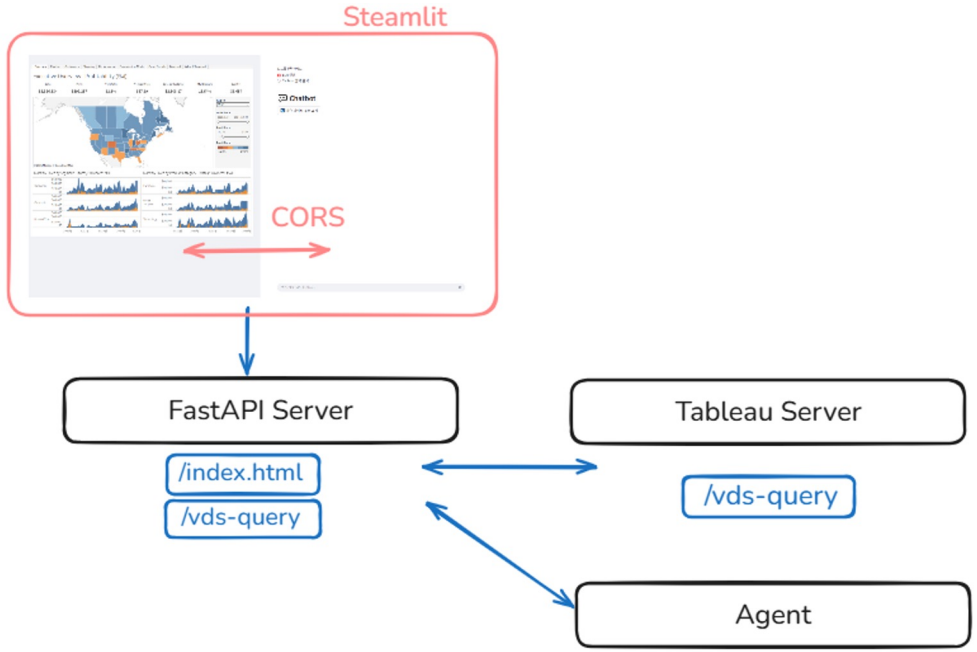
Tableau DataFest

5 Index.html 활용

① 브라우저는 보안을 위해 외부 도메인/포트로 직접 요청하는 것을 차단 (예: Tableau Server)

② FastAPI가 중계 서버 역할을 하여 CORS 회피 (동일 출처로 위장)

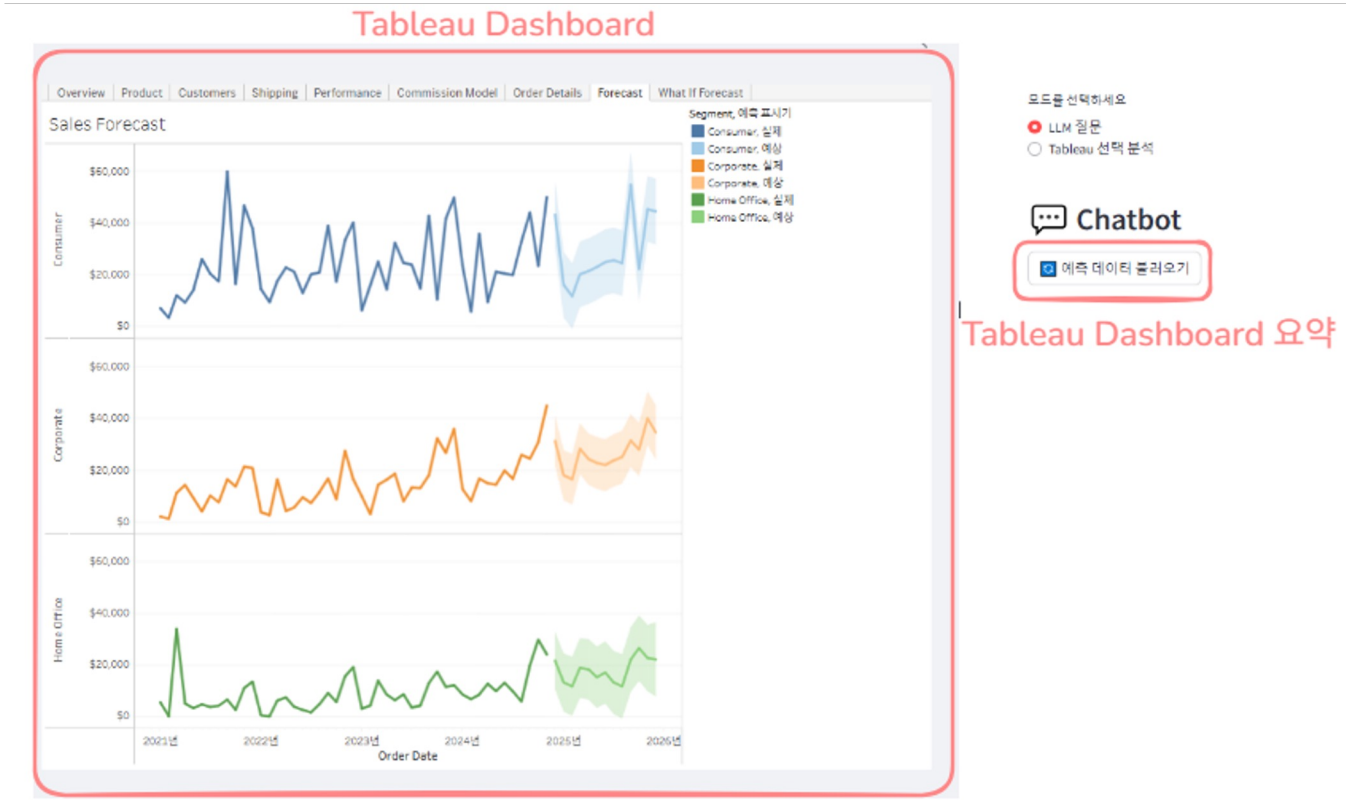
③ index.html은 FastAPI와 동일한 출처에서 열려 있기 때문에, 브라우저에서 CORS 문제 없이 호출 가능



VDS 활용 #1

⑥ Dashboard 요약

① index.html에서 dashboard ID 전달
② FastAPI /sheet-summary 호출 → Tableau VDS에서 시트 메타데이터 수집
③ 필드 정보, 시각화 유형, 데이터 관계 등을 추출
④ LLM 프롬프트 구성하여 요약 설명 생성

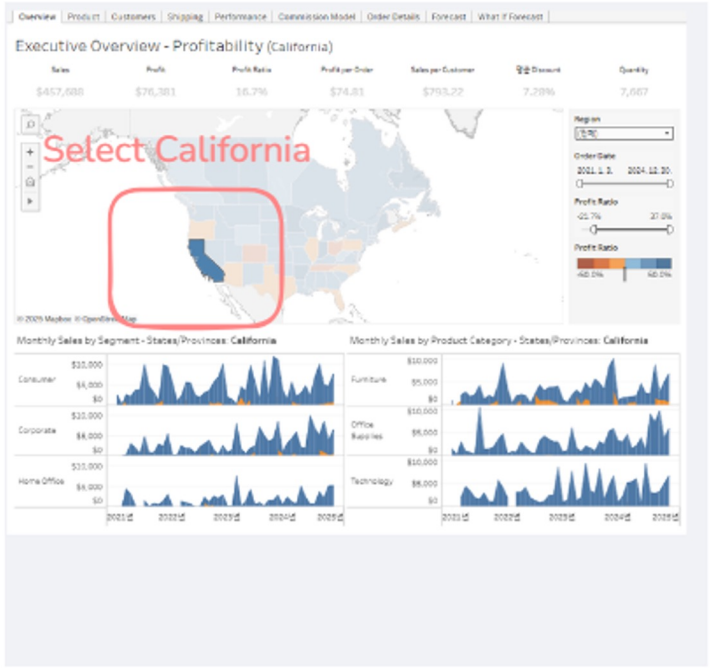


Demo #3

VDS #2

Dashboard Action

① 사용자가 특정 영역 또는 셀 선택
② FastAPI가 /mark-selection 호출하여 선택된 데이터 수집
③ 필드, 조건, 쿼리 필터 분석
④ 자연어 기반 설명 생성



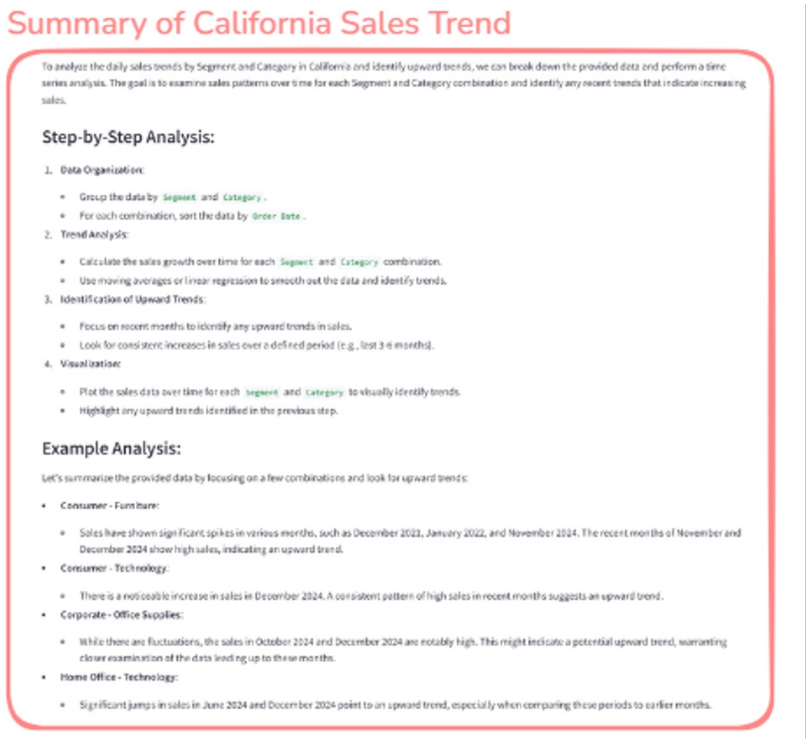


Tableau DataFest