SNOWFLAKE
Hands-on for AI Workspace

Senior Solution Engineer

Sep 2025

Agenda

01. data catalog
- Git 연계
- 데이터 적재
- Horizon catalog 에서 asset 검색 및 관리

02. data pipeline
- Dynamic table 을 활용한 데이터 파이프라인 구성
- 수집, 처리, 적재 자동화 및 작업 스케줄링

03. data endpoint API
- REST API 로 snowflake 에 request 해서 데이터 얻기
- Application 구성

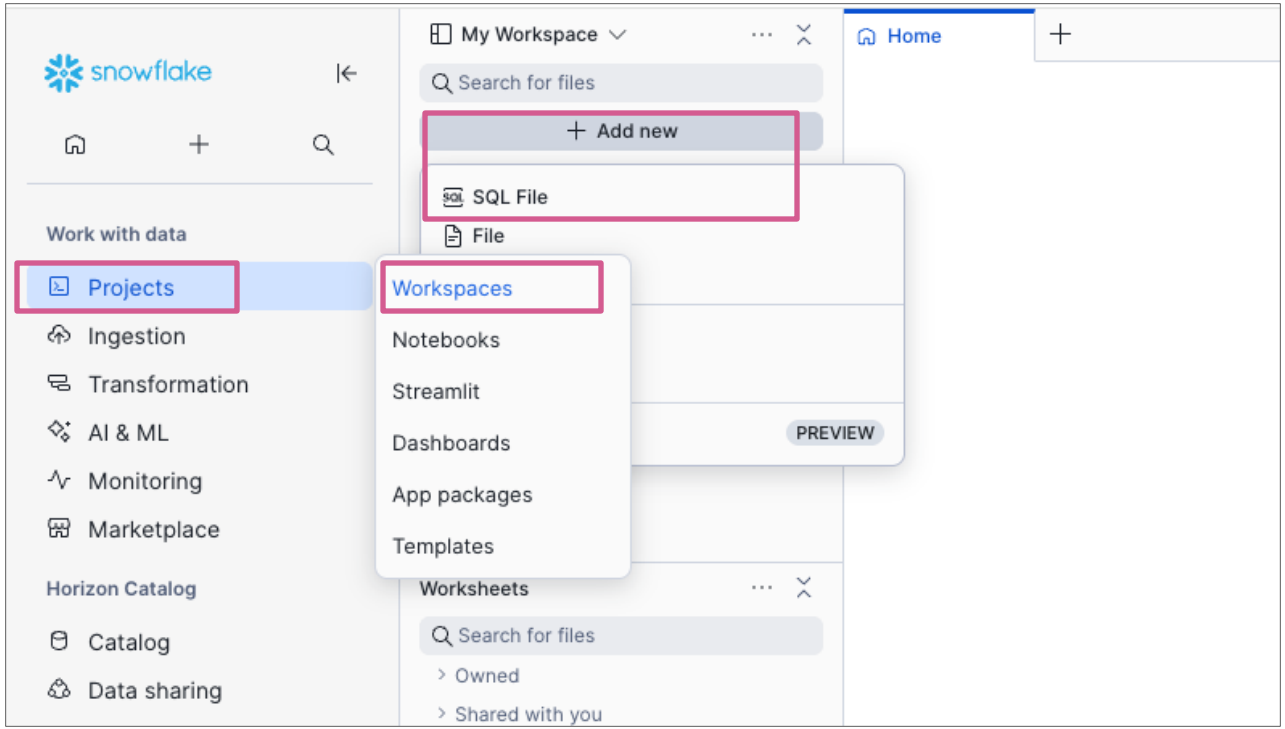
04. rbac
- Row access policy 를 활용한 역할기반 접근제어 구성
- 펀드 별 테이블 차단 정책 구현

05. python notebook
- Git 연계
- 노트북 생성 및 실행
- Snowpark Dataframe vs. Snowpark Pandas Dataframe vs. Native Pandas

06. cortex AI
- Snowflake Intelligence 구성 (Text to SQL)
- Snowflake 에서 호출 가능한 커스텀 API 연동

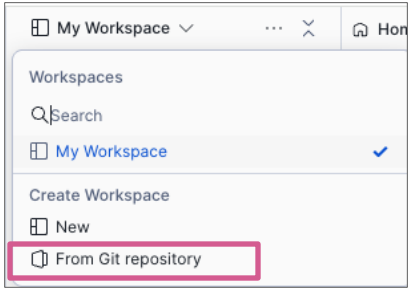
01. data catalog

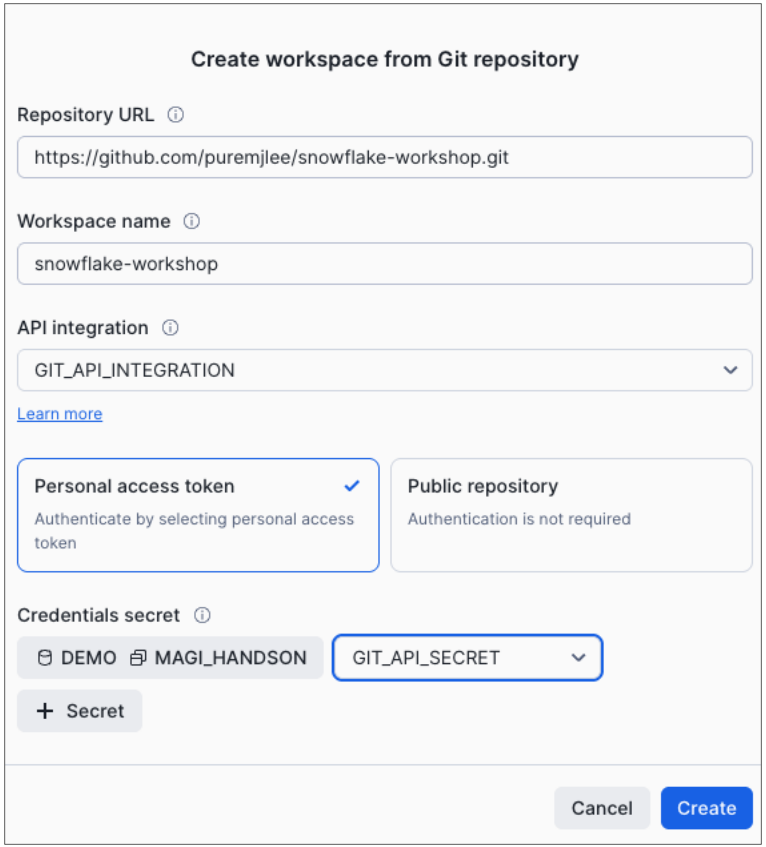
Workspace 에디터



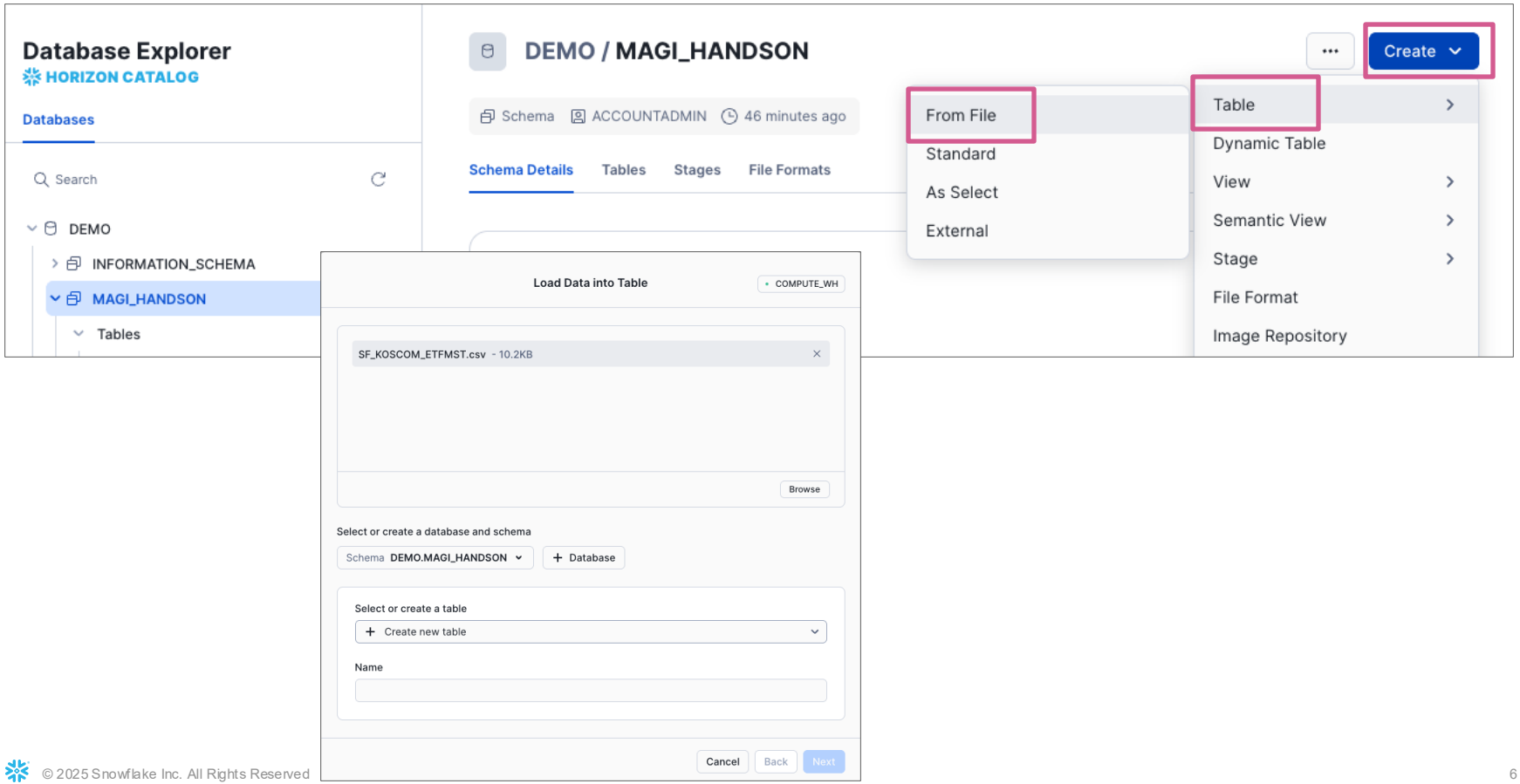
Git 연계

```
create database demo;
create schema magi_handson;
use warehouse compute_wh;

// 자신의 git access 정보로 대체
CREATE OR REPLACE SECRET git_api_secret
TYPE = password
USERNAME = 'user_name'
PASSWORD = 'xxxxxxxxxxxxxxx';

// 자신의 git 정보로 대체
CREATE OR REPLACE API INTEGRATION git_api_integration
API_PROVIDER = git_https_api
API_ALLOWED_PREFIXES = ('https://github.com/xxxxxxxxxx')
ALLOWED_AUTHENTICATION_SECRETS = (git_api_secret)
ENABLED = TRUE;

// 확인
desc INTEGRATION git_api_integration;
```

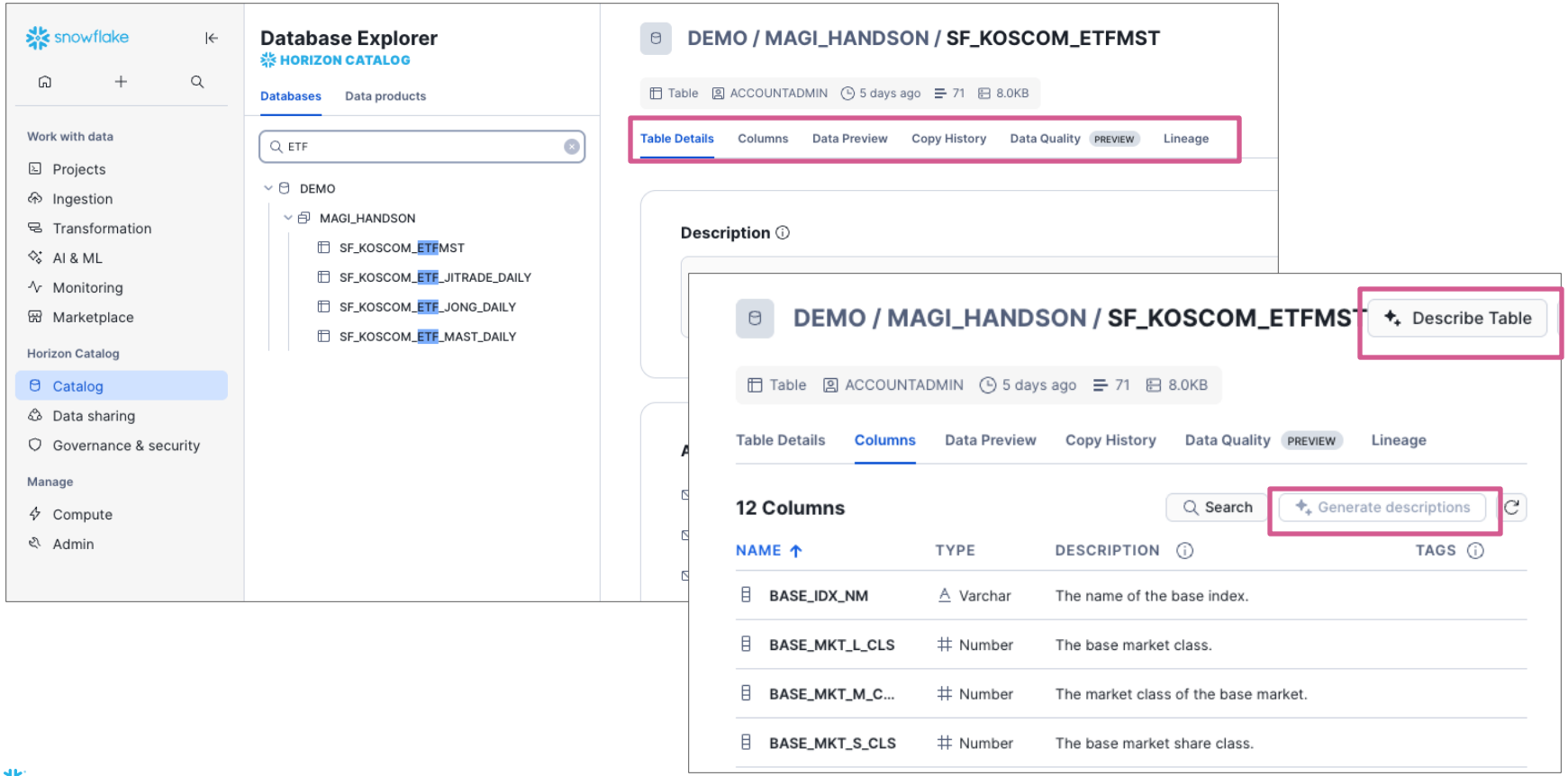




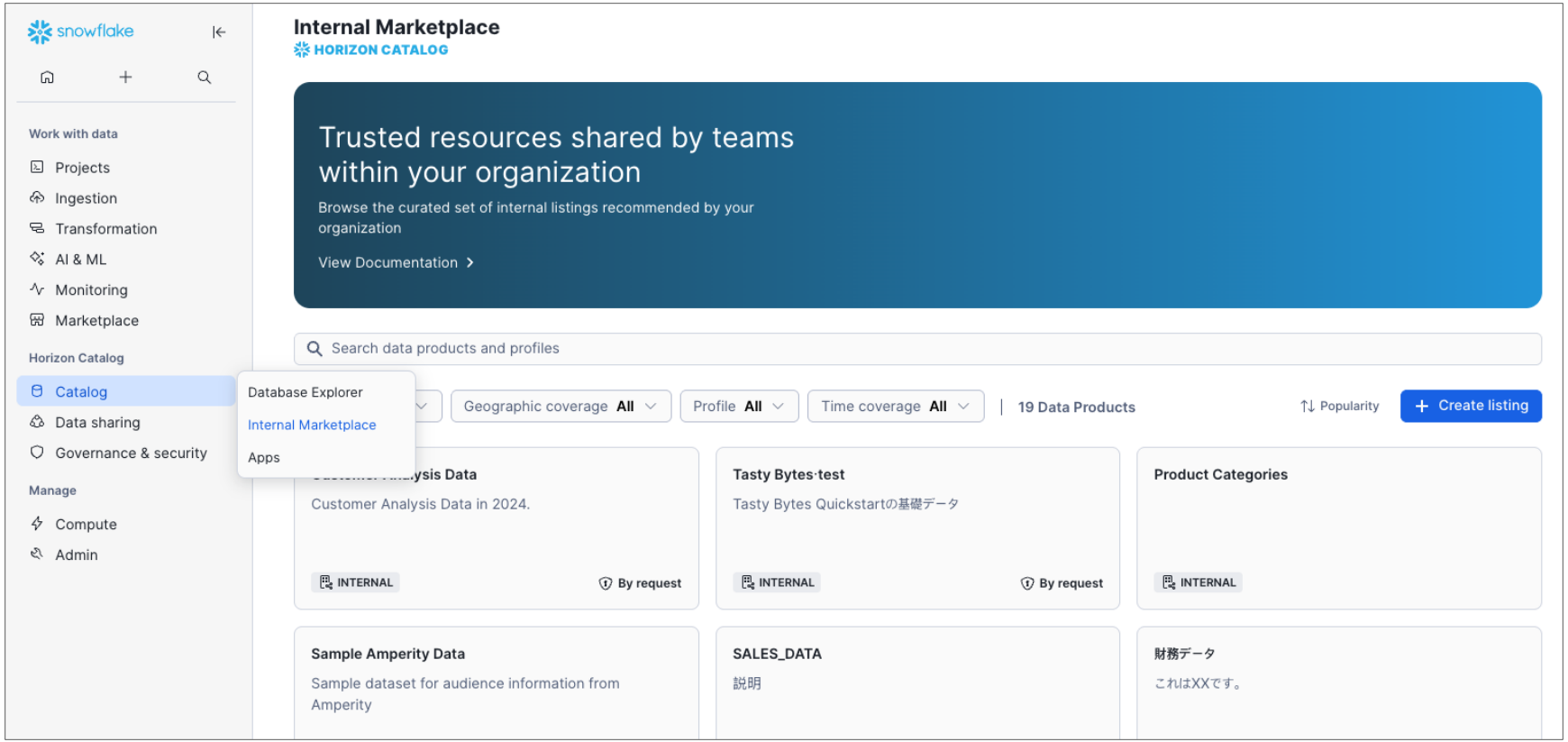
Data loading



Horizon Catalog



Horizon Catalog



Lab 01

- 데이터 로딩
- Gen AI 로 테이블/컬럼 description 생성

02. data pipeline

Stream

- 테이블에 insert, update, delete 를 포함한 DML 오퍼레이션을 자동으로 감지하는 객체로서, CDC 목적으로 사용
- Stream 객체에는 소스에 대한 오프셋만 저장되며, 버전 히스토리를 활용해서 CDC 레코드를 반환

```
create stream s1 on table t1;
```

<Table t1>

col1	col2
orange	2
banana	3
apple	9

<Stream s1>

col1	col2	METADATA$ACTION	METADATA$ISUPDATE	METADATA$ROW_ID
orange	2	INSERT	FALSE	d0a551cecbee0f9ad2b8a9e81bcc33b15a525a1e
banana	3	INSERT	FALSE	22e61a8d5d7db667b5674527252502b129d88c7e
apple	5	DELETE	TRUE	17ccc3966ddc95be4b0a563283f3dd66953e3ed8
apple	9	INSERT	TRUE	17ccc3966ddc95be4b0a563283f3dd66953e3ed8

Task

- 주기적으로 단일 SQL 문 실행 혹은 Stored Procedure 호출하는 객체
- DAG 구조로 여러 tasks 를 연속적으로 실행 가능

```
create task run_task
    <warehouse>
    <schedule>
    <after>
as
    insert into tar_table
    select key, plt_cd, ...
    from stream_a;

    or

    call root_run('pivot_key');
```

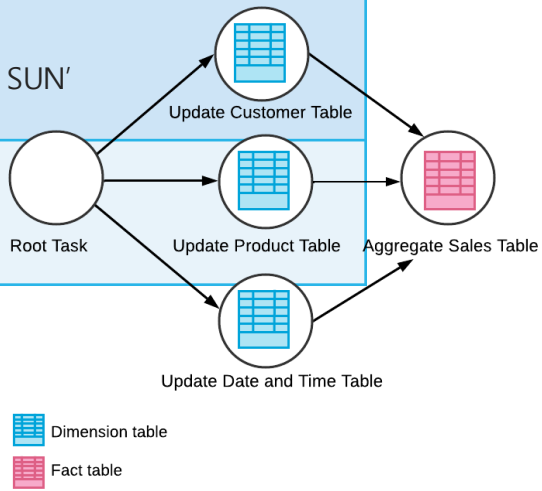
WAREHOUSE = HT_SMALL
: 지정한 고객 Warehouse를 기반으로 동작

USER_TASK_MANAGED_INITIAL_WAREHOUSE_SIZE = SMALL
: Serverless 방식으로 수행

SCHEDULE = '5 minute'
: 반복적 수행

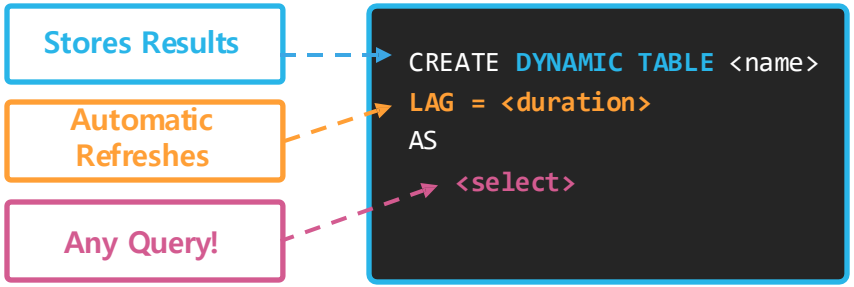
SCHEDULE = 'USING CRON 0 9-17 * * SUN'
: 시스템 시간 기반 동작

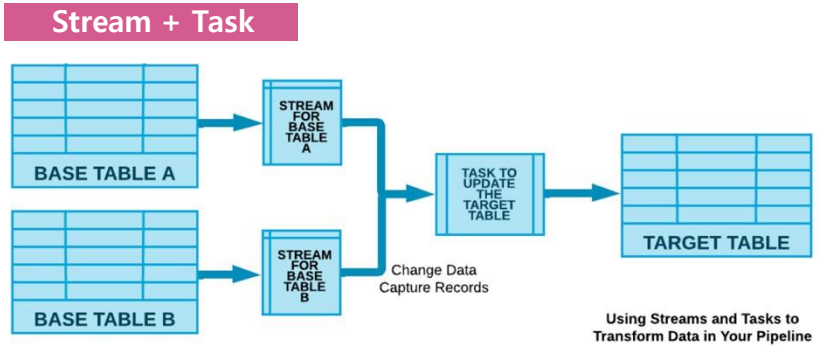
AFTER pre_task
: 각 Task 간의 선/후행 관계 정의

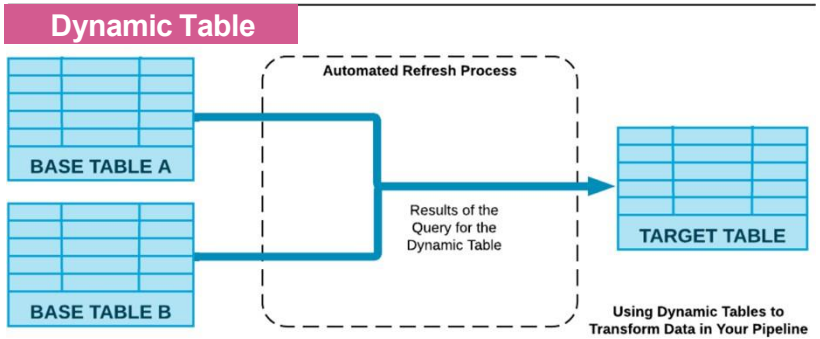


Dynamic Table

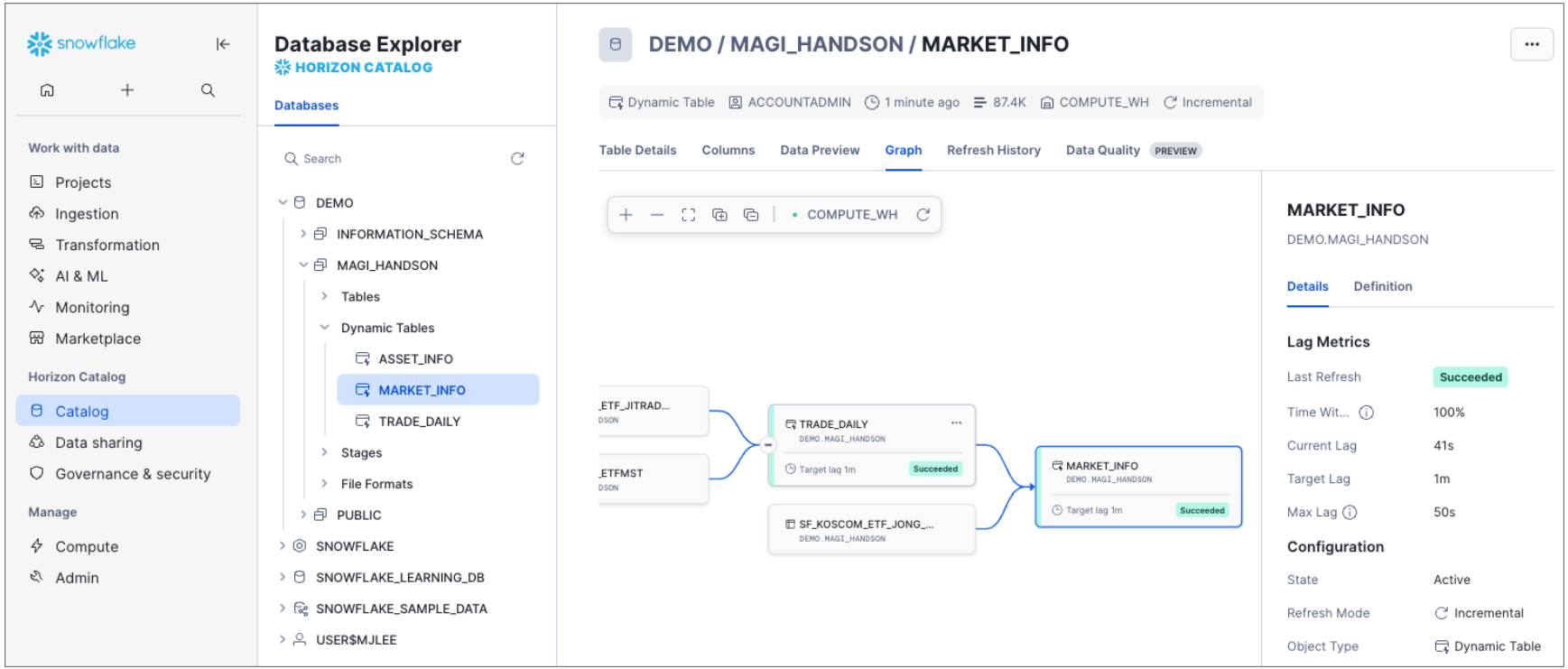
- 선언적 데이터 파이프라인
- 최종 타겟 테이블을 정의함으로써 복잡한 데이터 파이프라인을 관리할 필요가 없음
- 쿼리 결과 자체를 materialized 해서 저장
- Stream 과 Task 대체 가능
- 파이프라인 자동화를 위해 사용
- DAG 구성 가능. Lag 존재



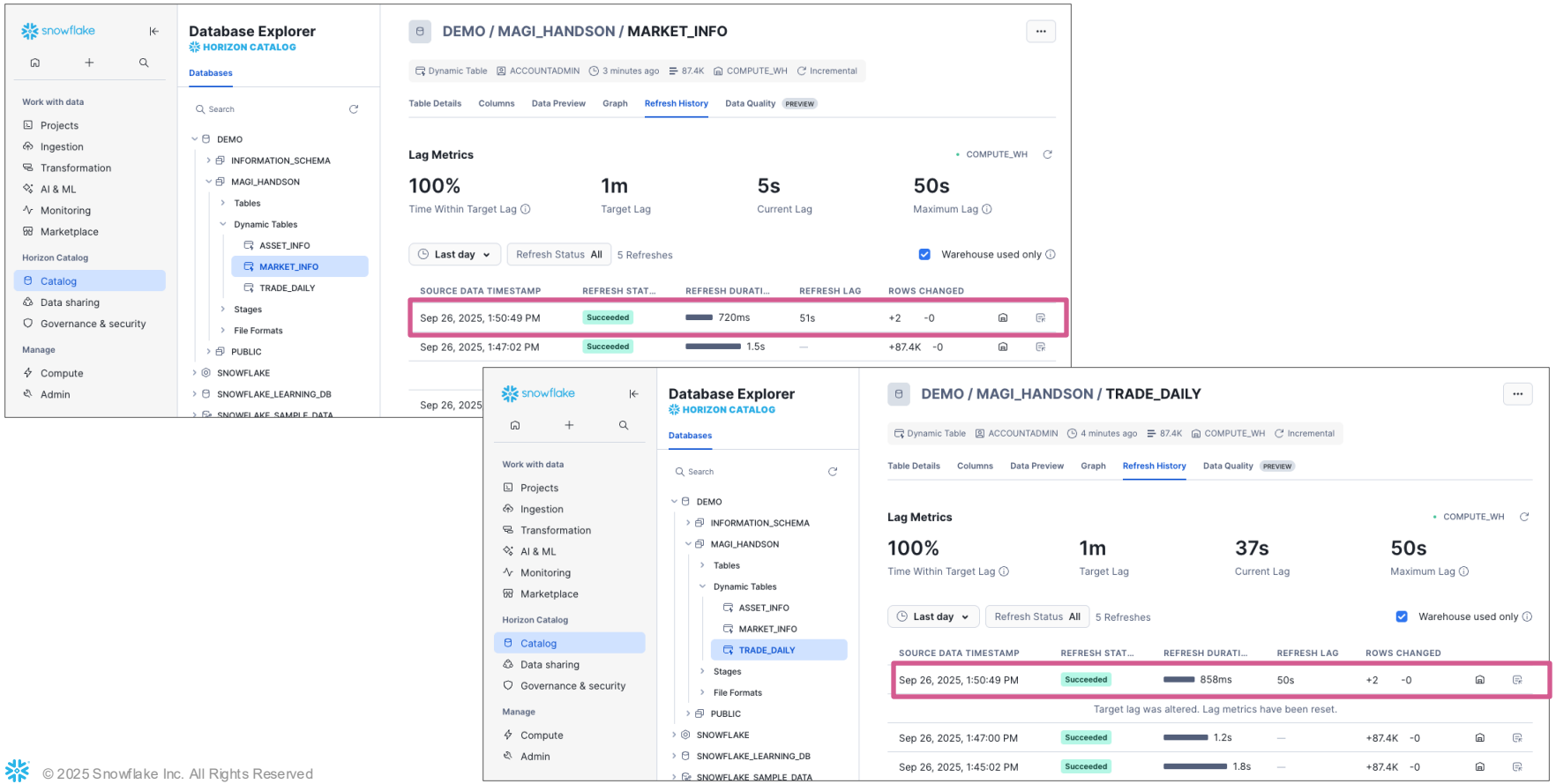




Dynamic Table - Graph

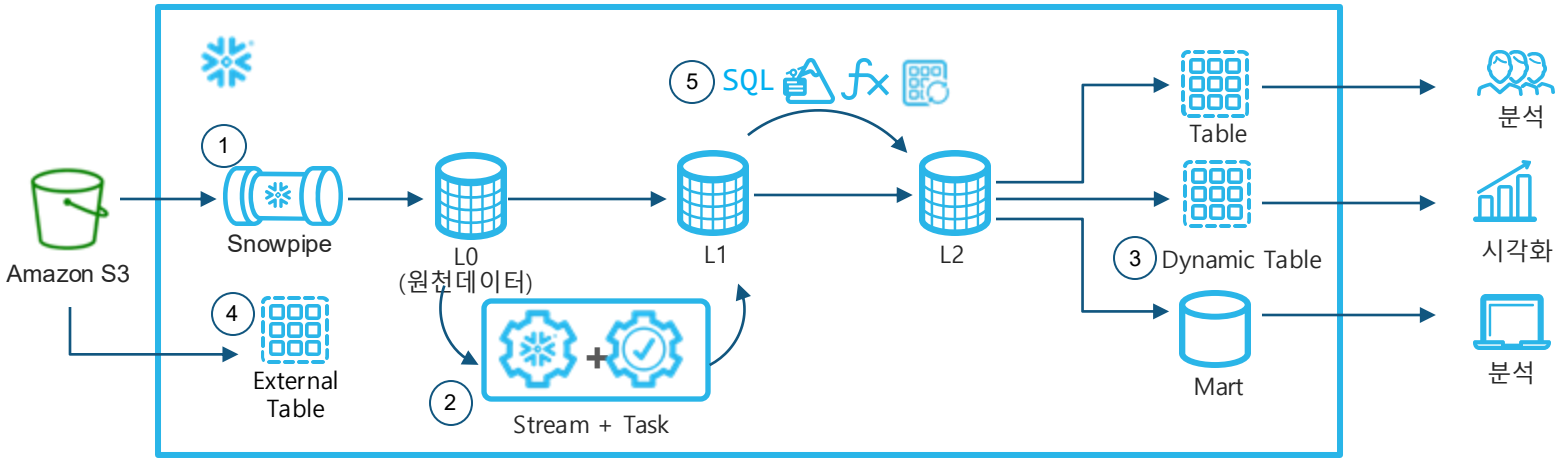


Dynamic Table - Refresh



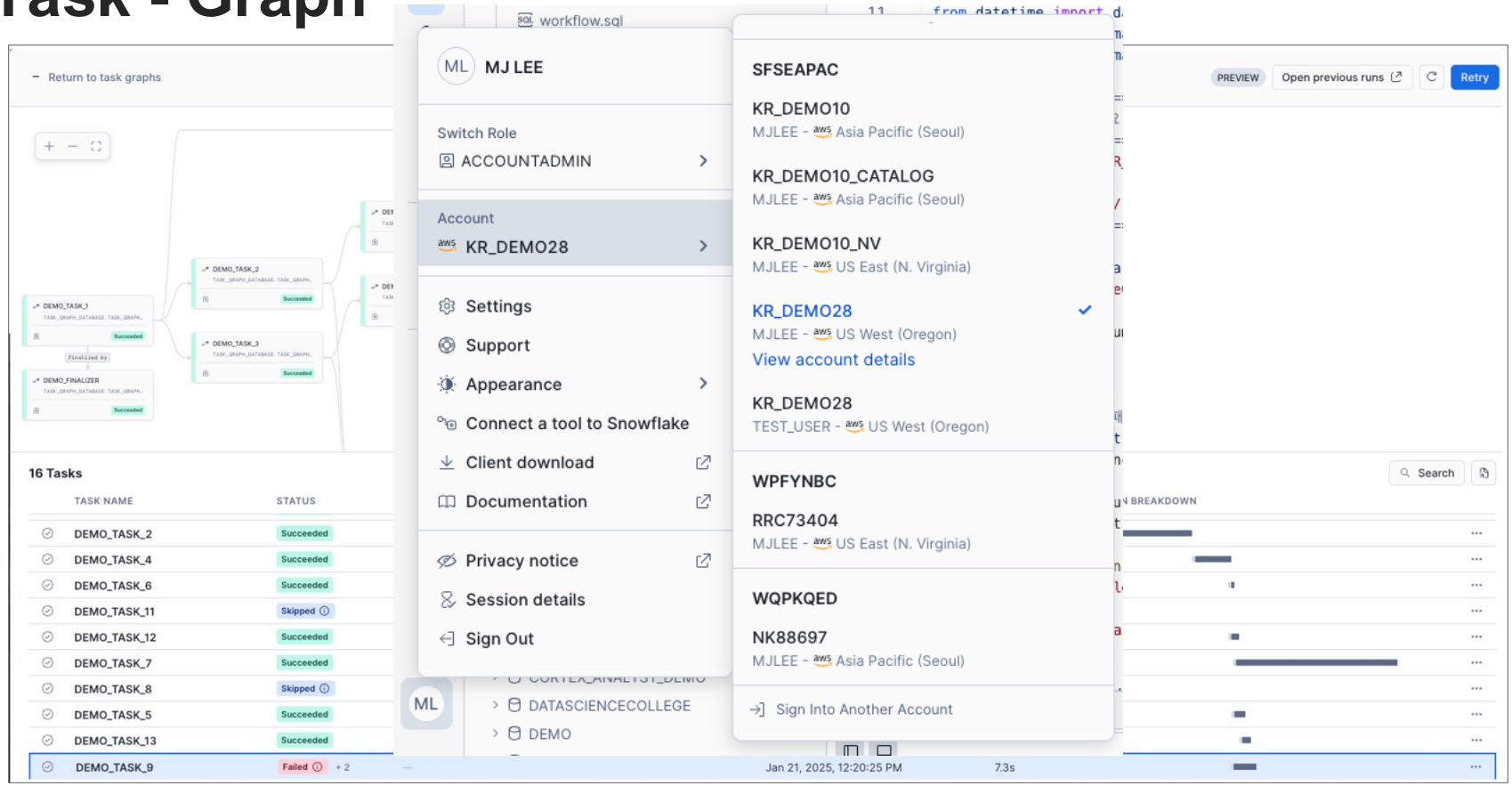
데이터 파이프라인

- 모든 데이터가 한곳에 위치하여 원하는 데이터를 빠르게 찾아 분석에 활용 / 운영 간소화
- SQL / Snowpark / Sproc / UDF 등을 활용한 쉬운 데이터 변환 파이프라인 (+자동화)

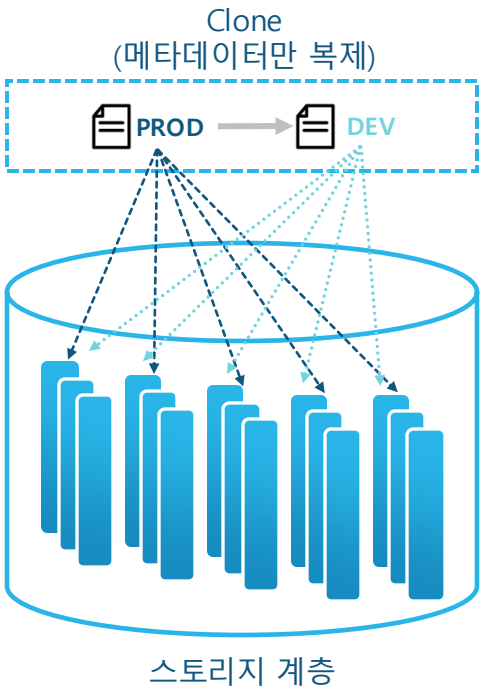


1.Snowpipe: 증분 데이터를 micro-batch 방식으로 스테이징 테이블에 적재
2.Stream + Task : Stream이 설정된 테이블에 대해 삽입, 업데이트, 삭제를 포함한 DML 오퍼레이션을 자동으로 감지 하여 CDC 기능 제공. 테이블 변환 정보를 적용하는 SQL을 Task 기능으로 스케줄링하여 CDC 자동화 가능.
3.Dynamic Table : Source 테이블의 변경 사항을 자동으로 Target 테이블에 동기화
4.External Table : Snowflake 적재 없이 S3 에 위치한 파일 read

Task - Graph



ZERO-COPY CLONING

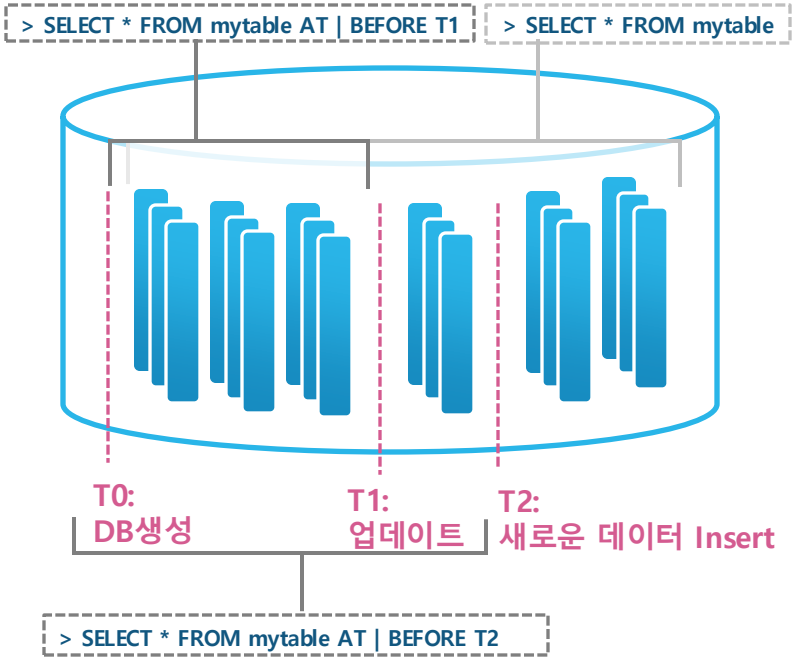


- 메타데이터 포인터 정보만 복사함으로써 물리적인 데이터 복사 없이 복제
- 순식간에 테스트 및 개발 환경 구축

```
create table sales_backup
clone sales;
```

- 메타데이터 기술을 활용한 무복사 복제 기능으로 추가적인 스토리지 불필요: 스토리지 비용 절감
- Clone 이후 데이터 변경분에 대해서만 과금

TIME TRAVEL



기능

- MVCC (Multi-version Concurrency Control) 방식
- 데이터 변경 내역이 snapshot
- 이전 버전의 데이터 자동 관리

사용 사례

- 사용자 실수나 삭제된 데이터 복원
- 과거 특정 시점 데이터 복제 및 백업
- 특정 기간 내 데이터 사용 및 변경 내용 분석
- 테이블, 스키마, 데이터베이스에 대한 UNDROP 명령어 지원

Lab02

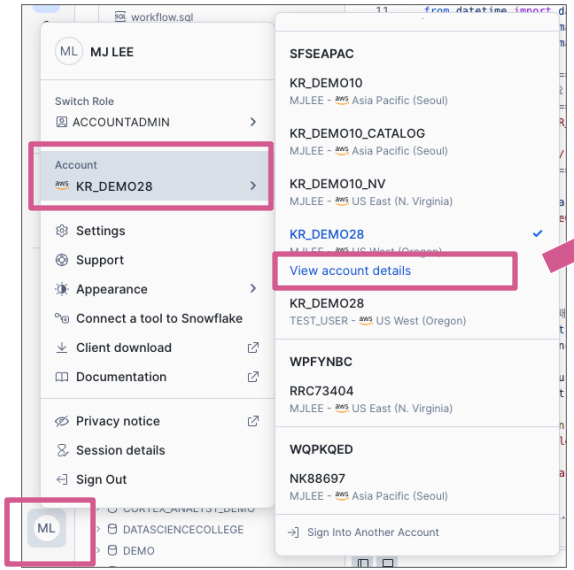
- 파이프라인 구성
- 데이터 변경 후 업데이트 확인 (@Snowsight)
- 테이블 원복

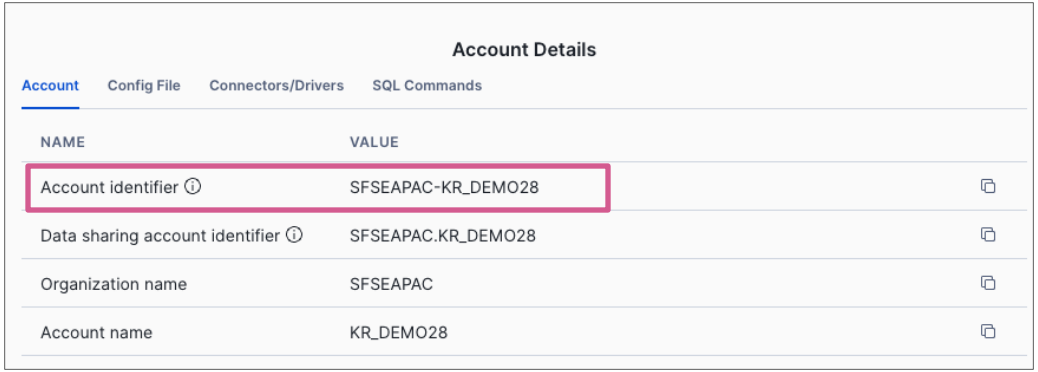
03. data endpoint API

Lab03

- API 실습 (+application 실행)

[Account 정보 확인 방법]

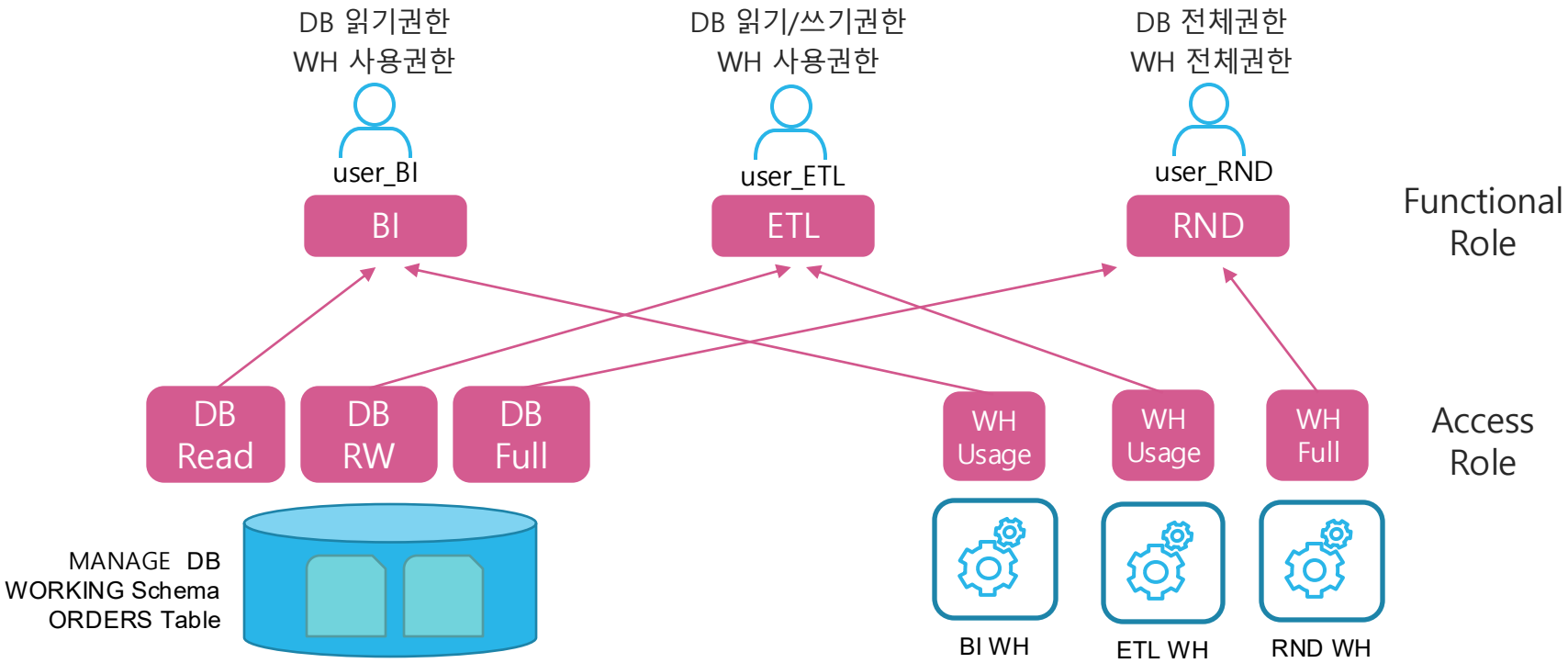




04. rbac

객체에 대한 권한 관리 및 제어

Role-based Access Control



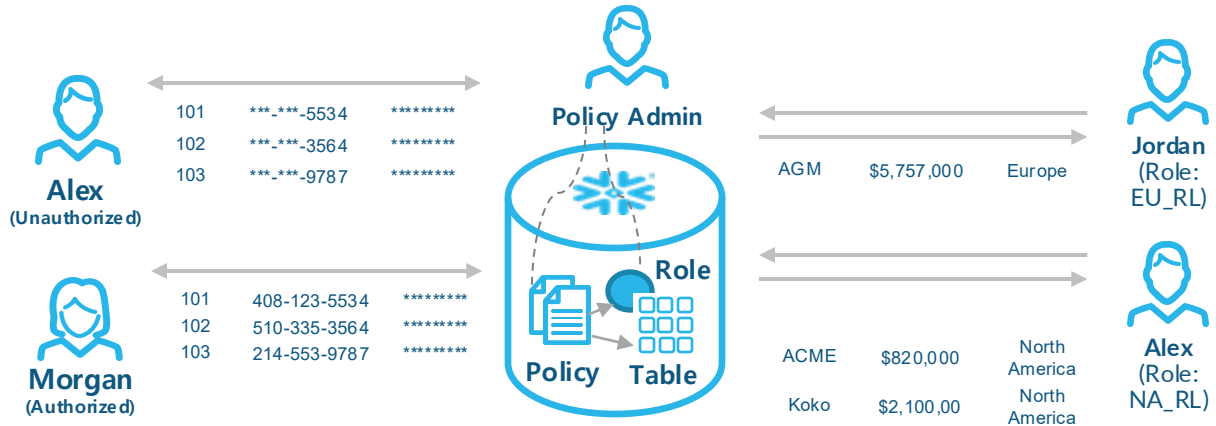
권한없는 사용자의 데이터 획득 방지

컬럼데이터 마스킹

- 저장된 데이터에 대한 변경없이 상수, 해쉬, UDF 를 통해 특정 값으로 마스킹 적용
- 권한 기반으로 인증된 사용자만 컬럼 값 확인 가능
- 단일 정책 생성 후 복수개의 컬럼에 적용 가능
- 정책은 Snowflake 중앙에서 저장 및 관리

Row 데이터 접근 제어

- 테이블, 뷰, 데이터 공유 및 외부 테이블에 대하여 정책 적용 가능
- 데이터를 하나의 테이블로 통합 후 사용자 권한에 따라 동적으로 접근 제어



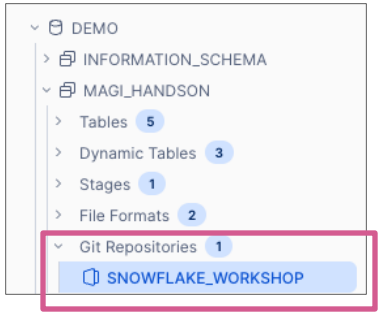
Lab04

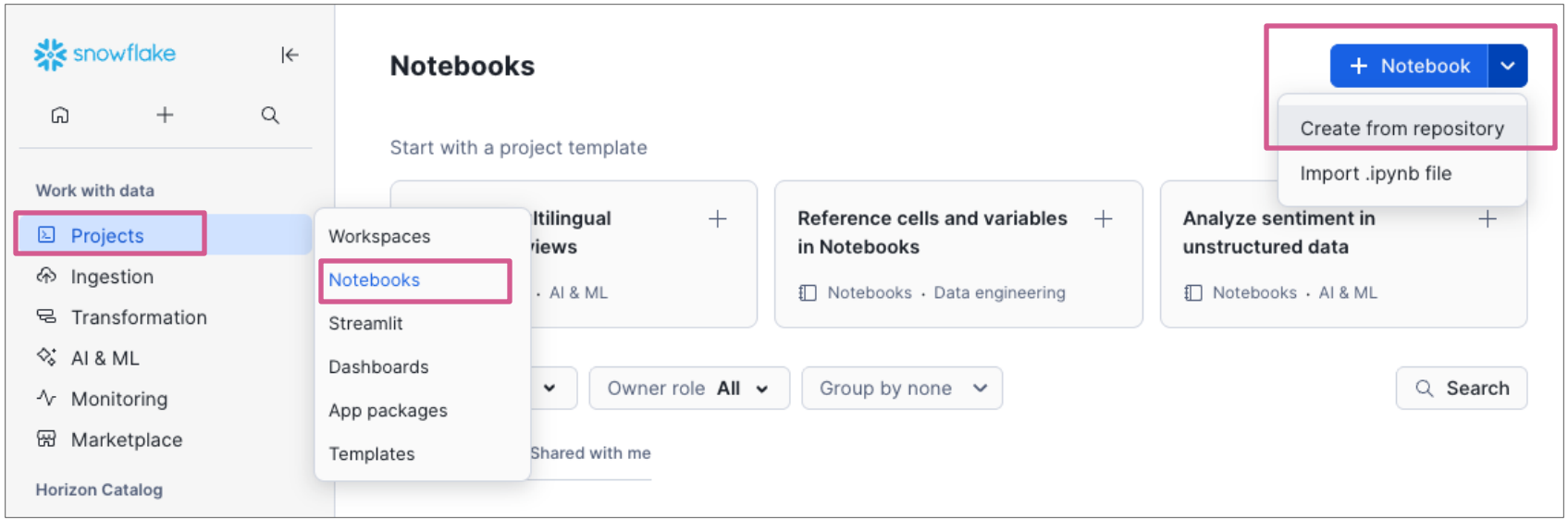
- RBAC 실습

05. python notebook

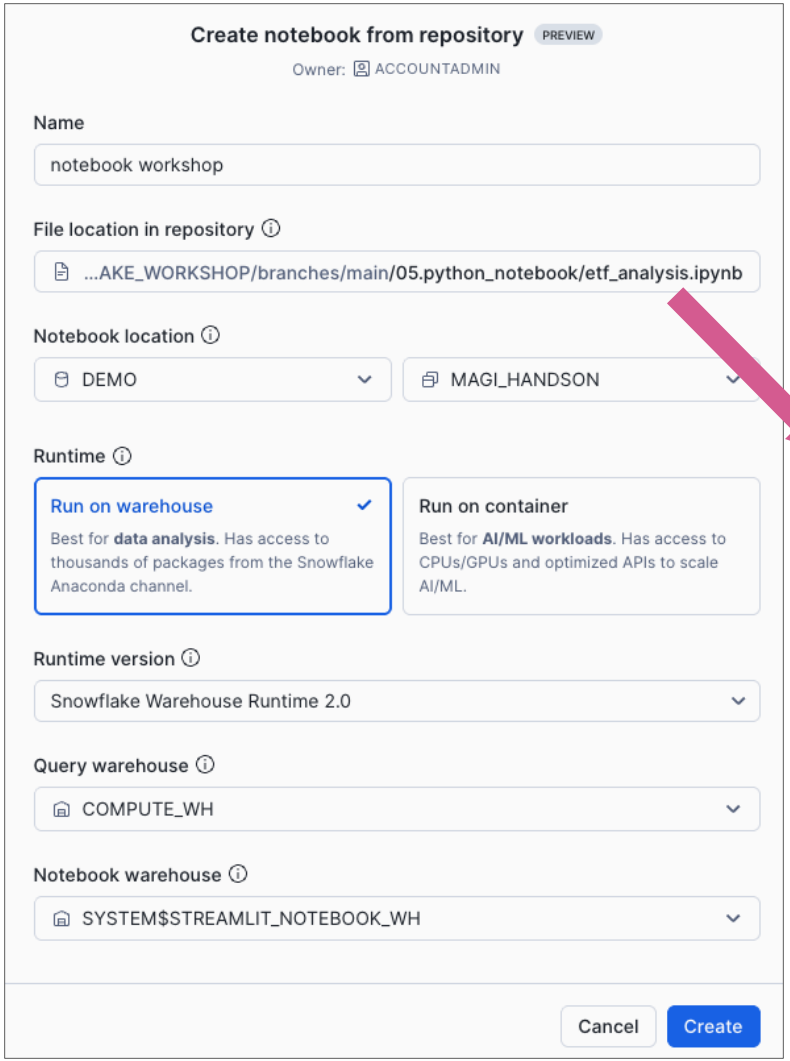
Git Repository 생성

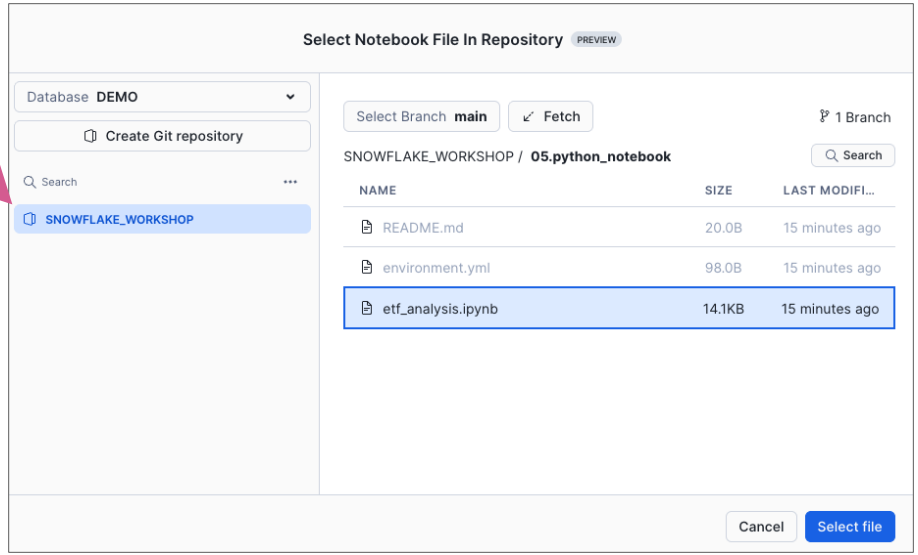
```
// 01. data loading 에서 설정한 git integration 사용
CREATE OR REPLACE GIT REPOSITORY snowflake_workshop
API_INTEGRATION = git_api_integration
GIT_CREDENTIALS = git_api_secret
ORIGIN = 'https://github.com/puremjlee/snowflake-workshop.git';
```



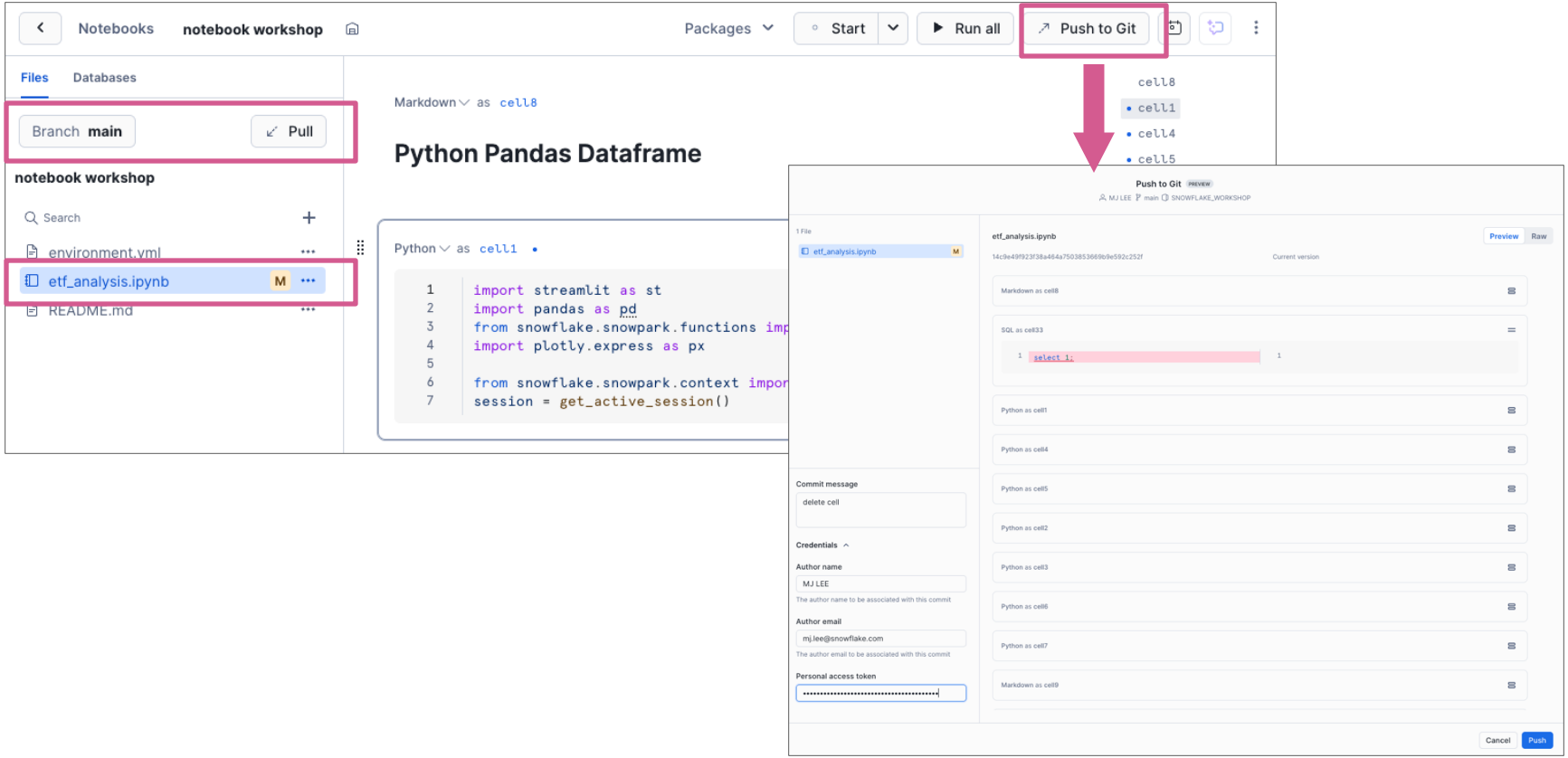


Notebook 생성


Create notebook from repository
PREVIEW
Owner: ACCOUNTADMIN
Name
notebook workshop
File location in repository
...AKE_WORKSHOP/branches/main/05.python_notebook/etf_analysis.ipynb
Notebook location
DEMO
MAGI_HANDSON
Runtime
Run on warehouse
Best for data analysis. Has access to thousands of packages from the Snowflake Anaconda channel.
Run on container
Best for AI/ML workloads. Has access to CPUs/GPUs and optimized APIs to scale AI/ML.
Runtime version
Snowflake Warehouse Runtime 2.0
Query warehouse
COMPUTE_WH
Notebook warehouse
SYSTEM$STREAMLIT_NOTEBOOK_WH
Cancel
Create


Select Notebook File In Repository
PREVIEW
Database DEMO
Create Git repository
Search
SNOWFLAKE_WORKSHOP
Select Branch main
Fetch
1 Branch
SNOWFLAKE_WORKSHOP / 05.python_notebook
Search
NAME
SIZE
LAST MODIFI...
README.md
20.0B
15 minutes ago
environment.yml
98.0B
15 minutes ago
etf_analysis.ipynb
14.1KB
15 minutes ago
Cancel
Select file

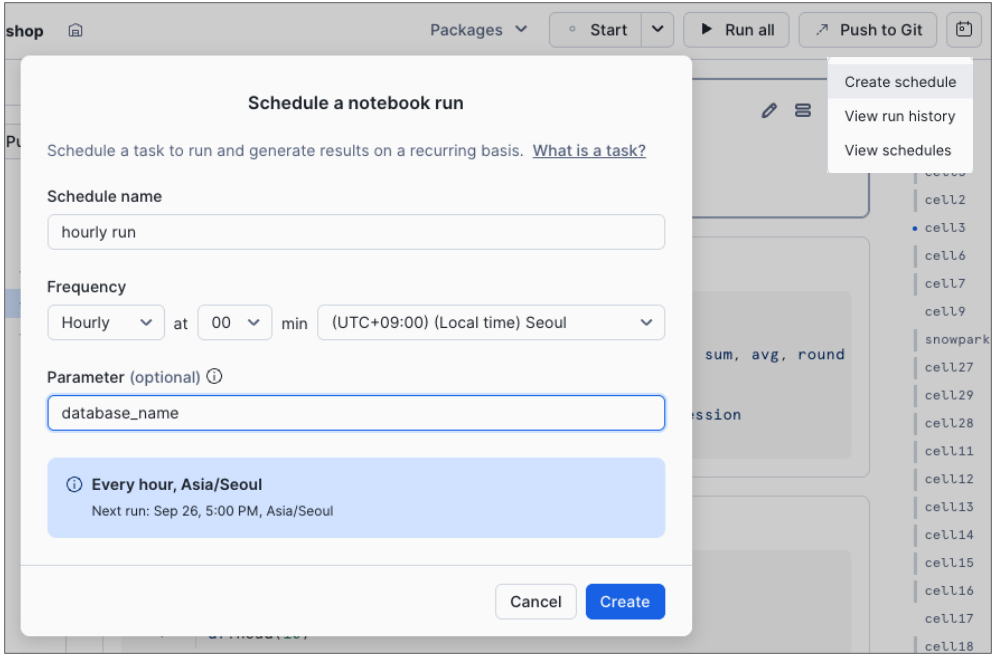
Git Push / Pull



Schedule

```
EXECUTE NOTEBOOK MY_DB.PUBLIC.MY_NOTEBOOK();
```

```
create task run_notebook_task
WAREHOUSE = mywh
SCHEDULE = 'USING CRON 0 9-17 * * SUN Asia/Seoul'
as
EXECUTE NOTEBOOK MY_DB.PUBLIC.MY_NOTEBOOK();
```



Python 기반 데이터 엔지니어링

데이터프레임

Pandas Dataframe

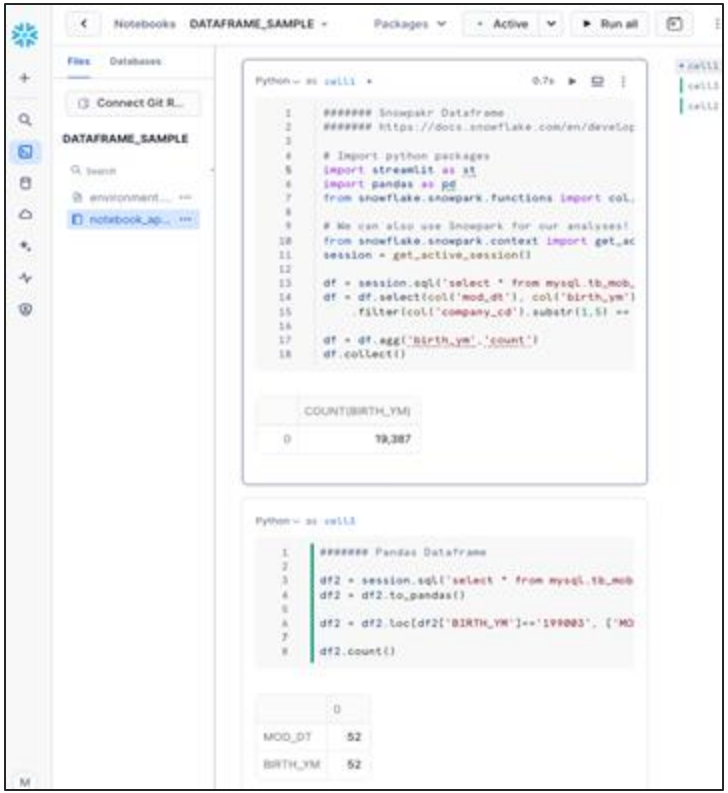
- Python 에서 지원하는 2차원 데이터 구조
- 클라이언트 머신 Local 리소스를 사용
- 연산할 때마다 결과가 메모리에 materialize 됨

Snowpark Dataframe

- Snowflake 의 데이터 처리 프레임에서 지원하는 2차원 데이터 구조
- Spark 과 유사한 Dataframe API
- Snowflake 에서 SQL 로 변환되어 실행되며 확장 리소스 지원
- Lazy evaluation
- 원본 데이터 기반으로 작업 수행하며, 분산처리 효율성을 위해 데이터 순서 보장하지 않음

Snowpark Pandas Dataframe

- Snowflake 의 데이터 처리 프레임에서 지원하는 2차원 데이터 구조
- Pandas 와 유사한 Dataframe
- Snowflake 에서 SQL 로 변환되어 실행되며 확장 리소스 지원
- Lazy evaluation
- 원본 데이터를 clone 하여 스냅샷 기반으로 작업 수행하며, 데이터 순서 보장



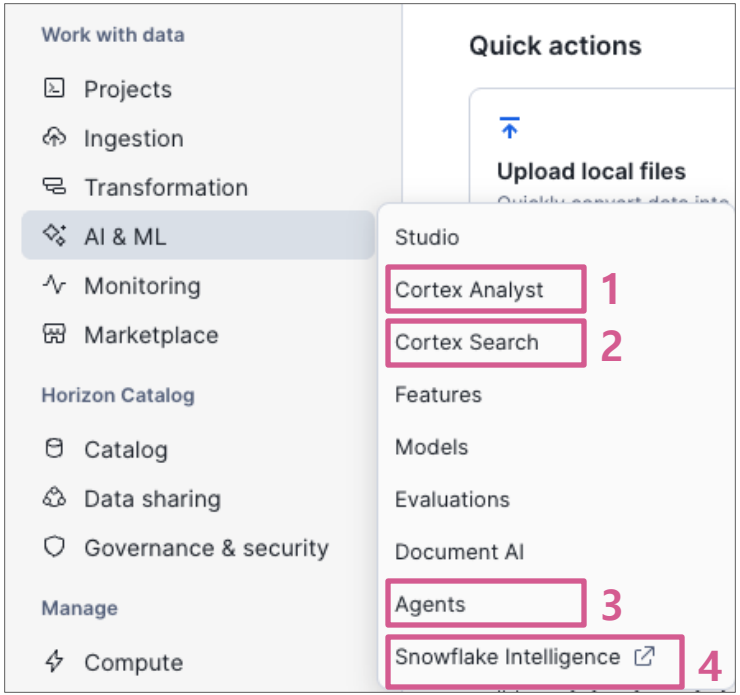
https://docs.snowflake.com/developer-guide/snowpark/python/snowpark-pandas

Lab05

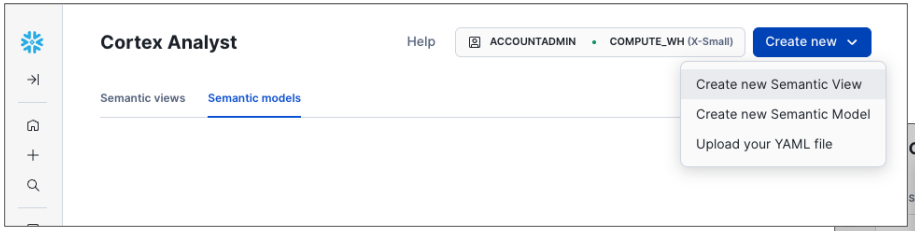
- Notebook 실행
- 성능 비교

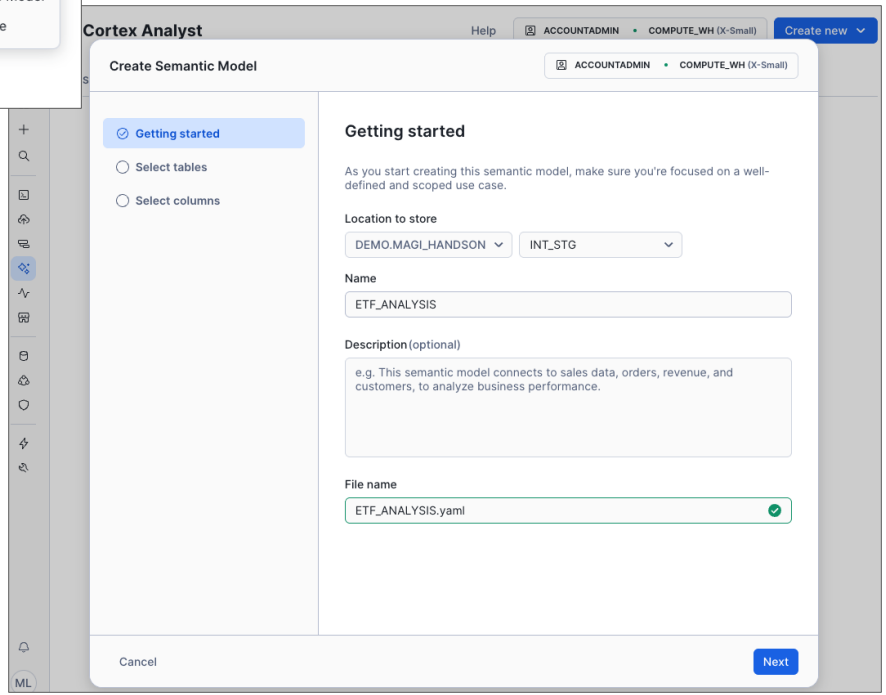
06. cortex AI

Snowflake Intelligence

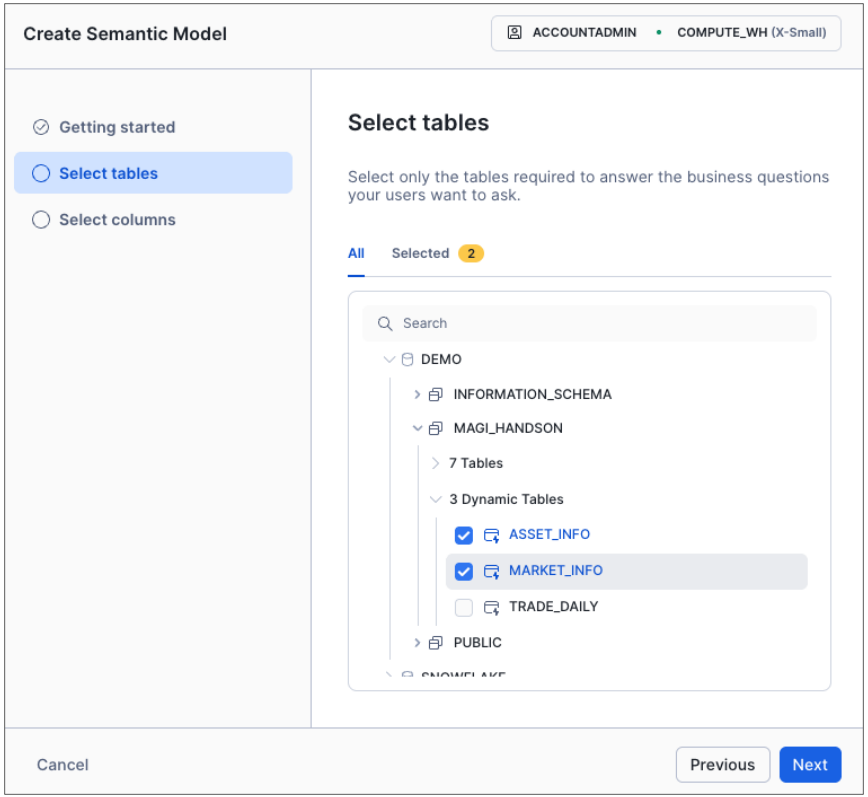


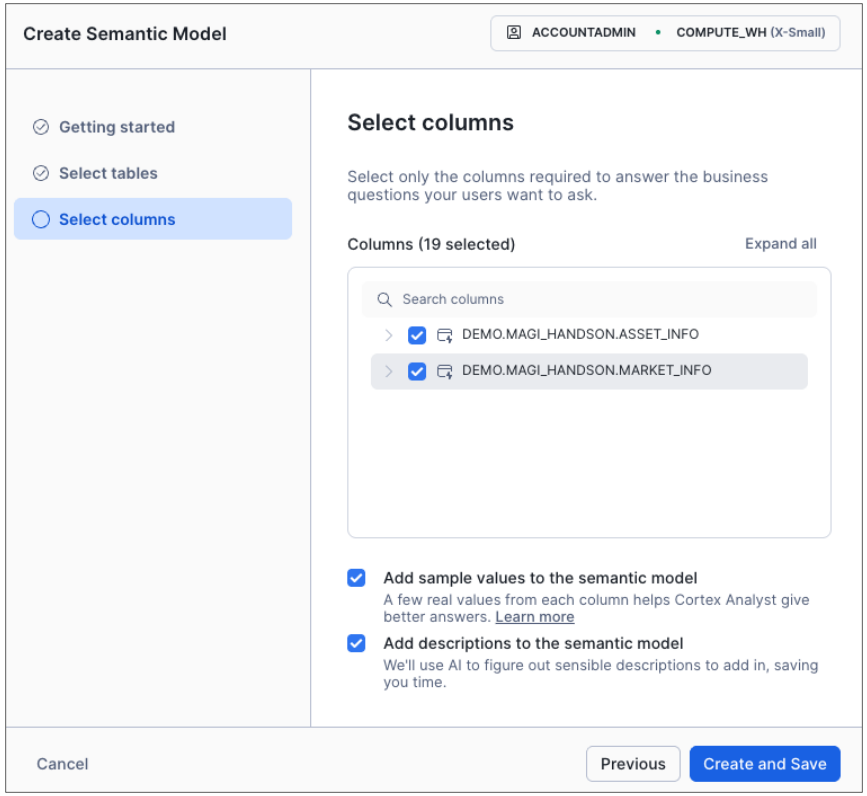
Cortex Analyst



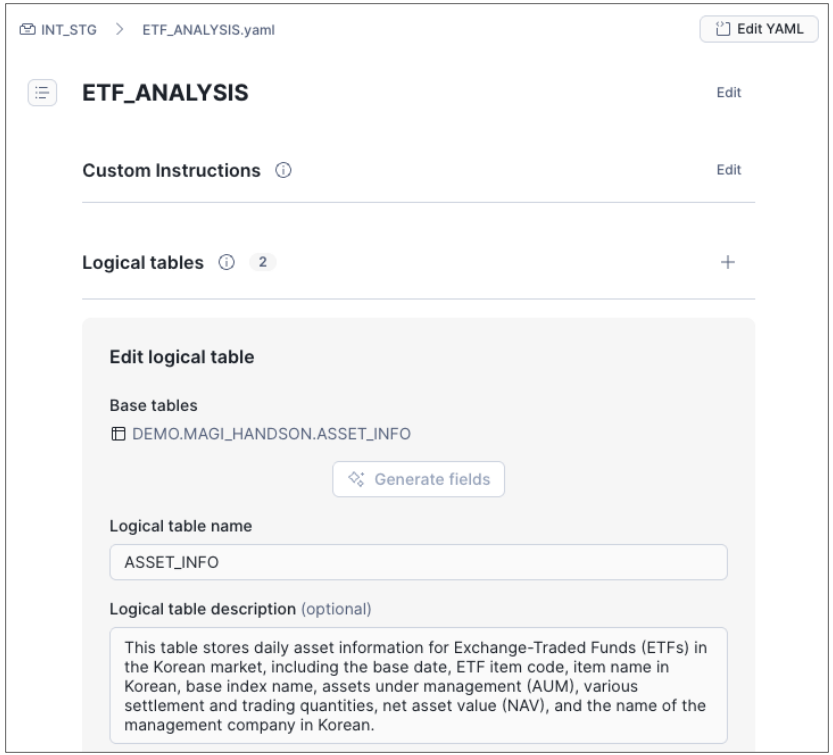


Cortex Analyst





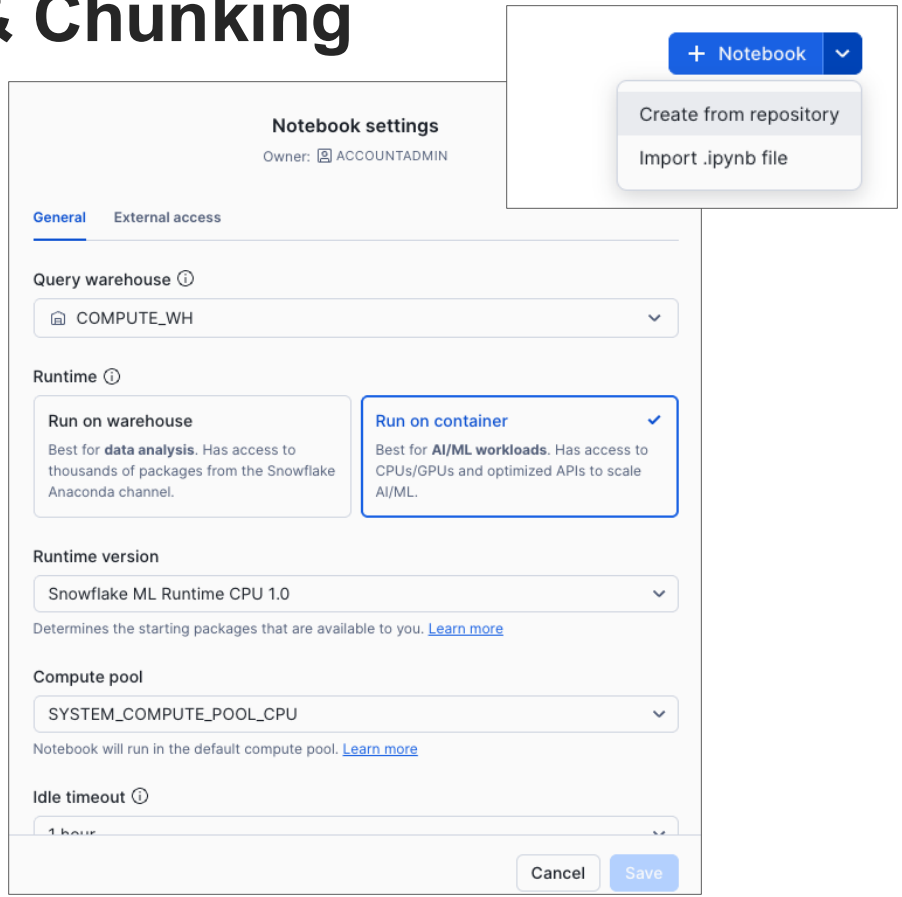
Cortex Analyst



Cortex Search – Parsing & Chunking

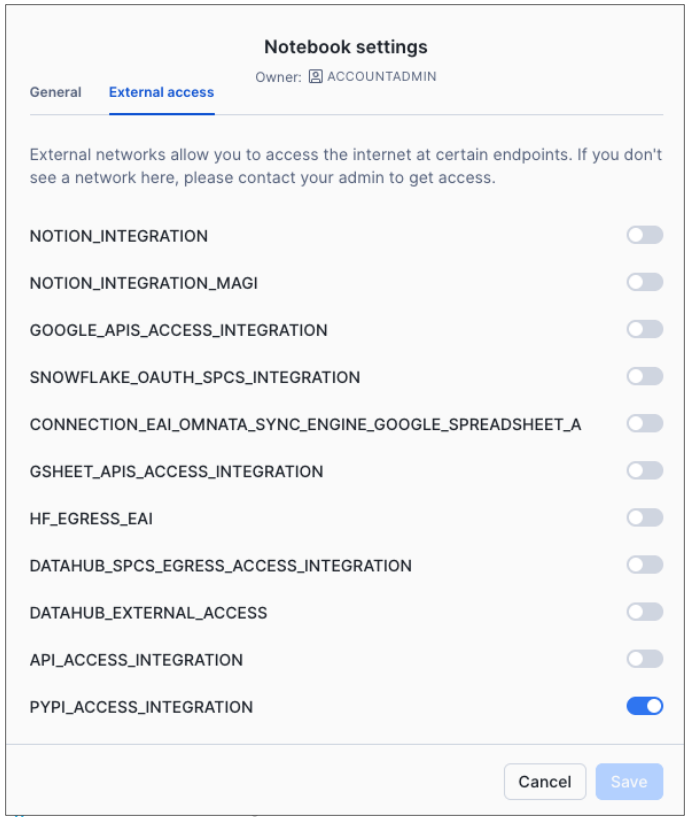
Trial 버전에서는 실행 불가

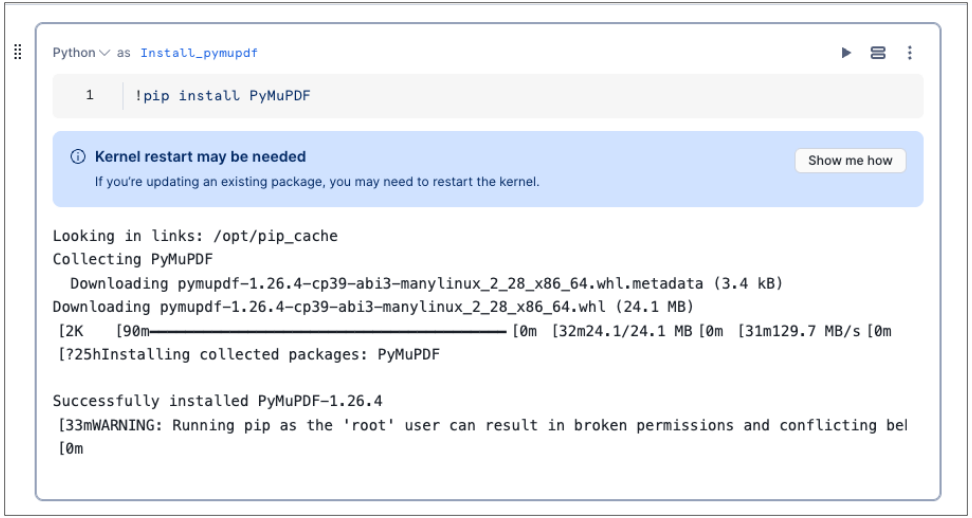
```
CREATE OR REPLACE NETWORK RULE pypi_network_rule
MODE = EGRESS
TYPE = HOST_PORT
VALUE_LIST = ('pypi.org', 'pypi.python.org',
'pythonhosted.org', 'files.pythonhosted.org');

CREATE OR REPLACE EXTERNAL ACCESS INTEGRATION
pypi_access_integration
ALLOWED_NETWORK_RULES = (pypi_network_rule)
ENABLED = true;
```



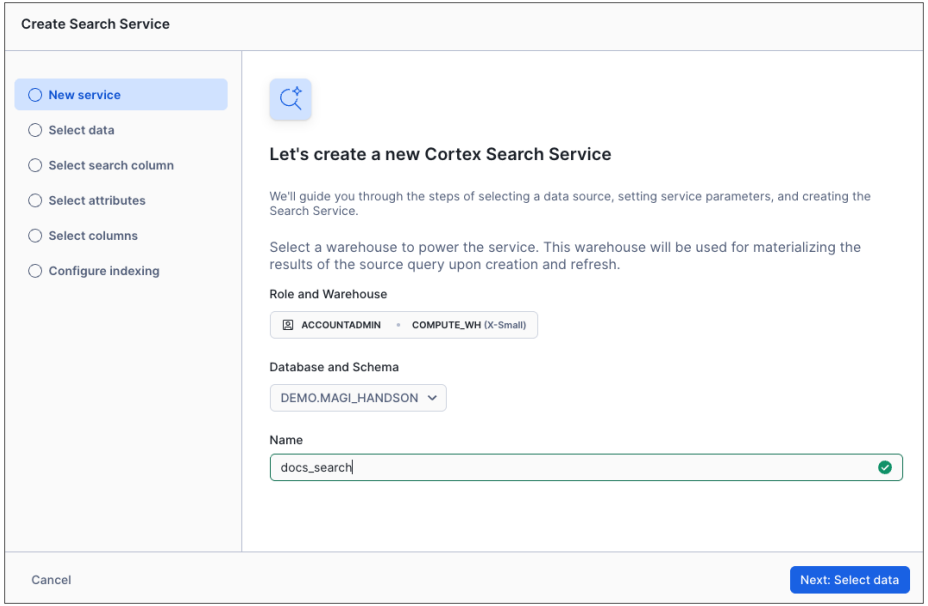
Cortex Search – Parsing & Chunking

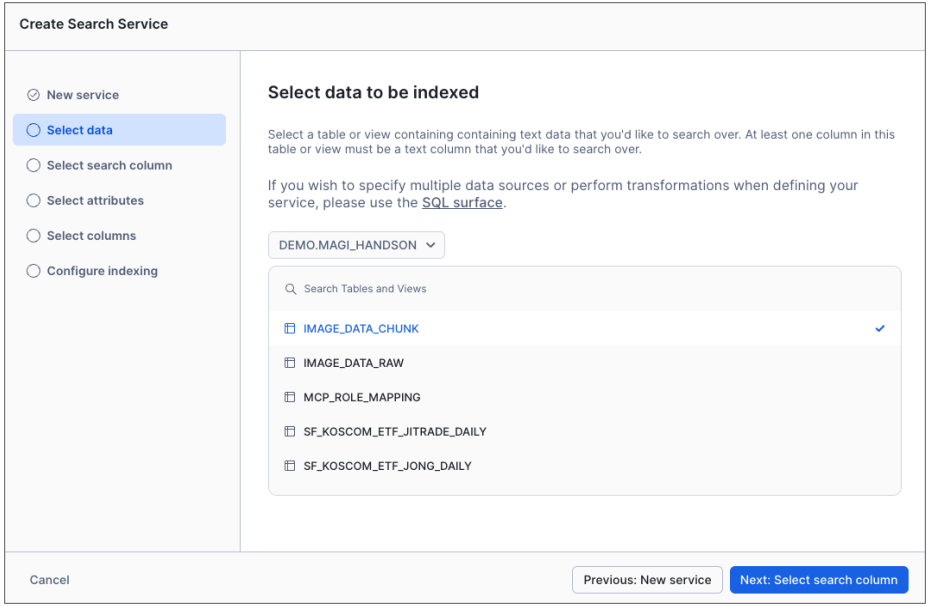
Trial 버전에서는 실행 불가



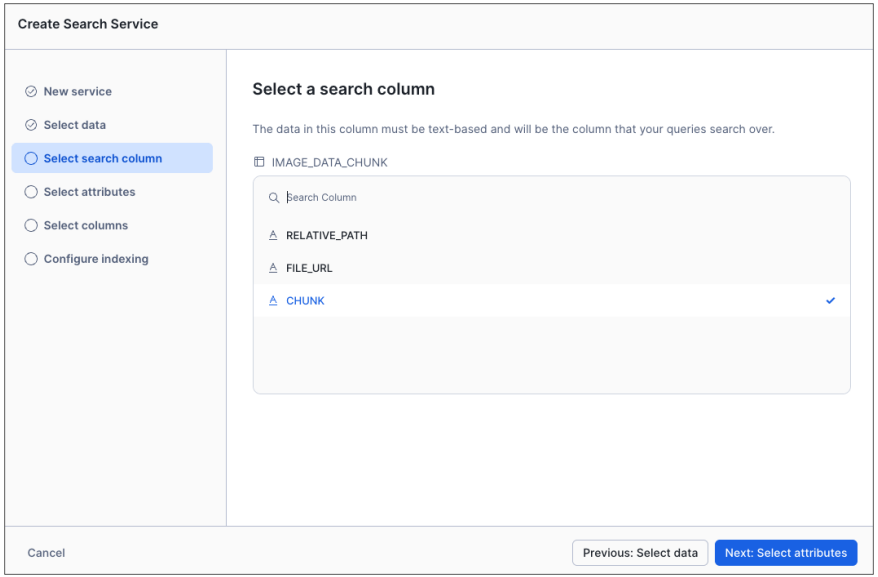


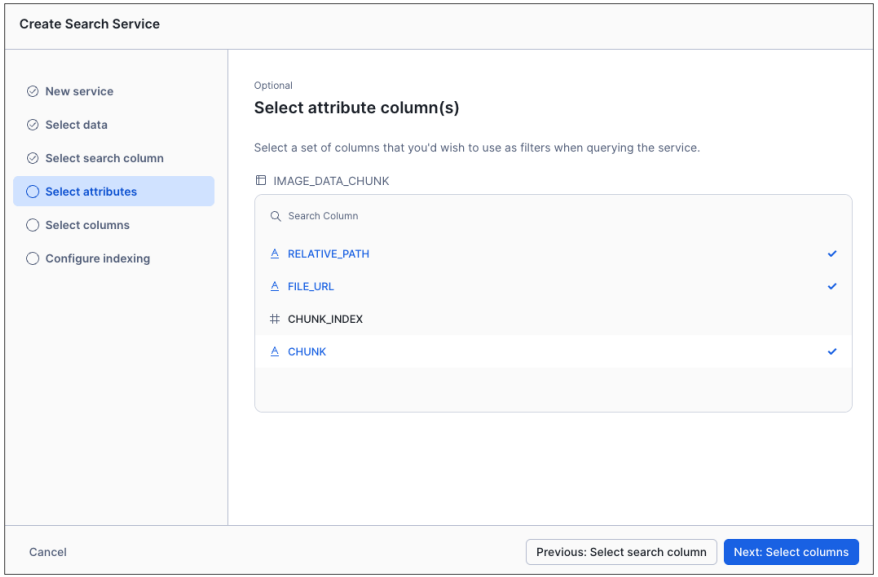
Cortex Search



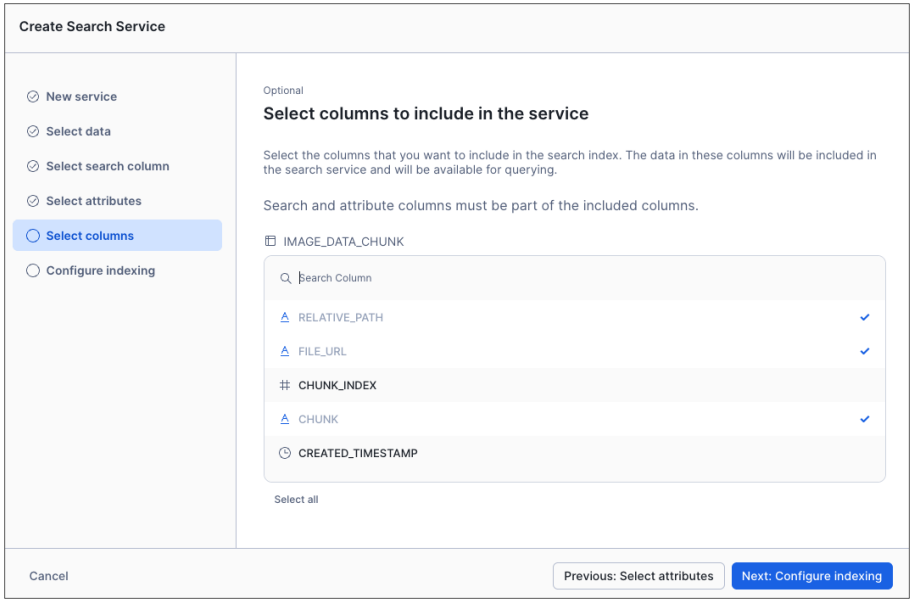


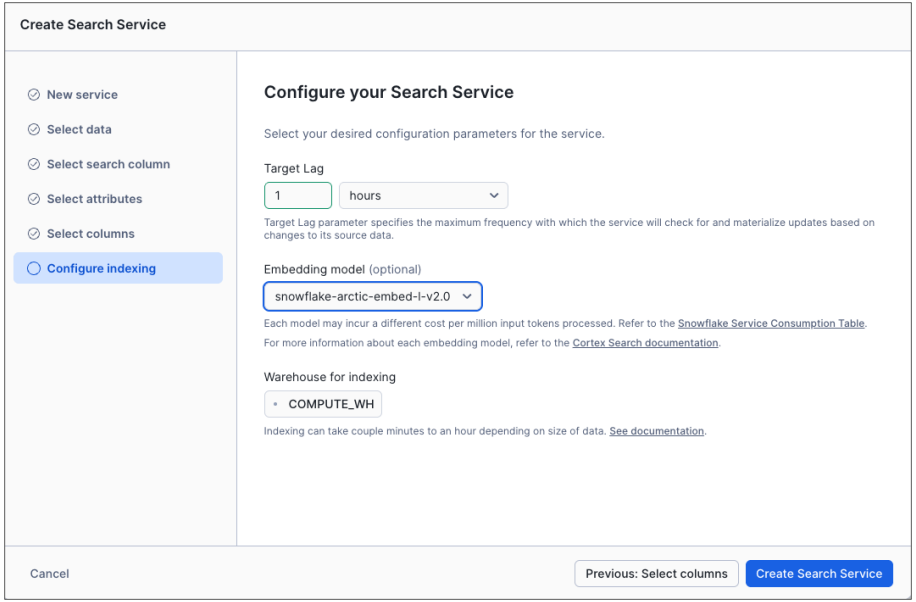
Cortex Search



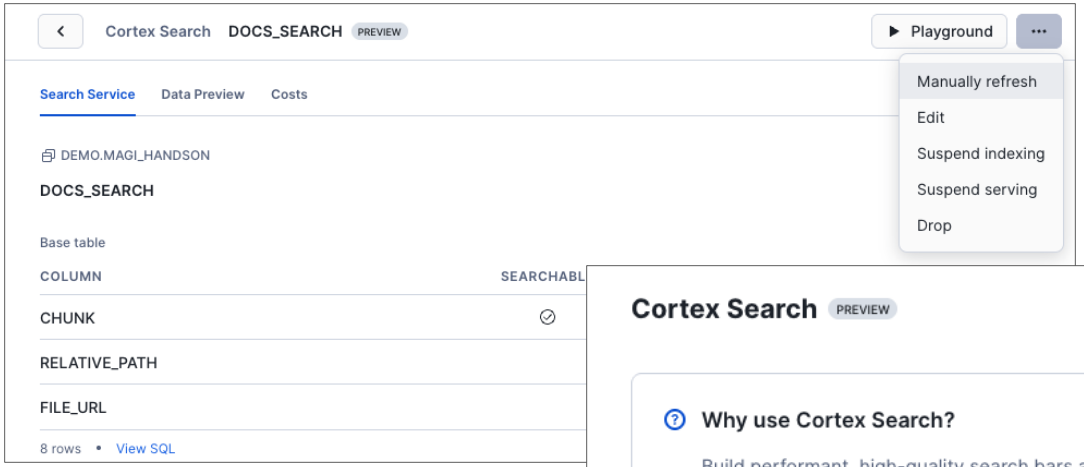


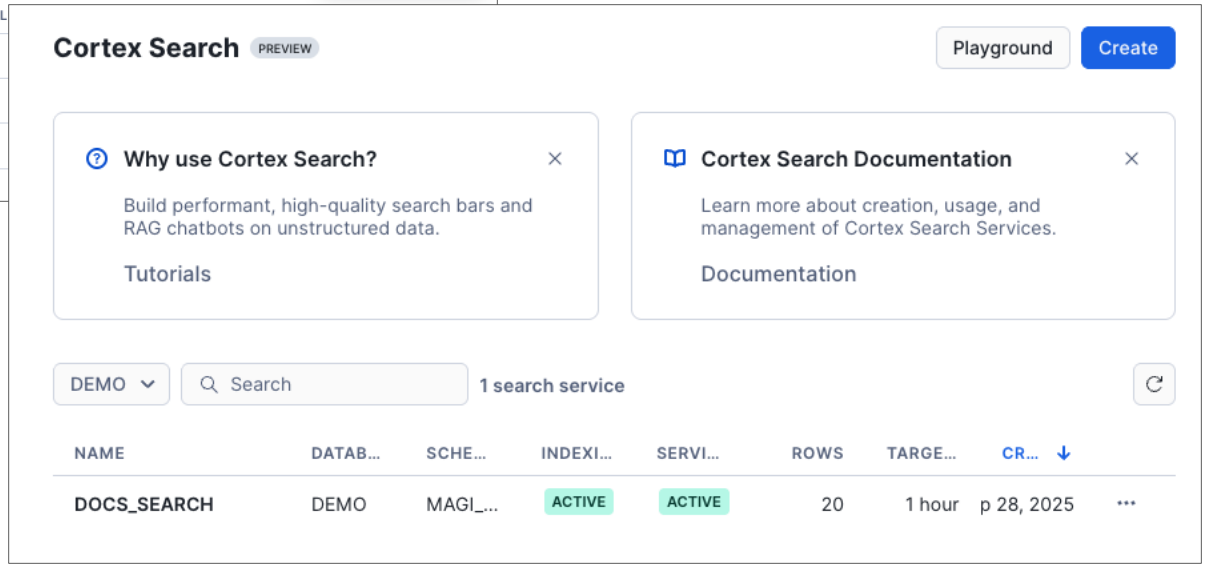
Cortex Search



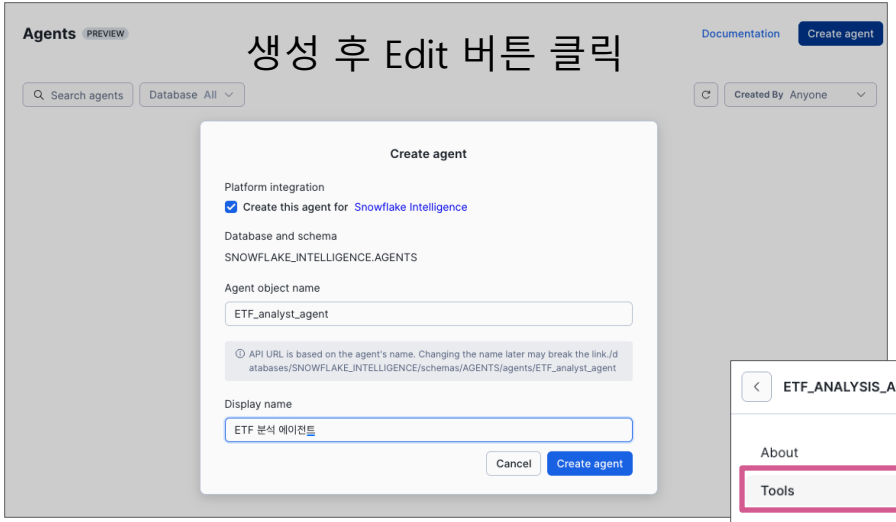


Cortex Search

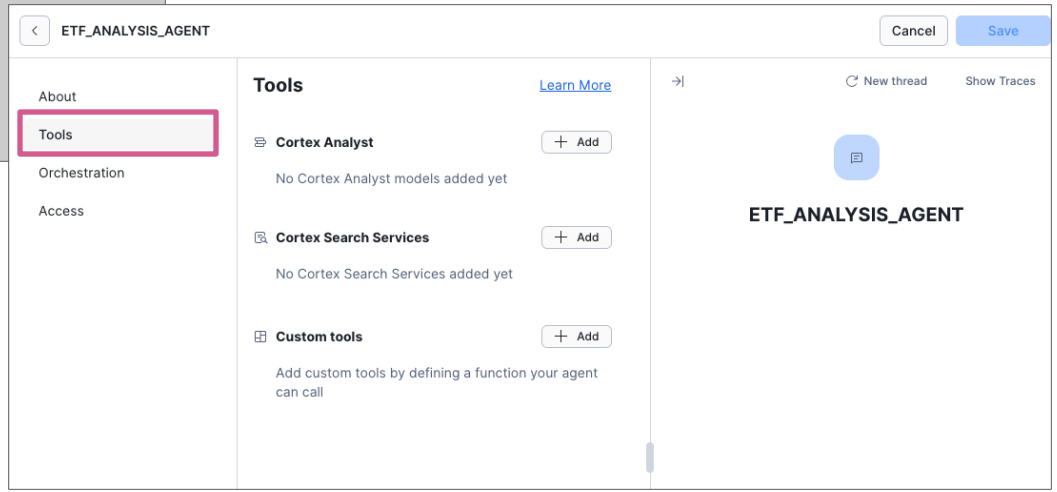




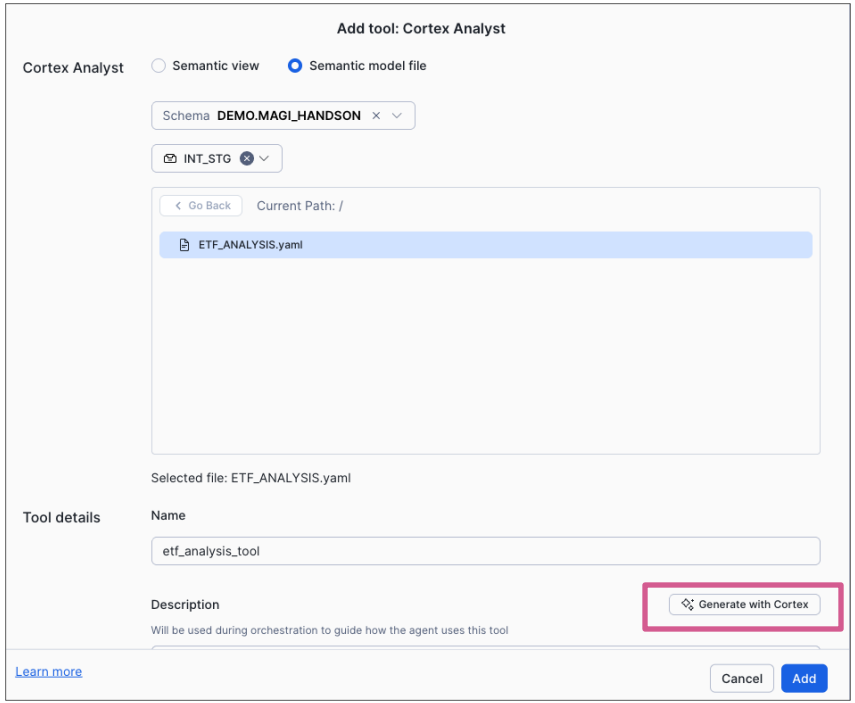
Cortex Agent

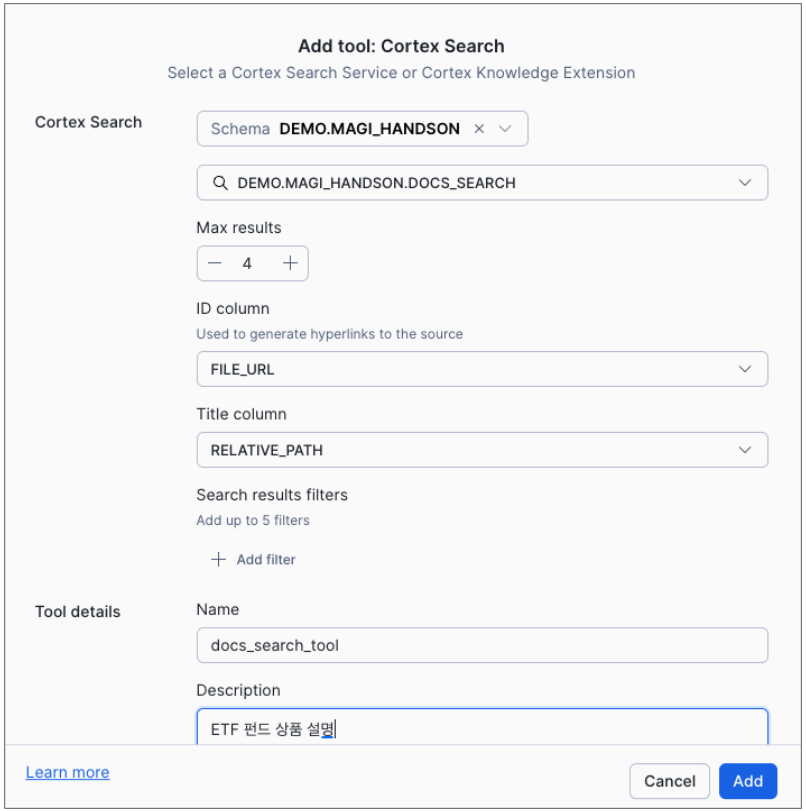


좌측 Tools 선택 후 이전에 생성한 Cortex Analyst 와 Cortex Search 등록



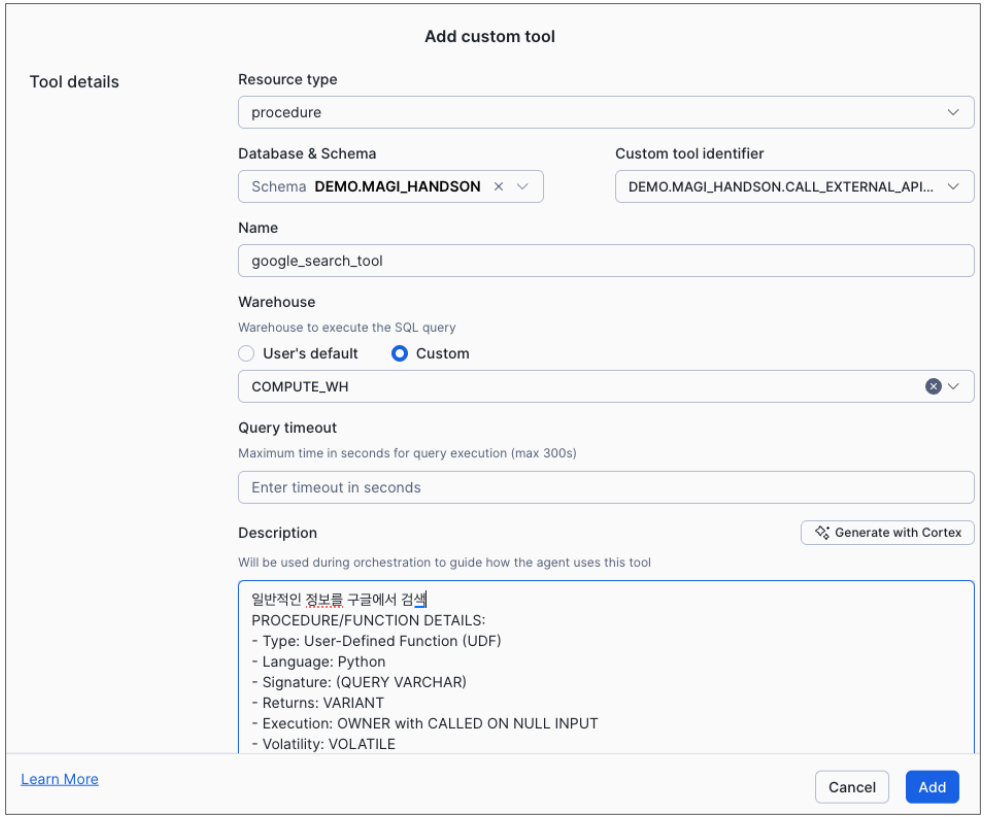
Cortex Agent – Cortex Analyst / Search 등록



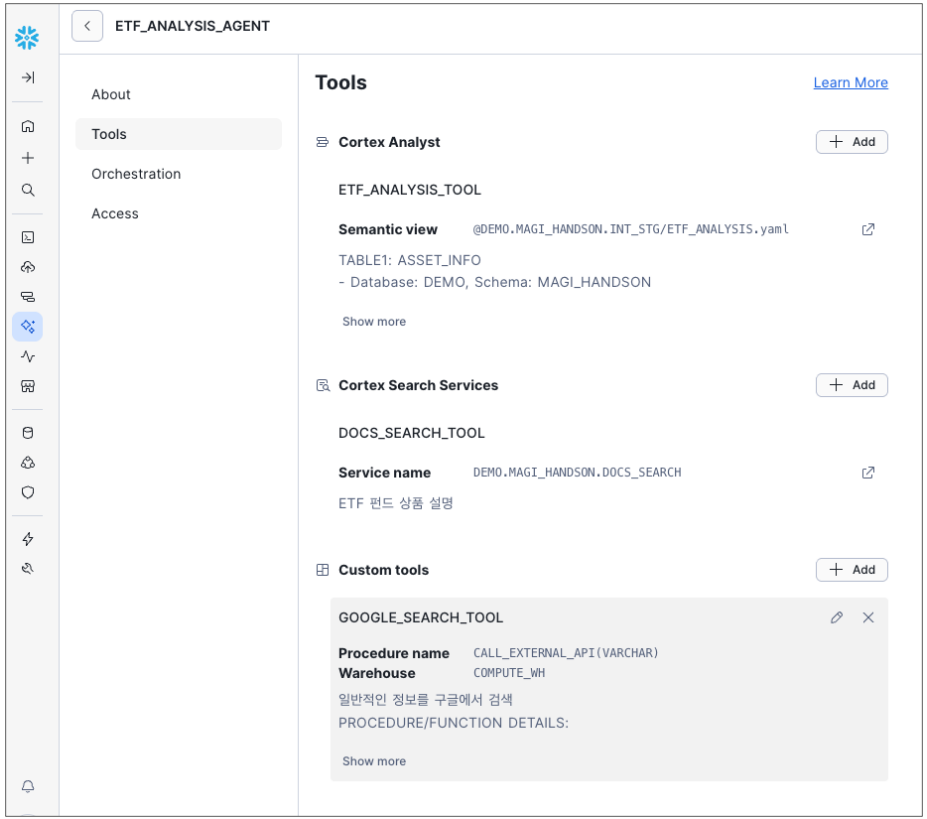


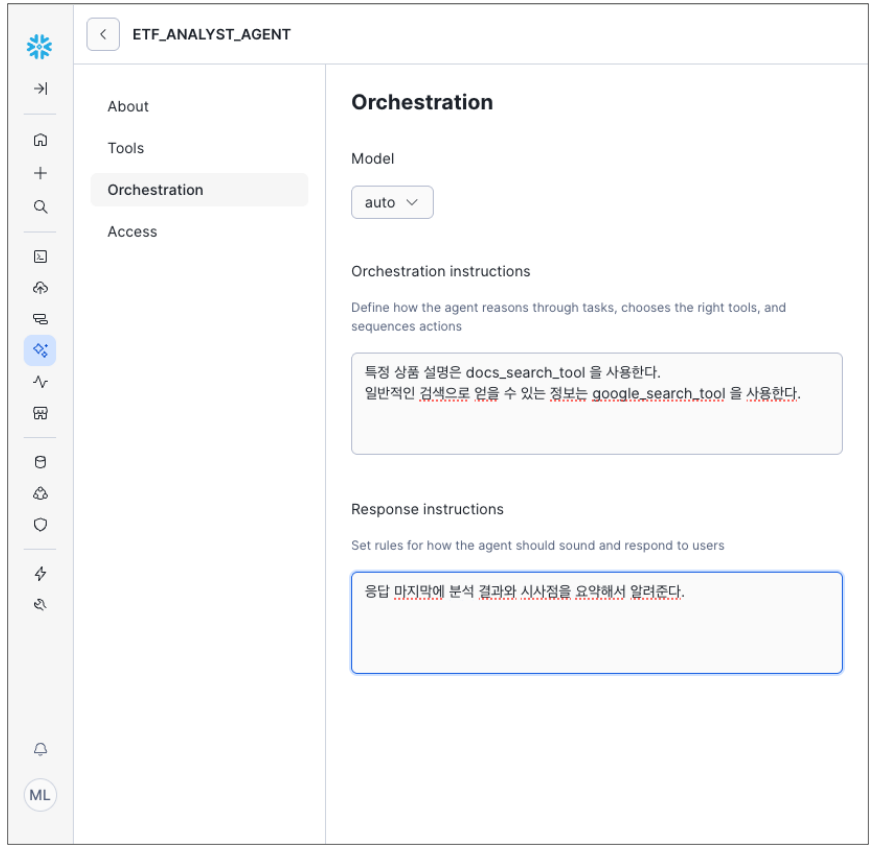
Cortex Agent – Custom tool 등록

Trial 버전에서는 실행 불가

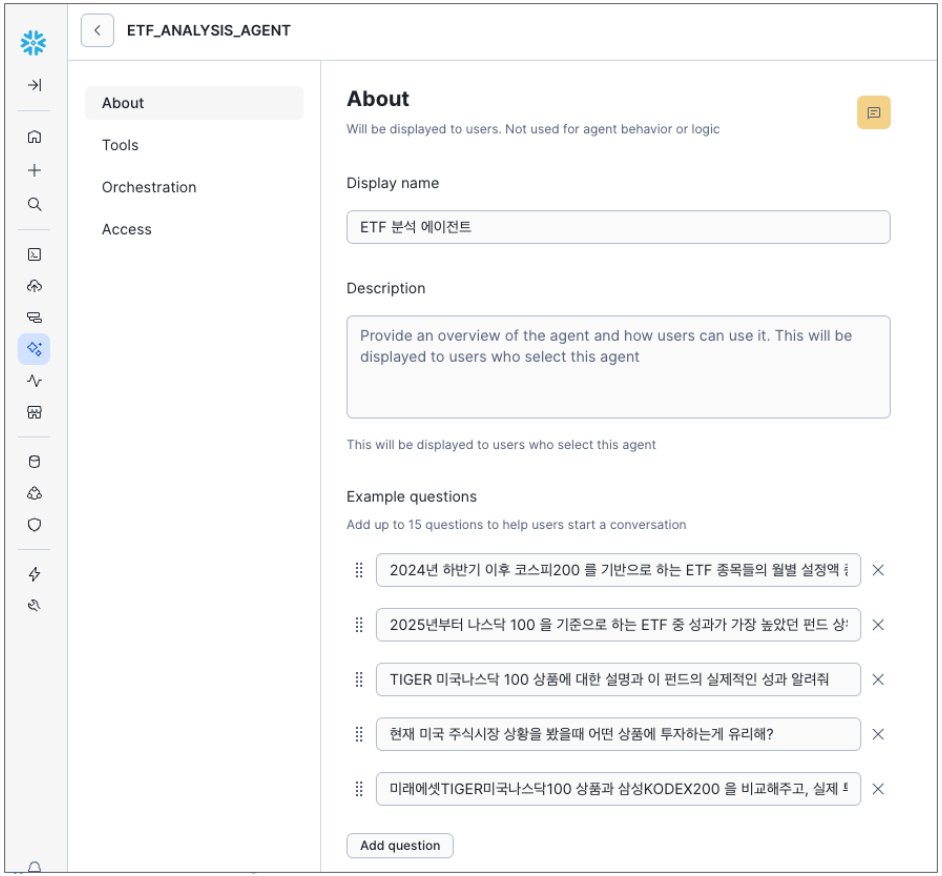


Cortex Agent 설정 화면





Cortex Agent



Lab06

- Agent 만들기
- 질문으로 얻은 Insight 발표 (+개선/적용 방안)

THANK YOU