**Case Study for the Medium level Data Engineer Role**

**Modernizing Metadata Pipelines for Scilit**

**Scenario**

Scilit.net is a metadata aggregator operated by MDPI, serving as the data backbone

for several internally developed AI applications. It sources metadata from a variety of

external providers, including CrossRef, PubMed, and ORCID.

Currently, the system lacks proper data engineering foundations. It relies on a PHPbased

backend with cron jobs to handle ingestion tasks. Metadata is stored in a

MySQL database, raw data files are placed on CEPH storage, and analytical

workloads are inconsistently computed from both MySQL and Apache Solr indexes.

To improve reliability and scalability, analytics workloads are being moved to a

modern data warehouse maintained by MDPI’s AI / Technology Innovation team.

This team uses Python, Polars, Parquet, PostgreSQL, dbt, DuckDB, and Airflow. The

Data Architect has decided to align Scilit on this technology stack. However, due to

existing dependencies on the Scilit MySQL data model, the data model in Scilit will

only be minimally changed. Instead, the focus of refactors are data laking, pipelines,

proper monitoring and orchestration, and improving data lineage.

I are brought in as a “Medior” (middle level of seniority) Data Engineer to help rebuild core ingestion and processing pipelines, support a more maintainable and testable development

workflow, and improve data quality and observability for Scilit.

**My Task**

This is a hands-on case study that you should prepare in advance. I can grab

sample metadata files (JSON) from the CrossRef API as indicated below, and

sample output requirements. Senior colleagues will review my deliverables together in the

interview session via a walk-through (ca. 15 minutes), where I can demonstrate

what I have built. Interviewers may ask questions to confirm my expertise and

sound data handling practices.

Dataset: I can grab 200 items or more from the public CrossRef API via

<https://api.crossref.org/works?sort=published&order=desc&rows=200>

What to deliver – please **send us the link to a public GitHub.com repository** before / by the time of the interview:

1. A high-level data model based on the CrossRef data model but suitable for our

downstream analytical tasks.

• A data ingestion pipeline prototype, built with Python

• Sample code that parses and ingests raw metadata into a staging format

(e. g., normalized tables in MySQL/PostgreSQL or Parquet)

• Optional: transformation logic that resolves simple data issues (e.g.,

inconsistent IDs, missing fields)

• Optional: Setup a standalone MinIO S3 as a Docker container and

save the raw data in this container for further usage with dbt.

2. Data transformation models

• Cleanup transformations using a Python dataframe library *or* in SQL

using dbt-DuckDB

• Example: shape the staged metadata into useful outputs (e.g., a clean

table of publications with authors, source, and dates or timestamps)

• Include some logic for schema tests, data quality checks, or freshness

indicators

3. Downstream data usage

• Based on ingested and processed data, create 2-3 processing queries for

analytics

• Alternatively, build an ingestion of the processed data in an

OLTP database of my choice.

4. A brief 15-minutes walkthrough during the interview (to do a walkthrough of my code with the interview participants)

• Outline my choices, what assumptions I made how my code is

structured, and how it could be extended into production

Here is a repository that you can use as a template and already includes a rough

framework: <https://github.com/MDPI-AG/case-study-data-engineering>

**Evaluation Criteria**

Candidates will be assessed on:

• Effective use of tools (e.g., dbt, SQL, Python libraries)

• Logical data modeling and transformation

• Data quality practices (e.g., testing, validation, observability)

• Clarity in communicating technical trade-offs

• Code quality, modularity, and reusability [we are aware of the time pressure –

this will be regarded as a “nice to have”]