Zadanie 2 Wybór technologii - składowanie danych w Databricks

- Porównanie technologii **DeltaLake** i **Iceberg**
- Podpowiedz klientowi w jakich scenariuszach każda z technologii będzie bardziej optymalna.

Delta Lake

Technologia stworzona przez Databricks, głęboko zintegrowana z Apache Spark i Databricks.

Zalety:

- Automatyczne zarządzanie transakcjami ACID
- Wbudowana obsługa MERGE, UPDATE, DELETE
- Optymalizacja poprzez **Z-Ordering**, OPTIMIZE, VACUUM
- Bardzo dobra integracja z Power BI i MLflow

Idealna dla scenariuszy:

- **Lakehouse** na platformie **Databricks** z częstymi aktualizacjami danych (np. merge, upsert)
- **ETL/ELT pipelines** z dużą liczbą operacji na danych
- Projekty ML, gdzie potrzebna jest wersjonowalność danych (time travel)

Apache Iceberg

Technologia rozwijana przez Netflix i wspierana przez społeczność open-source, zaprojektowana z myślą o skalowalności i otwartości.

Zalety:

- Skalowalna metadana (lepsza niż Delta Lake przy bardzo dużych tabelach)
- Pełna separacja metadanych od plików danych
- Optymalna dla **czytania danych w dużych zbiorach** (np. analizy ad-hoc)
- Wspierana przez wiele silników: Spark, Trino, Flink, Presto

Idealna dla scenariuszy:

- **Heterogeniczne środowiska** (wielu użytkowników/silników: Flink, Trino, Presto)
- Duże zbiory danych typu **append-only** (np. dane telemetryczne, logi)
- Analiza danych w otwartym ekosystemie

Podpowiedź dla klienta:

Scenariusz użycia	Technologia	Uzasadnienie
Praca **w Databricks** z pełnym wsparciem	Delta Lake	Natywna integracja, optymalizacja, operacje ACID
Duże, rosnące zbiory danych tylko do odczytu	Iceberg	Lepsze zarządzanie metadanymi i zgodność z wieloma silnikami
Machine Learning + ETL + BI	Delta Lake	Wersjonowanie, merge, integracja z MLflow i Power BI
Praca z **Trino/Flink/Presto**	Iceberg	Kompatybilność i otwarty standard tabel

Zadanie 3 Napisz krytykę architektury medalionu, może nie trzeba tworzyć medalionu, jakie ma błędy, wady

1. Złożoność implementacji
Wymaga tworzenia i zarządzania wieloma warstwami danych – co zwiększa nakład pracy, czas i błędy.

2. Opóźnienia przetwarzania
Dane muszą przejść przez wiele warstw, co powoduje większe opóźnienie od momentu załadunku do konsumpcji.

3. Zduplikowane dane
Te same dane występują w różnych warstwach – prowadzi to do wzrostu kosztów magazynowania.

4. Koszt obliczeniowy
Każda warstwa wymaga osobnego przetworzenia i obliczeń – co podnosi koszt działania Databricks/Spark.

5. Potrzeba wersjonowania
Brak spójnego mechanizmu wersjonowania między warstwami może prowadzić do niespójności danych.

6. Brak standaryzacji nazewnictwa
Warstwy Bronze/Silver/Gold nie mają technicznego znaczenia – mogą być różnie rozumiane przez zespoły.

7. Trudność w debugowaniu
Błędy przetwarzania są trudniejsze do prześledzenia, bo mogą wystąpić na różnych etapach transformacji.

8. Nadmierne przetwarzanie danych
Każda nowa warstwa może powodować niepotrzebne „kopiowanie” danych, nawet jeśli nie są one potrzebne.

9. Nadmiarowy kod
Każda warstwa wymaga własnych notebooków/pipeline'ów – co zwiększa złożoność kodu.

10. Niska elastyczność zmian
Zmiana schematu danych (schema evolution) wymaga często modyfikacji we wszystkich warstwach.

11. Słaba automatyzacja
Medalion wymaga ręcznej lub półautomatycznej orkiestracji wielu zadań (np. w ADF/Airflow).

12. Niejasna wartość warstw
Czasami nie wiadomo, co naprawdę różni Silver od Gold – różnice bywają sztuczne.

13. Wysoka krzywa uczenia
Nowi członkowie zespołu muszą zrozumieć koncepcję medalionu, co może spowolnić onboarding.

14. Niepotrzebne warstwy
W prostych projektach wystarcza jedna lub dwie warstwy – pełny medalion może być przesadą.

15. Utrudnione testowanie
Testowanie danych w każdej warstwie jest trudniejsze i wymaga osobnych testów jakości.

16. Brak widoczności lineage
Nie zawsze wiadomo, jak dane z Gold pochodzą z Bronze – brak przejrzystego śledzenia pochodzenia.

17. Wolniejsze wdrożenie MVP
Wymaga stworzenia pełnej struktury przed pierwszym użyciem, co opóźnia pierwszą wersję systemu.

18. Problemy z czasem życia danych
Trudno zarządzać retencją i usuwaniem danych – mogą zostać w wielu warstwach.

19. Zależność od konkretnej platformy
Medalion promowany głównie przez Databricks – może nie mieć sensu poza tym środowiskiem.

20. Brak zgodności z niektórymi standardami branżowymi
Niektóre firmy preferują inne podejścia (np. raw + curated + semantic) bardziej zgodne z ich architekturą DWH.