**Zadanie 2**

Delta Lake to natywny format transakcyjny dla Databricks, z pełnym wsparciem platformy i silnym rozwojem funkcji.

Zalety Delta Lake:

ACID transactions – pełna zgodność transakcyjna, ułatwia integrację z procesami ETL.

Time Travel – łatwy dostęp do wersji danych z przeszłości.

Optymalizacje systemowe – np. auto-compaction, Z-Ordering.

Ścisła integracja z Databricks – działa „out-of-the-box” bez dodatkowej konfiguracji.

Wsparcie dla streaming & batch w jednym formacie (unified batch & streaming).

Scenariusze, w których Delta Lake sprawdzi się najlepiej:

Gdy projekt opiera się w 100% na Azure Databricks.

Potrzeba szybkiego wdrożenia bez złożonej konfiguracji.

Przetwarzanie danych w trybie batch + streaming (structured streaming).

Gdy ważna jest łatwa obsługa wersjonowania danych.

W systemach Data Lakehouse, gdzie wydajność zapytań analitycznych ma kluczowe znaczenie.

Apache Iceberg to format tabel zaprojektowany z myślą o dużej skali i kompatybilności między platformami (multi-engine).

Zalety Iceberg:

Multi-engine compatibility – działa z Presto, Trino, Flink, Spark, Snowflake, itd.

Partition evolution – możliwość zmiany schematu partycjonowania w czasie bez utraty historii.

Better schema evolution – bardziej zaawansowane zarządzanie zmianami w strukturze danych.

Hidden partitioning – upraszcza zapytania i redukuje błędy użytkownika.

Lepsza współpraca w środowiskach multi-cloud / multi-tool.

Scenariusze, w których Iceberg będzie lepszym wyborem:

Gdy dane są używane przez wiele różnych silników poza Databricks.

Środowiska multi-cloud, gdzie potrzebna jest interoperacyjność.

Bardzo duże zbiory danych z częstą zmianą struktury tabel (np. zmiany partycji, kolumn).

Projekty z dużym naciskiem na otwarte standardy i vendor-neutralność.

**Zadanie 3**

1. **Złożoność operacyjna**  
   Wieloetapowe przetwarzanie danych (3 warstwy) zwiększa złożoność zarządzania pipeline'ami, testowaniem i monitoringiem.
2. **Opóźnienia w dostępności danych**  
   Dane muszą przejść przez wiele warstw, co wprowadza opóźnienia – nie zawsze spełnia wymagania near real-time.
3. **Kosztowne przechowywanie**  
   Dane są często duplikowane między warstwami, co zwiększa koszty magazynowania (szczególnie przy pełnym Delta Time Travel).
4. **Brak uniwersalności**  
   Nie każda organizacja potrzebuje trzech poziomów – dla prostszych przypadków to overengineering.
5. **Potencjalne rozmycie odpowiedzialności**  
   Niejasne może być, która warstwa jest źródłem prawdy – zwłaszcza przy wielu źródłach złożonych z różnych brązowych tabel.
6. **Nadmiarowe transformacje**  
   Konieczność pisania dodatkowych transformacji tylko po to, by przejść z brązu do srebra – bez realnej wartości biznesowej.
7. **Trudniejszy troubleshooting**  
   Gdy coś pójdzie nie tak w warstwie Gold, trzeba debugować przez wszystkie warstwy – co wydłuża czas analizy błędu.
8. **Skomplikowany lineage danych**  
   Przepływ danych między warstwami może być trudny do odtworzenia, jeśli nie stosuje się narzędzi do Data Lineage.
9. **Ryzyko niespójności między warstwami**  
   Brak synchronizacji (np. retry w brązie bez aktualizacji srebra) może prowadzić do niespójnych raportów.
10. **Wysoki próg wejścia dla zespołów**  
    Nowi inżynierowie danych muszą zrozumieć logikę całej architektury, co może wydłużać onboarding.
11. **Zbyt ogólny model danych**  
    Architektura nie narzuca struktury domenowej – dane są klasyfikowane tylko wg „stopnia przetworzenia”, co utrudnia skalowanie wg DDD (domain-driven design).
12. **Brak wsparcia dla event-driven architecture**  
    Medallion to model głównie **batchowy** – nie jest naturalnie dostosowany do architektur zdarzeniowych.
13. **Możliwa nadmierna optymalizacja**  
    Często dane są przekształcane do Gold „na zapas”, mimo że nie wszystkie modele biznesowe są jeszcze potrzebne.
14. **Nieefektywne wykorzystanie zasobów**  
    Wielowarstwowe transformacje często prowadzą do marnowania zasobów obliczeniowych (zwłaszcza w małych projektach).
15. **Wysoka zależność od konkretnych narzędzi (np. Databricks)**  
    Choć koncepcja jest teoretycznie niezależna, w praktyce często jest silnie powiązana z konkretnymi usługami (Delta Lake, Unity Catalog).
16. **Brak dynamicznej personalizacji**  
    Dane w Gold są często zdenormalizowane i trudne do dynamicznego filtrowania wg potrzeb użytkownika końcowego.
17. **Zbyt techniczne podejście**  
    Brak jasnego odwzorowania potrzeb biznesowych – klasyfikacja Bronze/Silver/Gold ma charakter techniczny, nie biznesowy.
18. **Trudność w testowaniu warstw pośrednich**  
    Trzeba testować każdą warstwę osobno – co komplikuje testy integracyjne i regresyjne.
19. **Brak elastyczności dla szybkich prototypów**  
    Wymaganie przejścia przez trzy warstwy spowalnia szybkie eksperymenty analityczne.
20. **Możliwość „utrwalenia błędów”**  
    Błędy w danych mogą być propagowane i utrwalane przez wszystkie warstwy, jeśli nie ma wbudowanej walidacji i czyszczenia.