Zadanie 2

Delta Lake to technologia opracowana przez Databricks, silnie zintegrowana z platformą Azure Databricks i natywnie współpracująca z Apache Spark. Umożliwia m.in. operacje `upsert` (merge), wersjonowanie danych, mechanizm time-travel (czyli dostęp do historycznych wersji danych) oraz zapewnia zgodność z transakcjami ACID. Jest zoptymalizowana zarówno pod kątem przetwarzania strumieniowego, jak i wsadowego. Delta Lake doskonale sprawdza się w środowiskach opartych na Databricks, szczególnie w kontekście analityki biznesowej, transformacji danych oraz projektów z zakresu machine learningu.

Apache Iceberg to otwarty format danych, wspierany przez społeczność oraz firmy takie jak Netflix i Apple. Jest bardziej neutralny technologicznie – obsługuje wiele silników, m.in. Spark, Flink, Trino, Presto oraz Snowflake. Iceberg jest szczególnie przydatny w projektach o dużej skali, gdzie kluczowa jest elastyczność, skalowalność i brak uzależnienia od konkretnej platformy. Podobnie jak Delta Lake, wspiera funkcje takie jak wersjonowanie, time-travel oraz transakcje ACID.

Rekomendacja dla klienta:

- Delta Lake będzie najlepszym wyborem w przypadku, gdy:
    - Klient korzysta z Azure Databricks.
    - Zależy mu na prostym, wydajnym i natywnie wspieranym rozwiązaniu.
- Apache Iceberg sprawdzi się lepiej, jeśli:
    - Klient planuje rozwiązanie wieloplatformowe z integracją różnych silników.
    - Potrzebna jest duża skalowalność i niezależność technologiczna.

Zadanie 3 – Krytyka architektury medalionowej

Architektura medalionowa, dzieląca dane na trzy warstwy: Bronze (dane surowe), Silver (dane przetworzone) i Gold (dane analityczne), jest powszechnie stosowana, jednak niepozbawiona istotnych wad. Poniżej przedstawiono 20 punktów krytyki wraz z krótkim uzasadnieniem:

1. Złożoność utrzymania – konieczność zarządzania wieloma pipeline'ami między warstwami.
2. Duplikacja danych – te same dane są zapisywane w wielu miejscach, co zwiększa koszty.
3. Wysokie zużycie zasobów – każda warstwa to osobny proces, co obciąża infrastrukturę.

4. Opóźniony dostęp do danych – pełna analiza możliwa dopiero po przetworzeniu danych do poziomu Gold.

5. Zbędna dla prostych zastosowań – w mniejszych projektach często wystarczy jedna lub dwie warstwy.

6. Utrudnione debugowanie – problemy mogą wystąpić na różnych etapach przetwarzania.

7. Wysoki próg wejścia – dla małych zespołów może być to zbyt skomplikowany model.

8. Brak automatyzacji – wiele przepływów wymaga ręcznego utrzymania.

9. Ograniczona elastyczność – sztywna struktura utrudnia szybkie zmiany logiki.

10. Wysokie zapotrzebowanie na przestrzeń dyskową – każda warstwa to osobne przechowywanie danych.

11. Ograniczenie eksploracji danych – narzucenie struktury może ograniczyć analityczne podejście.

12. Złożone monitorowanie – konieczność prowadzenia osobnych logów i alertów dla każdej warstwy.

13. Ryzyko niespójności danych – możliwość rozjazdu danych między warstwami.

14. Nadmierna złożoność dla danych tymczasowych – struktura może być niepotrzebna przy danych ad-hoc.

15. Nieefektywność dla strumieniowania – trzy warstwy mogą być przeszkodą przy danych w czasie rzeczywistym.

16. Generowanie dużej liczby plików pośrednich – wzrost kosztów przechowywania i zarządzania.

17. Utrudnione testowanie – trudności z tworzeniem testów jednostkowych w wielowarstwowej architekturze.

18. Zależność od konkretnego narzędzia/platformy – np. Databricks, co ogranicza mobilność rozwiązania.

19. Brak spójnych standardów – interpretacja warstw może się różnić między organizacjami.

20. Zwiększone ryzyko długu technologicznego – każda dodatkowa warstwa to nowy kod do utrzymania i aktualizacji.