Senior Data Engineer Assessment (Data Bricks)

Part 1: Pipeline Design & Data Modeling

Task 1: Enterprise Data Model Design (20 minutes)

- **Objective:** Design a detailed data model for customer journeys and marketplace activity.
- **Deliverable:** A comprehensive diagram and description, covering:
- **Key Entities** like Customers, Transactions, and Interactions.
- **Relationships** and how they support ML use cases.
- **ML Readiness**: How the model prepares data for ML features.

To create an effective data model for the marketplace, I began by focusing on the foundational entities: **Customers**, **Transactions**, **Interactions**, **Events**, and **Sessions**. I carefully structured the model to ensure it could support business analysis and machine learning use cases.

Step 1: Customers - The Core of the Model

I started with the **Customers** table, as it formed the heart of the model. It captured critical details like the customer's name, email, registration date, and segmentation. This table acted as the central hub, linking all other entities and providing insights into individual customer behavior.

- **Attributes**:
  customer_id (Primary Key), name, email, phone, registration_date, location, and customer_segment.

Step 2: Transactions - Understanding Financial Activity

The next step was creating the **Transactions** table to record customer purchases. This table was designed to enable analysis of spending patterns, payment methods, and transaction trends over time. By linking it to the **Customers** table via customer_id, it supported key business questions like customer lifetime value and transaction behavior.

- **Attributes**:
  transaction_id (Primary Key), customer_id (Foreign Key), amount, currency, payment_method, transaction_date, and transaction_status.

Step 3: Interactions - Tracking Communication

Afterward, I designed the **Interactions** table to capture customer communications with the business. This table provided visibility into how customers interacted with support or marketing teams. It also helped uncover patterns such as preferred communication channels and response efficiency, which are critical for improving customer engagement.

- **Attributes**:
  interaction_id (Primary Key), customer_id (Foreign Key), interaction_date, interaction_type, response_time, and interaction_outcome.

Step 4: Events - Analyzing Behavioral Data

To add granularity, I introduced the **Events** table. This table recorded individual actions customers performed on the platform, such as searches, clicks, or purchases. It was designed to enable behavioral segmentation and help track conversion rates.

- **Attributes**:
  event_id (Primary Key), customer_id (Foreign Key), event_type, and event_timestamp.

Step 5: Sessions - Capturing Activity Trends

Finally, I included the **Sessions** table to log customer activity during a session. This table provided insights into engagement levels, session duration trends, and device preferences. It helped identify patterns like session drop-off points and areas for improving user experience.

- **Attributes**:
  session_id (Primary Key), customer_id (Foreign Key), device_type, session_start, session_end, and session_duration.

Relationships and Use Cases

I carefully designed the relationships between these entities to ensure they support both business insights and machine learning use cases. All relationships follow a one-to-many (1: N) pattern, where one record in the Customers table can be associated with multiple records in the other tables:

1. **Customers ↔ Transactions**: Enabled analysis of customer lifetime value, churn prediction, and product recommendations.
2. **Customers ↔ Interactions**: Supported the optimization of engagement strategies by analyzing response times and customer satisfaction trends.
3. **Customers ↔ Events**: Facilitated personalized recommendations and real-time behavioral segmentation based on granular actions.
4. **Customers ↔ Sessions**: Provided insights into session trends, engagement levels, and device-specific preferences.

Making the Model ML-Ready

The design prioritized machine learning readiness by enabling feature engineering and real-time data processing:

1. **Structured Relationships**:

   Aggregated features like total_spend or average_purchase_value were derived from transactional data.

2. **Temporal Features**:

   Timestamps enabled the creation of time-based features like days_since_last_transaction or transaction_frequency_last_30_days.

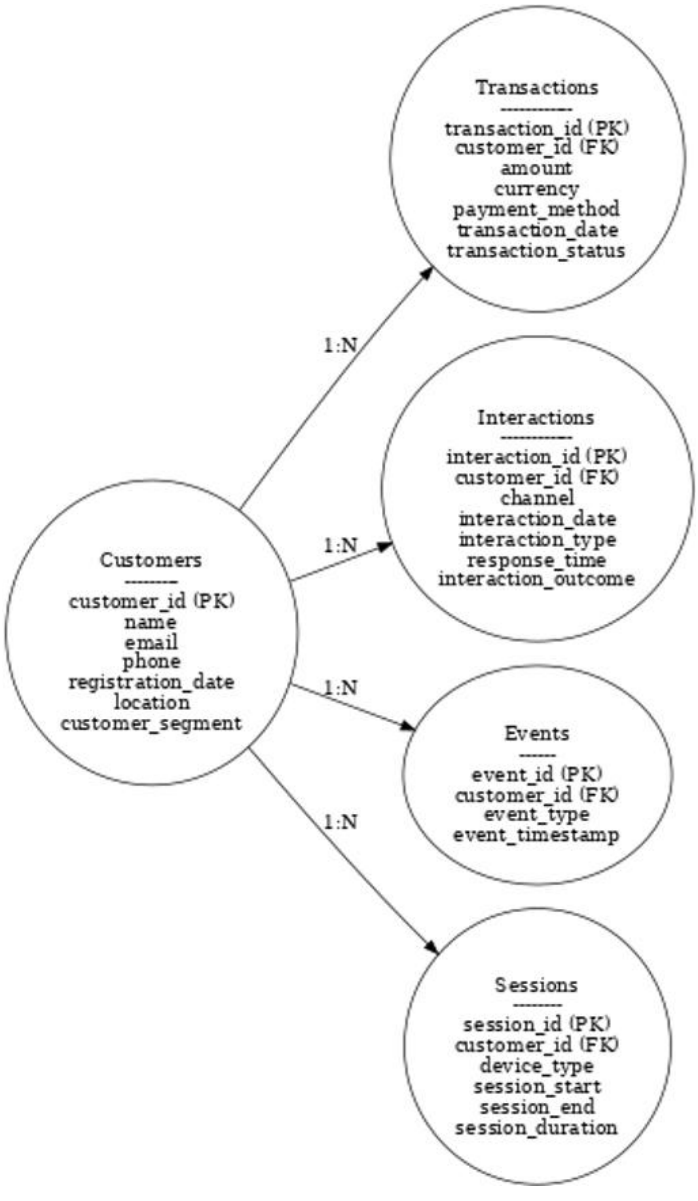
3. **Behavioral Insights**:

   Session and event data allowed for advanced features like average_session_duration or event_conversion_rate.

4. **Real-Time Processing**:

   The inclusion of real-time events and sessions supported use cases such as dynamic product recommendations.

Senior Data Engineer Assessment (Data Bricks)



Designed using Word

Additionally, I ensured that the design was scalable and efficient for large-scale data processing. By normalizing the schema and clearly defining relationships, I minimized redundancy and ensured compatibility with distributed processing frameworks like Databricks. I also kept flexibility in mind, allowing the model to handle both batch and streaming data, which is critical for real-time machine learning applications.

Task 2: Pipeline Architecture Diagram (25 minutes)

Objective: Design a robust Databricks pipeline architecture using Bronze-Silver-Gold layers.

- **Deliverable:** A diagram and explanation, including:
- **Ingestion Strategy** for both batch and streaming data.
- **Transformation Layers** with detailed transformations.
- **ML Integration**: How Gold tables serve as an ML feature store.

Senior Data Engineer Assessment (Data Bricks)

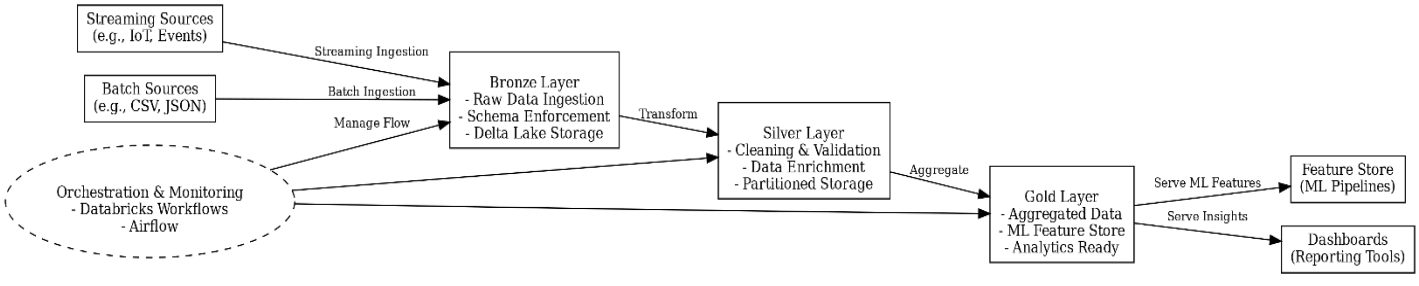
My goal was to design a scalable pipeline using the Medallion Architecture with three layers: Bronze, Silver, and Gold, to handle both batch and streaming data. Each layer serves a distinct purpose, ensuring the data is ingested, cleaned, and optimized for analytics and machine learning workflows.

The Bronze Layer serves as the landing zone for raw data from batch (e.g., CSV, JSON) and streaming sources. Batch data is ingested periodically, handling large volumes reliably, while real-time data is processed using Spark Structured Streaming. Schema enforcement ensures consistency, and Delta Lake provides fault tolerance and flexibility.

The Silver Layer cleans and validates data. This involves removing duplicates, handling null values, and standardizing formats like dates and numeric fields. Business rules ensure data integrity, and data is enriched by joining related datasets to make it ready for analysis.

The Gold Layer delivers curated datasets optimized for machine learning and analytics. Features such as purchase frequency, total spend, and recency are engineered, with data partitioned and Z-Ordered for efficient querying. Delta optimization ensures fast access for real-time and batch use cases.

The pipeline was designed to be resilient, scalable, and ready for orchestration using tools like Databricks Workflows or Apache Airflow, ensuring smooth automation and monitoring. This architecture enables efficient data preparation for analytics and machine learning.



Designed using Word

Part 2: Data Processing & Feature Engineering

Task 3: Real-Time Data Integration Task (Updated)

Objective: Set up a real-time ingestion pipeline using Structured Streaming and Delta Lake with advanced validation to ensure data quality.

Deliverable: Code with explanations, including:

- Real-Time Ingestion: Using Structured Streaming setup.
- Advanced Validation Apply schema enforcement, null handling, and deduplication.

To manage real-time data ingestion, I built a pipeline using **Spark Structured Streaming** and **Delta Lake**, based on the **Medallion Architecture**. In the Bronze Layer, I simulated transactional data with fields like transaction_id, user_id, product_id, quantity, price, and timestamp, ensuring realistic values through logical derivations. The data stream generated 10 rows per second and was written to a Delta table with checkpointing for fault tolerance and exactly-once processing. Logging was added to track key steps in the pipeline, making it easier to debug and monitor progress.

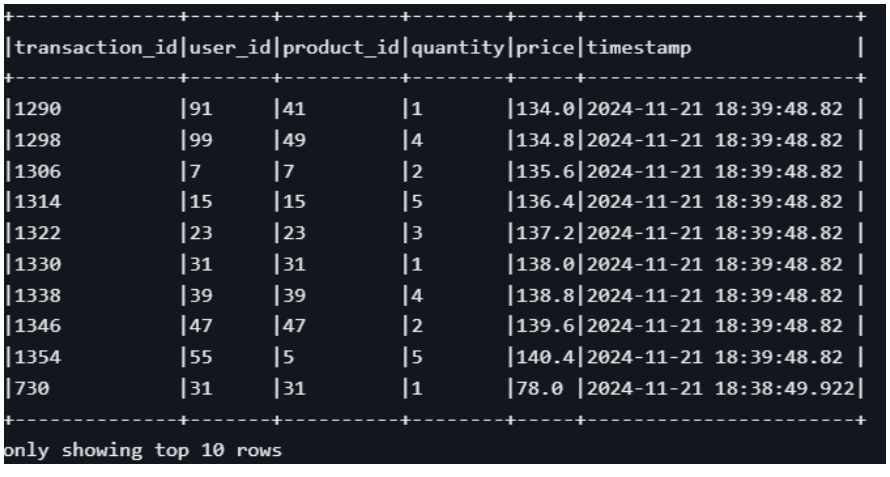
In the **Silver Layer**, the focus was on data cleaning and validation. I enforced schema consistency, performed **null handling** by filtering out invalid or null values, and implemented **deduplication** to remove duplicate records based on transaction_id. Additionally, I applied **business rules** to ensure positive values for quantity and price. These

steps ensured high-quality data, which was then stored in a Delta table for downstream use. Logging was used to capture metrics such as the total records processed, duplicates removed, and validation outcomes.

In the Gold Layer, I engineered **user-level features** such as purchase frequency, total spend, average transaction value, and recency days. To optimize performance, I used **Delta OPTIMIZE** to compact small files and **Z-Ordering** on total_spend for faster queries. The data was partitioned by user_id to further enhance query efficiency. These features are ready for machine learning and analytics, supporting tasks like user segmentation and behavior prediction.

Looking ahead, I would extend the pipeline by integrating monitoring tools for real-time insights, connecting to sources like Kafka for production-grade streaming, and leveraging tools like MLflow for seamless ML experimentation and deployment. This structured and robust pipeline ensures scalability, performance, and data quality at every stage.

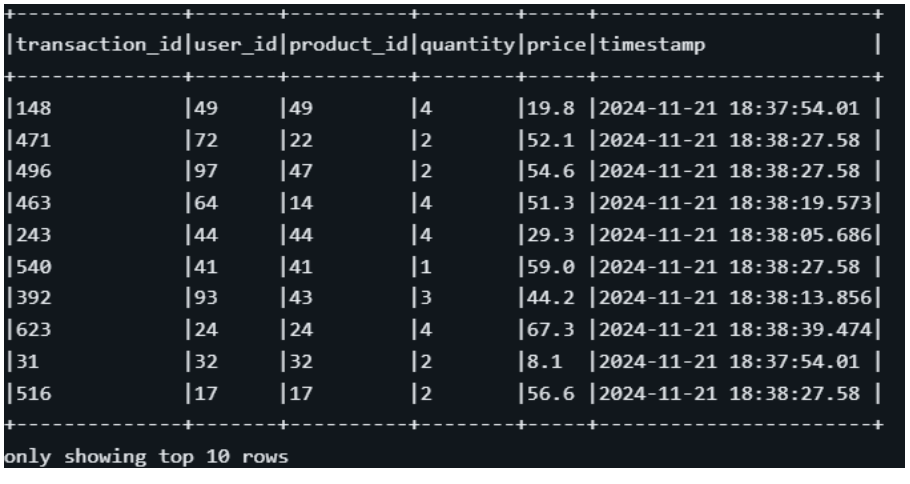
Bronze Layer Data:



transaction_id	user_id	product_id	quantity	price	timestamp
1290	91	41	1	134.0	2024-11-21 18:39:48.82
1298	99	49	4	134.8	2024-11-21 18:39:48.82
1306	7	7	2	135.6	2024-11-21 18:39:48.82
1314	15	15	5	136.4	2024-11-21 18:39:48.82
1322	23	23	3	137.2	2024-11-21 18:39:48.82
1330	31	31	1	138.0	2024-11-21 18:39:48.82
1338	39	39	4	138.8	2024-11-21 18:39:48.82
1346	47	47	2	139.6	2024-11-21 18:39:48.82
1354	55	5	5	140.4	2024-11-21 18:39:48.82
730	31	31	1	78.0	2024-11-21 18:38:49.922

only showing top 10 rows

Silver Layer Data:



transaction_id	user_id	product_id	quantity	price	timestamp
148	49	49	4	19.8	2024-11-21 18:37:54.01
471	72	22	2	52.1	2024-11-21 18:38:27.58
496	97	47	2	54.6	2024-11-21 18:38:27.58
463	64	14	4	51.3	2024-11-21 18:38:19.573
243	44	44	4	29.3	2024-11-21 18:38:05.686
540	41	41	1	59.0	2024-11-21 18:38:27.58
392	93	43	3	44.2	2024-11-21 18:38:13.856
623	24	24	4	67.3	2024-11-21 18:38:39.474
31	32	32	2	8.1	2024-11-21 18:37:54.01
516	17	17	2	56.6	2024-11-21 18:38:27.58

only showing top 10 rows

Senior Data Engineer Assessment (Data Bricks)

Task 4: Feature Engineering and ML Table Creation (20 minutes)

Objective: Generate a performance-optimized ML-ready feature table.

Deliverable: Code and explanation of:

- **Feature Calculations** (e.g., purchase frequency).
- **Advanced Optimization** (e.g., partitioning, caching).

I started with processed validated data from the Silver Layer of the pipeline using Spark, focusing on creating meaningful **user-level features**. Key metrics such as purchase frequency, total spend, average transaction value, and recency days were engineered to summarize user behaviour. For instance, purchase frequency was derived by counting unique transactions for each user, while total spend an average transaction value were computed as the sum and average of transaction amounts, rounded to two decimal places for precision. Recency days was calculated as the number of days since the user's last transaction. These features were designed to capture essential transactional patterns for machine learning models.

To optimize the feature table for performance, I implemented several strategies. The data was partitioned by user_id, enabling efficient queries on user-specific data. Additionally, I applied **Z-Ordering** on total_spend, frequently used in queries, to cluster related data for faster filtering and retrieval. The **Delta OPTIMIZE** command was used to compact small files into fewer larger files, further improving query performance and storage efficiency.

Throughout the process, I **cached intermediate results** to enhance computation speed and validated the schema to ensure data integrity. Logging was used to track feature generation metrics, ensuring transparency and traceability. This ML-ready feature table resides in the Gold Layer and supports tasks such as user segmentation, churn prediction, and customer lifetime value modeling. Going forward, I would integrate monitoring tools for feature stability, enable automatic drift detection, and connect the table to an ML pipeline using tools like MLflow for seamless experimentation and deployment. This setup ensures the feature table is robust, scalable, and optimized for downstream machine learning workflows.

Data Quality Metrics:

```
INFO:__main__:Data Quality Metrics for Gold Layer:
INFO:__main__:Total Records: 100
INFO:__main__:Columns: ['user_id', 'purchase_frequency', 'total_spend', 'avg_transaction_value', 'last_purchase_date', 'recency_days']
+-------+------------------+-----------+--------------------+---------------------+------------+
|user_id|purchase_frequency|total_spend|avg_transaction_value|last_purchase_date    |recency_days|
+-------+------------------+-----------+--------------------+---------------------+------------+
|16     |11                |621.5      |56.5                |2024-11-21 18:39:15.64|0           |
|27     |11                |633.6      |57.6                |2024-11-21 18:39:15.64|0           |
|7      |11                |611.6      |55.6                |2024-11-21 18:39:15.64|0           |
|3      |11                |607.2      |55.2                |2024-11-21 18:39:15.64|0           |
|23     |11                |629.2      |57.2                |2024-11-21 18:39:15.64|0           |
|88     |10                |587.0      |58.7                |2024-11-21 18:39:15.64|0           |
|91     |10                |590.0      |59.0                |2024-11-21 18:39:15.64|0           |
|68     |10                |567.0      |56.7                |2024-11-21 18:39:15.64|0           |
|73     |10                |572.0      |57.2                |2024-11-21 18:39:15.64|0           |
|32     |10                |531.0      |53.1                |2024-11-21 18:39:15.64|0           |
+-------+------------------+-----------+--------------------+---------------------+------------+
only showing top 10 rows
```

Acknowledgment:

Thank you for giving me the chance to work on this assessment. It was a great opportunity to use my real-world experience in solving data challenges and building efficient pipelines. However, It took longer than expected, But I have done it. I appreciate the effort put into creating this task, and I look forward to discussing my approach and ideas with you further.