## Use Case - Presenter and Artist Matchmaking

The objective would be to develop a data-driven recommendation system to match presenters with artists based on event preferences, artist attributes, and feedback.

### Real-World Scenarios and Benefits

1. Event Planning: Event managers can query AzureSQL to quickly identify the best artists for a presenter’s preferences, ensuring engaging lineups.
2. Data-Driven Insights: Use detailed recommendations to analyze presenter preferences and artist performance, driving better decision-making.
3. Efficiency: Automates the process of artist selection, reducing manual effort and increasing accuracy.

#### 1. Data Preparation

1. Data Extraction Tables -
   * Presenter: PresenterId, OrganizationName, PresenterType
   * Artist: ArtistId, ArtistName, GenreType, ActType, PerformanceRating
   * ArtistFeedback: ArtistId, FeedbackScore, Comment
   * BlueCardArtist: EventId, ArtistId, PresenterId
   * Lu\_GenreType: GenreTypeId, GenreName
   * Lu\_ActType: ActTypeId, ActTypeName
2. Data Cleaning by querying AzureSQL tables and perform:
   * Missing Value Handling: Use SQL queries to handle null values, e.g., filling missing PerformanceRating with averages.
   * Data Standardization: Normalize numeric columns (e.g., FeedbackScore, PerformanceRating) to a 0-1 range using SQL window functions.
3. Sentiment Analysis to extract sentiment scores from the Comment column in ArtistFeedback.
   * Use AzureML’s Python SDK to call Azure Text Analytics API and calculate sentiment scores for feedback comments.
   * Store the results (SentimentScore) back into the ArtistFeedback table in AzureSQL.

#### 2. Model Development

1. Collaborative Filtering - to recommend artists to presenters based on past collaborations.
   * Input: Data from BlueCardArtist table mapping presenters to artists.
   * Output: Predicted scores indicating the likelihood of a presenter working with an artist again.
   * Save the trained model as a serialized file (.pkl) for deployment.

OR

1. Content-Based Filtering - to match artists to presenters based on attributes like genre, act type, and performance metrics.
   * Create a similarity score matrix in AzureML by using cosine similarity to compare GenreType, ActType, PerformanceRating, and SentimentScore.
   * Combine this with collaborative filtering predictions using a weighted formula:
2. Model Evaluation:
   * Metrics:
     + Precision@K: Measures how many of the top-K recommendations are relevant.
     + Recall@K: Measures how well the model captures all relevant items.
   * Validation:
     + Split the data (80-20) for training and testing within AzureML.
     + Use AzureML’s evaluation metrics dashboard to visualize model performance.

#### 3. System Access and Reporting

1. Data Access:
   * Enable users to query recommendations directly from the RecommendedArtist table in AzureSQL.
   * Use SQL views to join the recommendations with artist and presenter metadata for detailed insights.
2. Reporting using PowerBI - if needed
   * Recommended artists for each presenter.
   * Metrics like artist performance, feedback sentiment, and historical collaboration trends.