MSBA 230

# Project 2

## Group 5

**Bala Gaurav Reddy Pasam**

**Khushbu Singh**

**Raja Krishna Srivastav Arra**

**Atlaf Khan**

# Database Design Report for Stockton Symphony Database

---

### Understanding the Data

From the data provided, we understood that the Stockton Symphony's operations and performances involve multiple facets that need to be recorded and analyzed. The datasets included:

1. **Concert Data**: This dataset provided concert details such as the series type (e.g., Pops or Classical), date, time, concert name, and combined ticket sales.
   - *Example*: A record like "Pops Series - Saturday Evening" with its associated date, time, and sales data.
2. **Ticket and Revenue Data**: This dataset contained historical revenue information for the past 10 years, including single-ticket revenue, subscription revenue, total revenue, and the number of tickets sold for each type.
   - *Example*: A record might show 150 single tickets sold, generating $3,000 in revenue, alongside 50 subscription tickets contributing $5,000.
3. **Music Performed Data**: This data listed the individual pieces performed during each concert, including the composer, piece name, and performance date.
   - *Example*: "Beethoven's Symphony No. 9" performed on "December 12, 2023."

We carefully segregated these datasets into logical entities (tables) to enable robust analysis and reporting.

---

**Table Design and Attributes**

Below, we detail the tables created, their attributes, and their roles in the database:

**1. Concerts Table**

- **Purpose**: To store details of each concert.
- **Attributes and Constraints**:

| **Attribute** | **Description** | **Constraint** |
|---|---|---|
| ConcertID | Unique identifier for each concert | Primary Key, Auto-incremented |
| Name | Name of the concert | NOT NULL, must have a length > 0 |
| Date | Date of the concert | NOT NULL, must be a valid date |
| Time | Time of the concert | NOT NULL, valid time format |
| Day | Day of the week | CHECK (valid days: Monday-Sunday) |
| SeriesType | Classification (e.g., Pops, Jazz) | CHECK (valid types: Pops, Classical, Jazz, etc.) |
| SeriesNumber | Sequence number in the series | CHECK (value > 0) |
| VenueName | Name of the venue | NOT NULL, must have a length > 0 |

**Example Usage**:
A record for a concert like "Pops Series - Saturday Evening" scheduled for a Saturday at 7:00 PM in the Stockton Symphony venue would be stored here. The SeriesType (e.g., Pops) helps classify it for analysis.

---

**2. StocktonCustomers Table**

- **Purpose**: To store customer details and their relationship with the symphony.
- **Attributes and Constraints**:

| **Attribute** | **Description** | **Constraint** |
|---|---|---|
| CustomerID | Unique identifier for each customer | Primary Key, Auto-incremented |
| Name | Full name of the customer | NOT NULL, must have a length > 0 |
| Age | Age of the customer | CHECK (value > 0) |
| SubscriptionDetails | Subscription type (Gold, Silver, etc.) | Default: Unsubscribed, CHECK (valid types) |
| LoyaltyScore | Numeric score indicating loyalty | Default: 0, CHECK (value >= 0) |
| PastAttendance | Number of past concerts attended | Default: 0, CHECK (value >= 0) |

**Example Usage**:
Customer "Alice Smith" might have a "Gold" subscription with a loyalty score of 100, indicating frequent attendance and engagement.

---

### 3. StocktonTickets Table

- **Purpose**: To store details of ticket purchases for concerts.
- **Attributes and Constraints**:

| Attribute | Description | Constraint |
|---|---|---|
| TicketID | Unique identifier for each ticket | Primary Key, Auto-incremented |
| ConcertID | References the associated concert | Foreign Key (Concerts.ConcertID) |
| CustomerID | References the customer who bought it | Foreign Key (StocktonCustomers.CustomerID) |
| TicketType | Type of ticket (Single, Subscription) | CHECK (valid types: Single, Subscription) |

**Example Usage**:
If customer "Alice Smith" purchases a single ticket for "Pops Series - Saturday Evening," this purchase is recorded in the StocktonTickets table, linked to the concert and customer.

---

### 4. StocktonRevenue Table

- **Purpose**: To store ticket sales and revenue details for concerts.
- **Attributes and Constraints**:

| Attribute | Description | Constraint |
|---|---|---|
| RevenueID | Unique identifier for each revenue record | Primary Key, Auto-incremented |
| ConcertID | References the associated concert | Foreign Key (Concerts.ConcertID) |
| SingleTicketsSold | Number of single tickets sold | CHECK (value >= 0) |
| SingleTicketRevenue | Revenue from single tickets | CHECK (value >= 0) |
| SubTicketsSold | Number of subscription tickets sold | CHECK (value >= 0) |
| SubRevenue | Revenue from subscription tickets | CHECK (value >= 0) |
| TotalRevenue | Total revenue (calculated automatically) | Computed as (SingleTicketRevenue + SubRevenue) |

**Example Usage**:
For "Pops Series - Saturday Evening," 200 single tickets and 50 subscription tickets might be sold, generating revenues of $4,000 and $5,000 respectively. This information is recorded here.

---

### 5. Composers Table

- **Purpose**: To store details about composers whose works are performed in concerts.
- **Attributes and Constraints**:

| Attribute | Description | Constraint |
|---|---|---|
| ComposerID | Unique identifier for each composer | Primary Key, Auto-incremented |
| Name | Full name of the composer | NOT NULL, must have a length > 0 |
| Country | Country of origin | NOT NULL, must have a length > 0 |

**Example Usage**:
Composer "Ludwig van Beethoven" from Germany would have a record in this table.

### 6. PerformedPieces Table

- **Purpose**: To store details of pieces performed in concerts.
- **Attributes and Constraints**:

| Attribute | Description | Constraint |
|---|---|---|
| PieceID | Unique identifier for each piece | Primary Key, Auto-incremented |
| ConcertID | References the associated concert | Foreign Key (Concerts.ConcertID) |
| ComposerID | References the composer | Foreign Key (Composers.ComposerID) |
| PieceName | Name of the musical piece | NOT NULL, must have a length > 0 |
| PerformanceDate | Date of the performance | NOT NULL |

**Example Usage**:
The piece "Symphony No. 9" performed by Beethoven on January 15, 2024, would be recorded here.

### 7. StocktonFeedback Table

- **Purpose**: To store customer feedback on concerts.
- **Attributes and Constraints**:

| Attribute | Description | Constraint |
|---|---|---|
| FeedbackID | Unique identifier for feedback | Primary Key, Auto-incremented |
| CustomerID | References the customer | Foreign Key (StocktonCustomers.CustomerID) |
| TicketID | References the ticket | Foreign Key (StocktonTickets.TicketID) |
| ConcertName | Name of the concert | NOT NULL, must have a length > 0 |
| FavouritePiece | Customer's favorite piece | NULLABLE |
| Comments | Additional feedback or comments | NULLABLE |

**Example Usage**:
Customer "Alice Smith" might provide feedback saying "The Symphony No. 9 was outstanding," recorded in this table.

---

**Connecting the Tables**

- **Relationships**:
  - Concerts links to StocktonTickets and StocktonRevenue through ConcertID, enabling connections between ticket sales and concerts.
  - PerformedPieces connects concerts to specific pieces and composers.
  - StocktonFeedback ties customer feedback to tickets and concerts.
- **Why It's Useful**:
  - The schema ensures seamless data navigation for analysis, such as identifying trends or assessing customer preferences.

---

**Meeting Predictive Analysis Requirements**

**1. Identifying Revenue Trends**

- By analyzing SingleTicketsSold, SubTicketsSold, and TotalRevenue in the StocktonRevenue table, we can identify revenue trends across concerts.
- **Example**:
  - Compare revenue between Saturday and Sunday concerts using the Day attribute in the Concerts table.
  - Analyze if specific SeriesType (e.g., Pops) consistently generates higher revenue.

---

**2. Predicting Revenue for Future Concerts**

- Historical data in StocktonRevenue, combined with concert attributes in Concerts, allows prediction of ticket demand and revenue for planned concerts.
- **Example**:
  - For a planned performance of "Carmina Burana," data from similar classical concerts like "Beethoven's 9th Symphony" can be analyzed to estimate ticket sales and revenue.

---

**3. Ranking Revenue-Generating Concerts**

- Using TotalRevenue in the StocktonRevenue table, concerts can be ranked to identify top-performing events.
- **Example**:
  - The database can easily retrieve the highest revenue-generating concert within a specific SeriesType or time frame.

---

**4. Exploring Subscription vs. Single-Ticket Preferences**

- The StocktonTickets table distinguishes between subscription and single-ticket sales via the TicketType attribute.

- Insights can reveal:
  - Which customer demographics prefer subscriptions over single tickets.
  - How subscription preferences correlate with loyalty scores and past attendance.

---

### 5. Additional Insights

- **Audience Preferences**:
  - Combining StocktonFeedback and PerformedPieces reveals popular pieces and composers.
  - **Example**:
    If "Symphony No. 9" appears frequently as the "FavouritePiece," it can influence future programming.
- **Scheduling Optimization**:
  - By analyzing ticket sales and revenue trends for specific days (Day attribute in Concerts), concerts can be scheduled for optimal audience turnout.

---

### 6. Customer Retention and Loyalty

- Loyalty scores in StocktonCustomers provide a quantitative measure to identify high-value customers.
- **Example**:
  - Offering personalized discounts or rewards to customers with high loyalty scores but declining attendance.

---

## Marketing View Point

### 1. Optimizing Marketing Campaigns

- Identify customer segments based on past attendance and feedback for targeted campaigns.
- **Example**:
  - Offering discounts to customers who frequently attend Pops series but have not attended Jazz concerts can boost Jazz attendance.

---

### 2. Dynamic Pricing Strategies

- By analyzing ticket sales patterns, dynamic pricing models can be implemented to adjust prices based on demand.

---

### 3. Venue Utilization Optimization

- Use venue-specific revenue data to determine which venues yield the highest returns and allocate more high-profile concerts to those venues.

---

### Conclusion

The Stockton Symphony database schema is designed to meet both operational needs and advanced analytical requirements. It facilitates robust data storage, seamless integration, and detailed reporting capabilities, enabling the organization to make informed decisions and optimize its operations.