Chinook Database Comprehensive Documentation

Overview of the Chinook Database

The Chinook database is a SQL-based dataset that models a digital media store. Its structure allows users to explore and
manipulate a variety of relational data, making it a valuable resource for testing SQL queries, exploring foreign key
relationships, and understanding a digital marketplace's structure. The primary entities include Artists, Albums, Tracks,
Genres, Playlists, Customers, Invoices, and Employees.

Database Entities and Relationships

----

1. Artists Table

   Purpose: Stores information about artists or bands represented in the store.

   Columns:

   - ArtistId (INTEGER, Primary Key): Unique identifier for each artist.
   - Name (TEXT): The name of the artist or band.

   Relationships:

   - One-to-Many relationship with the Albums table, where each artist can have multiple albums.

   Use Cases:

   - Retrieve all albums by a specific artist.
   - Count the total number of albums per artist to assess popularity.

2. Albums Table

Purpose: Contains album data, linking each album to a specific artist.

Columns:

- AlbumId (INTEGER, Primary Key): Unique identifier for each album.
- Title (TEXT): The album's title.
- ArtistId (INTEGER, Foreign Key): Links to the Artists table.

Relationships:

- Each album is linked to a single artist via ArtistId.
- One-to-Many relationship with the Tracks table (one album can contain many tracks).

Use Cases:

- Display all albums by a particular artist.
- Organize tracks by their album titles.

3. Tracks Table

Purpose: Stores individual tracks' details, including title, duration, and pricing.

Columns:

- TrackId (INTEGER, Primary Key): Unique identifier for each track.
- Name (TEXT): The track title.
- AlbumId (INTEGER, Foreign Key): Connects to an album in the Albums table.
- MediaTypeId (INTEGER, Foreign Key): Specifies the media format.
- GenreId (INTEGER, Foreign Key): Defines the genre classification.
- Composer (TEXT): The composer or artist associated with the track.
- Milliseconds (INTEGER): The track length in milliseconds.
- Bytes (INTEGER): The file size.
- UnitPrice (NUMERIC): The cost per track.

Relationships:

- Linked to Albums, MediaType, and Genre tables.
- Associated with multiple Invoices via InvoiceLine table, indicating track purchases.

Use Cases:

- Identify the best-selling tracks.
- Calculate the average duration of tracks by genre.

4. MediaType Table

Purpose: Stores information about the different types of media formats.

Columns:

- MediaTypeId (INTEGER, Primary Key): Unique identifier for each media type.
- Name (TEXT): The media format's name, such as MPEG audio or Protected AAC audio.

Relationships:

- Many-to-One relationship with the Tracks table, where each track has a media type.

Use Cases:

- Classify tracks by their media formats.
- Identify popular media types based on sales.

5. Genre Table

Purpose: Stores various genres to classify tracks.

Columns:

- GenreId (INTEGER, Primary Key): Unique identifier for each genre.
- Name (TEXT): The name of the genre.

Relationships:

- Many-to-One relationship with the Tracks table, where each track belongs to a specific genre.

Use Cases:

- Analyze sales by genre to understand genre popularity.
- Filter tracks by genre for genre-specific playlists.

6. Playlist Table

Purpose: Contains playlists that organize multiple tracks together.

Columns:

- PlaylistId (INTEGER, Primary Key): Unique identifier for each playlist.
- Name (TEXT): The name of the playlist.

Relationships:

- Many-to-Many relationship with Tracks, managed through the PlaylistTrack table.

Use Cases:

- Retrieve all tracks within a specific playlist.
- Create genre or theme-based playlists.

7. PlaylistTrack Table

Purpose: Acts as a bridge table for the many-to-many relationship between Playlists and Tracks.

Columns:

- PlaylistId (INTEGER, Foreign Key): Links to the Playlist table.
- TrackId (INTEGER, Foreign Key): Links to the Tracks table.

Relationships:

- Connects Playlists to Tracks in a many-to-many relationship.

Use Cases:

- Easily manage playlist contents.
- Count the number of tracks per playlist.

8. Customers Table

Purpose: Stores customer details and contact information.

Columns:

- CustomerId (INTEGER, Primary Key): Unique identifier for each customer.
- FirstName (TEXT): The customer's first name.
- LastName (TEXT): The customer's last name.

- Company (TEXT): The company associated with the customer (if any).
- Address, City, State, Country, PostalCode (TEXT): Contact details for the customer.
- Phone, Fax, Email (TEXT): Contact methods for the customer.
- SupportRepId (INTEGER, Foreign Key): Links to the Employees table, representing the employee handling this customer.

Relationships:

- Many-to-One relationship with Employees through SupportRepId, representing customer service assignment.
- One-to-Many relationship with Invoices, where each customer may have multiple invoices.

Use Cases:

- Track each customer's purchase history.
- Identify customer segments based on location or representative.

9. Invoice Table

Purpose: Stores purchase details, functioning as the primary sales record.

Columns:

- InvoiceId (INTEGER, Primary Key): Unique identifier for each invoice.
- CustomerId (INTEGER, Foreign Key): Links to the Customers table.
- InvoiceDate (DATETIME): Date the invoice was created.
- Billing details (Address, City, State, Country, PostalCode): Billing address information.
- Total (NUMERIC): Total amount billed.

Relationships:

- Many-to-One relationship with Customers, where each invoice is associated with a specific customer.
- One-to-Many relationship with InvoiceLine, where each invoice includes multiple invoice lines for individual items.

Use Cases:

- Calculate total sales over time.
- Analyze customer spending patterns.

10. InvoiceLine Table

Purpose: Stores individual items within each invoice, detailing each purchased track.

Columns:

- InvoiceLineId (INTEGER, Primary Key): Unique identifier for each line item.
- InvoiceId (INTEGER, Foreign Key): Links to the Invoice table.
- TrackId (INTEGER, Foreign Key): Links to the Tracks table.
- UnitPrice (NUMERIC): Price per track.
- Quantity (INTEGER): Quantity of the track purchased.

Relationships:

- Many-to-One relationship with Invoices, detailing specific items on each invoice.
- Many-to-One relationship with Tracks, where each line item represents a purchased track.

Use Cases:

- Analyze sales of individual tracks.
- Calculate total quantity sold per track.

11. Employees Table

Purpose: Stores information about employees, such as sales and support representatives.

Columns:

- EmployeeId (INTEGER, Primary Key): Unique identifier for each employee.
- FirstName (TEXT): Employee's first name.
- LastName (TEXT): Employee's last name.
- Title (TEXT): Job title of the employee.
- ReportsTo (INTEGER, Foreign Key): References another employee, establishing a hierarchy.
- BirthDate, HireDate (DATETIME): Personal information.

- Address, City, State, Country, PostalCode (TEXT): Address details.
- Phone, Fax, Email (TEXT): Contact information.

Relationships:

- Hierarchical relationship within Employees table through ReportsTo, supporting management hierarchy.
- One-to-Many relationship with Customers, where each employee is assigned multiple customers as a support representative.

Use Cases:

- View organizational structure and reporting lines.
- Identify employee performance metrics, like sales or customer support satisfaction.

Use Cases and Insights

The Chinook database supports a wide range of queries for analytics and customer insights, such as:

- Analyzing the most popular genres and tracks by customer segment.
- Identifying top-selling tracks, albums, and artists.
- Monitoring customer spending patterns by region or assigned employee.
- Evaluating employee performance based on customer service metrics.
- Building and maintaining targeted playlists and genre collections based on track popularity.