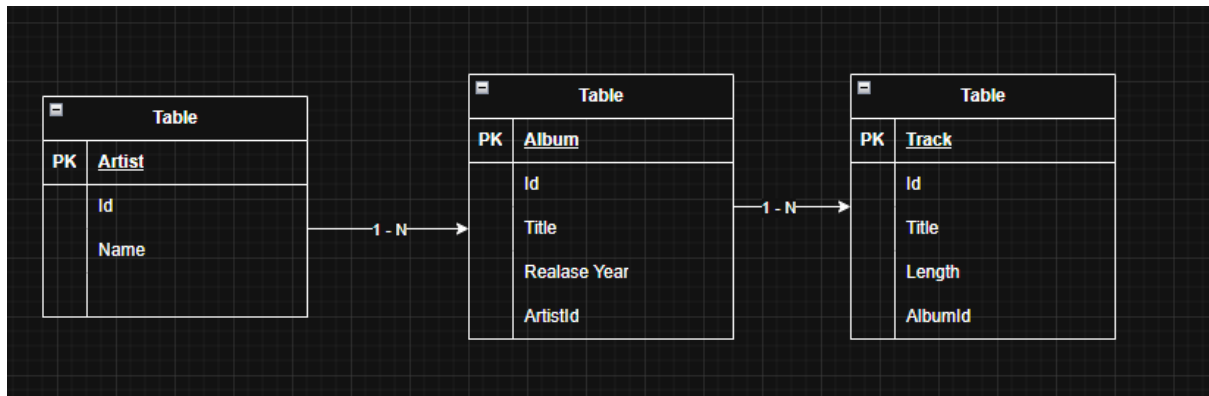


## ER-Diagram



## Database security:

Secure access to a database is important to protect information and to ensure that data is not read/modified by unauthorized users.

Authentication is used to verify the user's identity. One way is by using username and password. It also tells the database what kind of authorization the user has, and that will tell the database what kind of actions they can take, for example, change/delete or add.

In backend projects sensitive data should always be protected by using encryption at rest of transfers between systems. the access to the database should be restricted to the least amount of people you can. By using secure connections and proper config backend systems can be made more secure and robust.

## Version Control:

It's important to use database development because it allows changes in the scripts to be tracked over time. By using version control, it is easy to revert to earlier versions if an error occurs. Overall, version control improves structure and control during database development.

Milo Eriksson  
Databas  
SQL

Here is all the SQL code used:

```
INSERT INTO Artist (Name) VALUES
```

```
('mico');
```

```
CREATE DATABASE musicBase
```

```
-- Creating all the different tables
```

```
-- using IDENTITY (1,1) for auto-incremented ID, it automatically increases the ID nr.
```

```
-- using NVARCHAR(200) because it uses UNICODE so it also reads in 2,2,2. 200 is  
maximum amount of characters
```

```
-- Create the Artist table
```

```
-- This table stores information about music artists
```

```
CREATE TABLE Artist (
```

```
    Id INT IDENTITY(1,1) PRIMARY KEY, -- Primary key, auto-incremented unique ID for  
each artist
```

```
    Name NVARCHAR(200) NOT NULL    -- Artist name, must always have a value  
);
```

```
-- Create the Album table
```

```
-- Each album belongs to one artist
```

```
CREATE TABLE Album (
```

```
    Id INT IDENTITY(1,1) PRIMARY KEY, -- Primary key, auto-incremented unique ID for  
each album
```

```
    Title NVARCHAR(200) NOT NULL,    -- Album title, required
```

```
    ReleaseYear INT,                -- Year the album was released must be numbers
```

```
    ArtistId INT NOT NULL,          -- Foreign key that links the album to an artist
```

Milo Eriksson  
Databas  
SQL

FOREIGN KEY (ArtistId) REFERENCES Artist(Id) -- ArtistId must match an existing Artist.Id

);

-- Create the Track table

-- Each track belongs to one album

CREATE TABLE Track (

Id INT IDENTITY(1,1) PRIMARY KEY, -- Primary key, auto-incremented unique ID for each track

Title NVARCHAR(200) NOT NULL, -- Track title, required

Length INT NOT NULL, -- Length of the track in seconds

AlbumId INT NOT NULL, -- Foreign key that links the track to an album

FOREIGN KEY (AlbumId) REFERENCES Album(Id) -- AlbumId must match an existing Album.Id

);

-- Delete remove artist

DELETE FROM Artist

WHERE Id = 12;

INSERT INTO Artist (Name) VALUES

('Imagine Dragons'),

('Adele'),

('Coldplay'),

('Ed Sheeran'),

('Taylor Swift'),

Milo Eriksson  
Databas  
SQL

('The Weeknd'),  
('Bruno Mars'),  
('Billie Eilish'),  
('Linkin Park'),  
('Eminem');

INSERT INTO Album (Title, ReleaseYear, ArtistId) VALUES

('Night Visions', 2012, 1), -- Imagine Dragons  
('25', 2015, 2), -- Adele  
('Parachutes', 2000, 3), -- Coldplay  
('Divide', 2017, 4), -- Ed Sheeran  
('1989', 2014, 5), -- Taylor Swift  
('After Hours', 2020, 6), -- The Weeknd  
('24K Magic', 2016, 7), -- Bruno Mars  
('When We All Fall Asleep', 2019, 8), -- Billie Eilish  
('Hybrid Theory', 2000, 9), -- Linkin Park  
('The Eminem Show', 2002, 10); -- Eminem

INSERT INTO Track (Title, Length, AlbumId) VALUES

-- Night Visions  
('Radioactive', 186, 1),  
('Demons', 177, 1),  
('On Top of the World', 202, 1),  
  
-- 25  
('Hello', 295, 2),  
('Send My Love', 223, 2),

Milo Eriksson  
Databas  
SQL

('Water Under the Bridge', 240, 2),

-- Parachutes

('Yellow', 270, 3),

('Shiver', 299, 3),

('Trouble', 270, 3),

-- Divide

('Shape of You', 234, 4),

('Castle on the Hill', 261, 4),

('Perfect', 263, 4),

-- 1989

('Blank Space', 231, 5),

('Shake It Off', 242, 5),

('Style', 231, 5),

-- After Hours

('Blinding Lights', 200, 6),

('Save Your Tears', 215, 6),

('In Your Eyes', 239, 6),

-- 24K Magic

('24K Magic', 227, 7),

('Chunky', 210, 7),

('That's What I Like', 247, 7),

Milo Eriksson  
Databas  
SQL

-- When We All Fall Asleep

('Bad Guy', 194, 8),

('Bury a Friend', 193, 8),

('When the Party's Over', 199, 8),

-- Hybrid Theory

('In the End', 216, 9),

('Crawling', 203, 9),

('Papercut', 184, 9),

-- The Eminem Show

('Without Me', 290, 10),

('Cleanin Out My Closet', 297, 10),

('Sing for the Moment', 339, 10);

-- select\_basic.sql

-- using of WHERE, ORDER BY, LIKE, GROUP BY

-- 1. WHERE: Find all artists whose names start with 'E'

SELECT \*

FROM Artist

WHERE Name LIKE 'E%';

-- 2. ORDER BY: List albums sorted by release year (oldest first)

SELECT Title, ReleaseYear

FROM Album

ORDER BY ReleaseYear ASC;

Milo Eriksson  
Databas  
SQL

-- 3. LIKE: Find tracks containing the word 'Love'

```
SELECT Title
FROM Track
WHERE Title LIKE '%Love%';
```

-- 4. WHERE + ORDER BY: Albums released after 2010 sorted alphabetically

```
SELECT Title, ReleaseYear
FROM Album
WHERE ReleaseYear > 2010
ORDER BY Title;
```

-- 5. GROUP BY: Count how many albums each artist has

```
SELECT ArtistId, COUNT(*) AS AlbumCount
FROM Album
GROUP BY ArtistId;
```

-- 6. GROUP BY (sum): Calculate total track length per album

```
SELECT AlbumId, SUM(Length) AS TotalAlbumLength
FROM Track
GROUP BY AlbumId;
```

-- 7. Find all tracks where the title starts with 'B'

```
SELECT Title
FROM Track
WHERE Title LIKE 'B%';
```

-- THE QUERY shows all artist and their albums

-- it also counts the tracks in each album

Milo Eriksson  
Databas  
SQL

-- LEFT JOIN is used to show artists and albums even without tracks

-- 'ON' "Join each album to the artist where the album's ArtistId matches the artist's Id."

SELECT

Album.Title, -- The title of the album

Artist.Name, -- The name of the artist

COUNT(Track.Id) AS TrackCount -- Number of tracks per album

FROM Artist -- Start from the Artist table

LEFT JOIN Album

ON Album.ArtistId = Artist.Id -- Connect albums to their artist

LEFT JOIN Track

ON Track.AlbumId = Album.Id -- Connect tracks to their album

GROUP BY

Album.Title, -- Group results by album

Artist.Name; -- Group results by artist

-- THE QUERY shows all artist and their albums

-- it also counts the tracks in each album

-- INNER JOIN is used so only albums that have artist and tracks will show

SELECT

Album.Title, -- The title of the album

Artist.Name, -- The name of the artist

COUNT(Track.Id) AS TrackCount -- Number of tracks in the album

FROM Album -- Start from the Album table

INNER JOIN Artist

ON Album.ArtistId = Artist.Id -- Match albums to their artist (foreign key → primary key)

Milo Eriksson  
Databas  
SQL

INNER JOIN Track

ON Track.AlbumId = Album.Id -- Match tracks to their album (foreign key → primary key)

GROUP BY

Album.Title, -- Group rows by album title

Artist.Name; -- Group rows by artist name

-- UPDATES Statments that modify exsisting data

-- Change the name of an artist

-- This updates the artist with Id = 1 and sets a new name

UPDATE Artist

SET Name = 'Imagine Dragons (Updated)'

WHERE Id = 1;

-- Update the release year of an album

-- This changes the release year for a specific album

UPDATE Album

SET ReleaseYear = 2013

WHERE Title = 'Night Visions';

SQL vs LINQ

SQL	LINQ
SELECT * FROM Artist WHERE Name LIKE 'B%';	var artists = dbContext.Artists .Where(a => a.Name.StartsWith("B")) .ToList();
Mapping:	
FROM Artist	DbContext.Artist

Milo Eriksson  
Databas  
SQL

WHERE Name LIKE 'B%'	.Where(a => a.Name.StartsWith("B"))
SELECT*	Selects all columns automatically in LINQ

Sql returns rows where LINQ returns object.

SQL	LINQ
SELECT Title, ReleaseYear FROM Album ORDER BY ReleaseYear;	var albums = dbContext.Albums .OrderBy(a => a.ReleaseYear) .Select(a => new { a.Title, a.ReleaseYear }) .ToList();
mapping	
FROM Album	dbContext.Albums
ORDER BY	.OrderBy()
-	.Select(a => new { a.Title, a.ReleaseYear })

SQL	LINQ
SELECT Title FROM Track WHERE Title LIKE '%Love%';	var tracksWithLove = dbContext.Tracks .Where(t => t.Title.Contains("Love")) .Select(t => t.Title) .ToList();
FROM Track	DbContext.Tracks
WHERE	.Where()
LIKE '%Love%'	.Contains("Love")
SELECT Title	.Select(t => t.Title)

Milo Eriksson  
Databas  
SQL

Sql returns rows where LINQ returns a list of values

## REFELCTION

I have had a general struggle with the understanding. So there has been alot of asking ai, watching videos, reading material and googleing.

I feel like i now have a better understanding of how the DB works and how it translates into c#.

I dont really know how to make it better then me coding even more.

I need to use it more in diffrent context to have an even better understanding of it.

Milo Eriksson  
Databas  
SQL