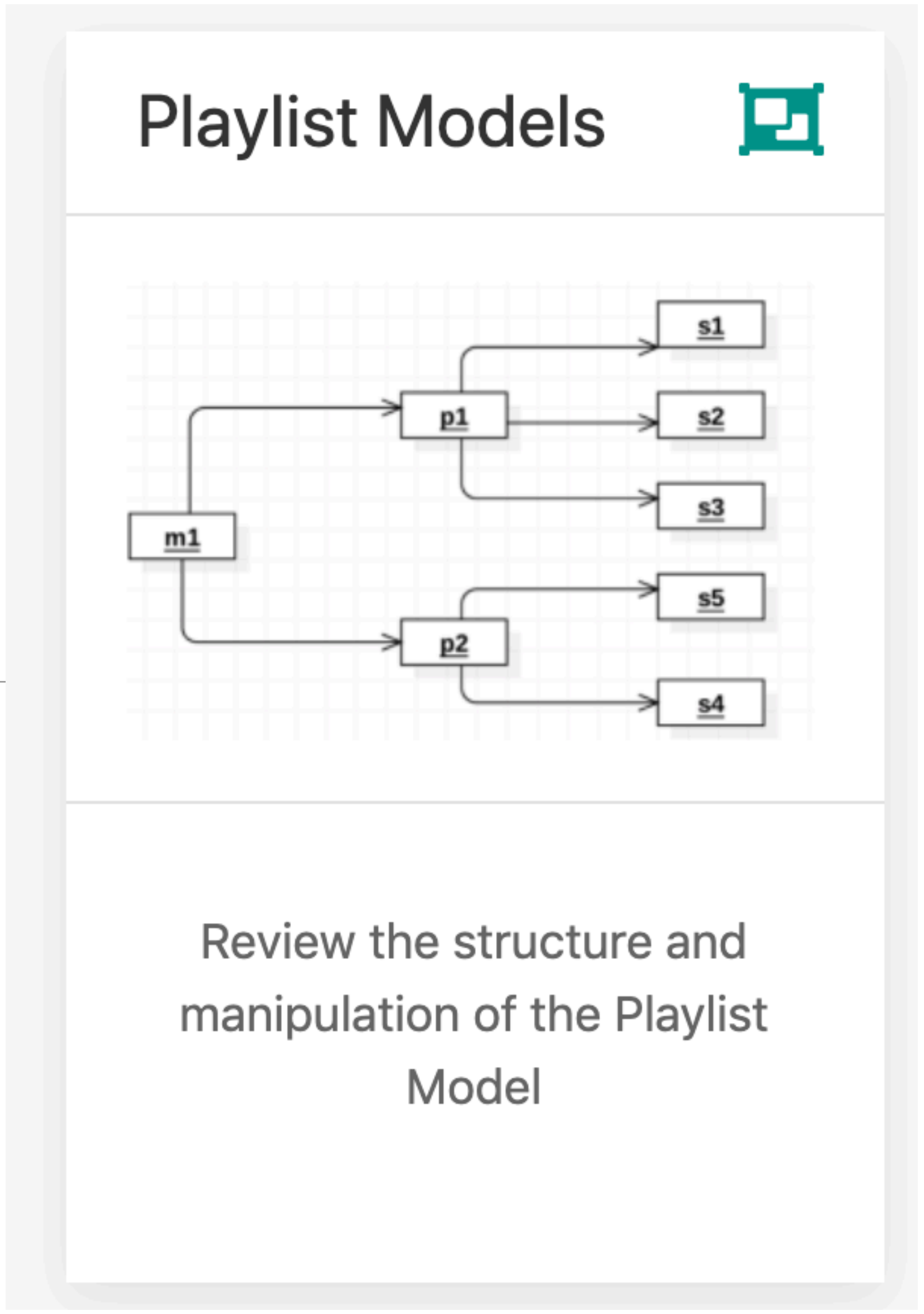
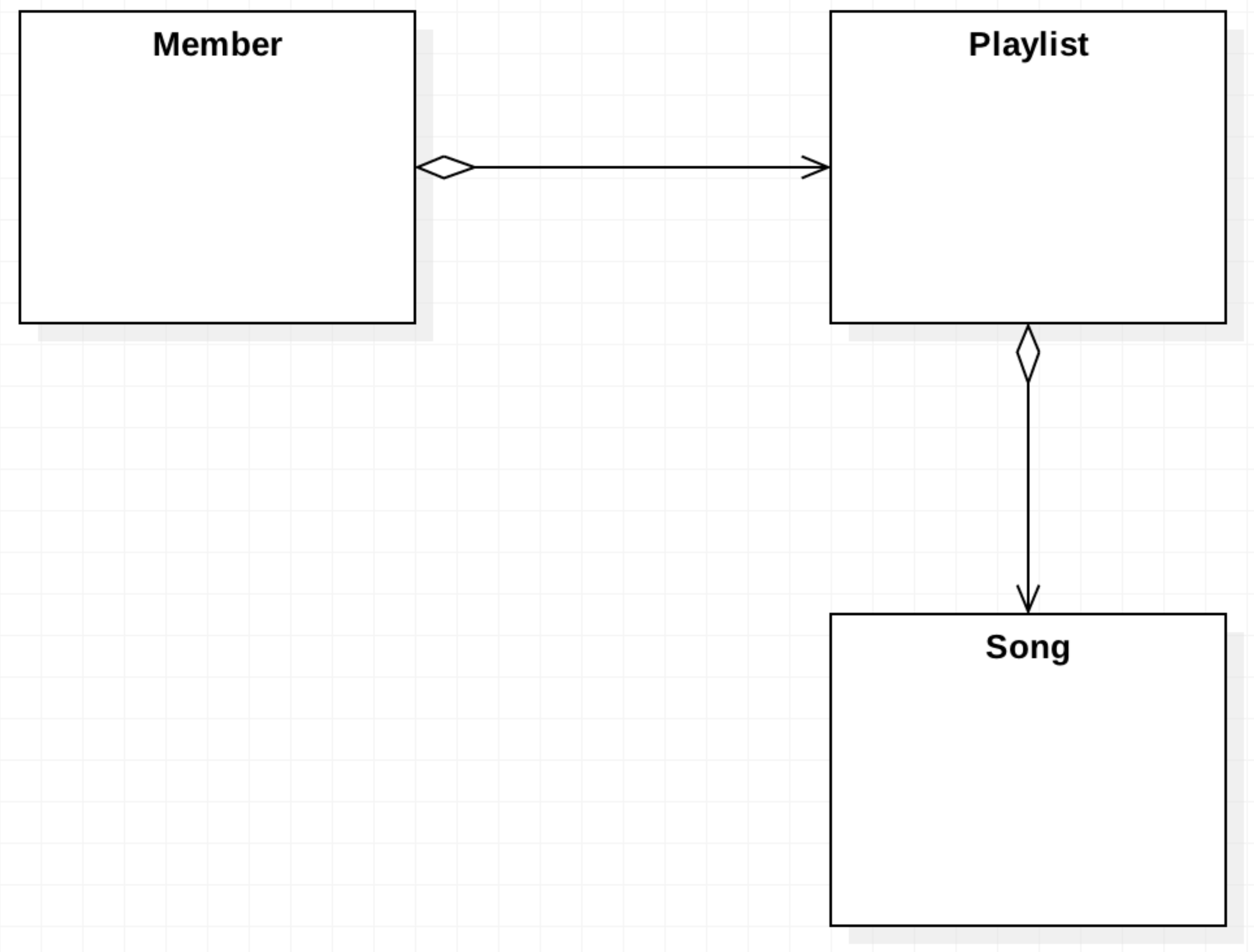
Playlist Models



Playlist Models

Review the structure and manipulation of the Playlist Model

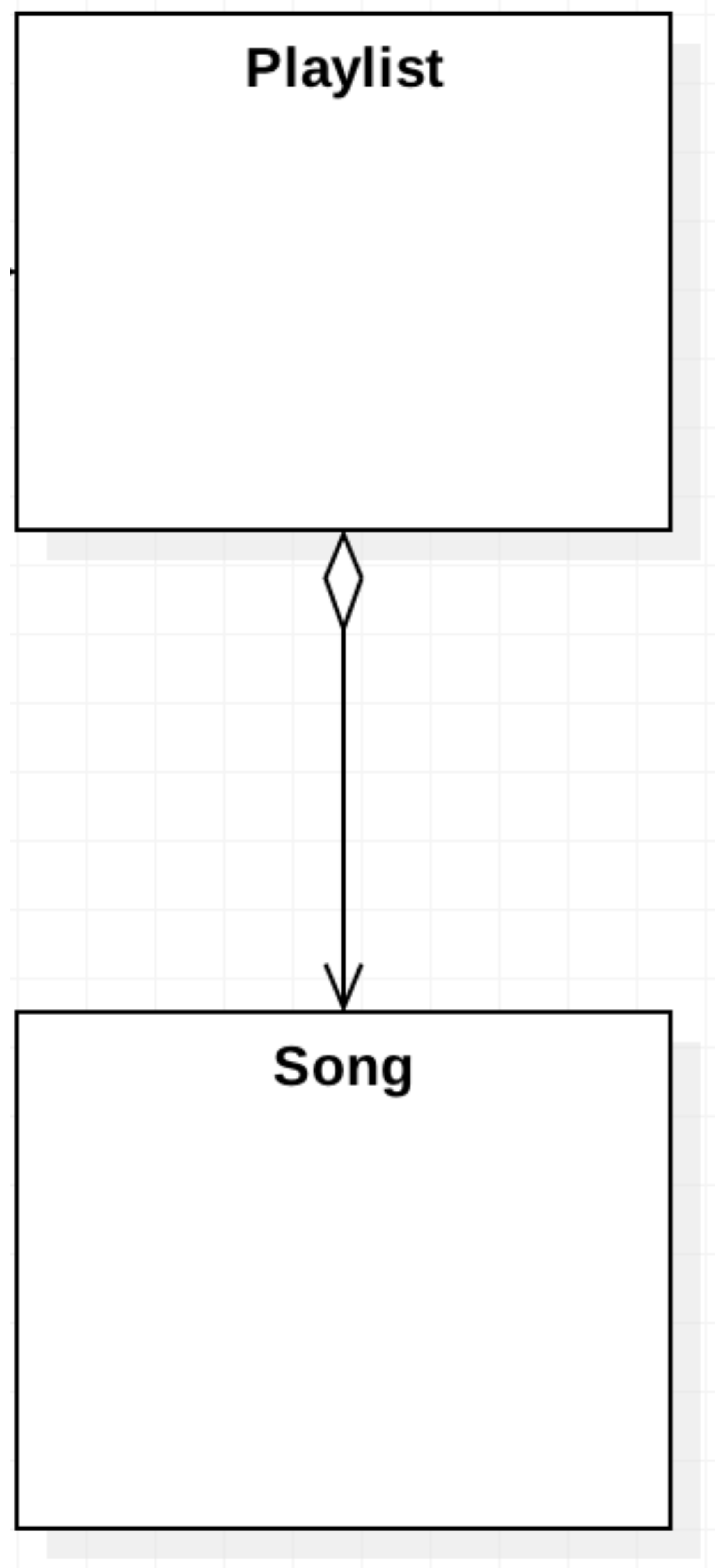
Playlist Model



Entity Classes: Playlist + Song

```java
@Entity
public class Playlist extends Model
{
  public String title;
  public int duration;

  @OneToMany(cascade = CascadeType.ALL)
  public List<Song> songs = new ArrayList<Song>();

  public Playlist(String title, int duration)
  {
    this.title = title;
    this.duration = duration;
  }
}
```

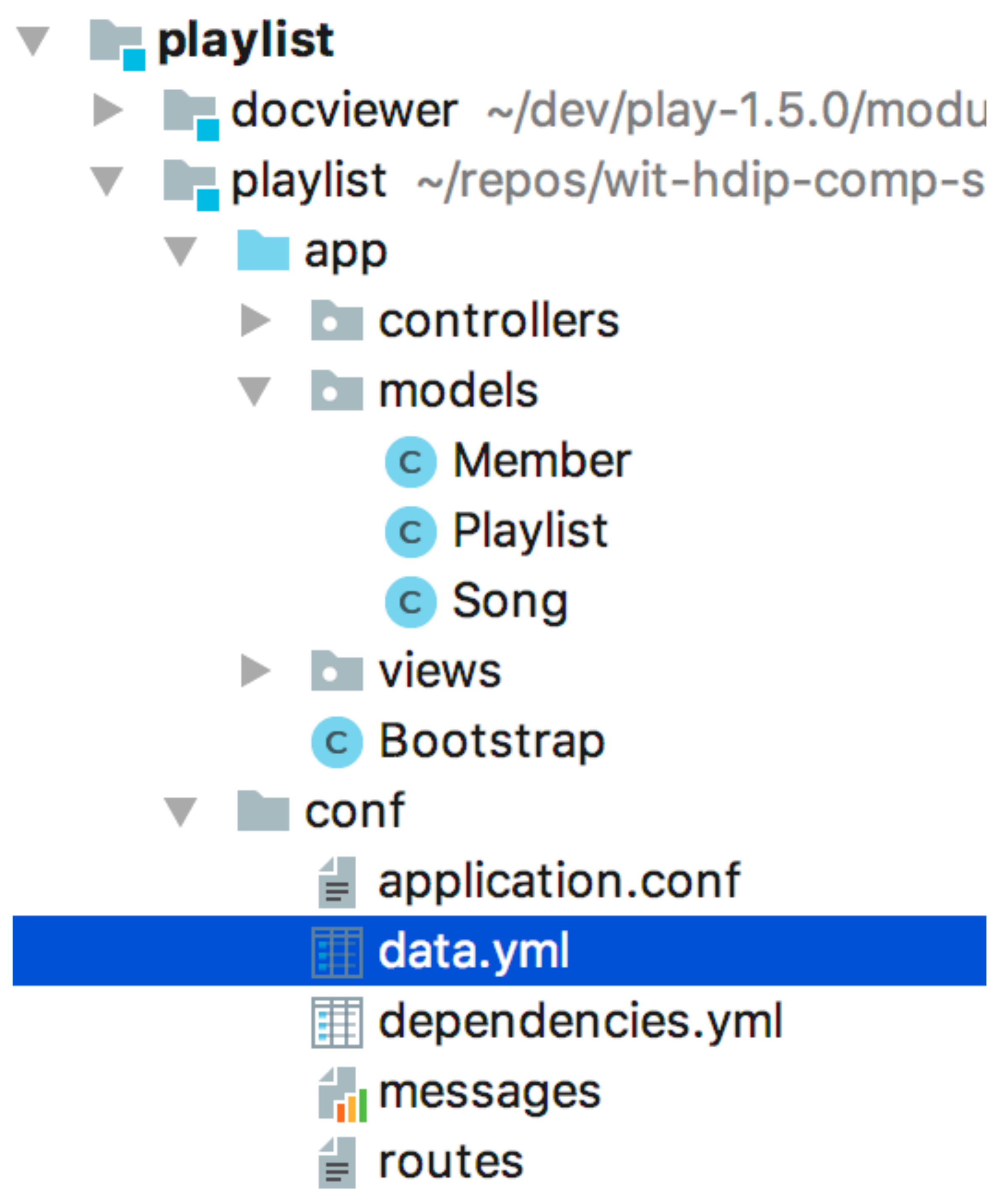
```java
@Entity
public class Song extends Model
{
  public String title;
  public String artist;
  public int duration;

  public Song(String title, String artist, int duration)
  {
    this.title = title;
    this.artist = artist;
    this.duration = duration;
  }
}
```

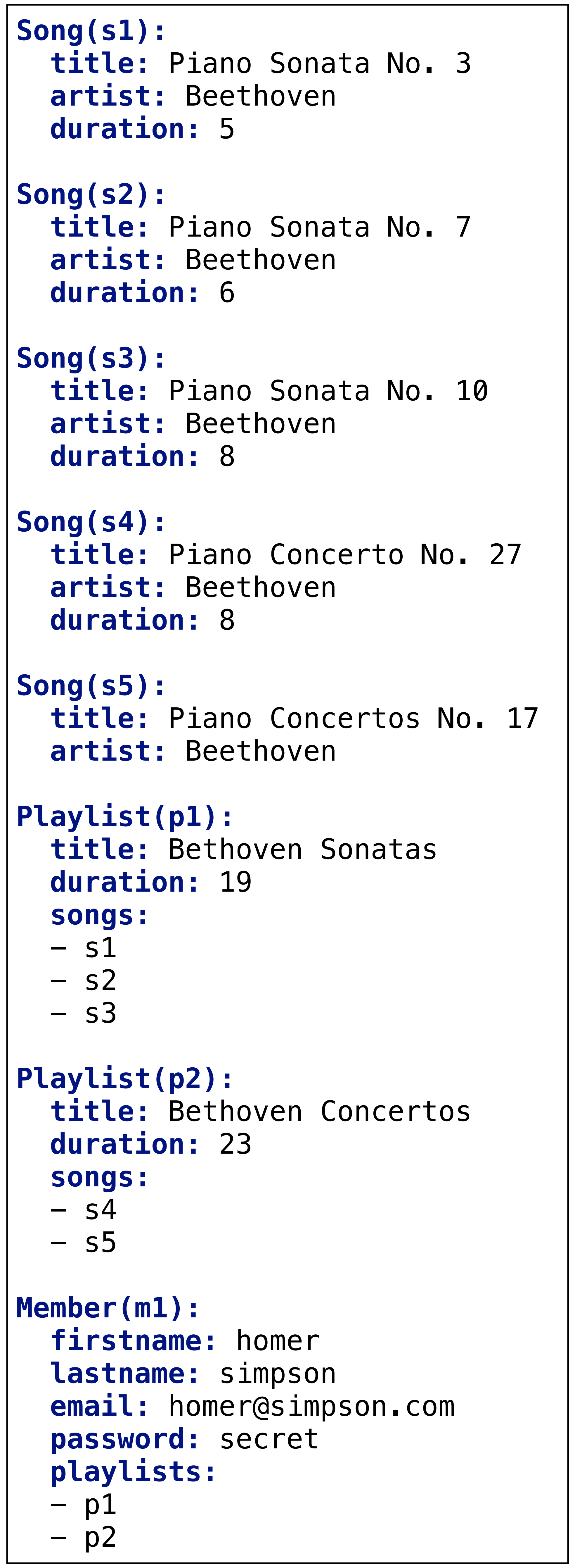


Entity Class: Member

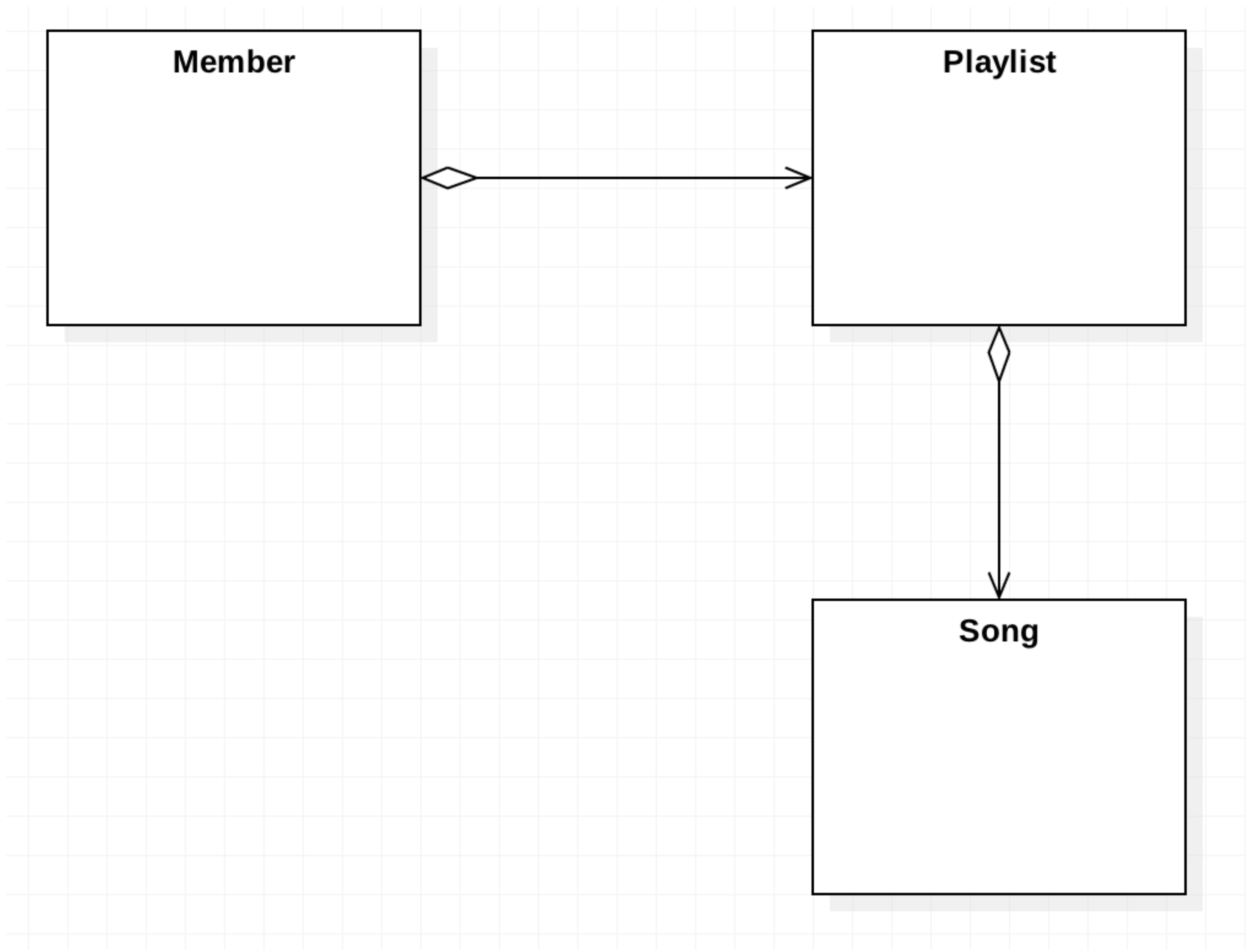
```
@Entity
public class Member extends Model
{
  public String firstname;
  public String lastname;
  public String email;
  public String password;

  @OneToMany(cascade = CascadeType.ALL)
  public List<Playlist> playlists = new ArrayList<Playlist>();

  public Member(String firstname, String lastname, String email, String password)
  {
    this.firstname = firstname;
    this.lastname = lastname;
    this.email = email;
    this.password = password;
  }

  public static Member findByEmail(String email)
  {
    return find("email", email).first();
  }

  public boolean checkPassword(String password)
  {
    return this.password.equals(password);
  }
}
```

data.yml



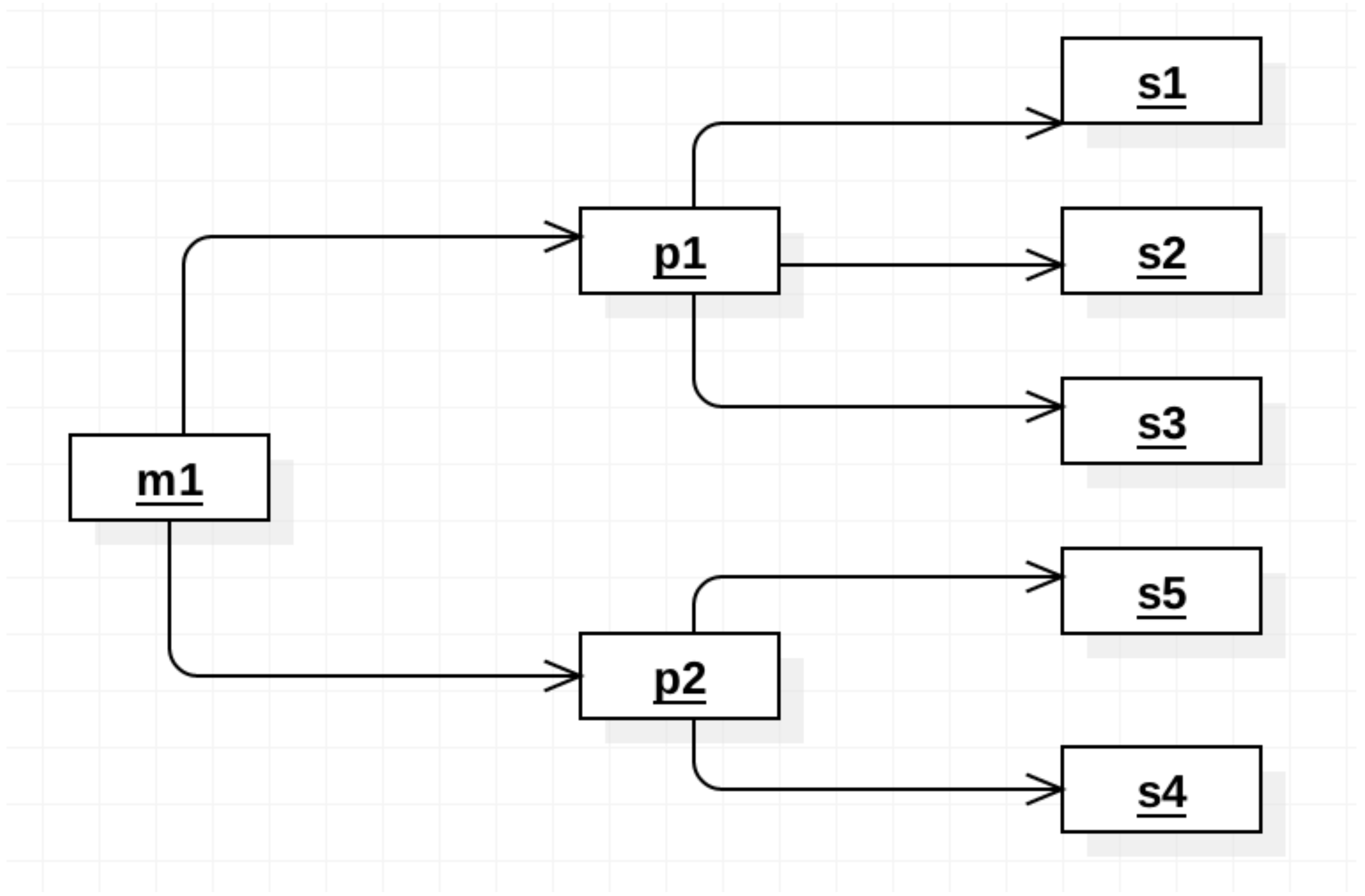


```
Song(s1):
  title: Piano Sonata No. 3
  artist: Beethoven
  duration: 5

Song(s2):
  title: Piano Sonata No. 7
  artist: Beethoven
  duration: 6

Song(s3):
  title: Piano Sonata No. 10
  artist: Beethoven
  duration: 8

Song(s4):
  title: Piano Concerto No. 27
  artist: Beethoven
  duration: 8

Song(s5):
  title: Piano Concertos No. 17
  artist: Beethoven

Playlist(p1):
  title: Bethoven Sonatas
  duration: 19
  songs:
  - s1
  - s2
  - s3

Playlist(p2):
  title: Bethoven Concertos
  duration: 23
  songs:
  - s4
  - s5

Member(m1):
  firstname: homer
  lastname: simpson
  email: homer@simpson.com
  password: secret
  playlists:
  - p1
  - p2
```

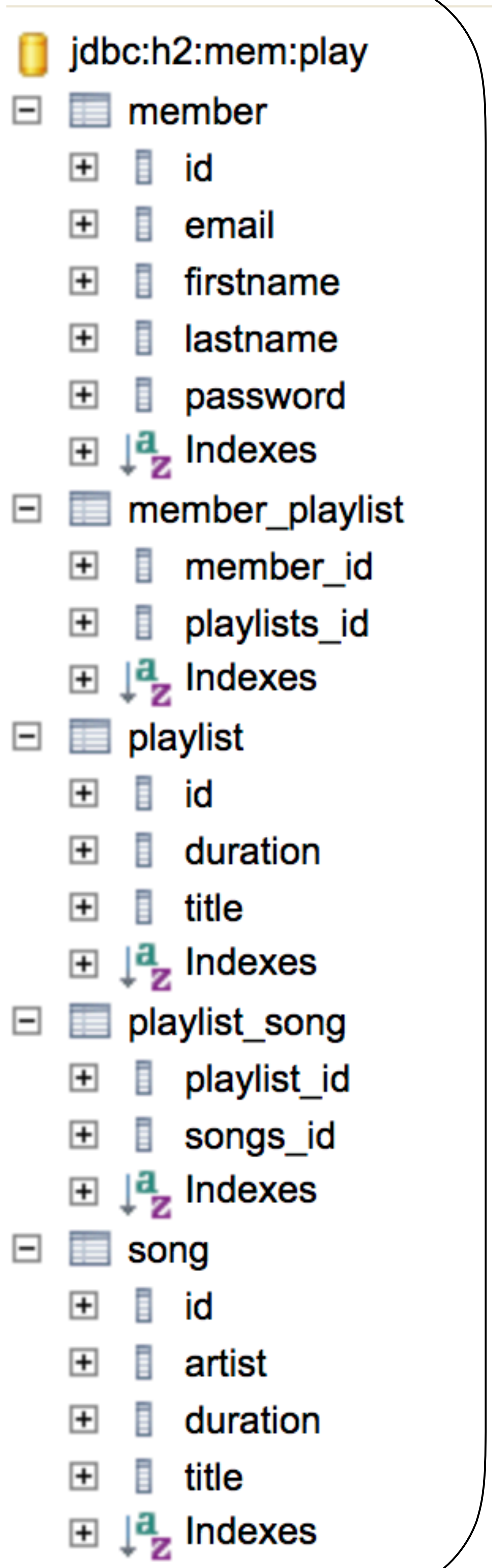


```
Song(s1):
  title: Piano Sonata No. 3
  artist: Beethoven
  duration: 5

Song(s2):
  title: Piano Sonata No. 7
  artist: Beethoven
  duration: 6

Song(s3):
  title: Piano Sonata No. 10
  artist: Beethoven
  duration: 8

Song(s4):
  title: Piano Concerto No. 27
  artist: Beethoven
  duration: 8

Song(s5):
  title: Piano Concertos No. 17
  artist: Beethoven

Playlist(p1):
  title: Bethoven Sonatas
  duration: 19
  songs:
  - s1
  - s2
  - s3

Playlist(p2):
  title: Bethoven Concertos
  duration: 23
  songs:
  - s4
  - s5

Member(m1):
  firstname: homer
  lastname: simpson
  email: homer@simpson.com
  password: secret
  playlists:
  - p1
  - p2
```

Logical View

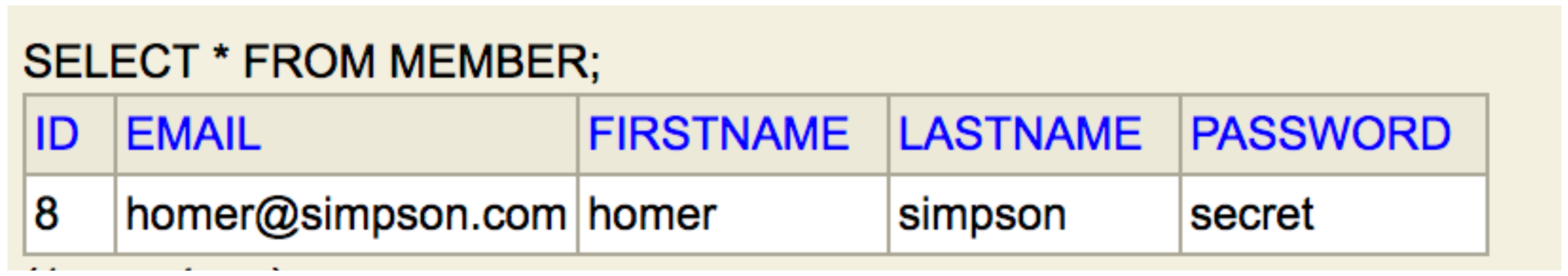


Physical View http://localhost:9000/@db



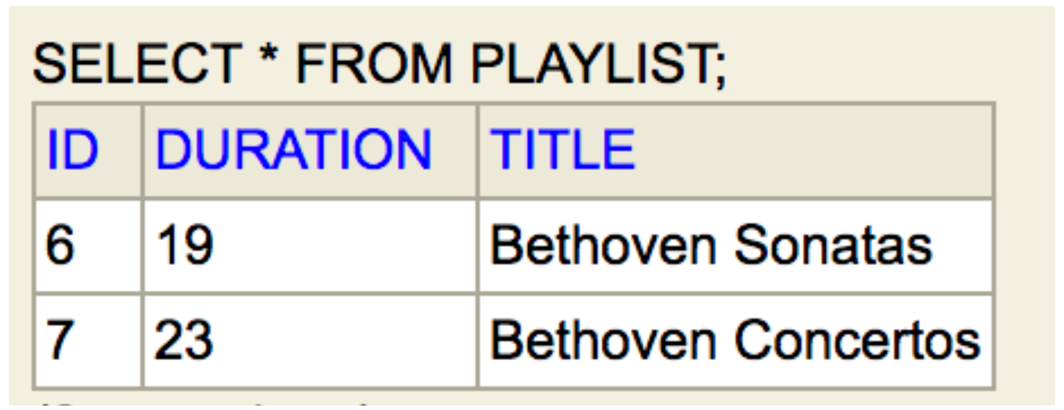
- jdbc:h2:mem:play
  - member
    - id
    - email
    - firstname
    - lastname
    - password
    - Indexes
  - member_playlist
    - member_id
    - playlists_id
    - Indexes
  - playlist
    - id
    - duration
    - title
    - Indexes
  - playlist_song
    - playlist_id
    - songs_id
    - Indexes
  - song
    - id
    - artist
    - duration
    - title
    - Indexes

Core Model Objects



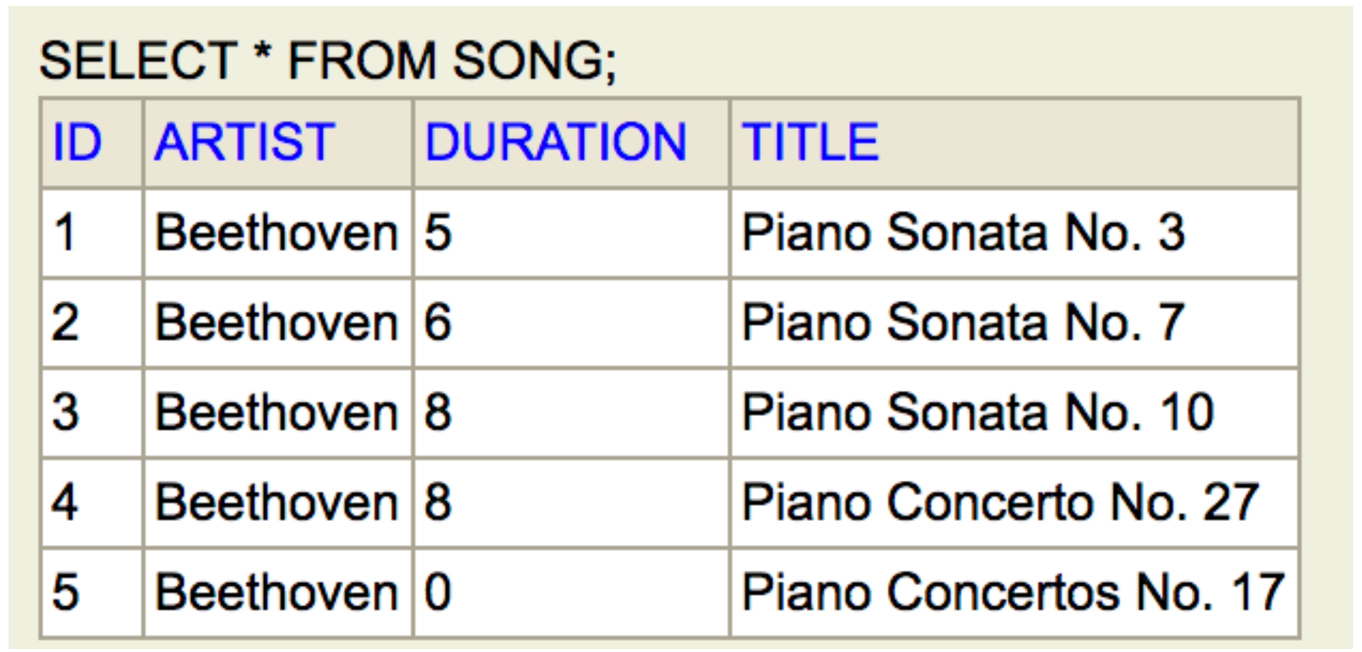
SELECT * FROM MEMBER;

ID	EMAIL	FIRSTNAME	LASTNAME	PASSWORD
8	homer@simpson.com	homer	simpson	secret



SELECT * FROM PLAYLIST;

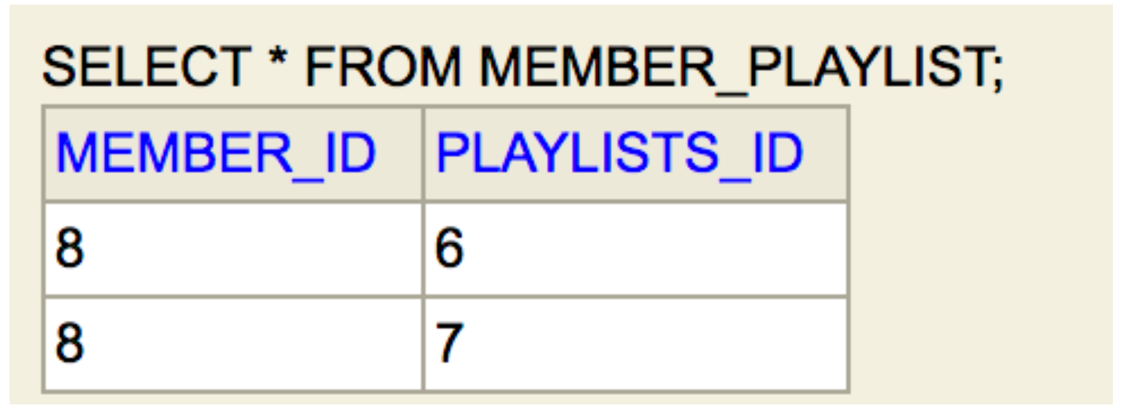
ID	DURATION	TITLE
6	19	Bethoven Sonatas
7	23	Bethoven Concertos



SELECT * FROM SONG;

ID	ARTIST	DURATION	TITLE
1	Beethoven	5	Piano Sonata No. 3
2	Beethoven	6	Piano Sonata No. 7
3	Beethoven	8	Piano Sonata No. 10
4	Beethoven	8	Piano Concerto No. 27
5	Beethoven	0	Piano Concertos No. 17

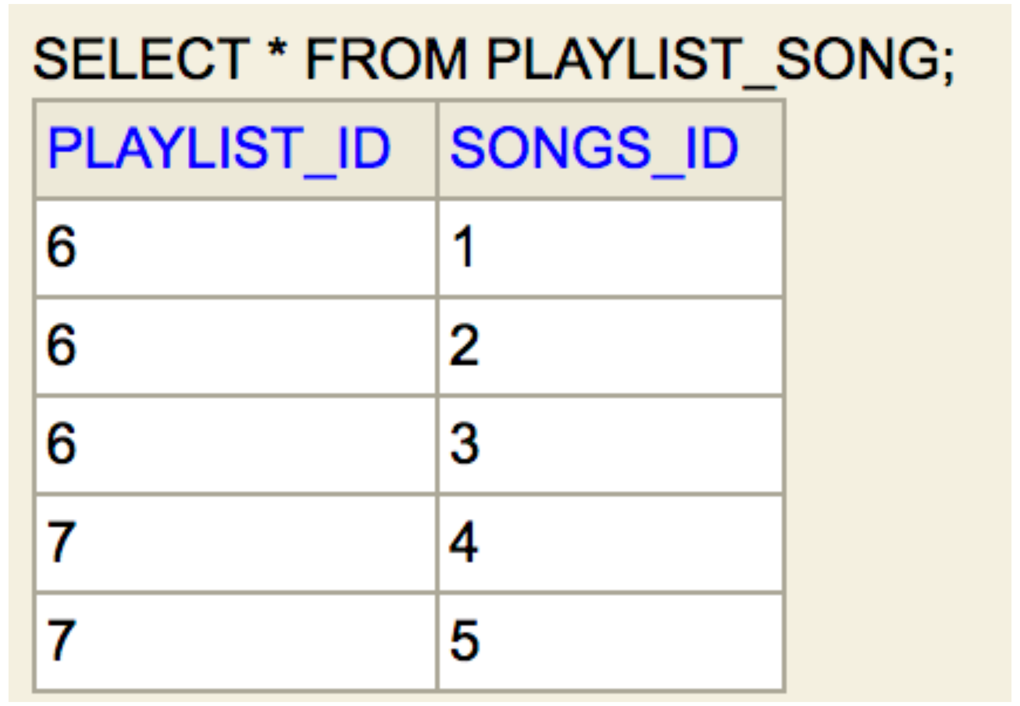
Member -> Playlist mapping table



SELECT * FROM MEMBER_PLAYLIST;

MEMBER_ID	PLAYLISTS_ID
8	6
8	7

Playlist -> Song mapping table



SELECT * FROM PLAYLIST_SONG;

PLAYLIST_ID	SONGS_ID
6	1
6	2
6	3
7	4
7	5

Manipulating Playlist

```
public class Dashboard extends Controller
{
  public static void index()
  {
    Member member = Accounts.getLoggedInMember();
    List<Playlist> playlists = member.playlists;
    render ("dashboard.html", playlists);
  }

  public static void addPlaylist (String title)
  {
    Member member = Accounts.getLoggedInMember();
    Playlist playlist = new Playlist (title, 0);
    member.playlists.add(playlist);
    member.save();
    redirect ("/dashboard");
  }

  public static void deletePlaylist (Long id)
  {
    Member member = Accounts.getLoggedInMember();
    Playlist playlist = Playlist.findById(id);
    member.playlists.remove(playlist);
    member.save();
    playlist.delete();
    redirect ("/dashboard");
  }
}
```

Display logged in members playlists

Add a new playlist

Delete a playlist

Manipulating Songs

```java
public class PlaylistCtrl extends Controller
{
  public static void index(Long id)
  {
    Playlist playlist = Playlist.findById(id);
    render("playlist.html", playlist);
  }

  public static void addSong(Long id,    String title,
                             String artist, int duration)
  {
    Song song = new Song(title, artist, duration);
    Playlist playlist = Playlist.findById(id);
    playlist.songs.add(song);
    playlist.save();
    redirect ("/playlists/" + id);
  }

  public static void deletesong (Long id, Long songid)
  {
    Playlist playlist = Playlist.findById(id);
    Song song = Song.findById(songid);
    playlist.songs.remove(song);
    playlist.save();
    song.delete();
    render("playlist.html", playlist);
  }
}
```

Display a playlist (given id)

add a song to a playlist

delete a song from a playlist