DBMS Project Report

PES University

Database Management Systems

UE18CS252

Submitted By

<<PES2201800486>> <<ADITYA ABHISHEK>>

Radio Station Management System

The project titled Radio Station Management System is an online music streaming platform. It is a web application for providing services such as music streaming, playlists, discovering new songs, search songs and inspirational quotes .

The Radio Station Management System is a web application designed for all kinds of operating systems capable of running a web browser. This software is easy to use for all kinds of people with little or no knowledge of computer operations. It features a familiar and well thought-out, an attractive user interface, combined with strong searching, insertion and reporting capabilities.

Introduction 2

Data Model 3

FD and Normalization 5

DDL 7

Triggers 10

SQL Queries 11

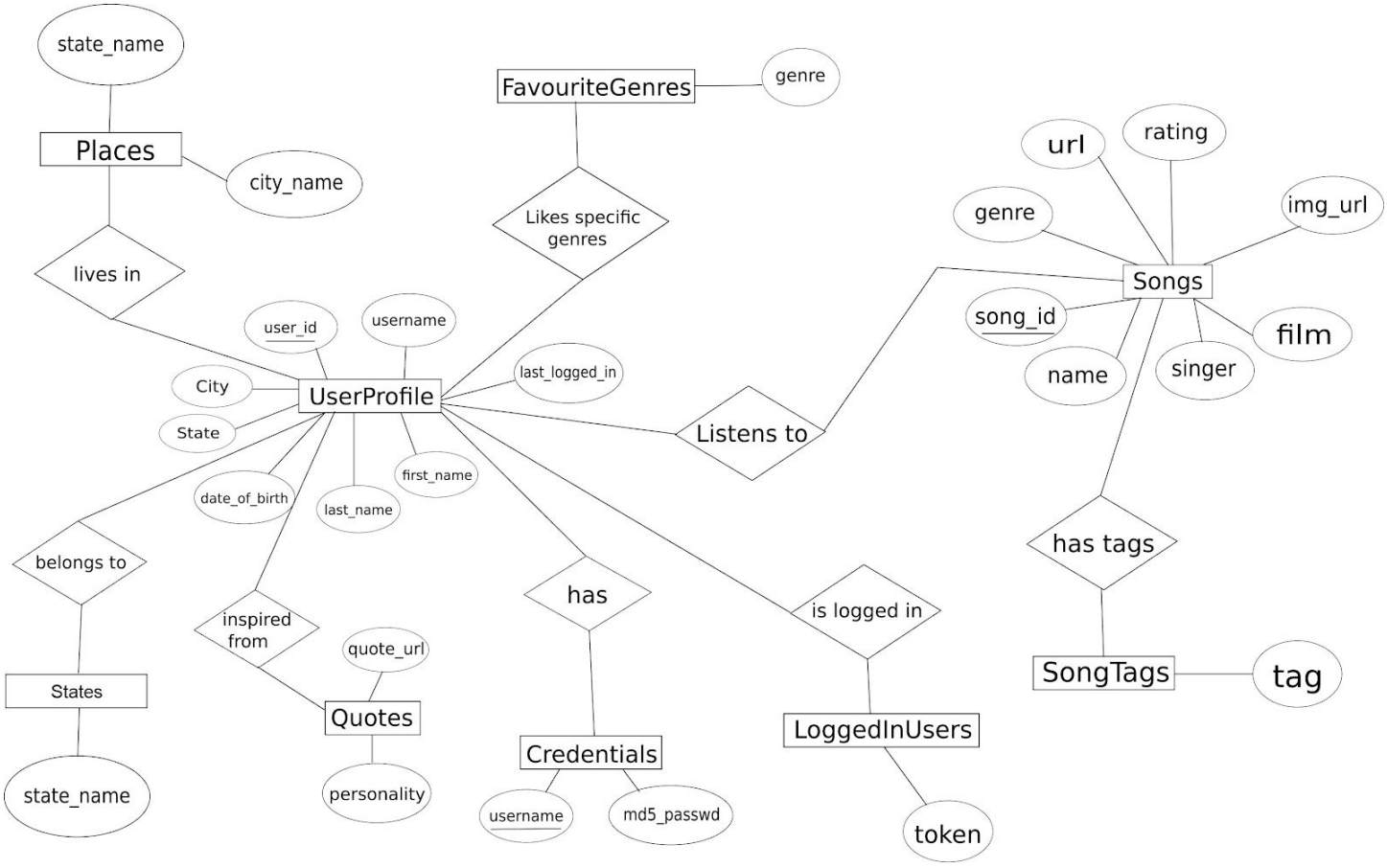
Conclusion 13

Introduction

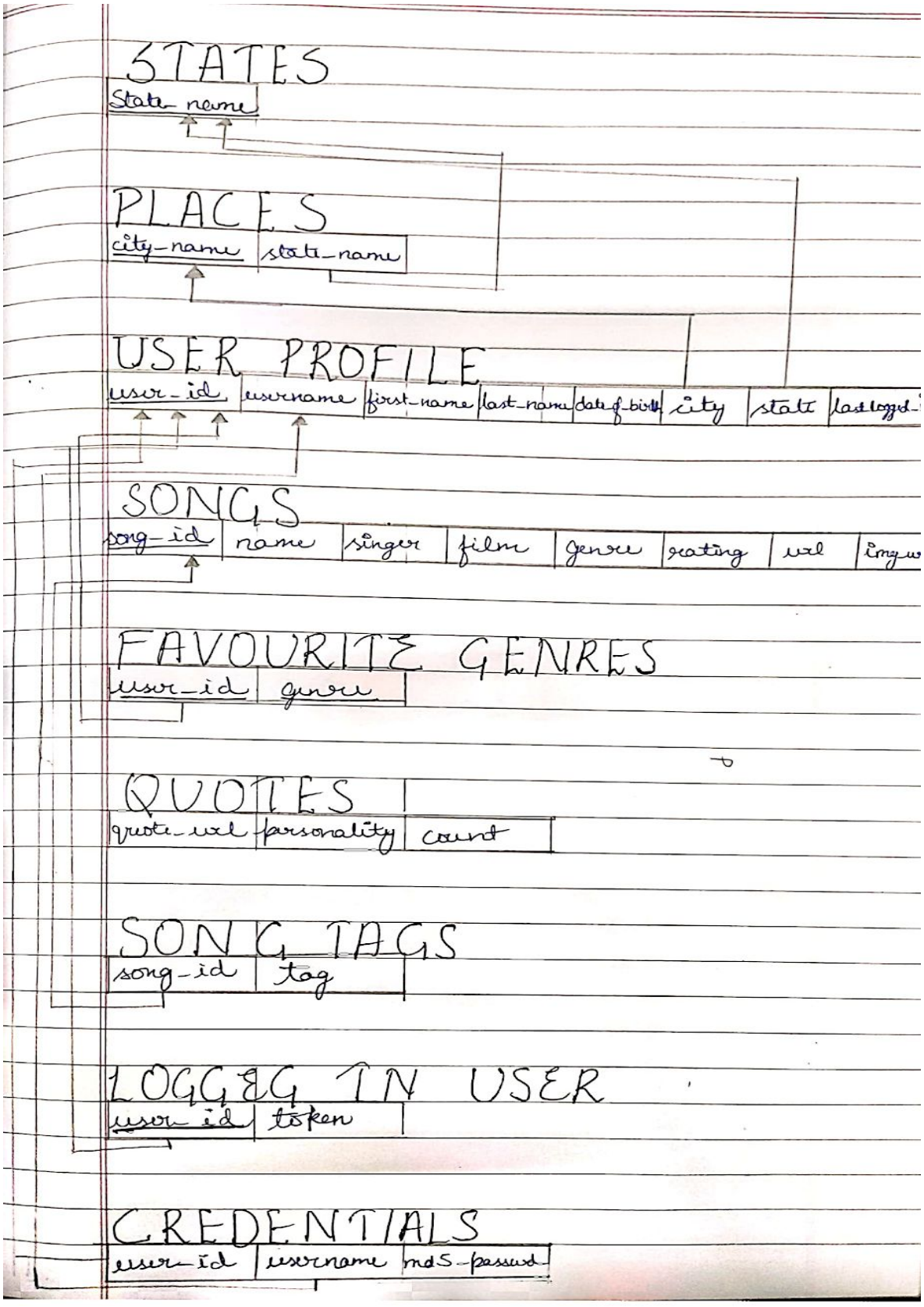
- The user can login using **UserProfile** and add details to username, user_id, city, State, date_of_birth, last_name, first_name and last_logged_in.
- The user then gives data for **States** in which he/she specifies from where he/she belongs to by giving state_name.
- The user then gives data for **Places** where he/she lives in by giving state_name and city_name.
- The user also has **Credentials** in the form of username and md5_passwd.
- The user gives his/her **FavouriteGenres** by filling in the genre.
- The user listens to **Songs** which has img_url, rating, url, genre, song_id, name, singer and film.
- The songs have tags in the form of **SongTags** tag.
- The user gets inspired from **Quotes** which has a quote_url and the personality it is coming from.
- And finally the user is logged in the **LoggedInUser** and gets a token.

Data Model

E.R. Diagram



Relational Schema



Keys

Primary Keys

- state_name in table **States**
- city_name in table **Places**
- user_id in table **UserProfile**
- song_id in table **Songs**
- User_id in table **LoggedInUsers**

Foreign Keys

- state_name in table **Places** with references to state_name in **States**.
- city in table **UserProfile** with references to city_name in **Places**.
- state in table **UserProfile** with references to state_name in **States**.
- username in **Credentials** with references to username in **UserProfile**.
- user_id in **Credentials** with references to user_id in **UserProfile**.
- user_id in **FavouriteGenres** with references to user_id in **UserProfile**.
- song_id in **SongTags** with references to song_id in **Songs**.
- user_id in **LoggedInUsers** with references to user_id in **UserProfile**.

Data Types

For most of the text attributes we have used varchar data type.For number attributes we have used integer data type and numeric data type and finally we have used date data type for dates.

FD and Normalization

FUNCTIONAL DEPENDENCIES

PLACES
(city_name) → state_name

USER PROFILE
(user_id) → username, first_name, last_name, date_of_birth, city, state, last_logged_in

SONGS
(song_id) → name, singer, film, genre, rating, url, img_url

FAVOURITE GENRES
(user_id) → genres

QUOTES
(quote_url) → personality, count

SONG TAGS
(song_id) → tag

LOGGED IN USER
(user_id) → token

CREDENTIALS
(user_id) → username, md5_passwd

NORMALISATION

1NF: Remove all multivalued attributes and nested relations
Since there are no multivalued and composite attributes as we can see in our ER diagram. Hence all the values are atomic and in first normal form.

1NF can get violated if we didn't convert the attribute ‘type’ into its own relation [non atomic]

2NF: Ensure full functional dependency
As we can see in the above functional dependencies, none of the relations have partial functional dependency. Hence all the given relations are in second normal form.
Let us discuss a case when 2NF can be violated.
Assume if we add user_id attribute to relation SongTag, then this attribute will not depend on primary key song_id hence it will violate 2NF.

3NF: Eliminate transitive dependency
As we can see there is no transitive dependency, all of our relation is in third normal form.

BCNF: Every dependency in a relation is either trivial or is dependent on a superkey

From the schema diagram, we can see the above rules for BCNF are abided in every relation. Thus, our relations are in BCNF.

DDL

```
-- Table structure for table States
--

CREATE TABLE States(
   state_name varchar(100) PRIMARY KEY
   );

-- Table structure for table Places
--

CREATE TABLE Places(
   city_name  varchar(100) PRIMARY KEY,
   state_name varchar(100) NOT NULL,
   FOREIGN KEY(state_name) REFERENCES States(state_name)
         ON DELETE CASCADE  ON UPDATE CASCADE
   );


-- Table structure for table UserProfile
--

CREATE TABLE UserProfile(
   user_id int PRIMARY KEY,
   username varchar(100) NOT NULL UNIQUE,
   first_name varchar(100) NOT NULL,
   last_name varchar(100),
   date_of_birth date NOT NULL,
   city varchar(100),
   state varchar(100),
   last_logged_in NUMERIC,
   FOREIGN KEY(city) REFERENCES Places(city_name)
         ON DELETE CASCADE  ON UPDATE CASCADE,
   FOREIGN KEY(state) REFERENCES States(state_name)
   );
```

```
-- Table structure for table Credentials
--

CREATE TABLE Credentials(
    user_id int,
    username varchar(100),
    md5_passwd varchar(100) NOT NULL,
    FOREIGN KEY(username) REFERENCES UserProfile(username)
          ON DELETE CASCADE  ON UPDATE CASCADE,
    FOREIGN KEY(user_id) REFERENCES UserProfile(user_id)
    );

-- Table structure for table Songs
--

CREATE TABLE Songs(
    song_id int PRIMARY KEY,
    name varchar(100) NOT NULL,
    singer varchar(100),
    film varchar(100),
    genre varchar(100),
    rating int,
    url varchar(100) NOT NULL UNIQUE,
    img_url varchar(100) default '/images/defaultsong.jpg'
    );


-- Table structure for table FavouriteGenres
--

CREATE TABLE FavouriteGenres(
    user_id int,
    genre varchar(100) NOT NULL,
    FOREIGN KEY(user_id) REFERENCES UserProfile(user_id),
    PRIMARY KEY(user_id,genre)
);

-- Table structure for table Quotes
--

CREATE TABLE Quotes(
    quote_url varchar(100) PRIMARY KEY,
    personality varchar(100),
    count int
```

```
);

-- Table structure for table SongTags
--

CREATE TABLE SongTags(
    song_id int,
    tag varchar(100),
    foreign key(song_id) REFERENCES Songs(song_id)
          ON DELETE CASCADE  ON UPDATE CASCADE
    );

-- Table structure for table LoggedInUsers
--

CREATE TABLE LoggedInUsers(
    user_id int PRIMARY KEY,
    token varchar(100) NOT NULL UNIQUE,
    FOREIGN KEY(user_id) REFERENCES UserProfile(user_id)
    );
```

Triggers

We can apply a trigger in such a way that whenever a user logs into the website, last_logged_in for the user gets affected.

```
CREATE TRIGGER trgAfterUpdate ON LoggedInUsers
After Insert
AS
declare @user_id int;
select @user_id=i.user_id from inserted i;
UPDATE UserProfile
SET last_logged_in=getdate()
WHERE user_id=@user_id;
GO
```

th	city	state	last_logged_in
4	Ranchi	Jharkhand	2020-05-27
4	Silchar	Assam	2020-05-27
2	Bhopal	Madhya Pradesh	2020-05-26
1	jammu	Jummu & Kashmir	2020-05-27
8	Lucknow	Uttar Pradesh	2020-05-26
5	Kanpur	Uttar Pradesh	2020-05-27
0	Lucknow	Uttar Pradesh	2020-05-26
4	Silchar	Assam	2020-05-28
9	hydrabad	Andra Pradesh	2020-05-26
4	Itanagar	Arunachal Pradesh	2020-05-26
9	Kochi	kerala	2020-05-28
4	surat	Gujrat	2020-05-27
6	kolkata	West Bengal	2020-05-26
5	Mumbai	Maharashtra	2020-05-28
9	Ranchi	Jharkhand	2020-05-28
9	patna	Bihar	2020-05-28

Here, we can see that the last_logged_in is changed to the current date.

SQL Queries

- List all the songs which belong to the favourite genre of first name “aditya” .

```
SELECT name
FROM Songs
WHERE genre IN (SELECT f.genre
        FROM FavouriteGenres AS f, UserProfile AS u
        WHERE u.first_name='aditya' AND u.user_id=f.user_id);
```

	name
1	Aye mere watan ke logo
2	Ao naujavan milake desh ko
3	Jogada siri belakinalli
4	Sare jahan se achha (bhai bahen)
5	Main gaoon tum so jao
6	Is mod pe jate hain
7	Muraliya baje re jamuna ke teer
8	Prabhuji tum chandan
9	Mere to girdhar gopal
10	Sarveshwari jagdishwari
11	Laaj rakho girdhari
12	Om anandamayi chaitanyamayi
13	Jai radha madhav
14	Laun kahan se chand
15	Rang de chunaryan
16	Jinke hrurde hari naam base
17	Surdasji ka ek tara
18	Aisi laalagan
19	Jag mein hai sundar do naam

- List the personality whose quotes are the most liked.

```
SELECT personality
FROM Quotes
WHERE count=(SELECT MAX(count)
            FROM Quotes);
```

	personality
1	Sonu Nigam

- List all the genres along with the number of users who like them.

```
SELECT genre, count(*)
FROM FavouriteGenres
GROUP BY genre ;
```

	genre	(No column name)
1	classical	1
2	country	1
3	dubstep	1
4	electronic	1
5	indie rock	1
6	jazz	1
7	love	1
8	opera	1
9	pop	3
10	rock	2
11	salsa	2
12	techno	1

- List out the songs with their name, id, genre, rating, url, img_url,tag whose tag is present and the song_id is between 1-100.

```
SELECT t.song_id, name, genre, rating, url, img_url,tag
FROM Songs AS s JOIN SongTags AS t ON s.song_id=t.song_id
WHERE t.song_id BETWEEN 1 AND 100;
```

	song_id	name	genre	rating	url	img_url	tag
1	1	Aye mere watan ke logo	pop	9	media/PATRIOTIC_SONGS/aye_mere_watan_ke_logo.mp3	/images/defaultsong.jpg	tag1
2	2	Dil diya hai jaan bhi	classical	10	media/PATRIOTIC_SONGS/dil diya hai jaan bhi (www.songs.pk...	/images/defaultsong.jpg	tag2
3	3	Ao naujavan milake desh ko	salsa	8	media/PATRIOTIC_SONGS/Ao_Naujavan_Milake_Desh_Ko.m...	/images/defaultsong.jpg	tag3
4	5	Aao bachho tumhe dekhaye	rock	9	media/PATRIOTIC_SONGS/Aao-Bachho-Tumhe-Dekha-Ye.mp3	/images/defaultsong.jpg	tag4
5	8	Mere desh	classical	3	media/PATRIOTIC_SONGS/Mere_Desh_-_(PunjabiMob.Com)....	/images/defaultsong.jpg	tag5
6	10	Mere desh ki dhar(upkar)	country	10	media/PATRIOTIC_SONGS/Independence Special - CD1 - 05 -...	/images/defaultsong.jpg	tag6
7	14	Honge kaamyaab hum honge kaamyaab	classical	8	media/PATRIOTIC_SONGS/Honge_kaamyaab_hum_honge_k...	/images/defaultsong.jpg	tag7
8	18	Kabuliwala aye mere pyare watankabuliwala	NULL	10	media/PATRIOTIC_SONGS/kabuliwala_aye_mere_pyare_wata...	/images/defaultsong.jpg	tag8
9	24	Rangdebasanti	dubstep	7	media/PATRIOTIC_SONGS/Rang-De-Basanti.mp3	/images/defaultsong.jpg	tag9
10	45	Is mod pe jate hain	pop	8	media/KISHOR_LATA/Is mod pe jate hain.mp3	/images/defaultsong.jpg	tag10
11	69	Laun kahan se chand	pop	4	media/ANUP_JALOTA/07 - Laun Kahan Se Chand.mp3	/images/defaultsong.jpg	tag11
12	73	Aisi laalagan	pop	6	media/ANUP_JALOTA/02_AISI LAAGI LAGAN.mp3	/images/defaultsong.jpg	tag12

- Find all the inactive users and print when they were last logged in

```
SELECT last_logged_in, username
FROM UserProfile
WHERE user_id IN ( (SELECT user_id
                    FROM UserProfile)
                    EXCEPT
                    (SELECT user_id
                        FROM LoggedInUsers));
```

	last_logged_in	username
1	2020-05-27	aditya31
2	2020-05-27	pranav21
3	2020-05-26	simran6
4	2020-05-26	suraj69
5	2020-05-26	coolbird
6	2020-05-27	deepu
7	2020-05-26	coolshu

Conclusion

As we have seen throughout this Report, very valuable statistics can be obtained by studying and querying the database in hand. With this database design, a Radio Station Service company can write similar queries periodically, and study the trends of the content they serve, the efficiency they provide with streaming,
From this project the user can either listen to songs based on his/her choice.
The users can update their profile and mention their favourite genres.
Listening and streaming songs are free but to download in high quality format the user has to pay.
Users can register for our website free of any charge. Users can enrich their mind and get inspired by the thoughts of great personalities.
Register with minimum details and then update profile
Many fields are optional which can be updated in their profile.
User mode is the default mode provided to the user. Users can listen to or buy songs in this mode. The user can select developer mode if required. The queries executed at the backend will be shown. The user will have an option to create a personal playlist of songs or select from the recently played.