1 DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

2 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

2.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attribute of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on another non-key attribute.

TABLE DESIGN

Table No : 01

Table Name : registration

Primary Key : uid

Foreign Key :

Table Description : To store user registration information

Fieldname	Data Type	Size	Description
uid	int	30	Primary key of the table
fname	varchar	30	First name of the user
uname	varchar	30	Email id of the user
email	varchar	30	Email id of the user
phno	bigint	100	Phone number of the user
pass	varchar	200	Password to login
status	varchar	20	Status of approval
isAdmin	int	5	Admin or not

Table No : 02

Table Name : tbl_booking

Primary Key : book_id

Foreign Key : screen_id

Table Description : To store booking information

Fieldname	Data Type	Size	Description
book_id	int	11	Primary key of the table
ticket_id	varchar	30	Ticket id of the user
t_id	int	11	Theatre id
user_id	int	11	User id
show_id	int	11	Show id
screen_id	int	11	Foreign key (references

			tbl_screens)
no_seats	int	3	Number of seats
seatRow	int	11	Seat position row
seatCol	Int	11	Seat position column
amount	int	5	Amount
ticket_date	date		Ticket date
date	date		date
status	int	1	Status of approval

Table No : 03

Table Name : tbl_login

Primary Key : id

Foreign Key : uid

Table Description : To store all login information

Fieldname	**Data Type**	**Size**	**Description**
id	int	11	Primary key of the table
uid	int	11	Foreign key (references registration)
username	varchar	50	User name
password	varchar	50	Password
user_type	int	1	User type

Table No : 04

Table Name : tbl_screens

Primary Key : screen_id

Foreign Key :

Table Description : To store Screen Informations

Fieldname	**Data Type**	**Size**	**Description**
screen_id	int	11	Primary key of the table
t_id	int	11	theatre id
screen_name	varchar	110	Screen name

Table No : 05

Table Name : tbl_show_time

Primary Key : st_id

Foreign Key : screen_id

Table Description : To store Informations about Show timing details

Fieldname	**Data Type**	**Size**	**Description**
st_id	int	11	Primary key of the table
screen_id	int	11	Foreign key (references tbl_screens)
name	varchar	40	Show name
start_time	time		Show start time

Table No : 06

Table Name : tbl_movie

Primary Key : movie_id

Foreign Key :

Table Description : To store movie details

Fieldname	**Data Type**	**Size**	**Description**
movie_id	int	11	Primary key of the table
t_id	varchar	11	Theatre id
movie_name	varchar	256	Name of the movie
cast	varchar	500	Cast of the movie
desc	varchar	500	Movie description
release_date	date		Movie releasing date
image	varchar	200	Image of the movie
video_url	varchar	200	Youtube Link of the Movie
status	int	1	Status of approval

Table No : 07

Table Name : tbl_news

Primary Key : news_id

Foreign Key :

Table Description : To store Upcoming Movie details

Fieldname	**Data Type**	**Size**	**Description**
news_id	int	11	Primary key of the table
name	varchar	100	Name of the movie
cast	varchar	100	Cast in the movie
news_date	date		Releasing date
description	varchar	800	Movie description
attachment	varchar	200	Movie image

Table No : 08

Table Name : tbl_shows

Primary Key : sid

Foreign Key : st_id, movie_id

Table Description : To store Informations of each Shows

Fieldname	**Data Type**	**Size**	**Description**
sid	int	11	Primary key of the table
st_id	int	11	Foreign key (references tbl_show_time)
theatre_id	int	11	Theatre id
movie_id	int	11	Foreign key (references tbl_movie)
start_date	date		Movie release date in theatre
status	int	11	Show available status
r_status	int	11	Status of approval

Table No : 09

Table Name : tbl_theatre

Primary Key : id

Foreign Key :

Table Description : To store Theatre Details

Fieldname	**Data Type**	**Size**	**Description**
Id	int	11	Primary key of the table
name	varchar	100	Name of the theatre
address	varchar	100	Address of the theatre
place	varchar	100	Theatre location
state	varchar	50	state
phno	bigint	100	Theatre owner phone no
mail	varchar	30	Email address
pin	int	11	pincode

Table No : 10

Table Name : tbl_categories

Primary Key : category_id

Foreign Key : t_id

Table Description : To store snack category

Fieldname	**Data Type**	**Size**	**Description**
category_id	int	12	Primary key of the table
category_name	varchar	255	Name of the Category
t_id	int	11	Theatre id
category_desc	text		Category Description
image	varchar	200	Image
category_createDate	datetime		Create Date

Table No : 11

Table Name : tbl_screentype

Primary Key : sty_id

Foreign Key : screen_id

Table Description : To store Seat Type

Fieldname	**Data Type**	**Size**	**Description**
screen_id	int	11	Foriegn key (references tbl_screens)
sty_id	int	11	Primary key of the table
type_name	varchar	100	Seat Type Name
position	int	11	Seat Type rank
scRow	varchar	100	Seat Row
scCol	varchar	100	Seat Column
seats	int	11	seats
charge	int	11	Amount

Table No : 12

Table Name : tbl_snackbook

Primary Key : order_id

Foreign Key :

Table Description : To store snack booking information

Fieldname	**Data Type**	**Size**	**Description**
order_id	int	11	Primary key of the table
snackId	int	11	Snack id
itemQuantity	int	100	No of Quantites
user_id	int	11	User id
amount	int	200	Price
orderDate	datetime		Date
status	int	20	status

Table No : 13

Table Name : tbl_snacks

Primary Key : snackId

Foreign Key :

Table Description : To store snacks details

Fieldname	**Data Type**	**Size**	**Description**
snackId	int	12	Primary key of the table
snackName	varchar	255	Snack Name
snackPrice	int	12	Price
snackDesc	text		Description
image	varchar	200	image
snackCategoryId	int	12	Category Id
snackPubDate	datetime	20	Publish Date