DATABASE FINAL PROJECT PHRASE I

TEAM MEMBERS

Jing Liu, Sixiang Chen

TARGET DOMAIN

A Broadway show database

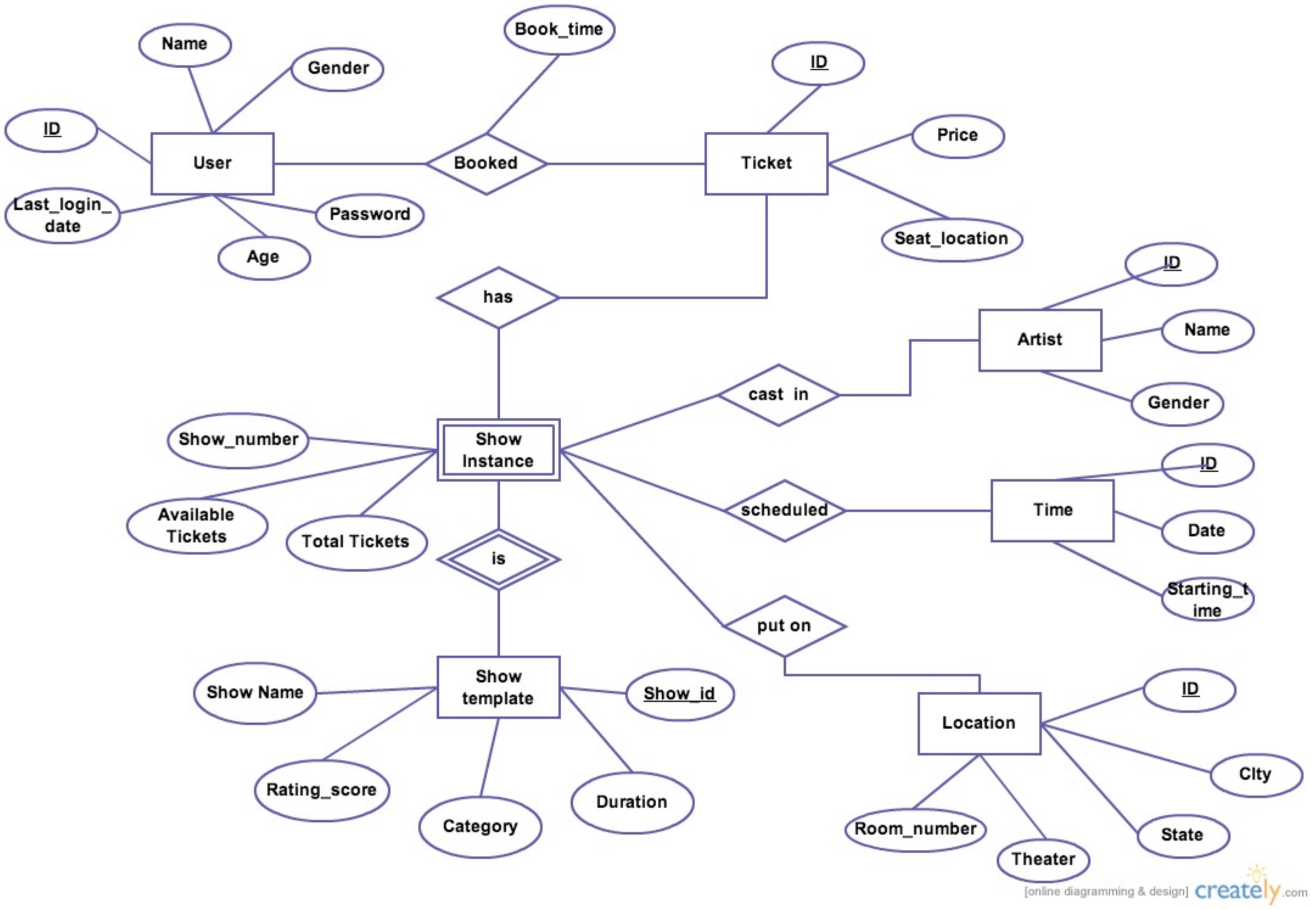
QUESTIONS THAT OUR SYSTEM IS ABLE TO ANSWER

1. List the name, category, score, ticket prices and seat location for a show at a specific time.
2. List the name of all artists who have been in a specific show
3. List all the shows that the chosen artists have been in.
4. List all the shows available during the chosen period of time
5. List the number of tickets left for the specific show
6. List all the shows in a specific category
7. List the booking record for a user
8. List all the shows that have the ratings above the given score
9. List the shows that are played in a specific theater
10. List all the shows that are not shown in anywhere else except for the chosen city.
11. List the amount of selling tickets within the last month
12. List the IDs and names of all users who logged in within the last month
13. List the IDs and names of all users who have booked more than 5 tickets in the past
14. Compute the average ticket price for shows broadcasted in the last month with a specific category and seat location, grouped by theatre locations
15. List the date, time, theater location and price of a show which has the lowest ticket price, grouped by city
16. The system should be able to distinguish administrator from normal users via password
17. List the average age of audience for each show (Only administrator permitted)
18. List the show according to their popularity among female audience (Only administrator permitted)

(18) and (19) are more interesting challenge questions asking for the statistical information

E-R MODEL

Entity-Relationship graph



E-R MODEL TABLE

USER	UserID	User Name	Last_log_in date	Gender	Age	Password
	Jliu88	Jing Liu	2013/11/25	Female	24	123456
	Schen97	Sixiang Chen	2013/09/09	Female	20	234567

BOOKED	UserID	TicketID	Book Time
	Jliu88	0010001A67	2013/11/23
	Jliu88	0020001B32	2013/11/20

TICKET	TicketID	Price	Seat Location
	0010001A67	250.43	Orchestra
	0020001C32	115.28	Rear Mazzanine

HAS	TicketID	ShowID
	0010001A67	0010001
	0020001C32	0020001

LOCATION	LID	Room No.	Theater	City	State
	NYNY0023	1	**GERSHWIN THEATRE**	New York	New York
	MABS0047	5	Boston Opera House	Boston	Massachusetts

PUT ON	LID	ShowID
	NYNY0023	0010001
	MABS0047	0020001

TIME	TimeID	Date	Starting Time
	131123A	2013/11/23	2:00 pm
	131126C	2013/11/26	7:00 pm

SCHEDUALED	TimeID	ShowID
	131123A	0010001
	131126C	0020001

SHOW TEMPLATE	Show Number	Show Name	Category	Duration	Rating Score

	001	The Lion King	Broadway	2 hrs, 30 minutes	9.5
	002	Wicked	Kids-friendly	2 hrs, 45 minutes	9.0

ARTIST	AID	Artist Name	Gender
	0013579	Chantel Riley	Female
	0013580	Alton Fitzgerald White	Male

CAST_IN	AID	ShowID
	0013579	0010001
	0013580	0020001

SHOW INSTANCE	Show Number	ShowID	Total Tickets	Available Tickets
	001	0010001	1500	500

SQL QUERIES

```
-- 1
SELECT ST.show_name, ST.category, ST.rating_score, T.price, T.seat_location
FROM Show_template AS ST INNER JOIN show_instance AS SI ON
ST.show_number = SI.show_number
                INNER JOIN Has AS H ON SI.show_id = H.show_id
                INNER JOIN Ticket AS T ON T.ticket_id = H.ticket_id
                INNER JOIN Schedule AS S ON S.show_id = Ticket.show_id
                INNER JOIN Time AS T1 ON T1.time_id= S.time_id
WHERE T1.Date = '08/21/2013' AND T1.starting_time = '2:00pm'

-- 2
SELECT A.Name, A.Gender
FROM Artist AS A INNER JOIN Cast_In AS CI ON A.artist_id = CI.artist_id
           INNER JOIN Show_instance AS SI ON CI.show_id = SI.show_id
                         INNER JOIN Show_template AS ST ON SI.show_number =
ST.show_number
WHERE SI.show_name = 'The Lion King'

--3
SELECT SI.show_name
FROM Show_template AS ST INNER JOIN Show_instance AS SI ON
SI.show_number = ST.show_number
                    INNER JOIN Cast_in AS CI ON CI.show_id = SI.show_id
                    INNER JOIN Artist AS A ON A.artist_id = CI.artist_id
WHERE A.artist_name ='Ann'

--4
SELECT ST.show_name, SI.left_ticket, L.city, L.theatre, T.Date, T.starting_time
FROM Show_template AS ST INNER JOIN show_instance AS SI ON
ST.show_number = SI.show_number
                   INNER JOIN Perform_in AS PI ON SI.show_id = PI.show_id
                   INNER JOIN Location AS L ON L.location_id = PI.location_id
                   INNER JOIN Schedule AS S ON S.show_id = SI.show_id
                   INNER JOIN Time AS T ON T.time_id = S.time_id
WHERE T.date BETWEEN '03/12/2014' AND '03/14/2014'

-- 5
SELECT ST.show_name, SI.left_ticket
FROM Show_template AS ST INNER JOIN Show_instance AS SI ON
ST.show_number = SI.show_number
WHERE ST.name = 'The Lion King'

--6
SELECT ST.show_name
FROM Show_template AS ST
WHERE ST.Category = 'Music'

-- 7
SELECT B.book_time, T.price, T.seat_location, ST.show_name
FROM User AS U INNER JOIN Booked AS B ON U.user_id = B.user_id
            INNER JOIN Ticket AS T ON T.ticket_id = B. ticket_id
            INNER JOIN Has ON Has.ticket_id = Show_Instance.ticket_id
            INNER JOIN Show_template AS SH ON SH.show_id = SI.show_id
WHERE U.user_id = 'schen97'
```

```
-- 8
SELECT ST.show_name
FROM Show_template AS ST
WHERE T.rating_score > 8.5



-- 9
SELECT ST.show_name, ST.Category, ST.duration
FROM Show_template AS ST INNER JOIN Show_instance AS SI ON
ST.show_number= SI.show_number
                INNER JOIN Perform_in AS P ON P.show_id = SI.show_id
                INNER JOIN Location AS L ON P.LID = L.LID
WHERE L.Theatre = 'Boston Opera House'

-- 10
SELECT ST.show_name, L.Theatre
FROM Show_template AS ST INNER JOIN Show_instance AS SI ON
ST.show_number= SI.show_number
                INNER JOIN Perform_in AS P ON P.show_id = SI.show_id
                INNER JOIN Location AS L ON P.LID = L.LID
WHERE ST.show_name NOT IN
            (SELECT ST.show_name
             FROM Show_template AS ST1 INNER JOIN Show_instance AS SI1 ON
ST1.show_number= SI.show_number
                                    INNER JOIN Perform_in AS P1 ON P1.show_id =
SI.show_id
                            INNER JOIN Location AS L1 ON P1.LID = L1.LID
            WHERE L.State = 'MD' AND L.City = 'Baltimore')

-- 11
SELECT COUNT(H.ticket_id) AS TicketsSold
FROM   Has AS H, Schedualed AS S, Time AS T
WHERE H.show_id=S.show_id AND S.timeid=T.timeid AND
Date_sub(CURDATE(), INTERVAL 30 DAY) <=T.date;

-- 12
SELECT U.userid, U.username
FROM   User AS U
WHERE  Date_sub(CURDATE(),INTERVAL 30 DAY) <=U.LastLoginDate;
```

```
-- 13
SELECT U.userid, U.username, R.TicketsBooked
FROM
(
SELECT B.userid, COUNT(B.ticketid) AS TicketsBooked
FROM   Booked AS B
GROUP BY B.userid
) AS R, User AS U
WHERE R.userid=U.userid AND R.TicketsBooked>=5;

-- 14
delimiter //
CREATE PROCEDURE AveragePrice (IN category varchar(20), IN seat
varchar(20),IN theater varchar(30))
BEGIN
SELECT AVG(T.ticketprice)
FROM   Ticket AS T, Has AS H, Puton AS P, Location AS L, ShowTemplate AS S
WHERE T.ticketid=H.ticketid AND H.showid=P.showid AND P.lid=L.lid AND
L.theater=theater AND T.seatlocation=seat AND S.category=category;
END //
delimiter ;

call AveragePrice('Broadway', 'Ochestra','GERSHWIN THEATER');


-- 15
SELECT ST.show_name, T.Date, T.starting_time, L.state, L.city, L.theatre,
Ticket.Price, Ticket.seat_location
FROM Show_template As ST INNER JOIN Show_instance AS SI ON
ST.show_number = SI.show_number
                INNER JOIN Has AS H ON H.show_id = SI.show_id
                INNER JOIN Ticket ON Ticket.show_id = SI.show_id
                NATURAL JOIN Schedule NATURAL JOIN Time AS T
                NATURAL JOIN Perform_in NATURAL JOIN Location AS L
                INNER JOIN ( SELECT L.city, MIN(T.Price) AS Lowest
                         FROM Ticket AS T NATURAL JOIN Has NATURAL JOIN
Show_instance
                              NATURAL JOIN Perform_in NATURAL JOIN Location
AS L
                    GROUP BY L.city
                      ) AS MinPrice ON MinPrice.lowest = T.price AND MinPrice.city =
L.city
WHERE ST.show_name = 'The Lion King'
```

```
-- 16
DELIMITER $$
DROP PROCEDURE IF EXISTS IdentifyAdministrator;
CREATE PROCEDURE IdentigyAdministrator(In pwd VARCHAR(100))      --
Password should be entered using ''
BEGIN
  IF EXISTS ( SELECT * FROM User AS U WHERE U.Name='Administrator' AND
pwd = U.password ) THEN
    -- (...the behavior the administrator can do here...)
  END IF;
END;
$$
DELIMITER ;
```

```
-- 17
SELECT ST.show_name, AVG(U.age) AS AverageAge
FROM User AS U INNER JOIN Booked ON U.user_id = Booked.user_id
            INNER JOIN Ticket ON Ticket.ticket_id = Booked_ticket_id
            INNER JOIN Has ON Has.ticket_id = Ticket.ticket_id
            INNER JOIN Show_Instance ON Show_Instance.show_id = Has.show_id
                            INNER JOIN Show_template AS ST ON St.show_number =
Show_Instance.show_number
GROUP BY ST.show_name
```

```
-- 18
SELECT ST.Category, COUNT(U.user_id) AS WomenWhoLike
FROM User AS U INNER JOIN Booked ON U.user_id = Booked.user_id
            INNER JOIN Ticket ON Ticket.ticket_id = Booked_ticket_id
            INNER JOIN Has ON Has.ticket_id = Ticket.ticket_id
            INNER JOIN Show_Instance ON Show_Instance.show_id = Has.show_id
```

```
                    INNER JOIN Show_template AS ST ON St.show_number =
Show_Instance.show_number
WHERE U.Gender = 'Female'
GROUP BY ST.Category
ORDER BY WomenWhoLike
```

LOAD DATABASE WITH VALUES

We plan to collect the data we need form the official website of Broadway.

The source URL: http://www.broadway.com/

We can have access to almost every information we need for our system from this website, the problems involved include user information, ID for show, artists, theaters and etc. Since we cannot have access to the above information, we need to create some imaginary users and assign IDs if needed.

DESCRIPTION OF USER INTERFACE

We plan to generate web pages where users can search for the information about Broadway shows. We also have the idea that our system can distinguish administrator from ordinary users and the administrator can have access to some overall statistical information. Moreover, our system has the function of booking tickets.

FOCUS TOPICS

In our design, we plan to cover some advanced topics. We mainly plan to focus on advanced GUI interface. We hope to create the interface that is user-friendly. Users can search for information by typing in some keywords and can conveniently book tickets.

We also cover issues of advanced SQL topics. We plan to use SQL procedure, triggers, cursors and JDBC in our programming.

DATABASE PLATFORM

The platform we plan to use is MYSQL 5.1.63 on MacBook Pro with 8GB of memory.