CMSC 508 Semester Project
Phase II
11/30/19

Angel De La Rosa
Heman Baral
Zachary Smith

<u>Problem Statement:</u>

A modern museum has an incredible amount of data to track from works of art, employees, events and much more. For the museum database we have developed, the intended end user is a trusted employee of the museum. One who can change, add and remove works from wings, add artists to the museum, hire and fire other employees. In short, our database is for use by the managerial staff of the museum.

We will take a look at a real world museum and simplify it to the scope of this project. A museum at its core will be made up of different wings. A wing will be home to many different types of works of art and have a specified available and total space for displaying works of art. Each work of art displayed will have a creator, here tracked by artist. Information about an artist will be kept. A wing may also play host to an event, which may or may not be open to the public depending on the nature of the event. Hundreds of qualified people may be necessary for the real world operation of a museum, which in our database will be simplified down into 3 different types of distinct employees, curators, security and housekeepers.

A museum will be made up of multiple wings, which itself would be made up of a collection of works of art. Each wing of the museum has a total amount of display space, and a value for total space that is used. Each wing will be identified by a unique wing ID, and will have its own wing name, for example “European Art Wing”. A wing will host events, be worked in by employees, be visited by guests, have curators and be home to many works of art.

Each employee of the museum will have one of three specializations. Information kept on every employee will be a name, social security number, first and last, an empID used to uniquely identify them and a single contact phone number. The first of the three employee types is a security worker. Additional information tracked about security workers will be their salary and the shift that they work. A shift will either be day or night shift. The second of the employee specializations is the housekeeper. Additional information kept about housekeepers is their hourly pay rate, and how many hours they work every week. The final type of employee is the curator. A curator will have their yearly salary and what wing they are employed in as the additional information kept. An employee may work at any number of events, including not working any events.

The museum will host events. An event will be identified by its unique eventID, it will have an event name, or title, and will be hosted in a specific wing of the museum. This is to reflect the real world occurrence of museums often hosting fundraisers, weddings or any number of other such things. The event will have a special binary field, public, which will be a measure of whether or not the event is open to the public. An event can be attended by one or many guests, is hosted in one and only one wing, and will be worked by one or more employees.

A guest is the model of a real world visitor to the museum. As such we will require a sign in to visit the museum. Information of the guest will consist of a guestID, their first and last names, an email and their status, a binary measure for whether or not they are currently in the museum. A guest will attend zero or many events and visit one or many wings of the museum.

Like any good museum, ours will contain a large variety of different kinds of art in different styles and in different mediums. Each work of art will have its own unique artID, the medium in which it was made, the genre it falls under and the artist's ID of the artist who made it as well as the wing of the museum that plays host to it. Every piece of art is hosted in one and only one wing and is made by one and only one artist.

Every artist that is tracked in our database will create zero or many art works. Information tracked for the artist is their name, first and last, their nationality and their specialty. Specialty is a special string attribute from a limited selection, for example an artist is a "painter" or "sculptor" or a few other limited possibilities.

The database's potential uses include keeping track of inventory for a museum, changing staff positions including hiring and firing employees, and tracking events that will take place at the museum. Many of the following functionalities fulfill these and other purposes, and are listed alongside their mySQL queries.

As the intended user of the database is the museum management staff, all queries are implemented as to be used exclusively by them. Since a guest would not have access to this

1. How to add new employee?

2. How to deleting new employees?

3. How to add a new art piece?

4. How to deleting an art piece that are no longer in the museum?

5. How to deleting artist that are no longer in the museum?

6. How to add a new artist?

7. List all the employee working?

8. List all the art piece in a particular wing?

9. Displaying art pieces with their title?

10. List all the artist who has their art listed?
    a. SELECT CONCAT(firstName, ' ', lastName) AS 'NAME' FROM artist;

11. How to delete an event?

12. Looking up upcoming events?
    a. SELECT * FROM Event;

13. Which wing is hosting an event?
    a. SELECT wing FROM event WHERE eventId = X;

14. Who is working on each event?
    a. SELECT empID FROM works WHERE eventID = X;

15. How many more works can be placed in each wing?

16. Lists of all curators with their respective wing?
    a. SELECT * FROM supervises;

17. List of all the people that are attending an event?
    a. SELECT * FROM attends WHERE eventID = X;

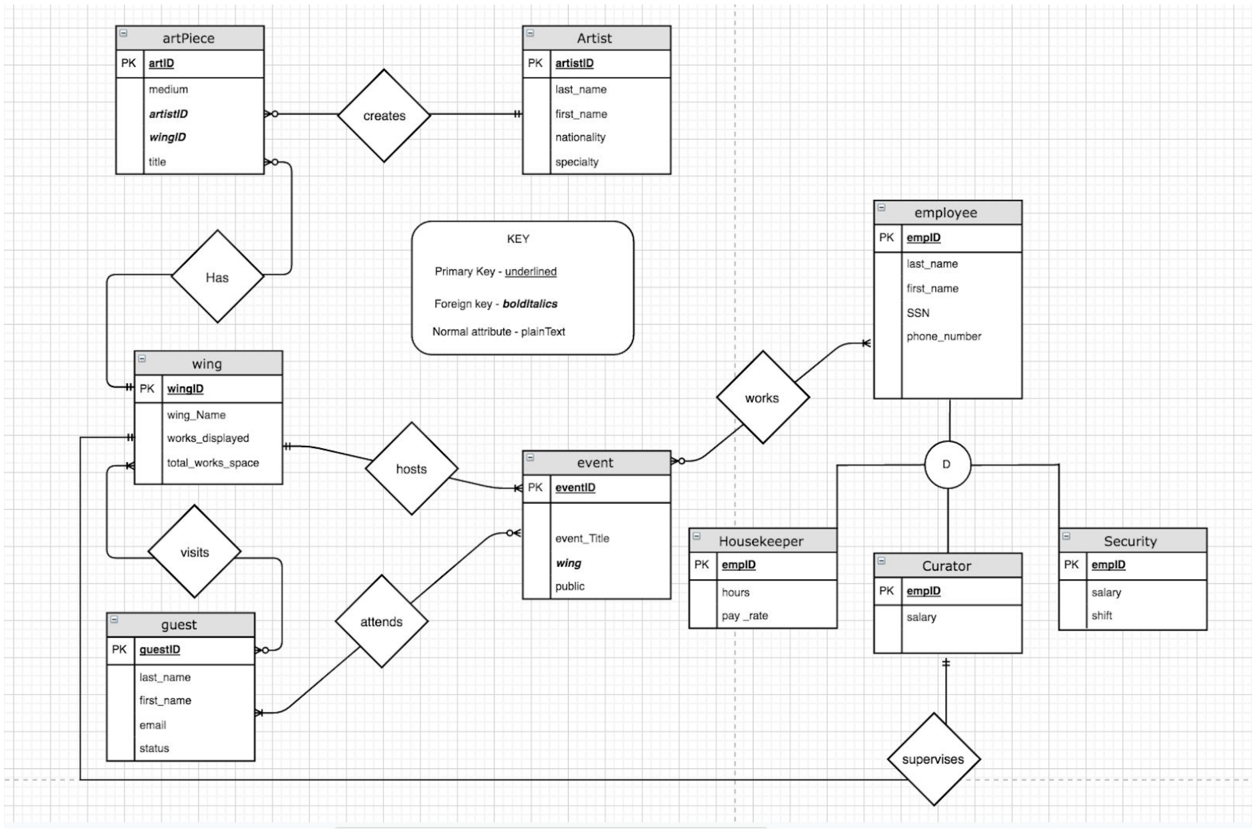
18. Housekeeper currently working in a museum?
    a. #How many housekeepers are currently working?
       SELECT COUNT(*) FROM housekeeper;

       #Information about housekeepers?
       SELECT * FROM housekeepers;

19. List of security working in each shift?
    a. SELECT * FROM security WHERE shift = 'night';
       #OR
       SELECT * FROM security WHERE shift = 'day';

20. How to add an event?

ERD:



Heman Baral *Zachary Smith* *Angel De La Rosa*

Relational Model:

ArtPiece(artID, medium, artistID, wing, genre)
artID – PK, Number from 0-1000, not null
medium – set of possible mediums used to create that art, not null, varchar
artistId – FK references ArtistId from Artist, number, not null
wingId – FK references wingID from Wing, number, not null
title – set of possible titles of work of art, varchar
artId, artistId and wingId are unique

Artist(artistId, last_name, first_name, nationality, specialty)
artistId – PK, Number form 0-1000
last_name – set of possible last names, not null, varchar
first_name – set of possible first names, not null, varchar
nationality – set of possible nationalities, not null, varchar
specialty - set of possible specialty, not null, varchar
artistId is only attribute that is unique

Wing(wingId, wing_Name, works_displayed, total_works_space)
wingId – PK, number from 0-20. Not null
wing_Name – set of possible wing names in the building, varchar, not null
Works_displayed - number from 0-100 of the amount of work in the wing
Total_works_space – number from 0-100 of the amount of possible works to hang
wingId is only attribute that is unique

Event(eventid, event_Title, wingID, public)
eventId – PK, number from 0-100, not null
event_Title – set of possible event titles, varchar, not null
wingID – FK from Wing, number from 0-100, not null
public - number from 0 or 1 to indicate whether it is open to the public, varchar, not null

eventId and wingId are unique

Guest(guestId, last_name, first_name, email, status)
guestID- PK, number 0-100
last_name – set of possible last names, not null
first_name – set of possible first names, not null
email – set of possible emails with unlimited characters, not null
status- numeric value of 0 or 1 to indicate if they are in a museum, not null
guestID and email are unique

Employee(empID, last_name, first_name, phone_number, SSN)
empId- PK, number from 0-100
last_name – set of possible last names, not null
first_name – set of possible first names, not null
Phone_number – numeric value with 10 digits for area code and number , not null
SSN - numeric value with 4 digits, not null, unique
empId is only attribute that is unique

Housekeeper(empID, hours, pay_rate)
empID – PK, foreign key that references Employee, number from 0-100
hours- number from 0-100
pay_rate – Numeric Currency value with 2 decimal places
empId is only attribute that is unique

Curator(empId, salary)
empID – PK, foreign key that references Employee, number from 0-100
salary – Numeric Currency value with 2 decimal places
empId is only attribute that is unique

Security(empId, salary, shift)
empID – PK, foreign key that references Employee, number from 0-100
salary – Numeric Currency value with 2 decimal places
shift – Time from when there work will start and end
empId is only attribute that is unique

Works(empId, eventId)
empId - PK, foreign key that references Employee, number from 0-100
eventId- PK, forieign key that references Event, number from 0-100

ArtPiece(artID, medium, artistID, wingId, genre)

Candidate Keys: artId
Primary Key: artId
FD: artId -> medium, artistId, wingId, genre

Artist(<u>artistId</u>, last_name, first_name, nationality, specialty)
Candidate Keys: artistID
Primary Key: artistID
FD: artistID -> last_name, first_name, nationality, specialty

Wing(<u>wingId</u>, wing_Name, works_displayed, total_works_space)
Candidate Keys: wingId
Primary Key: wingId
FD: wingId -> wing_Name, works_displayed, total_works_space

Event(<u>eventid</u>, event_Title, wingID, public)
Candidate Keys: eventid
Primary Key: eventid
FD: eventid -> event_Title, wingID, public

Guest(<u>guestId</u>, last_name, first_name, email, status)
Candidate Keys: guestId
Primary Key: guestId
FD: guestId -> last_name, first_name, email, status

Employee(<u>empID</u>, last_name, first_name, phone_number, SSN)
Candidate Keys: empID
Primary Key: empID
FD: empID-> last_name, first_name, phone_number, SSN

Housekeeper(<u>empID</u>, hours, pay_rate)
Candidate Keys: empID
Primary Key: empID
FD: empID -> hours, pay_rate

Curator(<u>empId</u>, salary)
Candidate Keys: empID
Primary Key: empID
FD: empID -> salary

Security(<u>empId</u>, salary, shift)
Candidate Keys: empID
Primary Key: empID
FD: empID -> salary, shift

Works(<u>empId</u>, <u>eventId</u>)
Candidate Keys: empId, eventId
Primary key: empId, eventId

Example Tables:

Artist

artistID	**last_name**	**first_name**	**nationality**	**speciality**
1	Da Vinci	Leonardo	Italian	Painter
2	Van Gogh	Vincent	Dutch	Painter
3	Picasso	Pablo	Spainish	Painter
4	Rodin	Auguste	French	Sculptor

artPiece

artID	**Title**	**artistID**	**wingID**	**Medium**
1	Les Femmes d’Alger	3	1	Oil on canvas

2	Portrait of Dr. Gachet	2	3	Oil on canvas
3	Mona Lisa	1	4	Oil on poplar panel
4	The Walking Man	4	2	Bronze sculpture

Wing

wingID	**wing_name**	**works_displayed**	**total_works_capacity**
1	Eastern Art	50	50
2	American Art	100	100
3	European Art	65	75
4	African Art	48	50

Guest

guestID	**last_name**	**first_name**	**email**	**status**
1	Smith	Joe	smith@email.com	1
2	Jones	Bob	jones@email.com	0
3	White	Bill	white@email.com	0

Event

eventID	**event_Title**	**wing**	**public**
1	Cancer Fundraiser	2	1
2	Bob Wedding	3	0

3	Special Display	4	1

Employee

empID	**last_name**	**first_name**	**phone_number**
1	Smith	John	8043334421
2	Davis	Carol	3430334890
3	Moore	Bobby	5450091823
4	Robertson	Andy	7179903920
5	Howard	Dwight	5159901234

Housekeeper

empID	**hours**	**pay_rate**
2	30	14.0
1	40	14.0

Curator

empID	**Salary**
3	60,000

Security

empID	**Salary**	**Shift**
4	40,000	Night
5	40,000	Day

Works

empId	**eventId**
1	3