Database Design Project

Phase3 – Implementation

Gary Steelman | gxs112030@utdallas.edu

Contents

Introduction

This report comprises multiple sections detailing the design of the database system for a given problem description.

- Problem Description – In this section I outline the customer's problem and all of their requirements and statements. I also include answers to the three questions posed in the instruction document.
- Enhanced Entity Relationship (EER) Diagram – In this section I show the EER diagram corresponding to the database's design. This design is a visual, high level design intended for the reader's conceptual understanding of the organization of the data.
- Relation Schemas Diagram – In this section I show the finalized relation schemas in 3NF and their key constraints and foreign key constraints. I also include data type constraints for the implementation in SQL.
- Functional Dependency Diagram – In this section I show a diagram containing the functional dependencies of each relation.
- Sample SQL Transactions and Demonstration Script – In this section I include a sample script that creates the tables and views, populates them with sample data, performs some sample queries including those requested in the instructions, and then drops the tables.

Problem Description

Listed below is the problem description directly from the instruction document. All of the later design sections are based on this problem description. The description is as follows:

Customer Requirements and Explanations

ABC City Library benefits much to people living in the cities around it, which depends on abundant reading resources, a great number of volunteers and employees.

Reading resources can be mainly divided into three types: Book, Video and Mag-Pap (including magazines and papers published periodically). All the reading resources share some common attributes: Call_Number (like "QA123456", unique), Name, Borrow_Status (can only be "available" or "unavailable"), Reading_Status (can only be "for borrow" or "in library reading only"). For every book, there is a summary to describe the main idea of the book. For each video, the system will record its type (VCD, DVD, cassette, etc.). Publish cycle (bi-weekly, monthly etc.) and subjects (fashion, cooking, business etc.) are recorded for every kind of magazine and paper. One magazine or paper may include several subjects.

For convenience, the library will record the information of the publishers who supply reading materials and the information of the authors. The library will record the publishers? Name, Webpage, E-mail, and Phone Number. The name should be unique for each publisher because of name patent right. Several reading resources may share the same publisher. However, one reading resource can only be published by one publisher. For the authors, the library will record their Author_ID (unique) and Name (including first name and last name), Phone_Number. The value of Author_ID is between "00001"and "99999".

One book may have several authors, while different volumes of magazine or paper may have different author sets. Meanwhile, one author may have many works.

To organize the library better, the library record information in details for those people who serve or use the library. The library needs to record each person?s name (including first name and last name), address (APT, RD, CITY, STATE, and ZIPCODE), and age. The system will assign a unique id to each person generated by picking out the first letter of the first name and the last name with a randomly generated letter in the middle, then, putting a randomly generated integer with six digits at the end. For example, for the person named Mary Lee, the id can be "mxl000001", where x and 000001 are randomly generated. There are mainly three roles of people involved in the library --- Reader, Employee, and Volunteer. One employee can also be a volunteer. For the employees, their responsibility should be recorded, while for the volunteers, their available weekday and time-slots should be recorded. For safety, the age of each volunteer cannot be over 75. For each reader, he/she can sponsor at most five friends or relatives to use the resources in the library and only needs to offer the sponsored person?s name. The sponsored person enjoys all rights that one reader has. In the library, only reader and his/her sponsored person(s) can borrow readings. The system needs to record every time?s check out --- borrow date, due date, and return date, where return date can be earlier or later than due date. And one reader and his/her sponsored person(s) together cannot keep more than 10 readings at the same time. In addition, one person can make a reservation if the resources are unavailable. The policy is that if one reading is reserved, the person keeping it must return it within one week no matter what the due day should be. Thus, the due time of last check out will be updated correspondingly.

The library is a non-profit institution and it needs investments from the government. Since the resources of the library are limited, it cannot be open to all people. The policy of the library is that a qualified reader must be a resident living in the city whose government invests on the library, however, there is no constraint on his/her sponsored person. There is no constraint on volunteers, either. For each city, the system will record whether the city invests on the library or not (can only be „YES? or „NO?).

The library often holds special events of different themes for its readers, like health lecture, tutor etc. The event id (unique), held time and rough introduction of each event will be recorded. The event holders can be employees or volunteers. These events may be held in different cities. Thus, the system needs to record the holders, city for each event. Every reader and his/her sponsored person can attend every event alone or together. And attendees need evaluate the events they attend. The evaluation score varies from 0 to 100

Answers to Additional Questions

Question 1

Can you think 5 more rules (other than the one explicitly described above) that are likely to be used in a library.

1. Not all resources the library has are books, videos, magazines, or papers. The customer specifications state "most resources" are of the aforementioned types, but do not explicitly state all. For example: there may be software or electronic hardware cameras available.

2. Videos may not have more than one format. For example: a specific documentary may be available on both DVD and Blu-Ray formats, but each item will be borrowed individually and should be entered into the system as different resources.
3. An author may have more than one contact phone number. For example: an author may have a cell and work phone number.
4. Not all persons involved with library operations are employees, readers, guests, or volunteers. The customer specifications state "most people" are of the aforementioned types, but do not explicitly state all. For example: there may be a director person who does not actively work for the specific library, but a system of libraries.
5. A resource may only be of one format at a time. For example: a book may not also be a video.
6. A person may hold any combination of titles. For example: an employee may also be a reader.

Question 2

Is the ability to model super-class/subclass relationships likely to be important in such environment? Why or why not?

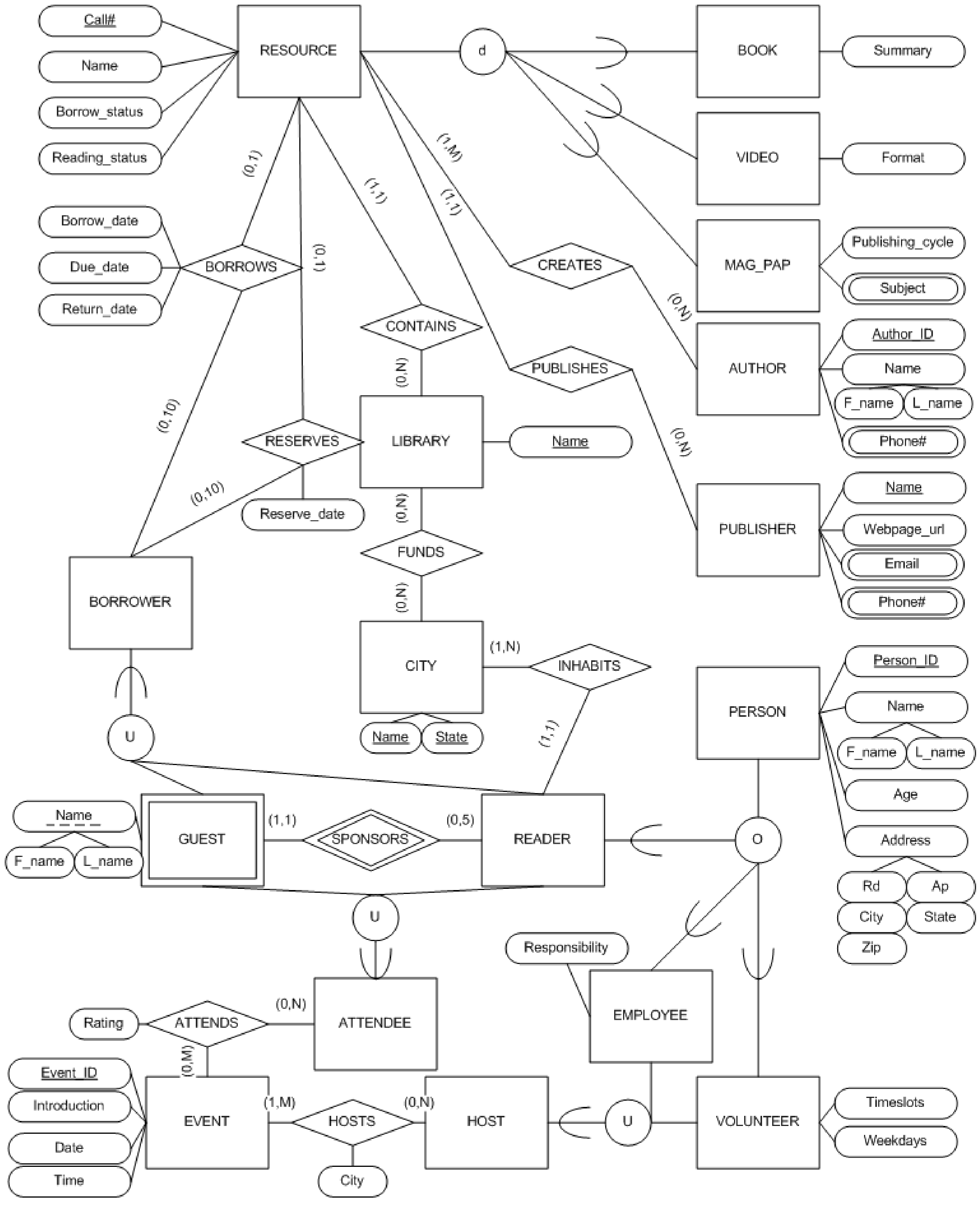
The ability to model a superclass/subclass relationship is important in this environment because there are many "is-a" relationships, such as a video or book both being a resource or a reader or employee both being a person. I can reduce data redundancy and also capture the polymorphism of the project using these relationships.

Question 3

Justify using a Relational DBMS like Oracle for this project.

A relational DBMS is a good fit for this system. The data in the database will not mutate rapidly and will not require constant changes. A book's title will not change once entered into the system, etc. Fast query time is desirable. These two observations lend themselves toward a relational DBMS.

Enhanced Entity Relationship (EER) Diagram



Relation Schemas Diagram

Relational Schemas, Primary Keys, and Foreign Keys

RELATION	Referential (Integrity) Constraints
RESOURCE(Call#, Name, Borrow_status, Reading_status, P_name)	P_name -> PUBLISHER.Name
RESOURCE_LOCATION(R#, L_name)	[FK] R# -> RESOURCE.Call# L_name -> LIBRARY.Name
BOOK(R#, Summary)	[FK] R# -> RESOURCE.Call#
VIDEO(R#, Format)	[FK] R# -> RESOURCE.Call#
MAG_PAP(R#, Publishing_cycle)	[FK] R# -> RESOURCE.Call#
AUTHOR(Author_ID, F_name, L_name)	
PUBLISHER(Name, Webpage_url)	
LIBRARY(Name)	
CITY(Name, State)	
PERSON(Person_ID, F_name, L_name, Age, Rd, Ap, City, State, Zip)	
EMPLOYEE(P_ID, Responsibility)	[FK] P_ID -> PERSON.Person_ID
VOLUNTEER(P_ID, Timeslots, Weekdays)	[FK] P_ID -> PERSON.Person_ID
READER(P_ID, C_name, C_state)	[FK] P_ID -> PERSON.Person_ID C_name -> CITY.Name C_state -> CITY.State
GUEST(F_name, L_name, R_ID)	R_ID -> READER.P_ID
HOST(Host_ID, E_ID, V_ID)	[FK] E_ID -> EMPLOYEE.P_ID [FK] V_ID -> VOLUNTEER.P_ID
ATTENDEE(Attendee_ID, G_f, G_l, R_ID)	G_f -> GUEST.F_name G_l -> GUEST.L_name R_ID -> READER.P_ID
EVENT(Event_ID, Introduction, Date, Time)	
BORROWER(Borrower_ID, G_f, G_l, R_ID)	G_f -> GUEST.F_name G_l -> GUEST.L_name R_ID -> READER.P_ID
BORROWS(R#, B_ID, Borrow_date, Due_date, Return_date)	[FK] R# -> RESOURCE.Call# [FK] B_ID -> BORROWER.Borrower_ID
RESERVES(R#, B_ID, Reserve_date)	[FK] R# -> RESOURCE.Call# [FK] B_ID -> BORROWER.Borrower_ID
CREATES(A_ID, R#)	[FK] A_ID -> AUTHOR.Author_ID [FK] R# -> RESOURCE.Call#
FUNDS(C_name, C_state, L_name)	[FK] C_name -> CITY.Name [FK] C_state -> CITY.State [FK] L_name -> LIBRARY.Name [FK] {C_name, C_state} -> {CITY.Name, CITY.State}
HOSTS(H_ID, E_ID, City)	[FK] H_ID -> HOST.Host_ID [FK] E_ID -> EVENT.Event_ID
ATTENDS(A_ID, E_ID, Rating)	[FK] A_ID -> ATTENDEE.Attendee_ID

	[FK] E_ID -> EVENT.Event_ID
MAG_PAP_SUBJECT(Mp#, Subject)	[FK] Mp# -> MAG_PAP.R#
AUTHOR_PHONE(A_ID, Phone#)	A_ID -> AUTHOR.Author_ID
PUBLISHER_EMAIL(P_name, Email)	P_name -> PUBLISHER.Name
PUBLISHER_PHONE(P_name, Phone#)	P_name -> PUBLISHER.Name

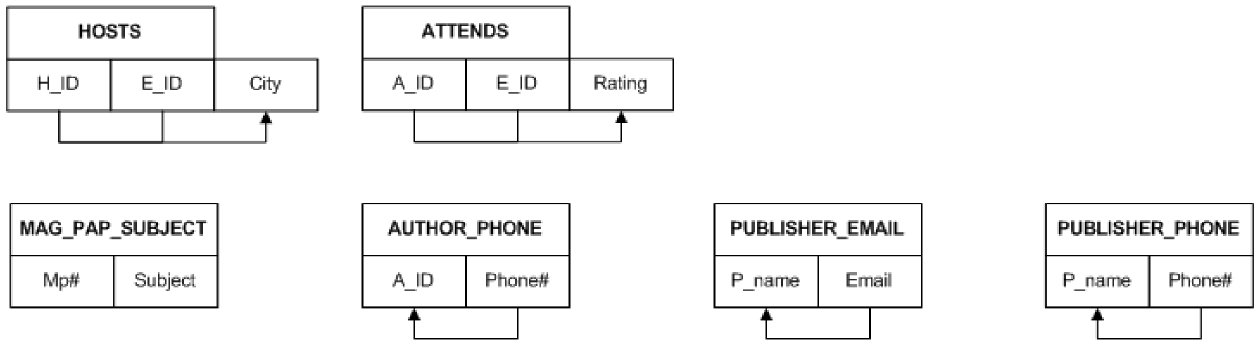
Relation Attribute Datatypes

RELANTION	**Attributes**	**Data type and constraints**
RESOURCE	Call#	string, 8 chars, non-null, unique
	Name	string <= 60 chars
	Borrow_status	string <= 20 chars; “available” or “unavailable”
	Reading_status	string <= 20 chars; “for borrow” or “in library reading only”
	P_name	string <= 60 chars
RESOURCE_LOCATION	R#	string, 8 chars, non-null, unique
	L_name	string <= 60 chars
BOOK	R#	string, 8 chars, non-null, unique
	Summary	text <= 500 chars
VIDEO	R#	string, 8 chars, non-null, unique
	Format	string <= 10 chars; “VCD”, “DVD”, “cassette”, “USB”
MAG_PAP	R#	string, 8 chars, non-null, unique
	Publishing_cycle	string <= 20 chars; “bi-weekly”, “monthly”, “bi-annually”, “annually”
AUTHOR	Author_ID	string, 5 chars; [“00001”, “99999”], non-null, unique
	F_name	string <= 20 chars
	L_name	string <= 20 chars
PUBLISHER	Name	string <= 30 chars, non-null, unique
	Webpage_url	string <= 60 chars
LIBRARY	Name	string <= 60 chars, non-null, unique
CITY	Name	string <= 60 chars, non-null
	State	string <= 15 chars, non-null
	{Name, State}	unique
PERSON	Person_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null, unique
	F_name	string <= 60 chars, non-null
	L_name	string <= 60 chars, non-null
	Age	integer
	Rd	string <= 30 chars
	Ap	string <= 30 chars
	City	string <= 60 chars
	State	string <= 15 chars
	Zip	integer
EMPLOYEE	P_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null, unique
	Responsibility	string <= 60 chars
VOLUNTEER	P_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null, unique
	Timeslots	integer
	Weekdays	integer

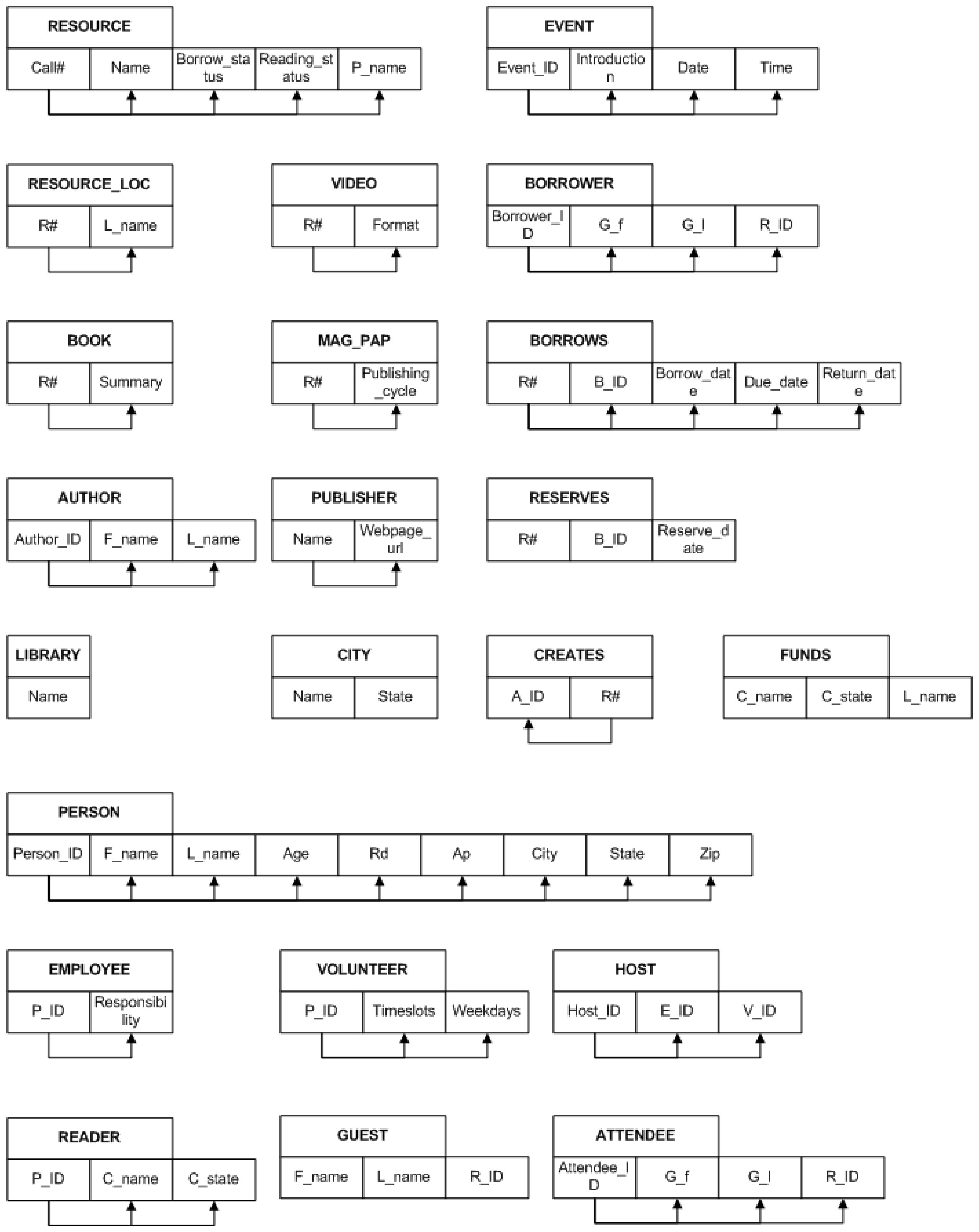
	(Age)	integer <= 75
READER	P_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null, unique
	C_name	string <= 60 chars, non-null
	C_state	string <= 15 chars, non-null
GUEST	F_name	string <= 60 chars, non-null
	L_name	string <= 60 chars, non-null
	R_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null
	{F_name, L_name, R_ID}	unique
HOST	Host_ID	integer, non-null, unique, auto-increment
	E_ID	string, 9 chars; {f + c + l + xxxxxx}
	V_ID	string, 9 chars; {f + c + l + xxxxxx}
	{E_ID, V_ID}	at least one is non-null
ATTENDEE	Attendee_ID	integer, non-null, unique, auto-increment
	G_f	string <= 60 chars
	G_l	string <= 60 chars
	R_ID	string, 9 chars; {f + c + l + xxxxxx}
	{{G_f, G_l}, R_ID}	at least one is non-null
EVENT	Event_ID	integer, non-null, unique, auto-increment
	Introduction	text <= 500 chars
	Date	string, 10 chars, “MM/DD/YYYY”
	Time	string, 8 chars, “HH:MM:SS”
BORROWER	Borrower_ID	integer, non-null, unique, auto-increment
	G_f	string <= 60 chars
	G_l	string <= 60 chars
	R_ID	string, 9 chars; {f + c + l + xxxxxx}
	{{G_f, G_l}, R_ID}	at least one is non-null
BORROWS	R#	string, 8 chars, non-null
	B_ID	integer, non-null
	Borrow_date	string, 10 chars, “MM/DD/YYYY”
	Due_date	string, 10 chars, “MM/DD/YYYY”, >= Borrow_date
	Return_date	string, 10 chars, “MM/DD/YYYY”, >= Borrow_date
	{R#, B_ID}	unique
RESERVES	R#	string, 8 chars, non-null
	B_ID	integer, non-null
	Reserve_date	string, 10 chars, “MM/DD/YYYY”
	{R#, B_ID}	unique
CREATES	A_ID	string, 5 chars; [“00001”, “99999”], non-null
	R#	string, 8 chars, non-null
	{A_ID, R#}	unique
FUNDS	C_name	string <= 60 chars, non-null
	C_state	string <= 15 chars, non-null
	L_name	string <= 60 chars, non-null
	{C_name, C_state, L_name}	unique
HOSTS	H_ID	integer, non-null

	E_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null
	City	string <= 60 chars
	{H_ID, E_ID}	unique
ATTENDS	A_ID	integer, non-null
	E_ID	integer, non-null
	Rating	integer [0,10]
	{A_ID, E_ID}	unique
MAG_PAP_SUBJECT	Mp#	string, 8 chars, non-null
	Subject	string <= 20 chars, non-null
	{Mp#, Subject}	unique
AUTHOR_PHONE	A_ID	string, 5 chars; [“00001”, “99999”], non-null
	Phone#	string, 12 chars; “xxx-xxx-xxxx”, non-null
	{A_ID, Phone#}	unique
PUBLISHER_EMAIL	P_name	string <= 30 chars, non-null
	Email	string <= 40 chars, non-null
	{P_name, Email}	unique
PUBLISHER_PHONE	P_name	string <= 30 chars, non-null
	Phone#	string, 12 chars; “xxx-xxx-xxxx”, non-null
	{P_name, Phone#}	unique

Functional Dependency Diagram



(Continued on next page)


RESOURCE
Call# | Name | Borrow_status | Reading_status | P_name
EVENT
Event_ID | Introduction | Date | Time
RESOURCE_LOC
R# | L_name
VIDEO
R# | Format
BORROWER
Borrower_ID | G_f | G_l | R_ID
BOOK
R# | Summary
MAG_PAP
R# | Publishing_cycle
BORROWS
R# | B_ID | Borrow_date | Due_date | Return_date
AUTHOR
Author_ID | F_name | L_name
PUBLISHER
Name | Webpage_url
RESERVES
R# | B_ID | Reserve_date
LIBRARY
Name
CITY
Name | State
CREATES
A_ID | R#
FUNDS
C_name | C_state | L_name
PERSON
Person_ID | F_name | L_name | Age | Rd | Ap | City | State | Zip
EMPLOYEE
P_ID | Responsibility
VOLUNTEER
P_ID | Timeslots | Weekdays
HOST
Host_ID | E_ID | V_ID
READER
P_ID | C_name | C_state
GUEST
F_name | L_name | R_ID
ATTENDEE
Attendee_ID | G_f | G_l | R_ID

Sample SQL Transactions and Demonstration Script

The following SQL script was created to run in SQL*Plus on the UT Dallas campus Oracle machine. Part of the way down the sample queries for project phase 3 begin. Each query is labeled by "-- <letter>".

```
spool output.txt

set echo on

DROP TABLE LIBRARY CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER CASCADE CONSTRAINTS;
DROP TABLE R_RESOURCE CASCADE CONSTRAINTS;
DROP TABLE RESOURCE_LOCATION CASCADE CONSTRAINTS;
DROP TABLE BOOK CASCADE CONSTRAINTS;
DROP TABLE VIDEO CASCADE CONSTRAINTS;
DROP TABLE MAG_PAP CASCADE CONSTRAINTS;
DROP TABLE AUTHOR CASCADE CONSTRAINTS;
DROP TABLE CITY CASCADE CONSTRAINTS;
DROP TABLE PERSON CASCADE CONSTRAINTS;
DROP TABLE EMPLOYEE CASCADE CONSTRAINTS;
DROP TABLE VOLUNTEER CASCADE CONSTRAINTS;
DROP TABLE READER CASCADE CONSTRAINTS;
DROP TABLE GUEST CASCADE CONSTRAINTS;
DROP TABLE HOST CASCADE CONSTRAINTS;
DROP TABLE ATTENDEE CASCADE CONSTRAINTS;
DROP TABLE EVENT CASCADE CONSTRAINTS;
DROP TABLE BORROWER CASCADE CONSTRAINTS;
DROP TABLE BORROWS CASCADE CONSTRAINTS;
DROP TABLE RERSERVES CASCADE CONSTRAINTS;
DROP TABLE CREATES CASCADE CONSTRAINTS;
DROP TABLE FUNDS CASCADE CONSTRAINTS;
DROP TABLE HOSTS CASCADE CONSTRAINTS;
DROP TABLE ATTENDS CASCADE CONSTRAINTS;
DROP TABLE MAG_PAP_SUBJECT CASCADE CONSTRAINTS;
DROP TABLE AUTHOR_PHONE CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER_EMAIL CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER_PHONE CASCADE CONSTRAINTS;

DROP VIEW BORROWED_RESOURCES;
DROP VIEW LIKED_EVENTS;
DROP VIEW RESOURCES_OUT_ON_DATE;

CREATE TABLE LIBRARY (
Name VARCHAR(60) NOT NULL,
PRIMARY KEY(Name)
);

CREATE TABLE PUBLISHER (
Name VARCHAR(30) NOT NULL UNIQUE,
Webpage_url VARCHAR(60) NOT NULL UNIQUE,
PRIMARY KEY (Name, Webpage_url)
);

CREATE TABLE R_RESOURCE (
Call_num CHAR(8) NOT NULL,
Name VARCHAR(60) NOT NULL,
Borrow_status VARCHAR(25) NOT NULL,
Reading_status VARCHAR(25) NOT NULL,
P_name VARCHAR(60) NOT NULL,
PRIMARY KEY (Call_num),
FOREIGN KEY (P_name) REFERENCES PUBLISHER(Name) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE RESOURCE_LOCATION (
R_num CHAR(8) NOT NULL,
L_name VARCHAR(60) NOT NULL,
PRIMARY KEY (R_num, L_name),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (L_name) REFERENCES LIBRARY(Name) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE BOOK (
R_num CHAR(8) NOT NULL UNIQUE,
Summary VARCHAR(500) NOT NULL,
PRIMARY KEY (R_num, Summary),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE
);

CREATE TABLE VIDEO (
R_num CHAR(8) NOT NULL UNIQUE,
V_format VARCHAR(10) NOT NULL,
PRIMARY KEY (R_num, V_format),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE
);

CREATE TABLE MAG_PAP (
R_num CHAR(8) NOT NULL UNIQUE,
Publishing_cycle VARCHAR(20) NOT NULL,
PRIMARY KEY (R_num, Publishing_cycle),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE
);

CREATE TABLE AUTHOR (
Author_ID CHAR(5) NOT NULL,
F_name VARCHAR(20) NOT NULL,
L_name VARCHAR(20) NOT NULL,
PRIMARY KEY (Author_ID)
);

CREATE TABLE CITY (
Name VARCHAR(60) NOT NULL,
C_state VARCHAR(15) NOT NULL,
PRIMARY KEY(Name, C_state)
);

CREATE TABLE PERSON (
Person_ID CHAR(9) NOT NULL,
F_name VARCHAR(60) NOT NULL,
L_name VARCHAR(60) NOT NULL,
Age INT NOT NULL,
Rd VARCHAR(30) NOT NULL,
Ap VARCHAR(30),
City VARCHAR(30) NOT NULL,
C_State VARCHAR(30) NOT NULL,
Zip INT NOT NULL,
PRIMARY KEY (Person_ID)
);

CREATE TABLE EMPLOYEE (
P_ID CHAR(9) NOT NULL,
Responsibility VARCHAR(60),
PRIMARY KEY (P_ID),
FOREIGN KEY (P_ID) REFERENCES PERSON(Person_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE VOLUNTEER (
P_ID CHAR(9) NOT NULL,
Timeslots INT NOT NULL,
Weekdays INT NOT NULL,
PRIMARY KEY (P_ID),
FOREIGN KEY (P_ID) REFERENCES PERSON(Person_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE READER (
P_ID CHAR(9) NOT NULL,
C_name VARCHAR(60) NOT NULL,
C_state VARCHAR(15) NOT NULL,
PRIMARY KEY (P_ID),
FOREIGN KEY (P_ID) REFERENCES PERSON(Person_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE GUEST (
R_ID CHAR(9) NOT NULL,
F_name VARCHAR(60) NOT NULL,
L_name VARCHAR(60) NOT NULL,
PRIMARY KEY (R_ID, F_name, L_name),
FOREIGN KEY (R_ID) REFERENCES READER(P_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE HOST (
Host_ID INT NOT NULL,
E_ID CHAR(9),
V_ID CHAR(9),
PRIMARY KEY (Host_ID),
FOREIGN KEY (E_ID) REFERENCES EMPLOYEE(P_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE,
FOREIGN KEY (V_ID) REFERENCES VOLUNTEER(P_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);
```

```
CREATE TABLE ATTENDEE (
Attendee_ID INT NOT NULL,
G_f VARCHAR(60),
G_l VARCHAR(60),
R_ID CHAR(9),
PRIMARY KEY (Attendee_ID),
FOREIGN KEY (R_ID, G_f, G_l) REFERENCES GUEST(R_ID, F_name, L_name) ON DELETE
CASCADE INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (R_ID) REFERENCES READER(P_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE EVENT (
Event_ID INT NOT NULL,
Introduction VARCHAR(500) NOT NULL,
E_time TIMESTAMP NOT NULL,
PRIMARY KEY (Event_ID)
);

CREATE TABLE BORROWER (
Borrower_ID INT NOT NULL,
G_f VARCHAR(60),
G_l VARCHAR(60),
R_ID CHAR(9),
PRIMARY KEY (Borrower_ID),
FOREIGN KEY (R_ID, G_f, G_l) REFERENCES GUEST(R_ID, F_name, L_name) ON DELETE
CASCADE INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (R_ID) REFERENCES READER(P_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE BORROWS (
R_num CHAR(8) NOT NULL,
B_ID INT NOT NULL,
Borrow_date DATE NOT NULL,
Due_date DATE NOT NULL,
Return_date DATE,
PRIMARY KEY (R_num, B_ID, Borrow_date),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (B_ID) REFERENCES BORROWER(Borrower_ID) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE
);

CREATE TABLE RERSERVES (
R_num CHAR(8) NOT NULL,
B_ID INT NOT NULL,
Reserve_date DATE NOT NULL,
PRIMARY KEY(R_num, B_ID, Reserve_date),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (B_ID) REFERENCES BORROWER(Borrower_ID) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE
);

CREATE TABLE CREATES (
A_ID CHAR(5) NOT NULL,
R_num,
PRIMARY KEY(A_ID, R_num),
FOREIGN KEY (R_num) REFERENCES R_RESOURCE(Call_num) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (A_ID) REFERENCES AUTHOR(Author_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE FUNDS (
C_name VARCHAR(60) NOT NULL,
C_state VARCHAR(15) NOT NULL,
L_name VARCHAR(60) NOT NULL,
PRIMARY KEY (C_name, C_state, L_name),
FOREIGN KEY (C_name, C_state) REFERENCES CITY(Name, C_state) ON DELETE
CASCADE INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (L_name) REFERENCES LIBRARY(Name) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE HOSTS (
H_ID INT NOT NULL,
E_ID INT NOT NULL,
City VARCHAR(60) NOT NULL,
PRIMARY KEY (H_ID, E_ID),
FOREIGN KEY (H_ID) REFERENCES HOST(Host_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE,
FOREIGN KEY (E_ID) REFERENCES EVENT(Event_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE ATTENDS (
A_ID INT NOT NULL,
E_ID INT NOT NULL,
Rating INT NOT NULL,
PRIMARY KEY (A_ID, E_ID),
FOREIGN KEY (A_ID) REFERENCES ATTENDEE(Attendee_ID) ON DELETE CASCADE
INITIALLY DEFERRED DEFERRABLE,
FOREIGN KEY (E_ID) REFERENCES EVENT(Event_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE MAG_PAP_SUBJECT (
Mp_num CHAR(8) NOT NULL,
Subject VARCHAR(20) NOT NULL,
PRIMARY KEY (Mp_num, Subject),
FOREIGN KEY (Mp_num) REFERENCES MAG_PAP(R_num) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE AUTHOR_PHONE (
A_ID CHAR(5) NOT NULL,
Phone_num CHAR(12) NOT NULL,
PRIMARY KEY (A_ID, Phone_num),
FOREIGN KEY (A_ID) REFERENCES AUTHOR(Author_ID) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);
```

```
CREATE TABLE PUBLISHER_EMAIL (
P_name VARCHAR(30) NOT NULL,
Email VARCHAR(42) NOT NULL UNIQUE,
PRIMARY KEY(P_name, Email),
FOREIGN KEY (P_name) REFERENCES PUBLISHER(Name) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

CREATE TABLE PUBLISHER_PHONE (
P_name VARCHAR(30) NOT NULL,
Phone_num CHAR(12) NOT NULL UNIQUE,
PRIMARY KEY(P_name, Phone_num),
FOREIGN KEY (P_name) REFERENCES PUBLISHER(Name) ON DELETE CASCADE INITIALLY
DEFERRED DEFERRABLE
);

-- view1
CREATE VIEW BORROWED_RESOURCES AS
SELECT UNIQUE R_num, B_ID
FROM BORROWS;

-- view2
CREATE VIEW LIKED_EVENTS AS
SELECT Event_ID, Introduction
FROM EVENT
WHERE Event_ID IN (
  SELECT E_ID
  FROM ATTENDS
  GROUP BY E_ID
  HAVING AVG(Rating) > 60
);

-- view3
CREATE VIEW RESOURCES_OUT_ON_DATE AS
SELECT R_ID, R_num
FROM BORROWER, BORROWS
WHERE ('20-Nov-10' BETWEEN Borrow_date AND Return_date) AND (Borrower_ID =
B_ID);

-- insert some example data
-- example publishers
INSERT INTO PUBLISHER (Name, Webpage_url)
VALUES ('PENGUINS', 'penguins@penguingroup.boom');
INSERT INTO PUBLISHER (Name, Webpage_url)
VALUES ('Random House', 'randomhousepublishing@rh.com');

-- example authors
INSERT INTO AUTHOR (Author_ID, F_name, L_name)
VALUES ('00001', 'Austin', 'Steelman');
INSERT INTO AUTHOR (Author_ID, F_name, L_name)
VALUES ('12345', 'Kate', 'Beckensale');

-- example libraries
INSERT INTO LIBRARY (Name)
VALUES ('Mid continent Library');
INSERT INTO LIBRARY (Name)
VALUES ('Reading Rainbow Library');
```

```
-- example resources
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('abcdefgh', 'Game of Thrones', 'unavailable', 'for borrow', 'Random
House');
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('eeeeeee4', '1984', 'available', 'in library reading only', 'Random
House');
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('12345678', 'The Lord of The Rings', 'unavailable', 'in library
reading only', 'PENGUINS');
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('hunter42', 'Star Wars', 'available', 'for borrow', 'PENGUINS');
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('lolololo', 'Of Mice and Database Troubles', 'available', 'for
borrow', 'PENGUINS');

-- example resource locations
INSERT INTO RESOURCE_LOCATION (R_num, L_name)
VALUES ('abcdefgh', 'Mid continent Library');
INSERT INTO RESOURCE_LOCATION (R_num, L_name)
VALUES ('eeeeeee4', 'Reading Rainbow Library');
INSERT INTO RESOURCE_LOCATION (R_num, L_name)
VALUES ('12345678', 'Mid continent Library');
INSERT INTO RESOURCE_LOCATION (R_num, L_name)
VALUES ('hunter42', 'Reading Rainbow Library');

-- example books
INSERT INTO BOOK (R_num, Summary)
VALUES ('abcdefgh', 'Politics, Swords, Whitewalkers, and Drama');
INSERT INTO BOOK (R_num, Summary)
VALUES ('eeeeeee4', 'BIG BROTHER IS WATCHING YOU');

-- example videos
INSERT INTO VIDEO (R_num, V_format)
VALUES ('12345678', 'DVD');

-- example magzines
INSERT INTO MAG_PAP (R_num, Publishing_cycle)
VALUES ('hunter42', 'Weekly');

-- example people
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('gxs112030', 'Gary', 'Steelman', 23, '2200 Waterview Pkwy', 'Apt
2134', 'Richardson', 'TX', 75080);
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('ams000000', 'Annie', 'Slaughter', 70, '1408 Madison 1400', NULL,
'Huntsville', 'AR', 12345);
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
```

```
VALUES ('vuu111111', 'Vanessa', 'Underwood', 0, '1234 Somewhere Rd', NULL,
'Cape...Geraudeau', 'MO', 65236);
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('red222222', 'Randal', 'Derpystuff', 85, '1234 Somewhere Rd', NULL,
'Springfield', 'MO', 65298);
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('ams123456', 'Austin', 'Steelman', 38, '11th St', NULL, 'Blue
Springs', 'MO', 64015);

-- example volunteers
INSERT INTO VOLUNTEER (P_ID, Timeslots, Weekdays)
VALUES ('gxs112030', 0, 5);
INSERT INTO VOLUNTEER (P_ID, Timeslots, Weekdays)
VALUES ('red222222', 15, 25);

-- example cities
INSERT INTO CITY (Name, C_state)
VALUES ('Richardson', 'TX');
INSERT INTO CITY (Name, C_state)
VALUES ('Springfield', 'MO');

-- example READERS
INSERT INTO READER (P_ID, C_name, C_state)
VALUES ('gxs112030', 'Richardson', 'TX');
INSERT INTO READER (P_ID, C_name, C_state)
VALUES ('red222222', 'Springfield', 'MO');

-- example employees
INSERT INTO EMPLOYEE (P_ID, Responsibility)
VALUES ('ams000000', 'Librarian');
INSERT INTO EMPLOYEE (P_ID, Responsibility)
VALUES ('gxs112030', 'Librarian');
INSERT INTO EMPLOYEE (P_ID, Responsibility)
VALUES ('ams123456', 'Entertainment Prodigy');

-- example guests
INSERT INTO GUEST (F_name, L_name, R_ID)
VALUES ('Katy', 'Steelman', 'gxs112030');
INSERT INTO GUEST (F_name, L_name, R_ID)
VALUES ('John', 'Slaughter', 'ams000000');

-- example events
INSERT INTO EVENT (Event_ID, Introduction, E_time)
VALUES(15, 'How to speed read', to_timestamp('2010/10/10:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'));

-- example hosts
INSERT INTO HOST (Host_ID, E_ID, V_ID)
VALUES (1, 'ams123456', NULL);

-- example attendees
INSERT INTO ATTENDEE (Attendee_ID, G_f, G_l, R_ID)
VALUES (1, 'Katy', 'Steelman', 'gxs112030');
INSERT INTO ATTENDEE (Attendee_ID, G_f, G_l, R_ID)
VALUES (2, NULL, NULL, 'red222222');
```

```
-- example borrowers
INSERT INTO BORROWER (Borrower_ID, G_f, G_l, R_ID)
VALUES (1, 'Katy', 'Steelman', 'gxs112030');
INSERT INTO BORROWER (Borrower_ID, G_f, G_l, R_ID)
VALUES (2, NULL, NULL, 'red222222');

-- example borrows
INSERT INTO BORROWS (R_num, B_ID, Borrow_date, Due_date, Return_date)
VALUES ('abcdefgh', 1, '19-Nov-10', '29-Nov-10', NULL);
INSERT INTO BORROWS (R_num, B_ID, Borrow_date, Due_date, Return_date)
VALUES ('eeeeeee4', 2, '18-Nov-10', '30-Nov-10', '28-Nov-10');
INSERT INTO BORROWS (R_num, B_ID, Borrow_date, Due_date, Return_date)
VALUES ('eeeeeee4', 2, '18-Nov-10', '23-Nov-10', '28-Nov-10');

-- example reserves

-- example creates
INSERT INTO CREATES (A_ID, R_num)
VALUES ('00001', 'abcdefgh');
INSERT INTO CREATES (A_ID, R_num)
VALUES ('12345', '12345678');

-- example funds
INSERT INTO FUNDS (C_name, C_state, L_name)
VALUES ('Richardson', 'TX', 'Reading Rainbow Library');
INSERT INTO FUNDS (C_name, C_state, L_name)
VALUES ('Richardson', 'TX', 'Mid continent Library');

-- example attends
INSERT INTO ATTENDS (A_ID, E_ID, Rating)
VALUES (1, 15, 62);
INSERT INTO ATTENDS (A_ID, E_ID, Rating)
VALUES (2, 15, 59);

-- example mag subjects
INSERT INTO MAG_PAP_SUBJECT(Mp_num, Subject)
VALUES( 'hunter42', 'Fashion' );
INSERT INTO MAG_PAP_SUBJECT(Mp_num, Subject)
VALUES( 'hunter42', 'Explosions' );

-- example author phones
-- n/a for answering phase 3 queries

-- example publisher emails
-- n/a for answering phase 3 queries

-- example plublisher phones
-- n/a for answering phase 3 queries

-- perform requsted queries
-- n/a for answering phase 3 queries

-- 1
-- requires helen to be a person first
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
```

```
VALUES ('hxl000000', 'Helen', 'Liou', 45, '123 Library Rd', NULL,
'Readersville', 'TX', 75080);
INSERT INTO VOLUNTEER (P_ID, Timeslots, Weekdays)
VALUES ('hxl000000', 12, 6);

-- 2
SELECT R_num
FROM BORROWS
GROUP BY R_num
HAVING COUNT(R_num) > 10;

-- 3
SELECT Person_ID, City
FROM PERSON
WHERE City IN (
  SELECT UNIQUE City
  FROM FUNDS
  INTERSECT
  SELECT UNIQUE City
  FROM PERSON
  WHERE Person_ID IN (
    SELECT UNIQUE P_ID
    FROM VOLUNTEER
  )
);

-- 4
SELECT R_ID, R_num
FROM BORROWER, BORROWS
WHERE (Due_date = '19-Sep-10') AND (Borrower_ID = B_ID);

-- 5
SELECT Introduction
FROM (
  SELECT Introduction, COUNT(Introduction) as "count"
  FROM LIKED_EVENTS
  GROUP BY Introduction
  ORDER BY "count" DESC
  )
WHERE ROWNUM <= 1;

-- 6
SELECT Person_ID, F_name, L_name
FROM PERSON
WHERE Person_ID IN (
  SELECT P_ID
  FROM VOLUNTEER
  WHERE Weekdays = 6
);

-- 7
SELECT F_name, L_name, R_num
FROM BORROWER, BORROWS, PERSON
WHERE
  ( Return_date > Due_date
    OR ( Return_date = NULL AND Due_date < CURRENT_DATE )
  )
  AND (Borrower_ID = B_ID)
  AND (Person_ID = R_ID);

-- 8
SELECT P_name
FROM (
  SELECT P_name, COUNT(P_name) as "count"
  FROM R_RESOURCE
  GROUP BY P_name
  ORDER BY "count" DESC
  )
WHERE ROWNUM <= 1;

-- 9
-- must have person as a reader to check out stuff
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('mxl123456', 'Michelle', 'Luigi', 23, 'Rainbow Rd', NULL, 'Yoshi
Island', 'MA', 10000);
INSERT INTO READER (P_ID, C_name, C_state)
VALUES ('mxl123456', 'Richardson', 'TX');
INSERT INTO BORROWER (Borrower_ID, G_f, G_l, R_ID)
VALUES (3, NULL, NULL, 'mxl123456');
INSERT INTO BORROWS (R_num, B_ID, Borrow_date, Due_date, Return_date)
VALUES ('12345678', 3, '20-Nov-10', '23-Nov-10', '23-Nov-10');

-- 10
SELECT F_name, L_name
FROM AUTHOR
WHERE Author_ID IN (
  SELECT A_ID FROM (
    SELECT A_ID, COUNT(A_ID) as "count"
    FROM CREATES
    GROUP BY A_ID
    ORDER BY "count" DESC
  )
  WHERE ROWNUM <= 1
);

-- 11
-- must first have a person and resource with the specified data
-- and person must be a reader
INSERT INTO R_RESOURCE (Call_num, Name, Borrow_status, Reading_status,
P_name)
VALUES ('QA0001', 'Databases Example 11', 'unavailable', 'for borrow',
'Random House');
INSERT INTO PERSON (Person_ID, F_name, L_name, Age, Rd, Ap, City, C_state,
Zip)
VALUES ('cld089000', 'Celene', 'Donatello', 23, '123 Awesome Rd', NULL,
'Yupper', 'CA', 54298);
INSERT INTO READER (P_ID, C_name, C_state)
VALUES ('cld089000', 'Richardson', 'TX');

-- 12
SELECT Call_num
FROM R_RESOURCE
WHERE Call_num IN (
  SELECT UNIQUE R_num
  FROM MAG_PAP
  WHERE R_num IN (
    SELECT Mp_num
    FROM MAG_PAP_SUBJECT
    WHERE Subject LIKE '%Fashion%'
  )
);


-- 13
SELECT E.Event_ID AS "Event ID", H2.E_ID AS "Employee ID",
       H2.V_ID AS "Volunteer ID", A2.G_f AS "Guest Fname",
       A2.G_l AS "Guest Lname", A2.R_ID AS "Reader ID"
FROM EVENT E, HOSTS H1, HOST H2, ATTENDS A1, ATTENDEE A2
WHERE E.Event_ID = H1.E_ID
  AND E.Event_ID = A1.E_ID
  AND H1.H_ID = H2.Host_ID
  AND A1.A_ID = A2.Attendee_ID
  AND E.E_time = to_timestamp('2010/10/10:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam');

-- 14
SELECT Person_ID, L_name
FROM PERSON
WHERE Person_ID NOT IN (
  SELECT UNIQUE R_ID
  FROM GUEST
);

-- 15


-- 16
SELECT F_name, L_name
FROM PERSON
WHERE Person_ID IN (
  SELECT P_ID
  FROM READER
  WHERE P_ID IN (
    SELECT R_ID
    FROM ATTENDEE A
    WHERE NOT EXISTS (
      SELECT Introduction
      FROM EVENT E1
        MINUS
      SELECT Introduction
      FROM EVENT E2, ATTENDS A1, ATTENDEE A2
      WHERE A_ID = Attendee_ID
        AND E2.Event_ID = A1.E_ID
        AND A2.Attendee_ID = A1.A_ID
    )
  )
);

-- confirm tables show appropriate columns and data
SELECT * FROM LIBRARY ;
SELECT * FROM PUBLISHER ;
SELECT * FROM R_RESOURCE ;
SELECT * FROM RESOURCE_LOCATION ;
SELECT * FROM BOOK ;
SELECT * FROM VIDEO ;
SELECT * FROM MAG_PAP ;
SELECT * FROM AUTHOR ;
SELECT * FROM CITY ;
SELECT * FROM PERSON ;
SELECT * FROM EMPLOYEE ;
SELECT * FROM VOLUNTEER ;
SELECT * FROM READER ;
SELECT * FROM GUEST ;
SELECT * FROM HOST ;
SELECT * FROM ATTENDEE ;
SELECT * FROM EVENT ;
SELECT * FROM BORROWER ;
SELECT * FROM BORROWS ;
SELECT * FROM RERSERVES ;
SELECT * FROM CREATES ;
SELECT * FROM FUNDS ;
SELECT * FROM HOSTS ;
SELECT * FROM ATTENDS ;
SELECT * FROM MAG_PAP_SUBJECT ;
SELECT * FROM AUTHOR_PHONE ;
SELECT * FROM PUBLISHER_EMAIL ;
SELECT * FROM PUBLISHER_PHONE ;

-- confirm views show appropriate columns and data
SELECT * FROM BORROWED_RESOURCES;
SELECT * FROM LIKED_EVENTS;
SELECT * FROM RESOURCES_OUT_ON_DATE;

-- DROP THE TABLES
-- This is done so that this script can be run again next time without a
-- ton of errors.
DROP TABLE LIBRARY CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER CASCADE CONSTRAINTS;
DROP TABLE R_RESOURCE CASCADE CONSTRAINTS;
DROP TABLE RESOURCE_LOCATION CASCADE CONSTRAINTS;
DROP TABLE BOOK CASCADE CONSTRAINTS;
DROP TABLE VIDEO CASCADE CONSTRAINTS;
DROP TABLE MAG_PAP CASCADE CONSTRAINTS;
DROP TABLE AUTHOR CASCADE CONSTRAINTS;
DROP TABLE CITY CASCADE CONSTRAINTS;
DROP TABLE PERSON CASCADE CONSTRAINTS;
DROP TABLE EMPLOYEE CASCADE CONSTRAINTS;
DROP TABLE VOLUNTEER CASCADE CONSTRAINTS;
DROP TABLE READER CASCADE CONSTRAINTS;
DROP TABLE GUEST CASCADE CONSTRAINTS;
DROP TABLE HOST CASCADE CONSTRAINTS;
DROP TABLE ATTENDEE CASCADE CONSTRAINTS;
DROP TABLE EVENT CASCADE CONSTRAINTS;
DROP TABLE BORROWER CASCADE CONSTRAINTS;
DROP TABLE BORROWS CASCADE CONSTRAINTS;
DROP TABLE RERSERVES CASCADE CONSTRAINTS;
DROP TABLE CREATES CASCADE CONSTRAINTS;
DROP TABLE FUNDS CASCADE CONSTRAINTS;
DROP TABLE HOSTS CASCADE CONSTRAINTS;
DROP TABLE ATTENDS CASCADE CONSTRAINTS;
DROP TABLE MAG_PAP_SUBJECT CASCADE CONSTRAINTS;
DROP TABLE AUTHOR_PHONE CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER_EMAIL CASCADE CONSTRAINTS;
DROP TABLE PUBLISHER_PHONE CASCADE CONSTRAINTS;

DROP VIEW BORROWED_RESOURCES;
DROP VIEW LIKED_EVENTS;
DROP VIEW RESOURCES_OUT_ON_DATE;

spool off
```

Conclusion

In this report I have given the project description and each of the critical components for each phase of the design process for implementing an Oracle DBMS to realize the customer’s system. I give the EER diagram, relational schemas, functional dependencies, and an SQL script showing the database in action.

In the future I could implement an application interface to allow the system to be interactive.