Database Design Project

Phase3 – Implementation

Gary Steelman | gxs112030@utdallas.edu

Contents

Introduction

This report comprises four main sections:

1. Relational Schema UPDATED – In this section I show the updated relational schema from phase 2. This contains updates to normalize the schemas to 3NF.
2. Dependency Diagram – In this section I illustrate the dependency diagram for each relational schema in the database.
3. Database Implementation via SQL – I implement the relational schemas from section 1 and the data type constraints from phase 2 via SQL. This section is further broken into multiple subsections.
    a. Table Creation SQL Code.
    b. A sample of database instance.
    c. Answers for questions in the instructions including view creation and queries.
4. Conclusion – In this section I summarize the report.

Relational Schemas [UPDATED]

Phase three instructions require all relational schemas to be in third normal form (3NF). Accordingly, I have updated the relational schemas from phase two to the following. The changes include extracting the resource location from the RESOURCE table. Since a RESOURCE may exist in multiple copies and in multiple libraries, this makes sense. The Final version of the relational schema is shown below. The new FK constraints have been listed below this table.

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

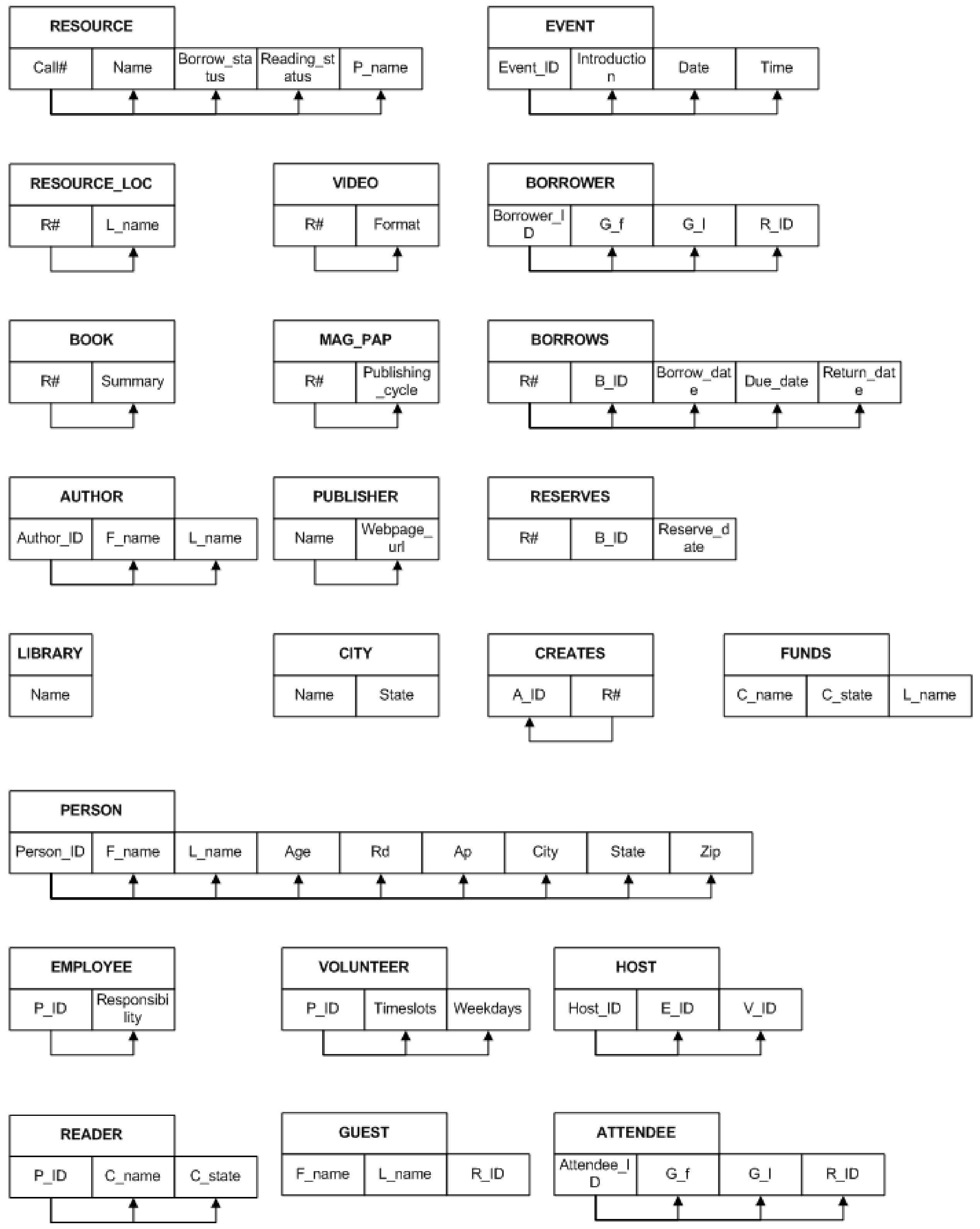
RELANTION	**Attributes**	**Data type and constraints**
RESOURCE	Call#	string, 8 chars, non-null, unique
	Name	string <= 60 chars
	Borrow_status	string <= 20 chars; “available” or “unavailable”
	Reading_status	string <= 20 chars; “for borrow” or “in library reading only”
	P_name	string <= 60 chars
RESOURCE_LOCATION	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

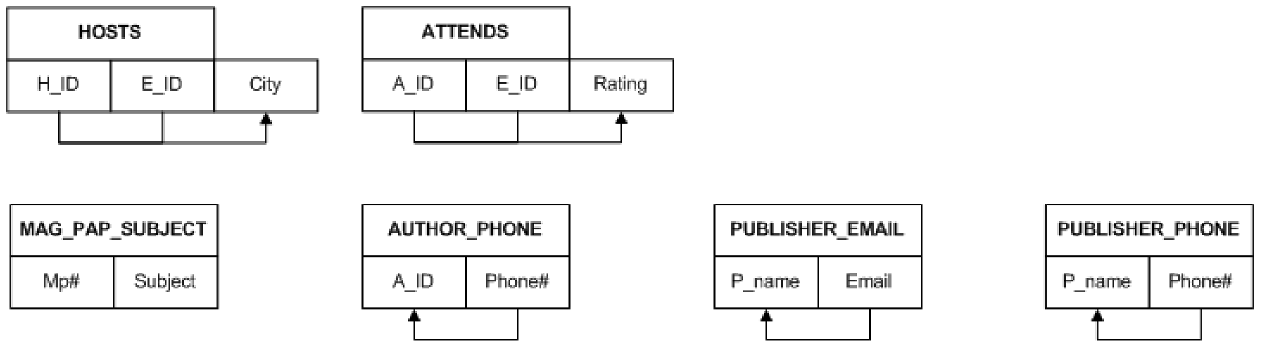
PUBLISHER	Name	string <= 30 chars, non-null, unique
	Webpage_url	string <= 60 chars
LIBRARY	Name	string <= 60 chars, non-null, unique
CITY	Name	string <= 60 chars, non-null
	State	string <= 15 chars, non-null
	{Name, State}	unique
PERSON	Person_ID	string, 9 chars; {f + c + l + xxxxxx}, non-null, unique
	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

Dependency Diagrams

Below are the dependency diagrams for each relation. Arrows point from the set of attributes X to the dependent set of attributes A.


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



Database Implementation via SQL

The database implementation uses Oracle SQL. The script included with this report submission shows the entire execution for creating tables, populating them with data, creating views, answering queries, and dropping the tables.

Table Creation

Tables: See script lines 45-269.

Views: See script lines 272-291.

Sample Database State

See script lines 295-444.

Queries for Questions in Instructions

See script lines 460-620.

These queries are numbered as "-- #" where # is the corresponding number from the instructions.

Conclusion

Summary

In this report I normalized the relational schemas from phase 1 into third normal form. I also recorded the discovered functional dependencies. Then I created an SQL script to create the tables, insert some data, perform the requested queries, and then drop all the tables.

Future Work

This report implemented the database via a simple SQL script. In the future I will create a nice front-end with canned queries for interacting with the database. For further questions contact me at the email address on the title page.