Database Design Project

Phase 2 – Relational Schema Design

Gary Steelman | gxs112030@utdallas.edu

Contents

Introduction

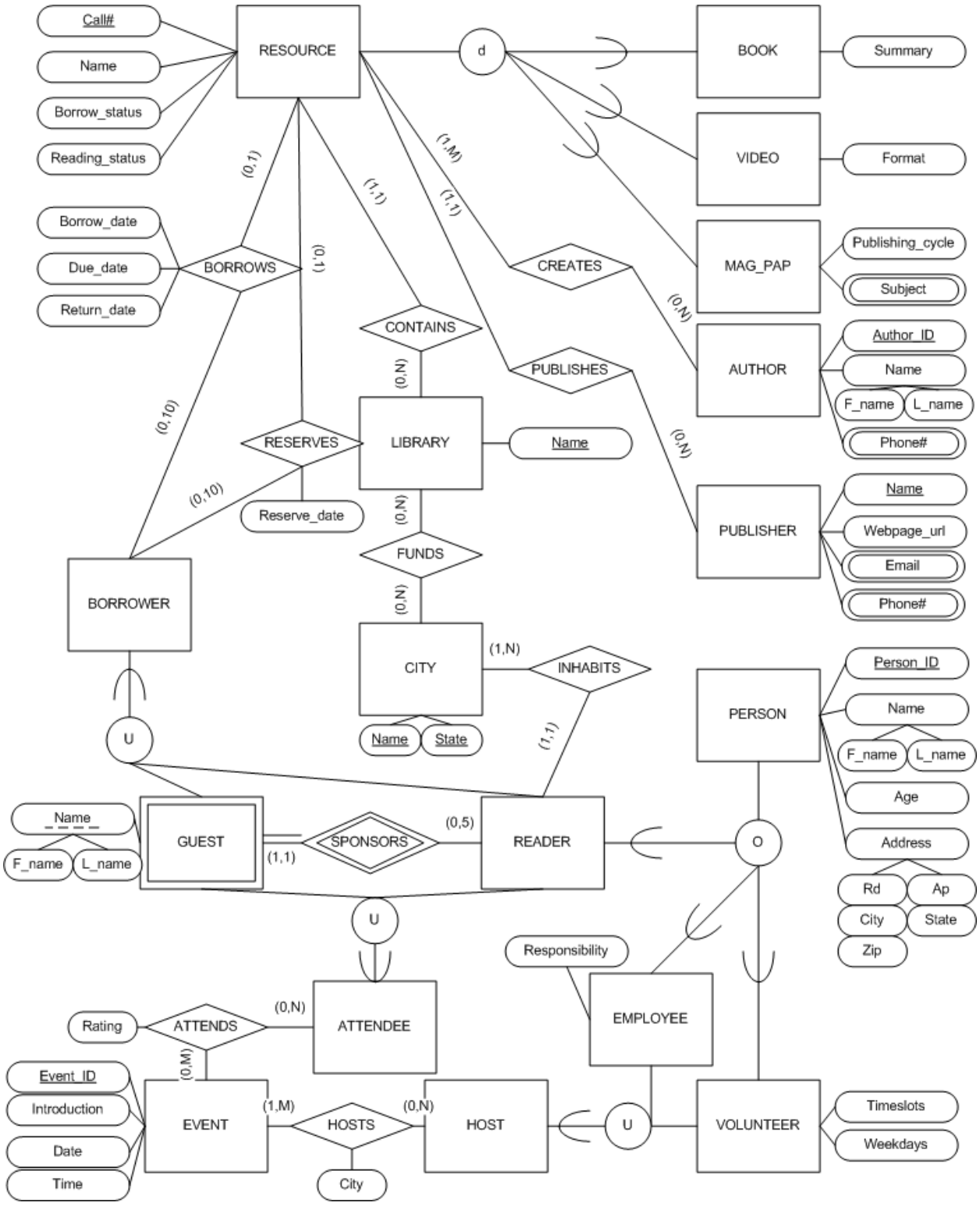
This report comprises four main sections:

1. EER Diagram UPDATED – In this section I show the updated EER diagram from phase 1. This has fixes from the phase 1 submission and from difficulties implementing the relational schema.
2. EER Diagram to Relational Schemas – In this section I step through the book's seven-step algorithm for mapping an EER diagram to a relational schema.
3. Documentation for Relation Schemas – In this section I elaborate on the relational schemas and introduce data type constraints, data type formats, etc.
4. Conclusion – In this section I summarize the report. Assumptions, Limitations, and Explanations

Enhanced Entity Relationship (EER) Diagram [UPDATED]

The updates are:

- Changed the line between SPONSORS and GUEST to a double line.


Call#
Name
Borrow_status
Reading_status
RESOURCE
d
BOOK
Summary
VIDEO
Format
MAG_PAP
Publishing_cycle
Subject
AUTHOR
Author_ID
Name
F_name
L_name
Phone#
PUBLISHER
Name
Webpage_url
Email
Phone#
Borrow_date
Due_date
Return_date
BORROWS
CREATES
PUBLISHES
CONTAINS
RESERVES
Reserve_date
LIBRARY
Name
FUNDS
CITY
Name
State
INHABITS
BORROWER
U
PERSON
Person_ID
Name
F_name
L_name
Age
Address
Rd
Ap
City
State
Zip
O
Name
F_name
L_name
GUEST
SPONSORS
READER
Responsibility
EMPLOYEE
Rating
ATTENDS
ATTENDEE
Event_ID
Introduction
Date
Time
EVENT
HOSTS
HOST
City
VOLUNTEER
Timeslots
Weekdays
(0,1)
(1,1)
(1,M)
(0,N)
(0,10)
(1,N)
(0,5)
(0,M)

EER Diagram to Relational Schemas

The text book uses a seven-step algorithm to completely map an EER diagram to a set of relational schemas. The following sections follow this algorithm step by step. The results of each step are accumulated and shown in the Final Relational Schemas section at the end. Referential integrity constraints are listed in the summary section.

Mapping Regular Entity Types, Specializations, and Unions

In this section I map regular entity types, specializations, and union types. Any multi-valued attributes are left for a later step in the algorithm.

RESOURCE and its subclasses – each subclass gets its own relation

- RESOURCE(Call#, Name, Borrow_status, Reading_status)
- BOOK(R#, Summary)
- VIDEO(R#, Format)
- MAG_PAP(R#, Publishing_cycle)

AUTHOR(Author_ID, F_name, L_name)

PUBLISHER(Name, Webpage_url)

LIBRARY(Name)

CITY(Name, State)

PERSON and its subclasses – each subclass gets its own relation. The READER type has a local relationship and the EMPLOYEE and VOLUNTEER types are involved in a union relationship. It's best to separate these into their own relations.

- PERSON(Person_ID, F_name, L_name, Age, Rd, Ap, City, State, Zip)
- READER(P_ID)
- EMPLOYEE(P_ID, Responsibility)
- VOLUNTEER(P_ID, Timeslots, Weekdays)

HOST is a union of the EMPLOYEE and VOLUNTEER types so it must have a surrogate key.

- HOST(Host_ID, E_ID, V_ID)

ATTENDEE is a union of the GUEST and READER types so it must have a surrogate key.

- ATTENDEE(Attendee_ID, G_f, G_l, R_ID)

EVENT(Event_ID, Introduction, Date, Time)

BORROWER is a union of the GUEST and READER types so it must have a surrogate key.

- BORROWER(Borrower_ID, G_f, G_l, R_ID)

Mapping Weak Entity Types

The single weak entity type is a basic mapping. It uses a Person_ID in conjunction with its F_name and L_name to uniquely identify GUESTs.

GUEST(F_name, L_name, R_ID)

Mapping Binary 1:1 Relationship Types

No binary 1:1 relationship types exist in the EER diagram.

Mapping Binary 1:N Relationship Types

It usually is a good idea to merge 1:N relationship with the 1 side of the relationship to reduce the number of relations in the database.

PUBLISHES can be merged with the RESOURCE type since each RESOURCE may have only one PUBLISHER. RESOURCE updates to

- RESOURCE(Call#, Name, Borrow_status, Reading_status, P_name)

CONTAINS can be merged with RESOURCE since each RESOURCE may be contained in only a single library at a time. RESOURCE updates to

- RESOURCE(Call#, Name, Borrow_status, Reading_status, P_name, L_name)

BORROWS can be merged with RESOURCE since each RESOURCE may be borrowed by only a single BORROWER at a time. However, since BORROWS has local attributes, it is better to make it another relation

- BORROWS(R#, B_ID, Borrow_date, Due_date, Return_date)

RESERVES can be merged with RESOURCE since each RESOURCE may be reserved by only a single BORROWER at a time. However since RESERVES has local attributes, it is better to make it another relation

- RESERVES(R#, B_ID, Reserve_date)

INHABITS can be merged with READER since each READER may only inhabit a single city at a time. READER updates to

- READER(P_ID, C_name, C_state)

SPONSORS is the identifying relationship for a GUEST and READER. It can be merged with GUEST since each GUEST must be sponsored by only one READER. No updates are necessary to the GUEST type.

Mapping Binary N:M Relationship Types

Each N:M relationship type gets its own relation.

CREATES(A_ID, R#)

FUNDS(C_name, C_state, L_name)

HOSTS(H_ID, E_ID, City)

ATTENDS(A_ID, E_ID, Rating)

Mapping Multi-Valued Attributes

Each multi-valued attributes becomes its own relation.

MAG_PAP_SUBJECT(Mp#, Subject)

AUTHOR_PHONE(A_ID, Phone#)

PUBLISHER_EMAIL(P_name, Email)

PUBLISHER_PHONE(P_name, Phone#)

Mapping of N-Ary Relationship Types

No non-binary relationship types exist in the EER diagram.

Final Relational Schemas

RELATION	Referential (Integrity) Constraints
RESOURCE(Call#, Name, Borrow_status, Reading_status, P_name, L_name)	P_name -> PUBLISHER.Name 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

Documentation for Relational Schemas

In this section I elaborate on the relational schemas created in the previous section. I introduce data types and constraints for attributes of relations. There are some database-wide rules for data types as follows:

- Dates are all in form “MM-DD-YYYY” as a string of characters.
- Times are all in form “HH:MM:SS” as a string of characters.
- Phone numbers are all in “xxx-xxx-xxxx” as a string of characters.
- IDs are numbers generated using the auto increment feature of a database system.
- Person_ID is generated by using the first letter of the F_name, a random character, the first letter of the L_name, and a 6 digit random integer.

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
	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

Conclusion

Summary

In this report I updated the EER diagram from phase 1 and mapped it to relational schemas using the book’s seven-step algorithm. I also elaborated on the data types for the relational schemas.

Future Work

This report derived the relational model of the ABC Library database system from the EER diagram. The next step is to create SQL queries to create the database in a real system. For further questions contact me at the email address on the title page.