Database Design Project

Phase3 – Implementation

Gary Steelman | [gxs112030@utdallas.edu](mailto:gxs112030@utdallas.edu)

Contents

[Introduction 2](#_Toc341653610)

[Relational Schemas [UPDATED] 2](#_Toc341653611)

[Dependency Diagrams 5](#_Toc341653612)

[Database Implementation via SQL 7](#_Toc341653613)

[Table Creation 7](#_Toc341653614)

[Sample Database State 7](#_Toc341653615)

[Queries for Questions in Instructions 7](#_Toc341653616)

[Conclusion 7](#_Toc341653617)

[Summary 7](#_Toc341653618)

[Future Work 7](#_Toc341653619)

# 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.
   1. Table Creation SQL Code.
   2. A sample of database instance.
   3. Answers for questions in the instructions including view creation and queries.
4. Conclusion – In this section I summarize the report.

# Relational Schemas [UPDATED]

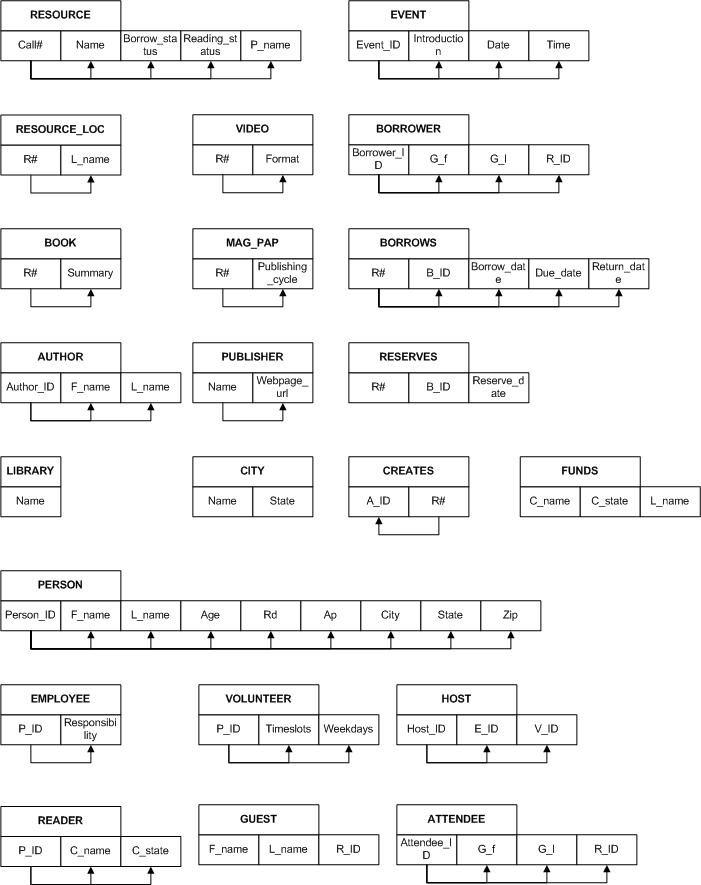
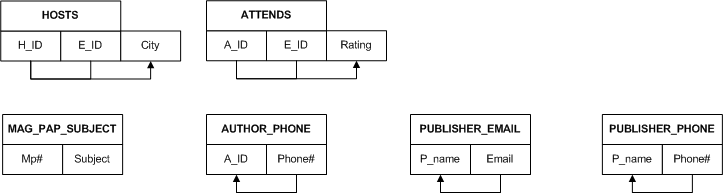
Phase three instructions require all relational schemas to be in third normal form (3NF). Accordingly, I have updated the relational schemas from phase two to the following. The changes include extracting the resource location from the RESOURCE table. Since a RESOURCE may exist in multiple copies and 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.

# 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.