



### FIT9132 Introduction to Databases

#### 2021 Semester 2

#### Assignment 2A - Monash Library Services (MLib)

#### Assignment weighting 10%

---

The local Monash Municipality maintains several libraries for its residents across the municipality.

For each branch, MLib assigns a branch code (an incremental number for each branch with the first branch using a code of 10). The branch name, address and contact phone number are also recorded. MLib also records the number of books registered at each branch. Each branch has a unique phone number and is assigned one manager. Due to the small size of some of the branches, a particular manager may manage several branches. Each manager is assigned a manager id. MLib Services record a manager's name and contact phone number.

MLib maintains records of current loans of books to borrowers.

Each borrower is identified by a unique borrower number. When a borrower first registers to borrow books, the branch where they register is recorded as their home branch. The name and address of each borrower is held so that communications, such as overdue loan reminders, can be sent when necessary.

The information held about a book includes its Dewey Decimal call no - this call no is used to identify a particular book's details. The title, classification (Reference or Fiction), number of pages, publication year and edition, if applicable, are recorded.

Branches hold copies of a book - each copy is the property of a particular branch and is identified by the branch number and a branch assigned local id number (these id numbers are repeated at each branch). Some book copies are placed on counter reserve, and are not available for loan - they may only be used in the library. A flag is added to a book copy to indicate if it is on counter reserve or not. There may also be other copies of the same title which are available for normal loan.

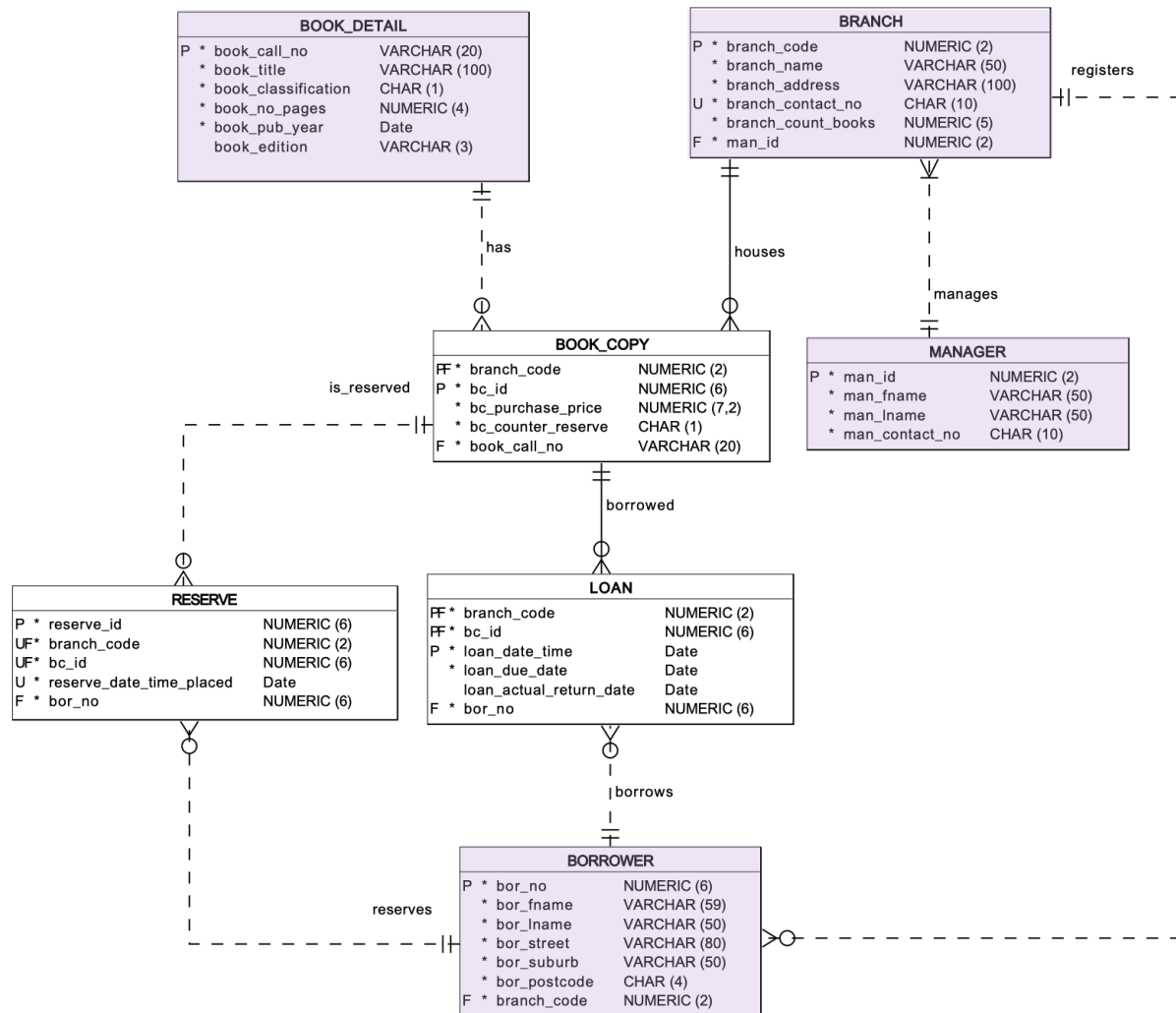
When a book copy is borrowed (goes out on loan), the return due date is recorded. A record of all loans which take place is maintained. When a book is returned from a loan its actual return date is recorded. Each book copy can be loaned for 14 days and then must be renewed to avoid a fine. For example, a book borrowed on the 2nd August 2021 at 10:00 AM will be due on the 16th August 2021.

Borrowers may reserve books currently out on loan. A reservation is assigned a reservation id (these id numbers are unique across all branches). The date and time on which the reservation was placed are recorded. A given book may be reserved by several borrowers, the book is made available based on the order in which the reserve was placed by the borrower.

When a borrower returns a book, they may, if they wish, renew their loan and take the book out for a further loan period provided the book has not been reserved by another borrower, the renewal is simply treated as a new loan for that borrower. Books must be returned to the branch from which they were borrowed (the branch owning the book copy).

A model to represent this system has been developed:

NOTE: you have been provided with the create table statements and sample data for the purple highlighted relations



You have been supplied with an SQL script **mllib\_initialSchemaInsert.sql** which partially implements the MLib model (it creates the tables coloured in purple on the above diagram and inserts some initial *sample* data). This file **may NOT be altered (edited) in any way**.

## Steps for working on Assignment 2A

1. Download the Assignment 2A Required Files zip archive from Moodle
2. Place the zip archive in your local (MoVE or local HDD) repository in the folder /Assignments/Ass2A
3. Unzip the archive, examine the files (i.e. read carefully through the files and understand their content). Then add, commit and push them to the FITGitLab server.
4. Write your answer for each task in its respective file (e.g. write your answer for task 1.1 in T1-ml-schm.sql and so on).
5. Save, add, commit and push the file/s **regularly** while you are working on the assignment
6. Finally, when you have completed all tasks, upload all required files from your local repository to Moodle (if you are using MoVE you will need to download them to your local HDD first - do not attempt to upload from MoVE). Check that the files you have uploaded are the correct files (download them from Moodle into a temporary folder and check they are correct). After you are sure they are correct, submit your assignment.

Note that the final SQL scripts you submit **MUST NOT contain SPOOL or ECHO commands** (you may include them as you work but **must** remove them before submission). Please carefully read the "CRITERIA FOR MARKING" on page 10 of this document.

### TASK 1: DDL [23 + 2 = 25 mks]

For this task you are required to complete the following:

- 1.1 Add to **T1-ml-schm.sql**, the CREATE TABLE and CONSTRAINT definitions which are missing from the supplied partial schema script in the positions indicated by the comments in the script. Appendix A provides details of the meaning of the attributes in the missing three tables. You **MUST** use exactly the same relation and attribute names shown in the data model above to name tables and attributes which you add. The schema file for 1.1 *must be hand coded, not generated in any manner*.

To test your code you will need to first run the provided script mlib\_initialSchemaInsert.sql to create the other required tables.

- 1.2 Add the full set of DROP TABLE statements to **T1-ml-drop.sql** in the location indicated by the 'Add Drop table statements'... comment. This should drop all of the tables created by the supplied partial schema script and your answer to 1.1 it does not need to drop any subsequent tables you may create for task 3. In completing this section you **must not use the CASCADE CONSTRAINTS clause** as part of your DROP TABLE statement (you must however include the PURGE clause). For example a drop table statement will have the form

DROP TABLE BORROWER PURGE;

*In a script you can run a section of the script by highlighting the lines you wish to run and selecting the run button. If at any stage your tables are corrupted during working on this assignment you simply need to run your drop commands from above (T1-ml-drop.sql) and then rerun mlib\_initialSchemaInsert and your extra definitions that you added above (T1-ml-schm.sql) .*

Before proceeding with Task 2, you must ensure you have run the file `mllib_initialSchemaInsert.sql` (which **must not be edited in any way**) followed by the extra definitions that you added in 1.1 above.

## TASK 2: DML ( 50 marks):

- (a) Load the BOOK\_COPY, LOAN and RESERVE tables with **your own test data** using the supplied **T2-ml-insert.sql** file script file, and SQL commands which will insert as *a minimum*, the following sample data -
- (i) 10 BOOK\_COPIES (you may select any reasonable purchase price)
    - representing at least 3 different book details,
    - distributed across 3 different libraries,
    - with at least 1 library holding multiple copies of a book, and
    - at least one copy on counter reserve
  - (ii) 10 LOANS
    - 8 of which must have been completed, with at least one of these returned late and one still due,
    - borrowed from at least 2 different libraries, and
    - by at least 3 different borrowers
  - (iii) 2 RESERVE entries (note MLib deletes reserve entries after they have been fulfilled ie. the user has borrowed the book)

In adding this data you must ensure that the test data thoroughly tests the model as supplied, so as to ensure your schema is correct.

Your inserted data must conform to the following rules:

- (i) You may treat all of the data that you add as a single transaction since you are setting up the initial test state for the database.
- (ii) The primary key values for this data should be hardcoded values (i.e. **NOT** make use of sequences) and must consist of values below 100.
- (iii) Dates used must be chosen between the 1st June 2021 and 13th September 2021 (inclusive). The data added must be sensible eg. the return of a loan must be after the loan was taken out.

For this task **ONLY**, Task 2(a), you may look up and include values for the loaded tables/data directly where required. However, if you wish, you can still use SQL to get any non-key values.

**In carrying out this task you must not modify any data or add any further data to the tables which were populated by the `mllib_partSchema.sql` script.**

**[25 marks]**

For all subsequent questions (Task 2(b) onwards) **you are NOT permitted to:**

- manually lookup a value in the database, obtain its primary key or the highest/lowest value in a column,
- manually calculate values external to the database, e.g. on a calculator and then use such values in your answers. **Any necessary calculations must be carried out as part of your SQL code**, or
- assume any particular contents in the database - rows in a table are potentially in a constant state of change

Your answers must recognise the fact that you have been given, with the supplied insert file, only a very small sample snapshot of a multiuser database, as such you must operate on the basis that there will be **more data in all of the tables of the database than you have been given. Your answers must work regardless of the extra quantity of this extra "real" data and the fact that multiple users will be operating in the tables at the same time. You must take this aspect into consideration when writing SQL statements.**

**You must ONLY use the data as provided in the text of the questions. Failure to adhere to this requirement will result in a mark of 0 for the relevant question.**

For the following tasks, **your SQL must correctly manage transactions and use sequences to generate new primary keys for numeric primary key values** (under no circumstances may a new primary key value be hardcoded as a number or value).

- (b) Your answers for these tasks (2b) must be placed in the supplied SQL Script **T2-ml-dm.sql**

For this task you are required to complete the following sub-tasks in the same order they have mentioned:

- (i) MLib has just purchased its first 3 copies of a recently released edition of a new reference book.

Some of the details of the new book are:

Call Number: 005.74 C824C  
Title: Database Systems: Design, Implementation, and Management  
Publication Year: 2019  
Edition: 13  
Number of Pages: 793  
Price: \$120 per copy

Each of the 3 MLib branches listed below will get a single copy of this book, the book will be available for borrowing (ie. not on counter reserve) at each branch:

Clayton (Ph: 0395413120)  
Glen Waverley (Ph: 0395601655)  
Mulgrave (Ph: 0395461253)

All of the above actions should be treated as a single transaction.

**[ 6 marks]**

- (ii) An Oracle sequence is to be implemented in the database for the subsequent insertion of records into the database for the BORROWER and RESERVE tables.

Provide the CREATE SEQUENCE statement to create a sequence which could be used to provide primary key values for the BORROWER and RESERVE tables. Both sequences should start at 100 and increment by 1. Immediately prior to the create sequence commands place appropriate DROP SEQUENCE commands so they will be dropped before being created if they exist.

**[ 2 marks]**

- (iii) On the 14<sup>th</sup> September, 2021 a new borrower:

Name: Ada LOVELACE

Home Branch: Clayton (Ph: 0395413120)

You may make up any address data values you need to be able to add this borrower.

joins the library as a member, at 3:30 PM, and places a reservation on a book at her home branch as part of her joining operation. Some of the details of the book that Ada has placed a reservation on are:

Call Number: 005.74 C824C

Title: Database Systems: Design, Implementation, and Management

Edition: 13

You may assume that the Clayton branch only has a single copy of this book.

**[ 7 marks]**

- (iv) Seven days after reserving the book, Ada receives a notification from the Clayton library that the book she had placed a reservation on is available. Ada goes to the library and borrows the book at 12:30 PM on the same day as she received the notification.

You may assume that there is no other borrower named Ada Lovelace, however please note that Ada has other reservations recorded which she has recently added to the system. You may assume that the Clayton branch still only has a single copy of this book.

When a reservation has been completed (ie. the book borrowed), MLib removes the reservation entry from their system.

**[ 10 marks]**

### TASK 3: DATABASE MODIFICATIONS ( 25 marks):

Your answers for these tasks (Task 3) must be placed in the supplied SQL Script **T3-ml-alter.sql**

The required changes must be made to the "live" database (the database *after* you have completed tasks 1 and 2) **not** by editing and executing your schema file again. Before carrying out the work below, please ensure that you have completed tasks 1 and 2 above.

If in answering these questions you need to create a table, please place a drop table statement prior to your create table statement.

After using the system for some time, MLib has realised that it is necessary to:

- (a) Record whether a book is damaged (D) or lost (L). If the book is not damaged or lost, then it is good (G) which means, it can be loaned. The value cannot be left empty for this. When a new book is added, if this value is not assigned it must be set to 'G'. Change the "live" database and add this required information for all the books currently in the database. You may assume that the condition of all existing books will be recorded as being good. The information can be updated later, if need be.

The book:

Call Number: 005.74 C824C

Title: Database Systems: Design, Implementation, and Management

Edition: 13

at the Glen Waverley (Ph: 0395601655) library has been noted as lost. A lost book is still counted as part of the branch's registered count of books. Record this change in the database. You may assume that the Glen Waverley branch only has a single copy of this book.

**[ 6 marks]**

- (b) Allow borrowers to be able to return the books they have loaned to any library branch as MLib is getting a number of requests regarding this from borrowers. As part of this process MLib wishes to record which branch a particular loan is returned to. Change the "live" database and add this required information for all the loans currently in the database. For all completed loans, to this time, books were returned at the same branch from where those were loaned.

**[ 6 marks]**

- (c) Some of the MLib branches have become very large and it is difficult for a single manager to look after all aspects of the branch. For this reason MLib is intending to appoint two managers for the larger branches - one manager for the Fiction collection and another for the Reference collection. The branches which continue to have only a single manager will ask this manager to manage all the branches collection (both fiction and reference) . The number of branches which will require two managers is quite small (around 10% of the total MLib branches). Change the "live" database to allow MLib the option of appointing two managers to a branch and record, for all managers, which collection/s they are managing.

Currently the Clayton branch (Ph: 0395413120) has a very large collection of books in comparison to the other branches, as a result, Indiana (Manager id: 10) who is currently managing the Clayton branch has been asked to manage only the Reference collection of the branch. Nathan (Manager id:12) will assist by managing the Fiction collection of the Clayton branch. Write the code to implement these changes.

Note that MLib does not wish to keep track of any history of the branches/collections which managers manage, only their current responsibilities. While carrying out this question, you can assume that other than the two changes made in the Clayton branch above, other managers will continue to manage multiple branches/collections as currently recorded in the system.

**[ 13 marks]**



## SUBMISSION REQUIREMENTS

**Due Date: Thursday 7th October 2021 4 PM (AEDT) (week 10)**

*Please note, if you need to resubmit, you **cannot** depend on your tutors' availability, for this reason, please be **VERY CAREFUL** with your submission. It is strongly recommended that you submit several hours before this time to avoid such issues.*

For this assignment there are five files you are **required** to submit:

- T1-ml-drop.sql
- T1-ml-schm.sql
- T2-ml-insert.sql
- T2-ml-dm.sql
- T3-ml-alter.sql

If you need to make any comments to your marker/tutor please place them at the head of each of your solution scripts in the "Comments for your marker:" section.

Do not zip these files into one zip archive, submit five independent SQL scripts. The individual files must also have been pushed to the FIT GitLab server with an appropriate history as you developed your solutions (a minimum of five pushes - 1 per file, however we would *strongly recommend more than this*). **Please ensure your commit comments are meaningful.**


***Late submission will incur penalties at the rate of -5 mark for every 12 hours the submission is late.***

Please note we **cannot mark any work on the GitLab Server**, you need to ensure that you submit correctly via Moodle since it is only in this process that you complete the required student declaration without which work **cannot be assessed**.

**It is your responsibility to ENSURE that the files you submit are the correct files - we strongly recommend after uploading a submission, and prior to actually submitting, that you download the submission and double-check its contents.**

Your assignment **MUST** show a status of "Submitted for grading" before it will be marked.

### Submission status

Attempt number	This is attempt 1.
Submission status	Submitted for grading 
Grading status	Not graded

If your submission shows a status of "Draft (not submitted)" it will not be assessed and **will incur late penalties after the due date/time**.

Please **carefully** read the documentation under the "Assignment Submission" on the Moodle Assessments page which covers things such as extensions and resubmission.

## CRITERIA FOR MARKING

Submissions **will be graded on**:

- the correct application of relational database principles,
- the correct handling of transactions and the setting of appropriate transaction boundaries i.e. correct placement of commits, and
- the correct application of SQL statements and constructs to:
  - create and alter tables including the required constraints and column comments,
  - populate tables,
  - modify existing data in tables, and
  - modify the "live" database structure to meet the expressed requirements (including appropriate use of constraints). In making these modifications there must be no loss of existing data or data integrity within the database.

Submissions **will be penalised** as follows:

- if SQL scripts contain contain SET ECHO ... or SPOOL commands [-10 marks]
- if SQL scripts makes use of views [-10 marks]
- if SQL scripts do not use to\_char/to\_date where appropriate in handling dates [-10 marks],
- if SQL scripts do not have an appropriate development history on the FIT GitLab server for all source files (**at least five pushes required, but more expected**) [-10 marks].

## APPENDIX A

### Attribute names/meaning for the missing tables

Table name	Attribute name	Meaning
<b>BOOK_COPY</b>		
	bc_id	Book copy id unique within the branch which owns this book copy (copy ids are reused in each branch)
	bc_purchase_price	Purchase price for this copy
	bc_counter_reserve	Flag to indicate if on Counter Reserve or not (Y/N)
<b>LOAN</b>		
	loan_date_time	Date and time loan taken out
	loan_due_date	Date loan due (no time is assigned)
	loan_actual_return_date	Actual date loan returned (no time is assigned)
<b>RESERVE</b>		
	reserve_id	Reservation number (unique across all branches)
	reserve_date_time_placed	Date and time reserve was placed