



Higher Nationals in Computing

# Unit 4: Website Design and Development

Learner’s name:

ID: GCS190634

Class: GCS0805\_PPT

Subject code: 1622

Assessor name: **NGUYEN VAN SON**

Assignment due: Assignment submitted:

**ASSIGNMENT 2 FRONT SHEET**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Qualification** | **TEC Level 5 HND Diploma in Computing** | | | |
| **Unit number and title** | **Unit 04: Database Design & Development** | | | |
| **Submission date** |  | | **Date Received 1st submission** |  |
| **Re-submission Date** |  | | **Date Received 2nd submission** |  |
| **Student Name** | Vu Hoai Thao My | | **Student ID** | GCS190634 |
| **Class** | GCS0805\_PPT | | **Assessor name** | Nguyen Van Son |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice. | | | | |
|  | |  | **Student’s signature** | MY |

**Grading grid**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P2 | P3 | P4 | P5 | M2 | M3 | M4 | M5 | D2 | D3 |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **Signature & Date:** | | |

**ASSIGNMENT 2 BRIEF**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | |
| **Unit number** | Unit 04: Database Design & Development | | |
| **Assignment title** |  | | |
| **Academic Year** |  | | |
| **Unit Tutor** |  | | |
| **Issue date** |  | **Submission date** |  |
| **IV name and date** |  | | |

|  |
| --- |
| **Submission Format:** |
| *Format:* This assignment is an Individual assignment and specifically including 2 documents:  *(1) sql file of your code and represent your code to your mananger (tutor)*  *(2) a report document*  You must use font *Calibri size 12, set number of the pages and use multiple line spacing at 1.3. Margins must be: left: 1.25 cm; right: 1 cm; top: 1 cm and bottom: 1 cm.* The reference follows Harvard referencing system. The recommended word limit is *2.000-2.500 words*. You will not be penalized for exceeding the total word limit. The cover page of the report has to be the Assignment front sheet 2.  *Submission* Students are compulsory to submit the assignment in due date and in a way requested by the Tutors. The form of submission will be a soft copy posted on <http://cms.greenwich.edu.vn/>  *Note:* The Assignment *must* be your own work, and not copied by or from another student or from  books etc. If you use ideas, quotes or data (such as diagrams) from books, journals or other sources, you must reference your sources, using the Harvard style. Make sure that you know how to reference properly, and that understand the guidelines on plagiarism. *If you do not, you definitely get fail* |
| **Unit Learning Outcomes:** |
| **LO2 Develop** a fully functional relational database system, based on an existing system design.  **LO3 Test the system against user and system requirements.**  **LO4 Produce technical and user documentation** |
| **Assignment Brief and Guidance:** |
| You are employed as a Database Developer for a large IT consultancy company. The company has been approached by FPT university which is expanding due to the growth of the number of students. FPT is currently facing difficulties in dealing with managing the university. It decided to develop several academic systems to manage the university easier including: **Online Library system, Student Grading System, Attendance System, CMS System, Scheduling System, Enrolment Systems, and so on.**  You are tasked to select one of those systems to develop database for FPT university. Your tasks are to:  Work with FPT to find out about current requirements for each system  Analyze the requirements and produce clear statements of user and system requirements.  Design a relational database system using appropriate design tools and techniques  Develop a fully functional relational database system, based on an existing system design.  Test the system against user and system requirements.  Produce technical and user documentation  **Part 2 (Assignment 2)**  Once the designs have been accepted by your manager you have been asked to:  1. Develop the database system using evidence of user interface, output and data validations and querying across multiple tables.  You want to include more than just the basics so you will implement a fully functional database system which will include system security and database maintenance features.  You have decided to implement a query language into the relational database system. The developed system will be demonstrated to your manager.  Your manager has asked you to include in the report:  2. Assessing whether meaningful data has been extracted through the use of query tools to produce appropriate management information.  3. Evaluating the effectiveness of the database solution in relation to user and system requirements, and suggest improvements.  4. Once the system has been developed, you will test the system and your manager will complete a witness statement indicating how your tests are performing against user and system requirements.  You will produce a brief report assessing the effectiveness of the testing, including an explanation of the choice of test data used.  5. Lastly you will produce technical and user documentation which will be given to the company.  You want to provide some graphical representations for ease of reference in the technical guide, so you have decided to produce a technical and user documentation for a fully functional system, including diagrams showing movement of data through the system, and flowcharts describing how the system works. |

|  |  |  |
| --- | --- | --- |
| **Learning Outcomes and Assessment Criteria** | | |
| **Pass** | **Merit** | **Distinction** |
| **LO2** Develop a fully functional relational database system, based on an existing system design.  **LO3** Test the system against user and system requirements.  **LO4** Produce technical and user documentation | | |
| **P2** Develop the database system with evidence of user interface, output and data validations, and querying across multiple tables.  **P3** Implement a query language into the relational database system. | **M2** Implement a fully functional database system which includes system security and database maintenance.  **M3** Assess whether meaningful data has been extracted through the use of query tools to produce appropriate management information. | **LO2 & 3 D2** Evaluate the effectiveness of the database solution in relation to user and system requirements, and suggest improvements. |
| **P4** Test the system against user and system requirements. | **M4** Assess the effectiveness of the testing, including an explanation of the choice of test data used. |
| **P5** Produce technical and user documentation. | **M5** Produce technical and user documentation for a fully functional system, including ER Diagram and normalization statements and describing how the system works. | **D3** Assess any future improvements that may be required to ensure the continued effectiveness of the database system. |

**Table of Contents**

***P2 Develop the database system with evidence of user interface, output and data validations, and querying across multiple tables: 1***

1. **Code snippets to create each table 1**
2. ***Create a database 1***
3. ***Create table 1***
4. **Code snippets to insert some sample data for each table 4**
5. **Add the data into the table “Staff”: 4**
6. **Add the data into the table “Reader”: 4**
7. **Add the data into the table “Author”: 5**
8. **Add the data into the table “Publisher”: 5**
9. **Add the data into the table “Category”: 6**
10. **Add the data into the table “Position”: 6**
11. **Add the data into the table “Language”: 6**
12. **Add the data into the table “Book”: 7**
13. **Add the data into the table “Bill”: 7**
14. **Add the data into the table “BillDetail”: 8**
15. **Database Diagram 9**
16. **Explanations about any changes comparing to your design (if there is) 10**

***P3 Implement a query language into the relational database system: 10***

***P4 Test the system against user and system requirements:***

***P5 Produce technical and user documentation:***

**ASSIGNMENT 2 ANSWERS**

***P2 Develop the database system with evidence of user interface, output and data validations, and querying across multiple tables:***

1. **Code snippets to create each table:**
2. ***Create a database:***

--- CREATE DATABASE "LIBRARY MANAGEMENT SYSTEM" ---

CREATE Database LibraryManagementSystem;

1. ***Create table:***

* ***Code table: Staff***

--- CREATE TABLE STAFF---

use LibraryManagementSystem;

CREATE Table Staff (

staffID char (10 primary key not null,

staffName varchar(30) not null,

staffAge int ,

staffPhone char (10) not null,

staffAddress varchar(30) ,

staffAccount varchar(30) ,

staffPassword varchar(30)

);

* ***Code table: Reader***

--- CREATE TABLE READER ---

use LibraryManagementSystem;

CREATE Table Reader (

readerID char (10) primary key not null,

readerName varchar(30) not null,

readerAge date ,

readerAddress varchar(30) ,

CitizenID int not null,

readerAccount varchar(30) ,

readerPasswordvarchar(30)

);

* ***Code table: Author***

--- CREATE TABLE AUTHOR ---

use LibraryManagementSystem;

CREATE Table Author (

authorID char (10) primary key not null,

authorName varchar(30) not null,

authorAddress varchar(30) not null

);

* ***Code table: Publisher***

--- CREATE TABLE PUBLISHER ---

use LibraryManagementSystem;

CREATE Table Publisher (

publisherID char (10) primary key not null,

publisherName varchar(30) not null,

publisherAddress varchar(30) not null

);

* ***Code table: Category***

--- CREATE TABLE CATEGORY ---

use LibraryManagementSystem;

CREATE Table Category (

categoryID char (10) primary key not null,

categoryName varchar(30) not null

);

* ***Code table: Position***

--- CREATE TABLE POSITION ---

use LibraryManagementSystem;

CREATE Table Position (

positionID char (10) primary key not null,

PZone int not null,

PShelf int not null,

PPart int not null

);

* ***Code table: Language***

--- CREATE TABLE LANGUAGE ---

use LibraryManagementSystem;

CREATE Table Languages (

languageID char (10) primary key not null,

languageName varchar(30) not null

);

* ***Code table: Book***

--- CREATE TABLE BOOK ---

use LibraryManagementSystem;

CREATE Table Book (

bookID char (10) primary key not null,

bookName varchar(30) not null,

authorID char (10) not null,

categoryID char (10) not null,

publisherID char (10) not null,

positionID char (10) not null,

languageID char (10) not null,

foreign key (authorID) references Author,

foreign key (categoryID) references Category,

foreign key (publisherID) references Publisher,

foreign key (positionID) references Position,

foreign key (languageID) references Languages

);

* ***Code table: Bill***

--- CREATE TABLE BILL ---

use LibraryManagementSystem;

CREATE Table Bill (

billID char (10) primary key not null,

readerID char (10) not null,

staffID char (10) not null,

borrowDate date not null,

duedate int

foreign key (readerID) references Reader,

foreign key (staffID) references Staff

);

* ***Code table: BillDetail***

--- CREATE TABLE BILLDETAIL ---

use LibraryManagementSystem;

CREATE Table BillDetail (

billID char (10) not null,

bookID char (10) not null,

Quantity float not null,

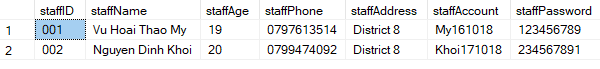
returnDate date not null,

foreign key (billID) references Bill,

foreign key (bookID) references Book

);

1. **Code snippets to insert some sample data for each table:**
2. ***Add the data into the table “Staff”:***



--- INSERT INFORMATION OF STAFF---

Insert into Staff (staffID,staffName,staffAge,staffPhone,staffAddress,staffAccount,staffPassword)

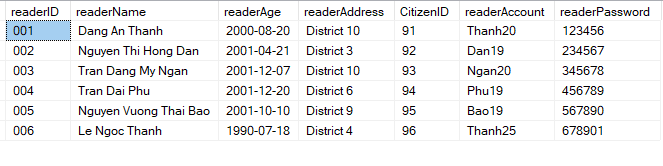
Values ('001','Vu Hoai Thao My',19,'0797613514','District 8','My161018','123456789');

Insert into Staff (staffID,staffName,staffAge,staffPhone,staffAddress,staffAccount,staffPassword)

Values ('002','Nguyen Dinh Khoi',20,'0799474092','District 8','Khoi171018','234567891');

select \* from Staff;

1. ***Add the data into the table “Reader”:***



--- INSERT INFORMATION OF READER ---

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('001','Dang An Thanh','2000-08-20','District 10','091','Thanh20','123456');

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('002','Nguyen Thi Hong Dan','2001-04-21','District 3','092','Dan19','234567');

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('003','Tran Dang My Ngan','2001-12-07','District 10','093','Ngan20','345678');

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('004','Tran Dai Phu','2001-12-20','District 6','094','Phu19','456789');

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('005','Nguyen Vuong Thai Bao','2001-10-10','District 9','095','Bao19','567890');

Insert into Reader (readerID,readerName,readerAge,readerAddress,CitizenID,readerAccount,readerPassword)

Values ('006','Le Ngoc Thanh','1990-07-18','District 4','096','Thanh25','678901');

select \* from Reader;

1. ***Add the data into the table “Author”:***



--- INSERT INFORMATION OF AUTHOR ---

Insert into Author (authorID,authorName,authorAddress)

Values ('101','Joly Mith','USA');

Insert into Author (authorID,authorName,authorAddress)

Values ('102','Zuki Gao','Japan');

Insert into Author (authorID,authorName,authorAddress)

Values ('103','Ngoc Khue','VietNam');

select \* from Author;

1. ***Add the data into the table “Publisher”:***



--- INSERT INFORMATION OF PUBLISHER ---

Insert into Publisher (publisherID,publisherName,publisherAddress)

Values ('111','MK Tuitec','USA');

Insert into Publisher (publisherID,publisherName,publisherAddress)

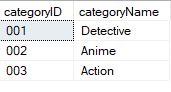
Values ('112','Zusi Sichi','Japan');

Insert into Publisher (publisherID,publisherName,publisherAddress)

Values ('113','NXB Anh Trang','Ngo Tat To');

select \* from Publisher;

1. ***Add the data into the table “Category”:***



--- INSERT INFORMATION OF CATEGORY ---

Insert into Category (categoryID,categoryName)

Values ('001','Detective');

Insert into Category (categoryID,categoryName)

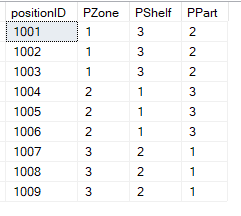
Values ('002','Anime');

Insert into Category (categoryID,categoryName)

Values ('003','Action');

select \* from Category;

1. ***Add the data into the table “Position”:***



--- INSERT INFORMATION OF POSITION ---

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1001', 1, 3, 2);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1002', 1, 3, 2);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1003', 1, 3, 2);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1004', 2, 1, 3);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1005', 2, 1, 3);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1006', 2, 1, 3);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1007', 3, 2, 1);

Insert into Position (positionID,PZone,PShelf,PPart)

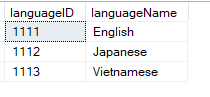
Values ('1008', 3, 2, 1);

Insert into Position (positionID,PZone,PShelf,PPart)

Values ('1009', 3, 2, 1);

select \* from Position;

1. ***Add the data into the table “Language”:***



--- INSERT INFORMATION OF LANGUAGES ---

Insert into Languages (languageID,languageName)

Values ('1111','English');

Insert into Languages (languageID,languageName)

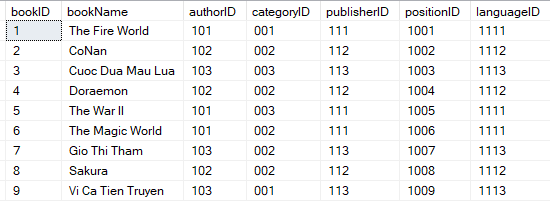
Values ('1112','Japanese');

Insert into Languages (languageID,languageName)

Values ('1113','Vietnamese');

select \* from Languages;

1. ***Add the data into the table “Book”:***



--- INSERT INFORMATION OF BOOK ---

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('1','The Fire World','101','001','111','1001','1111');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('2','CoNan','102','002','112','1002','1112');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('3','Cuoc Dua Mau Lua','103','003','113','1003','1113');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('4','Doraemon','102','002','112','1004','1112');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('5','The War II','101','003','111','1005','1111');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('6','The Magic World','101','002','111','1006','1111');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('7','Gio Thi Tham','103','002','113','1007','1113');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

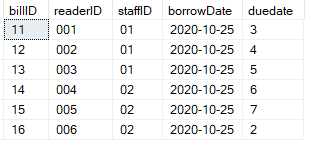
Values ('8','Sakura','102','002','112','1008','1112');

Insert into Book (bookID,bookName,authorID,categoryID,publisherID,positionID,languageID)

Values ('9','Vi Ca Tien Truyen','103','001','113','1009','1113');

select \* from Book;

1. ***Add the data into the table “Bill”:***



--- INSERT INFORMATION OF BILL ---

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

Values ('11','001','01','2020-10-25',3);

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

Values ('12','002','01','2020-10-25',4);

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

Values ('13','003','01','2020-10-25',5);

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

Values ('14','004','02','2020-10-25',6);

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

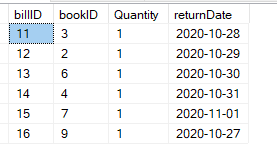
Values ('15','005','02','2020-10-25',7);

Insert into Bill (billID,readerID,staffID, borrowDate, duedate)

Values ('16','006','02','2020-10-25',2);

select \* from Bill;

1. ***Add the data into the table “BillDetail”:***



--- INSERT INFORMATION OF BILLDETAIL ---

Insert into BillDetail (billID,bookID,quantity,returnDate)

Values ('11','3',1,'2020-10-28');

Insert into BillDetail (billID,bookID,quantity,returnDate)

Values ('12','2',1,'2020-10-29');

Insert into BillDetail (billID,bookID,quantity,returnDate)

Values ('13','6',1,'2020-10-30');

Insert into BillDetail (billID,bookID,quantity,returnDate)

Values ('14','4',1,'2020-10-31');

Insert into BillDetail (billID,bookID,quantity,returnDate)

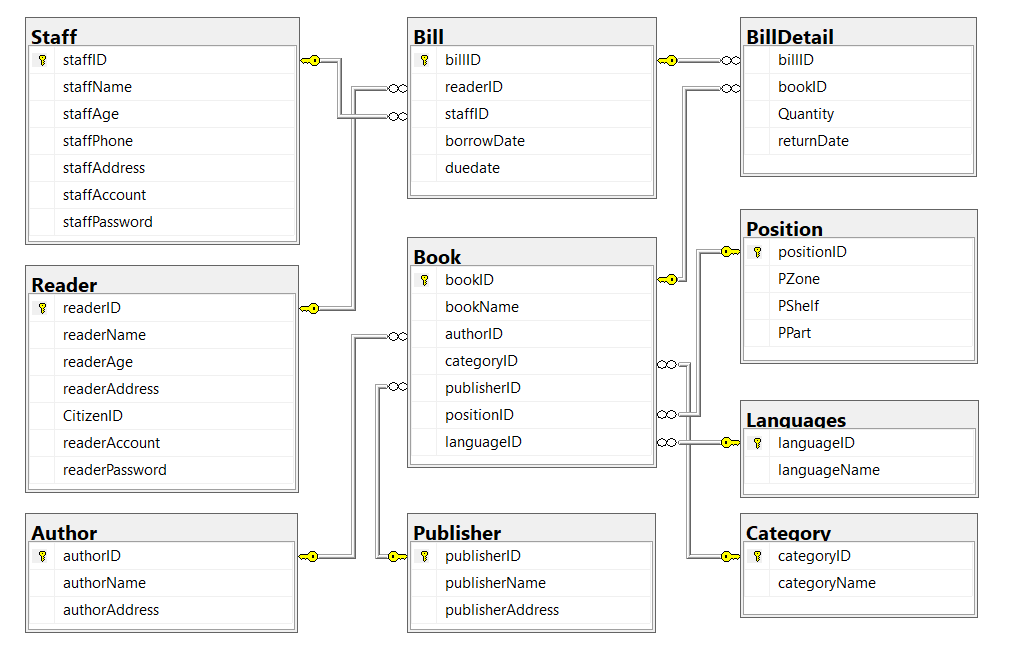
Values ('15','7',1,'2020-11-01');

Insert into BillDetail (billID,bookID,quantity,returnDate)

Values ('16','9',1,'2020-10-27');

select \* from BillDetail;

1. **Database Diagram:**

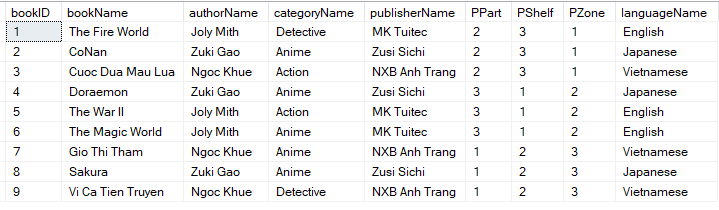


1. **Explanations about any changes comparing to your design:**

* The original tentative design concept was to design an offline library management system in style from past to present. But with the current 4.0 technology situation, modern technology is always at the forefront of the fields, so this design article has been edited into an online library management system. With this design, the management of the library system can be best ensured.

***P3 Implement a query language into the relational database system:***

1. **Queries to support the functionalities (insert, update, delete):**
2. ***LIST ALL INFORMATIONS IN TABLE "BOOK":***



---LIST ALL INFORMATIONS IN TABLE “BOOK”---

select b.bookID, b.bookName, au.authorName, ca.categoryName,

pu.publisherName, po.PPart, po.PShelf, po.PZone, lg.languageName

from Book b, Author au, Category ca, Publisher pu,

Position po, Languages lg

where b.authorID = au.authorID

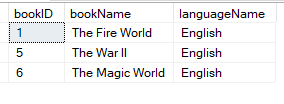
and b.categoryID = ca.categoryID

and b.publisherID = pu.publisherID

and b.positionID = po.positionID

and b.languageID = lg.languageID;

1. ***LIST ALL BOOKS HAVE LANGUAGE IS ENGLISH:***



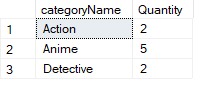
---LIST ALL BOOKS HAVE LANGUAGE IS ENGLISH---

select Bk.bookID, Bk.bookName, lg.languageName

from Book Bk inner join Languages lg on Bk.languageID = lg.languageID

where lg.languageID = '1111';

1. ***COUNT ALL BOOK OF EACH CATEGORY:***



---COUNT ALL BOOK OF EACH CATEGORY---

select Ca.categoryName, COUNT(Bk.bookID) as Quantity

from Book Bk inner join Category Ca on Bk.categoryID = Ca.categoryID

group by Ca.categoryName;

1. ***STATISTICS TABLE "BOOK":***



---STATISTICS TABLE "BOOK"---

select Bk.bookID, Bk.bookName, SUM (BD.Quantity) as Quantity

from BillDetail BD, Book Bk

where BD.bookID = Bk.bookID

group by Bk.bookID, Bk.bookName

1. ***STATISTICS TABLE "READER":***



---STATISTICS TABLE "READER"---

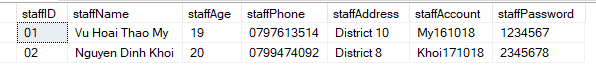
select R.readerID, R.readerName, SUM (BD.Quantity) as Total

from Reader R, Bill B, BillDetail BD

where B.billID = BD.billID and R.readerID = B.readerID

group by R.readerID, R.readerName

1. ***UPDATE STAFF TABLE:***



---UPDATE STAFF TABLE---

UPDATE Staff

SET staffAddress = 'District 10'

WHERE staffID = 01

1. ***UPDATE READER TABLE:***



---UPDATE READER TABLE---

UPDATE Author

SET authorAddress = 'America'

WHERE authorID = 101

1. ***UPDATE LANGUAGES TABLE:***



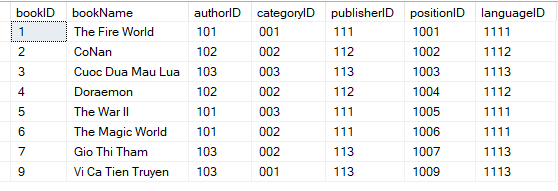
---UPDATE LANGUAGES TABLE---

UPDATE Languages

SET languageName = 'French'

WHERE languageID = 1111

1. ***DELETE BOOK TABLE:***



---DELETE BOOK TABLE---

DELETE FROM Book

WHERE bookID = '8';

1. ***EXCUSE:***



select B.billID, BD.returnDate, R.readerName, S.staffName, Bk.bookName, BD.Quantity

from Bill B, BillDetail BD, Reader R, Staff S, Book Bk

WHERE B.billID = BD.billID and B.staffID = S.staffID and B.readerID = R.readerID and BD.bookID = Bk.bookID

***P4 Test the system against user and system requirements:***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Test*** | ***What is being tested*** | ***How <syntax>*** | ***Test data used*** | ***Expected results*** |
| **1** | Input data into Staff | INSERT <table\_name>  VALUES <values> |  |  |
| **2** | Input data into Reader |  |  |  |
| **3** |  |  |  |  |
| ***4*** |  |  |  |  |
| ***5*** |  |  |  |  |
| ***6*** |  |  |  |  |
| ***7*** |  |  |  |  |
| ***8*** |  |  |  |  |
| ***9*** |  |  |  |  |

***P5 Produce technical and user documentation:***

1. **Technical Documentation:**
2. ***What is Technical Documentation?***

* The technical documentation term refers to different documents which contain product-related data and information that will be helpful for users. Technical documentation contains product definition and specification, quality assurance, manufacturing, description of features and functions, maintenance information, and more.
* The goal of technical documentation is to provide enough information for a user to understand the inner and outer dependencies of the product at hand. It also helps users:
* Troubleshoot their issues.
* Find the best ways to use a product.
* Learn the product from scratch and more.
* Technical documentation is not just manuals, it also includes science papers, research, reports, white papers, case studies and the like. You can find more information on types of technical documentation here: ‘Types of Technical Documents’.
* Usually, technical documentation was created in a printed version but nowadays it’s created online as tech writers use HATs for this purpose.

1. ***What is ERD?***

* An entity-relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system’s entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.
* The elements of an ERD are:
* Entities
* Relationships
* Attributes
* Steps involved in creating an ERD include:
* Identifying and defining the entities
* Determining all interactions between the entities
* Analyzing the nature of interactions/determining the cardinality of the relationships
* Creating the ERD

1. ***Definition of***[***Technology Improvements***](https://www.lawinsider.com/dictionary/technology-improvements)***:***

* Technology Improvements means any proprietary information, know-how, trade secrets, programs, designs, processes, methods, formulae, compositions of matter, documents, materials, technology, data, Intellectual Property Rights, or Derivative Works in developments and/or conceptions created, obtained or developed by either Party alone (including through the efforts of any independent contractor or affiliate of that Party) or together with the other Party that: (a) are based on, derived from or are direct improvements to Technology, (b) can be used in or in the production of Technology, or (c) provide alternatives for use in the production of Technology that, if so used, reasonably would: (i) add Technology capability or increase Technology efficiency or quality, (ii) reduce Technology manufacturing or Technology costs, and/or (iii) facilitate the manufacturing of Technology. Sample 1 Sample 2 Sample 3 Based on 4 documents Save Copy.
* Technology Improvements means any inventions or other intellectual property (including all Patent, Know-How and other intellectual property rights therein) that relates specifically to the Technology made by, or under authority of, the Licensee or any authorized Sublicensee during the Term in conducting the Activities contemplated by this Agreement. For the avoidance of doubt, any invention or other intellectual property (including Patent, Know-How and other intellectual property rights therein) relating to any Antibody or any Product shall not constitute a Technology Improvement under this Agreement to the extent that it is severable from and does not comprise of (in whole or part) or disclose or reveal any Licensor IP Rights. Sample 1 Sample 2 Sample 3 Based on 4 documents Save Copy.

1. ***Examples of Technology Improvements:***

* In the event that either Party desires to use a third party service provider ("Service Provider"), including, for example, an engineering design firm or a contract manufacturer, to develop or produce the Product using Technology or Technology Improvements, all Parties to this Agreement must first enter into at least an acceptable non-disclosure and technology ownership agreement with the Service Provider. This study is the first phase of the CUE Access and Technology Improvements Project. Except as expressly set out in Section 2.1(c) of the Consent Agreement, the University makes no representations, conditions, or warranties, either express or implied, with respect to the Technology, Improvements, Patents or any Products. At the request of UHN or MBI respectively, MBI or UHN shall acknowledge the contribution and ownership of UHN and/or MBI to the Additional Licensed Technology, Improvements by UHN PI(s), Improvements by UHN non-PI(s) or Improvements by MBI, as the case may be. The Public Hospitals major program is the most complicated and is broken down into: - Hospital Projects- Hospital Service Improvements- Hospital Buildings Infrastructure- Hospital Enhancement- Technology Improvements- Mental Health- Legionnaires Disease- Nurse Education- Computers- Annual Provisions Minor programs consists of Specific item (SI) and/or Miscellaneous Item (MI) projects.

1. ***Possible and realistic improvements:***

* Evaluate Primary Key Assignments:
* When you're adding specific entities to a database, it's important to understand the overall concept of how these entities are affecting the rest of the database. You will need to focus on specific entities, usually the Primary Keys, in order to make a successful and reliable database. Inputting information can be very repetitive when doing it consistently, making it more prone for errors leading to referential integrity violations. A surrogate key could be used in this instance by creating tables through primary keys. A surrogate key is generated though the DBMS and is usually hidden from the end-users. A SK is usually numeric as well.
* Evaluate Naming Conventions:
* The purpose of revising naming conventions is to provide a more accurate set of details or to help split up a broad entity/attribute into more of these categories. Lets say you have a broad category such Car parts but no specifics onto what the car parts do. Adding a Parts Description entity would do numerous wonders to describing each attribute more clearly.
* Refine Attribute Atomicity:
* An atomic attribute is a category to which it can no longer be divided into smaller categories. However having an attribute displaying atomicity is can be used in the same group but revised enough to get specific information. Using the example of User Address; the address is composed of various bits of information such as the specific address itself, the city, state, and zip code. You can easily add atomicity to this example by displaying User St Address, User City, User State, and User Zip all in separate areas.
* Identify New Attributes:
* When creating attributes that affect one another, it's important to understand the links they may have to other unknown attributes. A good example would be based off a Participant table in which each Participant may have categories such as the amount of surveys, any payments and specific identities. By adding a new participant attribute, you can track the processes each participant goes through and when they start these processes.
* Identify New Relationships:
* One of the main goals of a database is to reduce the number of undesired data duplications. It is important as a designer to lace the right attributesin the right tables through the normalization process.
* Refine Primary Keys as Required for Data Granularity:
* Obviously data that is created in the lower levels of the database (Atomic) and sub-levels are not as important as the higher, broader attributes. However using the granularity of these low key atomic values will help keep the database more detailed and specific on what needs to be categorized where. An example of this could involve assigning the number of hours an individual drives. However this attribute isn't clear enough by forgetting to state the specifics of the time periods the individual drives such as daily, weekly, monthly and so on. Be sure to focus on explaining these details clearly so if you're arranging a report you can sort by more specific details.

1. **User document:**
2. ***User Database:***

* Database Users are the one who interacts with the system. There can be four different types of users according to the way they interact with the system and for all the different users, different kind of user interfaces are designed as well. The four types of users are:
* Naive users
* Application Programmers
* Sophisticated Users
* Online Analytical Processing (OLAP)
  1. **Naive users:**
* Naive users are also termed as unsophisticated users and they interact with the system by calling anyone application program that has been written previously.
* For e.g. – To transfer the program from one account to another, there is a need for an application program called transfer.
* The user interface that is required for the naïve users is a forms interface, in which the user can fill the required fields.
* Naive users can also easily read the reports that are generated from the database.
  1. **Application programmers:**
* Application programmers are the one who is responsible to write the application programs. They develop user interfaces through different tools. To construct the forms and the reports such that there is no need to write the program, there is a tool named Rapid Application Development (RAD).
* Some special type of programming languages is also available such that includes vital control structures such as for loops, while loops and many others with the data manipulation language's statements. These special programming languages are termed as fourth-generation languages and they include the special features to provide the ability for the generation of the forms and to display the data on the screen.
* In today's world, a large variety of commercial database systems includes these fourth generation languages.
  1. **Sophisticated users:**
* Sophisticated users aren't interested in writing programs and they interact with the system without writing any programs. Contrary, they use database query languages to interact with the system.
* Sophisticated Users submit their queries to a query processor. Query Processor provides the facility to break the DML statements into the instruction that can be understood by the storage manager.
* Analysts are one among the sophisticated users. They use the tools to perform their task such as:
* Online analytical processing (OLAP) - It helps the analysts to view them the summaries of the data in different ways.
* Data Mining Tools – It helps the analysts find a certain kind of pattern in the given data.
  1. **Specialized users:**
* Among Sophisticated Users, there are Specialized Users too who used to write specialized database applications that differ to the traditional framework of data processing.
* These users use the interfaces that are computer-aided design systems, knowledge base and systems that store the complex data types and also environment-modeling systems.

1. ***User interfaces:***

* Pyrseas was started to improve on Andromeda. In addition to the schema version control I’ve been discussing in these posts, Andromeda provides two other capabilities: automations and web application programming.
* Andromeda’s web programming allows you to generate a database application with very little actual programming, since Andromeda takes into account the primary key, foreign key and other information present in the YAML specification to generate much of the application code for you. This is another area that I’d like to explore for implementation in Pyrseas.
  1. **Administrative Applications:**
* Nearly any database needs a user interface to add, retrieve, update and delete records, known by the unattractive acronym CRUD. There are a number of admin apps that provide such generic facilties. For example, for PostgreSQL we have pgAdmin which offers a GUI, and phpPgAdmin and Adminer which offer PHP web interfaces. These have a couple of limitations.
* Admin apps almost always operate on a single table at a time. Interfaces for end users often require interacting with or viewing two or more tables on a single screen or page, e.g., customers with their invoices, accounts with their transactions.
* By definition, admin apps are “raw” and cater to the needs of DBAs and programmers. After retrieving a customer record searching by name, the admin user can update any column, including the primary key and other columns that end users would normally be prevented from updating.
  1. **Django Admin:**
* Django includes an appealing admin site. With a moderate dose of programming, you can develop an application suitable for end users. However, you’re at the mercy of Django’s constraints on “modeling” a database. Take for example the film\_category table I used in my second post on version control:

CREATE TABLE film\_category (

film\_id INTEGER NOT NULL REFERENCES film (id),

category\_id INTEGER NOT NULL

REFERENCES category (category\_id),

last\_update TIMESTAMP WITH TIME ZONE NOT NULL,

PRIMARY KEY (film\_id, category\_id)

);

* Django cannot use this table “as is” because it insists in having a single column as the primary key. So you end up adding an unnecessary “id” column together with a SEQUENCE (and needless to say, you incur the wrath of the SSSKA).
* We could turn to SQLAlchemy, a Python-based ORM that understands that a primary key –albeit singular– can encompass more than one column, and some other web framework or even Django itself, and attempt to recreate the nice Django admin app. However, aside from the additional effort that would represent, we wouldn’t be taking advantage of the YAML database specification we can now store in our VCS.
* This is a very broad, and some may say, ambitious topic, but as I did with version control, in future posts I’ll survey existing “art” and progress towards designing and implementing a generic end user interface for PostgreSQL. Reader feedback will be much appreciated.

1. ***Solution and step to achieve the functionality provided:***
   1. **Solution:**

* The Oracle Database In-Memory (Oracle Database In-Memory) solution, supporting Oracle Database 12c, allows organizations to find important information instantly while increasing transaction efficiency.
* This solution provides maximum performance for the internal memory without any restrictions on functionality, quality, or risk and complexity.
* Specifically, Oracle Database In-Memory allows customers to improve database performance by dividing capacity for all three data analysis, storage and reporting while still speeding up data delivery. online translation (OLTP). Main memory's innovative, dual-format architecture includes the best horizontal and vertical formats for faster data analysis and more efficient OLTP.
* Database system solution on internal memory (Oracle Database In-Memory), supporting Oracle Database 12c
* Oracle Database In-Memory is suitable for all current Oracle Database applications, thereby automatically taking advantage of the advantages of handling the internal memory by column, without any program changes or changes. application.
* This database system can speed up from 100 to 1,000 times of all enterprise application modules, including Oracle E-Business Suite, Oracle's JD Edwards, Oracle's PeopleSoft, Oracle's Siebel, and Oracle Fusion Applications in test tests.
* In addition, the ability to analyze instant data and execute sub-transactions at the same time on an existing application allows organizations to achieve "real-time" and be able to make decisions based on data. is available, immediately responding to customer needs and continuously optimizing the efficiency of critical processes.
  1. **There are 5 steps to successfully providing functionality and ensuring database design articles:**
* Step # 1: Review, refine and re-engineer business processes - Today's increasingly sophisticated relational databases offer a myriad of functions: membership dues processing, events registration and physicals, publications sales, fundraising, exhibit management, and many more. One element common to all is process.
* We define process as how information moves through the association and through the database. For example, we process membership dues by reviewing the membership application, entering the key data elements, creating an invoice, entering payment against the invoice, printing a welcome letter, sending a welcome packet, and so on.
* When contemplating a new database or an upgrade, completely review your business processes. This will help you determine if you have inefficient or improper (or just plain outdated) procedures in place. Discuss these processes with your association software developer. Your developer has created the software to handle each process in a particular manner. This may differ from yours. Find out what it is, and if you can adapt your process to fit more closely what the developer had in mind, without sacrificing your business practice, do it. This may require some fairly radical changes to how you do business internally, but in the long run will help to make the database more useful.
* The primary danger of implementing a new database without reviewing your business process is that you may wind up automating bad process. Bad process automated will continue to give you the wrong information, only faster than before. Don't do it. (In addition, if bad process is causing your database to function improperly, you can bet the staff will blame the software.)
* Step #2: Develop Proper Training and Documentation — These two go hand-in-hand and are perhaps the issues most commonly complained about by association executives. Once a review of the business process is complete and the installation or upgrade has begun, training and documentation become vital. Unfortunately, association software providers are able to provide only generalized documentation and training for their system. Neither their training nor their documentation will be specific to your particular membership database and its special elements. This is where you need a specialist on the inside.
* Some associations are now creating full-time positions, such as a member records administrator or database coordinator. The position is responsible for providing documentation and training for the membership database. From processing new members, to changing membership entitlements, to simply changing an email address, this position is responsible for providing step-by-step instructions for each of the processes.
* If a full-time position is not possible, consider using part-time help. You may also want to contract with a database consultant. Have the consultant learn your processes and your database, and then work with staff to document processes and develop training materials.
* Step #3: Develop and Rigidly Adhere to Data Standards — You may have heard the term GIGO; garbage in, garbage out. Simply put, this means that if the information you store in your database is not entered accurately or uniformly, no amount of reporting, consulting, or crying is going to give you the information you need. This is where data standards come into play.
* At one association we spent many months working with key database users developing a set of standards by which to enter all data. Not only do the data standards state explicitly how to enter data (e.g., Street, not St., PO, not P.O., etc.), but it also clearly explains where to enter data and what fields must absolutely be entered in order for a record to be complete.
* Standardizing data entry is critical for developing useful and accurate management reports. Without strict standards, sorting of data can become difficult if not impossible. For example, without clearly defined standards for titles, selecting and sorting on the title of vice president quickly becomes impossible, due to the myriad variations of this title (Vice President, VP, V.P., Vice Pres., etc.). With a data entry standard that clearly states that one enters the title vice president as “Vice President,” selecting on this title now becomes very easy. The same will be true of any other data fields you may choose to select or sort on.
* Step #4: Continuously Collect Data — The old saying goes, the minute a directory is printed, it’s already out of date. The same is true for association databases. Depending on the industry or profession you serve, the addresses in your database may change as frequently as 5% per month. Over the course of the year, with no updating being done, you could have a database with over 50% incorrect information! So how do you keep up? Build data collection into the process.
* Develop a laundry list of methods, both passive and active, for collecting data to update your database. Some of the more obvious data collection mechanisms are publication order forms, meeting registration forms, and an annual directory update. But there are other methods for collecting and updating data in your database. These include establishing a web site that lets your members review their contact information, setting up computers with your database at your annual meeting for members to review, and working with your regions, chapters, or affiliates to assist you with data collection.
* Data is gold to an association. Keep your gold shining by actively and continuously seeking to update your data.
* Step #5: Know What You Need and Want From Your Database — Too often, associations select an association database system based on what it is capable of doing, rather than on whether it does what the association needs it to do. Understanding what you need from your database is not only key in the selection of your database but is essential to its continued effectiveness.
* Typically there are several functions that a database needs to provide: contact management (address, phone, fax, email), membership tracking and invoicing, volunteer/committee activity, and more. But what are some of the other items you need to track now or may need to track in the future? How about meeting attendance, publications purchasing history, continuing education credits, or areas of interests? All of these issues require the collection of data. If you’re not collecting it, you can’t report it out, and you have to know you want to report it out to begin collecting it.
* Understanding what your needs are now, and anticipating what your needs may be in the future can go a long way toward creating an effective database. Be sure that you understand your needs and can articulate them to your software provider.

1. ***Data validation:***

* In computer science, data validation is the process of ensuring data has undergone data cleansing to ensure they have data quality, that is, that they are both correct and useful. It uses routines, often called "validation rules", "validation constraints", or "check routines", that check for correctness, meaningfulness, and security of data that are input to the system. The rules may be implemented through the automated facilities of a data dictionary, or by the inclusion of explicit application program validation logic of the computer and its application.
* This is distinct from formal verification, which attempts to prove or disprove the correctness of algorithms for implementing a specification or property.

**REFERENCES**

1. techopedia.com, 2017. ***Entity-Relationship Diagram (ERD)*** [online]. Available: <https://www.techopedia.com/definition/1200/entity-relationship-diagram-erd#:~:text=An%20entity-relationship%20diagram%20%28ERD%29%20is%20a%20data%20modeling,infrastructure.%20The%20elements%20of%20an%20ERD%20are%3A%20Entities>. [4 Nov. 2020].
2. lawinsider.com, ***Definition of Technology Improvements*** [online]. Available: <https://www.lawinsider.com/dictionary/technology-improvements> [4 Nov. 2020].
3. wordpress.com, 2011. ***User Interfaces for Database*** [online]. Available: <https://pyrseas.wordpress.com/2011/05/23/user-interfaces-for-databases/> [4 Nov. 2020].
4. effectivedatabase.com. ***Five Steps to Ensure the Success of Your Database*** [online]. Available: <https://effectivedatabase.com/five-steps-to-ensure-the-success-of-your-database/> [4 Nov. 2020].
5. gimasys.com, 2018. ***Giải pháp hệ thống cơ sở dữ liệu trên bộ nhớ trong*** [online]. Available: <https://www.gimasys.com/tin-tuc/giai-phap-he-thong-co-so-du-lieu-tren-bo-nho-trong-oracle-database-in-memory-cho-doanh-nghiep> [4 Nov. 2020].
6. Fandom.com, 2020. ***Improving the Design of a Database*** [online]. Available: <https://databasemanagement.fandom.com/wiki/Improving_the_Design_of_a_Database> [4 Nov. 2020].
7. Wikipedia.org, 2020. ***Data validation*** [online]. Available: <https://en.wikipedia.org/wiki/Data_validation> [4 Nov. 2020].
8. includehelp.com, ***Database Users and User Interfaces in DBMS [***online]. Available: <https://www.includehelp.com/dbms/database-users-and-user-interfaces-in-dbms.aspx> [4 Nov. 2020].