# 21CSL55: DBMS LABORATORY WITH MINI PROJECT

**Course objectives:** This course will enable students to

* Foundation knowledge in database concepts, technology and practice to groom students into well-informed database application developers.
* Strong practice in SQL programming through a variety of database problems.
* Develop database applications using front-end tools and back-end DBMS.

**Database:** A Database is a collection of interrelated data and a Database Management System is a a software system that enables users to define, create and maintain the database and which provides controlled access to the database

**SQL:** It is structured query language, basically used to pass the query to retrieve and manipulate the information from database. Depending upon the nature of query, SQL is divided into different components:

* + **DDL**(Data Definition Language )
  + **DML**(Data Manipulation Language )
  + **DCL**(Data Control Language )

**DDL:** The Data Definition Language (DDL) is used to create the database (i.e. tables, keys, relationships etc), maintain the structure of the database and destroy databases and database objects.

**Eg.** Create, Drop, Alter, Describe, Truncate

1. **CREATE** statements: It is used to create the table.

# Syntax:

CREATE TABLE table\_name(columnName1 datatype(size), columnName2 datatype(size), );

1. **DROP statements:** To destroy an existing database, table, index, or view. If a table is dropped all records held within it are lost and cannot be recovered.

# Syntax:

DROP TABLE table\_name;

1. **ALTER statements:** To modify an existing database object.

# Adding new columns:

**Syntax:**

Alter table table\_name Add(New\_columnName1 datatype(size), New\_columnName2 datatype(size), )

# Dropping a columns from a table : Syntax:

Alter table table\_name DROP column columnName:

# Modifying Existing columns:

**Syntax:**

Alter table table\_name Modify (columnName1 Newdatatype(Newsize));

1. **Describe statements:** To describe the structure (column and data types) of an existing database, table, index, or view.

# Syntax:

DESC table\_name;

1. **Truncate statements:** To destroy the data in an existing database, table, index, or view. If a table is truncated all records held within it are lost and cannot be recovered but the table structure is maintained.

# Syntax :

TRUNCATE TABLE table\_name;

# Data Manipulation Language (DML):

* A Data Manipulation Language enables programmers and users of the database to retrieve insert, delete and update data in a database. e.g. INSERT, UPDATE, DELETE, SELECT.

**INSERT**: INSERT statement adds one or more records to any single table in a relational database.

# Syntax:

INSERT INTO tablename VALUES (expr1,expr2 );

**UPDATE:** UPDATE statement that changes the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition.

# Syntax:

UPDATE table\_name SET column\_name = value [, column\_name = value ] [WHERE

condition]

**DELETE:** DELETE statement removes one or more records from a table. A subset may be defined for deletion using a condition, otherwise all records are removed.

# Syntax:

DELETE FROM tablename WHERE condition:

**SELECT:** SELECT statement returns a result set of records from one or more tables. The select statement has optional clauses:

* + WHERE specifies which rows to retrieve
* GROUP BY groups rows sharing a property so that an aggregate function can be applied to each group having group.
  + HAVING selects among the groups defined by the GROUP BY clause.
  + ORDER BY specifies an order in which to return the rows.

# Syntax:

SELECT<attribute list> FROM<table list> WHERE<condition> Where

* Attribute list is a list of attribute name whose values to be retrieved by the query.
* Table list is a list of table name required to process query.
* Condition is a Boolean expression that identifies the tuples to be retrieved by query.

**Data Constraints** are the business Rules which are enforced on the data being stored in a table are called Constraints.

Types of Data Constraints

* 1. I/O Constraint This type of constraint determines the speed at which data can be inserted or extracted from an Oracle table. I/O Constraints is divided into two different types
     + The Primary Key Constraint
     + The Foreign Key Constraint
  2. Business rule Constraint This type of constraint is applied to data prior the data being Inserted into table columns.
     + Column level
     + Table level

# The PRIMARY KEY defined at column level Syntax:

CREATETABLEtablename (Columnname1DATATYPE CONSTRAINT <constraintname1> PRIMARY KEY, Columnname2 DATATYPE, columnname3 DATATYPE, );

# The PRIMARY KEY defined at table level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, **PRIMARY KEY (columnname1, columnname2));**

# The FOREIGN KEY defined at column level Syntax

CREATE TABLE tablename (Columnname1 tablename[(columnname)] [ON DELETE CASCADE], columnname3 DATATYPE , );

DATATYPE columnname2 REFERENCES DATATYPE ,

The table in which FOREIGN KEY is defined is called FOREIGN TABLE or DETAIL TABLE. The table in which PRIMARY KEY is defined and referenced by FOREIGN KEY is called PRIMARY TABLE or MASTER TABLE.

**ON DELETE CASCADE** is set then DELETE operation in master table will trigger the DELETE operation for corresponding records in the detail table.

# The FOREIGN KEY defined at table level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, PRIMARY KEY (columnname1, columnname2), FOREIGN KEY (columnname2) REFERENCES tablename2;

A CONSTRAINT can be given User Defined Name, the syntax is: CONSTRAINT < constraint name><constraint definition>

# The CHECK Constraint defined at column level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE CHECK (logical expression), columnname2 DATATYPE, columnname3 DATATYPE,...);

# The CHECK Constraint defined at table level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, CHECK (logical expression1), CHECK (logical expression2));

# The UNIQUE Constraint defined at the column level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE UNIQUE, columnname2 DATATYPE UNIQUE, columnname3 DATATYPE ...);

# The UNIQUE Constraint defined at the the table level Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, UNIQUE(columnname1));

# NOT NULL constraint defined at column level :

**Syntax:**

CREATE TABLE tablename (Columnname1 DATATYPE NOT NULL, columnname2 DATATYPE NOT NULL, columnname3 DATATYPE,...);

# Note:

The NOT NULL constraint can only be applied at column level.

**ER- Diagram:** It is an Entity **–**Relationship diagram which is used to represent the relationship between different entities. An entity is an object in the real world which is distinguishable from other objects. The overall logical structure of a database can be expressed graphically by an ER diagram, which is built up from following components.

* Rectangles: represent entity sets.
* Ellipses: represent attributes.
* Diamonds: represent relationships among entity sets.
* Lines: link attribute to entity sets and entity sets to relationships.

**Mapping Cardinalities:** It expresses the number of entities to which another entity can be associated via a relationship set. For a binary relationship set R between entity sets A and B. The Mapping Cardinalities must be one of the following.

* One to one
* One to many
* Many to one
* Many to many

**LAB EXPERIMENTS**

*PART A: SQL PROGRAMMING*

# 1.Consider the following schema for a Library Database:

**BOOK (*Book\_id, Title, Publisher\_Name, Pub\_Year*) BOOK\_AUTHORS (Book\_id, Author\_*Name*) PUBLISHER (*Name, Address, Phone*) BOOK\_COPIES (*Book\_id, Branch\_id, No-of\_Copies*)**

**BOOK\_LENDING (*Book\_id, Branch\_id, Card\_No, Date\_Out, Due\_Date*) LIBRARY\_BRANCH (*Branch\_id, Branch\_Name, Address*)**

# Write SQL queries to

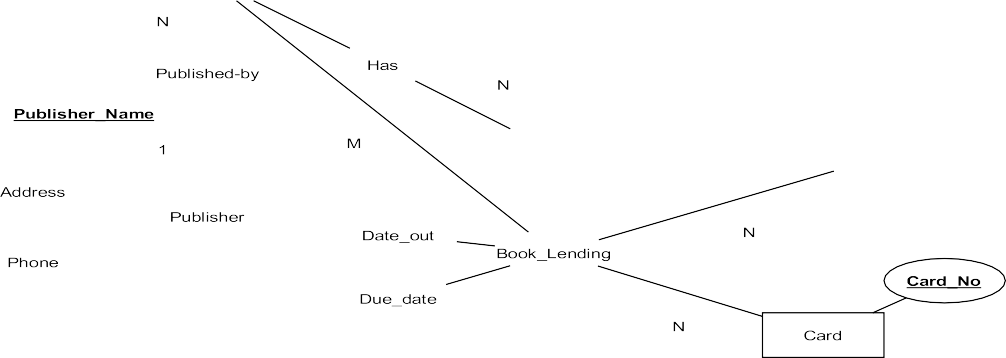
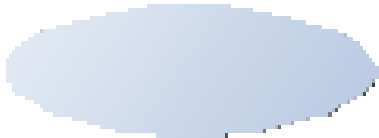
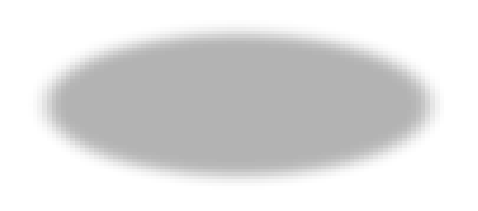
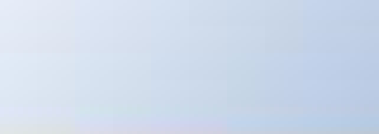
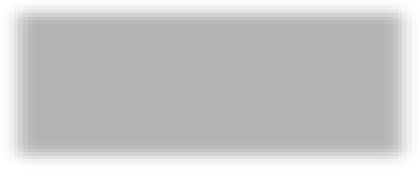
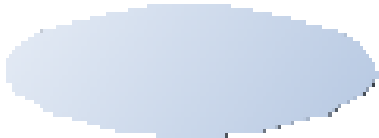
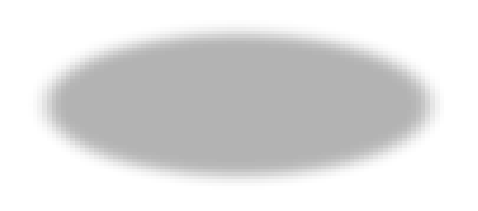
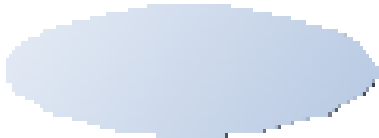
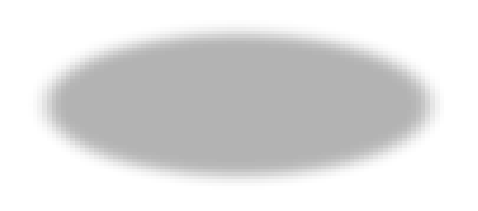
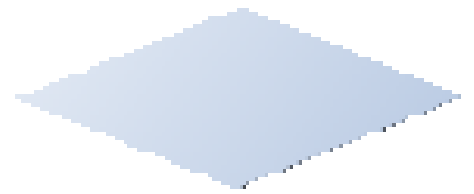
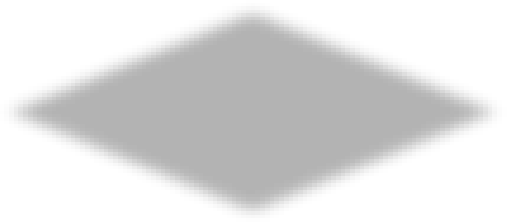
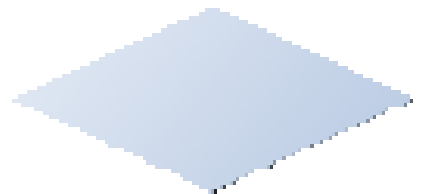
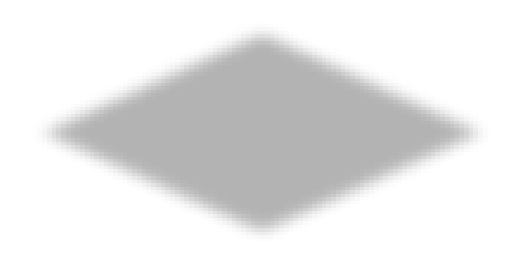
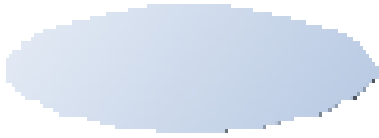
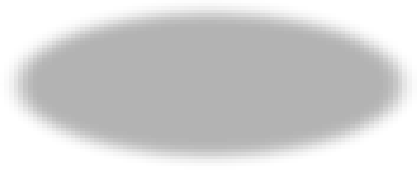
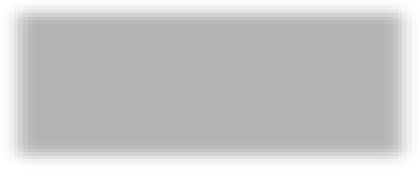
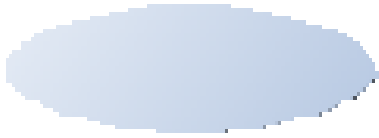
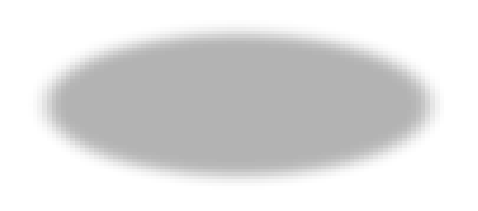
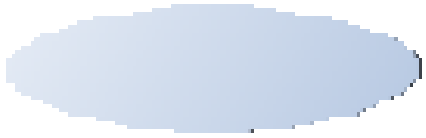
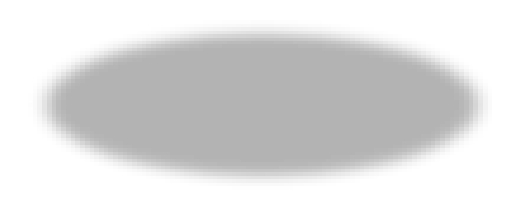
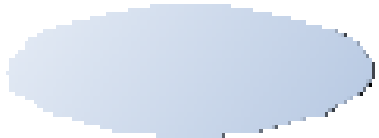
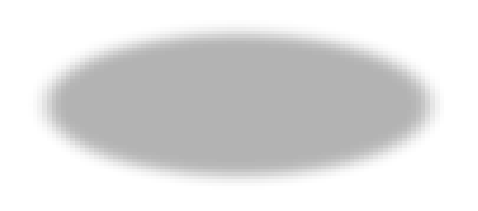
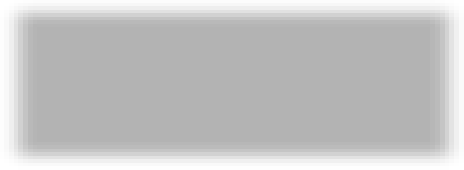
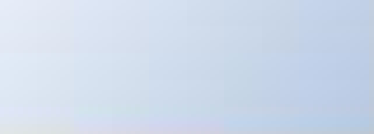
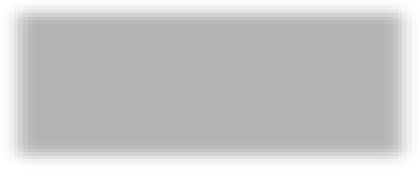
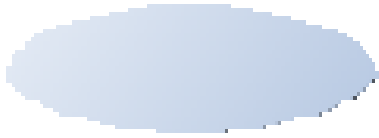
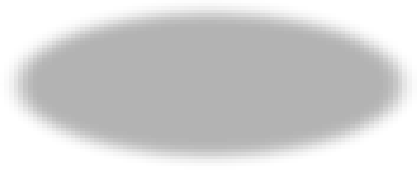
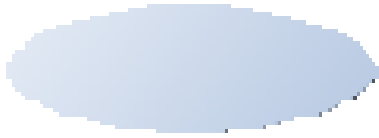
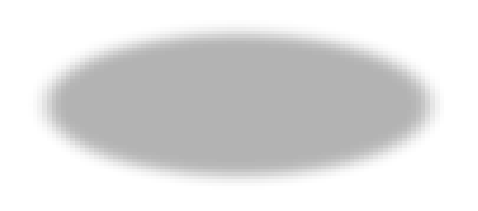
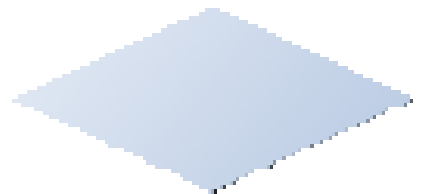
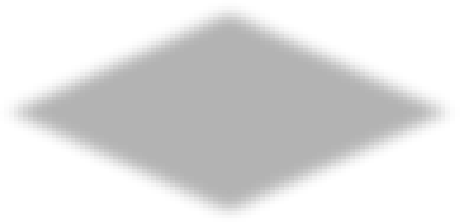
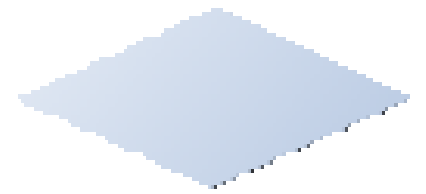
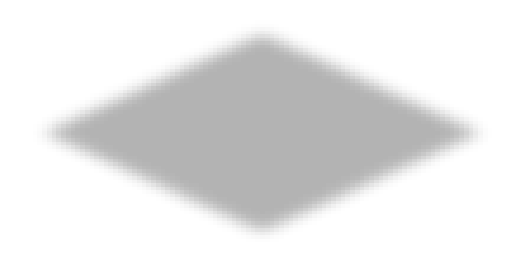
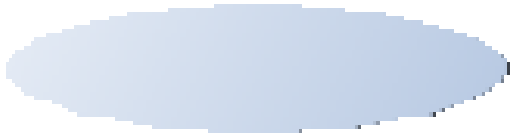
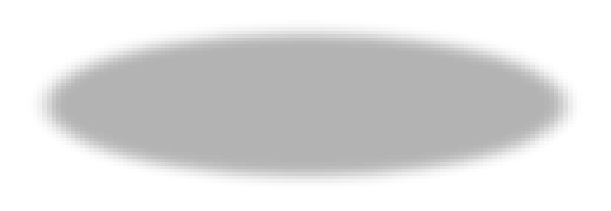
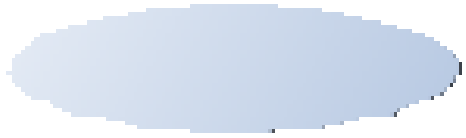
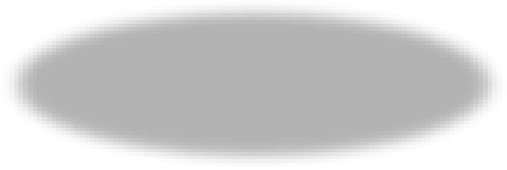
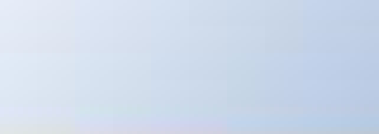
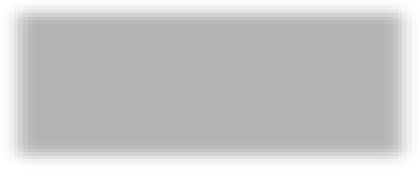
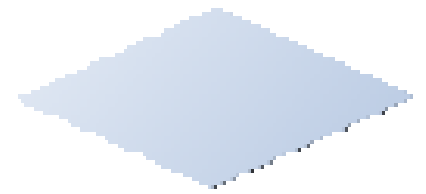
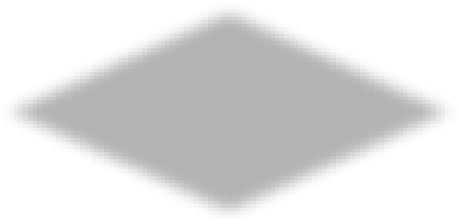
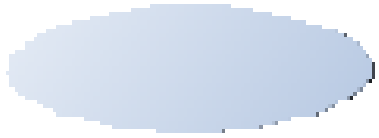
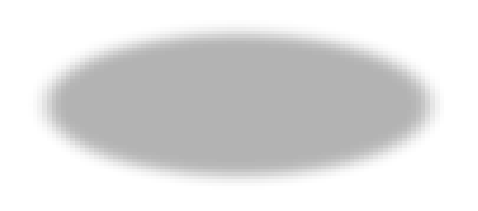
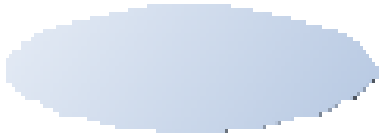
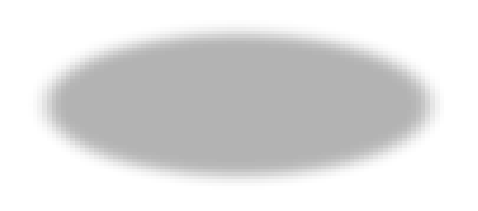
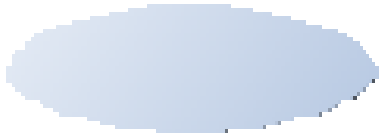
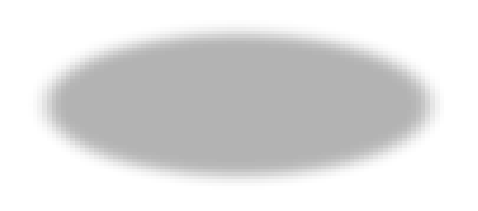
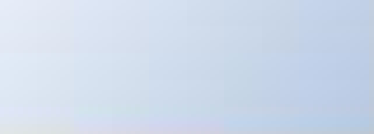
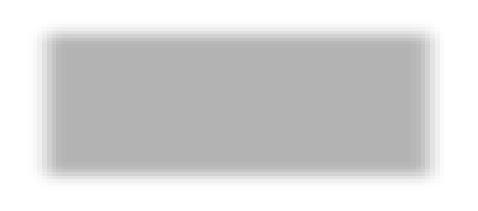
1. **Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.**

# Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017

1. **Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.**

# Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

1. **Create a view of all books and its number of copies that are currently available in the Library.**



**Solution:**

**Entity-Relationship Diagram**

M N

Library\_Branch

Book\_Copies

**Schema Diagram**

**Table Creation:**

**PUBLISHER**

SQL> **CREATE TABLE** PUBLISHER(

NAME VARCHAR(18) **PRIMARY KEY**, ADDRESS VARCHAR(10),

PHONE VARCHAR(10));

Table created.

**BOOK**

SQL> **CREATE TABLE** BOOK(

BOOK\_ID INTEGER **PRIMARY KEY**, TITLE VARCHAR(20), PUBLISHER\_NAME VARCHAR(20) PUB\_YEAR NUMBER(4),

**FOREIGN KEY**(PUBLISHER\_NAME) **REFERENCES** PUBLISHER(NAME)**ON DELETE**

**CASADE**

);

Table created.

**BOOK\_AUTHORS**

SQL> **CREATE TABLE** BOOK\_AUTHORS( BOOK\_ID INTEGER, AUTHOR\_NAME VARCHAR(20), **PRIMARY KEY**(BOOK\_ID),

**FOREIGN KEY**(BOOK\_ID) **REFERENCES** BOOK(BOOK\_ID) **ON DELETE CASCADE**);

Table created.

**LIBRARY\_BRANCH**

SQL> **CREATE TABLE** LIBRARY\_BRANCH( BRANCH\_ID INTEGER **PRIMARY KEY**, BRANCH\_NAME VARCHAR(18), ADDRESS VARCHAR(15));

Table created.

**BOOK\_COPIES**

SQL> **CREATE TABLE** BOOK\_COPIES( BOOK\_ID INTEGER, BRANCH\_ID INTEGER, NO\_OF\_COPIES INTEGER,

**FOREIGN KEY**(BOOK\_ID) **REFERENCES** BOOK(BOOK\_ID) **ON DELETE CASCADE**, **FOREIGN KEY**(BRANCH\_ID) **REFERENCES** LIBRARY\_BRANCH(BRANCH\_ID) **ON DELETE CASCADE**,

**PRIMARY KEY**(BOOK\_ID,BRANCH\_ID));

Table created.

**BOOK\_LENDING**

SQL> **CREATE TABLE** BOOK\_LENDING( BOOK\_ID INTEGER,

BRANCH\_ID INTEGER, CARD\_NO INTEGER, DATE\_OUT DATE, DUE\_DATE DATE,

**PRIMARY KEY**(BOOK\_ID,BRANCH\_ID,CARD\_NO)**,**

**FOREIGN KEY**(BOOK\_ID) **REFERENCES** BOOK(BOOK\_ID) **ON DELETE CASCADE**, **FOREIGN KEY**(BRANCH\_ID) **REFERENCES** LIBRARY\_BRANCH(BRANCH\_ID) **ON DELETE CASCADE**,

); Table created.

# Values for tables:

**PUBLISHER**

SQL>INSERT INTO PUBLISHER VALUES('PEARSON','BANGALORE','9875462530'); SQL> INSERT INTO PUBLISHER VALUES('MCGRAW','NEWDELHI','7845691234'); SQL> INSERT INTO PUBLISHER VALUES('SAPNA','BANGALORE','7845963210');

**BOOK**

SQL> INSERT INTO BOOK VALUES(1111,'SE','PEARSON',2005); SQL> INSERT INTO BOOK VALUES(2222,'DBMS','MCGRAW',2004);

SQL> INSERT INTO BOOK VALUES(3333,'ANOTOMY','PEARSON',2010); SQL> INSERT INTO BOOK VALUES(4444,'ENCYCLOPEDIA','SAPNA',2010);

**BOOK\_AUTHORS**

SQL> INSERT INTO BOOK\_AUTHORS VALUES(1111,'SOMMERVILLE'); SQL> INSERT INTO BOOK\_AUTHORS VALUES(2222,'NAVATHE'); SQL> INSERT INTO BOOK\_AUTHORS VALUES(3333,'HENRY GRAY'); SQL>

INSERT INTO BOOK\_AUTHORS VALUES(4444,'THOMAS');

**LIBRARY\_BRANCH**

SQL> INSERT INTO LIBRARY\_BRANCH VALUES(11,'CENTRAL TECHNICAL','MG ROAD');

SQL> INSERT INTO LIBRARY\_BRANCH VALUES(22,'MEDICAL','BH ROAD');

SQL> INSERT INTO LIBRARY\_BRANCH VALUES(33,'CHILDREN','SS PURAM'); SQL> INSERT INTO LIBRARY\_BRANCH VALUES(44,'SECRETARIAT','SIRAGATE'); SQL> INSERT INTO LIBRARY\_BRANCH VALUES(55,'GENERAL','JAYANAGAR');

**BOOK\_COPIES**

SQL> INSERT INTO BOOK\_COPIES VALUES(1111,11,5); SQL> INSERT INTO BOOK\_COPIES VALUES(3333,22,6); SQL> INSERT INTO BOOK\_COPIES VALUES(4444,33,10); SQL> INSERT INTO BOOK\_COPIES VALUES(2222,11,12); SQL> INSERT INTO BOOK\_COPIES VALUES(4444,55,3);

**BOOK\_LENDING**

SQL> INSERT INTO BOOK\_LENDING VALUES(2222,11,1,'10-JAN-2017','20-AUG-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(3333,22,2,'09-JUL-2017','12-AUG-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(4444,55,1,'11-APR-2017','09-AUG-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(2222,11,5,'09-AUG-2017','19-AUG-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(4444,33,1,'10-JUN-2017','15-AUG-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(1111,11,1,'12-MAY-2017','10-JUN-2017'); SQL> INSERT INTO BOOK\_LENDING VALUES(3333,22,1,'10-JUL-2017','15-JUL-2017');

SQL> SELECT \* FROM BOOK;

BOOK\_ID TITLE PUBLISHER\_NAME PUB\_YEAR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1111 SE |  | PEARSON |  | 2005 |
| 2222 DBMS |  | MCGRAW |  | 2004 |
| 3333 ANOTOMY |  | PEARSON |  | 2010 |
| 4444 ENCYCLOPEDIA |  | SAPNA |  | 2010 |

4 rows selected.



SQL> SELECT \* FROM BOOK\_AUTHORS; BOOK\_ID AUTHOR\_NAME

1111 SOMMERVILLE

2222 NAVATHE

3333 HENRY GRAY

4444 THOMAS

4 rows selected.

|  |  |  |
| --- | --- | --- |
| SQL> SELECT  NAME | \* FROM PUBLISHER;  ADDRESS | PHONE |
| PEARSON | BANGALORE | 9875462530 |
| MCGRAW | NEWDELHI | 7845691234 |
| SAPNA | BANGALORE | 7845963210 |

3 rows selected.

SQL> SELECT \* FROM BOOK\_COPIES; BOOK\_ID BRANCH\_ID NO\_OF\_COPIES

|  |  |  |
| --- | --- | --- |
| 1111 | 11 | 5 |
| 3333 | 22 | 6 |
| 4444 | 33 | 10 |
| 2222 | 11 | 12 |
| 4444 | 55 | 3 |

5 rows selected.

SQL> SELECT \* FROM BOOK\_LENDING;

BOOK\_ID BRANCH\_ID CARD\_NO DATE\_OUT DUE\_DATE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2222 | 11 | 1 |  | 10-JAN-17 |  | 20-AUG-17 |
| 3333 | 22 | 2 |  | 09-JUL-17 |  | 12-AUG-17 |
| 4444 | 55 | 1 |  | 11-APR-17 |  | 09-AUG-17 |
| 2222 | 11 | 5 |  | 09-AUG-17 |  | 19-AUG-17 |
| 4444 | 33 | 1 |  | 10-JUL-17 |  | 15-AUG-17 |
| 1111 | 11 | 1 |  | 12-MAY-17 |  | 10-JUN-17 |
| 3333 | 22 | 1 |  | 10-JUL-17 |  | 15-JUL-17 |

7 rows selected.

SQL> SELECT \* FROM LIBRARY\_BRANCH; BRANCH\_ID BRANCH\_NAME ADDRESS

11 CENTRAL TECHNICAL MG ROAD

22 MEDICAL BH ROAD

33 CHILDREN SS PURAM

44 SECRETARIAT SIRAGATE

55 GENERAL JAYANAGAR

5 rows selected.

Queries:

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.

SELECT LB.BRANCH\_NAME, B.BOOK\_ID,TITLE, PUBLISHER\_NAME,AUTHOR\_NAME, NO\_OF\_COPIES

FROM BOOK B, BOOK\_AUTHORS BA, BOOK\_COPIES BC, LIBRARY\_BRANCH LB WHERE B.BOOK\_ID = BA.BOOK\_ID AND

BA.BOOK\_ID = BC.BOOK\_ID AND BC.BRANCH\_ID = LB.BRANCH\_ID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BRANCH\_NAME | BOOK\_ID TITLE | PUBLISHER\_NAME | AUTHOR\_NAME | NO\_OF\_COPIES |
| GENERAL | 4444 ENCYCLOPEDIA | - SAPNA | THOMAS | -  3 |
| MEDICAL | 3333 ANOTOMY | PEARSON | HENRY GRAY | 6 |
| CHILDREN | 4444 ENCYCLOPEDIA | SAPNA | THOMAS | 10 |
| CENTRAL TECHNICAL | 1111 SE | PEARSON | SOMMERVILLE | 5 |
| CENTRAL TECHNICAL | 2222 DBMS | MCGRAW | NAVATHE | 12 |

1. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.

**SELECT** CARD\_NO

**FROM** BOOK\_LENDING

**WHERE** DATE\_OUT BETWEEN '01-JAN-2017' AND '30-JUN-2017'

**GROUP BY** CARD\_NO

**HAVING** COUNT(\*) > 3; CARD\_NO

1

1. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

**DELETE FROM** BOOK

**WHERE** BOOK\_ID = '3333';

1 row deleted.

SQL> SELECT \* FROM BOOK;

BOOK\_ID

TITLE

PUBLISHER\_NAME PUB\_YEAR

- -- --

1111 SE PEARSON 2005

2222 DBMS MCGRAW 2004

4444 ENCYCLOPEDIA SAPNA 2010

SQL> SELECT \* FROM

BOOK\_COPIES; BOOK\_ID BRANCH\_ID

NO\_OF\_COPIES

-

|  |  |  |
| --- | --- | --- |
| 1111 | 11 | 5 |
| 4444 | 33 | 10 |
| 2222 | 11 | 12 |
| 4444 | 55  SQL> SELECT | 3  \* FROM |

BOOK\_LENDING;

BOOK\_ID BRANCH\_ID CARD\_NO DATE\_OUT DUE\_DATE

-- -

20-AUG-

2222 11 1 10-JAN-17 17

09-AUG-

4444 55 1 11-APR-17 17

19-AUG-

2222 11 5 09-AUG-17 17

15-AUG-

4444 33 1 10-JUN-17 17

10-JUN-

1111 11 1 12-MAY-17 17

1. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

**CREATE VIEW V\_PUBLICATION AS SELECT** PUB\_YEAR

**FROM** BOOK;

SELECT \* FROM V\_PUBLICATIONS;

PUB\_YEAR

2004

2005

2010

2010

1. Create a view of all books and its number of copies that are currently available in the Library.

**CREATE VIEW** BOOKS\_AVAILABLE **AS SELECT** B.BOOK\_ID, B.TITLE, C.NO\_OF\_COPIES

**FROM** LIBRARY\_BRANCH L, BOOK B, BOOK\_COPIES C **WHERE** B.BOOK\_ID = C.BOOK\_ID AND L.BRANCH\_ID=C.BRANCH\_ID;

View created.

SQL> SELECT \* FROM BOOKS\_AVAILABLE;

|  |  |  |
| --- | --- | --- |
| BOOK\_I D | TITLE | NO\_OF\_COPIES |
| 1111 | SE | 5 |
| 3333 | ANOTOMY | 6 |
| 4444 | ENCYCLOPEDIA | 10 |
| 2222 | DBMS | 12 |
| 4444 | ENCYCLOPEDIA | 3 |