Vivekanand Education Society's

Institute of Technology

(Affiliated to University of Mumbai, Approved by AICTE & Recognized by Govt. of Maharashtra)

Since 1962

Department of Artificial Intelligence and Data Science

Database Management Systems Labs Journal

Roll No.	36
Name	Madhusudhana Naidu
Class	D6AD
Subject	DBMS Lab
Grade:	

1. Lab Objectives:

	Description
1	To learn the basic concepts of Object-oriented Programming
2	To study Java Programming Language.
3	To study various concepts of Java Programming like multithreading, exception Handling, packages etc
4	To explain components of GUI based programming

2. Lab Outcome:

LO	Description
LO 1	Design ER /EER diagrams and convert to relational models for the real world application.
LO 2	Apply DDL, DML, DCL and TCL commands
LO 3	Write simple and complex queries
LO 4	Use PL / SQL Constructs.
LO 5	Demonstrate the concept of concurrent transactions execution and frontend-backend connectivity

3. LO/PO Mapping

LO	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
LO1	**1**	**3**	**2**	**1**	**1**	**1**	**2**	**1**	**1**	**2**	**2**	**1**
LO2	**-**	**3**	**3**	**2**	**2**	**1**	**2**	**1**	**2**	**3**	**2**	**2**
LO3	**-**	**3**	**3**	**2**	**2**	**1**	**2**	**1**	**2**	**3**	**1**	**3**
LO4	**-**	**2**	**2**	**1**	**2**	**-**	**1**	**1**	**-**	**1**	**-**	**2**
LO5	**1**	**3**	**2**	**2**	**2**	**1**	**2**	**1**	**1**	**3**	**1**	**2**

DBMS Lab

Name: Madhusudhana Naidu

INDEX

List of Experiments:

Sr No	List of Experiments	LO
1	Identify the case study and detail **statement of problem** and draw **Entity-Relationship (ER) / Extended Entity-Relationship (EER)** Model Using Draw.io.	LO1
2.	Mapping **ER/EER to the Relational schema model and creating a schema diagram for your system.**	LO1, LO2
3	Create a database using **Data Definition Language (DDL)** and apply required **Integrity Constraints** for the specified system.	LO2, LO3
4	Populate database using **DML Commands** for your specified System.	LO2, LO3
5	Perform **Simple queries, string manipulation operations.**	LO2, LO3
6	Write **Nested queries using (in, not in, some, any, exist, not exist, with clause)**	LO2, LO3
7	Implement various types of **Joins and Views.**	LO3
8	Demonstrate **DCL** and **TCL** commands.	LO2
9	Implementation of **Functions and Stored Procedure** in PL-SQL.	LO4
10	Implement different types **of triggers.**	LO4
11	Implementation and demonstration of **Transaction and Concurrency control** techniques using locks.	LO2, LO4, LO5
12	Implement Database Connectivity (JDBC.ODBC)	LO2, LO3, LO5

Page No.
Date

Experiment - 1

Consider a university database, for a university registers office, courses include title, credits, number.
Students include course number, semester, time, class, room
Instructor includes, id, name, department, title.

ER: ER model stands for Entity Relationship model. It is a high level data model. It is used to define data elements and relationships for a specific system.

Components of ER diagram:
Entity: It may be an object, class, person or place, represented by triangles.
Weak entity: Depends on another entity, doesn't contain any primary key of its own, represented by double rectangle.

Attribute: It is used to define the property of an entity. represented by an eclipse. Eg: id, age, name, etc.
Key attribute: Used to represent the main characteristic of an entity. It represents a primary key, represented by an eclipse with underlined text.
Composite attribute: An attribute composed of many other attributes, represented by an eclipse, and are connected with an eclipse.
Multivalued attribute: An attribute that can have more than one value, represented by double oval. Eg: phone no.
Derived attribute: An attribute that can be derived from other attributes, can be represented by a dotted attribute.

Teacher's Sign.:

Page No.
Date

Relationship:

It is used to describe the relation between entities. Diamond or Rhombus is used to represent relationship.

a) One to one: One instance of an entity is associated with the relationship. Eg

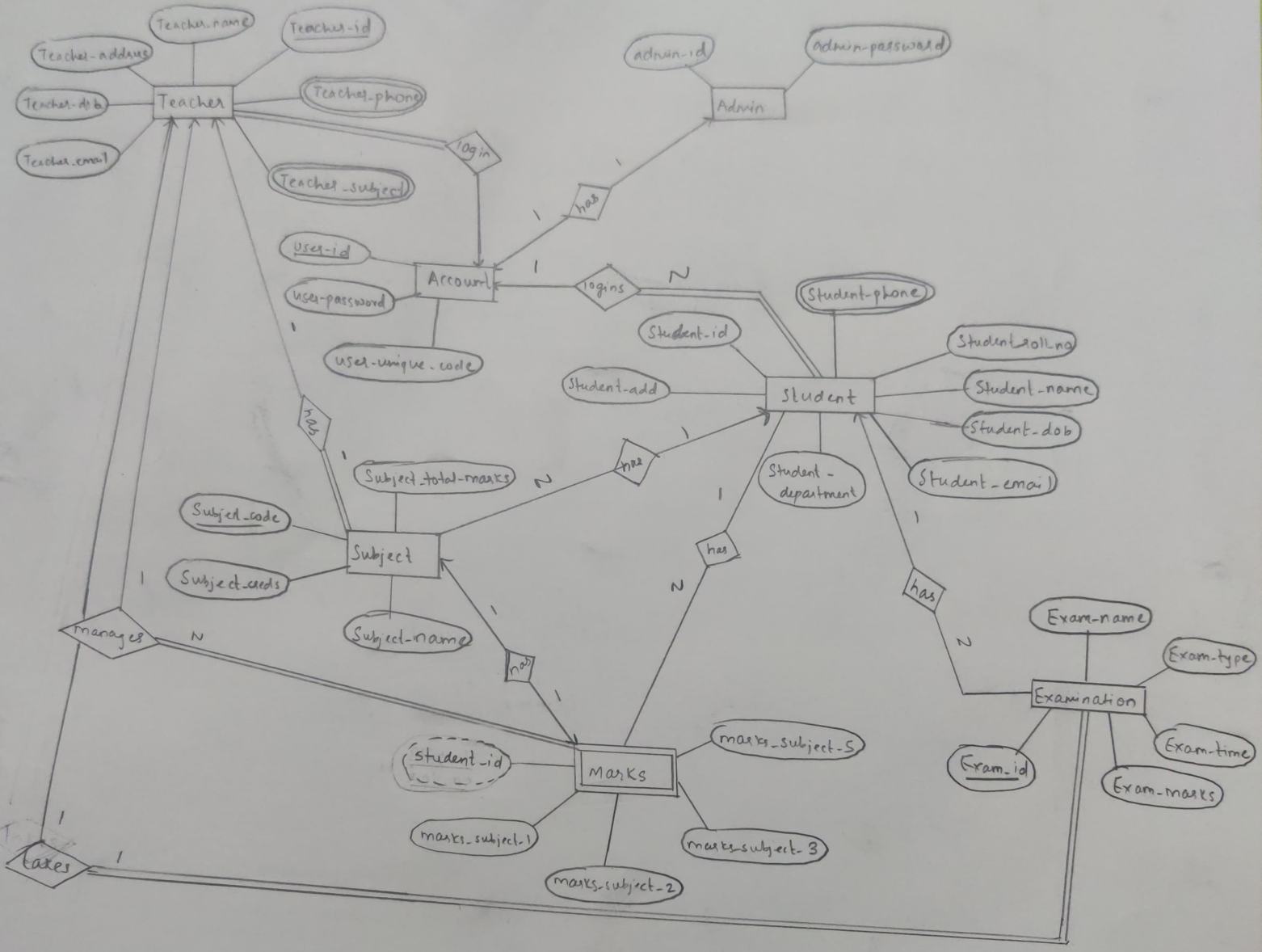
b) One to many: When only one instance of the entity one the left and more than one instance of an entity on the right. Eg one scientist can have many inventions.

c) Many to one: Many instances of entity on the left, and only one instance of an entity on the right.

d) Many to many: More than one instance of the entity on the left, and more than one instance of the entity on the right.

Teacher's Sign.: _____

ER Model for College Management Process



Experiment :- 02 (ER to Relational Schema)

All the table contents and their Commands :-

1. create table Admin (admin_id varchar(10) primary key not null, admin_password varchar(50) not null);
insert into Admin values ('Trustee','NewPassword');
insert into Admin values ("Principal","IamPrincipal");

2. create table Account(user_id int primary key, user_password varchar(20), user_unique_code varchar(20) not null);
insert into Account values(1001,'IamNew','HiIam1001');
insert into Account values(1002,'IamNew1','HiIam1002');
insert into Account values(1003,'IamNew2','HiIam1003');
insert into Account values(1004,'IamNew3','HiIam1004');

3. create table subject (subject_code int primary key, subject_creds int, subject_name varchar(30), subject_total_marks int);
insert into subject values(101, 150,'DBMS',125);
insert into subject values(102, 120,'OS',125);
insert into subject values(103, 120,'MP',125);
insert into subject values(104, 150,'AOA',125);
insert into subject values(105, 200,'Maths',125);

4. create table Teacher(teacher_id int primary key, teacher_name varchar(30), teacher_address varchar(50), teacher_dob varchar(20), teacher_email varchar(40), teacher_phoneno varchar(20), teacher_subject varchar(30), subject_code int foreign key (subject_code) references subject);
insert into teacher
values(111,'XYZ','Kurla','15-08-1980','xyz@ves.ac.in','9988776655','DBMS',101);
insert into teacher
values(222,'ABC','Thane','18-04-1975','abc@ves.ac.in','9182736455','OS',102);
insert into teacher
values(333,'PQR','Dadar','23-05-1979','pqr@ves.ac.in','9283746556','MP',103);
insert into teacher
values(444,'RTY','Chembur','18-11-1989','rty@ves.ac.in','9123456789','AOA',104);
insert into teacher
values(555,'ASD','Sion','20-08-1979','asd@ves.ac.in','9564728273','Maths',105);

5. create table Students(student_id int not null primary key, student_full_name varchar(50), student_phoneno varchar(50), student_email varchar(50), student_address varchar(100), student_rollno int, student_dob date, student_department varchar(10));
insert into students values(612, 'Manav Inder Pahilwani','9588660397','2020.manav.pahilwani@ves.ac.in','Flat 4 & 5 Achal Residency, Malegaon',37,12-12-2002,3001);
insert into students values(614, 'Om Gaydhane','8369000474','2020.om.gaydhane@ves.ac.in','Airoli',15,14-02-2002,3001);
insert into students values(645, 'Madhusudhana Naidu','9321451547','2020.madhusudhana.naidu@ves.ac.in','Seawoods',36,31-4-2002,3001);

```
insert into students values(669, 'Akshat
Tiwari','9876543210','2020.akshant.tiwari@ves.ac.in','Chembur',62,21-08-2002,3001);
```

6. create table Examination (exam_id int primary key, exam_name varchar(10), exam_time varchar(20),exam_marks int,exam_mode varchar(10));

```
insert into examination values(320, 'Maths-4','21-05-21 3:00:00',80,'ONLINE');
insert into examination values(310, 'DBMS','23-05-21 3:00:00',80,'ONLINE');
insert into examination values(330, 'AOA','25-05-21 3:00:00',80,'ONLINE');
insert into examination values(300, 'OS','27-05-21 3:00:00',80,'ONLINE');
insert into examination values(350, 'MP','29-05-21 3:00:00',80,'ONLINE');
```

7. create table Marks(student_id int primary key foreign key(student_id) references students, marks_sub1 int, marks_sub2 int, marks_sub3 int, marks_sub4 int, marks_sub5 int);

```
insert into marks values(612, 80,78,75,76,70);
insert into marks values(614, 79,65,78,80,78);
insert into marks values(645, 67,58,70,80,80);
insert into marks values(669, 60,60,60,60,59);
```

Experiment :- 02 (ER to Relational Schema)

```
1> select * from Admin;
2> go
admin_id   admin_password
---------- --------------------------------------------------
```

```
1> select * from Admin;
2> go
admin_id   admin_password
---------- --------------------------------------------------
Principal  IamPrincipal
Trustee    NewPassword
```

```
1> select * from Teacher;
2> go
teacher_id  teacher_name                teacher_address                                 teacher_dob      teacher_email                        teacher_phoneno teacher_subject
---------- ------------------------------ ---------------------------------------------- ---------------- ------------------------------------ --------------- ------------------------------

(0 rows affected)
1>
```

```
1> insert into Account values(1001,'IamNew','HiIam1001'
2> );
3> go

(1 rows affected)
1> insert into Account values(1002,'IamNew1','HiIam1002');
2> go

(1 rows affected)
1> insert into Account values(1003,'IamNew2','HiIam1003');
2> go

(1 rows affected)
1> insert into Account values(1004,'IamNew3','HiIam1004');
2> go

(1 rows affected)
1> select * from Account;
2> go
user_id     user_password        user_unique_code
----------- -------------------- --------------------
       1001 IamNew               HiIam1001
       1002 IamNew1              HiIam1002
       1003 IamNew2              HiIam1003
       1004 IamNew3              HiIam1004

(4 rows affected)
```

```
1> insert into examination values(320, 'Maths-4','21-05-21 3:00:00',80,'ONLINE');
2> go

(1 rows affected)
1> insert into examination values(310, 'DBMS','23-05-21 3:00:00',80,'ONLINE');
2> go

(1 rows affected)
1> insert into examination values(330, 'AOA','25-05-21 3:00:00',80,'ONLINE');
2> go

(1 rows affected)
1>  insert into examination values(300, 'OS','27-05-21 3:00:00',80,'ONLINE');
2> go

(1 rows affected)
1>  insert into examination values(350, 'MP','29-05-21 3:00:00',80,'ONLINE');
2> go

(1 rows affected)
1> select * from examination;
2> go
exam_id     exam_name  exam_time             exam_marks  exam_mode
----------- ---------- --------------------- ----------- ----------
        300 OS         27-05-21 3:00:00               80 ONLINE
        310 DBMS       23-05-21 3:00:00               80 ONLINE
        320 Maths-4    21-05-21 3:00:00               80 ONLINE
        330 AOA        25-05-21 3:00:00               80 ONLINE
        350 MP         29-05-21 3:00:00               80 ONLINE

(5 rows affected)
```

```
1> select * from Teacher
2> ;
3> select * from Students;
4> select * from Examination;
5> select * from account;
6> select * from marks;
7> select * from admin;
8> select * from subject;
9> go
teacher_id  teacher_name                teacher_address                                   teacher_dob      teacher_email                        teacher_phoneno teacher_subject             subject_code
----------- --------------------------- ------------------------------------------------- ---------------- ------------------------------------ --------------- --------------------------- ------------


(0 rows affected)
student_id  student_full_name                              student_phoneno student_email                                   student_address
                 student_rollno student_dob      student_department
----------- ---------------------------------------------- --------------- ------------------------------------------------ ------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------- --------------- ---------------- ------------------


(0 rows affected)
exam_id     exam_name  exam_time             exam_marks  exam_mode
----------- ---------- --------------------- ----------- ----------

(0 rows affected)
user_id     user_password       user_unique_code
----------- ------------------- -------------------

(0 rows affected)
student_id  marks_sub1  marks_sub2  marks_sub3  marks_sub4  marks_sub5
----------- ----------- ----------- ----------- ----------- -----------

(0 rows affected)
admin_id   admin_password
---------- --------------------------------------------------
Principal  IamPrincipal
Trustee    NewPassword

(2 rows affected)
subject_code subject_creds subject_name                subject_total_marks
------------ ------------ ------------------------------ -------------------

(0 rows affected)
```

```
1> insert into marks values(612, 80,78,75,76,70);
2> go

(1 rows affected)
1> insert into marks values(614, 79,65,78,80,78);
2> go

(1 rows affected)
1> insert into marks values(645, 67,58,70,80,80);
2> go

(1 rows affected)
1> insert into marks values(669, 60,60,60,60,59);
2> go

(1 rows affected)
1> select * from marks;
2> go
student_id  marks_sub1  marks_sub2  marks_sub3  marks_sub4  marks_sub5
----------- ----------- ----------- ----------- ----------- -----------
         612          80          78          75          76          70
         614          79          65          78          80          78
         645          67          58          70          80          80
         669          60          60          60          60          59
```

```
1> select * from Admin;
2> go
admin_id   admin_password
---------- ------------------------------------------------------------
Principal  IamPrincipal
Trustee    NewPassword

(2 rows affected)
1> select * from Account;
2> go
user_id     user_password       user_unique_code
----------- ------------------- -------------------
      1001 IamNew              Hilam1001
      1002 IamNew1             Hilam1002
      1003 IamNew2             Hilam1003
      1004 IamNew3             Hilam1004

(4 rows affected)
1> select * from teacher;
2> go
teacher_id  teacher_name                teacher_address                                    teacher_dob         teacher_email                      teacher_phoneno     teacher_subject             sub
ject_code
----------- --------------------------- -------------------------------------------------- ------------------- ---------------------------------- ------------------- --------------------------- ---
---------
        111 XYZ                         Kurla                                              15-08-1980          xyz@ves.ac.in                      9988776655          DBMS
     101
        222 ABC                         Thane                                              18-04-1975          abc@ves.ac.in                      9182736455          OS
     102
        333 PQR                         Dadar                                              23-05-1979          pqr@ves.ac.in                      9283746556          MP
     103
        444 RTY                         Chembur                                            18-11-1989          rty@ves.ac.in                      9123456789          AOA
     104
        555 ASD                         Sion                                               20-08-1979          asd@ves.ac.in                      9564728273          Maths
     105

(5 rows affected)
1> select * from Subject
2> ;
3> go
subject_code subject_creds subject_name                 subject_total_marks
------------ ------------- ---------------------------- -------------------
         101           150 DBMS                                         125
         102           120 OS                                           125
         103           120 MP                                           125
         104           150 AOA                                          125
         105           200 Maths                                        125
```

Experiment :- 02 (ER to Relational Schema)

```
1> select * from students;
2> go
student_id  student_full_name                                   student_phoneno                                    student_email                                      student_address
                                                       student_rollno student_dob          student_department
----------- --------------------------------------------------- -------------------------------------------------- -------------------------------------------------- --------------------------------------------------
--------------------------------------------------- -------------- -------------------- ------------------
        612 Manav Inder Pahilwani                               9588660397                                         2020.manav.pahilwani@ves.ac.in                     Flat 4 & 5 Achal Residency, Malegaon
                                                                    37 -2002           3001
        614 Om Gaydhane                                         8369000474                                         2020.om.gaydhane@ves.ac.in                         Airoli
                                                                    15 -1990           3001
        645 Madhusudhana Naidu                                  9321451547                                         2020.madhusudhana.naidu@ves.ac.in                  Seawoods
                                                                    36 -1975           3003
        669 Akshat Tiwari                                       9876543210                                         2020.akshant.tiwari@ves.ac.in                      Chembur
                                                                    62 -1989           3001

(4 rows affected)
1> select * from examination;
2> go
exam_id     exam_name  exam_time            exam_marks  exam_mode
----------- ---------- -------------------- ----------- ----------
        300 OS         27-05-21 3:00:00              80 ONLINE
        310 DBMS       23-05-21 3:00:00              80 ONLINE
        320 Maths-4    21-05-21 3:00:00              80 ONLINE
        330 AOA        25-05-21 3:00:00              80 ONLINE
        350 MP         29-05-21 3:00:00              80 ONLINE

(5 rows affected)
1> select * from marks;
2> go
student_id  marks_sub1  marks_sub2  marks_sub3  marks_sub4  marks_sub5
----------- ----------- ----------- ----------- ----------- -----------
        612          80          78          75          76          70
        614          79          65          78          80          78
        645          67          58          70          80          80
        669          60          60          60          60          59

(4 rows affected)
1>
```

```
1> insert into students values(612, 'Manav Inder Pahilwani','9588660397','2020.manav.pahilwani@ves.ac.in','Flat 4 & 5 Achal Residency, Malegaon',37,12-12-2002,3001);
2> go

(1 rows affected)
1> insert into students values(614, 'Om Gaydhane','8369000474','2020.om.gaydhane@ves.ac.in','Airoli',15,14-02-2002,3001);
2> go

(1 rows affected)
1> insert into students values(645, 'Madhusudhana Naidu','9321451547','2020.madhusudhana.naidu@ves.ac.in','Seawoods',36,31-4-2002,3003);
2> go

(1 rows affected)
1> insert into students values(669, 'Akshat Tiwari','9876543210','2020.akshant.tiwari@ves.ac.in','Chembur',62,21-08-2002,3001);
2> go

(1 rows affected)
1> select * from students;
2> go
student_id  student_full_name                                   student_phoneno                                    student_email                                      student_address
                                                       student_rollno student_dob          student_department
----------- --------------------------------------------------- -------------------------------------------------- -------------------------------------------------- --------------------------------------------------
--------------------------------------------------- -------------- -------------------- ------------------
        612 Manav Inder Pahilwani                               9588660397                                         2020.manav.pahilwani@ves.ac.in                     Flat 4 & 5 Achal Residency, Malegaon
                                                                    37 -2002           3001
        614 Om Gaydhane                                         8369000474                                         2020.om.gaydhane@ves.ac.in                         Airoli
                                                                    15 -1990           3001
        645 Madhusudhana Naidu                                  9321451547                                         2020.madhusudhana.naidu@ves.ac.in                  Seawoods
                                                                    36 -1975           3003
        669 Akshat Tiwari                                       9876543210                                         2020.akshant.tiwari@ves.ac.in                      Chembur
                                                                    62 -1989           3001
```

```
1> insert into subject values(101, 150,'DBMS',125);
2> go

(1 rows affected)
1> insert into subject values(102, 120,'OS',125);
2> go

(1 rows affected)
1> insert into subject values(103, 120,'MP',125);
2> go

(1 rows affected)
1> insert into subject values(104, 150,'AOA',125);
2> go

(1 rows affected)
1> insert into subject values(105, 200,'Maths',125);
2> go

(1 rows affected)
1> select * from subject;
2> go
subject_code subject_creds subject_name                    subject_total_marks
------------ ------------- ------------------------------ -------------------
         101           150 DBMS                                           125
         102           120 OS                                             125
         103           120 MP                                             125
         104           150 AOA                                            125
         105           200 Maths                                          125

(5 rows affected)
```

```
1> insert into teacher values(111,'XYZ','Kurla','15-08-1980','xyz@ves.ac.in','9988776655','DBMS',101);
2> go

(1 rows affected)
1> insert into teacher values(222,'ABC','Thane','18-04-1975','abc@ves.ac.in','9182736455','OS',102);
2> go

(1 rows affected)
1> insert into teacher values(333,'PQR','Dadar','23-05-1979','pqr@ves.ac.in','9283746556','MP',103);
2> go

(1 rows affected)
1> insert into teacher values(444,'RTY','Chembur','18-11-1989','rty@ves.ac.in','9123456789','AOA',104);
2> go

(1 rows affected)
1> insert into teacher values(555,'ASD','Sion','20-08-1979','asd@ves.ac.in','9564728273','Maths',105);
2> go

(1 rows affected)
1> select * from teacher;
2> go
teacher_id  teacher_name                teacher_address                           teacher_dob         teacher_email                     teacher_phoneno     teacher_subject             subject_code
----------- --------------------------- ----------------------------------------- ------------------- --------------------------------- ------------------- --------------------------- ------------
        111 XYZ                         Kurla                                     15-08-1980          xyz@ves.ac.in                     9988776655          DBMS                        101
        222 ABC                         Thane                                     18-04-1975          abc@ves.ac.in                     9182736455          OS                          102
        333 PQR                         Dadar                                     23-05-1979          pqr@ves.ac.in                     9283746556          MP                          103
        444 RTY                         Chembur                                   18-11-1989          rty@ves.ac.in                     9123456789          AOA                         104
        555 ASD                         Sion                                      20-08-1979          asd@ves.ac.in                     9564728273          Maths                       105

(5 rows affected)
1>
```

```
1> create table Examination (exam_id int primary key, exam_name varchar(10), exam_time time,exam_marks int,exam_mode varchar(10));
2> go
1> select * from Examination
2> ;
3> go
exam_id     exam_name  exam_time             exam_marks  exam_mode
----------- ---------- --------------------- ----------- ----------

(0 rows affected)
1>
```

Experiment :- 02 (ER to Relational Schema)

```
1> select * from students where student_full_name like '%____%'
2> go
student_id  student_full_name                                     student_phoneno                               student_email                                      student_address
                                                          student_rollno student_dob          student_department
----------- ----------------------------------------------------- -----------------------------------------  -------------------------------------------------- --------------------------------------------------
--------------------------------------------------- -------------- ------------------- ------------------
        612 Manav Inder Pahilwani                                  9588660397                                    2020.manav.pahilwani@ves.ac.in                     Flat 4 & 5 Achal Residency, Malegaon
                                                                       37 -2002            3001
        614 Om Gaydhane                                            8369000474                                    2020.om.gaydhane@ves.ac.in                         Airoli
                                                                       15 -1990            3001
        645 Madhusudhana Naidu                                     9321451547                                    2020.madhusudhana.naidu@ves.ac.in                  Seawoods
                                                                       36 -1975            3003
        669 Akshat Tiwari                                          9876543210                                    2020.akshant.tiwari@ves.ac.in                      Chembur
                                                                       62 -1989            3001

(4 rows affected)
```

```
1> select * from students where student_full_name like 'M%'
2> go
student_id  student_full_name                                     student_phoneno                               student_email                                      student_address
                                                          student_rollno student_dob          student_department
----------- ----------------------------------------------------- -----------------------------------------  -------------------------------------------------- --------------------------------------------------
--------------------------------------------------- -------------- ------------------- ------------------
        612 Manav Inder Pahilwani                                  9588660397                                    2020.manav.pahilwani@ves.ac.in                     Flat 4 & 5 Achal Residency, Malegaon
                                                                       37 -2002            3001
        645 Madhusudhana Naidu                                     9321451547                                    2020.madhusudhana.naidu@ves.ac.in                  Seawoods
                                                                       36 -1975            3003

(2 rows affected)
```

```
1> select upper(student_full_name) from students;
2> go

--------------------------------------------------
MANAV INDER PAHILWANI
OM GAYDHANE
MADHUSUDHANA NAIDU
AKSHAT TIWARI

(4 rows affected)
```

```
1> select lower(student_full_name) from students;
2> go

--------------------------------------------------
manav inder pahilwani
om gaydhane
madhusudhana naidu
akshat tiwari

(4 rows affected)
```

```
1> select * from marks where marks_sub1 not between 70 and 80;
2> go
student_id  marks_sub1  marks_sub2  marks_sub3  marks_sub4  marks_sub5
----------- ----------- ----------- ----------- ----------- -----------
        645          67          58          70          80          80
        669          60          60          60          60          59
```

```
1> select * from teacher as t, subject as s where t.subject_code=s.subject_code;
2> go
teacher_id  teacher_name                teacher_address                                  teacher_dob          teach
er_email                          teacher_phoneno      teacher_subject              subject_code subject_code subjec
t_creds subject_name                 subject_total_marks
---------- ------------------------------ ---------------------------------------------------- -------------------- -----
---------------------------------- ------------------ ------------------------------ ----------- ----------- ------
------- ------------------------------ ------------------
       111 XYZ                            Kurla                                            15-08-1980           xyz@v
es.ac.in                          9988776655           DBMS                                   101         101
    150 DBMS                                       125
       222 ABC                            Thane                                            18-04-1975           abc@v
es.ac.in                          9182736455           OS                                     102         102
    120 OS                                         125
       333 PQR                            Dadar                                            23-05-1979           pqr@v
es.ac.in                          9283746556           MP                                     103         103
    120 MP                                         125
       444 RTY                            Chembur                                          18-11-1989           rty@v
es.ac.in                          9123456789           AOA                                    104         104
    150 AOA                                        125
       555 ASD                            Sion                                             20-08-1979           asd@v
es.ac.in                          9564728273           Maths                                  105         105
    200 Maths                                      125

(5 rows affected)
```

```
1> select sum(marks_sub1), max(marks_sub1),student_id from marks group by student_id;
2> go
                        student_id
----------- ----------- -----------
         80          80         612
         79          79         614
         67          67         645
         60          60         669
```

```
1> select avg(marks_sub1), avg(marks_sub2),avg(marks_sub3), avg(marks_sub4), avg(marks_sub4) from marks;
2> go

----------- ----------- ----------- ----------- -----------
        71          65          70          74          74
```

```
1> select student_id, max(marks_sub1) from marks group by student_id;
2> go
student_id
----------- -----------
        612          80
        614          79
        645          67
        669          60
```

```
1> select * from teacher where subject_code in(select subject_code from subject) order by teacher_id;
2> go
teacher_id  teacher_name                teacher_address                                  teacher_dob          teach
er_email                          teacher_phoneno      teacher_subject               subject_code
----------- ------------------------------ ---------------------------------------------------- -------------------- -----
------------------------------------ -------------------- ------------------------------ ------------

        111 XYZ                        Kurla                                            15-08-1980           xyz@v
es.ac.in                          9988776655           DBMS                             101
        222 ABC                        Thane                                            18-04-1975           abc@v
es.ac.in                          9182736455           OS                               102
        333 PQR                        Dadar                                            23-05-1979           pqr@v
es.ac.in                          9283746556           MP                               103
        444 RTY                        Chembur                                          18-11-1989           rty@v
es.ac.in                          9123456789           AOA                              104
        555 ASD                        Sion                                             20-08-1979           asd@v
es.ac.in                          9564728273           Maths                            105

(5 rows affected)
```

```
1> create table Marks( student_id int primary key foreign key(student_id) references students, marks_sub1 int, marks_sub2 int, marks_sub3 int, marks_sub4 int, marks_sub5 int);
2> go
1> select * from table Marks;
2> go
Msg 156, Level 15, State 1, Server MANNS-LAPTOP, Line 1
Incorrect syntax near the keyword 'table'.
1> select * from Marks;
2> go
student_id  marks_sub1  marks_sub2  marks_sub3  marks_sub4  marks_sub5
----------- ----------- ----------- ----------- ----------- -----------

(0 rows affected)
```

Madhusudhana Naidu
D6AD
36
SPB
DATE :

Experiment - 3

Aim:
Create a database using Data Definition Language (DDL) and apply integrity constraints for the specified system.

Theory:
Data Definition Language deals with the structure of the database where the data is to be stored. It does not deal with the data itself. Thus, in any structured query language, a command that can modify the structures or the tables or relations of the database, is a DDL command.

Examples of DDL commands:

- CREATE - is used to create the database or its objects (like table, index, function, views, stole procedure and triggers)
- DROP - is used to delete objects from the database.
- ALTER - is used to alter the structure of the database.
- TRUNCATE - is used to remove all records from a table, including all spaces allocated for the records are removed
- RENAME - is used to rename an object existing in the database.

SPB
DATE :

Integrity Constraints:

- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- They ensure that the data insertion, updating and other processes have to be performed in such a way that data integrity is not affected.

Types of Integrity constraints:

- Domain constraints - is definition of a valid set of values for an attribute.
- Entity integrity constraints - it states that primary key value can't be null.
- Referential integrity constraints - it is specified between two tables, it states if a foreign key in Table 1 refers to a primary key in table 2, then every value of the foreign key in Table 1 should be either null or available in Table 2.

Key constraint - keys are the entity set that is used to identify an entity within its entity set uniquely.

QUERY EXECUTION:

1. Create Tablespace

```
SQL> create tablespace tbs
  2  datafile 'tbs.dat'
  3  size 50m
  4  online;

Tablespace created.
```

2. Create User

```
SQL> create user madhu identified by madhu;

User created.
```

3. Grant privileges to user

```
SQL> grant connect, resource to madhu;

Grant succeeded.
```

DDL Commands:

Table customer:

1.create command

```
SQL> create table customer(cid int,cname varchar(20),address char(10));

Table created.
```

2.Alter command (add | alter column | drop column)

adding new column in existing table

```
SQL> alter table customer add phno numeric(10);

Table altered.
```

Modifying existing column

```
SQL> alter table customer modify address varchar(20);

Table altered.
```

dropping column from existing table

```
SQL> alter table customer drop column address;

Table altered.
```

Table Employee

```
SQL> create table employee(ssn int primary key check(ssn like ' '), ename varchar(20) not null, salary int,dno int default 5);

Table created.
```

Table Department

```
SQL> create table dept(dno int primary key, dname varchar(20), mgrssn int references employee, startdate date);

Table created.
```

applying foreign key constraint on existing table

```
SQL> alter table employee add constraint fk_dno foreign key(dno) references dept(dno);

Table altered.
```

Table deptlocation

```
SQL> create table deptloc(dno int, dloc varchar(20), primary key(dno,dloc),foreign key(dno) references dept(dno));

Table created.
```

Table Project

```
SQL> create table project(pno int primary key, pname varchar(20), dno int references dept(dno));

Table created.
```

Conclusion:

Thus we have successfully implemented all DDL commands.

Madhusudhana Naidu

D6AD 36

DBMS Lab

Experiment 4

Madhusudhana Naidu
D6AD
36
SPB
DATE :

Experiment - 4

Aim:
To Study the Data Manipulation Language Commands

Theory:
Data Manipulation Language (DML) is used to retrieve, insert and modify database information. These commands will be used by all database users during the routine operation of the database. Basic DML commands:

- INSERT -
INSERT command in SQL is used to add records to an existing table.
Syntax: INSERT into table tablename values (values);
Example - insert into employee values ('bart', 'simpson', 12345, $45000);

- SELECT -
It is the most commonly used command in SQL. It allows database users to retrieve the specific information they desire from an operational database.

Syntax: select <attribute list> from <list of tables>
where predicate;
Example: select * from employee;
Alternatively users may want to limit the attributes that are retrieved from the database.
So we use, select last_name from employee
to retrieve only the last_name of all employees in employee table.

SPB

DATE :

The WHERE clause can be used to limit the records that are retrieved to those that meet specified criteria.

Select * from employee where salary > 50000 ;

- UPDATE -

The UPDATE command can be used to modify information contained within a table, either in bulk or individually.

Example:

Syntax : update employee set salary = salary * 1.03

- DELETE -
- The delete command with where clause can be used to remove specific record from employee table.

Syntax: delete from employee where employee_id = 12345

DML Commands:

- Insert-

```
SQL> INSERT INTO department values(3001, 'CMPN');

1 row created.

SQL> INSERT INTO department values(3002,'INSTR');

1 row created.

SQL> INSERT INTO department values(3003, 'ELEX');

1 row created.

SQL> INSERT INTO department values(3004,'ETRX');

1 row created.

SQL> INSERT INTO department values(3005, 'IT');

1 row created.
```

```
SQL> insert into book(book_id,book_name,book_issdn,book_author,dept_code)
  2  values(4001,'Database Management System',1234,'G.K.Gupta',3001);

1 row created.

SQL> INSERT INTO book values(4002,'Advanced Java', 156, 'Schildt', 56, 'Y', 'Tata McGraw
  2  Hill', 3001, 001);

1 row created.

SQL> INSERT INTO book values (4003, 'Microprocessors 8086', 007, 'Uffenbeck', 42, 'Y',
  2  'Wiley', 3003,003);

1 row created.
```

```
SQL> INSERT INTO book values(4005, 'Applied Mathematic', 987,'G.V Khumbojkar',100, 'Y',
  2  'G.V.Publication',3002,002);

1 row created.
```

```
SQL> insert into book values(4004,'Lets C',987,'Balguruswamy',0,'N','Indian Publication',3005,005);


1 row created.
```

```
SQL> INSERT INTO records(dept_code, book_id) values(3001,4001);

1 row created.

SQL> INSERT INTO records(dept_code,book_id) values(3002,4005);

1 row created.

SQL> INSERT INTO records(dept_code, book_id) values(3003,4003);

1 row created.

SQL> INSERT INTO records(dept_code,book_id) values(3001,4002);

1 row created.

SQL> INSERT INTO records(dept_code, book_id) values(3005,4004);

1 row created.
```

- Select-

```
SQL> select * from records;

   BOOK_ID DEPT_CODE
---------- ----------
      4001 3001
      4002 3001
      4005 3002
      4003 3003
      4004 3005
```

- Update-

```
SQL> update student set lib_id =1006,f_name='Ram', m_name='Mohan', l_name='Pyare';

1 row updated.
```

- Delete-

```
SQL> delete from staff where staff_id = 2003;

1 row deleted.
```

Conclusion:

Thus we have successfully implemented DML commands.

Madhusudhana Naidu

D6AD 36

DBMS Lab

Page No.
Date

Experiment - 5

Aim:
Perform simple queries, string manipulation operations.

Objective:
To understand simple queries and string manipulation operations.

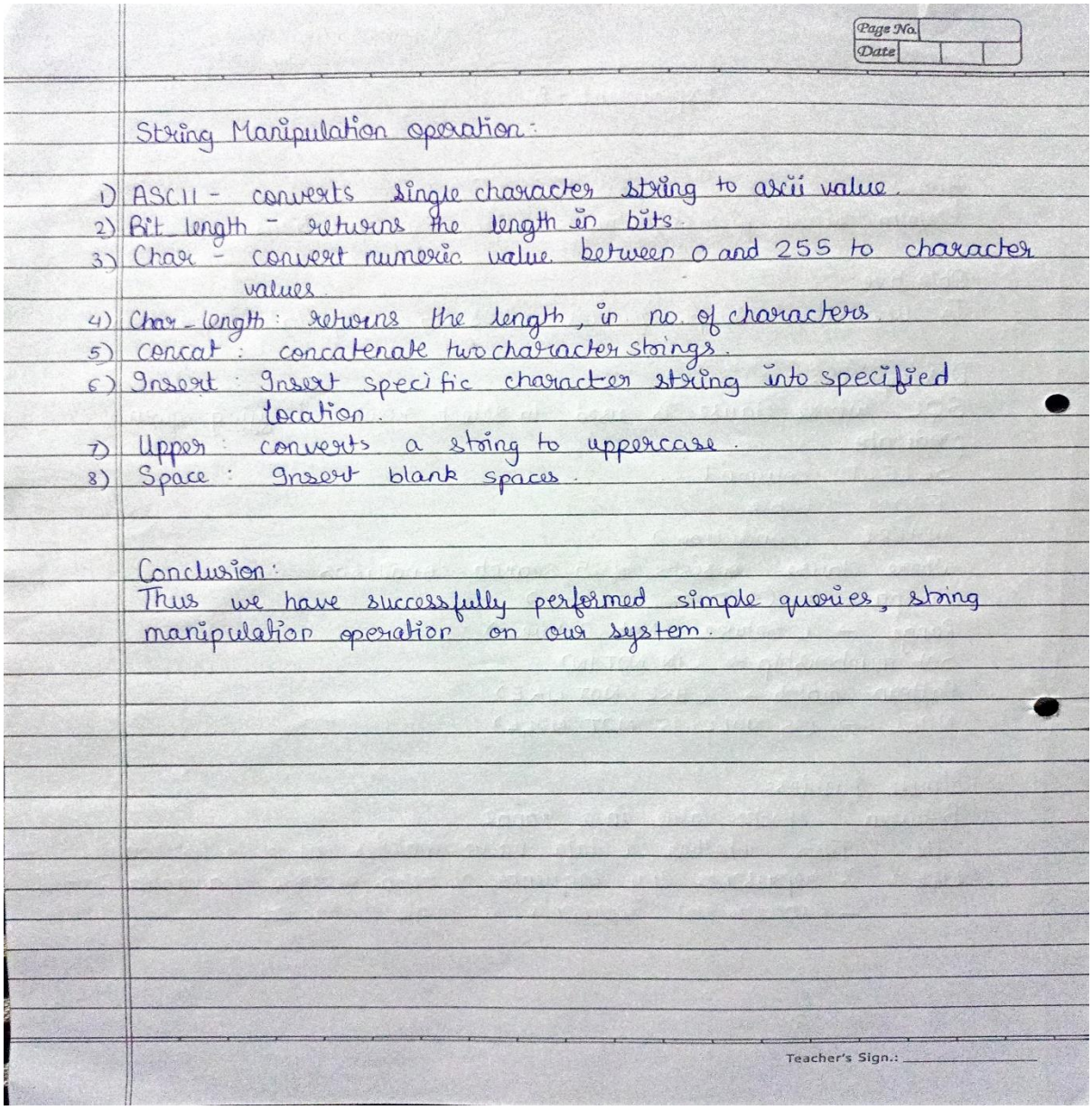
Descriptions:
SQL where clause is used to select rows satisfying given predicate.

```
SELECT  column1...
 FROM   table
 WHERE  (condition)
```

Where clause consists of 5 search conditions
Compare → (<, >, =, <=, >=)
Range → (Between / Not Between)
Set membership → (IN/NOT IN)
Pattern match → (LIKE / NOT LIKE)
NULL → (IS NULL / IS NOT NULL)

Simple Queries:
Between : checks value in a range
IN : Tests whether a data value matches one of a list value.
LIKE : % represents any sequence of zero or more character.
_ (underscore) represents a single character.

Teacher's Sign.:



String Manipulation operation:

1) ASCII - converts single character string to ascii value.
2) Bit length - returns the length in bits.
3) Char - convert numeric value between 0 and 255 to character values.
4) Char-length: returns the length, in no. of characters.
5) Concat: concatenate two character strings.
6) Insert: Insert specific character string into specified location.
7) Upper: converts a string to uppercase.
8) Space: Insert blank spaces.

Conclusion:
Thus we have successfully performed simple queries, string manipulation operation on our system.

Teacher's Sign.: _____

Command Snippets:

```
select * from employee;
```

eid	name	salary	address
101	John	45000.00	Houston
102	Smita	50000.00	Mumbai
103	Smith	65000.00	London
104	Neha	48000.00	Delhi

```
select * from employee where salary between 40000 and 50000;
```

	eid	name	salary	address
1	101	John	45000.00	Houston
2	102	Smita	50000.00	Mumbai
3	104	Neha	48000.00	Delhi

```
select * from employee where salary not between 40000 and 50000;
```

	eid	name	salary	address
1	103	Smith	65000.00	London

```
select * from employee where address in('Houston','Mumbai');
```

	eid	name	salary	address
1	101	John	45000.00	Houston
2	102	Smita	50000.00	Mumbai

```
select * from employee where address not in ('Houston');
```

	eid	name	salary	address
1	102	Smita	50000.00	Mumbai
2	103	Smith	65000.00	London
3	104	Neha	48000.00	Delhi

```
select * from employee where address like '%n';
```

	eid	name	salary	address
1	101	John	45000.00	Houston
2	103	Smith	65000.00	London

```
select * from employee where name like 'Smit_';
```

	eid	name	salary	address
1	102	Smita	50000.00	Mumbai
2	103	Smith	65000.00	London

```
select * from employee where salary >= 49000;
```

	eid	name	salary	address
1	102	Smita	50000.00	Mumbai
2	103	Smith	65000.00	London

```
select substring('Hello World',3,6);
```

	(No column name)
1	llo Wo

```
select trim('    Hello');
```

	(No column name)
1	Hello

```
select Lower(name) from employee;
```

	(No column name)
1	john
2	smita
3	smith
4	neha

```
select upper(name) from employee;
```

	(No column name)
1	JOHN
2	SMITA
3	SMITH
4	NEHA

Madhusudhana Naidu

D6AD 36

DBMS Lab

Experiment 6

Aim-
Write nested query for our system.

Objective-
To return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Description:
Nested query is a query within another sql query and embedded within the where clause. The data that is returned by subquery will be used by main query. It can be used with select, insert, update, delete and operators like =, <, >, >=, <=, <>, IN, between, etc.

Rules: must be enclosed within the parenthesis.
Can only have one column.
order by cannot be used in subquery.

Aggregate functions:
SUM() - It returns the sum of values from given set.
AVG() - It returns the average of values from given set.
Count() - It returns the number of values from a given set.
Min() - It returns the smallest value from a given set.
Max() - It returns the largest value from a given set.

Group By clause - It is used to create groups or group the records based on the criteria, it applies aggregate function on subgroups of some tuples

where subgroups are formed on some column value.

- Having clause - this clause is applied with group by, not with where.
  select column from table_name group by _ having _ ;

- Co-related query - Subquery that uses value from outer query. It contains a reference to value from row related by outer query.

  Nested subquery - runs only one for outer nesting query. It does not contain any reference to the outer query.
  exists and not exists - used to check whether the result of co-related query is empty or not, exists returns true if there is atleast one tuples, not exists returns true if there are no tuples in the output of the subquery.

  Derived relations - allows a subquery to be used in the from clause.
  Complex query : are hard to write as single sql block needs to compose multiple sql block to express complex query.

- Derived clause - occurs as nested select statements in the from clause of an outer select stat statement.
- With clause - Provides a way of defining a temporary view whose definition is available only to the query in which the clause occurs.

Command:

Query with '=' operator

```
select * from employee where dno = ( select dno from dept where dname = 'etrx');
```

ssn	ename	salary	superssn	dno
15	Nisha	40000.00	10	103

NULL

```
select * from employee where superssn is NULL;
```

ssn	ename	salary	superssn	dno
10	Neha	25000.00	NULL	102

Not In

```
select * from employee where dno not in (103,101);
```

ssn	ename	salary	superssn	dno
10	Neha	25000.00	NULL	102

Between

```
select ename from employee where salary between 35000 and 42000;
```

ename
Smita
Smit
Nisha

Less than

```
select ename from employee where salary < 40000;
```

ename
Smit
Neha

Order By

```
select * from employee order by superssn;
```

ssn	ename	salary	superssn	dno
10	Neha	25000.00	NULL	102
15	Nisha	40000.00	10	103
1	John	45000.00	10	101
2	Smita	42000.00	10	101
5	Smit	35000.00	15	101

Group By

```
select sum(salary) as totalsal from employee group by dno;
```

totalsal
122000.00
25000.00
40000.00

Sum, Count(*)

```
select dno, sum(salary) as totalsal,count(*) as no_of_emp from employee group by dno;
```

dno	totalsal	no_of_emp
101	122000.00	3
102	25000.00	1
103	40000.00	1

Avg()

```
select dno,avg(salary) as AverageSal from employee group by dno;
```

dno	AverageSal
101	40666.6666
102	25000.00
103	40000.00

Having

```
select dno, count(ssn) as no_of_emp from employee where salary between 30000 and 45000 group by dno having count(ssn)>=2;
```

dno	no_of_emp
101	3

Co-related

```
select * from employee as e1 where e1.salary > (select avg(salary) from employee as e2 where e1.dno = e2.dno);
```

ssn	ename	salary	superssn	dno
1	John	45000.00	10	101
2	Smita	42000.00	10	101

Nested subquery

```
select * from employee as e1 where e1.salary >= all(select salary from employee as e2 where e1.dno = e2.dno);
```

ssn	ename	salary	superssn	dno
1	John	45000.00	10	101
10	Neha	25000.00	NULL	102
15	Nisha	40000.00	10	103

Derived Relation

```
select dno from (select dno,count(*) from employee group by dno) as deptinfo(dno,noofemp) where noofemp > 1;
```

dno
101

Madhusudhana Naidu

D6AD 36

DBMS Lab

Page No.
Date

Experiment - 7

Aim:
Implement various types of joins and views.

Objective:
To combine records from two or more tables in a database.

Theory:
Join command is used to combine fields of table by using common values to each. It is used to retrieve data from 2 or more tables based on some logical relationship between tables.

SQL Join types:

- Inner Join - returns rows when there is a match in both tables.
- Left Join - returns all rows from the left table even if there are no matches in the right table.
- Right Join - returns all rows from the right table, even if there are no matches in the left table.
- Full Join - returns all the rows present in all the tables.
- Self Join - It is used to join the table to itself.
- Cartesian join - returns the cartesian product of records from two or more joined tables.
- Equi join - rows satisfying the selection criteria from both joined tables are selected.

Teacher's Sign.: _____

Command Snippets:

```
select * from employee;
select * from dept;
```

ssn	ename	salary	superssn	dno
1	John	45000.00	10	101
2	Smita	42000.00	10	101
5	Smit	35000.00	15	101
10	Neha	25000.00	NULL	102
15	Nisha	40000.00	10	103

dno	dname
101	comp
102	it
103	etrx
105	inst

Without using JOIN keyword:

```
select e.ename, e.dno, dname from employee as e, dept as d where e.dno = d.dno;
```

ename	dno	dname
John	101	comp
Smita	101	comp
Smit	101	comp
Neha	102	it
Nisha	103	etrx

With JOIN keyword:

```
select e.ename, e.dno, dname from (employee as e join dept as d on e.dno = d.dno);
```

ename	dno	dname
John	101	comp
Smita	101	comp
Smit	101	comp
Neha	102	it
Nisha	103	etrx

Left Outer Join:

```
e.dno, dname from (employee as e left outer join dept as d on e.dno=d.dno);
```

ename	dno	dname
John	101	comp
Smita	101	comp
Smit	101	comp
Neha	102	it
Nisha	103	etrx

Right Outer Join:

```
e.dno,dname from(employee as e right outer join dept as d on e.dno=d.dno);
```

ename	dno	dname
John	101	comp
Smita	101	comp
Smit	101	comp
Neha	102	it
Nisha	103	etrx
NULL	NULL	inst

Full Outer Join:

```
e.dno, dname from (employee as e full outer join dept as d on e.dno=d.dno);
```

ename	dno	dname
John	101	comp
Smita	101	comp
Smit	101	comp
Neha	102	it
Nisha	103	etrx
NULL	NULL	inst

Self Join:

```
select e.ename as empname, s.ename as supervisorname from employee as e, employee as s where e.superssn = s.ssn;
```

empname	supervisorname
John	Neha
Smita	Neha
Smit	Nisha
Nisha	Neha

Equi Join:

```
select e.ename,d.dname from employee as e, dept as d where e.dno = d.dno;
```

ename	dname
John	comp
Smita	comp
Smit	comp
Neha	it
Nisha	etrx

Madhusudhana Naidu

D6AD 36

DBMS Lab

Experiment 8

Aim:
Implement conditional/loops and cursors in PL/SQL.

Theory:

1) PL/SQL is a block structured language. The programs are logical blocks that can contain any number of nested blocks.
2) It includes procedural language elements like conditions and loops. It allows declaration of constants, variable, procedures and functions, types and variable of those types and triggers.
3)

```
Declare:
    declaration section
Begin:
    Executable section
Exception:
    Errors handling section
End;
```

4) DDL is not allowed in PL/SQL block.
5) Cursors:
   i) Oracle creates a certain portion in the memory for every SQL query that is executed.
   ii) Using PL/SQL this portion in the memory can be given a name, it is basically a pointer to the context area and thus represents a structure in memory.
   iii) Cursors are of 2 types:
      a) Implicit
      b) Explicit

Teacher's Sign.:

Queries:

```
select * from employee;
```

ID	NAME	DEPT	SALARY
1	lili	manager	10000
2	rubina	analyst	8000
3	sonu	clerk	1500
4	howard	clerk	1200

Q1. To give raise to all employees earning a salary less than 1500.

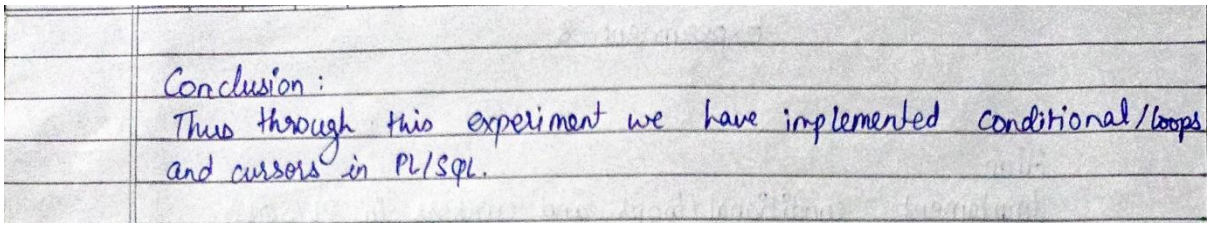
```
declare
    cursor employee_cur is
    select id, salary
    from employee
for update;
    incr_sal NUMBER;
begin
    for employee_rec in employee_cur loop
        if employee_rec.salary < 1500 then
    incr_sal := .15;
    else
    incr_sal := 0;
    end if;
    update employee
        set salary = salary + salary * incr_sal
        where current of employee_cur;
    end loop;
end;
```

ID	NAME	DEPT	SALARY
1	lili	manager	10000
2	rubina	analyst	8000
3	sonu	clerk	1500
4	howard	clerk	1380

Q2. To set job of all clerks with salary greater than 1300 as ‘Senior Clerk’.

```
declare
cursor empcursor is
select salary, dept
from employee;
sal employee.salary%type;
job employee.dept%type;
begin
open empcursor;
loop
fetch empcursor into sal,job;
exit when empcursor%notfound;
if sal > 1300 and job like 'clerk' then
update employee set dept = 'Senior Clerk' where sal > 1300 and dept like 'clerk';
end if;
end loop;
close empcursor;
end;
```

ID	NAME	DEPT	SALARY
1	lili	manager	10000
2	rubina	analyst	8000
3	sonu	Senior Clerk	1500
4	howard	Senior Clerk	1380



Madhusudhana Naidu

D6AD 36

DBMS Lab

Experiment 9

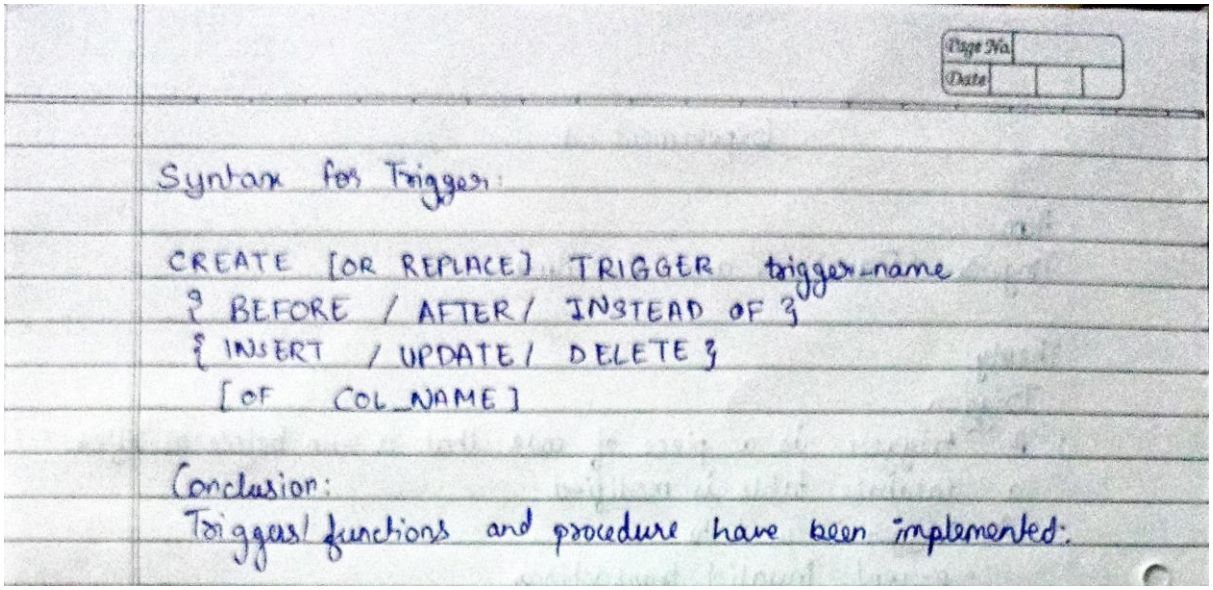
Aim:
Triggers / Functions and Procedure.

Theory:
Trigger

1. A trigger is a piece of code that is run before or after a database table is modified.
2. A trigger is used to
   - prevent invalid transactions
   - keep an audit trail of table
   - ensure rollback if database found inconsistent
3. A row trigger is fired each time a row in the table, is affected. It uses FOR EACH ROW clause, statement triggers are fired once on behalf of the statement, independent of the no. of rows the triggers statement affects.

* Procedure and Functions:

1. PL/SQL allows writing sub programs.
2. If we want to update data on table, a procedure is preferred while for retrieving info, a function is preferred.
3. A select SQL query can call a function but cannot call a procedure.
4. Procedure may return one or more values through parameters or may not return values at all. A function always returns a value using the return statement.



Queries:

Q1. Create a trigger that fires before insert or delete of a row in emp table and displays the count of rows

```
create trigger cnt
before insert or delete on employee
declare
val number;
begin
select count(*) into val from employee;
dbms_output.put_line('Number of rows are: '|| val);
end;
```

```
1  insert into employee values(7,'Roy','President',14000);
2
```

```
1 row(s) inserted.
Number of rows are: 4
```

Q2. Create a trigger that stops user from entering Dept no in emp table if that dept no doesn't exist in dept table.

```
create trigger insertcheck
before insert on employee
for each row
declare
val int;
begin
select count(*) into val from depart d where d.deptno = :new.deptno;
if val = 0 then
raise_application_error(-20101, 'Department does not exist');
else
dbms_output.put_line('Value Inserted');
end if;
end;
```

```
Trigger created.
```

```
1  insert into employee values(72,'James',105,12000);
2  insert into employee values(74,'Jim',103,12000);
3  select * from employee;
4
```

```
ORA-20101: Department does not exist ORA-06512: at "SQL_OUUHEMOQCFTDNUHHPRZJMCXUC.INSERTCHECK", line 6
ORA-06512: at "SYS.DBMS_SQL", line 1721


1 row(s) inserted.
Value Inserted
```

ID	NAME	DEPTNO	SALARY
74	Jim	103	12000

Q1. Write a procedure that:

Accepts department number and percentage of raise in sal

Updates the sal of all those employee under that department.

```
1  create procedure bonus(deptno in int, inc in float)
2  as
3  begin
4  update employee set salary = salary + salary * inc where deptno = deptno;
5  dbms_output.put_line('Value updated');
6  end;
```

```
Procedure created.
```

```
1  EXEC bonus(74,0.15);
2
```

```
Statement processed.
Value updated
```

```
1  select * from employee where id = 74;
2
```

ID	NAME	DEPTNO	SALARY
74	Jim	103	13800

Q2. Write a function that accepts :

Dept no and returns total sal of all employees in that department.

```
1  create or replace function raise
2  return number is
3  total employee.salary%type;
4  begin
5  select sum(salary) into total from employee where deptno = 103;
6  return (total);
7  end;
8
```

```
Function created.
```

```
1  declare
2  c float;
3  begin
4  c := raise();
5  dbms_output.put_line('Total salary =' || c);
6  end;
7
```

```
Statement processed.
Total salary =13800
```

Madhusudhana Naidu
D6AD 36

DBMS Lab
Experiment 10

Aim:
Demonstrate DCL and TCL commands.

Objective:
To execute commands that control databases and transactions.

Theory:
Data Control Language (DCL): Commands that control a database, including administering privileges and committing data. Used to create roles, permissions, and referential integrity as well it is used to control access to databases by securing it.

GRANT: Give users access privileges to the database. Only Database Administrator's or Owner's of the database object can provide/remove privileges on a database object. It is used to assign permissions to users

Eg: grant insert on emp to user1; //only user1 can insert
grant all on emp to public; //assign all permissions to all users.

REVOKE: Withdraw user's access privileges to database given with the GRANT command.
Syntax: revoke <permission> on <object_name> from <username>;
Eg: revoke all on emp from user1; //get back all permissions from user1

revoke select on emp from public; //get back select permissions from all users.

Transaction control:

A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.

Example: A bank employee transfers Rs. 500 from A's account to B's account.

A's account

Open_Account(A)

Old_Balance = A.balance

New_Balance = Old.Balance - 500

A.balance = New.Balance

Close_Account(A)

B's Account

Open Account(B)

Old_Balance = B.balance

New_Balance = Old.Balance +500

B.balance = New_Balance

Close_Account(B)

GRANT:

```
SQL> grant select, update on emp to flash
  2  ;

Grant succeeded.
```

REVOKE:

```
SQL> revoke select,update on emp from flash;

Revoke succeeded.
```

TCL COMMANDS: COMMIT, ROLLBACK, SAVEPOINT---

```
SQL> select * from customer;

       CID CNAME                     PHNO
---------- -------------------- ----------
        12 ramesh                   99201
        12 ramesh                   98021
        13 suresh                   86547
        14 mahesh                  675320
```

COMMIT:

```
SQL> delete from customer
  2  where cid=12;

2 rows deleted.

SQL> commit;

Commit complete.

SQL> select * from customer;

       CID CNAME                     PHNO
---------- -------------------- ----------
        13 suresh                   86547
        14 mahesh                  675320
```

ROLLBACK:

```
SQL> delete from customer
  2  where cid=12;

2 rows deleted.

SQL> rollback;

Rollback complete.
```

SAVEPOINT:

```
SQL> insert into customer values (12,'ramesh',99201);

1 row created.

SQL> insert into customer values(12,'ramesh',98021);

1 row created.

SQL> savepoint customers;

Savepoint created.

SQL> select * from customer;

       CID CNAME                     PHNO
---------- -------------------- ----------
        12 ramesh                   99201
        13 suresh                   86547
        14 mahesh                  675320
        12 ramesh                   98021

SQL> delete from customer where cid=12;

2 rows deleted.

SQL> select * from customer;

       CID CNAME                     PHNO
---------- -------------------- ----------
        13 suresh                   86547
        14 mahesh                  675320

SQL> rollback to customers;

Rollback complete.

SQL> select * from customer;

       CID CNAME                     PHNO
---------- -------------------- ----------
        12 ramesh                   99201
        13 suresh                   86547
        14 mahesh                  675320
        12 ramesh                   98021
```

Madhusudhana Naidu
D6AD 36

DBMS Lab
Experiment 11

Experiment - 11

Aim:
Implementation of transaction and concurrency control in PL-SQL.

Theory:

- Transaction control:

A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.

Example: A bank employee transfers Rs. 500 from A's account to B's account.

A's account
Open_Account(A)
Old_Balance = A.balance
New_Balance = Old.Balance - 500
A.balance = New.Balance
Close_Account(A)

B's Account
Open Account(B)
Old_Balance = B.balance
New_Balance = Old.Balance +500
B.balance = New_Balance
Close_Account(B)

Concurrency Control

- Concurrency control manages the transactions simultaneously without letting them interfere with each another.
- The main objective of concurrency control is to allow many users perform different operations at the same time.
- Using more than one transaction concurrently improves the performance of system.
- If we are not able to perform the operations concurrently, then there can be serious problems such as loss of data integrity and consistency.
- It reduces waiting time of transaction.

```
SQL> select * from emp;

NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------


subrato                  55000          1          3          1
manas                    51750          2          2          2
jayesh                   65000          3          1          3
sarthak                46287.5          4          4          4
arnav                    50000          5          5          5
```

- CHANGING SALARY OF SSN=1
- COMMIT,SAVING IT
- CHANGING SALARY OF SSN=2
- ROLLING BACK THE CHANGE

```
SQL> set serveroutput on;
SQL>  DECLARE
  2   salary emp.salary%type;
  3  BEGIN
  4  dbms_output.put_line('Updating salary of employee,ssn=1');
  5  update emp set salary=salary+10000 where ssn=1;
  6   dbms_output.put_line('Committing transaction');
  7  COMMIT;
  8   dbms_output.put_line('Adding a savepoint');
  9  SAVEPOINT sal1;
 10  dbms_output.put_line('Updating salary of employee,ssn=2');
 11  update emp set salary=salary+10000 where ssn=2;
 12  dbms_output.put_line('Rolling back to savepoint');
 13  ROLLBACK TO sal1;
 14  END;
 15  /
Updating salary of employee,ssn=1
Committing transaction
Adding a savepoint
Updating salary of employee,ssn=2
Rolling back to savepoint

PL/SQL procedure successfully completed.
```

- WE CAN SEE THAT ONLY SALARY OF EMP SSN=1 GETS UPDATED

```
SQL> select * from emp;

NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------


subrato                  65000          1          3          1
manas                    51750          2          2          2
jayesh                   65000          3          1          3
sarthak                46287.5          4          4          4
arnav                    50000          5          5          5
```

```
SQL>  set serveroutput on;
SQL> DECLARE
  2   rollno student.sno%type;
  3  s_age student.age%type;
  4  snm student.sname%type;
  5  s_cr student.course%type;
  6  BEGIN
  7  rollno:=&sno;
  8  s_age:=&age;
  9  snm:='&name';
 10  s_cr:='&course':
 11  insert into student values (rollno,s_age,snm,s_cr);
 12  dbms_output.put_line('one record inserted');
 13  COMMIT;
 14  SAVEPOINT save1;
 15   rollno:=&sno;
 16   s_age:=&age;
 17  snm:='&name';
 18  s_cr:='&course':
 19  INSERT into student values (rollno,s_age,snm,s_cr);
 20   dbms_output.put_line('one record inserted');
 21  ROLLBACK to SAVEPOINT save1;
 22  END;
 23  /
```

```
Enter value for sno: 60
old   7: rollno:=&sno;
new   7: rollno:=60;
Enter value for age: 19
old   8: s_age:=&age;
new   8: s_age:=19;
Enter value for name: subrato
old   9: snm:='&name';
new   9: snm:='subrato';
Enter value for course: ai
old  10: s_cr:='&course':
new  10: s_cr:='ai':
Enter value for sno: 61
old  15:  rollno:=&sno;
new  15:  rollno:=61;
Enter value for age: 19
old  16:  s_age:=&age;
new  16:  s_age:=19;
Enter value for name: vivek
old  17: snm:='&name';
new  17: snm:='vivek';
Enter value for course: it
old  18: s_cr:='&course':
new  18: s_cr:='it':
```

```
SQL> desc employee;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPLOYEE_ID                               NOT NULL NUMBER(38)
 ENAME                                     NOT NULL VARCHAR2(20)
 SALARY                                             NUMBER
 FK_EMPLOYEE                                        NUMBER(38)
 D_NO                                               NUMBER(38)

SQL> lock table employee in share mode;

Table(s) Locked.

SQL> commit;

Commit complete.
```

```
SQL> lock table employee in exclusive mode;

Table(s) Locked.

SQL> rollback;

Rollback complete.
```

```
SQL> lock table system.Donor in exclusive mode;

Table(s) Locked.

SQL> lock table system.Receiver in exclusive mode;
lock table system.Receiver in exclusive mode
*
ERROR at line 1:
ORA-00060: deadlock detected while waiting for resource


SQL> rollback;

Rollback complete.
```

```
SQL> insert into emp values('mahi',30000,6,6,6);

1 row created.
```

```
SQL> savepoint s1;

Savepoint created.

SQL> update emp
  2  set salary=salary+1000;

13 rows updated.

SQL> select * from emp;

NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------


subrato                  56000          1          3          1
manas                    52750          2          2          2
jayesh                   66000          3          1          3
sarthak                47287.5          4          4          4
arnav                    51000          5          5          5




NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------
Ramesh                   11000          1          2          1
mahi                     31000          6          6          6

13 rows selected.

SQL> rollback to s1;

Rollback complete.
```

```
SQL> select * from emp;

NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------


subrato                  55000          1          3          1
manas                    51750          2          2          2
jayesh                   65000          3          1          3
sarthak                46287.5          4          4          4
arnav                    50000          5          5          5




NAME                    SALARY        SSN        DNO   SUPERSSN
-------------------- ---------- ---------- ---------- ----------
Ramesh                   10000          1          2          1
mahi                     30000          6          6          6
```

Madhusudhana Naidu
D6AD 36

DBMS Lab
Experiment 12

Aim:
Implement Database connectivity (JDBC, ODBC)

Theory:
JDBC architecture -
In the 2-tier model, a Java app talks directly to data source. This requires a JDBC driver that can communicate with the particular data source being accessed. The data source may be located on another machine to which the user is connected via network. This is referred to as a client/server config, with user's machine as client and machine housing the data source as server. The network can be an intranet.

In the 3-tier model, commands are sent to a 'middle tier' of services, which then sends the commands to the data source. Data source processes the commands and sends the result back to middle tier, which then sends them to the user.

Fundamental steps in JDBC:
1) Import JDBC packages
2) Load and register the JDBC driver
3) Open a connection to the database
4) Create a statement object to perform the query
5) Execute the statement object and return a query result set.
6) Process the result set.
7) Close the result set and statement objects.
8) Close the connection.

```java
import java.sql.*;
public class jdbc {

    static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/";
    static final String USER = "java";
    static final String PASS = "subo";
public static void main(String[] args) {
Connection conn = null;
Statement stmt = null;
try{

Class.forName("com.mysql.cj.jdbc.Driver");

System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);

System.out.println("Creating database...");
stmt = conn.createStatement();

String sql1 = ("CREATE DATABASE STUDENTS1");
stmt.executeUpdate(sql1);
System.out.println("Database created successfully...");
}

catch(SQLException se){
se.printStackTrace();
}

catch(Exception e){
e.printStackTrace();
}

finally{
    try{
        if(stmt!=null)
        stmt.close();
        }

    catch(SQLException se2){
}
try{
    if(conn!=null)
    conn.close();
    }
    catch(SQLException se){
    se.printStackTrace();}
}

System.out.println("Goodbye!");
}
}
```

```
Connecting to database...
Creating database...
java.sql.SQLSyntaxErrorException: Access denied for user 'java'@'localhost' to database 'students1'
        at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:120)
        at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
        at com.mysql.cj.jdbc.StatementImpl.executeUpdateInternal(StatementImpl.java:1340)
        at com.mysql.cj.jdbc.StatementImpl.executeLargeUpdate(StatementImpl.java:2089)
        at com.mysql.cj.jdbc.StatementImpl.executeUpdate(StatementImpl.java:1251)
        at jdbc.main(jdbc.java:22)
Goodbye!
```

```
Connecting to database...
Creating database...
Goodbye!
```