Panipat Institute of Engineering & Technology,
Samalkha (Haryana)
Computer Science & Engineering Department

Database Management Systems Lab

PC-CS-309-LA

Submitted to:
Mr. Sandeep Singh Bindra
(Assistant Professor)
CSE Department

Submitted by:
Anmol Baranwal
2820208
B.Tech CSE 5th Sem A3

Affiliated to

Kurukshetra University Kurukshetra, India

INDEX

S no.	Name of the Program	Date	Signature
1	Write the queries for Data Definition Language (DDL) in RDBMS.		
2	Write the queries for Data Manipulation Language (DML) in RDBMS.		
3	Write SQL queries using logical operators.		
4	Create a database and perform the following operations:- i)Arithmetic and Relational Operations ii)Group By and Having clause iii)Like predicate for pattern matching in database		
5	Write SQL queries using Character, Number, Date.		
6	Write SQL queries for relational algebra.		
7	Write SQL queries for extracting data from more than one table.		
8.	Write SQL statements for nested sub queries.		
9.	Perform the concept of views in the table.		
10.	To perform various integrity constraints on relational database.		

Practical No. 1

Problem Statement:

Write the queries for Data Definition Language (DDL) in RDBMS

COMMANDS

- Create:

Syntax -
Create table <table_name>(column_name datatype(length), column_name datatype(length), …)

create table Employee1109(E_ID number(4) primary key, NAME char(20), AGE number(3), SALARY number(8), ADDRESS varchar(40))

Output -

```
Table created.

0.11 seconds
```

- Alter:

Syntax -
alter table <table_name> MODIFY column_name datatype(length)

ALTER TABLE Employee1109 MODIFY ADDRESS VARCHAR(50)

Output -

```
Table altered.

0.04 seconds
```

Syntax -
alter table <table_name> RENAME COLUMN prev_column_name TO new_column_name

ALTER TABLE Employee1109 RENAME COLUMN E_ID TO U_ID

Output -

```
Table altered.

0.01 seconds
```

Syntax -
alter table <table_name> add(column_name datatype(length))

Alter table Employee1109 add(Department char(20))

Output -

```
Table altered.

0.02 seconds
```

- Rename:

Syntax –
Rename <old_table_name> to <new_table_name>

Rename Employee1109 to Employee009

Output -

```
Statement processed.

0.07 seconds
```

- Drop:

Syntax –
Drop table <table_name>

drop table Employee1109

Output -

```
Table dropped.

0.06 seconds
```

- Truncate:

Syntax –
Truncate table <table_name>

Truncate table Employee1109

Output -

```
Table truncated.

0.02 seconds
```

Practical No. 2

Problem Statement:

Write the queries for Data Manipulation Language (DML) in RDBMS.

COMMANDS

- Insert:

Syntax –
Insert into <table_name> values(<expression1>, <expression2>)

insert into Employee1109 values(208, 'Anmol', 20, 7000,'Bhadohi - 221401, Uttar Pradesh')

Output -

```
1 row(s) inserted.

0.01 seconds
```

- Select:

Syntax –
Select * from <table_name>

select * from Employee1109

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	18500	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	7500	Vatican City, 239 Penn Street

7 rows returned in 0.00 seconds Download

- Update:

Syntax –
Update <table_name> set field=value where condition

Update Employee1109 set Salary='17000' where E_ID='230'

Output -

```
1 row(s) updated.

0.01 seconds
```

After updating :

select * from Employee1109 where E_ID='230'
Output -

E_ID	NAME	AGE	SALARY	ADDRESS
230	John	40	15000	Street 12, Sector 5, California

1 rows returned in 0.00 seconds Download

- Delete:

Syntax –
Delete from <table_name> where field=condition

Delete from Employee1109 where name='Peter'

Output -

```
1 row(s) deleted.

0.00 seconds
```

After deleting :

select * from Employee1109
Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
230	John	40	15000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	7500	Vatican City, 239 Penn Street

6 rows returned in 0.00 seconds Download

Practical No. 3

Problem Statement:

Write SQL queries using logical operators.

Table Creation

select * from Employee1109

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	18500	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	7500	Vatican City, 239 Penn Street

7 rows returned in 0.00 seconds Download

COMMANDS

- AND

select Name from Employee1109 where Name = 'Jonas' AND Salary = 3000

Output -

NAME
Jonas

1 rows returned in 0.00 seconds

- OR

select Name from Employee1109 where Name = 'Jonas' OR Salary = 3000

Output -

NAME
Anmol
Jonas
Jonas

3 rows returned in 0.01 seconds

- NOT

select Name from Employee1109 where NOT Age=22

Output -

NAME
Anmol
Peter
John
Martha
Jonas

5 rows returned in 0.00 seconds

- BETWEEN

select * from Employee1109 where AGE between 20 AND 25

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	7500	Vatican City, 239 Penn Street

4 rows returned in 0.00 seconds Download

- IN

select * from Employee1109 where AGE in ('20','22')

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street

3 rows returned in 0.00 seconds Download

Practical No. 4

Problem Statement:

Create a database and write the programs to carry out the following operations-

1. Arithmetic and Relational Operations
2. Group By and having clause
3. Like predicate for pattern matching

Arithmetic Operations

- Add Operation (+):

Update Employee1109 set Salary=salary+'2000' where E_ID<'230'

```
3 row(s) updated.

0.29 seconds
```

Select * from Employee1109
Output –

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	9000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	11500	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	8000	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	6000	Vatican City, 241 Main Street
272	Jonas	25	9000	Vatican City, 239 Penn Street

7 rows returned in 0.01 seconds Download

- Subtract Operation (-):

Update Employee1109 set Salary=salary-'3000' where E_ID>'250'

```
2 row(s) updated.

0.00 seconds
```

Select * from Employee1109
Output –

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	9000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	11500	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	8000	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

7 rows returned in 0.00 seconds Download

- Multiply Operation (*):

Update Employee1109 set Salary=salary*'1.5' where E_ID='220'

```
1 row(s) updated.

0.00 seconds
```

Select * from Employee1109
Output –

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	9000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	17250	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	8000	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

7 rows returned in 0.00 seconds Download

- Divide Operation (/):

Update Employee1109 set Salary=salary/'1.5' where E_ID='220'

```
1 row(s) updated.

0.00 seconds
```

Select * from Employee1109
Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

7 rows returned in 0.00 seconds Download

Relational Operations

- Equal (=):

select * from Employee1109 where Age=20

Output –

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh

1 rows returned in 0.00 seconds Download

- Less than operator (<):

select * from Employee1109 where Name<'Martha'

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
230	John	40	17000	Street 12, Sector 5, California
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

5 rows returned in 0.00 seconds Download

- Greater than operator (>):

select * from Employee1109 where Name>'Martha'

Output –

E_ID	NAME	AGE	SALARY	ADDRESS
220	Peter	30	3833	512 Baker Street, San Diego

1 rows returned in 0.00 seconds Download

- Less than or equal to (<=):

select * from Employee1109 where salary<=6000

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	3833	512 Baker Street, San Diego
210	Martha	28	2667	Street 231 N Canada
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

5 rows returned in 0.00 seconds Download

- Greater than or equal to (>=):

select * from Employee1109 where age>=25
Output -

E_ID	NAME	AGE	SALARY	ADDRESS
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
272	Jonas	25	6000	Vatican City, 239 Penn Street

4 rows returned in 0.00 seconds Download

- Not equal to (<>):

select * from Employee1109 where age<>22

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
272	Jonas	25	6000	Vatican City, 239 Penn Street

5 rows returned in 0.01 seconds Download

Group By

Select count(Name),Age from Employee1109 Group by (Age)

Output -

COUNT(NAME)	AGE
1	30
2	22
1	25
1	28
1	20
1	40

6 rows returned in 0.00 seconds

Having Clause

Select count(Name),Age from Employee1109 Group by (Age) having count(E_ID)>'1'

Output -

COUNT(NAME)	AGE
2	22

1 rows returned in 0.00 seconds

Like Predicate for pattern matching

1. select * from Employee1109 where name like 'A__%'

Output -

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh

1 rows returned in 0.00 seconds Download

2. select * from Employee1109 where address like '%__re%'

Output –

E_ID	NAME	AGE	SALARY	ADDRESS
220	Peter	30	3833	512 Baker Street, San Diego
230	John	40	17000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

6 rows returned in 0.00 seconds Download

3. select * from Employee1109 where address like '%__San_D%'

Output –

E_ID	NAME	AGE	SALARY	ADDRESS
220	Peter	30	3833	512 Baker Street, San Diego

1 rows returned in 0.00 seconds Download

4. select * from Employee1109 where address like '%__eet'

Output –

E_ID	NAME	AGE	SALARY	ADDRESS
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	6000	Vatican City, 239 Penn Street

2 rows returned in 0.00 seconds Download

Practical No. 5

Problem Statement:

Write SQL queries using Character, Number, Date.

Table Creation:

- select * from Employee1109

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	DATEVAL
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh	05/07/2016
230	John	40	15000	Street 12, Sector 5, California	02/15/2021
210	Martha	28	2667	Street 231 N Canada	03/21/2017
250	Dwayne	22	9000	Street 47 W New York	12/11/2020
270	Jonas	22	3000	Vatican City, 241 Main Street	08/19/2020
272	Jonas	25	7500	Vatican City, 239 Penn Street	11/02/2018

6 rows returned in 0.00 seconds Download

Commands:

- AVERAGE(AVG)

Select Avg(Salary) as Avg_Salary from Employee1109

Output –

AVG_SALARY
6694.5

1 rows returned in 0.01 seconds

- COUNT

select Count(Name) as Count_Name from Employee1109

Output –

COUNT_NAME
6

1 rows returned in 0.00 seconds

- MIN

select Min(Salary) as Min_Salary from Employee1109

Output –

MIN_SALARY
2667

1 rows returned in 0.01 seconds

- MAX

select Max(DateVal) as Date_MAX from Employee1109

Output –

DATE_MAX
02/15/2021

1 rows returned in 0.01 seconds

Practical No. 6

Problem Statement:

Write SQL queries for relational algebra.

Table Creation:

- select * from Student119

Output –

S_ID	NAME	FEES
S1	Anmol Baranwal	200
S2	Peter	300
S3	James	400
S4	John	500

4 rows returned in 0.00 seconds

- select * from Faculty119

Output –

F_ID	NAME	SALARY
F1	Anmol Baranwal	3000
F2	Peter	4000
F3	Smith	5000
F4	Johnson	4000

4 rows returned in 0.00 seconds

Commands

- UNION:

Syntax –
SELECT column_name(s) FROM table1 UNION
SELECT column_name(s) FROM table2

select Name from Student119 UNION select name from Faculty119

Output –

NAME
Anmol Baranwal
James
John
Johnson
Peter
Smith

6 rows returned in 0.00 seconds

- INTERSECT:

Syntax –
SELECT column_name(s) FROM table1 INTERSECT
SELECT column_name(s) FROM table2

select Name from Student119 INTERSECT select name from Faculty119

Output –

NAME
Anmol Baranwal
Peter

2 rows returned in 0.01 seconds

- MINUS:

Syntax –
SELECT column_name(s) FROM table1 MINUS
SELECT column_name(s) FROM table2

select Name from Student119 MINUS select name from Faculty119

Output –

NAME
James
John

2 rows returned in 0.01 seconds

Practical No. 7

Problem Statement:

Write SQL queries for extracting data from more than one table.

Table Creation:

- select * from Student119

Output –

S_ID	NAME	FEES
S1	Anmol Baranwal	200
S2	Peter	300
S3	James	400
S4	John	500

4 rows returned in 0.00 seconds

- select * from Faculty119

Output –

F_ID	NAME	SALARY
F1	Anmol Baranwal	3000
F2	Peter	4000
F3	Smith	5000
F4	Johnson	4000

4 rows returned in 0.00 seconds

JOINS

RIGHT JOIN:

- select Student119.S_ID, Student119.Name, Student119.Fees, Faculty119.F_ID, Faculty119.Name, Faculty119.Salary from Student119 RIGHT JOIN Faculty119 on Student119.Name = Faculty119.Name

- select * from Student119 RIGHT JOIN Faculty119 on Student119.Name = Faculty119.Name

Output –

S_ID	NAME	FEES	F_ID	NAME	SALARY
S1	Anmol Baranwal	200	F1	Anmol Baranwal	3000
S2	Peter	300	F2	Peter	4000
-	-	-	F3	Smith	5000
-	-	-	F4	Johnson	4000

4 rows returned in 0.01 seconds Download

LEFT JOIN:

select * from Student119 LEFT JOIN Faculty119 on Student119.Name = Faculty119.Name

Output –

S_ID	NAME	FEES	F_ID	NAME	SALARY
S1	Anmol Baranwal	200	F1	Anmol Baranwal	3000
S2	Peter	300	F2	Peter	4000
S3	James	400	-	-	-
S4	John	500	-	-	-

4 rows returned in 0.00 seconds Download

FULL JOIN:

select * from Student119 FULL JOIN Faculty119 on Student119.Name = Faculty119.Name

Output –

S_ID	NAME	FEES	F_ID	NAME	SALARY
S1	Anmol Baranwal	200	F1	Anmol Baranwal	3000
S2	Peter	300	F2	Peter	4000
-	-	-	F3	Smith	5000
-	-	-	F4	Johnson	4000
S3	James	400	-	-	-
S4	John	500	-	-	-

6 rows returned in 0.00 seconds Download

INNER JOIN:

select * from Student119 INNER JOIN Faculty119 on Student119.Name = Faculty119.Name

Output –

S_ID	NAME	FEES	F_ID	NAME	SALARY
S1	Anmol Baranwal	200	F1	Anmol Baranwal	3000
S2	Peter	300	F2	Peter	4000

2 rows returned in 0.00 seconds Download

Practical No. 8

Problem Statement:

Write SQL statements for nested sub queries.

Table Display:

- select * from Student119

Output –

S_ID	NAME	FEES
S1	Anmol Baranwal	200
S2	Peter	300
S3	James	400
S4	Peter	500

4 rows returned in 0.00 seconds

Commands:

- select Name from Student119 where FEES > (select MIN(FEES) from Student119)

Output –

NAME
Peter
James
Peter

3 rows returned in 0.00 seconds

- select Name from Student119 where FEES < (select AVG(FEES) from Student119)

Output –

NAME
Anmol Baranwal
Peter

2 rows returned in 0.00 seconds

Practical No. 9

Problem Statement:

Perform the concept of views in the table.

Table Creation:

select * from Employee1109

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	D_ID
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh	10
220	Peter	30	3833	512 Baker Street, San Diego	30
230	John	40	17000	Street 12, Sector 5, California	40
210	Martha	28	2667	Street 231 N Canada	30
250	Dwayne	22	9000	Street 47 W New York	20
270	Jonas	22	3000	Vatican City, 241 Main Street	10
272	Jonas	25	6000	Vatican City, 239 Penn Street	50

7 rows returned in 0.00 seconds Download

create View V09 as
select * from Employee1109 where age>'22' or E_Id='10'

Output –

```
View created.
```

0.01 seconds

select * from V09

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	D_ID
220	Peter	30	3833	512 Baker Street, San Diego	30
230	John	40	17000	Street 12, Sector 5, California	40
210	Martha	28	2667	Street 231 N Canada	30
272	Jonas	25	6000	Vatican City, 239 Penn Street	50

4 rows returned in 0.00 seconds Download

Changing Parent Table:

Update Employee1109 set salary='18500' where E_ID='230'

Output –

```
1 row(s) updated.

0.00 seconds
```

select * from Employee1109

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	D_ID
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh	10
220	Peter	30	3833	512 Baker Street, San Diego	30
230	John	40	18500	Street 12, Sector 5, California	40
210	Martha	28	2667	Street 231 N Canada	30
250	Dwayne	22	9000	Street 47 W New York	20
270	Jonas	22	3000	Vatican City, 241 Main Street	10
272	Jonas	25	6000	Vatican City, 239 Penn Street	50

7 rows returned in 0.01 seconds Download

select * from V09

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	D_ID
220	Peter	30	3833	512 Baker Street, San Diego	30
230	John	40	18500	Street 12, Sector 5, California	40
210	Martha	28	2667	Street 231 N Canada	30
272	Jonas	25	6000	Vatican City, 239 Penn Street	50

4 rows returned in 0.00 seconds Download

Practical No. 10

Problem Statement: To perform various integrity constraints on relational database.

create table Employee1109(E_ID number(4) primary key, NAME char(20), AGE number(3), SALARY number(8), ADDRESS varchar(40))

select * from Employee1109

E_ID	NAME	AGE	SALARY	ADDRESS
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh
230	John	40	15000	Street 12, Sector 5, California
210	Martha	28	2667	Street 231 N Canada
250	Dwayne	22	9000	Street 47 W New York
270	Jonas	22	3000	Vatican City, 241 Main Street
272	Jonas	25	7500	Vatican City, 239 Penn Street

6 rows returned in 0.02 seconds Download

- **Domain Constraints**

Domain constraints can be defined as the definition of a valid set of values for an attribute. The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

Example –

insert into Employee1109 values('E272', 'Louise', 28, 13000, 'Montana, 1645 LEWIS AVE BILLINGS')

Output -

```
ORA-01722: invalid number
```

The error occurs because we try to insert a value which is not in correct format.

- **Entity Integrity Constraints**

The entity integrity constraint states that primary key value can't be null. This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows. A table can contain a null value other than the primary key field.

Example –
insert into Employee1109 values(‘’, 'Robin', ‘24’, ‘8000’,’Barcelona Spain’)

Output –

```
ORA-00911: invalid character
```

The error occurs because we try to insert null value in primary key attribute which is not possible.

- **Referencial Integrity Constraints**

A referential integrity constraint is specified between two tables. In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.

Table 1-

select * from Department1109

Output –

D_NO	D_NAME
10	SDE
20	Frontend
30	UI
40	UX
50	Backend

5 rows returned in 0.00 seconds Download

Table 2-

alter table Employee1109 add(D_Id number(3) references Department1109 (D_No))

select * from Employee1109

Output –

E_ID	NAME	AGE	SALARY	ADDRESS	D_ID
208	Anmol	20	3000	Bhadohi - 221401, Uttar Pradesh	30
230	John	40	15000	Street 12, Sector 5, California	10
210	Martha	28	2667	Street 231 N Canada	50
250	Dwayne	22	9000	Street 47 W New York	40
270	Jonas	22	3000	Vatican City, 241 Main Street	10
272	Jonas	25	7500	Vatican City, 239 Penn Street	20

6 rows returned in 0.00 seconds Download

Update Employee1109 set D_Id='70' where E_ID='270'

Output –

```
ORA-02291: integrity constraint (ANMOL.SYS_C007089) violated - parent key not found
```

The error occurs because we try to insert D_ID as 70 in table 2 but the value is not present in table 1 and violate Foreign Key Constraint.

- **Key constraints**

Keys are the entity set that is used to identify an entity within its entity set uniquely. An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique but not null values in the relational table.

Example -
insert into Employee1109 values(250, 'Ian', 24, 8500, 'Arizona, Coplin Avenue')

Output -

```
ORA-00001: unique constraint (ANMOL119.SYS_C007273) violated
```

The error occurs because we try to insert duplicate values in our table which violate the Primary key rules.

Practical No. 11

Problem Statement:

Write the queries for Data Control Language (DCL) in RDBMS.

The list of object privileges is as follows:

1. **ALTER:** allows the grantee to change the table definitions with the ALTER table command.
2. **DELETE:** allows the grantee to remove the records from the table with the DELETE command.
3. **INDEX:** allows the grantee to create an index on the table with the CREATE INDEX command.
4. **INSERT:** allows the grantee to add records to the table with the INSERT command.
5. **SELECT:** allows the grantee to query the table with SELECT command.
6. **UPDATE:** allows the grantee to modify the records in the table with the UPDATE Command.

Commands

- GRANT:

Syntax -

```
GRANT privilege_name
ON object_name
TO {user_name | PUBLIC | role_name}
```

- REVOKE:

Syntax -

```
REVOKE privilege_name
ON object_name
FROM {user_name | PUBLIC | role_name}
```

Practical No. 12

Problem Statement:

To perform after triggers.

Table Creation:

- select * from Student119

Output –

S_ID	NAME	FEES
S1	Anmol Baranwal	200
S2	Peter	300
S3	James	400
S4	Peter	500

4 rows returned in 0.00 seconds

- create table SecurityMX(S_ID varchar(3), Name char(20), Fees number(4), Operator char(15), Date_and_Time timestamp)

Output –

```
Table created.

0.00 seconds
```

Commands:

- Trigger :

create or replace trigger trig1 after insert or update on Student119 for each row declare
Operator char(15); Date_and_Time timestamp;begin
if updating then Operator:='update'; Date_and_Time:=SYSTIMESTAMP; end if;
if inserting then Operator:= 'insert'; Date_and_Time:=SYSTIMESTAMP; end if;
insert into SecurityMX values(:old.S_ID, :old.Name, :old.Fees, operator, Date_and_Time); end

Output –

```
Trigger created.

0.02 seconds
```

Update table Student119

- Update :

update Student119 set Fees='600' where S_ID='S4'

Output –

```
1 row(s) updated.

0.00 seconds
```

select * from SecurityMX

Output –

S_ID	NAME	FEES	OPERATOR	DATE_AND_TIME
S4	Peter	500	update	29-NOV-22 10.37.52.819000 AM
S4	Peter	600	update	29-NOV-22 10.40.29.007000 AM

2 rows returned in 0.00 seconds Download

- insert into Student119 values('S5','Jonas','450')

Output –

```
1 row(s) inserted.

0.00 seconds
```

select * from SecurityMX

Output –

S_ID	NAME	FEES	OPERATOR	DATE_AND_TIME
S4	Peter	500	update	29-NOV-22 10.37.52.819000 AM
S4	Peter	600	update	29-NOV-22 10.40.29.007000 AM
-	-	-	insert	29-NOV-22 10.48.17.561000 AM

3 rows returned in 0.00 seconds Download

Practical No. 13

Problem Statement:

To perform before triggers.

Table Creation:

- select * from Student119

Output –

S_ID	NAME	FEES
S1	Anmol Baranwal	200
S2	Peter	300
S3	James	400
S4	Peter	500

4 rows returned in 0.00 seconds

Commands:

- Trigger :

```
create or replace trigger trig2
  before update of fees on Student119
  for each row
begin
  if :new.fees <=0 then
     raise_application_error(-20001, 'The salary cannot be less than or equal to zero');
  end if;
end;
```

Output –

```
Trigger created.
```

0.00 seconds

- Update :

```
update Student119 set Fees='-20' where S_ID='S4'
```

Output –

```
ORA-20000: The salary cannot be less than zero
ORA-06512: at "ANMOL.TRIG2", line 3
ORA-04088: error during execution of trigger 'ANMOL.TRIG2'


1. update Student119 set Fees='-20' where S_ID='S4'
```

Practical No. 14

Problem Statement:

Write a procedure for computing income tax of employee on the basic of following conditions:-
if gross pay<=40,000 then I.T rate is 0%.
if gross pay>40,000 but <60000 then I.T rate is 10%.
if gross pay>60,000 but <1,00,0000 then I.T rate is 20%.
if gross pay>1,00,0000 then I.T rate is 30%.

For this purpose create a table with name, ssn, gross salary and income tax of the employee.

Procedure Creation:

```
create table server1109(SSN number(3), name varchar(10),salary number(7),tax number(10,3))
```

Output –

```
Table created.

0.03 seconds
```

```
create or replace procedure incometax1109(SSN in number, name in varchar, salary in varchar)
is tax number(10,3);
begin
if(salary<=40000) then
tax:=0.000;
end if;
if(salary>40000 and salary<=60000) then
tax:=(10*salary)/100;
end if;
if(salary>60000 and salary<=1000000) then
tax:=(20*salary)/100;
end if;
if(salary>1000000) then
tax:=(30*salary)/100;
end if;
insert into server1109 values(SSN,name,salary,tax);
end;
```

Output –

```
Procedure created.

0.02 seconds
```

To be executed on Command Line:

```
execute incometax1109('251' , ' Sabhyata' ,'300000');
execute incometax1109 ('6872' , 'Sam' ,'700000');
execute incometax1109 ('7844' , 'John' ,'1100000');
execute incometax1109 ('9852' , 'Peter' ,' 500000');
commit;

select * from server1109;
```

Output –

SSN	NAME	SALARY	TAX
251	Sabhyata	300000	0
6872	Sam	700000	140000
7844	John	1100000	330000
9852	Peter	500000	50000