KEEGAN VAZ
ROLL NO-42
SE-CMPN-B
PID-192120

# ST. FRANCIS INSTITUTE OFTECHNOLOGY

MT. POINSUR, BORIVALI (W), MUMBAI

# LAB MANUAL

## EXPERIMENT NO. 3

**Aim:** Apply Integrity Constraints for the specified system. Use DCL and TCL commands to provide security to data.

**Theory:**

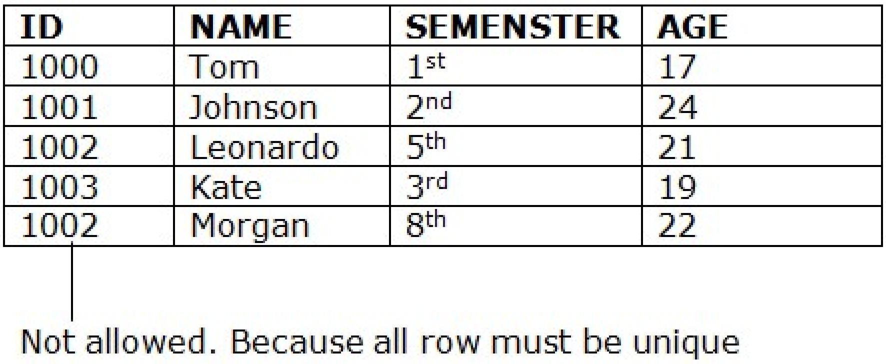
1. Explain different SQL integrity Constraint with example

**Ans:** Integrity constraints guard against accidental damage to the database, by ensuring that authorized changes to the database do not result in a loss of data consistency.

### 1. 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 and null value in the relational table.

**Example:**



| ID | NAME | SEMENSTER | AGE |
|---|---|---|---|
| 1000 | Tom | 1st | 17 |
| 1001 | Johnson | 2nd | 24 |
| 1002 | Leonardo | 5th | 21 |
| 1003 | Kate | 3rd | 19 |
| 1002 | Morgan | 8th | 22 |

Not allowed. Because all row must be unique

### 2. Domain Constraints:

a. They define valid values for attributes.
b. They are the most elementary form of integrity constraint.
c. They test values inserted in the database, and test queries to ensure that the comparisons make sense.
d. Domain constraints on a single relation:
   - Null- unknown value is acceptable in columns

Example: email_id varchar(10) null;

- not null- Declare name and budget to be not null
  Example: name varchar(20) not null
  budget numeric(12,2) not null
- primary key- Combination of unique and not null. i.e. each row should have known distinct value.
  Example: emp_id number(10) primary key;
- Unique- It does not allow the same value to be present on two different rows.
  Example: emp_id varchar(10) unique;
- Default- It specifies the default value for a column using default clause. The default value for column is inserted for a row, when a user does not enter a value for that column
  Example: country varchar(10) default 'India'
- check (P), where P is a predicate- If we want values in the columns based on some conditions
  Example: Ensure that semester is one of fall, winter, spring or summer:

create table section ( course_id varchar (8), sec_id varchar (8), semester varchar (6),
year numeric (4,0), building varchar (15), room_number varchar (7), time slot id varchar (4),
primary key (course_id, sec_id, semester, year),
check (semester in ('Fall', 'Winter', 'Spring', 'Summer')));

- Foreign Key Constraint: References primary key of another table

  Example: D_id references Employee

## 3. Referential Integrity:

a. Ensures that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another relation.

b. Let A be a set of attributes. Let R and S be two relations that contain attributes A and where A is the primary key of S. A is said to be a foreign key of R if for any values of A appearing in R these values also appear in S.

c. Ensures that a value that appears in one relation for a given set of attributes also appears for a certain set of attribute in another relation.
If an account exists in the database with branch name "SBI", then the branch "SBI" must actually exist in the database.

d. Consider relationship R between entity E1 and E2. R is represented as a relation including primary keys K1 of E1 and K2 of E2. Then K1 and K2 form foreign keys on the relational schemas for E1 and E2 respectively.

e. Weak entity sets are also a source of referential integrity constraints. For, the relation schema for a weak entity set must include the primary key of the entity set which it depends.
Dependent ( employee-no, dependent-name, age, sex )

f. Example:
create table customer (customer-name char(20) not null, customer-street char(30), customer-city char(30),

```
primary key (customer-name))

create table branch (branch-name char(15) not null, branch-city char(30), assets integer,
primary key (branch-name))

create table account (branch-name char(15), account-number char(10) not null, balance
integer,
 primary key(account-number),
foreign key (branch-name) references branch)

create table depositor (customer-name char(20) not null, account-number char(10) not
null,
primary key (customer-name, account-number),
foreign key (account-number) references account,
foreign key (customer-name) references customer)
```

g. Cascading actions in SQL:

```
create table account
.....
foreign key (branch-name) references branch
on delete cascade
on update cascade,
...)
```

- Due to the on delete cascade clauses, if a delete of a tuple in branch results in referential-integrity constraint violation, the delete "cascades" to the account relation, deleting the tuple that refers to the branch that was deleted.
- Cascading updates are similar.

h. Cascading actions in Referential integrity:

- create table course (

```
course_id char(5) primary key,
title varchar(20),
dept_name varchar(20) references department
)
```

- create table course (

```
...
dept_name varchar(20),
foreign key (dept_name) references department
on delete cascade
on update cascade,
. . .
)
```

2. Explain DCL and TCL with Syntaxes and examples.

Ans:

**DCL Commands(Data Control Language):** It includes **commands** such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of the database system.

**a) Grant**

1. The grant statement is used to confer authorization

grant <privilege list> on <relation name or view name> to <user list>

2. <user list> is:
- a user-id
- public, which allows all valid users the privilege granted
3. Granting a privilege on a view does not imply granting any privileges on the underlying relations.
4. The grantor of the privilege must already hold the privilege on the specified item (or be the database administrator).
5. DCL privileges in SQL:
   - select: allows read access to relation, or the ability to query using the view

   Example: grant users U1, U2, and U3 select authorization on the instructor relation:

   grant select on instructor to U1 , U2 , U3;
   - insert: the ability to insert tuples
   - update: the ability to update using the SQL update statement
   - delete: the ability to delete tuples.
   - all privileges: used as a short form for all the allowable privileges

Syntax: Grant <permissions> 'select, insert, delete, update on <object_name> to <username>;

Example: Grant insert on emp to user1; //only user1 can insert

Grant all on emp to public; //assign all permissions to all users.

**b) Revoke**

1. The revoke statement is used to revoke authorization.

revoke <privilege list>
on <relation name or view name> from <user list>

Example: revoke select on branch from U1 , U2 , U3;

2. <privilege-list> may be all to revoke all privileges.
3. If <revokee-list> includes public, all users lose the privilege except those granted it explicitly.
4. If the same privilege was granted twice to the same user by different grantees, the user may retain the privilege after the revocation.
5. All privileges that depend on the privilege being revoked are also revoked.

Syntax: Revoke <permission> on <object_name> from <username>;

Example: revoke all on emp from user1; //get back all permissions from user1.

Revoke select on emp from public; get back select permission from all users

**TCL Commands(Transaction Control Language):** Transaction Control Language commands are used to manage transactions in the database. These are used to manage the changes made by DML- statements. It also allows statements to be grouped together into logical transactions.

## 1. Commit:

Commit command is used to permanently save any transaction into the database.

Syntax: Commit;// end or start a transaction

Example:
DELETE FROM CUSTOMERS WHERE AGE = 25;
COMMIT;

## 2. Rollback:

This command restores the database to last committed state. Rollback command is used to undo transactions that have not already been saved to the database.

Syntax: Rollback; //undo up to commit
Example:
DELETE FROM CUSTOMERS WHERE AGE = 25;
ROLLBACK;

## 3. Savepoint:

Savepoint command is used to temporarily save a transaction so that you can rollback to that point whenever necessary. It is used to roll the transaction back to a certain point without rolling back the entire transaction.

Syntax: SAVEPOINT SAVEPOINT_NAME;

Example:

SAVEPOINT S1;

ROLLBACK TO SAVEPOINT S1;

## 4. Rollback to:

It is also used with savepoint command to jump to a savepoint in a transaction.
Example: rollback to P1;

Using ROLLBACK without the TO SAVEPOINT clause performs the following operations:

- Ends the transaction
- Undoes all changes in the current transaction
- Erases all savepoints in the transaction

Using ROLLBACK with the TO SAVEPOINT clause performs the following operations:

- Rolls back just the portion of the transaction after the savepoint.
- Erases all savepoints created after that savepoint. The named savepoint is retained, so you can roll back to the same savepoint multiple times.
- Prior savepoints are also retained.

**Implementation:**

1. Apply SQL Integrity constraint for given exercise
2. Use DCL and TCL syntax for given exercise

**Lab Manual:**

1. **Null Constraint:** It means an Unknown Value.

   **E.g. mobile number (10) null**

2. **Not Null Constraint:** It means always a Known Value.

   **E.g. Name varchar2 (20) not null**

3. **Unique Constraint:** It ensures that no two rows have the same value in the specified column(s). I.e. Known Value (Distinct) or Unknown Value.

   **E.g. ecode number (5) unique**

4. **Primary Key Constraint:** It is similar to unique constraint except that the Primary Key cannot allow Null values so that this constraint must be applied to columns declared as Not Null. I.e. Always Known Value (Distinct).

   **E.g. Empid char (5) primary key**

5. **Default Constraint:** A default value can be specified for a column using default clause when a user does not enter a value for that column.

   **E.g. Grade char (2) default 'E1'**

6. **Check Constraint:** It limits values that can be inserted into a column.

   **E.g. Sal number (10) check (Sal > 2000)**

7. **Foreign Key Constraint:** References primary key of another table

   **E.g.** D_id references Employee

**DCL Commands:**

**a) Grant**

Grant <permissions> 'select, insert, delete, update on <object_name> to <username>;
E.g. Grant insert on emp to user1; //only user1 can insert
Grant all on emp to public; //assign all permissions to all users.

**b) Revoke**

Revoke <permission> on <object_name> from <username>;
E.g. revoke all on emp from user1; //get back all permissions from user1.
Revoke select on emp from public; get back select permission from all users

**TCL Commands:**

### 1. <u>Commit</u>:

E.g. Commit;// end or start a transaction

### 2. Rollback:

E.g. Rollback; //undo up to commit

### 3. Savepoint:

E.g. savepoint P1;

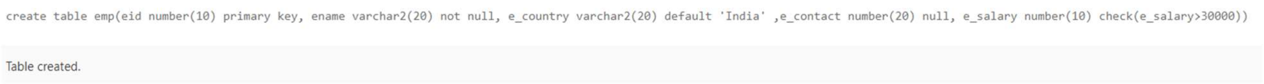
### 4. Rollback to:

E.g. rollback to P1;

**Exercise:**

**1. SQL Integrity Constraint:**

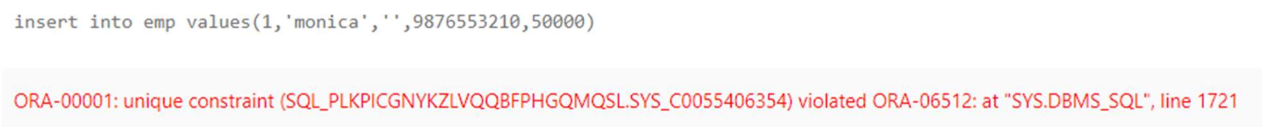
a. Create table Employee (Eid, Ename, E_country, E_contact,E_Salary)



```
create table emp(eid number(10) primary key, ename varchar2(20) not null, e_country varchar2(20) default 'India' ,e_contact number(20) null, e_salary number(10) check(e_salary>30000))
```

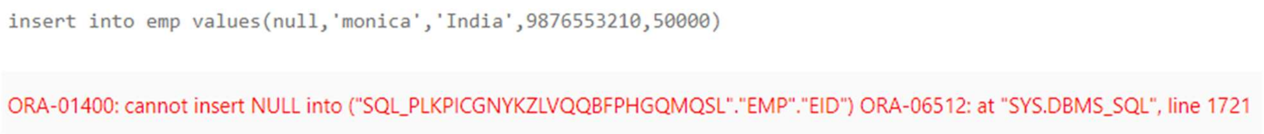
```
Table created.
```

b. EID: Eid has to be unique and no null accepted.



```
insert into emp values(1,'monica','',9876553210,50000)
```

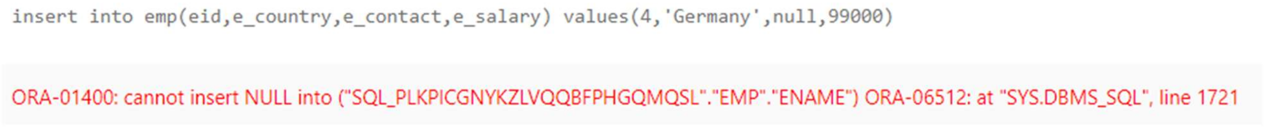
```
ORA-00001: unique constraint (SQL_PLKPICGNYKZLVQQBFPHGQMQSL.SYS_C0055406354) violated ORA-06512: at "SYS.DBMS_SQL", line 1721
```



```
insert into emp values(null,'monica','India',9876553210,50000)
```

```
ORA-01400: cannot insert NULL into ("SQL_PLKPICGNYKZLVQQBFPHGQMQSL"."EMP"."EID") ORA-06512: at "SYS.DBMS_SQL", line 1721
```

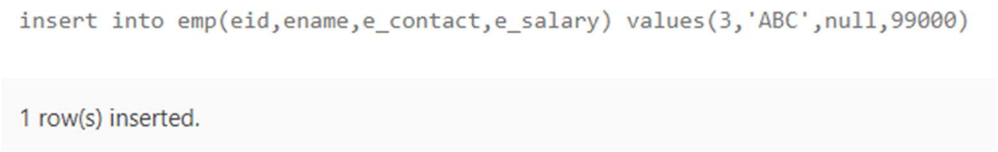
c. Ename: Ename has to be provided by the user.



```
insert into emp(eid,e_country,e_contact,e_salary) values(4,'Germany',null,99000)
```

```
ORA-01400: cannot insert NULL into ("SQL_PLKPICGNYKZLVQQBFPHGQMQSL"."EMP"."ENAME") ORA-06512: at "SYS.DBMS_SQL", line 1721
```

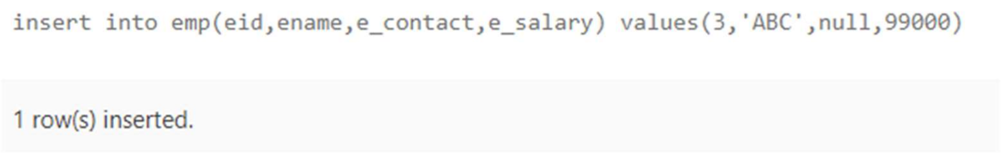
e. E_country: If user doesn’t provide E_country value then country should be India



```
insert into emp(eid,ename,e_contact,e_salary) values(3,'ABC',null,99000)
```

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

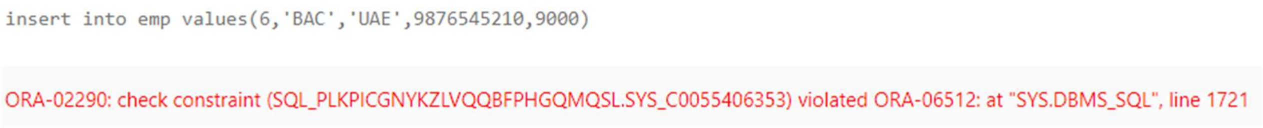
e. E_contact: If user doesn’t’t have contact details then null can be accepted .



```
insert into emp(eid,ename,e_contact,e_salary) values(3,'ABC',null,99000)
```

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

f. E_Salary: Make Salary as check (salary inserted should not be less than 30000)



```
insert into emp values(6,'BAC','UAE',9876545210,9000)
```

```
ORA-02290: check constraint (SQL_PLKPICGNYKZLVQQBFPHGQMQSL.SYS_C0055406353) violated ORA-06512: at "SYS.DBMS_SQL", line 1721
```

**2. DCL and TCL Exercise:**

a. create one more user (your user account already there and above employee table is also in your user account )

**SQL> alter session set"_ORACLE_SCRIPT" =true;**

**Session altered.**

**SQL> create user keegan1 identified by 1234;**

**User created.**

b. Grant inserts privilege from your user login to other user which you have just created.

**SQL> grant connect, resource to keegan1;**

**Grant succeeded.**

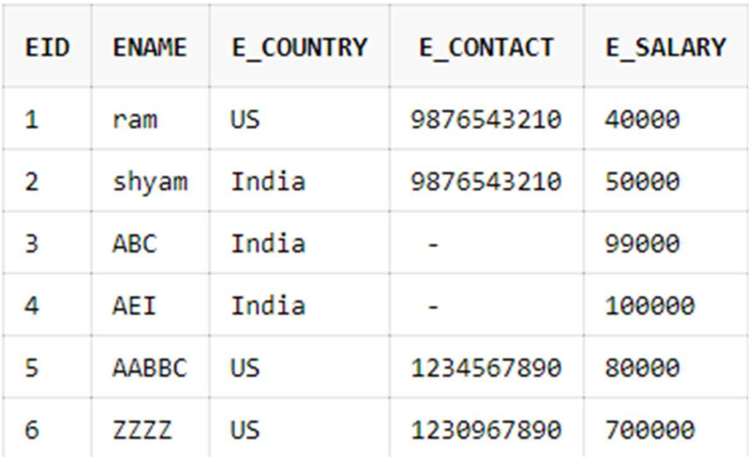
**SQL> grant insert on emp to keegan1;**

**Grant succeeded.**

c. Add 2 records using DML Syntax insert through other user login.

```
SQL> connect;
Enter user-name: keegan1
Enter password:
Connected.
SQL> insert into system.emp values (5,'AABBC','US',1234567890,80000);
1 row created.
SQL> insert into system.emp values (6,'ZZZZ','US',1230967890,700000);
1 row created.
```

```
SQL> connect;
Enter user-name: system
Enter password:
Connected.
SQL> select * from emp;
```



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |

```
SQL> commit;
Commit complete.
```

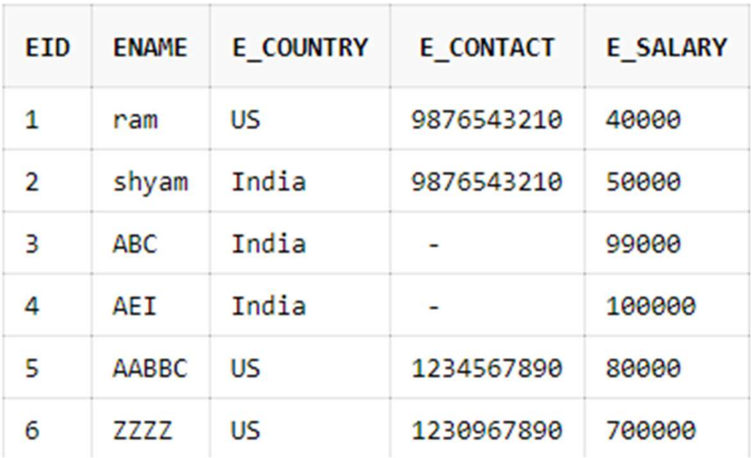
d. Then revoke the insert privilege from your account and check again trying to insert record from other user account.

```
SQL> revoke insert on emp from keegan1
Revoke succeeded.

SQL> connect;
```

```
Enter user-name: keegan1
Enter password:
Connected.

SQL> insert into system.emp values (7,'ZIIU','UK',1230967890,700000);
insert into system.emp values (7,'ZIIU','UK',1230967890,700000)
            *
ERROR at line 1:
ORA-00942: table or view does not exist
```

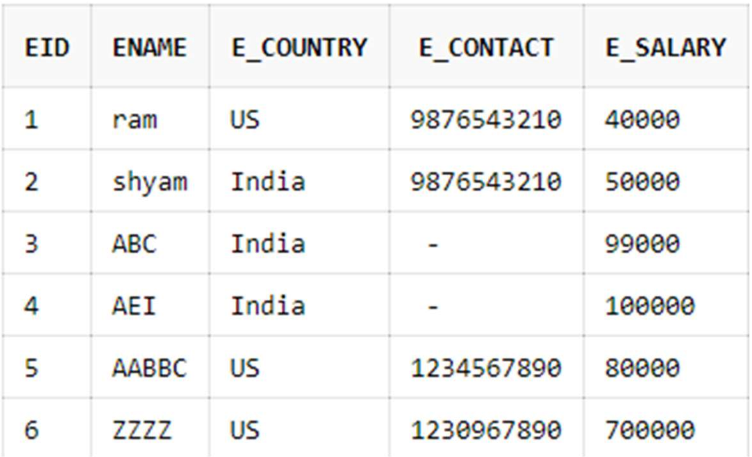
e. login to your user account, retrieve the table records using select

```
SQL> connect;
Enter user-name: system
Enter password:
Connected.

SQL> select * from emp;
```



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |

```
SQL> grant select on emp to keegan1;

Grant succeeded.
SQL> connect;
Enter user-name: keegan1
Enter password:
Connected.
SQL> select * from system.emp;
```



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |

**6 rows selected.**

f. Save your transaction using commit.

**SQL> commit;**

**Commit complete.**

g. Insert 1 records using DML Syntax insert through your user login.

**SQL> connect;**

**Enter user-name: system**

**Enter password:**

**Connected.**

**SQL> insert into emp values (7,'RIYA','US',123123890,900000);**

**1 row created.**

h. Create savepoint A

**SQL> savepoint s1;**

**Savepoint created.**

i. Insert 1 records using DML Syntax insert through your user login.

```
SQL> insert into emp values (8,'MAYA','US',1230967890,600000);
1 row created.
```

j. Create savepoint B

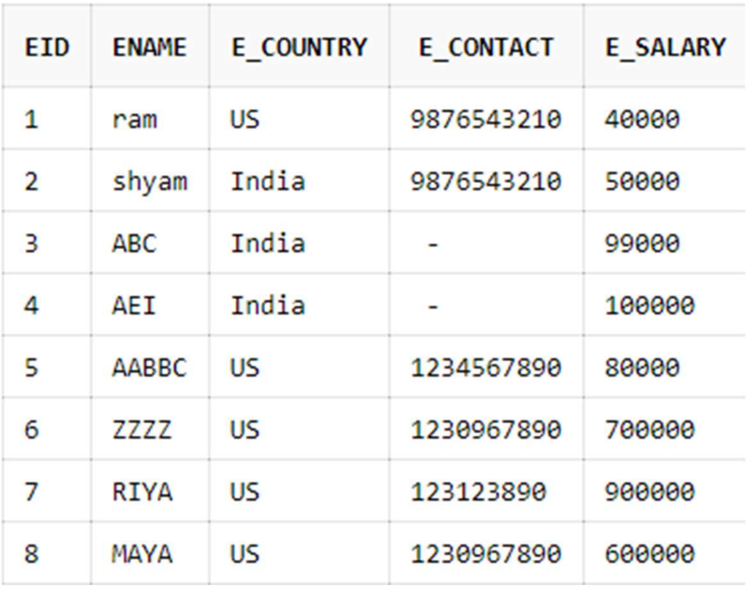
```
 SQL> savepoint s2;
Savepoint created.
```

k. Roll back till save point B

```
SQL> rollback to s2;
Rollback complete.
```

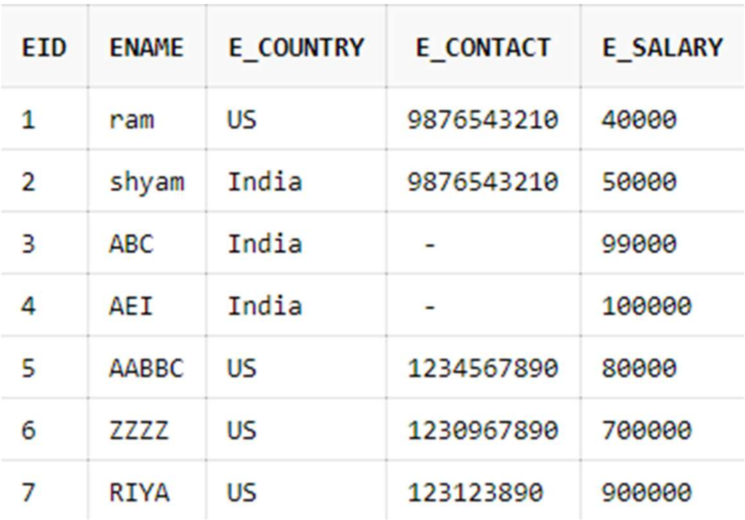
l. Check the table records

```
SQL> select * from emp;
```



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |
| 7 | RIYA | US | 123123890 | 900000 |
| 8 | MAYA | US | 1230967890 | 600000 |

```
8 rows selected.
SQL> rollback to s1;
Rollback complete.

SQL> select * from emp;
```



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |
| 7 | RIYA | US | 123123890 | 900000 |

**7 rows selected.**

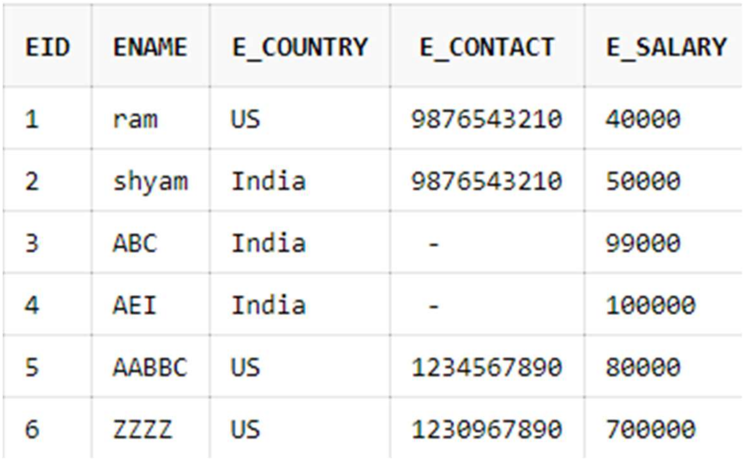
m. perform Rollback

**SQL> rollback;**

**Rollback complete.**

n. Check the table records again.

**SQL> select * from emp;**



| EID | ENAME | E_COUNTRY | E_CONTACT | E_SALARY |
|---|---|---|---|---|
| 1 | ram | US | 9876543210 | 40000 |
| 2 | shyam | India | 9876543210 | 50000 |
| 3 | ABC | India | - | 99000 |
| 4 | AEI | India | - | 100000 |
| 5 | AABBC | US | 1234567890 | 80000 |
| 6 | ZZZZ | US | 1230967890 | 700000 |

**6 rows selected.**

**SQL> commit;**

**Commit complete.**

**Conclusion:**

We learned about SQL integrity constraints and how to use them with examples and we learned how to implement various TCL and DCL commands to provide security to data.