UCS310- DATA BASE MANAGEMENT SYSTEM

A

Project Report

On

Data Sharing in DC++ Alternative

THAPAR UNIVERSITY

Submitted By

Ansh Gujral (101603046)

Submitted To

Miss. Anika

Computer Science and Engineering Department,

Thapar University, Patiala.

`

Table of Contents

`

INTRODUCTION

DC++ is a free and open-source, peer-to-peer file-sharing client that can be used for connecting to the Direct Connect network or to the ADC protocol. It is developed primarily by Jacek Sieka, nicknamed arnetheduck.

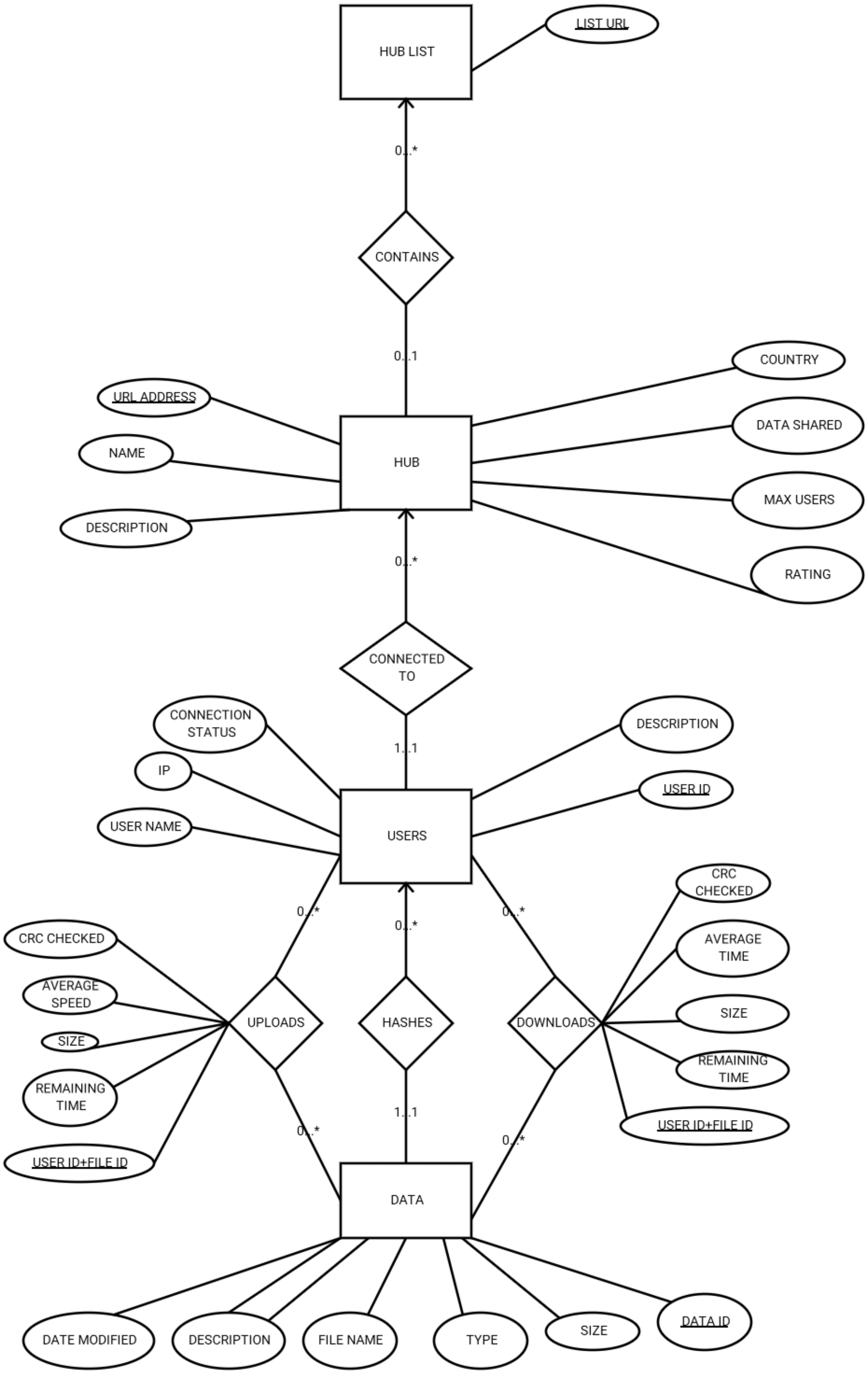
An alternative version of DC++ has been prepared i.e its database which describes the detailed information of the hubs,users using it and data uplaoded and downloaded by the respective users.

ADVANTAGES OF DC++

Typically, a hub on DC++ contains more than 200TB data and you can find almost everything you want on it. Another reason why it is so close to the hearts of students is the fact that data never really gets lost on it, so you can still find those awesome pictures you shared 3 years ago and never have to worry about losing them. This feature is very attractive from an ex-student’s perspective. For alumni, the DC++ is an integral part of their memories and can help in rebuilding the events they went through during their graduation.

`

ER-DIAGRAM


HUB LIST
LIST URL
0..*
CONTAINS
0..1
URL ADDRESS
NAME
DESCRIPTION
HUB
COUNTRY
DATA SHARED
MAX USERS
RATING
0..*
CONNECTED TO
1..1
CONNECTION STATUS
IP
USER NAME
USERS
DESCRIPTION
USER ID
0..*
0..*
0..*
CRC CHECKED
AVERAGE SPEED
SIZE
REMAINING TIME
USER ID+FILE ID
UPLOADS
HASHES
DOWNLOADS
CRC CHECKED
AVERAGE TIME
SIZE
REMAINING TIME
USER ID+FILE ID
0..*
1..1
0..*
DATA
DATE MODIFIED
DESCRIPTION
FILE NAME
TYPE
SIZE
DATA ID

`

ER TO TABLE (BEFORE NORMALIZATION)

HUB LIST

COLUMN	**CONSTRAINT TYPE**	**DATA TYPE**
LIST URL	PRIMARY KEY	VARCHAR2(100)

HUB

COLUMN	**CONSTRAINT TYPE**	**DATA TYPE**
URL ADDRESS	PRIMARY KEY	VARCHAR2(100)
NAME	NOT NULL	VARCHAR2(100)
DESCRIPTION	NOT NULL	VARCHAR2(1000)
COUNTRY	NOT NULL	VARCHAR2(30)
DATA SHARED	NOT NULL	VARCHAR2(30)
MAX USERS	NOT NULL	NUMBER(10)
RATING	CHECK(BETWEEN 0 AND 5)	NUMBER(1)
LIST URL	FOREIGN KEY(HUB LIST LIST URL)	VARCHAR2(100)

USERS

COLUMN	**CONSTRAINT TYPE**	**DATA TYPE**
IP	NOT NULL	VARCHAR2(15)
USER ID	PRIMARY KEY	VARCHAR2(10)
DESCRIPTION	NOT NULL	VARCHAR2(1000)
USER NAME	NOT NULL	VARCHAR2(30)
CONNECTION STATUS	CHECK(CONNECTED/NOT CONNECTED)	VARCHAR2(20)
URL ADDRESS	FOREIGN KEY(URL ADDRESS HUB)	VARCHAR2(100)

DATA

COLUMN	**CONSTRAINT TYPE**	**DATA TYPE**
USER ID	FOREIGN KEY(USER ID USERS)	VARCHAR2(10)
DATA ID	PRIMARY KEY	VARCHAR2(10)
DATE MODIFIED	NOT NULL	DATE
DESCRIPTION	NOT NULL	VARCHAR2(1000)
FILE NAME	NOT NULL	VARCHAR2(30)
TYPE	NOT NULL	VARCHAR2(10)
SIZE	NOT NULL	VARCHAR2(20)

`

UPLOADS

COLUMN	CONSTRAINT TYPE	DATA TYPE
USER ID+DATA ID	PRIMARY KEY	VARCHAR2(20)
REMAINING TIME	NOT NULL	VARCHAR2(20)
SIZE	NOT NULL	VARCHAR2(20)
AVERAGE SPEED	NOT NULL	VARCHAR2(20)
CRC CHECKED	CHECK(YES/NO)	VARCHAR2(3)
USER ID	FOREIGN KEY(USER ID USERS)	VARCHAR(10)

DOWNLOADS

COLUMN	CONSTRAINT TYPE	DATA TYPE
USER ID+DATA ID	PRIMARY KEY	VARCHAR2(20)
REMAINING TIME	NOT NULL	DATE(HH:MM:SS)
SIZE	NOT NULL	VARCHAR2(20)
AVERAGE SPEED	NOT NULL	VARCHAR2(20)
CRC CHECKED	CHECK(YES/NO)	VARCHAR2(3)
USER ID	FOREIGN KEY(USER ID USERS)	VARCHAR(10)
DATA ID	FOREIGN KEY(DATA ID DATA)	VARCHAR(10)

`

CASE STUDY

This is case study of Entity-Relationship of alternative version of DC++ which is a well-known peer to peer data sharing client.

One hub list contains many hubs. Many hubs can be contained by only one hub list.

One hub has connected many users. Many users are connected to only one hub. Each and every user is connected a hub.

One user hashes many files. Many files are being hashed by only one user. Each and every file is being hashed.

Many users can upload many files.

Many users can download many files.

`

TABLES AFTER NORMALISATION

HUB_LIST

Name	Null?	Type
URL_ADDRESS	NOT NULL	VARCHAR2(100)
LIST_URL	NOT NULL	VARCHAR2(100)

HUB

Name	Null?	Type
URL_ADDRESS	NOT NULL	VARCHAR2(100)
NAME	NOT NULL	VARCHAR2(100)
DESCRIPTION	NOT NULL	VARCHAR2(1000)
COUNTRY	NOT NULL	VARCHAR2(100)
DATA_SHARED	NOT NULL	VARCHAR2(100)
MAX_USERS	NOT NULL	NUMBER(10)
RATING	NOT NULL	NUMBER(1)

CONNECTION

Name	Null?	Type
URL_ADDRESS	NOT NULL	VARCHAR2(100)
USER_ID	NOT NULL	VARCHAR2(100)

UPLOADS

Name	Null?	Type
DATA_ID	NOT NULL	VARCHAR2(100)
USER_ID	NOT NULL	VARCHAR2(100)
REMAINING_TIME	NOT NULL	VARCHAR2(100)
AVERAGE_SPEED	NOT NULL	VARCHAR2(20)
CRC_CHECKED		VARCHAR2(5)

`

DOWNLOADS

Name	Null?	Type
DATA_ID	NOT NULL	VARCHAR2(100)
USER_ID	NOT NULL	VARCHAR2(100)
REMAINING_TIME	NOT NULL	VARCHAR2(100)
AVERAGE_SPEED	NOT NULL	VARCHAR2(20)
CRC_CHECKED		VARCHAR2(5)

DATA

Name	Null?	Type
DATA_ID	NOT NULL	VARCHAR2(100)
DATE_MODIFIED	NOT NULL	DATE
DESCRIPTION	NOT NULL	VARCHAR2(1000)
FILE_NAME	NOT NULL	VARCHAR2(15)
TYPE	NOT NULL	VARCHAR2(10)
FILE_SIZE	NOT NULL	VARCHAR2(20)

USER

Name	Null?	Type
USER_ID	NOT NULL	VARCHAR2(100)
USER_NAME	NOT NULL	VARCHAR2(100)
DESCRIPTION	NOT NULL	VARCHAR2(1000)
IP	NOT NULL	VARCHAR2(15)
CONNECTION_STATUS	NOT NULL	VARCHAR2(20)

HASHING

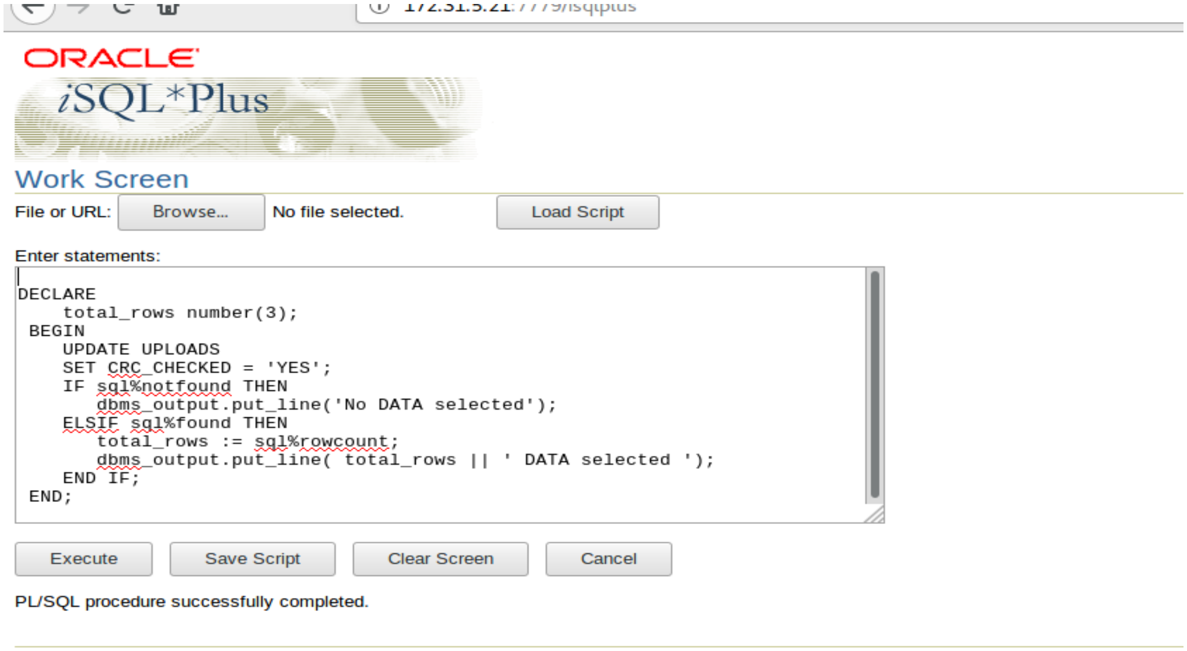
Name	Null?	Type
DATA_ID	NOT NULL	VARCHAR2(100)
USER_ID	NOT NULL	VARCHAR2(100)

`

PL/SQL CODES

---CURSOR THAT UPLOADS ALL THE ENTRIES INSIDE CRC_CHECKED OF TABLE UPLAODS TO "YES"

```
DECLARE
  total_rows number(3);
BEGIN
  UPDATE UPLOADS
  SET CRC_CHECKED = 'YES';
  IF sql%notfound THEN
    dbms_output.put_line('No DATA selected');
  ELSIF sql%found THEN
    total_rows := sql%rowcount;
    dbms_output.put_line( total_rows || ' DATA selected ');
  END IF;
END;
```



```
DECLARE
    total_rows number(3);
 BEGIN
    UPDATE UPLOADS
    SET CRC_CHECKED = 'YES';
    IF sql%notfound THEN
       dbms_output.put_line('No DATA selected');
    ELSIF sql%found THEN
       total_rows := sql%rowcount;
       dbms_output.put_line( total_rows || ' DATA selected ');
    END IF;
 END;
```

PL/SQL procedure successfully completed.

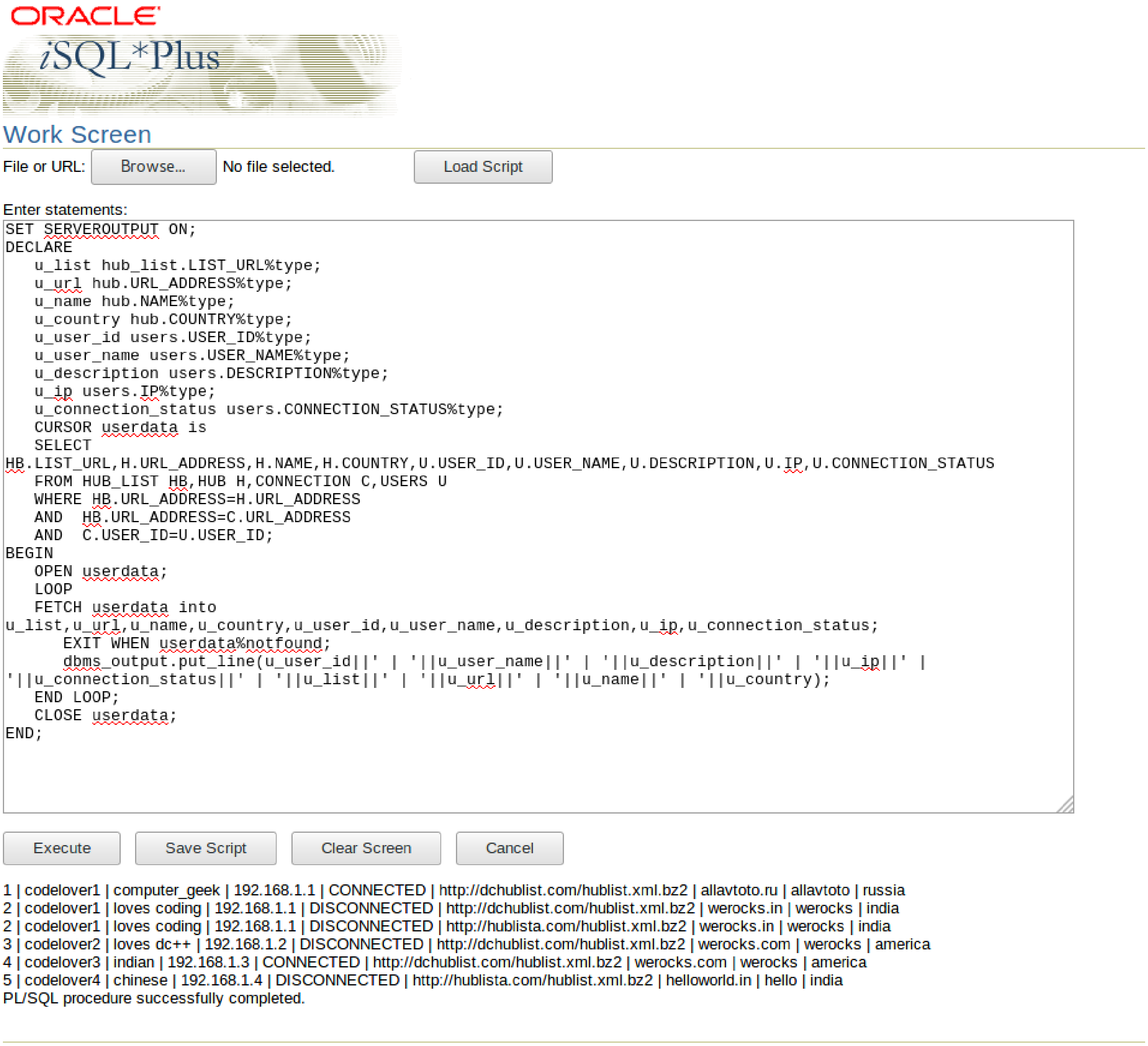
`

--CURSOR USED TO RETRIEVE THE DETAILED INFORMATION OF THE USERS WHO ARE CONNECTED TO ATLEAST ONE HUB

```
DECLARE
  u_list hub_list.LIST_URL%type;
  u_url hub.URL_ADDRESS%type;
  u_name hub.NAME%type;
  u_country hub.COUNTRY%type;
  u_user_id users.USER_ID%type;
  u_user_name users.USER_NAME%type;
  u_description users.DESCRIPTION%type;
  u_ip users.IP%type;
  u_connection_status users.CONNECTION_STATUS%type;
  CURSOR userdata is
  SELECT HB.LIST_URL,H.URL_ADDRESS,H.NAME,H.COUNTRY,U.USER_ID,U.USER_NAME,U.DESCRIPTION,U.IP,U.CONNECTION_STATUS
  FROM HUB_LIST HB,HUB H,CONNECTION C,USERS U
  WHERE HB.URL_ADDRESS=H.URL_ADDRESS
  AND  HB.URL_ADDRESS=C.URL_ADDRESS
  AND  C.USER_ID=U.USER_ID;
BEGIN
  OPEN userdata;
  LOOP
  FETCH userdata into u_list,u_url,u_name,u_country,u_user_id,u_user_name,u_description,u_ip,u_connection_status;
    EXIT WHEN userdata%notfound;
```

`

```
    dbms_output.put_line(u_user_id||' | '||u_user_name||' | '||u_description||' | '||u_ip||' |
'||u_connection_status||' | '||u_list||' | '||u_url||' | '||u_name||' | '||u_country);

  END LOOP;

  CLOSE userdata;

END;
```



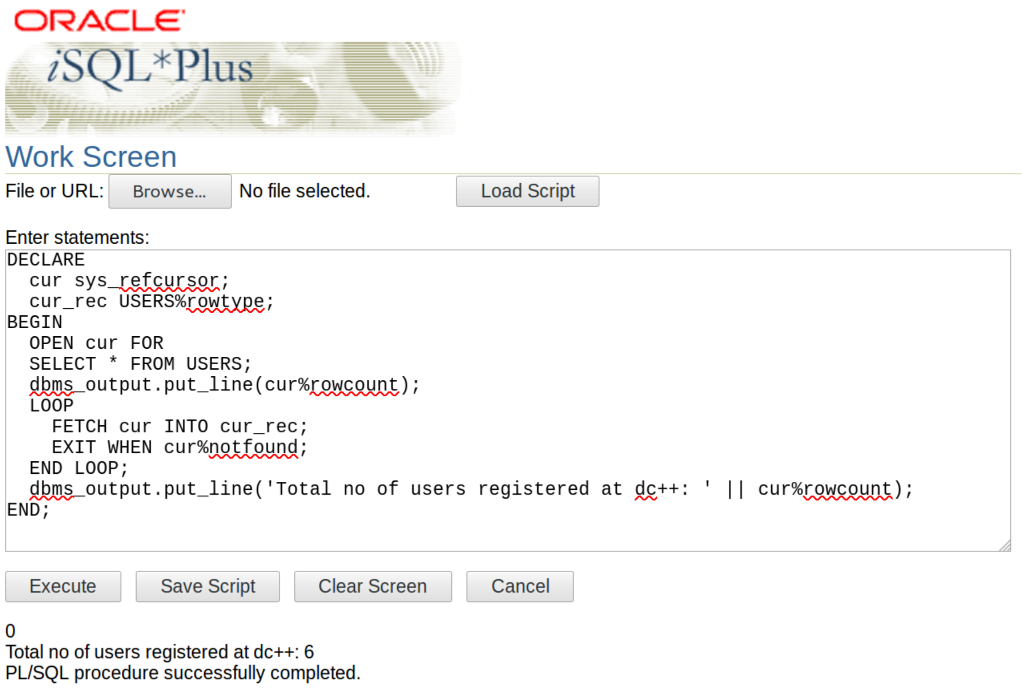
```
SET SERVEROUTPUT ON;
DECLARE
   u_list hub_list.LIST_URL%type;
   u_url hub.URL_ADDRESS%type;
   u_name hub.NAME%type;
   u_country hub.COUNTRY%type;
   u_user_id users.USER_ID%type;
   u_user_name users.USER_NAME%type;
   u_description users.DESCRIPTION%type;
   u_ip users.IP%type;
   u_connection_status users.CONNECTION_STATUS%type;
   CURSOR userdata is
   SELECT
HB.LIST_URL,H.URL_ADDRESS,H.NAME,H.COUNTRY,U.USER_ID,U.USER_NAME,U.DESCRIPTION,U.IP,U.CONNECTION_STATUS
   FROM HUB_LIST HB,HUB H,CONNECTION C,USERS U
   WHERE HB.URL_ADDRESS=H.URL_ADDRESS
   AND  HB.URL_ADDRESS=C.URL_ADDRESS
   AND  C.USER_ID=U.USER_ID;
BEGIN
   OPEN userdata;
   LOOP
   FETCH userdata into
u_list,u_url,u_name,u_country,u_user_id,u_user_name,u_description,u_ip,u_connection_status;
      EXIT WHEN userdata%notfound;
      dbms_output.put_line(u_user_id||' | '||u_user_name||' | '||u_description||' | '||u_ip||' |
'||u_connection_status||' | '||u_list||' | '||u_url||' | '||u_name||' | '||u_country);
   END LOOP;
   CLOSE userdata;
END;
```

```
1 | codelover1 | computer_geek | 192.168.1.1 | CONNECTED | http://dchublist.com/hublist.xml.bz2 | allavtoto.ru | allavtoto | russia
2 | codelover1 | loves coding | 192.168.1.1 | DISCONNECTED | http://dchublist.com/hublist.xml.bz2 | werocks.in | werocks | india
2 | codelover1 | loves coding | 192.168.1.1 | DISCONNECTED | http://hublista.com/hublist.xml.bz2 | werocks.in | werocks | india
3 | codelover2 | loves dc++ | 192.168.1.2 | DISCONNECTED | http://dchublist.com/hublist.xml.bz2 | werocks.com | werocks | america
4 | codelover3 | indian | 192.168.1.3 | CONNECTED | http://dchublist.com/hublist.xml.bz2 | werocks.com | werocks | america
5 | codelover4 | chinese | 192.168.1.4 | DISCONNECTED | http://hublista.com/hublist.xml.bz2 | helloworld.in | hello | india
PL/SQL procedure successfully completed.
```

`

---CURSOR THAT COUNTS THE NUMBER OF USERS REGISTERED AT DC++

```
DECLARE
  cur sys_refcursor;
  cur_rec USERS%rowtype;
BEGIN
  OPEN cur FOR
  SELECT * FROM USERS;
  dbms_output.put_line(cur%rowcount);
  LOOP
    FETCH cur INTO cur_rec;
    EXIT WHEN cur%notfound;
  END LOOP;
  dbms_output.put_line('Total no of users registered at dc++: ' || cur%rowcount);
END;
```



```
DECLARE
  cur sys_refcursor;
  cur_rec USERS%rowtype;
BEGIN
  OPEN cur FOR
  SELECT * FROM USERS;
  dbms_output.put_line(cur%rowcount);
  LOOP
    FETCH cur INTO cur_rec;
    EXIT WHEN cur%notfound;
  END LOOP;
  dbms_output.put_line('Total no of users registered at dc++: ' || cur%rowcount);
END;
```

```
0
Total no of users registered at dc++: 6
PL/SQL procedure successfully completed.
```

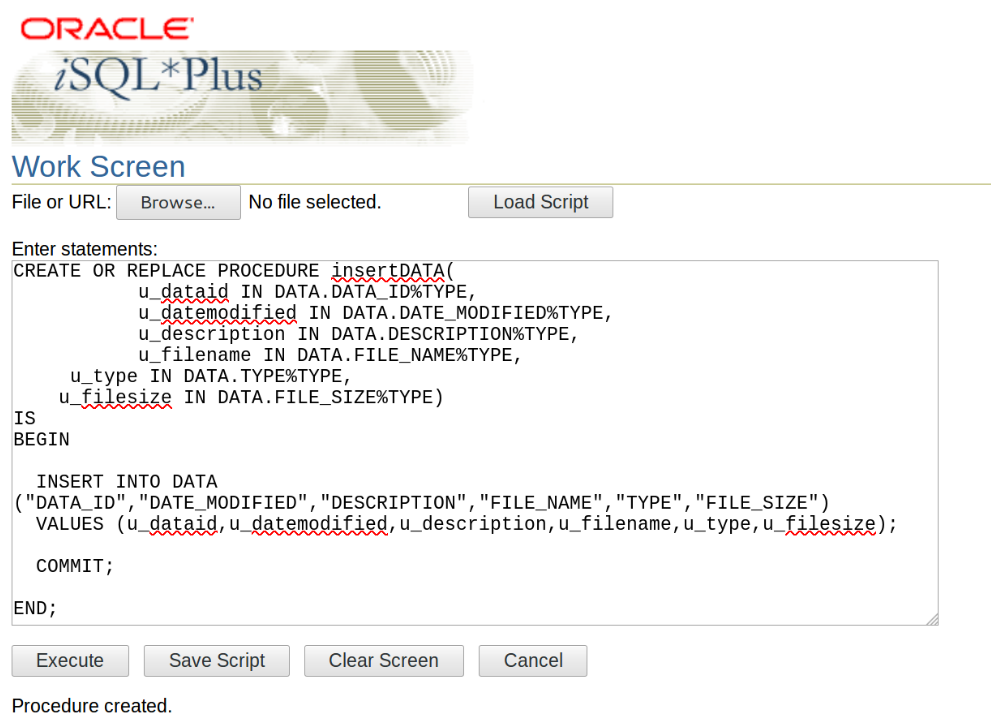
`

---PROCEDURE FOR INSERTING DATA IN DATA TABLE

```
CREATE OR REPLACE PROCEDURE insertDATA(
            u_dataid IN DATA.DATA_ID%TYPE,
            u_datemodified IN DATA.DATE_MODIFIED%TYPE,
            u_description IN DATA.DESCRIPTION%TYPE,
            u_filename IN DATA.FILE_NAME%TYPE,
   u_type IN DATA.TYPE%TYPE,
  u_filesize IN DATA.FILE_SIZE%TYPE)
IS
BEGIN

 INSERT INTO DATA ("DATA_ID","DATE_MODIFIED","DESCRIPTION","FILE_NAME","TYPE","FILE_SIZE")
 VALUES (u_dataid,u_datemodified,u_description,u_filename,u_type,u_filesize);

 COMMIT;

END;
```

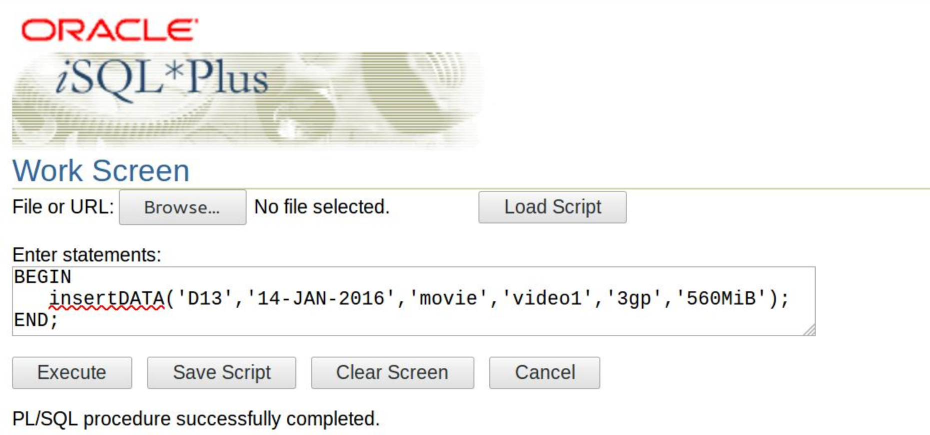
```
BEGIN
  insertDATA('D13','14-JAN-2016','movie','video1','3gp','560MiB');
END;
```

`



```
CREATE OR REPLACE PROCEDURE insertDATA(
           u_dataid IN DATA.DATA_ID%TYPE,
           u_datemodified IN DATA.DATE_MODIFIED%TYPE,
           u_description IN DATA.DESCRIPTION%TYPE,
           u_filename IN DATA.FILE_NAME%TYPE,
     u_type IN DATA.TYPE%TYPE,
    u_filesize IN DATA.FILE_SIZE%TYPE)
IS
BEGIN

  INSERT INTO DATA
("DATA_ID","DATE_MODIFIED","DESCRIPTION","FILE_NAME","TYPE","FILE_SIZE")
  VALUES (u_dataid,u_datemodified,u_description,u_filename,u_type,u_filesize);

  COMMIT;

END;
```

```
Procedure created.
```



```
BEGIN
   insertDATA('D13','14-JAN-2016','movie','video1','3gp','560MiB');
END;
```

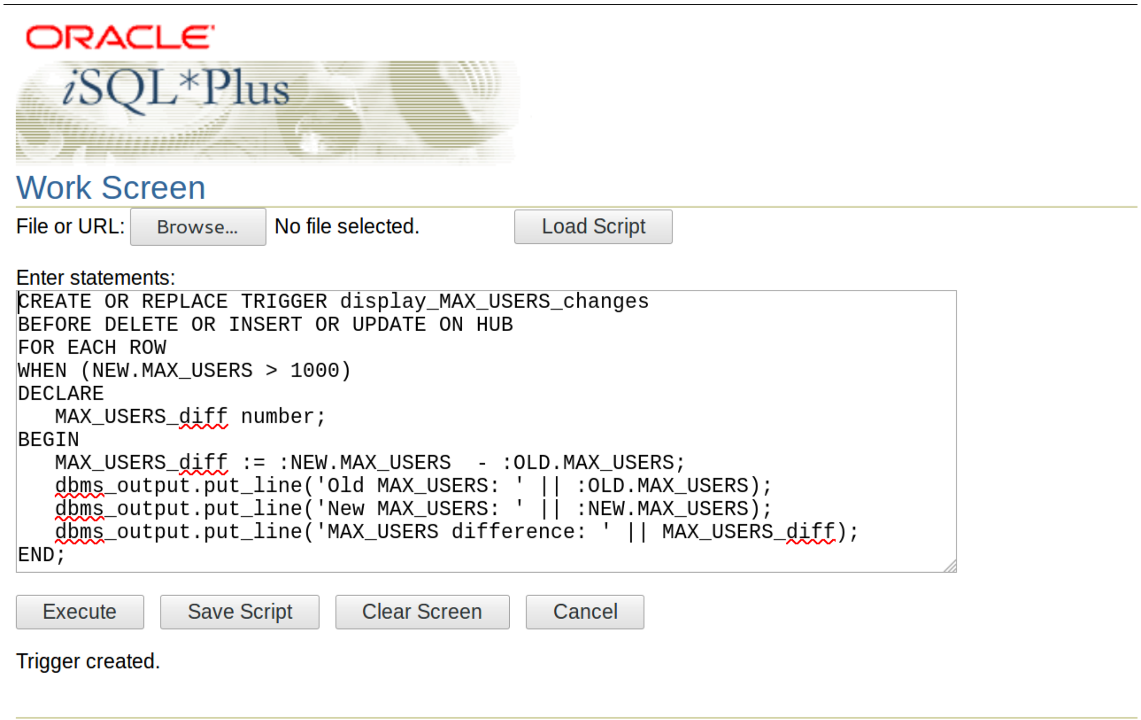
```
PL/SQL procedure successfully completed.
```

---TRIGGER THAT TELLS THE UPDATE DONE IN MAX USER CAPACITY IN HUB TABLE

```
CREATE OR REPLACE TRIGGER display_MAX_USERS_changes

BEFORE DELETE OR INSERT OR UPDATE ON HUB

FOR EACH ROW

WHEN (NEW.MAX_USERS > 1000)

DECLARE

  MAX_USERS_diff number;
```

`

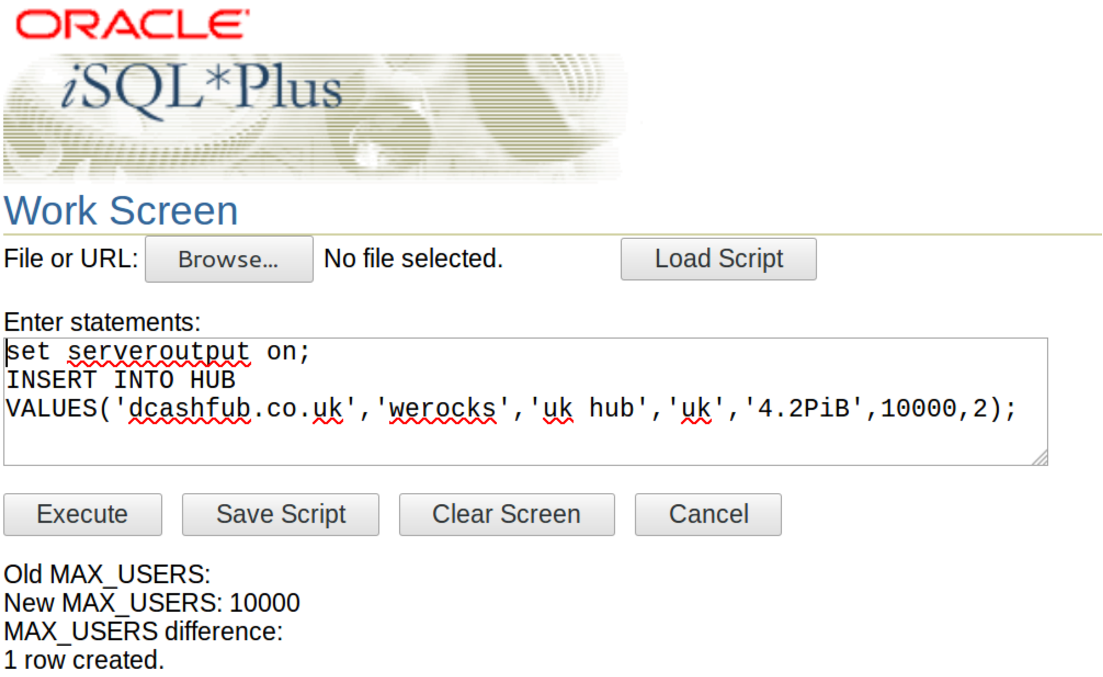
```
BEGIN
   MAX_USERS_diff := :NEW.MAX_USERS  - :OLD.MAX_USERS;
   dbms_output.put_line('Old MAX_USERS: ' || :OLD.MAX_USERS);
   dbms_output.put_line('New MAX_USERS: ' || :NEW.MAX_USERS);
   dbms_output.put_line('MAX_USERS difference: ' || MAX_USERS_diff);
END;
---
set serveroutput on;
INSERT INTO HUB
VALUES('dcashfub.co.uk','werocks','uk hub','uk','4.2PiB',10000,2);
```



```
CREATE OR REPLACE TRIGGER display_MAX_USERS_changes
BEFORE DELETE OR INSERT OR UPDATE ON HUB
FOR EACH ROW
WHEN (NEW.MAX_USERS > 1000)
DECLARE
   MAX_USERS_diff number;
BEGIN
   MAX_USERS_diff := :NEW.MAX_USERS  - :OLD.MAX_USERS;
   dbms_output.put_line('Old MAX_USERS: ' || :OLD.MAX_USERS);
   dbms_output.put_line('New MAX_USERS: ' || :NEW.MAX_USERS);
   dbms_output.put_line('MAX_USERS difference: ' || MAX_USERS_diff);
END;
```

```
Trigger created.
```

`



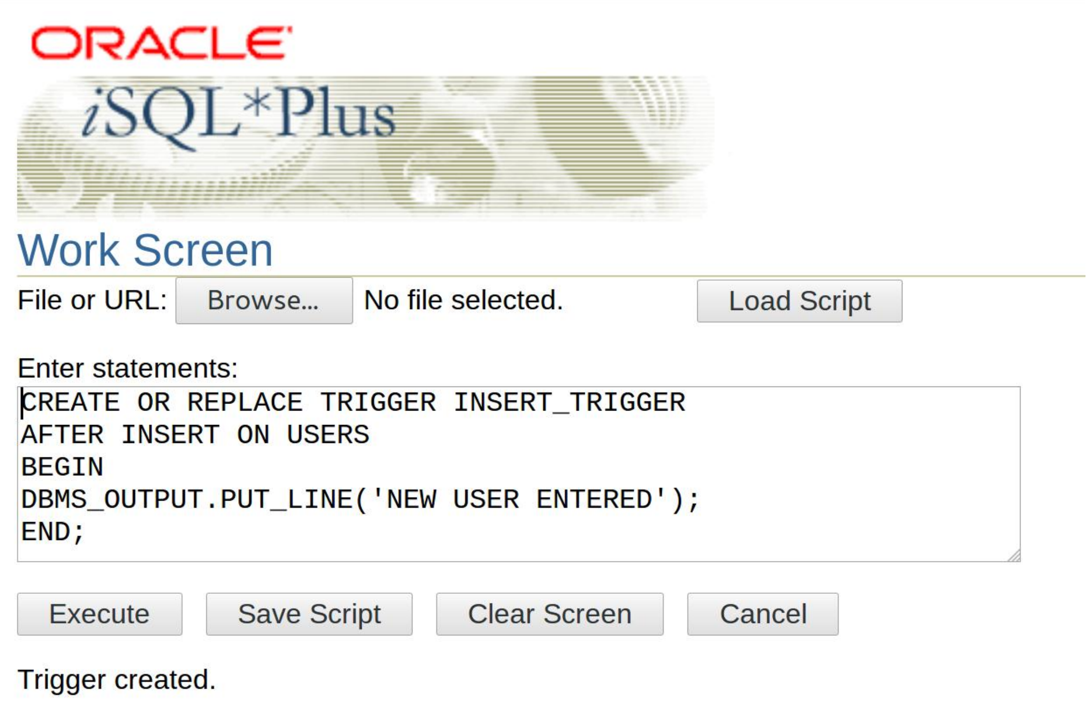
```
set serveroutput on;
INSERT INTO HUB
VALUES('dcashfub.co.uk','werocks','uk hub','uk','4.2PiB',10000,2);
```

```
Old MAX_USERS:
New MAX_USERS: 10000
MAX_USERS difference:
1 row created.
```

---TRIGGER THAT NOTIFIES THAT NEW USER HAS REGISTERED TO DC++

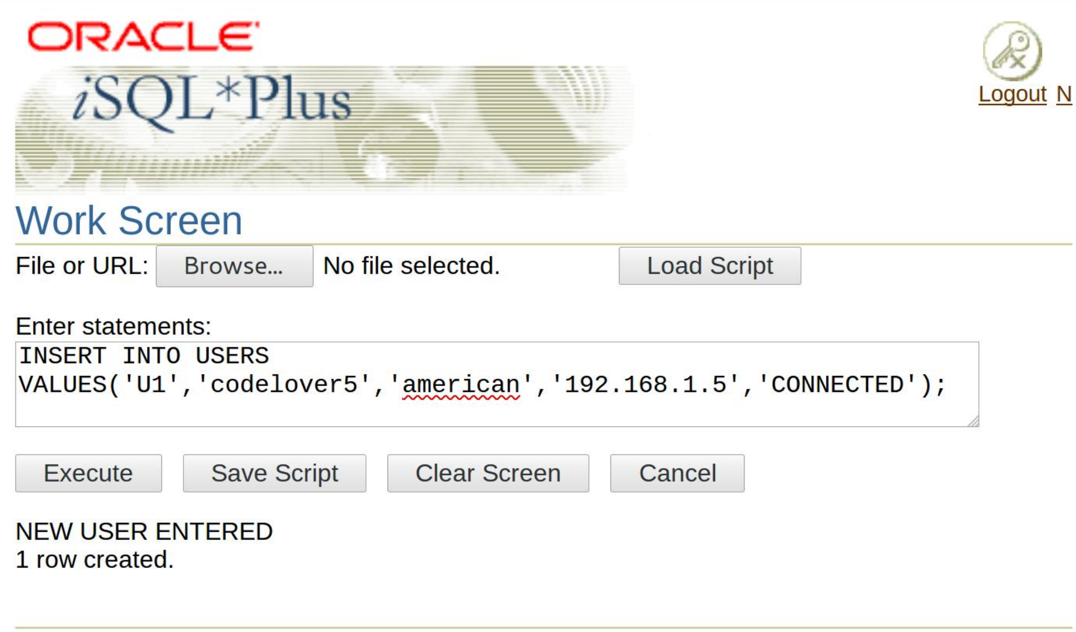
```
CREATE OR REPLACE TRIGGER INSERT_TRIGGER

AFTER INSERT ON USERS

BEGIN

DBMS_OUTPUT.PUT_LINE('NEW USER ENTERED');

END;

---

INSERT INTO USERS

VALUES('U1','codelover5','american','192.168.1.5','CONNECTED');
```

`



```
CREATE OR REPLACE TRIGGER INSERT_TRIGGER
AFTER INSERT ON USERS
BEGIN
DBMS_OUTPUT.PUT_LINE('NEW USER ENTERED');
END;
```

```
Trigger created.
```



```
INSERT INTO USERS
VALUES('U1','codelover5','american','192.168.1.5','CONNECTED');
```

```
NEW USER ENTERED
1 row created.
```

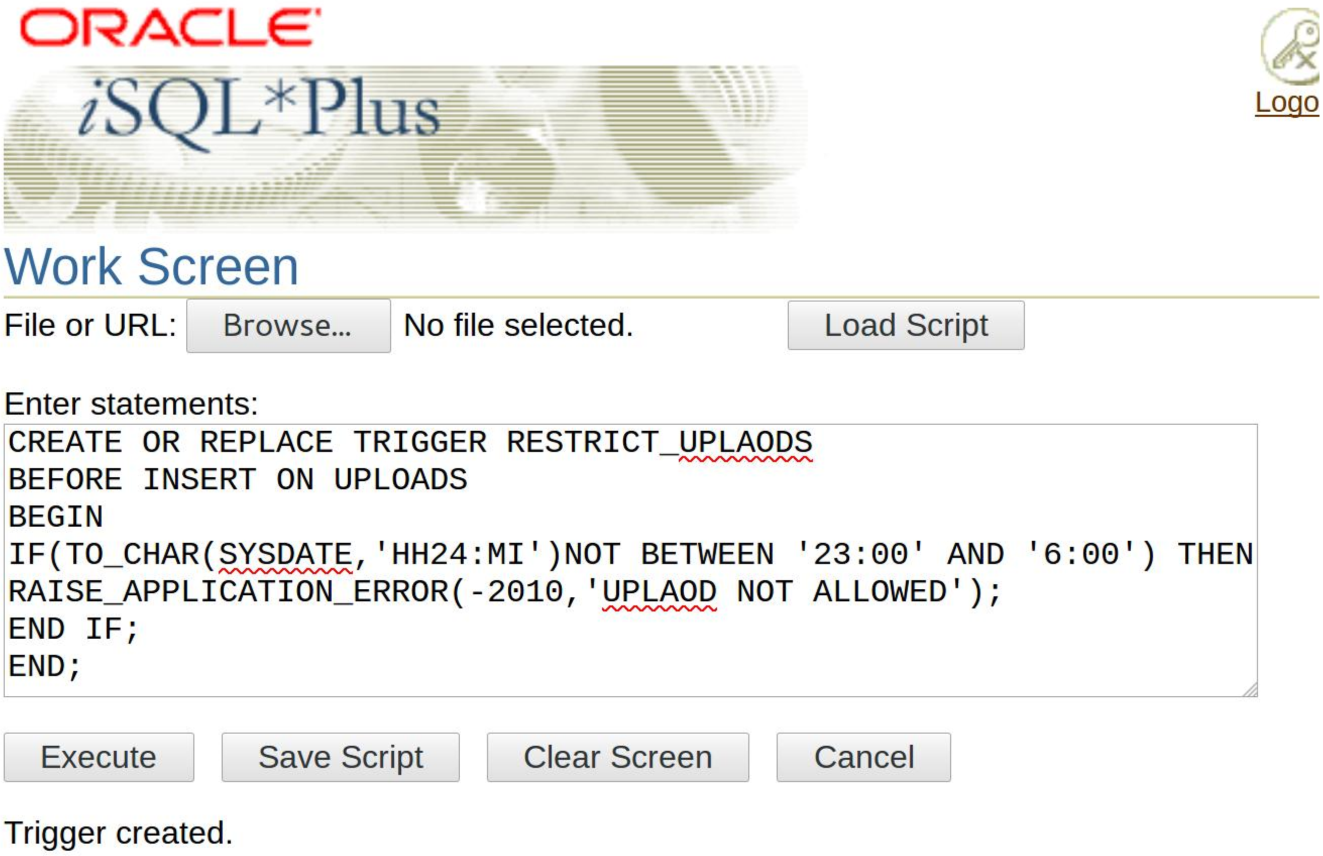
---TRIGGER THAT DOESNT ALLOW UPLOAD BETWEEN 11 PM AND 6 AM

CREATE OR REPLACE TRIGGER RESTRICT_UPLAODS

BEFORE INSERT ON UPLOADS

`

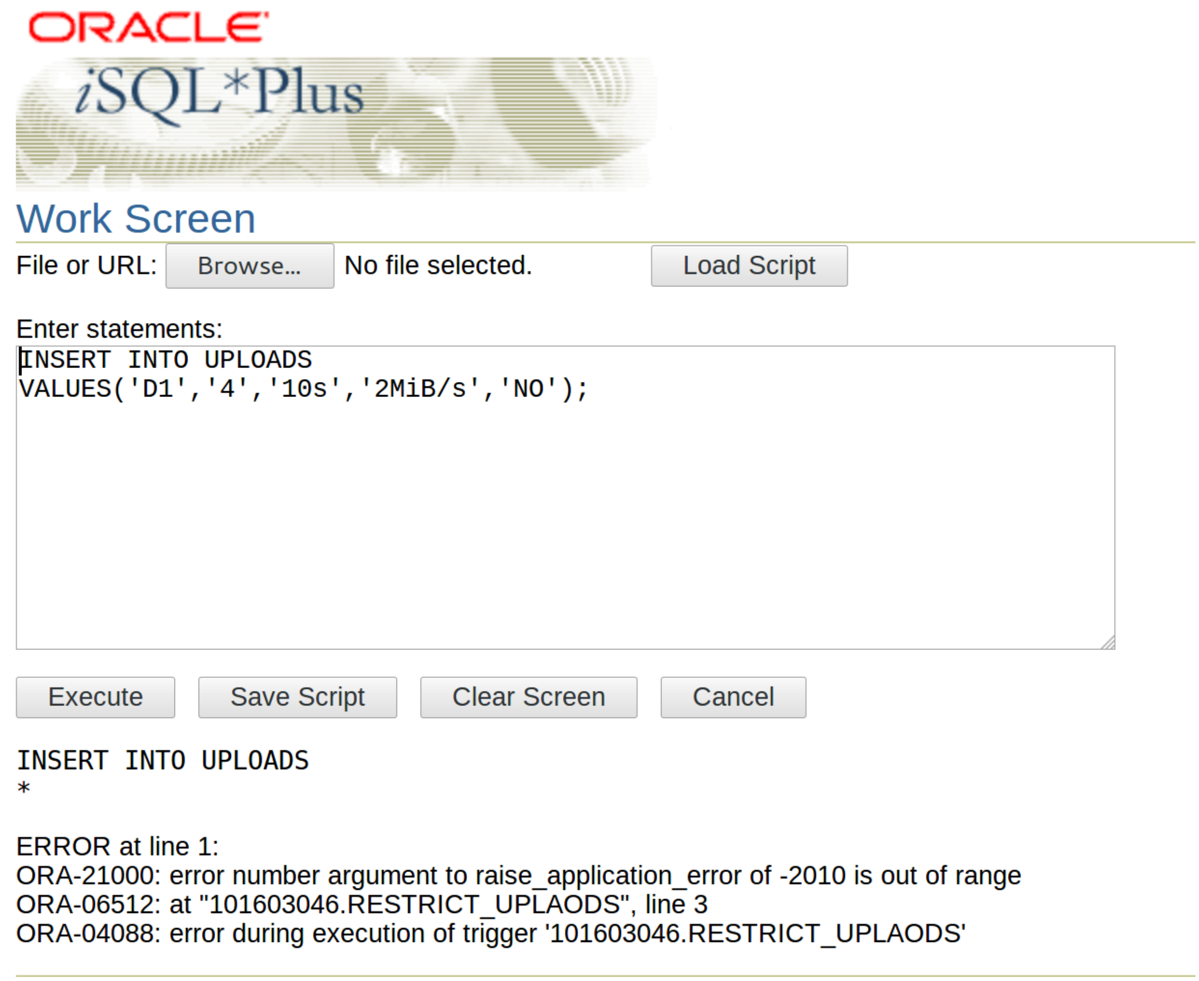
```
BEGIN

IF(TO_CHAR(SYSDATE,'HH24:MI')NOT BETWEEN '23:00' AND '6:00') THEN

RAISE_APPLICATION_ERROR(-2010,'UPLAOD NOT ALLOWED');

END IF;

END;

---

INSERT INTO UPLOADS

VALUES('D2','4','10s','2MiB/s','NO');
```



```
CREATE OR REPLACE TRIGGER RESTRICT_UPLAODS
BEFORE INSERT ON UPLOADS
BEGIN
IF(TO_CHAR(SYSDATE,'HH24:MI')NOT BETWEEN '23:00' AND '6:00') THEN
RAISE_APPLICATION_ERROR(-2010,'UPLAOD NOT ALLOWED');
END IF;
END;
```

Trigger created.

`



```
INSERT INTO UPLOADS
VALUES('D1','4','10s','2MiB/s','NO');
```

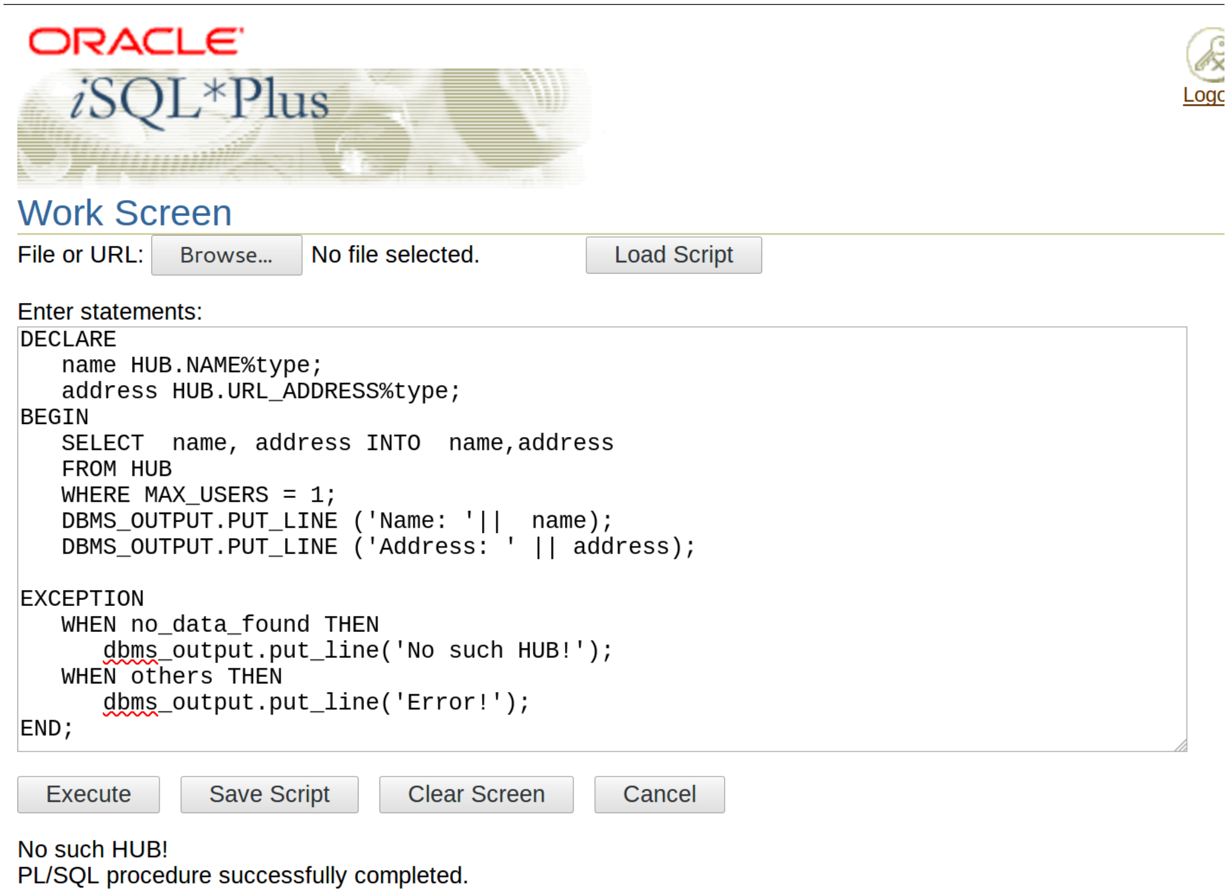
```
INSERT INTO UPLOADS
*

ERROR at line 1:
ORA-21000: error number argument to raise_application_error of -2010 is out of range
ORA-06512: at "101603046.RESTRICT_UPLAODS", line 3
ORA-04088: error during execution of trigger '101603046.RESTRICT_UPLAODS'
```

--EXCEPTION thrown when MAX_USERS=1 DONT EXSISTS

```
DECLARE

  name HUB.NAME%type;

  address HUB.URL_ADDRESS%type;

BEGIN

  SELECT  name, address INTO  name,address

  FROM HUB

  WHERE MAX_USERS = 1;

  DBMS_OUTPUT.PUT_LINE ('Name: '||  name);

  DBMS_OUTPUT.PUT_LINE ('Address: ' || address);
```

`

```
EXCEPTION
  WHEN no_data_found THEN
    dbms_output.put_line('No such HUB!');
  WHEN others THEN
    dbms_output.put_line('Error!');
END;
```

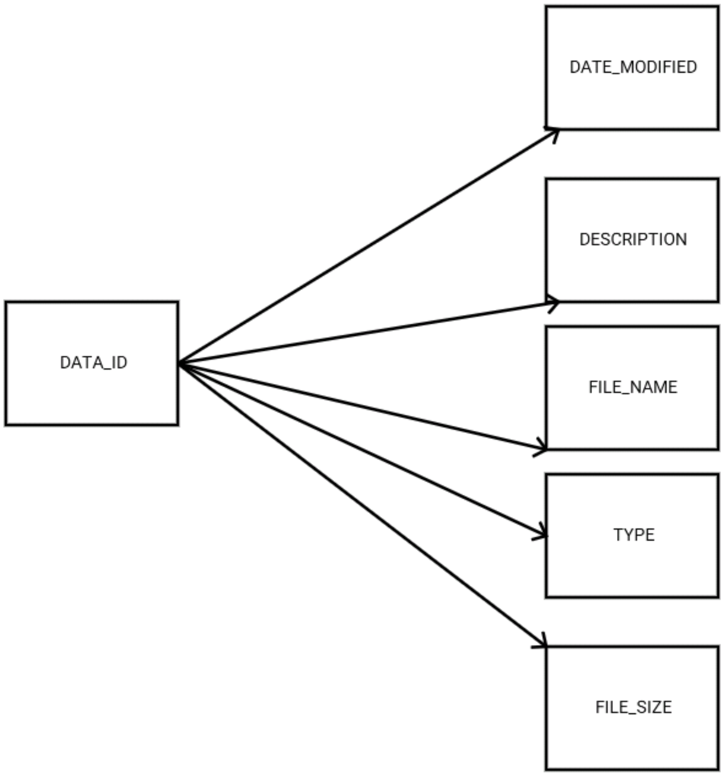


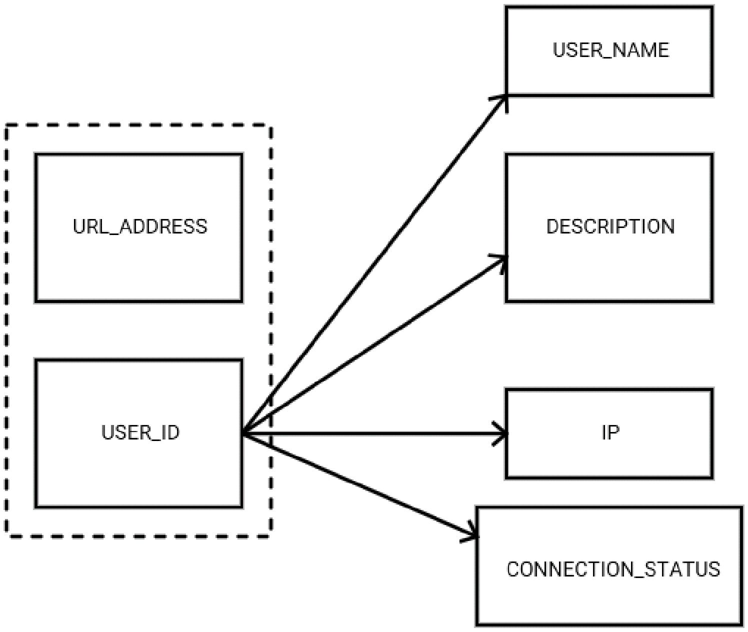
```
DECLARE
   name HUB.NAME%type;
   address HUB.URL_ADDRESS%type;
BEGIN
   SELECT  name, address INTO  name,address
   FROM HUB
   WHERE MAX_USERS = 1;
   DBMS_OUTPUT.PUT_LINE ('Name: '||  name);
   DBMS_OUTPUT.PUT_LINE ('Address: ' || address);

EXCEPTION
   WHEN no_data_found THEN
      dbms_output.put_line('No such HUB!');
   WHEN others THEN
      dbms_output.put_line('Error!');
END;
```

```
No such HUB!
PL/SQL procedure successfully completed.
```

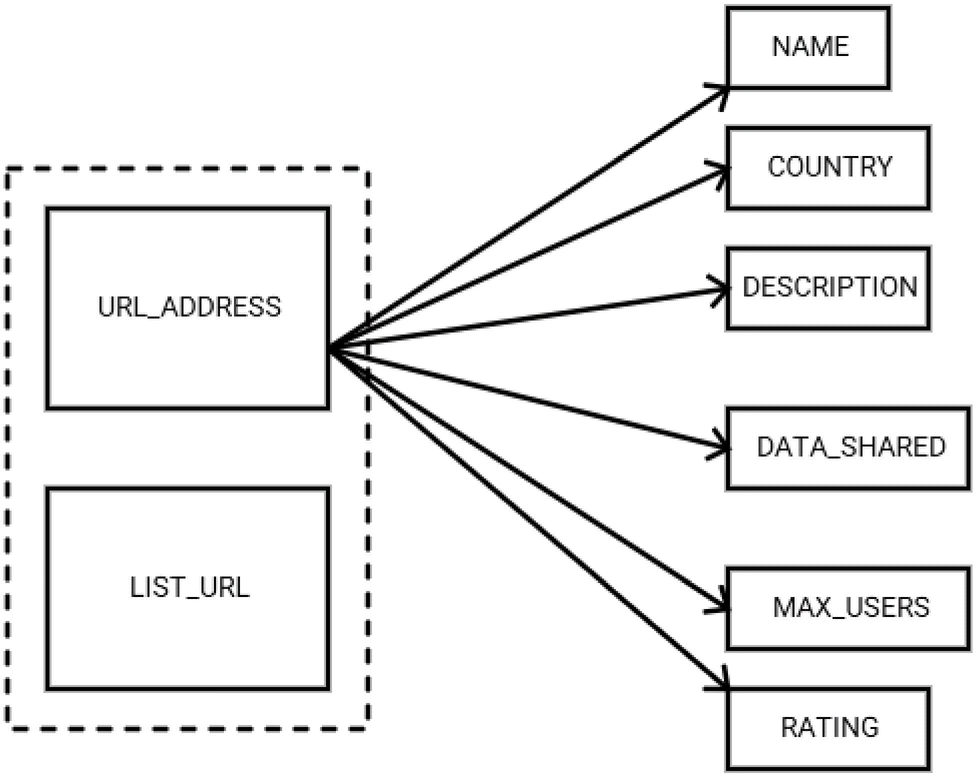
`

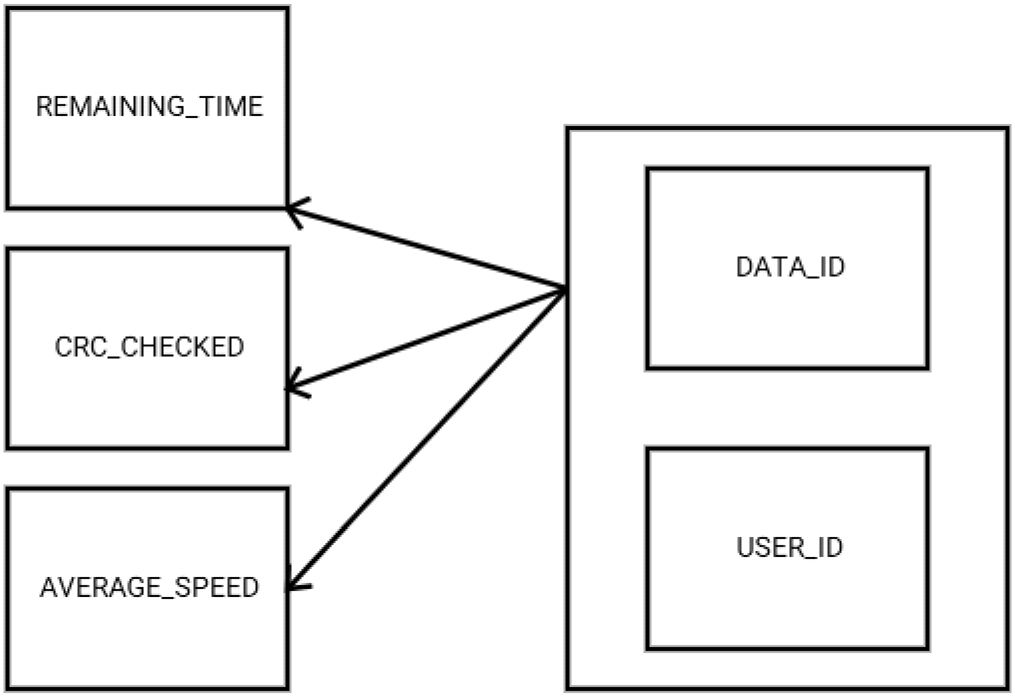
FUNCTIONAL DEPENDENCIES





`


URL_ADDRESS
LIST_URL
NAME
COUNTRY
DESCRIPTION
DATA_SHARED
MAX_USERS
RATING


REMAINING_TIME
CRC_CHECKED
AVERAGE_SPEED
DATA_ID
USER_ID

`

REFERENCES

1) Stack Overflow
2) Tutorials Point
3) Lecture Slides

Soft copy of this project has been uploaded on git hub.

Download link: https://github.com/gujral1997/dbms_project_on_data_sharing_in_dc-_alternative