/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Smalls Federal Database - Version 1.4

Script: Smalls\_Federal\_Oracle.sql

Description: Creates and populates the Smalls Federal database.

DB Server: Oracle

Author: Brandon Smalls

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Create database

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CREATE USER Smalls

IDENTIFIED BY p4ssw0rd

DEFAULT TABLESPACE users

TEMPORARY TABLESPACE temp

QUOTA 10M ON users;

GRANT connect to Smalls;

GRANT resource to Smalls;

GRANT create session TO Smalls;

GRANT create table TO Smalls;

GRANT create view TO Smalls;

conn Smalls/p4ssw0rd

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Create Tables

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

DROP TABLE user\_transaction;

DROP TABLE acct\_user;

DROP SEQUENCE user\_id\_seq;

CREATE TABLE acct\_user

(

user\_id NUMBER PRIMARY KEY,

user\_name VARCHAR2(30) NOT NULL UNIQUE,

password VARCHAR(20),

balance number DEFAULT(0.0)

);

CREATE TABLE user\_transaction

(

trans\_id number PRIMARY KEY,

amount number,

time TIMESTAMP,

user\_id REFERENCES acct\_user(user\_id)

);

CREATE SEQUENCE user\_id\_seq START WITH 1;

CREATE SEQUENCE trans\_id\_seq START WITH 1;

CREATE OR REPLACE TRIGGER trans\_id\_trig

BEFORE INSERT OR UPDATE ON user\_transaction

FOR EACH ROW

BEGIN

IF INSERTING THEN

SELECT trans\_id\_seq.nextVal INTO :new.trans\_id FROM dual;

ELSIF UPDATING THEN

SELECT :old.trans\_id INTO :new.trans\_id FROM dual;

END IF;

END;

/

CREATE OR REPLACE TRIGGER user\_id\_trig

BEFORE INSERT OR UPDATE ON acct\_user

FOR EACH ROW

BEGIN

IF INSERTING THEN

SELECT user\_id\_seq.nextVal INTO :new.user\_id FROM dual;

ELSIF UPDATING THEN

SELECT :old.user\_id INTO :new.user\_id FROM dual;

END IF;

END;

/

CREATE OR REPLACE TRIGGER pop\_trans

AFTER UPDATE ON acct\_user

FOR EACH ROW

BEGIN

INSERT INTO user\_transaction

(trans\_id, amount, time, user\_id)

VALUES(trans\_id\_seq.nextVal,:new.balance, systimestamp, :new.user\_id);

END;

/

INSERT INTO acct\_user(user\_name, password, balance) VALUES('Brandon', 'b123', 500);

INSERT INTO acct\_user(user\_name, password, balance) VALUES('Miguel', 'm123', 400);

INSERT INTO acct\_user(user\_name, password, balance) VALUES('Jack', 'j123', 300);

COMMIT;

SELECT \* FROM acct\_user;

SELECT \* FROM user\_transaction;

SELECT \* FROM acct\_user WHERE user\_name= 'Brandon' AND password= 'b123';

DELETE FROM acct\_user WHERE balance = 0;

CREATE OR REPLACE PROCEDURE pop\_the\_trans

(u\_id IN number,

action IN varcahr2,

amount IN number,

generated\_id OUt number)

AS

BEGIN

INSERT INTO user\_transaction(action, amount, u\_id)

VALUES(action, amount, user\_id)

RETURNING trans\_id INTO generated\_id;

END pop\_tran;

DROP PROCEDURE pop\_the\_trans;

DROP TRIGGER pop\_trans;