Team : 70

Magazine

Management

System

TA : Marwa Helaly

فرح معتز محمد فؤاد عباس 2021170393 مايا محمد محمد حلمى الطيب 2021170439 بيشوى سدرة صابر سدرة 2021170130 مهند خالد على محمد 2021170558 مؤمن محمد احمد قدرى محمد 2021170418

نور الدين هشام محمد نور 2021170590

Department : IS

--------------------

------ TABLES ------

--------------------

-- Create USER table

CREATE TABLE USERS (

id NUMBER,

email VARCHAR2(255) UNIQUE,

username VARCHAR2(100),

password VARCHAR2(255),

role VARCHAR2(20) CHECK (role IN ('ADMIN', 'AUTHOR', 'READER')),

PRIMARY KEY (id)

);

-- Create CATEGORY table

CREATE TABLE CATEGORIES (

id NUMBER,

name VARCHAR2(100),

PRIMARY KEY (id)

);

-- Create MAGAZINE table

CREATE TABLE MAGAZINES (

id NUMBER,

name VARCHAR2(255),

admin\_id NUMBER,

publication\_date DATE,

PRIMARY KEY (id),

FOREIGN KEY (admin\_id) REFERENCES USERS(id)

);

-- Create ARTICLE table

CREATE TABLE ARTICLES (

id NUMBER,

title VARCHAR2(255),

magazine\_id NUMBER,

author\_id NUMBER,

category\_id NUMBER,

content VARCHAR2(2000),

status VARCHAR2(20) CHECK (status IN ('PENDING', 'APPROVED', 'REJECTED')),

PRIMARY KEY (id),

FOREIGN KEY (magazine\_id) REFERENCES MAGAZINES(id),

FOREIGN KEY (author\_id) REFERENCES USERS(id),

FOREIGN KEY (category\_id) REFERENCES CATEGORIES(id)

);

-- Create MAGAZINE\_ARTICLES table

CREATE TABLE MAGAZINE\_ARTICLES (

magazine\_id NUMBER,

article\_id NUMBER,

PRIMARY KEY (magazine\_id, article\_id),

FOREIGN KEY (magazine\_id) REFERENCES MAGAZINES(id),

FOREIGN KEY (article\_id) REFERENCES ARTICLES(id)

);

-- Create SUBSCRIPTION table

CREATE TABLE SUBSCRIPTIONS (

reader\_id NUMBER,

magazine\_id NUMBER,

start\_date DATE,

end\_date DATE,

price NUMBER,

PRIMARY KEY (reader\_id, magazine\_id),

FOREIGN KEY (reader\_id) REFERENCES USERS(id),

FOREIGN KEY (magazine\_id) REFERENCES MAGAZINES(id)

);

------------------------------

--------- SEQUENCES ----------

------------------------------

-- Create sequences

CREATE SEQUENCE user\_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE category\_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE article\_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE magazine\_seq START WITH 1 INCREMENT BY 1;

--------------------------

-------- TRIGGERS --------

--------------------------

-- Trigger for USERS table

CREATE OR REPLACE TRIGGER user\_trigger

BEFORE INSERT ON USERS

FOR EACH ROW

BEGIN

SELECT user\_seq.nextval INTO :new.id FROM dual;

END;

/

-- Trigger for CATEGORY table

CREATE OR REPLACE TRIGGER category\_trigger

BEFORE INSERT ON CATEGORIES

FOR EACH ROW

BEGIN

SELECT category\_seq.nextval INTO :new.id FROM dual;

END;

/

-- Trigger for ARTICLE table

CREATE OR REPLACE TRIGGER article\_trigger

BEFORE INSERT ON ARTICLES

FOR EACH ROW

BEGIN

SELECT article\_seq.nextval INTO :new.id FROM dual;

END;

/

-- Trigger for MAGAZINE table

CREATE OR REPLACE TRIGGER magazine\_trigger

BEFORE INSERT ON MAGAZINES

FOR EACH ROW

BEGIN

SELECT magazine\_seq.nextval INTO :new.id FROM dual;

END;

/

-----------------------------

---- MAGAZINE PROCEDURES ----

-----------------------------

-- Create procedure for inserting a new magazine

CREATE OR REPLACE PROCEDURE insert\_magazine(

name\_in IN VARCHAR2,

admin\_id\_in IN NUMBER,

publication\_date\_in IN DATE

) AS

BEGIN

INSERT INTO MAGAZINES (name, admin\_id, publication\_date)

VALUES (name\_in, admin\_id\_in, publication\_date\_in);

COMMIT;

END insert\_magazine;

/

-- Create procedure for updating magazine information

CREATE OR REPLACE PROCEDURE update\_magazine(

id\_in IN NUMBER,

name\_in IN VARCHAR2,

admin\_id\_in IN NUMBER

) AS

BEGIN

UPDATE MAGAZINES

SET name = name\_in,

admin\_id = admin\_id\_in

WHERE id = id\_in;

COMMIT;

END update\_magazine;

/

-- Create procedure for deleting a magazine

CREATE OR REPLACE PROCEDURE delete\_magazine(

id\_in IN NUMBER

) AS

BEGIN

DELETE FROM MAGAZINES

WHERE id = id\_in;

COMMIT;

END delete\_magazine;

/

-- Create procedure for retrieving magazine information

CREATE OR REPLACE PROCEDURE get\_magazine(

id\_in IN NUMBER,

magazine\_out OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN magazine\_out FOR

SELECT \* FROM MAGAZINES

WHERE id = id\_in;

END get\_magazine;

/

CREATE OR REPLACE PROCEDURE get\_all\_magazines(

magazine\_cursor OUT SYS\_REFCURSOR

)

AS

BEGIN

OPEN magazine\_cursor FOR

SELECT \*

FROM MAGAZINES;

END get\_all\_magazines;

/

CREATE OR REPLACE PROCEDURE get\_all\_other\_magazines (

p\_reader\_id IN NUMBER,

magazines\_cursor OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN magazines\_cursor FOR

SELECT ID, NAME, ADMIN\_ID, PUBLICATION\_DATE

FROM MAGAZINES

WHERE ID NOT IN (

SELECT MAGAZINE\_ID

FROM SUBSCRIPTIONS

WHERE READER\_ID = p\_reader\_id

);

END get\_all\_other\_magazines;

/

CREATE OR REPLACE PROCEDURE get\_all\_my\_magazines (

p\_reader\_id IN NUMBER,

magazines\_cursor OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN magazines\_cursor FOR

SELECT ID, NAME, ADMIN\_ID, PUBLICATION\_DATE

FROM MAGAZINES

WHERE ID IN (

SELECT MAGAZINE\_ID

FROM SUBSCRIPTIONS

WHERE READER\_ID = p\_reader\_id

);

END get\_all\_my\_magazines;

/

CREATE OR REPLACE PROCEDURE get\_all\_my\_magazines\_articles (

p\_reader\_id IN NUMBER,

articles\_cursor OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN articles\_cursor FOR

SELECT a.ID, a.TITLE, a.MAGAZINE\_ID, a.CONTENT

FROM ARTICLES a

WHERE a.MAGAZINE\_ID IN (

SELECT s.MAGAZINE\_ID

FROM SUBSCRIPTIONS s

WHERE s.READER\_ID = p\_reader\_id

) AND a.STATUS != 'PENDING'; -- corrected the condition here

END get\_all\_my\_magazines\_articles;

/

-----------------------------------

------- CATEGORY PROCEDURES -------

-----------------------------------

-- Create procedure for inserting a new category

CREATE OR REPLACE PROCEDURE insert\_category(

name\_in IN VARCHAR2

) AS

BEGIN

INSERT INTO CATEGORIES (id, name)

VALUES (category\_seq.nextval, name\_in);

COMMIT;

END insert\_category;

/

CREATE OR REPLACE PROCEDURE get\_all\_categories(

category\_cursor OUT SYS\_REFCURSOR

)

AS

BEGIN

OPEN category\_cursor FOR

SELECT Id, Name

FROM categories;

END get\_all\_categories;

/

-- Create procedure for updating category information

CREATE OR REPLACE PROCEDURE update\_category(

id\_in IN NUMBER,

name\_in IN VARCHAR2

) AS

BEGIN

UPDATE CATEGORIES

SET name = name\_in

WHERE id = id\_in;

COMMIT;

END update\_category;

/

-- Create procedure for deleting a category

CREATE OR REPLACE PROCEDURE delete\_category(

id\_in IN NUMBER

) AS

BEGIN

DELETE FROM CATEGORIES

WHERE id = id\_in;

COMMIT;

END delete\_category;

/

-- Create procedure for retrieving category information

CREATE OR REPLACE PROCEDURE get\_category(

id\_in IN NUMBER,

category\_out OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN category\_out FOR

SELECT \* FROM CATEGORIES

WHERE id = id\_in;

END get\_category;

/

-----------------------------------

----- SUBSCRIPTION PROCEDURES -----

-----------------------------------

-- Create procedure for inserting a new subscription

CREATE OR REPLACE PROCEDURE insert\_subscription(

reader\_id\_in IN NUMBER,

magazine\_id\_in IN NUMBER,

start\_date\_in IN DATE,

end\_date\_in IN DATE,

price\_in IN NUMBER

) AS

BEGIN

INSERT INTO SUBSCRIPTIONS (reader\_id, magazine\_id, start\_date, end\_date, price)

VALUES (reader\_id\_in, magazine\_id\_in, start\_date\_in, end\_date\_in, price\_in);

COMMIT;

END insert\_subscription;

/

-- Create procedure for updating subscription information

CREATE OR REPLACE PROCEDURE update\_subscription(

reader\_id\_in IN NUMBER,

magazine\_id\_in IN NUMBER,

start\_date\_in IN DATE,

end\_date\_in IN DATE,

price\_in IN NUMBER

) AS

BEGIN

UPDATE SUBSCRIPTIONS

SET start\_date = start\_date\_in,

end\_date = end\_date\_in,

price = price\_in

WHERE reader\_id = reader\_id\_in AND magazine\_id = magazine\_id\_in;

COMMIT;

END update\_subscription;

/

CREATE OR REPLACE PROCEDURE get\_subscribed\_magazines(

reader\_id\_in IN NUMBER,

magazine\_cursor OUT SYS\_REFCURSOR

)

AS

BEGIN

OPEN magazine\_cursor FOR

SELECT m.Id, m.Name, m.Admin\_Id, m.Publication\_Date

FROM Subscriptions s

JOIN Magazines m ON s.Magazine\_Id = m.Id

WHERE s.Reader\_Id = reader\_id\_in;

END get\_subscribed\_magazines;

/

CREATE OR REPLACE PROCEDURE get\_not\_subscribed\_magazines(

reader\_id\_in IN NUMBER,

magazine\_cursor OUT SYS\_REFCURSOR

)

AS

BEGIN

OPEN magazine\_cursor FOR

SELECT Id, Name, Admin\_Id, Publication\_Date

FROM Magazines

WHERE Id NOT IN (

SELECT Magazine\_Id

FROM Subscriptions

WHERE Reader\_Id = reader\_id\_in

);

END get\_not\_subscribed\_magazines;

/

CREATE OR REPLACE PROCEDURE get\_subscribed\_articles(

reader\_id\_in IN NUMBER,

article\_cursor OUT SYS\_REFCURSOR

)

AS

BEGIN

OPEN article\_cursor FOR

SELECT a.Id, a.Title, a.Content, a.Magazine\_Id, a.Author\_Id

FROM Articles a

JOIN Subscriptions s ON a.Magazine\_Id = s.Magazine\_Id

WHERE s.Reader\_Id = reader\_id\_in;

END get\_subscribed\_articles;

/

-- Create procedure for deleting a subscription

CREATE OR REPLACE PROCEDURE delete\_subscription(

reader\_id\_in IN NUMBER,

magazine\_id\_in IN NUMBER

) AS

BEGIN

DELETE FROM SUBSCRIPTIONS

WHERE reader\_id = reader\_id\_in AND magazine\_id = magazine\_id\_in;

COMMIT;

END delete\_subscription;

/

-- Create procedure for retrieving subscription information

CREATE OR REPLACE PROCEDURE get\_subscription(

reader\_id\_in IN NUMBER,

magazine\_id\_in IN NUMBER,

subscription\_out OUT SYS\_REFCURSOR

) AS

BEGIN

OPEN subscription\_out FOR

SELECT \* FROM SUBSCRIPTIONS

WHERE reader\_id = reader\_id\_in AND magazine\_id = magazine\_id\_in;

END get\_subscription;

/

-------------------------------

------- USER PROCEDURES -------

-------------------------------

-- Create procedure for inserting a new user

CREATE OR REPLACE PROCEDURE insert\_user(

email\_in IN VARCHAR2,

username\_in IN VARCHAR2,

password\_in IN VARCHAR2,

role\_in IN VARCHAR2

) AS

BEGIN

INSERT INTO USERS (email, username, password, role)

VALUES (email\_in, username\_in, password\_in, role\_in);

COMMIT;

END insert\_user;

/

-- Create procedure for updating user information

CREATE OR REPLACE PROCEDURE update\_user(

id\_in IN NUMBER,

email\_in IN VARCHAR2,

username\_in IN VARCHAR2,

password\_in IN VARCHAR2,

role\_in IN VARCHAR2

) AS

BEGIN

UPDATE USERS

SET email = email\_in,

username = username\_in,

password = password\_in,

role = role\_in

WHERE id = id\_in;

COMMIT;

END update\_user;

/

-- Create procedure for deleting a user

CREATE OR REPLACE PROCEDURE delete\_user(

id\_in IN NUMBER

) AS

BEGIN

DELETE FROM USERS

WHERE id = id\_in;

COMMIT;

END delete\_user;

/

-- Create procedure for retrieving user information

CREATE OR REPLACE PROCEDURE get\_user(

id\_in IN NUMBER,

username\_out OUT VARCHAR2,

email\_out OUT VARCHAR2,

password\_out OUT VARCHAR2,

role\_out OUT VARCHAR2

) AS

BEGIN

SELECT username, email, password, role

INTO username\_out, email\_out, password\_out, role\_out

FROM USERS

WHERE id = id\_in;

END get\_user;

/

/

CREATE OR REPLACE PROCEDURE get\_user\_by\_email\_password(

email\_in IN VARCHAR2,

password\_in IN VARCHAR2,

id\_out OUT NUMBER

) AS

user\_id NUMBER;

BEGIN

SELECT id INTO user\_id

FROM USERS

WHERE email = email\_in

AND password = password\_in;

id\_out := user\_id;

END get\_user\_by\_email\_password;

/