BÁO CÁO

Các công việc đã thực hiện cho bài Assignment 02 – DBI202

Tên: Nguyễn Hải Dương
Mã SV: FX16133

I. Các câu lệnh DDL để tạo thành phần cơ sở dữ liệu

Source code 1: ASM2-DDL1-DataBase

1. Tạo cơ sở dữ liệu

```
--A.TẠO DATABASE
CREATE DATABASE [dbfx16133]
```

2. Tạo các bảng cùng rằng buộc
   a. Bảng Article

```
--Bảng Article
CREATE TABLE Article (
  article_id VARCHAR(10) PRIMARY KEY,
  title VARCHAR(255),
  excerpt TEXT,
  content TEXT NOT NULL,
  pub_date DATE,
  status VARCHAR(20) NOT NULL,
  created_date DATE NOT NULL,
  approved_date DATE,
);
```

   b. Bảng Category

```
--Bảng Category
CREATE TABLE Category (
  category_id VARCHAR(10) PRIMARY KEY,
  cat_name VARCHAR(255) UNIQUE NOT NULL
);
```

   c. Bảng Account

```
CREATE TABLE Account (
  userid VARCHAR(10) PRIMARY KEY,
  username VARCHAR(255) UNIQUE NOT NULL,
  password VARCHAR(64) CHECK (LEN(password) >= 6) NOT NULL,
  email VARCHAR(255) UNIQUE NOT NULL
);
```

   d. Bổ sung thuộc tính author_id vào bảng article

```
--Thêm thuộc tính Author ID vào
ALTER TABLE Article
ADD Author_id VARCHAR(10);

ALTER TABLE Article
ADD CONSTRAINT FK_Author_Account
FOREIGN KEY (Author_id)REFERENCES ACCOUNT(userid)
```

e. Bảng Comment

```
--Bảng Comment
CREATE TABLE Comment (
  comment_id VARCHAR(10) NOT NULL PRIMARY KEY,
  comment_content TEXT NOT NULL,
  status VARCHAR(20) NOT NULL,
  comment_date DATE NOT NULL,
  article_id VARCHAR(10) NOT NULL,
  FOREIGN KEY (article_id) REFERENCES Article(article_id),
  userid VARCHAR(10) NOT NULL,
  FOREIGN KEY (userid) REFERENCES Account(userid)
);
```

f. Bảng reporter

```
--Bảng Reporter
CREATE TABLE Reporter (
  userid VARCHAR(10) NOT NULL,
  promote_date DATE NOT NULL,
  PRIMARY KEY (userid, promote_date),
  FOREIGN KEY (userid) REFERENCES Account(userid)
);
```

g. Bảng Editor

```
--Bảng Editor
CREATE TABLE Editor (
  userid VARCHAR(10) NOT NULL,
  promote_date DATE NOT NULL,
  PRIMARY KEY (userid, promote_date), --Không sử dụng UNIQUE
  FOREIGN KEY (userid) REFERENCES Account(userid)
);
```

h. Bảng ArticleCategory

```
--Bảng ArticleCategory
CREATE TABLE ArticleCategory (
  article_id VARCHAR(10) NOT NULL,
  category_id VARCHAR(10) NOT NULL,
  PRIMARY KEY (article_id, category_id),
  FOREIGN KEY (article_id) REFERENCES Article(article_id),
  FOREIGN KEY (category_id) REFERENCES Category(category_id)
);
```

3. Các câu lệnh thêm dữ liệu demo:

```
INSERT INTO Article(article_id, title, excerpt, content, pub_date, status, created_date, approved_date,author_id)
VALUES
    ('A001', 'Title 1', 'Excerpt 1', 'Content 1', '2022-01-01', 'published', '2022-01-01', '2022-01-02','U001'),
    ('A002', 'Title 2', 'Excerpt 2', 'Content 2', '2022-01-02', 'published', '2022-01-02', '2022-01-03','U001'),
    ('A003', 'Title 3', 'Excerpt 3', 'Content 3', '2022-01-03', 'draft', '2022-01-03', NULL,'U001'),
    ('A004', 'Title 4', 'Excerpt 4', 'Content 4', '2022-01-04', 'published', '2022-01-04', '2022-01-05','U001'),
    ('A005', 'Title 5', 'Excerpt 5', 'Content 5', '2022-01-05', 'published', '2022-01-05', '2022-01-06','U001'),
    ('A006', 'Title 6', 'Excerpt 6', 'Content 6', '2022-01-06', 'draft', '2022-01-06', NULL,'U001'),
    ('A007', 'Title 7', 'Excerpt 7', 'Content 7', '2022-01-07', 'published', '2022-01-07', '2022-01-08','U001'),
    ('A008', 'Title 8', 'Excerpt 8', 'Content 8', '2022-01-08', 'published', '2022-01-08', '2022-01-09','U001'),
    ('A009', 'Title 9', 'Excerpt 9', 'Content 9', '2022-01-09', 'published', '2022-01-09', '2022-01-10','U001'),
    ('A010', 'Title 10', 'Excerpt 10', 'Content 10', '2022-01-10', 'draft', '2022-01-10', NULL,'U001');

INSERT INTO Category(category_id, cat_name)
VALUES
    ('C001', 'Technology'),
    ('C002', 'Science'),
    ('C003', 'Politics'),
    ('C004', 'Sports'),
    ('C005', 'Entertainment'),
    ('C006', 'Health'),
    ('C007', 'Economics'),
    ('C008', 'Culture'),
    ('C009', 'Environment'),
    ('C010', 'Education')    ;
```

```
INSERT INTO ArticleCategory (article_id, category_id)
VALUES
  ('A001', 'C001'),
  ('A001', 'C002'),
  ('A002', 'C003'),
  ('A003', 'C002'),
  ('A003', 'C003'),
  ('A004', 'C001'),
  ('A005', 'C003'),
  ('A006', 'C001'),
  ('A007', 'C002'),
  ('A008', 'C003');

INSERT INTO Editor (userid, promote_date)
VALUES
    ('U001', '2022-02-01'),
    ('U002', '2022-02-01')  ,
    ('U003', '2022-02-01'),
    ('U004', '2022-03-01'),
    ('U005', '2022-04-01'),
    ('U006', '2022-05-01'),
    ('U007', '2022-06-01'),
    ('U008', '2022-07-01'),
    ('U009', '2022-08-01'),
    ('U010', '2022-09-01');

INSERT INTO Reporter (userid, promote_date)
VALUES
    ('U001', '2022-02-02'),
    ('U002', '2022-02-02'),
    ('U004', '2022-02-02'),
    ('U005', '2022-03-03'),
    ('U006', '2022-04-04'),
    ('U007', '2022-05-05'),
    ('U008', '2022-06-06'),
    ('U009', '2022-07-07'),
    ('U010', '2022-08-08'),
    ('U011', '2022-09-09');
```

```sql
INSERT INTO Comment (comment_id, comment_content, status, comment_date, article_id, userid)
VALUES
    ('C001', 'Comment1', 'approved', '2023-04-20', 'A001', 'U001'),
    ('C002', 'comment2', 'pending', '2023-04-21', 'A002', 'U002'),
    ('C003', 'Comment3', 'approved', '2023-04-22', 'A003', 'U003'),
    ('C004', 'Comment4', 'pending', '2023-04-23', 'A004', 'U004'),
    ('C005', 'Comment5', 'approved', '2023-04-23', 'A001', 'U005'),
    ('C006', 'Comment6', 'approved', '2023-04-22', 'A002', 'U006'),
    ('C007', 'Comment7', 'pending', '2023-04-20', 'A003', 'U007'),
    ('C008', 'Comment8', 'approved', '2023-04-19', 'A004', 'U008'),
    ('C009', 'Comment9', 'pending', '2023-04-18', 'A001', 'U009'),
    ('C010', 'Comment10', 'approved', '2023-04-24', 'A002', 'U010');
```

4. Tạo TRIGGER:
   Trong bảng article, khi thay đổi ‘status’ của một dòng từ ‘draft’ thành ‘published’ thì sẽ UPDATE ‘approved_date’ thành thời điểm thay đổi ‘status’

```sql
CREATE TRIGGER update_approved_date
ON Article
AFTER UPDATE
AS
BEGIN
  IF UPDATE(status) AND EXISTS (SELECT * FROM inserted WHERE status = 'published')
    BEGIN
        UPDATE Article SET approved_date = GETDATE() FROM Article a
        JOIN inserted i ON a.article_id = i.article_id WHERE i.status = 'published'
    END;
END;
```

5. Tạo STORED PROCEDURE:
   Tìm thông tin của bài viết (Aritcle_id, title và pub_date) dựa trên author_id.

```sql
-- STORED PROCEDURE --
-- Tạo stored procedure: Tìm thông tin article_id, title và pub_date theo author
CREATE PROCEDURE get_author_articles
@author_id VARCHAR(10)
AS
BEGIN
SELECT article_id, title, pub_date
FROM Article
WHERE Author_id = @author_id
ORDER BY pub_date DESC;
END

GO
EXEC get_author_articles @author_id = 'U001';
```

6. Tạo FUNCTION:
   Function đếm số lượng bài viết đã xuất bản trong một categery khi nhập category_id. Function dùng để thống kê số lượng của bài viết trong từng category, áp dụng cho việc đếm, hiển thị dữ liệu này trong giao diện.

```sql
-- Function đếm số lượng bài viết đã xuất bản trong category
CREATE FUNCTION dbo.CountPublishedArticlesByCategory (@category_id VARCHAR(10))
RETURNS INT
AS
BEGIN
  DECLARE @count INT;
  SELECT @count = COUNT(*)
  FROM ArticleCategory ac
  INNER JOIN Article a ON ac.article_id = a.article_id
  WHERE ac.category_id = @category_id AND a.status = 'published';
  RETURN @count
END

GO
-- Execute Function
SELECT dbo.CountPublishedArticlesByCategory('C001') as PublishedArticlesCount
```

7. Tạo INDEX

   Tạo index cho Article_id để các bài viết được sắp xếp theo thứ tự khi bản ghi được tạo hay chỉnh sửa

```sql
--Tạo Index --
GO
CREATE INDEX index_articleid ON Article (article_id);
```

8. Tạo TRANSACTION:
   a. Transaction 01 - Xuất bản bài viết:

      Cập nhật status từ ‘draft’ thành ‘published’ và đổi approved_date thành thời điểm hiện tại. Rollback khi status của bài viết không phải là draft hoặc không tồn tại.

```sql
-- Hàm transaction (1) - Publish Article
GO
BEGIN TRANSACTION;
-- Update the status of an article from 'draft' to 'published'
UPDATE Article
SET status = 'published', approved_date = GETDATE()
WHERE article_id = 'A003' AND status = 'draft';
-- Check if the update was successful
IF @@ROWCOUNT = 0
BEGIN
    --The article was not a draft or not exist: ROLLBACK
    ROLLBACK TRANSACTION;
    PRINT 'Update failed: The article was not a draft or not exist';
END
ELSE
BEGIN
    -- Successfully Update
    COMMIT TRANSACTION;
    PRINT 'Article published successfully.';
END
```

   b. Transaction 02 – Thêm comment mới vào bài viết

      Cập cập nhật comment mới vào một bài viết. Rollback khi bài viết chưa được xuất bản hoặc cập nhật các thông tin của comment không thành công.

```
GO
BEGIN TRANSACTION;
DECLARE @NewCommentID VARCHAR(10) = 'C010';
DECLARE @CommentContent VARCHAR(200) = 'This is a new comment';
DECLARE @ArticleID VARCHAR(10) = 'A001';
DECLARE @UserID VARCHAR(10) = 'U002'; -- User adding the comment
DECLARE @Status VARCHAR(20) = 'pending'; -- Initial status of the comment
DECLARE @CommentDate DATE = GETDATE(); -- Use current date for comment date

IF EXISTS (SELECT 1 FROM Article WHERE article_id = @ArticleID AND status = 'published') --Check if the article is publis
BEGIN
    INSERT INTO Comment (comment_id, comment_content, status, comment_date, article_id, userid)
    VALUES (@NewCommentID, @CommentContent, @Status, @CommentDate, @ArticleID, @UserID);
    IF @@ROWCOUNT = 0 --Check if the comment is published
    BEGIN
        ROLLBACK TRANSACTION;
        PRINT 'Insert failed: Unable to add comment.';
    END
    ELSE
    BEGIN
        COMMIT TRANSACTION;
        PRINT 'Comment added successfully.';
    END
END
ELSE
BEGIN
    ROLLBACK TRANSACTION;
    PRINT 'Insert failed: Article is not published or no exist.';
END
```

II. TRUY VẤN DỮ LIỆU

Source Code: ASM2-DDL2-Query

1. Truy vấn dữ liệu trên một bảng

```
-- Truy vấn dữ liệu trên một bảng
SELECT title, pub_date, status,created_date FROM Article
```

Truy vấn dữ liệu về thông tin của một bài viết

2. Truy vấn có sử dụng Order by

```
-- Truy vấn có sử dụng Order by
SELECT * FROM Comment
ORDER BY comment_date DESC, userid
```

Truy vấn dữ liệu của tất cả comment được sắp xếp từ ngày gần đây nhất.

3. Truy vấn sử dụng toán tử Like và các so sánh xâu ký tự.

```
-- Truy vấn sử dụng toán tử Like và các so sánh xâu ký tự.
SELECT * FROM Article
WHERE title LIKE '%1%'
```

Truy vấn những bài viết có số ‘1’ trong tiêu đề

4. Truy vấn liên quan tới điều kiện về thời gian

```sql
-- Truy vấn liên quan tới điều kiện về thời gian
SELECT * FROM Article
WHERE pub_date BETWEEN '2022-01-8' AND '2022-01-31';
```

Truy vấn những bài viết có ngày xuất bản trong khoảng thời gian.

5. Truy vấn dữ liệu từ nhiều bảng sử dụng Inner join

```sql
-- Truy vấn dữ liệu từ nhiều bảng sử dụng Inner join
SELECT Article.article_id, Article.title, Category.cat_name
FROM Article
INNER JOIN ArticleCategory ON Article.article_id = ArticleCategory.article_id
INNER JOIN Category ON Category.category_id = ArticleCategory.category_id;
```

Truy vấn dữ liệu về tiêu đề bài vết và category của bài viết đó dựa trên các bảng Article, Category, ArticleCategory

6. Truy vấn sử dụng Self join, Outer join.

```sql
-- Truy vấn sử dụng Self join, Outer join.
SELECT u1.username, u2.username AS referral_username
FROM Account u1
INNER JOIN Account u2 ON u1.userid = u2.userid;
```

Truy vấn những người được giới thiệu từ một người dùng khác.

7. Truy vấn sử dụng truy vấn con.

```sql
--Truy vấn sử dụng truy vấn con.
-- Tìm những người có trên 2 bài viết
SELECT DISTINCT Author_id
FROM Article
WHERE status = 'published' AND Author_id IN (
  SELECT Author_id
  FROM Article
  WHERE status = 'published'
  GROUP BY Author_id
  HAVING COUNT(*) >= 2
);
```

Truy vấn những người có trên 2 bài viết. Sử dụng truy vấn con cho bảng Article để đếm số bài viết của một người

8. Truy vấn sử dụng With.

```sql
-- Truy vấn sử dụng With.
-- Tổng số bài viết được xuất bản theo từng tháng
WITH PublishedArticles AS (
  SELECT YEAR(approved_date) AS PubYear, MONTH(approved_date) AS PubMonth, COUNT(*) AS TotalPublished
  FROM Article
  WHERE status = 'published'
  GROUP BY YEAR(approved_date), MONTH(approved_date)
)
SELECT PubYear, PubMonth, TotalPublished
FROM PublishedArticles
ORDER BY PubYear, PubMonth;
```

Dùng WITH tạo bảng PublishedArticles với thời gian xuất bản của từng Tháng để đếm tất cả số bài viết của tháng

9. Truy vấn thống kê sử dụng Group by và Having

```sql
-- Truy vấn thống kê sử dụng Group by và Having
-- Thống kê những người có trên 2 bài viết kèm theo số bài viết
SELECT author_id, COUNT(*) AS num_articles
FROM Article
WHERE status = 'published'
GROUP BY author_id
HAVING COUNT(*) >= 2;
```

10. Truy vấn sử dụng function (hàm) đã viết trong bước trước.

```sql
--Truy vấn sử dụng function (hàm) đã viết trong bước trước.
SELECT cat_name, dbo.CountPublishedArticlesByCategory(category_id) AS published_articles_count
FROM Category
```

Truy vấn Số bài viết trong từng Category sử dụng hàm đã viết ở phần trên