**BÁO CÁO**

TRIỂN KHAI VÀ THAO TÁC CƠ SỞ DỮ LIỆU

1. **Lược đồ cơ sở dữ liệu và tạo bảng**
   1. **Lược đồ**

****

* 1. **Tạo bảng**
     1. **Bảng Users (7 cột)**

Create Table Users (

UserID int identity (1,1) not null,

FullName nvarchar(55) not null,

Birthday date not null,

Phone char(10) not null,

Email varchar(255) not null,

Pw varchar(20) not null,

JoinedDay date not null Constraint DEF\_JoinedDay DEFAULT CURRENT\_TIMESTAMP,

Constraint PK\_Users PRIMARY KEY (UserID),

Constraint CK\_Phone CHECK (LEN(Phone) = 9),

Constraint UQ1\_Phone UNIQUE (Phone),

Constraint UQ2\_Email UNIQUE (Email)

);

* + 1. **Bảng Reposter (1 cột)**

Create Table Reposter (

ReposterID int,

Constraint PK\_Reposter PRIMARY KEY (ReposterID),

Constraint FK\_ReposterID FOREIGN KEY (ReposterID) references dbo.Users(UserID)

);

* + 1. **Bảng Editor (1 cột)**

Create Table Editor (

EditorID int,

Constraint PK\_Editor PRIMARY KEY (EditorID),

Constraint FK\_EditorID FOREIGN KEY (EditorID) references dbo.Users(UserID)

);

* + 1. **Bảng Tag (2 cột)**

Create Table Tag (

TagID int identity (1,1) not null,

TagName nvarchar(255) not null,

Constraint PK\_Tag PRIMARY KEY (TagID),

Constraint UQ\_TagName UNIQUE (TagName)

);

* + 1. **Bảng Category (3 cột)**

Create Table Category (

CategoryID char(4), -- C001, C002, C003

CategoryName nvarchar(255) not null,

SubCatOf char(4),

Constraint PK\_Category PRIMARY KEY (CategoryID),

Constraint UQ\_CategoryName UNIQUE (CategoryName),

Constraint FK\_SubCatOf FOREIGN KEY (SubCatOf) references dbo.Category(CategoryID)

);

* + 1. **Bảng Article (12 cột)**

Create Table Article (

ArticleNo int identity (1,1) not null,

Title nvarchar(50) not null,

Summary nvarchar(255) not null,

Content ntext not null,

CreatedTime datetime not null Constraint DEF\_CreatedTime DEFAULT CURRENT\_TIMESTAMP,

UpdatedTime datetime not null Constraint DEF\_UpdatedTime DEFAULT CURRENT\_TIMESTAMP,

PostedTime datetime null,

ArticleStatus varchar(10) not null Constraint DEF\_ArticleStatus DEFAULT 'Pending',

Visit int,

WrittenBy int not null,

ApprovedBy int not null,

CategoryID char(4) not null,

Constraint PK\_Article PRIMARY KEY (ArticleNo),

Constraint UQ\_Title UNIQUE (Title),

Constraint CK\_ArticleStatus CHECK (ArticleStatus IN ('Pending', 'Editing', 'Scheduled','Posted')),

Constraint FK\_WrittenBy FOREIGN KEY (WrittenBy) references dbo.Reposter(ReposterID),

Constraint FK\_ApprovedBy FOREIGN KEY (ApprovedBy) references dbo.Editor(EditorID),

Constraint FK\_CategoryID FOREIGN KEY (CategoryID) references dbo.Category(CategoryID)

);

* + 1. **Bảng Comment (3 cột)**

Create Table Comment (

ArticleNo int not null,

UserID int not null,

Detail ntext not null,

Constraint PK\_CMT PRIMARY KEY (ArticleNo, UserID),

Constraint FK\_CMT\_ArticleNo FOREIGN KEY (ArticleNo) references dbo.Article(ArticleNo),

Constraint FK\_CMT\_UserID FOREIGN KEY (UserID) references dbo.Users(UserID)

);

* + 1. **Bảng Article\_Tag (2 cột)**

Create Table Article\_Tag (

ArticleNo int not null,

TagID int not null,

Constraint PK\_AT PRIMARY KEY (ArticleNo, TagID),

Constraint FK\_AT\_ArticleNo FOREIGN KEY (ArticleNo) references dbo.Article(ArticleNo),

Constraint FK\_AT\_TagID FOREIGN KEY (TagID) references dbo.Tag(TagID)

);

1. **Chèn dữ liệu**

Dữ liệu đã được thêm vào Database nằm trong tệp đính kèm (Asm2\_Insert\_Into\_Table.sql)

1. **Tạo Trigger**
   1. **Trigger tự động cập nhật thời gian đăng bài (For Insert, Update)**

-- AritcleStatus được Update thành 'Posted' thì tự cập nhật PostedTime theo giờ hiện tại

Create Trigger PostedTimeAtuo ON dbo.Article

For Insert, Update

As

Begin

If (select ArticleStatus from inserted) = 'Posted'

Begin

Update dbo.Article

Set PostedTime = CURRENT\_TIMESTAMP

Where ArticleNo = (select ArticleNo from inserted)

End

End

Go

* 1. **Trigger tự động cập nhật thời gian Update**

-- Khi thực hiện update dữ liệu thuộc 3 cột Title, Summary, Content thì UpdatedTime cũng sẽ được tự động cập nhật

Create Trigger UpdatedTimeAtuo ON dbo.Article

For Update

As

Begin

If Update(Title) or Update(Summary) or Update(Content)

-- Hàm Update() để kiểm tra nội dung có được cập nhật hay không

Begin

Update dbo.Article

Set UpdatedTime = CURRENT\_TIMESTAMP

Where ArticleNo = (select ArticleNo from inserted)

End

End

Go

1. **Tạo Store Procedure**
   1. **Top 1 bài báo được quan tâm**

-- Bài báo có nhiều lượt truy cập và comment nhất, xếp theo thứ tự truy cập, sau đó đến comment

Create Proc Top\_Article

As

Begin

Select Top(1) A.ArticleNo, A.Visit, Count(\*) as CommentCounted

From dbo.Article as A

Join dbo.Comment as C ON A.ArticleNo = C.ArticleNo

Group by A.ArticleNo, A.Visit

Order by A.Visit DESC, Count(\*) DESC

End

Go

* 1. **Kiểm tra cấp bậc của Users**

-- Kiểm tra cấp bậc của một User (độc giả / phóng viên / viên tập viên) với đầu vào là UserID

-- Nếu UserID trùng với ReposterID, User đó dưới vai trò Phóng Viên

-- Nếu UserID trùng với EditorID, User đó dưới vai trò Biên Tập Viên

-- Ngược lại, User chỉ là Độc Giả bình thường

Create Proc Check\_Level (@UserID int)

As

Begin

If @UserID in (select \* from dbo.Reposter)

Select UserID, FullName, N'Phóng Viên' as Position From dbo.Users

Where UserID = @UserID

Else if @UserID in (select \* from dbo.Editor)

Select UserID, FullName, N'Biên Tập Viên' as Position From dbo.Users

Where UserID = @UserID

Else

Select UserID, FullName, N'Độc giả' as Position From dbo.Users

Where UserID = @UserID

End

Go

1. **Tạo Function**
   1. **Kiểm tra Users (Multi Statement Table Function)**

-- Kiểm tra một User có phải là độc giả / phóng viên / viên tập viên

-- Nếu là độc giả thì hiển thị thêm số lần comment, phóng viên thì thêm số bài báo đã đăng, biên tập viên thì thêm số bài báo đã duyệt

Create Function Check\_User (@UserID int)

Returns @tblCheckUser Table(

UserID int,

FullName nvarchar(55),

Position nvarchar(55),

Counted int

)

As

Begin

If @UserID in (select \* from dbo.Reposter)

Begin

Insert Into @tblCheckUser (UserID, FullName, Position, Counted)

Select UserID, FullName, N'Phóng Viên' as Position,

(Select count(\*) from Article where WrittenBy = @UserID) as Counted

From Users

Where UserID = @UserID

End

Else if @UserID in (select \* from dbo.Editor)

Begin

Insert Into @tblCheckUser (UserID, FullName, Position, Counted)

Select UserID, FullName, N'Biên Tập Viên' as Position,

(Select count(\*) from Article where ApprovedBy = @UserID) as Counted

From Users

Where UserID = @UserID

End

Else

Begin

Insert Into @tblCheckUser (UserID, FullName, Position, Counted)

Select UserID, FullName, N'Độc Giả' as Position,

(Select count(\*) from Comment where UserID = @UserID) as Counted

From Users

Where UserID = @UserID

End

Return

End

Go

* 1. **Lọc bài báo theo Category (Inline Table Function)**

-- Lọc và hiển thị nội dung tất cả các bài báo nằm trong 1 Category

Create Function Category\_Filter (@CategoryID char(4))

Returns Table As Return (

Select \* from Article

Where CategoryID = @CategoryID

)

Go

* 1. **Lọc bài báo theo Tag (Inline Table Function)**

-- Lọc và hiển thị nội dung tất cả các bài báo nằm trong 1 Tag

Create Function Tag\_Filter (@TagID int)

Returns Table As Return (

Select A.\*, AT.TagID

From Article A

Join Article\_Tag AT on A.ArticleNo = AT.ArticleNo

Where TagID = @TagID

)

Go

1. **Tạo Index**
   1. **Index có sẵn**

Các Index được SQL SERVER tạo sẵn khi ta tạo Primary Key và các cột Unique

Bảng Article: Clustered Index (UserID), Non Clustered Index (Phone, Email)

Bảng Reposter: Clustered Index (ReposterID)

Bảng Editor: Clustered Index (EditorID)

Bảng Tag: Clustered Index (TagID), Non Clustered Index (TagName)

Bảng Category: Clustered Index (CategoryID), Non Clustered Index (CategoryName)

Bảng Article: Clustered Index (ArticleNo), Non Clustered Index (Title)

Bảng Comment: Clustered Index (ArticleNo, UserID)

Bảng Article\_Tag: Clustered Index (ArticleNo, TagID)

* 1. **Index tạo thêm**

-- Ngoài các Index được tạo sẵn khi tạo Primary Key và các cột Unique

-- Ta tạo thêm 3 Non-Clustered Index với các trường WrittenBy, ApprovedBy và CategoryID

-- Các Index này sẽ giúp tăng tốc độ, hiệu suất khi ta dùng lệnh truy vấn thông qua ReposterID, EditorID và CategoryID

CREATE NONCLUSTERED INDEX non\_idx\_ReposterID ON Article (WrittenBy);

CREATE NONCLUSTERED INDEX non\_idx\_EditorID ON Article (ApprovedBy);

CREATE NONCLUSTERED INDEX non\_idx\_CategoryID ON Article (CategoryID);

Go

1. **Tạo Transaction**

-- Tìm và xoá các phóng viên chưa có bài viết nào được đăng

Begin Tran

Select \* from Reposter;

With BadReposter as (

Select R.ReposterID as ReposterID

From Reposter as R

Left Join Article As A ON R.ReposterID = A.WrittenBy

WHERE A.ArticleNo IS NULL

)

Delete Reposter

From Reposter as R

Join BadReposter as BR ON R.ReposterID = BR.ReposterID;

Select \* from Reposter;

Rollback Tran

Go

1. **Tạo các truy vấn (bao gồm cả mức độ đơn giản lẫn phức tạp)**
   1. **Truy vấn dữ liệu trên một bảng**

-- Hiển thị thông tin của tất cả các User có trong CSDL

Select \*

From dbo.Users;

Go

* 1. **Truy vấn có sử dụng Order by**

-- Sắp xếp thứ tự Comment theo Mã bài báo tăng dần, và UserID giảm dần

Select \*

From dbo.Comment

Order By ArticleNo ASC, UserID DESC;

Go

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

-- Tìm User có họ bắt đầu bằng chữ T

Select \*

From dbo.Users

Where FullName Like N'T%'

Go

-- Tìm tên của User ngắn nhất và cho biết có bao nhiêu ký tự

Select Top(1) FullName, Len(FullName) as NameLength From dbo.Users

Order By Len(FullName) ASC

Go

* 1. **Truy vấn liên quan tới điều kiện về thời gian**

-- Tính số lượng Users đã tham gia từ năm 2020 đến nay

Select Count(\*) as Total\_Users\_From\_2020

From Users

Where JoinedDay Between '20200101' and getdate()

Go

-- Tính tuổi của Users và xếp theo thứ tự tuổi giảm dần

-- Không thể dùng getdate() trừ trực tiếp Birthday để tính tuổi vì chúng khác kiểu dữ liệu

-- Do đó ta phải dùng thêm hàm Year()

Select UserID, FullName, Birthday, (Year(getdate()) - Year(Birthday)) as YearOld

From Users

Order By (Year(getdate()) - Year(Birthday)) DESC

Go

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

-- Hiển thị thông tin của tất cả User là Editor

Select U.\*

From Users as U

Inner Join Editor E ON U.UserID = E.EditorID

Go

-- Tìm ID, họ tên, tiêu đề bài báo của các Reporter đăng bài ngày 29/05/2023

-- Bảng Reposter không có thông tin họ tên nên ta cần Join thêm bảng thứ 3 là Users

Select R.ReposterID, U.FullName, A.Title

From Article as A

Inner Join Reposter R ON A.WrittenBy = R.ReposterID

Inner Join Users U ON R.ReposterID = U.UserID

Where PostedTime = '20230529'

Go

* 1. **Truy vấn sử dụng Self join, Outer join.**

-- Self Join các Users có cùng năm sinh

Select U1.UserID, U1.FullName, U2.UserID, U2.FullName, Year(U1.Birthday) as U1\_Year, Year(U2.Birthday) as U2\_Year

From Users U1

Join Users U2

-- Năm sinh trùng nhau nhưng không được trùng ID

ON U1.UserID <> U2.UserID and Year(U1.Birthday) = Year(U2.Birthday)

-- Thêm điều kiện Where để tránh bị lặp lại hàng

Where U1.UserID < U2.UserID

Order By U1.UserID, U2.UserID

Go

-- Outer Join, tìm các độc giả chưa có bất kỳ comment nào (các độc giả này không phải là Phóng viên hoặc Biên tập viên)

Select U.UserID, U.FullName, Detail

From Users U

Left Outer Join Comment C ON U.UserID = C.UserID

Where Detail is null

And U.UserID not in (Select \* From Reposter)

And U.UserID not in (Select \* From Editor)

Go

* 1. **Truy vấn sử dụng truy vấn con.**

-- Hiển thị toàn bộ nội dung của bài báo có nhiều lượt xem nhất

Select \*

From Article

Where Visit = (Select Max(Visit) From Article)

Go

-- Hiển thị ID, họ tên của các Phóng Viên viết bài chưa được duyệt (Pending)

Select \*

From (

Select UserID, FullName, Phone, ArticleNo, ArticleStatus

From Users U

Join Article A ON U.UserID = A.WrittenBy

) as ReporterInfo

Where ArticleStatus = 'Pending'

Go

* 1. **Truy vấn sử dụng With**

-- Hiển thị tên Tag và số lượng các bài báo được gắn với mỗi Tag đó

With GroupByTag as (

Select Distinct TagID, count(A.ArticleNo) Over (Partition By TagID) as CountedArticle

From Article A

Join Article\_Tag T ON A.ArticleNo = T.ArticleNo

)

Select T.TagID, T.TagName, Case When CountedArticle IS NULL Then 0 Else CountedArticle End as CountedArticle

From Tag T

Left Join GroupByTag GT ON T.TagID = GT.TagID

Order By T.TagID

Go

* 1. **Truy vấn thống kê sử dụng Group by và Having**

-- Đếm số lượng các bài báo có trong thẻ Kinh Doanh

Select C.CategoryID, C.CategoryName, Count(ArticleNo) as CountedArticle

From Article A

Join Category C ON A.CategoryID = C.CategoryID

Group By C.CategoryID, C.CategoryName

Having C.CategoryName = N'Kinh Doanh'

Go

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

-- Dựa theo Function Check\_User được tạo, hãy lập ra 1 bảng hiển thị tên của tất cả User kèm vị trí của họ (User, Reposter, Editor)

Declare @UserID int = 1

Declare @CheckTable Table (

UserID int,

FullName nvarchar(255),

Position nvarchar(55),

-- Giá trị trong cột Counted được nhóm theo vị trí Position

-- Là số lượng cmt (đối với User), số lượng bài báo đã đăng (đối với Reposter) hoặc số lượng bài báo đã duyệt (đối với Editor)

Counted int

)

-- Khi giá trị @UserID vượt quá giá trị lớn nhất của UserID trong bảng Users thì thoát vòng lặp

While @UserID <= (Select Max(UserID) From Users)

Begin

Insert Into @CheckTable

Select UserID, FullName, Position, Counted From dbo.Check\_User(@UserID)

Set @UserID = @UserID + 1

End

Select \* From @CheckTable

Order By Position, UserID

Go