Lab 06: Views

※\(^o^)/※

September, 2019

Contents

Introduction

This lab aims to help students get used to working with views in T-SQL.

Lab Activities

Create a simple view

Drop the view if it already exists

```
16 USE AccountPayables;
17 IF OBJECT_ID('VendorGeneral') IS NOT NULL
```

```
    DROP VIEW OutstandingInvoices;
GO
```

Create the view

```
CREATE VIEW VendorsGeneral
AS
SELECT VendorID, VendorName, VendorPhone
FROM Vendors;
```

Test the view

```
SELECT * FROM VendorsGeneral;
```

Create a view from two tables

```
USE AccountPayables;
IF OBJECT_ID('VendorInvoices') IS NOT NULL
    DROP VIEW OutstandingInvoices;
GO

CREATE VIEW VendorInvoices
AS
SELECT VendorName, InvoiceNumber, InvoiceDate, InvoiceTotal
FROM Vendors JOIN Invoices ON Vendors.VendorID = Invoices.VendorID;
```

Create a view and rename all columns

```
USE AccountPayables;
IF OBJECT_ID('OutstandingInvoices') IS NOT NULL
    DROP VIEW OutstandingInvoices;
GO

CREATE VIEW OutstandingInvoices
    (InvoiceNumber, InvoiceDate, InvoiceTotal, BalanceDue)
AS
SELECT InvoiceNumber, InvoiceDate, InvoiceTotal,
    InvoiceTotal - PaymentTotal - CreditTotal
FROM Invoices
WHERE InvoiceTotal - PaymentTotal - CreditTotal > 0;
```

Create a view and rename some columns

```
USE AccountPayables;
IF OBJECT_ID('OutstandingInvoices') IS NOT NULL
```

```
    DROP VIEW OutstandingInvoices;
GO

CREATE VIEW OutstandingInvoices
AS
SELECT InvoiceNumber, InvoiceDate, InvoiceTotal,
    InvoiceTotal - PaymentTotal - CreditTotal AS BalanceDue
FROM Invoices
WHERE InvoiceTotal - PaymentTotal - CreditTotal > 0;
```

Create a view with schema_binding

```
USE AccountPayables;
IF OBJECT_ID('VendorsDue') IS NOT NULL
    DROP VIEW OutstandingInvoices;
GO

CREATE VIEW VendorsDue
WITH SCHEMABINDING
AS
SELECT InvoiceDate AS Date, VendorName AS Name,
    VendorContactFName + ' ' + VendorContactLName AS Contact,
    InvoiceNumber AS Invoice,
    InvoiceTotal - PaymentTotal - CreditTotal AS BalanceDue
FROM dbo.Vendors JOIN dbo.Invoices
    ON Vendors.VendorID = Invoices.VendorID
WHERE InvoiceTotal - PaymentTotal - CreditTotal > 0;
```

Create an updatable view

```
USE AccountPayables;
IF OBJECT_ID('VendorPayment') IS NOT NULL
    DROP VIEW VendorPayment;
GO

CREATE VIEW VendorPayment
AS
SELECT VendorName, InvoiceNumber, InvoiceDate, PaymentDate,
    InvoiceTotal, CreditTotal, PaymentTotal
FROM Invoices JOIN Vendors ON Invoices.VendorID = Vendors.VendorID
WHERE InvoiceTotal - PaymentTotal - CreditTotal > 0;

SELECT *
FROM VendorPayment;

UPDATE VendorPayment
```

```
126 SET PaymentTotal = 19351.18, PaymentDate = '2016-04-02'
127 WHERE VendorName = 'Malloy Lithographing Inc' AND InvoiceNumber = 'P-0608';
```

Exercises

Exercise 01

Create a view named InvoiceBasic that returns VendorName, InvoiceNumber, and InvoiceTotal.
Write a select statement that returns all columns in the view, sorted by VendorName, where the first letter of the vendor names is N, O, or P.

Exercise 02

Create a view named Top10PaidInvoices that returns three columns for each vendor: VendorName, LastInvoice (the most recent invoice date), and SumOfInvoices (the sum of the InvoiceTotal column). Return only the 10 vendors with the largest SumOfInvoices and include only paid invoices (InvoiceTotal - PaymentTotal - CreditTotal = 0).

Exercise 03

Create an updatable view named VendorAddress that returns the VendorID, both address columns, and the city, state, zip code columns for each vendor.
Write a selecte statement to examine the result set where VendorID=4.
Write an update statement that changes the address so that the suite number (Ste 260) is stored in VendorAddress2 rather than in VendorAddress1.
Rerun the select query to verify the changes.

Exercise 04

Write a query to count the number of foreign keys in the database.

Exercise 05

Use the view designer to modify the InvoiceBasic view created in exercise 01 to sort the result set by VendorName.
What clause does the system automatically generate to allow the use of order by in the view?