

Лабораторная работа № 10

Представления в БД

1.1 Создание представлений

Представление – это сохранение результата SQL-запроса, с помощью которых можно осуществлять доступ к данным таблиц, для которых был создан запрос.

```
CREATE VIEW [ < database_name > . ] [ < owner > . ] view_name [ ( column [ ,...n ] ) ]  
[ WITH [ ENCRYPTION ] | [, SCHEMABINDING] | [, VIEW_METADATA] ]  
AS  
select_выражение [ WITH CHECK OPTION ]
```

1.1.1 Создание представлений в SQL Server Management Studio

1. Раскройте узел Вашей БД и щелкните правой кнопкой мыши на пункте **Представления (Views)**. Выберите строку **Создать представление (New Views)**.

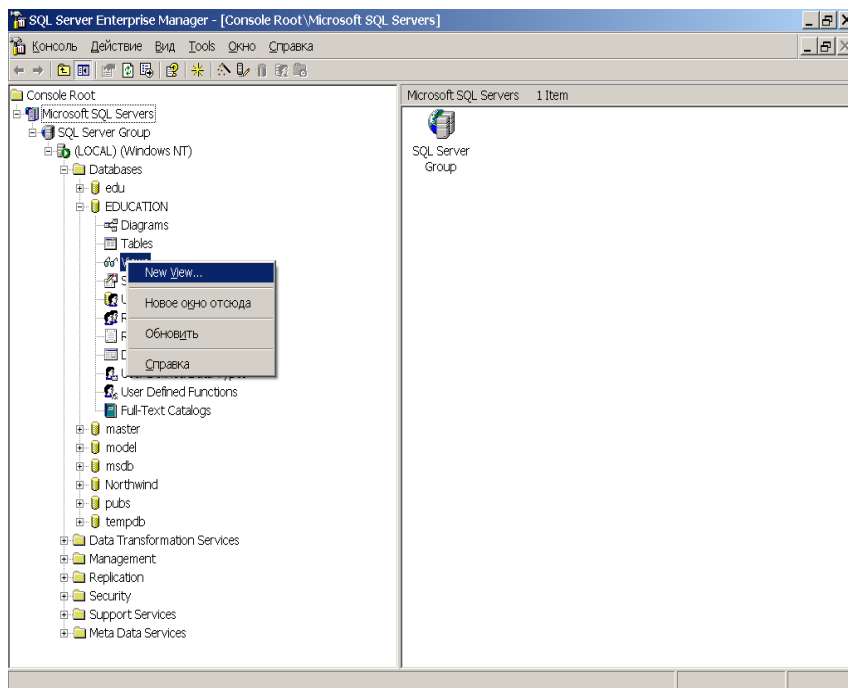


Рис. 7.1 Создание представлений в SQL Server Management Studio

2. Добавим в окно диаграмм требуемые таблицы.

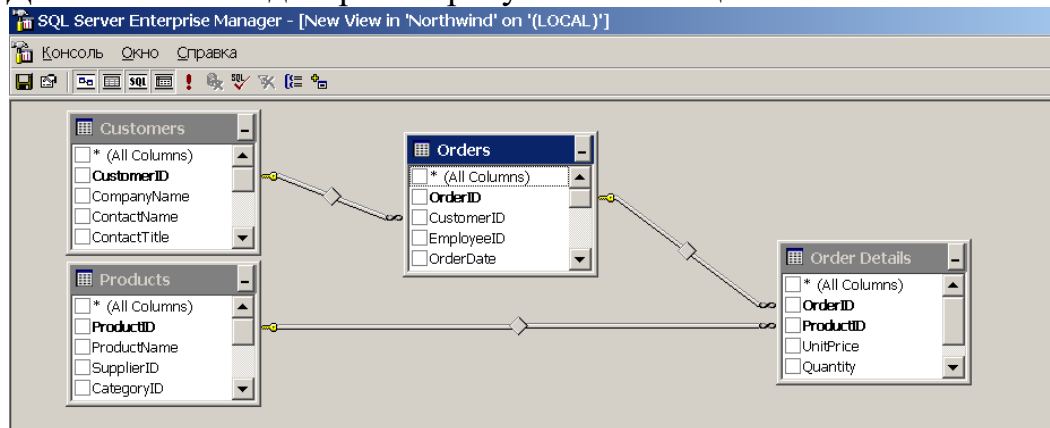


Рис. 7.2 Добавление таблиц в представление

3. Сформируем представление. При необходимости вручную добавляем вычисляемый столбец **ExtendedPrice**, введя выражение в столбец **Column** и псевдоним (название) в столбец **Alias**.

The diagram shows the same four tables: Customers, Products, Orders, and Order Details. The 'Column' tab is selected, showing the 'ExtendedPrice' column being added to the 'Order Details' table. The 'Table' tab is also selected, showing the 'Order Details' table. The 'Column' tab also shows the 'ExtendedPrice' column being added to the 'Order Details' table.

Column	Table	Output	Sort
OrderDate	Orders	✓	
Quantity	[Order Details]	✓	
ProductName	Products	✓	
dbo.[Order Details].Quantity * dbo.[Order Details].UnitPrice		✓	

```
SELECT dbo.Customers.CompanyName, dbo.[Order Details].ProductID, dbo.[Order Details].UnitPrice, dbo.Orders.OrderDate, dbo.[Order Details].Quantity,
       dbo.Products.ProductName, dbo.[Order Details].Quantity * dbo.[Order Details].UnitPrice AS ExtendedPrice
FROM   dbo.Customers INNER JOIN
       dbo.Orders ON dbo.Customers.CustomerID = dbo.Orders.CustomerID INNER JOIN
       dbo.[Order Details] ON dbo.Orders.OrderID = dbo.[Order Details].OrderID INNER JOIN
       dbo.Products ON dbo.[Order Details].ProductID = dbo.Products.ProductID
```

Рис. 7.3 Редактирование структуры представления

4. Сохраните созданное представление под именем **CustomersOrders_vw** и закройте окно мастера.

Задание на самостоятельную работу.

1. Создайте модифицируемое представление, выводящее данные о студентах имеющих отличные оценки (представление создать **только на базе таблицы USP**) Предусмотреть невозможность модификации таблицы USP командой *INSERT INTO имя_представления VALUES (код_оценки, 3415,4)* через созданное представление.

Примечание: Используйте при создании представления опцию WITH CHECK OPTION. При размещении ее в определении представления все команды модификации будут подвергаться проверке. Таким образом, можно регулировать процесс ввода значений, которые пользователь впоследствии сам не в состоянии корректировать. Опция WITH CHECK OPTION не поддерживает каскадного изменения данных. Поэтому она применяется только в представлениях, которые основаны на базовых таблицах, а не на других представлениях.

2. Модифицируйте представление с целью последующего создания для него индексов. Создайте индекс по полям UNUM и UDATE. Просмотрите информацию о созданном индексе, используя хранимую процедуру **sp_helpindex** <имя_представления>.

Дайте объяснения, что изменилось с точки зрения СУБД при ведении данного представления после создания индекса.

3. Создайте представление, выводящее список оценок по математике за вчерашний день. (Для проверки добавьте подобную запись в БД.)

Примечание: Для реализации задания, при работе с датами необходимо воспользоваться следующими функциями

GETDATE () – возвращает текущую дату и время.

```
SELECT GETDATE ()
GO
-----
July 29 1998    2:50    PM
```

DATEADD (date_part, number, date) – возвращает значение, которое равно указанной в параметре **date** дате плюс интервал *number*. Параметр *date_part* принимает значения `day| month| year`.

Например, увеличим на 21 день каждую дату `pubdate` в таблице `titles`.

```
SELECT DATEADD (day, 21, pubdate) AS timeframe
FROM titles
timeframe
-----
Jul 3 1991 12:00AM
Jun 30 1991 12:00AM
.....
Jul 3 1991 12:00AM
Nov 11 1991 12:00AM
Jul 3 1991 12:00AM
Jul 3 1991 12:00AM
```

CONVERT (datatype[length],expression,[,style]) — явное преобразование типов.

Применение **CONVERT()** обусловлено тем, что возвращаемый функцией **GETDATE()** результат включает также текущее время суток. Поэтому необходимо преобразовать возвращаемую функцией **GETDATE()** дату, отбросив текущее время перед сравнением (например, в **VARCHAR(12)**). Параметр **style** принимает значения:

Without century (yy)	With century (yyyy)	Standard	Input/Output**
-	0 or 100 (*)	Default	mon dd yyyy hh:miAM (or PM)
1	101	USA	mm/dd/yy
2	102	ANSI	yy.mm.dd
3	103	British/French	dd/mm/yy
4	104	German	dd.mm.yy
5	105	Italian	dd-mm-yy
6	106	-	dd mon yy
7	107	-	Mon dd, yy
8	108	-	hh:mm:ss
-	9 or 109 (*)	Default + milliseconds	mon dd yyyy hh:mi:ss:mmmAM (or PM)
10	110	USA	mm-dd-yy
11	111	JAPAN	yy/mm/dd
12	112	ISO	yymmdd
-	13 or 113 (*)	Europe default +	dd mon yyyy

		milliseconds	hh:mm:ss:mmm(24h)
14	114	-	hh:mi:ss:mmm(24h)
-	20 or 120 (*)	ODBC canonical	yyyy-mm-dd hh:mi:ss(24h)
-	21 or 121 (*)	ODBC canonical (with milliseconds)	yyyy-mm-dd hh:mi:ss:mmm(24h)
-	126(***)	ISO8601	yyyy-mm-dd Thh:mm:ss:mmm(no spaces)
-	130*	Kuwaiti	dd mon yyyy hh:mi:ss:mmmAM
-	131*	Kuwaiti	dd/mm/yy hh:mi:ss:mmmAM