Úkoly na procvičení jazyka T-SQL

Lehké

Úkol 1

Zadání A

Vytvořte seznam všech zákazníků, seřazený vzestupně podle jména a druhotně podle příjmení. Vraťte všechny sloupce tabulky.

```
SELECT * FROM Person.Contact
ORDER BY FirstName, LastName
```

Zadání B

Vytvořte seznam všech zákazníků, seřazený vzestupně podle jména a druhotně podle příjmení. Vraťte jen sloupce Title, FirstName, LastName, Suffix a ContactID.

```
SELECT ContactID, Title, FirstName, LastName, Suffix
FROM Person.Contact
ORDER BY FirstName, LastName
```

Zadání C

Vytvořte seznam všech zákazníků kteří mají titul před jménem i za jménem, seřazený **sestupně** podle jména a druhotně podle příjmení. Vraťte jen sloupce Title, FirstName, LastName, Suffix a ContactID.

```
SELECT ContactID, Title, FirstName, LastName, Suffix
FROM Person.Contact
WHERE Title IS NOT NULL AND Suffix IS NOT NULL
ORDER BY FirstName DESC, LastName DESC
```

Zadání D

Vytvořte seznam všech zákazníků jejichž titul před jménem začíná na písmeno „M“, seřazený **sestupně** podle jména a druhotně **vzestupně** podle příjmení. Vraťte jen sloupce Title, FirstName, LastName, Suffix a ContactID.

```
SELECT ContactID, Title, FirstName, LastName, Suffix
FROM Person.Contact
WHERE Title LIKE 'M%'
ORDER BY FirstName DESC, LastName
```

Zadání E

Vytvořte seznam zákazníků, který bude obsahovat všechny zákazníky, ale vraťte jen sloupce ContactID a PersonName, kde sloupec PersonName bude dopočítaný do následující podoby:
“ Ing. Petr Novák, PhD.“ ⇔ {Title} {FirstName} {LastName}, {Suffix}

Využijte funkce **COALESCE** pro ošetření nullových hodnot.

```
SELECT ContactID,
      COALESCE(Title+' ','') + FirstName + ' '
      + LastName + COALESCE(', ' + Suffix, '') AS PersonName
FROM Person.Contact
```

Definice tabulek

Person.Contact

Úkol 2

Zadání A

Vytvořte seznam produktů které bylo možno koupit od 1. 1. 2000 do 1. 1. 2001.

```
SELECT * FROM Production.Product
WHERE SellStartDate <= '2000/1/1' AND
      (SellEndDate >= '2001/1/1' OR SellEndDate IS NULL)
```

Zadání B

Vytvořte parametrizovanou uloženou proceduru **Production.uspGetAvailableProducts**, která umožní vracet seznam prodávaných produktů v daných mezích dat.

```
CREATE PROCEDURE Production.uspGetAvailableProducts
      @SellFrom DATETIME, @SellTo DATETIME
AS
BEGIN
      SELECT * FROM Production.Product
      WHERE SellStartDate <= @SellFrom AND
            (SellEndDate >= @SellTo OR SellEndDate IS NULL)
END
```

Zadání C

Vytvořte pohled **Production.vAvailableProducts** který bude vracet produkty, které se v současné chvíli prodávají. Využijte **GETDATE** pro získání aktuálního data a času.

```
CREATE VIEW Production.vAvailableProducts
AS

SELECT * FROM Production.Product
WHERE SellStartDate <= GETDATE() AND
      (SellEndDate >= GETDATE() OR SellEndDate IS NULL)
```

Zadání D

Určete pro každý produkt celkovou délku jak dlouho je na trhu, počítáno ve dnech. Využijte funkce **DATEDIFF**. Nezapomeňte ošetřit situaci, kdy produkt je na trhu až do současnosti např. využitím **CASE**.

```
SELECT ProductID, Name,
      CASE
            WHEN SellEndDate IS NULL THEN DATEDIFF(d, SellStartDate, GETDATE())
            ELSE DATEDIFF(d, SellStartDate, SellEndDate)
      END AtMarketPeriod
FROM Production.Product
```

Definice tabulek

Production.Product

Středně těžké

Úkol 3

Zadání A

Určete, která 3 jména jsou mezi zákazníky nejčastěji používána. Vypište také počet jejich užití.

```
SELECT TOP 3 FirstName, COUNT(*) AS UsedTimes
FROM Person.Contact
GROUP BY FirstName
ORDER BY COUNT(*) DESC
```

Zadání B

Určete, která 3 mužská jména jsou mezi zákazníky nejčetnější. Vypište také počet jejich užití.

```
SELECT TOP 3 FirstName, COUNT(*) AS UsedTimes
FROM Person.Contact
WHERE Title = 'Mr.'
GROUP BY FirstName
ORDER BY COUNT(*) DESC
```

Zadání C

Vytvořte uloženou proceduru **Person.uspGetFirstNameUsage** , která bude po zadání jména vracet počet jeho výskytů a relitivní počet jeho výstytů vzhledem k celkovému počtu jmen (v %, zaokrouhleno na 4 desetinná místa).

```
CREATE PROCEDURE Person.uspGetFirstNameUsage
      @FirstName nvarchar(50)
AS
BEGIN
      SELECT TOP 3 FirstName, COUNT(*) AS UsedTimes,
            ROUND(CAST(COUNT(*) AS FLOAT)/(SELECT COUNT(FirstName) FROM
Person.Contact)*100,4) AS [Relative]
      FROM Person.Contact
      WHERE FirstName = @FirstName
      GROUP BY FirstName
END
```

Zadání D

Určete počet různých jmen, která byla u zákazníků použita.

```
SELECT COUNT(DISTINCT FirstName) AS UsedNames
FROM Person.Contact
```

Definice tabulek

Person.Contact

Úkol 4

Zadání A

Určete, kterých 10 zákazníků naposledy nakupovalo. Využijte propojení tabulek Sales.SalesOrderHeader, Sales.Individual a Person.Contact. Vypište jen základní údaje o zákazníkovi.

```
SELECT TOP 10 c.* FROM Person.Contact c
      INNER JOIN Sales.SalesOrderHeader soh ON c.ContactID = soh.ContactID
ORDER BY soh.OrderDate DESC
```

Zadání B

Vytvořte pohled **Sales.vBestCustomers**, který bude zobrazovat 10 nejvíce nakupujících zákazníků (uvažujme podle ceny objednávky).

```
CREATE VIEW Sales.vBestCustomers
AS
SELECT TOP 10 c.ContactID, c.FirstName, c.LastName, SUM(TotalDue) AS TotalDue
FROM Person.Contact c
      INNER JOIN Sales.SalesOrderHeader soh ON c.ContactID = soh.ContactID
GROUP BY c.ContactID, c.FirstName, c.LastName
ORDER BY SUM(TotalDue) DESC
```

Zadání C

Vytvořte pohled **Sales.vMostActiveCustomers**, který bude zobrazovat 10 nejvíce nakupujících zákazníků (uvažujme podle počtu objednávek).

```
CREATE VIEW Sales.vMostActiveCustomers
AS
SELECT TOP 10 c.ContactID, c.FirstName, c.LastName, COUNT(*) AS TotalOrders
FROM Person.Contact c
      INNER JOIN Sales.SalesOrderHeader soh ON c.ContactID = soh.ContactID
GROUP BY c.ContactID, c.FirstName, c.LastName
ORDER BY COUNT(*) DESC
```

Definice tabulek

Sales.SalesOrderHeader, Sales.Individual a Person.Contact

Těžké

Úkol 5

Zadání A

Tabulka Purchasing.PurchaseOrderHeader obsahuje informace o nákupech zboží, které je potom přeprodáváno. Spočítejte průměrnou nákupní cenu za kus pro jednotlivé produkty, které byly nakoupeny. Využijte dat z tabulky Purchasing.PurchaseOrderDetail.

```
SELECT p.Name, AVG(pod.UnitPrice) AS AverageUnitPrice
FROM Purchasing.PurchaseOrderDetail pod
      INNER JOIN Production.Product p ON pod.ProductID = p.ProductID
GROUP BY pod.ProductID, p.Name
```

Zadání B

Pomocí tabulky Production.Product určete, které produkty jsou vlastní výroby (neexistuje pro ně záznam v Purchasing.PurchaseOrderDetail). Využijte v podmínce **EXISTS**.

```
SELECT *
FROM Production.Product p
WHERE NOT EXISTS
      (SELECT * FROM Purchasing.PurchaseOrderDetail pod
            WHERE pod.ProductID = p.ProductID)
```

Zadání C

Pomocí tabulky Production.Product určete, které produkty **nejsou** vlastní výroby (existuje pro ně záznam v Purchasing.PurchaseOrderDetail). Využijte v podmínce **EXISTS**.

```
SELECT *
FROM Production.Product p
WHERE EXISTS
      (SELECT * FROM Purchasing.PurchaseOrderDetail pod
            WHERE pod.ProductID = p.ProductID)
```

Zadání D

Pro každý nakoupený produkt spočítejte průměrnou nákupní cenu (Purchasing.PurchaseOrderDetail).

```
SELECT p.Name, AVG(pod.UnitPrice) AS AverageUnitPrice
FROM Purchasing.PurchaseOrderDetail pod
      INNER JOIN Production.Product p ON pod.ProductID = p.ProductID
GROUP BY pod.ProductID, p.Name
```

Zadání E

Vytvořte pohled **Purchasing.vPurchasedProducts**, který bude obsahovat sloupce ProductID a vypočítaný sloupec UnitPrice, který bude obsahovat průměrnou nákupní cenu.

```
CREATE VIEW Purchasing.vPurchasedProducts
AS
SELECT ProductID, AVG(UnitPrice) AS AverageUnitPrice
FROM Purchasing.PurchaseOrderDetail
GROUP BY ProductID
```

Zadání F

Určete pro každou objednávku ze Sales.SalesOrderHeader, kolik se prodělalo nebo vydělalo na přeprodání nakoupených dílů.

```
SELECT soh.SalesOrderID, SUM((sod.UnitPrice-AverageUnitPrice)*sod.OrderQty) AS
Incomes
FROM Sales.SalesOrderHeader soh
      INNER JOIN Sales.SalesOrderDetail sod ON soh.SalesOrderID =
sod.SalesOrderID
      INNER JOIN Purchasing.vPurchasedProducts pp ON sod.ProductID =
pp.ProductID
GROUP BY soh.SalesOrderID
```

Zadání G

Upravte pohled Purchasing.vPurchasedProducts tak, aby byl rozšířen o jeden sloupec, kde budou pomocí Sales.SalesOrderDetail vypočítány zisky nebo ztráty na prodeji daného produktu.

```
ALTER VIEW Purchasing.vPurchasedProducts
AS
SELECT ProductID, AVG(pod.UnitPrice) AS AverageUnitPrice,
      (SELECT SUM((sod.UnitPrice-pod2.AverageUnitPrice)*sod.OrderQty)
            FROM Sales.SalesOrderDetail sod
                  INNER JOIN (SELECT ProductID, AVG(pod.UnitPrice) AS
AverageUnitPrice
                        FROM Purchasing.PurchaseOrderDetail pod
                        GROUP BY ProductID) pod2 ON pod2.ProductID =
sod.ProductID
            WHERE sod.ProductID = pod.ProductID) AS Incomes
FROM Purchasing.PurchaseOrderDetail pod
GROUP BY ProductID
ORDER BY ProductID
```

Zadání H

Vytvořte uloženou proceduru, která má parametr ProductID a vrací informace o ziscích či ztrátách na prodeji nakupovaného produktu.

```
CREATE PROCEDURE Production.uspProductIncomes
      @ProductID int
AS
BEGIN
      SELECT ProductID, AVG(pod.UnitPrice) AS AverageUnitPrice,
            (SELECT SUM((sod.UnitPrice-pod2.AverageUnitPrice)*sod.OrderQty)
                  FROM Sales.SalesOrderDetail sod
                        INNER JOIN (SELECT ProductID,
                                    AVG(pod.UnitPrice) AS AverageUnitPrice
                              FROM Purchasing.PurchaseOrderDetail pod
                              GROUP BY ProductID) pod2
                                    ON pod2.ProductID = sod.ProductID
                  WHERE sod.ProductID = pod.ProductID) AS Incomes
      FROM Purchasing.PurchaseOrderDetail pod
      WHERE pod.ProductID = @ProductID
      GROUP BY ProductID
      ORDER BY ProductID
END
```

Definice tabulek

Purchasing.PurchaseOrderDetail, Sales.SalesOrderHeader, Sales.SalesOrderDetail, Production.Product

Úkol 6

Zadání A

Určete kolik objednávek vystavili jednotliví obchodníci v Sales.SalesPerson

```
SELECT sp.SalesPersonID, COUNT(*) AS OrderCount
FROM Sales.SalesPerson sp
      INNER JOIN Sales.SalesOrderHeader soh ON sp.SalesPersonID =
soh.SalesPersonID
GROUP BY sp.SalesPersonID
```

Zadání B

Určete celkovou cenu zboží co prodali jednotliví obchodníci, pro zjednodušení počítejte se sloupcem TotalDue v rámci Sales.SalesOrderHeader.

```
SELECT sp.SalesPersonID, SUM(soh.TotalDue) AS TotalSales
FROM Sales.SalesPerson sp
      INNER JOIN Sales.SalesOrderHeader soh ON sp.SalesPersonID =
soh.SalesPersonID
GROUP BY sp.SalesPersonID
```

Zadání C

Určete nejlepšího prodejce pro každou oblast, ve výsledku vypište ID prodejce, název oblasti a celkovou sumu prodejů.

```
CREATE FUNCTION Sales.udfGetBestTerritorySalesPerson
(
      @TerritoryID int
)
RETURNS TABLE
AS
RETURN
(
      SELECT TOP 1 sp.SalesPersonID, SUM(soh.TotalDue) AS SalesPersonTotalDue
      FROM Sales.SalesPerson sp
            INNER JOIN Sales.SalesOrderHeader soh ON sp.SalesPersonID =
soh.SalesPersonID AND sp.TerritoryID = soh.TerritoryID
      WHERE soh.TerritoryID = @TerritoryID
      GROUP BY sp.SalesPersonID
      ORDER BY SUM(soh.TotalDue) DESC
)
GO

SELECT st.TerritoryID, ts.*
FROM Sales.SalesTerritory st
      OUTER APPLY Sales.udfGetBestTerritorySalesPerson(st.TerritoryID) AS ts
ORDER BY TerritoryID
```

Zadání D

Každý obchodník má stanovený limit, kolik zboží musí prodat za rok, aby dostal k výplatě bonusy. Protože se tento limit mění s časem, tak je uchováván v závislosti na roce v tabulce Sales.PersonQuotaHistory.

Vytvořte uloženou proceduru **Sales.uspRewardedSalesPersons**, která po vložení roku (jako číslo) vypíše obchodníky, kteří v daném roce získali svůj bonus.

```
CREATE PROCEDURE Sales.uspRewardedSalesPersons
	@Year int
AS
BEGIN
	SELECT *
	FROM (
		SELECT spqh.SalesPersonID,
			(SELECT SUM(TotalDue) AS RealSales
				FROM Sales.SalesOrderHeader soh
				WHERE soh.SalesPersonID = spqh.SalesPersonID
					AND soh.OrderDate > spqh.QuotaDate
					AND soh.OrderDate < DATEADD(m,3,spqh.QuotaDate)
			) AS RealSales,
			spqh.SalesQuota
	FROM Sales.SalesPersonQuotaHistory spqh
	WHERE spqh.QuotaDate >= CAST(CAST(@Year AS nvarchar(4))+'/1/1' AS DATE)
		AND spqh.QuotaDate <= CAST (CAST(@Year AS nvarchar(4))+'/12/31' AS
DATE)
	) AS sales
	WHERE sales.RealSales >= sales.SalesQuota
END
```

Definice tabulek

Sales.SalesPerson, Sales.SalesPersonQuotaHistory, Sales.SalesOrderHeader, Sales.Territory