# Datenbanken 2 – Praktikum 2

*Stored Procedures und Funktionen, Übungs-DB erstellen*

## Vorbemerkungen

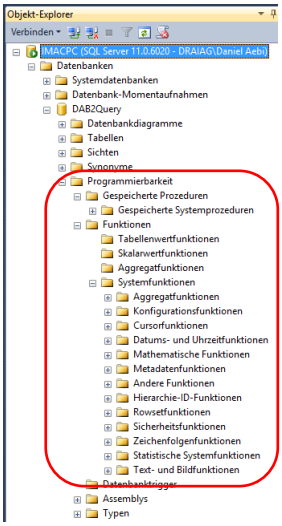
Gespeicherte Prozeduren sind Routinen, die in der Datenbank aufbewahrt werden und Code kapseln. In SQL Server sind die folgenden Arten von gespeicherten Prozeduren zulässig:

- Gespeicherte T-SQL-Prozeduren, die in T-SQL geschrieben sind.
- Gespeicherte CLR-Prozeduren, die als Microsoft .NET-Assemblys in der Datenbank gespeichert werden (im Praktikum nicht behandelt).
- Erweiterte gespeicherte Prozeduren, die extern kompilierte DLLs aufrufen (im Praktikum nicht behandelt).

Benutzerdefinierte Funktionen sind T-SQL- oder CLR-Routinen, die Parameter entgegennehmen und Skalarwerte oder Tabellen zurückgeben.

Ein Trigger ist eine besondere Art von gespeicherter Prozedur, die mit ausgewählten Ereignissen in einer Tabelle oder Sicht verknüpft wird. Immer, wenn dieses Ereignis auftritt, wird der Trigger ausgelöst und sein Code ausgeführt. Eine explizite Ausführung ist nicht möglich.

Es gibt in jeder SQL Server-Datenbank eine Vielzahl von bereits vorhandenen gespeicherten Prozeduren bzw. Funktionen. Diese findet man im Objekt-Explorer bei den einzelnen Datenbanken unter der Rubrik „Programmierbarkeit“:




Viele dieser Prozeduren dienen der Administration des Servers oder der Abfrage von Metadaten. Es gibt aber auch sehr viele nützliche Funktionen, die in eigenen Queries verwendet werden können.

Wir arbeiten im Praktikum 2 mit der Datenbank DAB2Query. Nutzen Sie für die Aufgaben auch das Dokument `SQL Server 2012 Transact-SQL DDL Reference.pdf`.

## Grundlagen von gespeicherten Prozeduren

Gespeicherte T-SQL-Prozeduren bestehen aus T-SQL-Code und weisen folgende wichtige Eigenschaften auf:

- Sie werden in T-SQL-Code mit dem Befehl EXECUTE aufgerufen.
- Über Eingabeparameter können Sie Daten an gespeicherte Prozeduren übergeben, und mit Ausgabeparametern können Sie Daten von ihnen empfangen.
- Gespeicherte Prozeduren können an die Clientanwendung die Resultsets von Abfragen zurückgeben.
- Gespeicherte Prozeduren können Daten in Tabellen bearbeiten sowie Tabellen und Indizes erstellen, ändern und löschen.

## Aufgabe 1: Abfrage ohne Parameter

Formulieren Sie in SQL folgende Abfrage: Gesucht sind alle Angaben zu Bestellungen (Tabelle: `Sales.Orders`) des Kunden mit der ID 37 im 2. Quartal des Jahres 2014.

## Aufgabe 2: Abfrage parametrieren

Die Abfrage von Aufgabe 1 hat den Nachteil, dass sie nur gerade für diesen einen Fall taugt. Besser wäre es, wenn die Abfrage „parametrisiert“ werden könnte. Recherchieren Sie, wie in T-SQL-Skripten Variablen definiert werden und schreiben Sie die Abfrage von Aufgabe 1 so um, dass Variabeln benutzt werden.

## Aufgabe 3: Abfrage als gespeicherte Prozedur

Auch die Lösung mit Variablen ist noch recht „sperrig“. Besser ist eine Prozedur, die mit entsprechenden Parametern (Kundennummer, BestelldatumVon, BestelldatumBis) aufgerufen werden kann, also z.B. so:

```
EXECUTE Sales.GetOrders 37, '2014-04-01', '2014-07-01'
```

Recherchieren Sie, wie man gespeicherte Prozeduren erstellt und schreiben Sie eine solche mit dem Namen `Sales.GetOrders`, die so wie oben gezeigt, aufgerufen werden kann.

## Aufgabe 4: Fibonacci-Zahlen

Schreiben Sie eine gespeicherte Prozedur `dbo.Fibonacci`, die die n-te Fibonacci-Zahl ausgibt (n soll als Parameter angegeben werden). Die Fibonacci-Folge ist wie folgt definiert:

$$\text{Fibonacci}(n) = \begin{cases} n \leq 2: 1 \\ sonst: f(n-1) + f(n-2) \end{cases}$$

Schreiben Sie die Prozedur in einer iterativen Variante. Sie brauchen dazu eine Bedingung, eine Schleife sowie die Anweisung PRINT.

## Grundlagen von Funktionen

Wie gespeicherte Prozeduren können auch benutzerdefinierte Funktionen Parameter annehmen, die dann innerhalb der Funktion als Variablen zugänglich sind. Im Gegensatz zu gespeicherten Prozeduren werden benutzerdefinierte Funktionen jedoch in T-SQL-Anweisungen eingebettet und als Teil von ihnen ausgeführt. Benutzerdefinierte Funktionen können auf SQL Server-Daten zugreifen, aber keine DDL-Anweisungen ausführen. Auch die Änderung von Daten in permanenten Tabellen mithilfe von DML-Anweisungen ist nicht möglich. Es gibt zwei Hauptarten von benutzerdefinierten Funktionen, skalare und tabellenwertige. Die skalaren Funktionen geben einen einzelnen Wert an den Aufrufer zurück, die tabellenwertigen dagegen eine Tabelle. Beide Varianten können sowohl aus einer einzelnen als auch aus mehreren Zeilen T-SQL-Code bestehen und auch Kontrollflussanweisungen enthalten.

## Aufgabe 5: Skalarwertige Funktion

Schreiben Sie eine Funktion `Sales.Value`, die das Produkt aus einem Preis und einer Menge berechnet. Setzen Sie die Funktion dann ein, um in der Tabelle `Sales.OrderDetails` die Attribute `orderID`, `unitprice` und `qty` sowie das Produkt aus `unitprice` und `qty` derjenigen Tupel auszugeben, bei denen dieses Produkt grösser als 10000 ist.

## Aufgabe 6: Tabellenwertige Funktion

Schreiben Sie eine Funktion `Sales.QtyRange`, die als Resultat eine Tabelle zurückgibt mit denjenigen Tupeln der Tabelle `Sales.OrderDetails`, deren Wert für qty in einem Bereich qtylow und qtyhigh liegt (das sind die beiden Parameter für die Funktion). Setzen Sie die Funktion dann ein, um diejenigen Tupel der Tabelle `Sales.OrderDetails` zurückzugeben, deren Wert für `qty` zwischen 100 und 200 liegt.

## Aufgabe 7: Fibonacci-Zahlen rekursiv

Schreiben Sie eine Funktion `dbo.FibonacciRec`, die die n-te Fibonacci-Zahl rekursiv berechnet und zurückgibt.

## Aufgabe 8: Performance-Datenbank erzeugen

Für künftige Optimierungsüberlegungen und Leistungsmessungen brauchen wir eine „grosse“ Datenbank. Die Datenbank DAB2Query enthält zu wenig Tupel. Zu diesem Zweck erzeugen wir eine Testdatenbank `DAB2Perf` und „generieren“ darin Testdaten. Dafür verwenden wir eine Funktion `dbo.GetNums`, die mit Hilfe von mehrfacher „Produktbildung“ eine Tabelle erzeugt, die eine Sequenz von Ganzzahlen liefert. Laden Sie dazu das Skript `S_Praktikum_02.sql`. Scrollen Sie durch das SQL-Skript und versuchen Sie es zu verstehen. Beschreiben Sie dann in eigenen Worten, was das Skript macht. Führen Sie das Skript aus (Abfrage → Ausführen oder F5, das dauert etwas!) und schauen Sie anschliessend die entstandenen Tabellen, ihre Attribute mit ihren Datentypen und die Schlüssel bzw. Indexe im Objekt-Explorer genauer an.