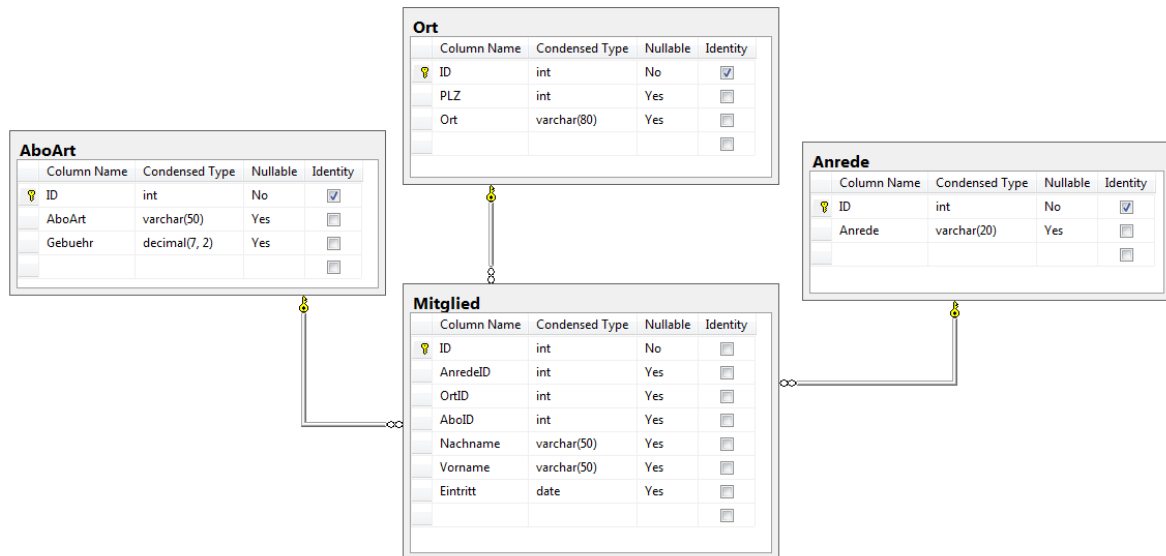


Aufgaben zu XML Datenbanken (Lösungen)

Datenmodell



1. Abfragen

A1.1

Schreiben Sie eine SQL Abfrage, die ID, Nachname, Vorname und Eintritt aus der Tabelle *Mitglied* ausliest und alles in einem XML-Dokument mit dem Wurzelement *MitarbeiterList* ausgibt.

```

SELECT [ID]
      ,[Nachname]
      ,[Vorname]
      ,[Eintritt]
FROM [Mitglied]
FOR XML AUTO, ROOT('MitarbeiterList')
  
```

A1.2

Schreiben Sie eine SQL Abfrage, die pro Anrede (Herr, Frau) die Mitglieder im XML Format mit Wurzelement *MitarbeiterList* ausliest. Beachten Sie, dass die XML Spalte für die Hyperlinks Anzeige mit der Convert Funktion (*convert(xml, spalte)*) konvertiert werden muss.

Beispielausgabe:

| | ID | Anrede | MitarbeiterListXML |
|---|----|--------|-----------------------------------------------------------------------------------------------------|
| 1 | 1 | Frau | <Gruppe><m ID="44" Nachname="Bürgin" Vorname="Sandra" Eintritt="1989-05-01" /><m ID="77" Nachna... |
| 2 | 2 | Herr | <Gruppe><m ID="33" Nachname="Balmelli" Vorname="Marco" Eintritt="1990-01-01" /><m ID="55" Nachna... |

```

select *,
convert(xml, (SELECT [ID]
      ,[Nachname]
      ,[Vorname]
      ,[Eintritt]
FROM [Mitglied] m
where m.AnredeID = a.ID
for xml auto, root('MitarbeiterList'))
) as MitarbeiterListXML
from Anrede a
  
```

A1.3

Schreiben Sie eine SQL Abfrage, die ID, Nachname, Vorname und Eintritt aus der Tabelle *Mitglied* ausliest und alles in einem JSON-Dokument mit dem Wurzelement *MitarbeiterList* ausgibt.

```
SELECT [ID]
      ,[Nachname]
      ,[Vorname]
      ,[Eintritt]
FROM [Mitglied]
FOR JSON AUTO, ROOT('MitarbeiterList')
```

A1.4


Schreiben Sie eine SQL Abfrage, die pro Aboart (Student, Jahresabo etc.) die Mitglieder als JSON Dokument mit Wurzelement *MitarbeiterList* ausliest.

```
select *,
convert(xml, (SELECT [ID]
      ,[Nachname]
      ,[Vorname]
      ,[Eintritt]
FROM [Mitglied] m
where m.AboID = a.ID
for json auto, root('MitarbeiterListJSON'))
) as MitarbeiterListJSON
from AboArt a
```

2. XML Datentyp / Funktionen

A2.1

Schreiben Sie den CREATE TABLE Anweisung um die untenstehende Tabelle in Ihrer IBZ Datenbank anzulegen.

| tblOrders | | | |
|-----------------------------------------------------------------------------------|--------------|----------------|----------|
| | Column Name | Condensed Type | Nullable |
|  | OrderID | int | No |
| | OrderDate | datetime | Yes |
| | OrderNumber | nvarchar(25) | Yes |
| | CustomerID | int | Yes |
| | OrderDetails | xml | Yes |

Bemerkung: Verwenden Sie für das OrderDetails Attribut den XML Datentyp.

CREATE TABLE tblOrders

```
(
    OrderID int PRIMARY KEY,
    OrderDate datetime,
    OrderNumber nvarchar(25),
    CustomerID int,
    OrderDetails xml
)
```

Go

A2.2

Fügen Sie mit INSERT INTO folgenden Datensatz zur tblOrders Tabelle hinzu.

| OrderID | OrderDate | OrderNumber | CustomerID | OrderDetails |
|---------|-------------------------|-------------|------------|---------------------------------------------------------------------------------------------------------------|
| 43659 | 2001-07-01 00:00:00.000 | SO43659 | 676 | <Root><OrderDetail OrderDetailID="1" OrderID="43659" Quantity="1" ProductID="776" UnitPrice="2024.9940" />... |

Vollständiger Dateninhalt von OrderDetails:

```
<Root>
  <OrderDetail OrderDetailID="1" OrderID="43659" Quantity="1" ProductID="776" UnitPrice="2024.9940" />
  <OrderDetail OrderDetailID="2" OrderID="43659" Quantity="3" ProductID="777" UnitPrice="2024.9940" />
  <OrderDetail OrderDetailID="3" OrderID="43659" Quantity="1" ProductID="778" UnitPrice="2024.9940" />
</Root>
```

INSERT INTO tblOrders (OrderID, OrderDate, OrderNumber, CustomerID, OrderDetails)

VALUES (43659, '2001-07-01 00:00:00.000', 'SO43659', 676,

```
N'<Root>
  <OrderDetail OrderDetailID="1" OrderID="43659" Quantity="1" ProductID="776" UnitPrice="2024.9940" />
  <OrderDetail OrderDetailID="2" OrderID="43659" Quantity="3" ProductID="777" UnitPrice="2024.9940" />
  <OrderDetail OrderDetailID="3" OrderID="43659" Quantity="1" ProductID="778" UnitPrice="2024.9940" />
</Root>')
```

GO

A2.3

Schreiben Sie eine Abfrage welche mit der **XPath** Query Methode die Bestellzeile zum Produkt 776 listet.

```
SELECT OrderId, OrderDetails.query('/Root/OrderDetail[@ProductID = 776]') AS OrderDetails
FROM tblOrders
```

A2.4

Schreiben Sie eine Abfrage welche mit der **XQuery** Query Methode die Bestellzeile zum Produkt 776 listet.

```
SELECT OrderDetails.query('
    for $i in /Root/OrderDetail
    where /Root/OrderDetail[@ProductID = 776]
    return $i
') AS OrderDetails
FROM tblOrders
```

A2.5

Schreiben Sie eine Abfrage welche mit der **Exists** Methode die Bestellzeile zum Produkt 776 listet.

```
SELECT OrderDetails
FROM tblOrders
WHERE OrderDetails.exist('/Root/OrderDetail[@ProductID = 776]') = 1
```

A2.6

Listen Sie mit einer SELECT Abfrage die ProductID der ersten Bestellzeile (**Root/OrderDetail/@ProductID** [1]). Verwenden Sie hierzu für den Daten Zugriff die **value** Methode.

```
SELECT OrderDetails.value('/Root/OrderDetail/@ProductID[1]', 'int') AS ProductID
FROM tblOrders
WHERE OrderDetails.exist('/Root/OrderDetail[@OrderDetailID = 1]') = 1
```