



SRH Hochschule  
Heidelberg

Prof. Alfred Moos  
SRH Hochschule  
Heidelberg  
Ludwig-Guttman-Str. 6

# **Modellgetriebene Publikation von XML-Dokumenten aus relationalen Daten**

# Modellgetriebene Publikation von XML- Dokumenten aus relationalen Daten

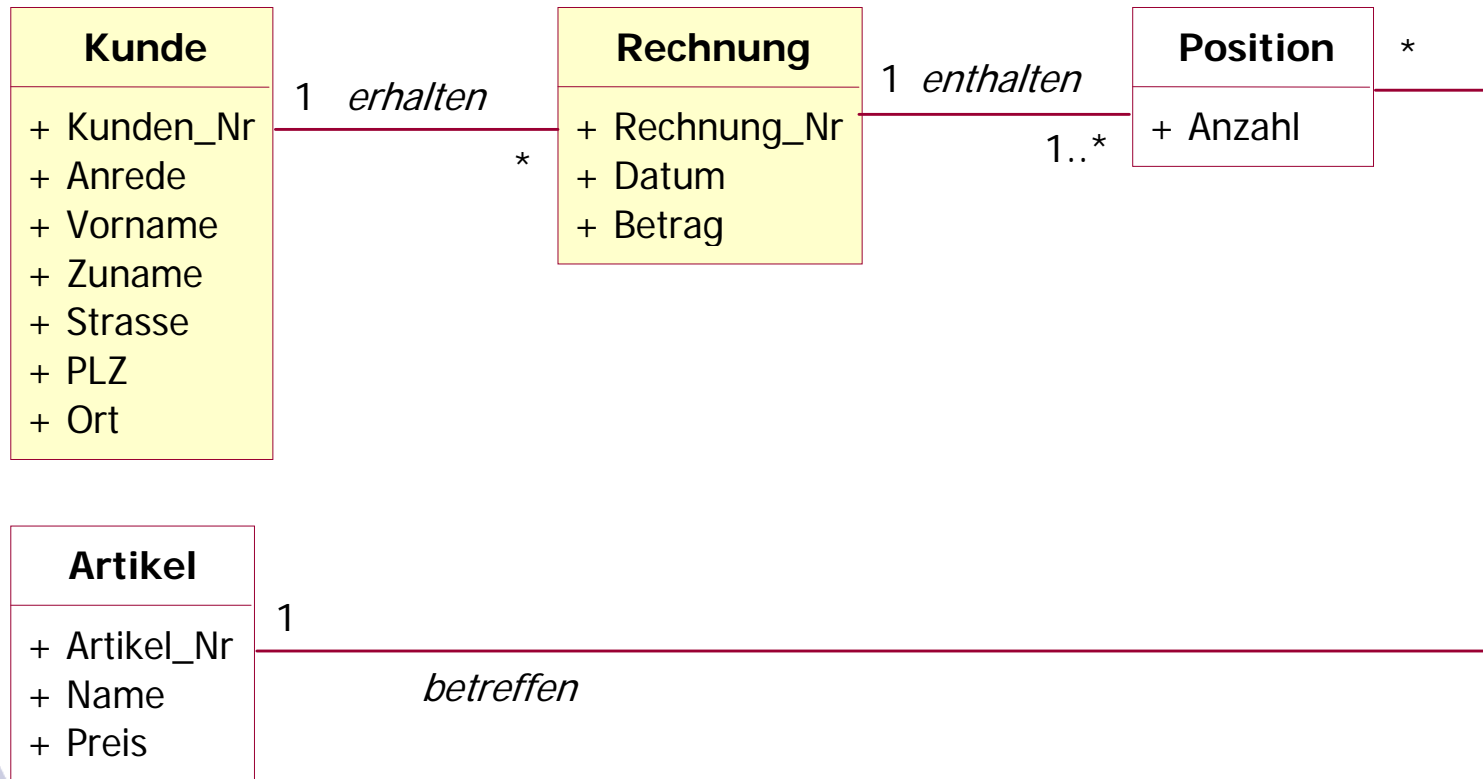
## UML - SQL/XML - DB2 V9.1

- **Das relationale Datenmodell wird in UML formuliert**
- **Das hierarchische XML-Schemamodell wird in UML formuliert**
- **Die Strukturvorgaben aus dem XML-Schemamodell werden mit Hilfe der SQL/XML-Publikationsfunktionen implementiert**
- **Bei der Programmierung wird auf Programmästhetik größter Wert gelegt**
- **Das Programm wird von DB2 V9.1 ausgeführt**

# Das relationale Datenmodell



Einfaches ERM Rechnungsschreibung

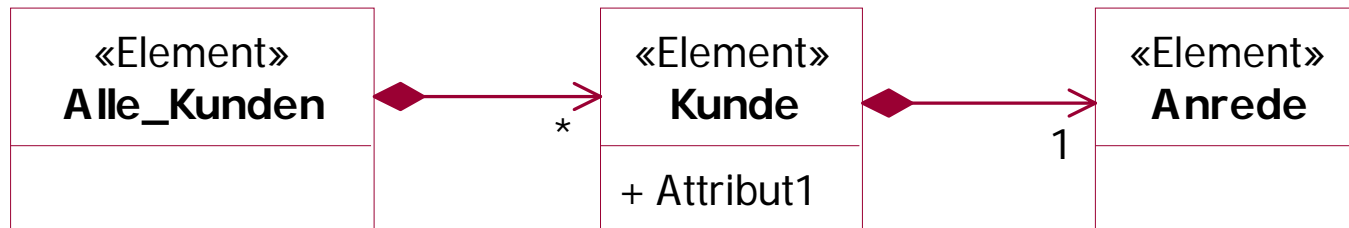


# Das hierarchische XML-Schemamodell

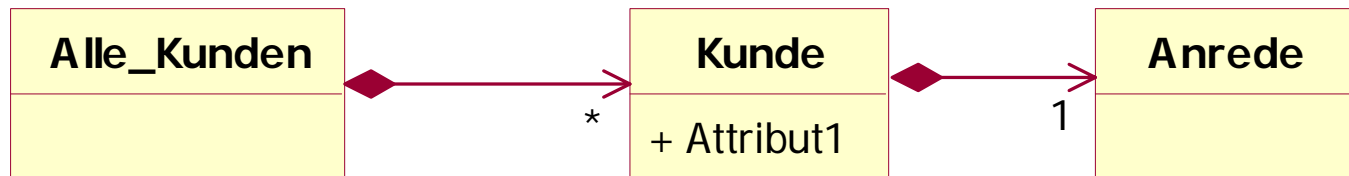
## Vorüberlegungen



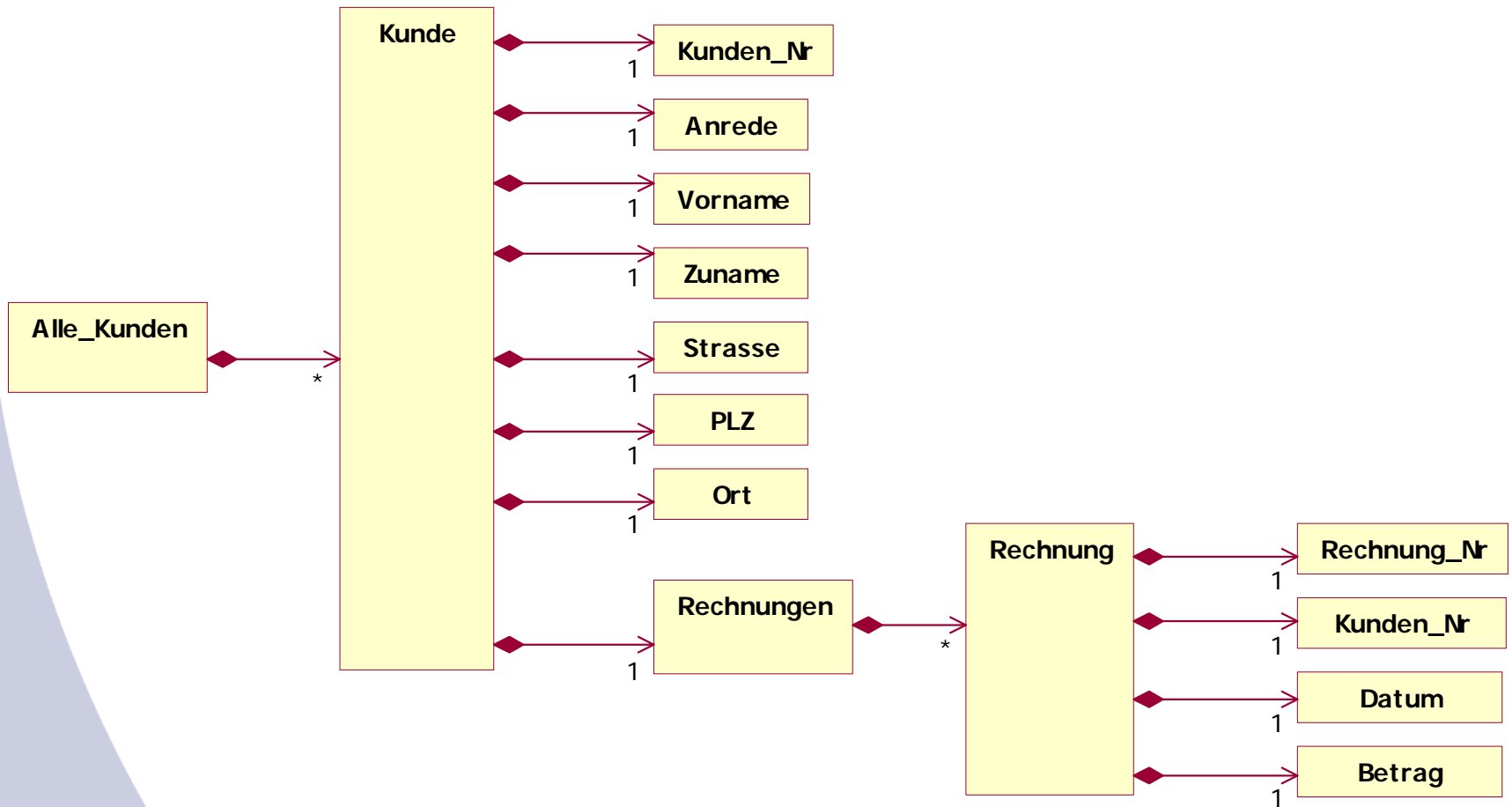
Das Stereotyp <<Element>> ist dargestellt



Aus Platzgründen weggelassenes Stereotyp <<Element>>



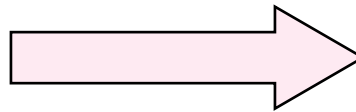
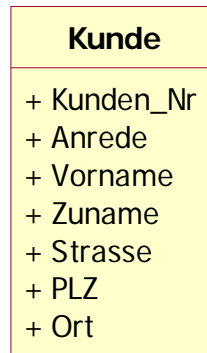
# Das hierarchische XML-Schemamodell für Beispiel



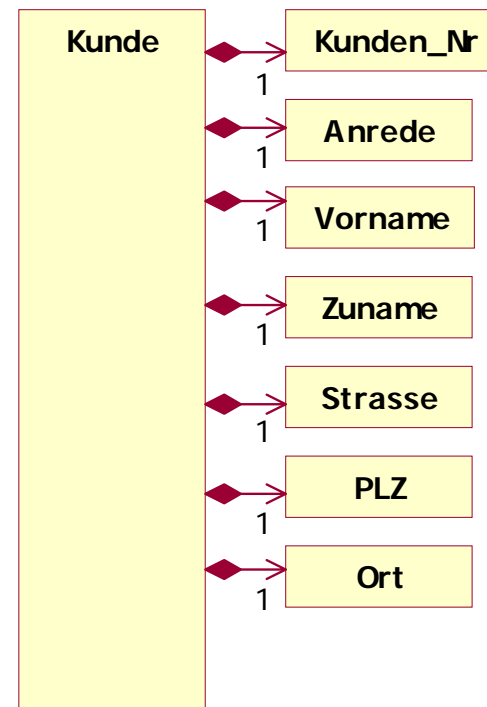
# Modell für Lösungsschritt 1

## Nur eine Kundenzeile wird publiziert

Relationen-Modell



XML-Schema-Modell



# Publikationsfunktionen von SQL/XML

## fortan verwendete Funktionen



Name	Verwendungszweck
<b>XMLAGG</b>	Sie ist eine Spaltenfunktion (Mengenfunktion). Sie fasst alle Werte einer Spalte vom Typ XML zu einem XML-Wert zusammen.
<b>XMLATTRIBUTES</b>	Sie erzeugt XML-Attribute eines XML-Elementes aus relationalen Daten.
<b>XMLELEMENT</b>	Sie erzeugt ein XML-Element im XML-Typ aus relationalen Daten.
<b>XMLFOREST</b>	Sie erzeugt eine Folge von XML-Elementen im XML-Typ aus relationalen Daten.
<b>XMLSERIALIZE</b>	Sie konvertiert einen Wert im XML-Typ in einen zeichenartigen Typ oder einen BLOB. Hierdurch entsteht aus der datenbankinternen Repräsentation eines XML-Wertes das korrespondierende XML-Dokument in der üblichen lesbaren Form.

# Verkürzte Grammatik der anschließend verwendeten Funktionen



**XMLELEMENT**( NAME "Name des XML-Elementes"  
[, Wertausdruck]...)

**XMLFOREST**( Wertausdruck AS "Name des XML-Elementes"  
[,Wertausdruck AS "Name des XML-  
Elementes"]...)

**XMLSERIALIZE**( CONTENT XML-Ausdruck AS CLOB )

# Lösungsschritt 1

## Programm

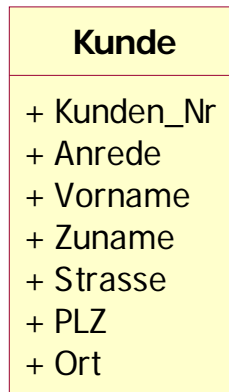


```
SELECT XMLSERIALIZE(  
    CONTENT  
    XMLELEMENT(  
        NAME "Kunde",  
        XMLFOREST(  
            K.Kunden_Nr AS "Kunden_Nr",  
            K.Anrede AS "Anrede",  
            K.Vorname AS "Vorname",  
            K.Zuname AS "Zuname",  
            K.Strasse AS "Strasse",  
            K.PLZ AS "PLZ",  
            K.Ort AS "Ort"  
        )  
    )  
    AS CLOB  
    ) AS Ergebnis  
FROM Kunde K  
WHERE K.Kunden_Nr = 'K001'  
;
```

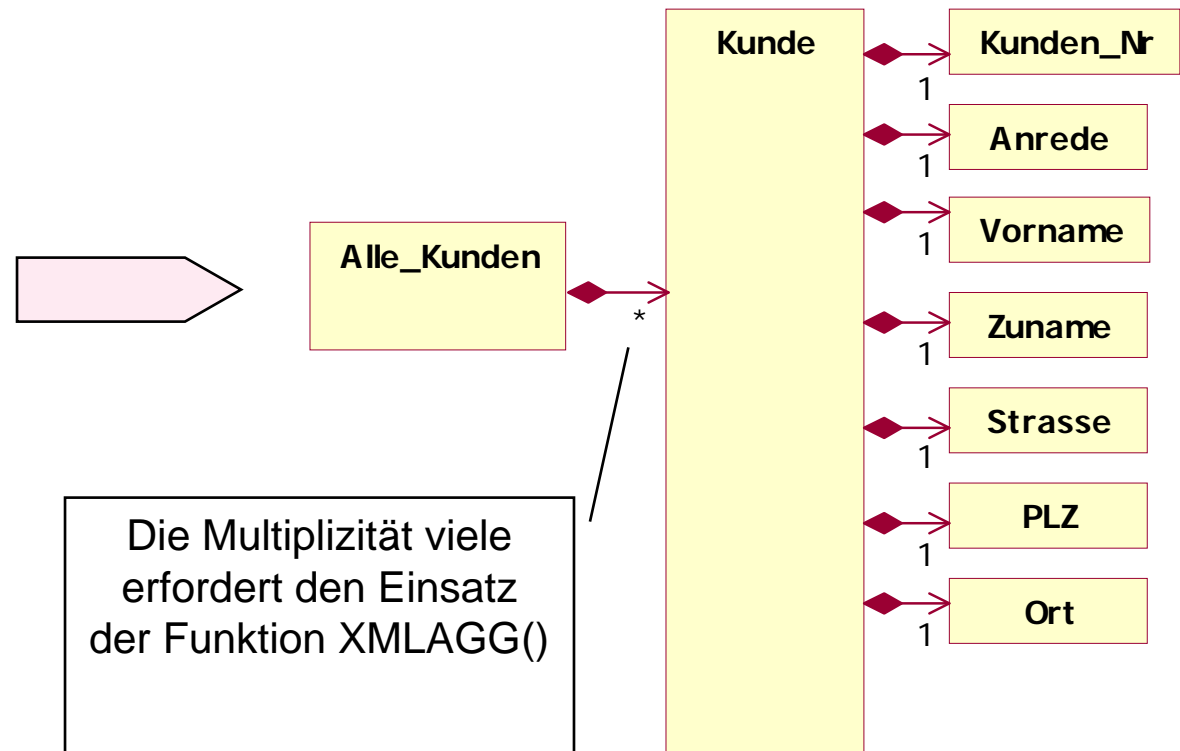
# Modell für Lösungsschritt 2

## Alle Kundenzeilen werden publiziert

### Relationen-Modell



### XML-Schema-Modell



# Verkürzte Grammatik der anschließend verwendeten Funktionen XMLAGG()



```
XMLAGG( XML-Spalte )
```

# Lösungsschritt 2 Programm



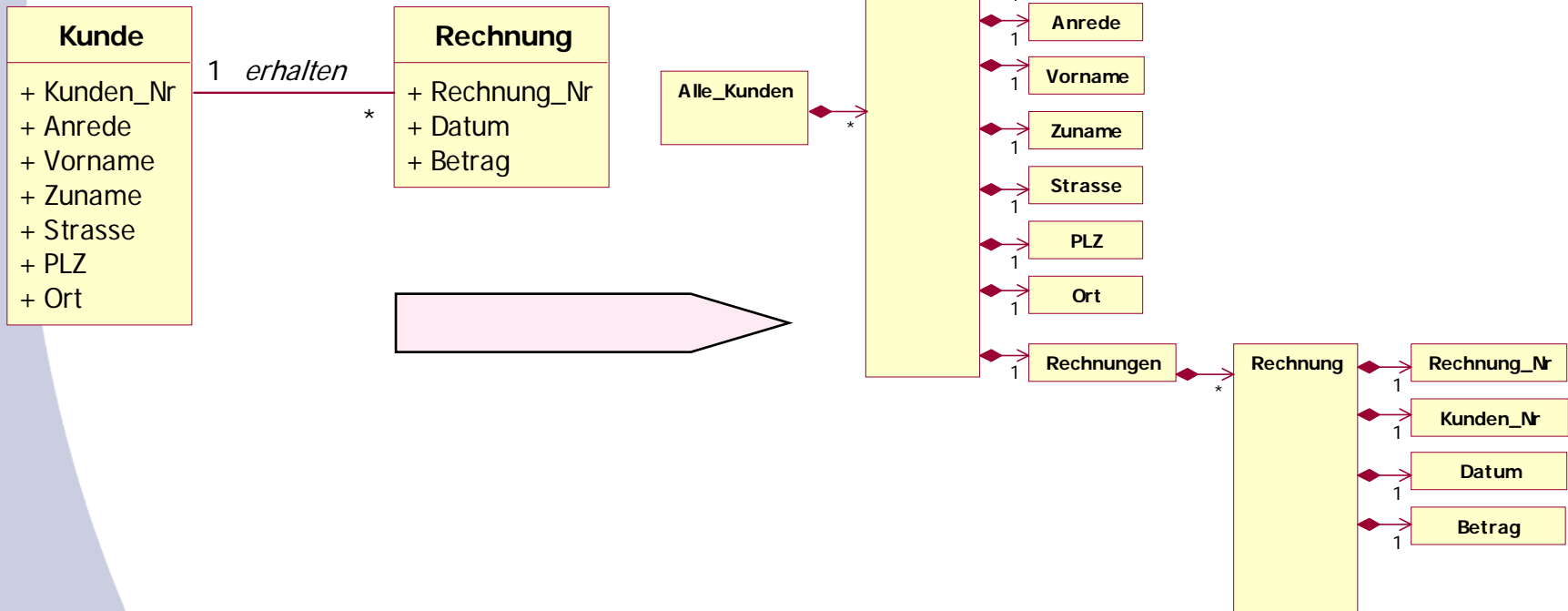
```
SELECT XMLSERIALIZE(  
    CONTENT  
    XMLELEMENT(  
        NAME "Alle_Kunden",  
        XMLAGG(XML_Kunde.XML_Spalte_Kunde)  
    )  
    AS CLOB  
    ) AS Ergebnis  
FROM LATERAL  
    (SELECT XMLELEMENT(  
        NAME "Kunde",  
        XMLFOREST(  
            K.Kunden_Nr AS "Kunden_Nr",  
            K.Anrede AS "Anrede" ,  
            K.Vorname AS "Vorname" ,  
            K.Zuname AS "Zuname" ,  
            K.Strasse AS "Strasse" ,  
            K.PLZ AS "PLZ" ,  
            K.Ort AS "Ort"  
        )  
    )  
    FROM Kunde K  
    ) AS XML_Kunde (XML_Spalte_Kunde)  
;
```

# Modell für Lösungsschritt 3

## Alle Kunden mit allen ihren Rechnungen

Relationen-Modell

XML-Schema-Modell



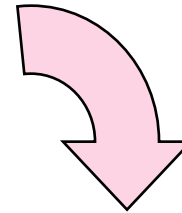
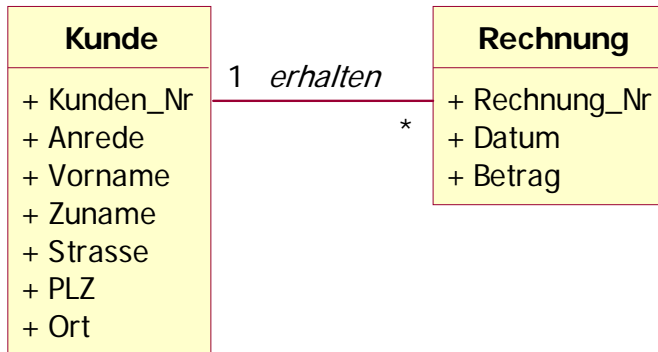
# Lösungsschritt 3 Programm mit einer korrelierten Unteranfrage



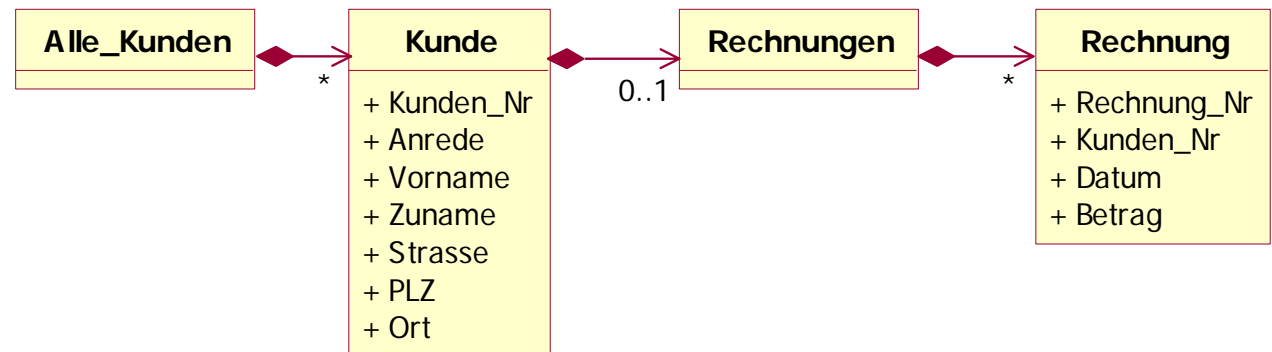
```
SELECT XMLSERIALIZE(  
  CONTENT  
  XMLELEMENT( NAME "Alle_Kunden",  
    XMLAGG(XML_Kunde.XML_Spalte_Kunde  
  )  
  )  
  AS CLOB  
  ) AS Ergebnis  
FROM LATERAL  
  (SELECT XMLELEMENT(  
    NAME "Kunde",  
    XMLFOREST(  
      K.Kunden_Nr AS "Kunden_Nr",  
      K.Anrede AS "Anrede" ,  
      K.Vorname AS "Vorname" ,  
      K.Zuname AS "Zuname" ,  
      K.Strasse AS "Strasse" ,  
      K.PLZ AS "PLZ" ,  
      K.Ort AS "Ort"  
    ),  
    XMLELEMENT(  
      NAME "Rechnungen",  
      (SELECT XMLAGG(XML_Rechnung.XML_Spalte_Rechnung  
      )  
      FROM LATERAL  
        (SELECT  
          XMLELEMENT(  
            NAME "Rechnung",  
            XMLFOREST(  
              R.Rechnung_Nr AS "Rechnung_Nr",  
              R.Kunden_Nr AS "Kunden_Nr" ,  
              R.Datum AS "Datum" ,  
              R.Betrag AS "Betrag"  
            )  
          )  
          FROM Rechnung R  
          WHERE R.Kunden_Nr = K.Kunden_Nr  
        ) AS XML_Rechnung (XML_Spalte_Rechnung)  
      )  
    )  
  )  
  )  
  ) AS XML_Kunde (XML_Spalte_Kunde)  
;
```

# Modell für Lösung in attributzentrierter Form

## Relationen-Modell



## XML-Schema-Modell



# Verkürzte Grammatik der anschließend verwendeten Funktionen



**XMLEMENT**( NAME "Name des XML-Elementes"  
[, XML-Attributs-Funktion]  
[, Wertausdruck]...)

**XMLATTRIBUTES**( Wertausdruck AS "Name des XML-Attributes"  
[,Wertausdruck AS "Name des XML-  
Attributes"]...)

# Programm für attributzentrierte From



```
SELECT XMLSERIALIZE(
  CONTENT
  XMLELEMENT( NAME "Alle_Kunden",
    XMLAGG(XML_Kunde.XML_Spalte_Kunde
  )
)
AS CLOB
) AS Ergebnis
FROM LATERAL
(SELECT XMLELEMENT(
  NAME "Kunde",
  XMLATTRIBUTES(
    K.Kunden_Nr AS "Kunden_Nr",
    K.Anrede AS "Anrede" ,
    K.Vorname AS "Vorname" ,
    K.Zuname AS "Zuname" ,
    K.Strasse AS "Strasse" ,
    K.PLZ AS "PLZ" ,
    K.Ort AS "Ort"
  ),
  XMLELEMENT(
    NAME "Rechnungen",
    (SELECT XMLAGG(XML_Rechnung.XML_Spalte_Rechnung
  )
  FROM LATERAL
  (SELECT
    XMLELEMENT(
      NAME "Rechnung",
      XMLATTRIBUTES(
        R.Rechnung_Nr AS "Rechnung_Nr",
        R.Kunden_Nr AS "Kunden_Nr" ,
        R.Datum AS "Datum" ,
        R.Betrag AS "Betrag"
      )
    )
  FROM Rechnung R
  WHERE R.Kunden_Nr = K.Kunden_Nr
  ) AS XML_Rechnung (XML_Spalte_Rechnung)
  )
)
)
FROM Kunde K
) AS XML_Kunde (XML_Spalte_Kunde)
;
```

# Programm in unästhetischer Form



```
SELECT XMLSERIALIZE(CONTENT XMLELEMENT(NAME
"Alle_Kunden", XMLAGG( XML_Kunde.XML_Spalte_Kunde))
AS CLOB) AS Ergebnis FROM LATERAL (SELECT
XMLELEMENT(NAME "Kunde", XMLATTRIBUTES(K.Kunden_Nr
AS "Kunden_Nr", K.Anrede AS "Anrede", K.Vorname AS
"Vorname", K.Zuname AS "Zuname", K.Strasse AS
"Strasse", K.PLZ AS "PLZ", K.Ort AS "Ort"),
XMLELEMENT(NAME "Rechnungen", (SELECT
XMLAGG(XML_Rechnung.XML_Spalte_Rechnung
) FROM LATERAL (SELECT XMLELEMENT(NAME "Rechnung",
XMLATTRIBUTES(R.Rechnung_Nr AS "Rechnung_Nr",
R.Kunden_Nr AS "Kunden_Nr", R.Datum AS "Datum",
R.Betrag AS "Betrag"))FROM Rechnung R WHERE
R.Kunden_Nr = K.Kunden_Nr) AS XML_Rechnung
(XML_Spalte_Rechnung)))) FROM Kunde K) AS XML_Kunde
(XML_Spalte_Kunde);
```



SRH Hochschule  
Heidelberg

# Danke

Prof. Alfred Moos

SRH Hochschule

Heidelberg

Ludwig-Guttman-Str. 6

E-Mail: [moos@fh-heidelberg.de](mailto:moos@fh-heidelberg.de)