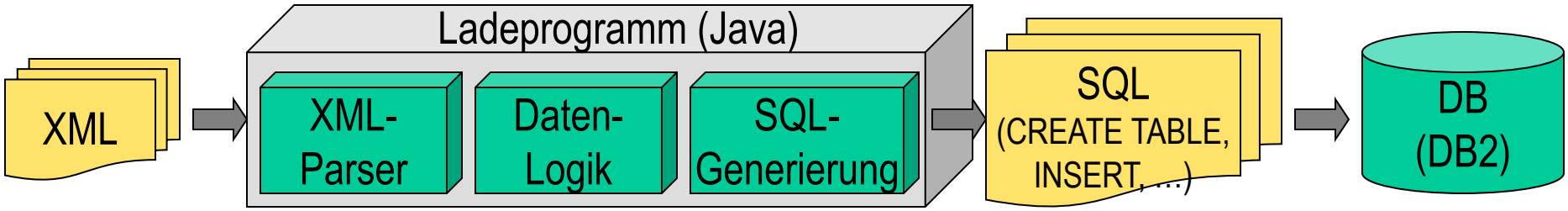
Relationales Datenbankpraktikum

Zusätzliche Hinweise zu Aufgabe 2 “Media Store”

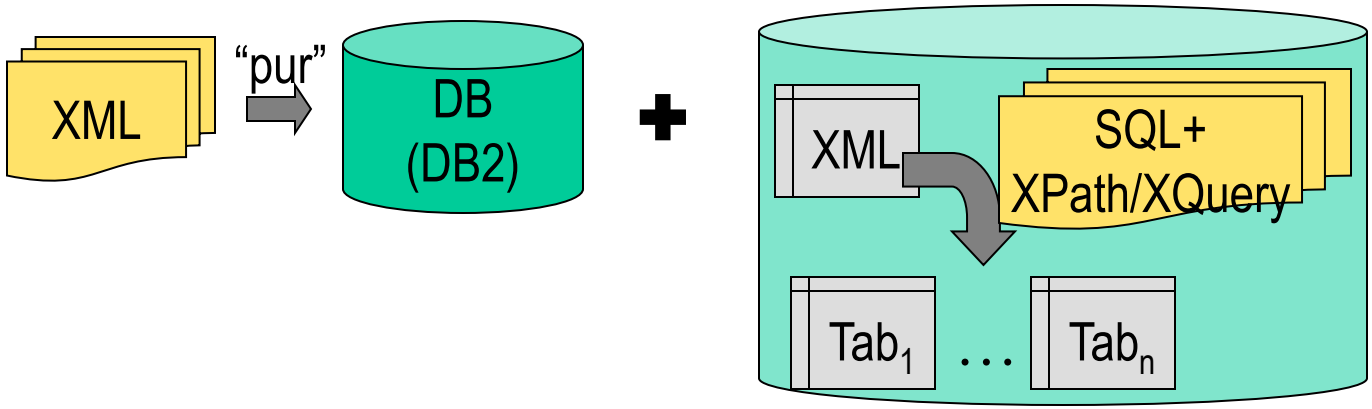
L. Kolb

A2: Speichern von XML-Daten in RM

- Variante 1: Java-Ladeprogramm



- Variante 2: Nutzung der XML-Funktionalität der DB



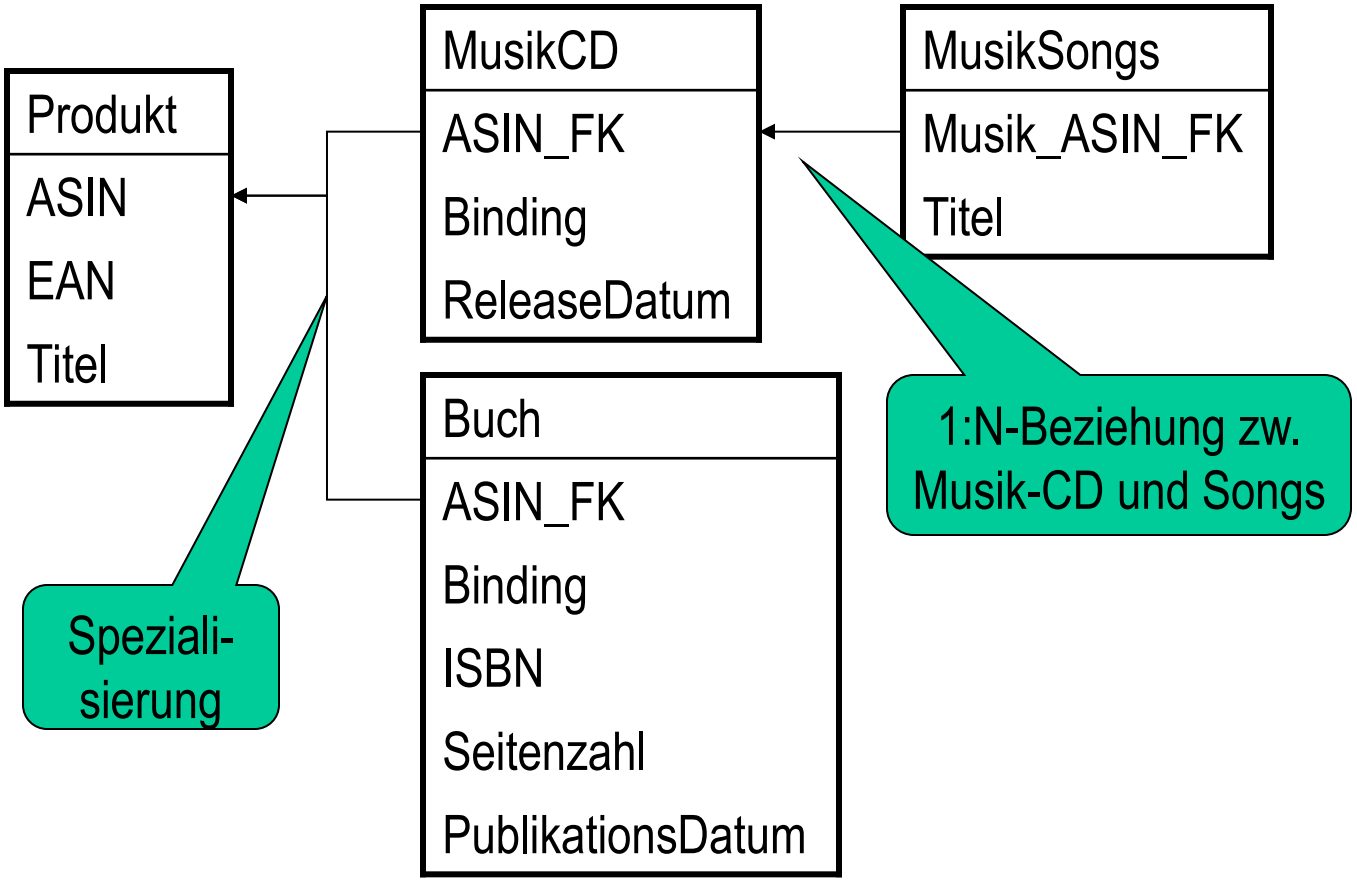
Beispiel-XML-Dokument: “Media Store”

- XML-Dokument, das Informationen zu Produkten enthält
- Zwei Produktarten
  - Musik-CDs (<Music>)
  - Bücher (<Book>)
- Gemeinsame XML-Elemente für alle Produkte
  - ASIN, EAN, Titel
- Unterschiedliche XML-Elemente je nach Produktart
  - Releasedatum und Tracks für Musik-CD
  - Seitenzahl, Publikationsjahr für Bücher

```
<leipzig>
 <Music asin="B0000668PG" ean="0721616028027">
  <title>In a Pig's Eye: Reflections on the …</title>
  <musicspec>
   <binding>CD</binding>
   <releasedate>2002-06-11</releasedate>
   <tracks>
     <title>Christian Parenti Introduction</title>
     <title>Hello My Relatives</title>
     <title>Anniversary of Pine Ridge</title>
   </tracks>
  </musicspec>
 </Music>
 <Book asin="3407788738" ean="9783407788733">
   <title>Von mir @n dich</title>
   <bookspec>
     <binding>Taschenbuch</binding>
     <isbn val="3407788738"
     <pages>200</pages>
     <publication date="2002-03-13" />
   </bookspec>
 </Book>
</leipzig>
```

Relationales Modell

- Ziel: Konvertierung XML in relationales Modell
- Allgemeine Produkttabelle + Spezialtabellen für einzelne Produkttypen mit entsprechenden Fremdschlüsselbeziehungen auf die Produkttabelle



Allgemeines Vorgehen

1. Speichern der XML-Dokumente in DB (pur)
2. Anfragen mittels XMLTABLE-Funktion, deren Ergebnisse mittels INSERT in (relationalen) Tabellen abgespeichert werden kann
   - Speicherung wird durch mehrere Anfragen realisiert
   - “Nicht alles auf einmal”, evtl. nur Teil des XML-Dokuments relevant

- Anfragen realisieren (befüllen) i.A. “typische Teile” des Datenmodells
  - Einfache Attribute
    - Beispiel: Produkt hat ASIN und Titel
  - Spezialisierungen
    - Beispiel: Musik-CDs und Bücher sind Produkte mit jeweils spezifischen Attributen
  - 1:N-Beziehungen = Vater-Kind-Elemente im XML
    - Beispiel: Eine Musik-CD hat mehrere Songs
  - XML-Hierarchien
    - Beispiel: Produkte sind in Kategorien eingeordnet, die wiederum hierarchisch angeordnet sind

Speichern des XML-Dokumentes in DB

- Hilfstabelle XMLDOK enthält “pures” XML-Dokument
- Verwendung des DB2-Datentyps “XML”
- Mehrere XML-Dokumente können in Tabelle gespeichert werden
  - Mehrere XML-Dokumente können später mit einer Anfrage verarbeitet werden
- Speicherung u.a. mittels Assistenten in IBM Data Server Developer Workbench
  - Rechte Maustaste auf XML-Spalte
  - “Insert XML content” → File auswählen
- Beispiel-Ergebnis (mit zusätzlicher Spalte für Dateinamen)

dbprak21.XMLDATEN	
datei(VARCHAR)	dok (XML)
leipzig	<leipzig><Music asin="B0000668PG“ ean="0721616028027"> <title>In a Pig's Eye: Reflections on the …

Einfache Attribute: Produktdaten

- XMLTABLE-Funktion liefert als Ergebnis relationale Tabelle
  - Kann für INSERT genutzt werden
- Innerhalb von XMLTABLE kann mittels XPath auf den XML-Inhalt zugegriffen werden
- Basiselemente
  - Für jedes Element wird ein Datensatz erzeugt
- Definition des/der XML-Dokumente(s) mittels PASSING
- Attributdefinition
  - Für jedes Attribut ein XPath-Ausdruck ausgehend vom Basiselement

```
INSERT INTO dbprak01.PRODUKT
(ASIN, EAN, Title)

SELECT X.ASIN, X.EAN, X.TITLE
FROM dbprak21.XMLDATEN AS XMLDOK,

XMLTABLE ('$MSXML/leipzig/child::node()'
   PASSING XMLDOK.dok AS MSXML
  COLUMNS
     "ASIN“   VARCHAR (15)   PATH '@asin',
     "EAN"    VARCHAR (50)   PATH '@ean',
     "TITLE"  VARCHAR (500) PATH 'title'
)AS X

WHERE XMLDOK.datei = 'leipzig'
```

Einfache Attribute: Produktdaten (2)

```
<leipzig>
 <Music asin="B0000668PG" ean="0721616028027">
    <title>In a Pig's Eye: Reflections on the …</title>
  …
 </Music>
 <Book asin="3407788738" ean="978340778873
    <title>Von mir @n dich</title>
  ….
 </Book>
</leipzig>
```

```
SELECT X.ASIN, X.EAN, X.TITLE
FROM dbprak21.XMLDATEN AS XMLDOK,
XMLTABLE ('$MSXML/leipzig/child::node()'
   PASSING XMLDOK.dok AS MSXML
   COLUMNS
      "ASIN“   VARCHAR (15)   PATH '@asin',
      "EAN“    VARCHAR (15)   PATH '@ean',
      "TITLE"  VARCHAR (250) PATH 'title'
)AS X
WHERE XMLDOK.datei = 'leipzig'
```

ASIN	EAN	TITLE
B0000668PG	0721616028027	In a Pig's Eye: Reflections on the …
3407788738	9783407788733	Von mir @n dich

Spezialisierung: Musik-CDs

```
<leipzig>
 <Music asin="B0000668PG" ean="0721616028027">
  <title>In a Pig's Eye: Reflections on the …</title>
  <musicspec>
   <binding>CD</binding>
   <releasedate>2002-06-11</releasedate>
   <tracks>
      <title>Christian Parenti Introduction</title>
      <title>Hello My Relatives</title>
      <title>Anniversary of Pine Ridge</title>
    </tracks>
   </musicspec>
  </Music>
</leipzig>
```

- Fremdschlüsselbeziehung zwischen Tabelle “Produkt” und “MusikCD” über eindeutiges Attribut (ASIN)
- Auch Nutzung automatisch generierter Ids möglich
  - JOIN (ASIN) + SELECT (ID)

```
SELECT X.ASIN, X.BINDING, X.RELEASE
FROM dbprak21.XMLDATEN AS XMLDOK,
XMLTABLE ('$MSXML/leipzig/Music'
   PASSING XMLDOK.dok AS MSXML
   COLUMNS
      "ASIN“    VARCHAR (15)   PATH '@asin',
      “BINDING“     VARCHAR (15)   PATH './musicspec/binding‘,
      “RELEASE"   VARCHAR (250) PATH './musicspec/releasedate'
)AS X
WHERE XMLDOK.datei = 'leipzig'
```

1:N-Beziehung: CD-Songs (1)

```
<leipzig>
 <Music asin="B0000668PG" ean="0721616028027">
  <title>In a Pig's Eye: Reflections on the .
  <musicspec>
   <binding>CD</binding>
   <releasedate>2002-06-11</releasedate>
   <tracks>
     <title>Christian Parenti Introduction<
     <title>Hello My Relatives</title>
     <title>Anniversary of Pine Ridge</title
    </tracks>
    </musicspec>
  </Music>
</leipzig>
```

```
SELECT X.ASIN, X.SONG
FROM dbprak21.XMLDATEN AS XMLDOK,
XMLTABLE ('$MSXML/leipzig/Music//tracks/title'
   PASSING XMLDOK.dok AS MSXML
   COLUMNS
      "ASIN"    VARCHAR (15)   PATH '../../@asin',
      "SONG"     VARCHAR (225)   PATH '. '
) AS X
WHERE XMLDOK.datei = ‘leipzig'
```

- Basiselemente = <title>-Elemente von Track
  - da jeder Songtitel in einem Datensatz gespeichert werden soll
- Beziehung zu Musik-CD durch “Hochnavigieren” zur ASIN (../../@asin)

1:N-Beziehung: CD-Songs (2)

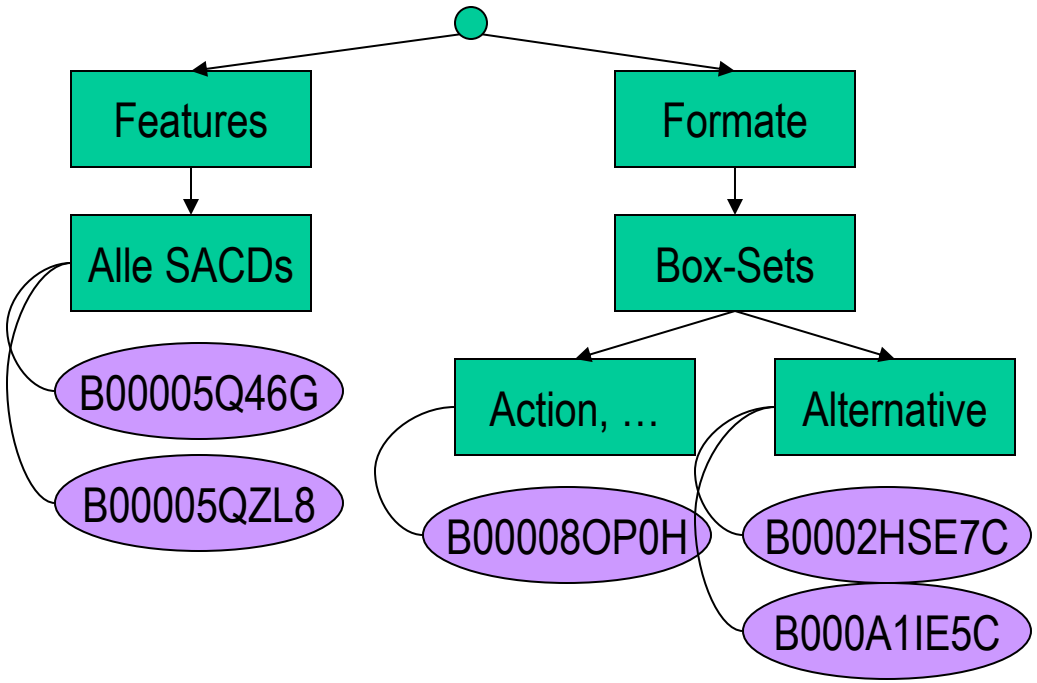
```
<leipzig>
 <Music asin="B0000668PG">
    <musicspec>
     <tracks>
      <title>Christian Parenti Introduction<
      <title>Hello My Relatives</title>
      <title>Anniversary of Pine Ridge</title
    </tracks>
    </musicspec>
  </Music>
 <Music asin="B00005YH0L">
  <musicspec>
     <tracks>
      <title>10: Bolero - Ljubljana Sym … </ti
      <title>Immortal Beloved: Symphony …
    </tracks>
    </musicspec>
  </Music>
</leipzig>
```

```
SELECT X.ASIN, X.SONG
FROM dbprak21.XMLDATEN AS XMLDOK,
XMLTABLE ('$MSXML/leipzig/Music//tracks/title'
   PASSING XMLDOK.dok AS MSXML
   COLUMNS
      "ASIN "    VARCHAR (15)   PATH '../../@asin',
      "SONG"     VARCHAR (255)   PATH '. '
) AS X
WHERE XMLDOK.datei = 'leipzig'
```

ASIN	TITLE
B0000668PG	Christian Parenti Introduction
B0000668PG	Hello My Relatives
B0000668PG	Anniversary of Pine Ridge
B00005YH0L	10: Bolero - Ljubljana Sym …
B00005YH0L	Immortal Beloved: Symphony …

Hierarchien

- Produkte sind Kategorien zugeordnet, die selbst hierarchisch angeordnet sind
  - Schwierig, wenn Kategoriename nicht eindeutig (wie bei uns)
  - Workaround, wenn Tiefe bekannt




```
<categories>
  <category>Features
    <category>Alle SACDs
      <item>B00005Q46G</item>
      <item>B00005QZL8</item>
    </category>
  </category>
  <category>Formate
    <category>Box-Sets
      <category>Action, Thriller & Horror
        <item>B00008OP0H</item>
      </category>
      <category>Alternative
        <item>B0002HSE7C</item>
        <item>B000A1IE5C</item>
      </category>
    </category>
   </category>
</categories>
```

Schritt 1: Denormalisierte Darstellung

```
<categories>
  <category>Features
    <category>Alle SACDs</category>
  </category>
  <category>Formate
    <category>Box-Sets
      <category>Action, Thriller & Horror</ca
      <category>Alternative</category>
    </category>
  </category>
</categories>
```

```
INSERT INTO Kategorie
(Kat1, Kat2, Kat3)
SELECT X.CAT1, X.CAT2, X.CAT3
FROM dbprak21.XMLDATEN AS XMLDOK,
XMLTABLE ('$MSXML//category'
   PASSING XMLDOK.dok AS MSXML
   COLUMNS
      "CAT1"   VARCHAR (255) PATH 'text()',
      "CAT2"  VARCHAR (255)  PATH '../text()',
      "CAT3"  VARCHAR (255)  PATH '../../text()'
) AS X
WHERE XMLDOK.datei = 'categories'
```

ID	Kat1	Kat2	Kat3
1	Features		
2	Alle SACDs	Features	
3	Formate		
4	Box-Sets	Formate	
5	Action, Thriller & Horror	Box-Sets	Formate
6	Alternative	Box-Sets	Formate

Zur Eliminierung von Zeilenumbrüchen bei Mixed-Content Anwendung von fn:normalize-space in PATH-Ausdruck, also z.B. für CAT2 '../fn:normalize-space(text())'

Schritt 2: Normalisierung mittels SQL

ID	Kat1	Kat2	Kat3
1	Features		
2	Alle SACDs	Features	
3	Formate		
4	Box-Sets	Formate	
5	Action, Thriller & Horror	Box-Sets	Formate
6	Alternative	Box-Sets	Formate

SQL! SQL! SQL!

ID	KatName	KatPfad	Level	OberKat	HauptKat
1	Features		1	*NULL*	1
2	Alle SACDs	/Features	2	1	1
3	Formate		1	*NULL*	3
4	Box-Sets	/Formate	2	3	3
5	Action, Thriller & Horror	/Formate/Box-Sets	3	4	3
6	Alternative	/Formate/Box-Sets	3	4	3