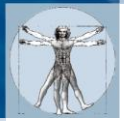


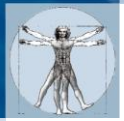
SIMPL

-

SimTech: Information Management, Processes and Languages

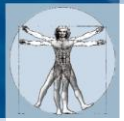
Datenmodelle und Datenbanksprachen

TU,XI

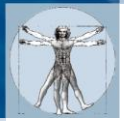


Agenda

- Relationales Datenmodell
- Relationale Operationen
- Standardsprache SQL
- XPath
- XQuery



RELATIONALES DATENMODELL



Relationale Datenmodell

Einführung

- Eine R(A1,A2,…,An) ⇔ Ein RM
- Beispiel:

Attributen

Student ← Tabellenname

MatrNr	Name	Semester	PLZ	Wohnort
1001	Andy	6	70100	Stuttgart
1002	Rene	7	70569	Stuttgart
1003	Stephen	7	80100	München
1004	Martin	6	80100	München

PS

Tupel/Zeile

Relationale Datenmodell

Fremdschlüssel

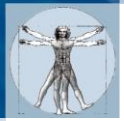
- Fremdschlüssel

Student

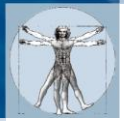
MatrNr	Name	Semester	PLZ	Wohnort
1001	Andy	6	70100	Stuttgart
1002	Rene	7	70569	Stuttgart
1003	Stephan	7	80100	München
1004	Martin	6	80100	München

Prüfung

VorlNr	MatrNr	Datum	Note
2520	1002	01.04.2007	1.7
2628	1003	01.03.2007	2



RELATIONALE OPERATIONEN



Relationale Operationen

Zeichnungen

- Klassische Mengenoperationen:
  - Vereinigung: $R \cup S$
  - Differenz: $R - S$
  - Durchschnitt: $R \cap S = R – (R – S)$
  - Symmetrische Differenz: $R \; X \; S = (R \cup S) – (R \cap S)$

Relationenoperationen:

- Restriktion (Selektion) : σP
- Projektion: π
- E. kartesisches Produkt: X
- Verbund (Join) : join

Relationale Operationen

Restriktion(Selektion): σp

Auswahl von Zeilen/Tupeln einer Relation über ein Prädikat.

Beispiel:

P=(semester=6 Λ wohnort=Stuttgart) bei Relation Student

MatrNr	Name	Semester	PLZ	Wohnort
1001	Andy	6	70100	Stuttgart
1002	Rene	7	70569	Stuttgart
1003	Stephen	7	80100	München
1004	Martin	6	80100	München

Relationale Operationen

Projektion: π

Auswahl von Spalten (Attribute) aus einer Relation

Beispiel:

π PLZ,Wohnort(Student)

MatrNr	Name	Semester	PLZ	Wohnort
100020	Albrecht	6	70100	Stuttgart
100300	Rene	7	70156	Stuttgart
100400	Carl	7	80100	München
100500	Daniela	6	80100	München

PLZ	Wohnort
70100	Stuttgart
70156	Stuttgart
80100	München

Relationale Operationen
erweitertes Kartesisches Produkt: R X S

- Teammates

PerNr	Name	Wohnort
1001	Beckenbauer	München
1002	Müller	Stuttgart

Status

Name	Höhe	Gewicht
Beckenbauer	183	85
Müller	163	70

- Teammte X Status

PerNr	Name	Wohnort	Name	Höhe	Gewicht
1001	Beckenbauer	München	Beckenbauer	183	85
1001	Beckenbauer	München	Müller	163	70
1002	Müller	Stuttgart	Beckenbauer	183	85
1002	Müller	Stuttgart	Müller	163	70

Relationale Operationen

Verbund

Relationen:R und S

Attribut: A von R und B von S.

$\Theta \in \{<, =, >, \leq, \neq, \geq\}$

$V = (R \text{ join } S) = \sigma_{A \Theta B}(R \times S)$

$A \Theta B$

R

A	B	C
A1	B1	5
A1	B2	6

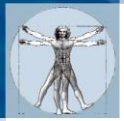
S

B	E
B1	3
B2	7
B3	10

R join S

C<E

A	R.B	C	S.B	E
A1	B1	5	B2	7
A1	B1	5	B3	10
A1	B2	6	B2	7
A1	B2	6	B2	10



Relationale Operationen

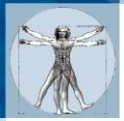
Natuerlicher Verbund

A	R.B	C	S.B	E
A1	B1	5	B1	3
A2	B2	6	B2	7

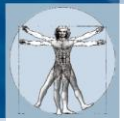
Nomale Gleichverbund

A	B	C	E
A1	B1	5	3
A1	B2	6	7

Natürlicher Verbund



STANDARDSPRACHE SQL

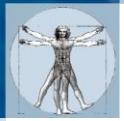


Standardsprache SQL

SQL-Anfragen

- Form des Statements:

select-exp ::= **SELECT** [**ALL** | **DISTINCT**] select-item-commalist
FROM table-ref-commalist
[**WHERE** cond-exp]
[**GROUP BY** column-ref-commalist 1][**HAVING** cond-exp]
[**ORDER BY** column-ref-commalist 2[ASC|DESC]]



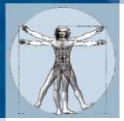
Standardsprache SQL

Beispiel von SQL

- Beispiel:

```
SELECT *
FROM   Student;
WHERE  (Semester=6) AND (Wohnort=Stuttgart)
```

```
SELECT  *
FROM    Π3,5 (Student Natürlich Join Prüfung)
Where (Semester=7) AND (Wohnort=Stuttgart)
```



Standardsprache SQL

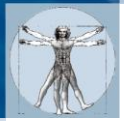
Aggregatfunktionen

- aggregate-function-ref ::= **COUNT**(*) |

  {**AVG** | **MIN** | **MAX** | **SUM** | **COUNT**} ([**ALL** | **DISTINCT**] scalar-exp)

  Beispiel:

  SELECT AVG(Note) **AS** Durchschnittsnote
  FROM Student
  WHERE Wohnort=Stuttgart;

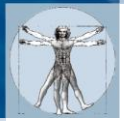


Standardsprache SQL

Mit Mengenoperationen

- Beispiel:

```
SELECT  *
FROM    Student
WHERE   Wohnort=‚Stuttgart‘
INTERSECT(UNION,EXCEPT)
SELECT  *
FROM    Student
WHERE   Semester=‚6‘
```



Datenmanipulation: Einfügen und Löschen von Tupeln

Einfügen VON TUPELN

INSERT INTO table [(column-commalist)]
{ **VALUES** row-constr.-commalist | table-exp | **DEFAULT VALUES** }

Beispiel:

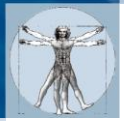
INSERT INTO Student (MatrNr, NAME, Semester, PLZ, Wohnort) **VALUES** (1006, „xi“,7, 70569, “Stuttgart”);

Löschen VON TUPELN

DELETE FROM table **[WHERE** cond-exp**]**

Beispiel:

DELETE FROM Student **WHERE** MatrNr = 1001;



Standardsprache SQL

Datenmanipulation: Ändern

UPDATE VON TUPELN

UPDATE table
SET update-assignment-commalist
[WHERE cond-exp**]**

Beispiel:

UPDATE Student
SET PLZ = 70569
WHERE Wohnort = “Stuttgart”

Standardsprache SQL

Datendefinition

CREATE SCHEMA[schema] [**AUTHORIZATION** user]
[**DEFAULT CHARACTER SET** char-set]
[schema-element-list]

Beispiel:
CREATE SCHEMA Student **AUTHORIZATION** TU

Standardsprache

Attributen: Definition

column-def ::= column { data-type | domain }
[**DEFAULT** { literal | niladic-function-ref | **NULL**}]
[column-constraint-def-list]

Beispiel:
NAME **CHAR** (20)
SEMESTER(**AS INT DEFAULT 1**)

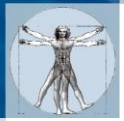
Standardsprache

CREATE TABLE

CREATE TABLE base-table (base-table-element-commalist)
base-table-element ::= column-def | base-table-constraint-def

Beispiel:

CREATE TABLE Teammate
(PerNr Int NOT NULL,
Höhe Int NOT NULL**,**
Gewicht Int NOR NULL,
NAME CHAR (30) NOT NULL,
Wohnort CHAR (30) NOT NULL,
Semester INT NOT NULL**,**
PLZ INT NOT NULL
PRIMARY KEY (PerNr))



Standardsprache SQL

Tabelle: Löschen und Änderung

DROP

DROP { **TABLE** base-table | **VIEW** view | **DOMAIN** domain |
SCHEMA schema } { **RESTRICT** | **CASCADE** }

Beispiel:

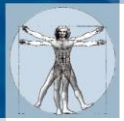
DROP TABLE Pers **RESTRICT**

ALTER

ALTER TABLE base-table
{**ADD** [**COLUMN**] column-def
| **ALTER** [**COLUMN**] column {**SET** default-def | **DROP DEFAULT**}
| **DROP** [**COLUMN**] column {**RESTRICT** | **CASCADE**}
| **ADD** base-table-constraint-def
| **DROP CONSTRAINT** constraint {**RESTRICT** | **CASCADE**}}

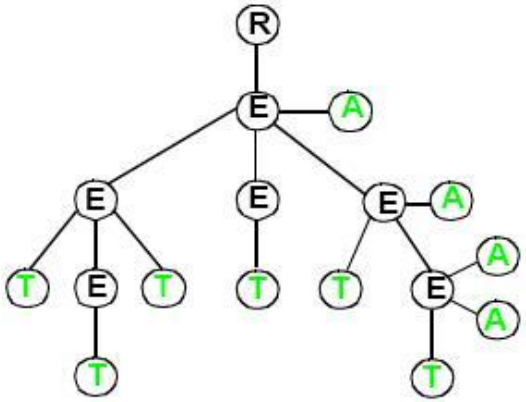
Beispiele:

ALTER TABLE Teammate **ADD Alter INT**
ALTER TABLE Teammate **DROP COLUMN** Wohnort **RESTRICT**

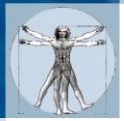


XPATH

Xpath
Baum Modell



Baum Modell von XML Daten



XPath
Xpath:Funktionen

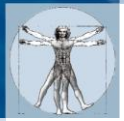
4 Untermengen von Funktionen:
Node Set Functions; String Functions; Boolean Functions; Number Functions.

```
<Teammate>
    <peron Höhe=„180“>
        <name>
            Beckenbauer
        </name>
        <job>Fussballer</job>
        <job>Manager</job>
    </person>
      ......
</Teammate>
```

XML-Beispiel

```
/child::Teammate
/child::person[child::name=„Beckenbauer“]
/attribute::Höhe
```

XPath-Ausdrücke

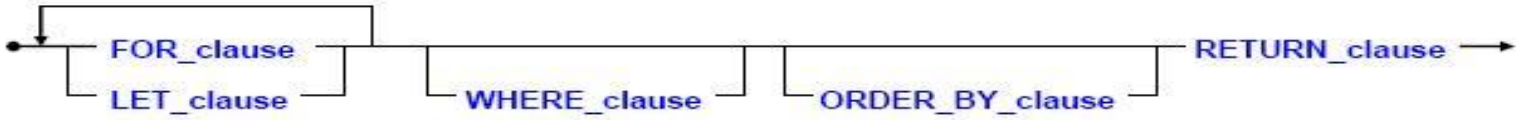


XQUERY

Xquery

Standard-Ausdruck FLOWR

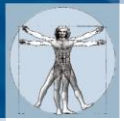
Die standarde Ausdruck-Flowr



Beispiel:

suche alle Teammate, die Höhe mehr als 180 haben. Und versammelt alle Ergebniss in eine List, die Name heisst.

FOR $x **IN** /Teammate/Höhe
LET $pn:= $x/@Name
WHERE $x/Höhe > 180
RETURN <Name> {$pn} </Name>



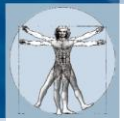
Xquery

Joins in XQuery

```
FOR $t IN /Person/teammate,
        $c IN /Person/coach,
WHERE $a/Staat = $d/Staat
RETURN <National>{ $c $a }</National>
```

Kartesiches Produkt:

```
FOR $t IN doc("tt.xml")/person/teammate,
      $c IN doc("tt.xml")/person/coach
RETURN <national>{$c}{$a}</national>
```



XQuery

Datenmanipulation

Einfügen von Knoten

Syntax: **do insert**<source-expression> ([**as(first| last)] into| after| before)** <target-expression>.

Beispiel:

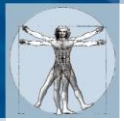
do insert <year>1949</year> after
fn:doc(“tt.xml")/person/teammate/Geburtstag

Löschen null oder mehr Knoten

Syntax: **do delete**<target-expression>

Beispiel:

do delete fn:doc(“tt.xml")/person/teammate/wohnort



XQuery

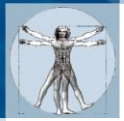
Datenmanipulation (2)

Ersetzen von Knoten/Werten

Syntax: **do replace[value of]** <target-expression> **with**<new-expression>

Beispiel:

do replace fn:doc(“tt.xml")/person/teammate/name
with fn:doc(“tt.xml")/person/coach/name



XQuery

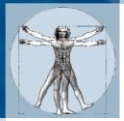
Erstellen einer geänderten Kopie

Syntax: **transform**

copy<var> :=<expr> {,<var> :=<expr>}*
modify<updating-expression>
return<return-expression>

Beispiel:

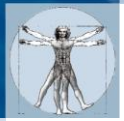
```
FOR $b IN fn:doc("tt.xml")/person/teammate[contains(name, "Müller")]
RETURN
    transform
            copy $xb:= $b
            modify do delete $xb/Höhe
            return $xb
```



Zusammenfassung

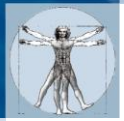
Ich habe bei Kapitel 1 ueber die RM,Kapitel 2 ueber die Relationale Operation,Kapitel 3 Ueber die SQL-Sprache von Relationdatenbank beschrieben, und bei Kapitel 4,5 die XPath und XQuery grob erklaert.

Ich hoffe, durch Erklärung der Datenmodell und Datensprachen von Relational und XML kann man die SQL und XQuery in den SIMPL leicht verwenden.



Bibliography

- Kemper, A.; Eickler, A.: Datenbanksysteme - Eine Einführung, Oldenbourg, 6. Auflage, 2006 .
- Date, C. J.: An Introduction to Database Systems, Addison-Wesley Publ. Comp., Reading, Mass., 8th Ed., 2004
- Lehner, W.; Schöning, H.: XQuery: Grundlagen und fortgeschrittene Methoden, dpunkt, 2004
- W3C: XQuery – Technical Report, http://www.w3.org/TR/xquery/



VIELEN DANK FÜR IHRE AUFMERKSAMKEIT