

Kapitel 2: Einstieg in SQL

- ▶ SQL (Structured Query Language) ist die in der Praxis am weitesten verbreitete Datenbanksprache für relationale Datenbanken.
- ▶ Die Historie von SQL geht zurück bis 1974, die Anfangszeit der Entwicklung relationaler Datenbanken.
- ▶ Alles begann mit SEQUEL, der *Structured English Query Language*.
- ▶ Der Sprachumfang von SQL ist einer permanenten Weiterentwicklung und Standardisierung unterworfen. Derzeit relevant sind der Stand von 1992, 1999, 2003 und 2008 entsprechend bezeichnet mit SQL-92, SQL:1999, SQL:2003 und SQL:2008.

Ein *Anfrageausdruck* in SQL besteht aus einer SELECT-Klausel, gefolgt von einer FROM-Klausel, gefolgt von einer WHERE-Klausel.

SFW-Ausdruck

```
SELECT A1, ..., An (...Attribute der Ergebnisrelation)
FROM R1, ..., Rm (...benötigte Relationen)
WHERE F (...Auswahlbedingung)
```

2.1 Beispiels-Datenbank

Mondial-Datenbank Teil 1

Land				Provinz		
LName	LCode	HStadt	Fläche	PName	LCode	Fläche
Austria	A	Vienna	84	Baden	D	15
Egypt	ET	Cairo	1001	Bavaria	D	70,5
France	F	Paris	547	Berlin	D	0,9
Germany	D	Berlin	357	Ile de France	F	12
Italy	I	Rome	301	Franken	D	null
Russia	RU	Moscow	17075	Lazio	I	17
Switzerland	CH	Bern	41			
Turkey	TR	Ankara	779			

Stadt					
SName	LCode	PName	Einwohner	LGrad	BGrad
Berlin	D	Berlin	3472	13,2	52,45
Freiburg	D	Baden	198	7,51	47,59
Karlsruhe	D	Baden	277	8,24	49,03
Munich	D	Bavaria	1244	11,56	48,15
Nuremberg	D	Franken	495	11,04	49,27
Paris	F	Ile de France	2125	2,48	48,81
Rome	I	Lazio	2546	12,6	41,8

Mondial-Datenbank Teil 2

Lage			Mitglied		
LCode	Kontinent	Prozent	LCode	Organisation	Art
D	Europe	100	A	EU	member
F	Europe	100	D	EU	member
TR	Asia	68	D	WEU	member
TR	Europe	32	ET	UN	member
ET	Africa	90	I	EU	member
ET	Asia	10	I	NAM	guest
RU	Asia	80	TR	UN	member
RU	Europe	20	TR	CERN	observer

2.2 Einfache Anfragen

..... Anfragen über einer Relation: *gesamter Inhalt*.

Gib den vollständigen Inhalt der Tabelle Stadt.

```
SELECT * FROM Stadt
```

Stadt					
SName	LCode	PName	Einwohner	LGrad	BGrad
Berlin	D	Berlin	3472	13,2	52,45
Freiburg	D	Baden	198	7,51	47,59
Karlsruhe	D	Baden	277	8,24	49,03
Munich	D	Bavaria	1244	11,56	48,15
Nuremberg	D	Franken	495	11,04	49,27
Paris	F	Ile de France	2125	2,48	48,81
Rome	I	Lazio	2546	12,6	41,8

..... Anfragen über einer Relation: *einzelne Spalten*.

In welchen Provinzen liegen die Städte der Tabelle Stadt?

```
SELECT PName FROM Stadt
```

PName
Berlin
Baden
Baden
Bavaria
Franken
Ile de France
Lazio

In welchen *unterschiedlichen* Provinzen liegen die Städte der Tabelle Stadt?

```
SELECT DISTINCT PName FROM Stadt
```

PName
Berlin
Baden
Bavaria
Franken
Ile de France
Lazio

..... Anfragen über einer Relation: *einzelne Zeilen*.

Stadt

SName	LCode	PName	Einwohner	LGrad	BGrad
Berlin	D	Berlin	3472	13,2	52,45
Freiburg	D	Baden	198	7,51	47,59
Karlsruhe	D	Baden	277	8,24	49,03
Munich	D	Bavaria	1244	11,56	48,15
Nuremberg	D	Franken	495	11,04	49,27
Paris	F	Ile de France	2125	2,48	48,81
Rome	I	Lazio	2546	12,6	41,8

Wie heißen die Städte, die mehr als 1 Mio. Einwohner haben?

```
SELECT * FROM Stadt
WHERE Einwohner > 1000
```

Stadt

SName	LCode	PName	Einwohner	LGrad	BGrad
Berlin	D	Berlin	3472	13,2	52,45
Munich	D	Bavaria	1244	11,56	48,15
Paris	F	Ile de France	2125	2,48	48,81
Rome	I	Lazio	2546	12,6	41,8

..... Anfragen über einer Relation: *geänderte Spaltenbezeichnungen und neue Spalten*.

Stadt

SName	LCode	PName	Einwohner	LGrad	BGrad
Berlin	D	Berlin	3472	13,2	52,45
Freiburg	D	Baden	198	7,51	47,59
Karlsruhe	D	Baden	277	8,24	49,03
Munich	D	Bavaria	1244	11,56	48,15
Nuremberg	D	Franken	495	11,04	49,27
Paris	F	Ile de France	2125	2,48	48,81
Rome	I	Lazio	2546	12,6	41,8

Kennzeichne die Tatsache, dass eine Stadt mehr als 1 Mio. Einwohner hat, durch den Wert 'Großstadt' einer neuen Spalte mit Namen StadtKategorie; die Spalte SName soll die Bezeichnung Stadt erhalten.

```
SELECT SName AS Stadt, 'Großstadt' AS StadtKategorie FROM Stadt
WHERE Einwohner > 1000
```

Stadt	StadtKategorie
Berlin	Großstadt
Munich	Großstadt
Paris	Großstadt
Rome	Großstadt

..... Anfragen über einer Relation: *Pattern Matching* (a).

Land			
LName	LCode	HStadt	Fläche
Austria	A	Vienna	84
Egypt	ET	Cairo	1001
France	F	Paris	547
Germany	D	Berlin	357
Italy	I	Rome	301
Russia	RU	Moscow	17075
Switzerland	CH	Bern	41
Turkey	TR	Ankara	779

Erstelle eine Namensliste der Länder, deren Namen mit 'G' anfängt oder mit 'y' aufhört?

```
SELECT LName FROM Land WHERE LName LIKE 'G%' OR LName LIKE '%y'
```

LName
Germany
Italy
Turkey

..... Anfragen über einer Relation: *Pattern Matching* (b).

Land			
LName	LCode	HStadt	Fläche
Austria	A	Vienna	84
Egypt	ET	Cairo	1001
France	F	Paris	547
Germany	D	Berlin	357
Italy	I	Rome	301
Russia	RU	Moscow	17075
Switzerland	CH	Bern	41
Turkey	TR	Ankara	779

Erstelle eine Namensliste der Länder, deren dritter Buchstabe des Namen 'y' ist?

```
SELECT LName FROM Land WHERE LName LIKE ' _y%'
```

LName
Egypt

..... Anfragen über mehreren Relationen.

Land				Stadt					
LName	LCode	HStadt	Fläche	SName	LCode	PName	Einwohner	LGrad	BGrad
Austria	A	Vienna	84	Berlin	D	Berlin	3472	13,2	52,45
Egypt	ET	Cairo	1001	Freiburg	D	Baden	198	7,51	47,59
France	F	Paris	547	Karlsruhe	D	Baden	277	8,24	49,03
Germany	D	Berlin	357	Munich	D	Bavaria	1244	11,56	48,15
Italy	I	Rome	301	Nuremberg	D	Franken	495	11,04	49,27
Russia	RU	Moscow	17075	Paris	F	Ile de France	2125	2,48	48,81
Switzerland	CH	Bern	41	Rome	I	Lazio	2546	12,6	41,8
Turkey	TR	Ankara	779						

Gib zu jedem Land die zugehörigen Städte an.

```
SELECT DISTINCT Land.LName AS Land, Stadt.SName AS Stadt
FROM Land, Stadt
WHERE Land.LCode = Stadt.LCode
```

Land	Stadt
Germany	Berlin
Germany	Freiburg
Germany	Karlsruhe
Germany	Munich
Germany	Nuremberg
France	Paris
Italy	Rome

..... Anfragen über mehreren Relationen mit Auswahlbedingung.

Land				Stadt					
LName	LCode	HStadt	Fläche	SName	LCode	PName	Einwohner	LGrad	BGrad
Austria	A	Vienna	84	Berlin	D	Berlin	3472	13,2	52,45
Egypt	ET	Cairo	1001	Freiburg	D	Baden	198	7,51	47,59
France	F	Paris	547	Karlsruhe	D	Baden	277	8,24	49,03
Germany	D	Berlin	357	Munich	D	Bavaria	1244	11,56	48,15
Italy	I	Rome	301	Nuremberg	D	Franken	495	11,04	49,27
Russia	RU	Moscow	17075	Paris	F	Ile de France	2125	2,48	48,81
Switzerland	CH	Bern	41	Rome	I	Lazio	2546	12,6	41,8
Turkey	TR	Ankara	779						

Gib zu jedem Land die zugehörigen Städte mit mehr als 1 Mio Einwohner an.

```
SELECT DISTINCT Land.LName AS Land, Stadt.SName AS Stadt
FROM Land, Stadt
WHERE Land.LCode = Stadt.LCode AND Einwohner > 1000
```

Land	Stadt
Germany	Berlin
Germany	Munich
France	Paris
Italy	Rome

Land				Stadt							
LName	LCode	HStadt	Fläche	SName	LCode	PName	Einvohner	LGrad	BGrad	Land	Stadt
Austria	A	Vienna	84	Berlin	D	Berlin	3472	13,2	52,45	Germany	Berlin
Egypt	ET	Cairo	1001	Freiburg	D	Baden	198	7,51	47,59	Germany	Freiburg
France	F	Paris	547	Karlsruhe	D	Baden	277	8,24	49,03	Germany	Karlsruhe
Germany	D	Berlin	357	Munich	D	Bavaria	1244	11,56	48,15	Germany	Munich
Italy	I	Rome	301	Nuremberg	D	Franken	495	11,04	49,27	Germany	Nuremberg
Russia	RU	Moscow	17075	Paris	F	Ile de France	2125	2,48	48,81	France	Paris
Switzerland	CH	Bern	41	Rome	I	Lazio	2546	12,6	41,8	Italy	Rome
Turkey	TR	Ankara	779								

Gib zu jedem Land die zugehörigen Städte an.

```
SELECT DISTINCT Land.LName AS Land, Stadt.SName AS Stadt
FROM Land, Stadt
WHERE Land.LCode = Stadt.LCode
```

intuitive deklarative Semantik

Das Ergebnis besteht aus denjenigen Tupeln des kartesischen Produktes $\text{Land} \times \text{Stadt}$, die die Bedingung der WHERE-Klausel erfüllen, wobei nur die Werte der Attribute der SELECT-Klausel angegeben werden.

Algorithmus (nested-loop-Semantik)

```
FOR each Tupel t1 in Relation Land DO
  FOR each Tupel t2 in Relation Stadt DO
    IF Die WHERE-Klausel ist erfüllt nach Ersetzen der Attributnamen
      Land.LCode, Stadt.LCode durch die entsprechenden Werte der gerade betrachteten Tupel t1, t2,
    THEN Bilde ein Antwort-Tupel aus den Werten der in der
      SELECT-Klausel angegebenen Attributen Land.LName, Stadt.SName dieser Tupel t1, t2.
```

... Verwende optionale Korrelationsnamen.

```
SELECT DISTINCT S.SName, L.LName
FROM Stadt S, Land L
WHERE S.LCode = L.LCode
```

..... Anfragen mehrmals über dieselbe Relation.

Lage		
LCode	Kontinent	Prozent
D	Europe	100
F	Europe	100
TR	Asia	68
TR	Europe	32
ET	Africa	90
ET	Asia	10
RU	Asia	80
RU	Europe	20

Bestimme alle Paare von Ländern, die im selben Kontinent liegen.

```
SELECT DISTINCT L1.LCode AS Land1, L2.LCode AS Land2
FROM Lage L1, Lage L2
WHERE L1.Kontinent = L2.Kontinent
AND L1.LCode < L2.LCode
```

(1) FROM Lage L1, Lage L2

Lage L1 (8 Tupel)			Lage L2 (8 Tupel)			
LCode	Kontinent	Prozent		LCode	Kontinent	Prozent
D	Europe	100	×	D	Europe	100
F	Europe	100		F	Europe	100
TR	Asia	68		TR	Asia	68
TR	Europe	32		TR	Europe	32
ET	Africa	90		ET	Africa	90
ET	Asia	10		ET	Asia	10
RU	Asia	80		RU	Asia	80
RU	Europe	20		RU	Europe	20

$L_1 \times L_2$ (64 Tupel)

L1.LCode	L1.Kontinent	L1.Prozent	L2.LCode	L2.Kontinent	L2.Prozent
D	Europe	100	D	Europe	100
D	Europe	100	F	Europe	100
...
D	Europe	100	RU	Asia	80
...
RU	Asia	20	D	Europe	100
...
RU	Europe	20	RU	Europe	20

(2) FROM Lage L1, Lage L2
WHERE L1.Kontinent = L2.Kontinent

$L_1 \times L_2$ (26 Tupel)

L1.LCode	L1.Kontinent	L1.Prozent	L2.LCode	L2.Kontinent	L2.Prozent
D	Europe	100	D	Europe	100
D	Europe	100	F	Europe	100
...
D	Europe	100	RU	Europe	80
...
RU	Europe	20	D	Europe	100
...
RU	Europe	20	RU	Europe	20

(3) FROM Lage L1, Lage L2
WHERE L1.Kontinent = L2.Kontinent
AND L1.LCode < L2.LCode

(9 Tupel)

L1.LCode	L1.Kontinent	L1.Prozent	L2.LCode	L2.Kontinent	L2.Prozent
D	Europe	100	F	Europe	100
ET	Asia	10	TR	Asia	68
RU	Asia	80	TR	Asia	68
D	Europe	100	TR	Europe	32
F	Europe	100	TR	Europe	32
RU	Europe	20	TR	Europe	32
ET	Asia	10	RU	Asia	80
D	Europe	100	RU	Europe	20
F	Europe	100	RU	Europe	20

(4) SELECT DISTINCT L1.LCode AS Land1,
L2.LCode AS Land2
FROM Lage L1, Lage L2
WHERE L1.Kontinent = L2.Kontinent
AND L1.LCode < L2.LCode

(8 Tupel)

Land1	Land2
D	F
ET	TR
RU	TR
D	TR
F	TR
ET	RU
D	RU
F	RU

Auswertung einfacher SQL-Anfragen

SFW-Ausdruck

```
SELECT A1, ..., An (...Attribute der Ergebnisrelation)
FROM R1, ..., Rm (...benötigte Relationen)
WHERE F (...Auswahlbedingung)
```

Algorithmus (nested-loop-Semantik)

```
FOR each Tupel t1 in Relation R1 DO
  FOR each Tupel t2 in Relation R2 DO
    :
    :
  FOR each Tupel tm in Relation Rm DO
    IF Die WHERE-Klausel ist erfüllt nach Ersetzen der Attributnamen
       in F durch die entsprechenden Werte der gerade
       betrachteten Tupel t1, ..., tm.
    THEN Bilde ein Antwort-Tupel aus den Werten der in der
       SELECT-Klausel angegebenen Attributen A1, ..., An
       bezüglich der gerade betrachteten Tupel t1, ..., tm.
```

Verbund (engl. join)

Anfragen mit mehreren Relationen in der FROM-Klausel sind sogenannte *Verbund-Anfragen* (engl. join-queries).

Gib zu jedem Land die zugehörigen Städte an.

```
SELECT DISTINCT S.SName, L.LName
FROM Stadt S, Land L
WHERE S.LCode = L.LCode
```

.... explizit als Verbund:

```
SELECT DISTINCT S.SName, L.LName
FROM Stadt S JOIN Land L
ON S.LCode = L.LCode
```

.... wird Gleichheit über Attributen mit identischen Bezeichnern gefordert redet man von einem *natürlichen Verbund* (engl. natural join) und schreibt kürzer:

```
SELECT DISTINCT S.SName, L.LName
FROM Stadt S NATURAL JOIN Land L
```

.... Spezialfall *kartesisches Produkt*:

```
SELECT DISTINCT S.SName, L.LName
FROM Stadt S CROSS JOIN Land L
```

Sortierung.

Sortiere die Zeilen der Tabelle Stadt aufsteigend nach LCode und für gemeinsame Werte zu LCode absteigend nach dem Breitengrad.

```
SELECT * FROM Stadt
ORDER BY LCode ASC, BGrad DESC
```

2.3 empfohlene Lektüre

SEQUEL: A STRUCTURED ENGLISH QUERY LANGUAGE

by

Donald D. Chamberlin
Raymond F. Boyce

IBM Research Laboratory
San Jose, California

ABSTRACT: In this paper we present the data manipulation facility for a structured English query language (SEQUEL) which can be used for accessing data in an integrated relational data base. Without resorting to the concepts of bound variables and quantifiers SEQUEL identifies a set of simple operations on tabular structures, which can be shown to be of equivalent power to the first order predicate calculus. A SEQUEL user is presented with a consistent set of keyword English templates which reflect how people use tables to obtain information. Moreover, the SEQUEL user is able to compose these basic templates in a structured manner in order to form more complex queries. SEQUEL is intended as a data base sublanguage for both the professional programmer and the more infrequent data base user.

1

¹In: Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control. Kann aus dem Institutsnetz heraus vom ACM-Portal heruntergeladen werden.