6.5 Relationale Datenbank-Anfragesprachen

- 6.5.1: SQL-89
- 6.5.2: SQL-92
- 6.5.3: SQL3
- 6.5.4: QUEL
- 6.5.5: QBE
- 6.5.6: Weitere Anfragesprachen

Relationale Datenbank-Anfragesprachen

Historie

- SQUARE (1975)
- SEQUEL (1975, IBM Research Labs San Jose)
- SEQUEL2 (1976, IBM Research Labs San Jose)
- SQL (1982, IBM)
- ANSI-SQL (1986)
- ISO-SQL (1989, drei Sprachen Level 1, Level 2, +IEF)
- div. Firmenstandards (IBM SQL, OS/2 SQL, X/Open UNIX SQL, ...)
- SQL2 (SQL-92)
- SQL3 (in Bearbeitung)

SQL-Normen

ANSI/ISO Level 1

- keine Nullwerte
- kein !=, kein EXISTS
- kein UNION
- ...

ANSI/ISO Level 2

- wie bisher vorgestellt

ANSI/ISO Level 2 + IEF

- CHECK-Klausel: WHERE-Klausel als Integritätsbedingungen
- Primärschlüssel, Fremdschlüssel

SQL2 (SQL-92)

- neue Datentypen (INTERVAL, DATE, TIME) (in Oracle: nur DATE)
- ALTER TABLE flexibler (auch Spalten löschbar, nicht in Oracle)
- allgemeine Integritätsbedingungen
- andere String-Operationen

- Namen für abgeleitete Spalten
- Join als eigener Operator (nicht in Oracle)
- neben UNION auch INTERSECT und EXCEPT
- Abgeschlossenheit!!

SQL3 (4 Ausbaustufen)

6.5.1 SQL-89

SQL-Anfrage-Teil (SQL-IQL)

- (fast) vollständig
- nicht ganz abgeschlossen
- kann einige Zusatzfunktionen (Arithmetik, Aggregatfunktionen,...)

Überblick: SFW-Block

- SELECT
  - Projektionsliste
  - arithmetische Operationen und Aggregatfunktionen
- FROM
  - zu verwendende Relationen
  - evtl. Umbenennungen (durch Tupelvariable oder 'alias')
- WHERE
  - Selektionsbedingungen
  - Verbundbedingungen
  - geschachtelte Anfragen (wieder ein SFW-Block)

- GROUP BY
  - Gruppierung für Aggregatfunktionen
- HAVING
  - Selektionsbedingung an Gruppen

SQL-89

Syntax allgemein:

SELECT *
FROM relationenliste
[WHERE bedingungen]
[GROUP BY attributliste [HAVING bedingungen]]

Beispiele:

1. SELECT * FROM Artikel;
   gibt gesamte Relation Artikel aus (z.B. 20 Einträge)

2. SELECT * FROM Artikel, fuehrt;
   berechnet kartesisches Produkt (20 * 2 =40 Tupel) statt Verbund (2 Tupel)

3. SELECT * FROM Angestellter eins, Angestellter zwei;
   berechnet kartesisches Produkt von Angestellter mit sich selbst, liefert bei 4 Angestellten 16 Tupel, 4 Spalten (eins.angname, eins.gehalt, zwei.angname, zwei.gehalt)

SQL-89

SELECT

- Syntax: SELECT [DISTINCT] attribute, arithm. ausdrücke, aggr.-funktionen FROM....;

- SELECT abtlgnr FROM fuehrt;   ergibt

abtlgnr
5
5

  keine Projektion: doppelte werden nicht eliminiert (einfacher zu implementieren, wird manchmal bei Aggregatfunktionen benötigt)

- SELECT DISTINCT abtlgnr FROM fuehrt;
  ergibt genau Projektion aus Relationenalgebra

- SELECT Artnr, Preis /1.65 FROM liefert;
  ergibt Preise der Artikel in Dollar

- SELECT avg(Preis) FROM liefert WHERE Art-nr=2;
  ergibt Durchschnitts-Lieferantenpreis für Eier

avg(preis)
1.335

SQL-89

WHERE

- Syntax: SELECT...FROM...WHERE bedingung
- bedingung kann sein:
  - Konstantenselektion (*): attribut {=,!=,<,>,<=,>=} konstante
  - Attributselektion (*): attribut1 {=,!=,<,>,<=,>=} attribut2
  - Verbundbedingung: relation1.attribut = relation2.attribut

    ```
    SELECT Artikel.ArtName FROM Artikel, liefert
    WHERE Artikel.Artnr = liefert.Artnr;
    ```

    gibt alle lieferbaren Artikel (deren Bezeichnung) aus
  - Bereichsselektion (*): attribut BETWEEN konstante1 AND konstante2
  - Ungewißheitsselektion (*): attribut LIKE spezialkonstante
    spezialkonstante kann auch '%' beinhalten, steht für kein oder beliebig viele Zeichen
    SELECT * FROM Artikel WHERE ArtName LIKE '%milch%';
    ergibt

Artnr	ArtName
1	1 L frische Vollmilch
5	0.5 L frische Vollmilch

- Nullselektion (*): attribut IS NULL
- Quantifizierte Bedingungen (*): ALL, ANY, SOME, EXISTS
- Verknüpfungen von Bedingungen (*): NOT, OR, AND

(*): in Oracle entsprechend

SQL-89

Variationen bei Verbunden

- bisher: $\bowtie$ (natürlicher Verbund)
- in SQL-89: Verbundbedingung mit = (Gleichverbund)
- auch möglich: allgemeinerer θ-Verbund durch Verbundbedingung mit θ
- ab SQL-92: auch äußere Verbunde möglich (outer joins)
  - Outer Join: Dangling Tupel beider Operanden werden zusätzlich in das Verbundergebnis übernommen und mit Nullwerten aufgefüllt
  - Left (Right) Outer Join: Nur die Dangling Tupel des linken (rechten) Operanden werden zusätzlich in das Ergebnis übernommen und mit Nullwerten aufgefüllt
- Beispiele (in Oracle):

  Right Outer Join:
  SELECT Artikel.Artnr, LName, Preis
  FROM Artikel, liefert
  WHERE Artikel.Artnr (+) = liefert.Artnr;

Left Outer Join:
SELECT Artikel.Artnr, LName, Preis
FROM Artikel, liefert
WHERE Artikel.Artnr = liefert.Artnr (+);
ergibt

Artnr	LName	Preis
1	Suedmilch	.99
1	Kerry	.99
2	Suedmilch	1.29
2	Kerry	1.38
3	Harry	1.25
4	Bauer Doenges	1.73
5	Bauer Doenges	.64
6		
7		
...		
20		

SQL-89

Schachtelung

- Geschachtelte Bedingung: attribut IN (sfw-block)
- Beispiel:
  SELECT ArtName FROM Artikel
  WHERE Artnr IN (SELECT Artnr FROM liefert);
- verwirklicht ebenfalls Verbund mit anschließender Projektion
- Abarbeitung:
  - zunächst bestimmt System das Ergebnis der inneren SELECT-Anweisung
  - Ergebnis wird hinter IN als Konstantenliste eingesetzt
  - Anfrage wird zu
    SELECT ArtName FROM Artikel
    WHERE Artnr IN (1,2,3,4,5);

SQL-89

- Verzahnung von innerer und äußerer Abfrage: innere Abfrage ist von äußerer abhängig

```
SELECT Preis FROM liefert
WHERE Artnr IN (SELECT Artnr FROM enthaelt,erteilt
          WHERE enthaelt.auftragnr=erteilt.auftragnr
          AND KName=LName);
```

```
SELECT LName FROM liefert X
WHERE X.Artnr IN
          (SELECT Y.Artnr FROM liefert Y
          WHERE Y.Preis=X.Preis
          AND Y.LName != X.LName);
```

ergibt

LName
Suedmilch
Kerry

SQL-89

Gruppierung

- GROUP BY: Gruppierung der Tabelle, Aggregatfunktionen wirken auf einzelnen Gruppen

- Syntax: SELECT...FROM...WHERE...GROUP BY attribute

- Beispiel:
  SELECT count(*),LName FROM liefert
  GROUP BY LName;
  ergibt

count(*)	LName
2	Bauer Doenges
1	Harry
2	Kerry
2	Suedmilch

- HAVING: Selektion von Gruppen

- Syntax: SELECT...FROM...WHERE...
  GROUP BY...HAVING bedingung

- Beispiel:
  SELECT count(*),LName FROM liefert
  GROUP BY LName HAVING count(*) >1;

  liefert nur die Tupel mit Kerry, Suedmilch, Bauer Doenges

- Quicky: was liefert

  SELECT Artnr FROM liefert
  GROUP BY Artnr HAVING avg(Preis) <
  (SELECT avg(Preis) FROM liefert);

SQL-89

Mengenoperation:

- nur UNION, keine Differenz, kein Durchschnitt
  Syntax: sfw-block1 UNION sfw-block2
- SFW-Blöcke müssen gleiche Anzahl von Attributen mit den gleichen Domänen haben
- UNION nur als 'äußerste' Operation erlaubt $\Rightarrow$ SQL-89 nicht abgeschlossen

SQL-89

Quantoren, Mengenvergleiche:

Relationenschemata hier Lieferant={LName, LAdresse},
Artikel={Artnr, Artname, Verkaufspreis},
liefert={LName, Artnr, Lieferpreis, Anzahl}

- Existenzquantor

  ```
  SELECT LAdresse
  FROM Lieferant
  WHERE LName = ANY ( SELECT LName FROM liefert);
  ```

  gibt die Orte aus, aus denen Lieferanten kommen, die bereits Ware geliefert haben

- Allquantor

  ```
  SELECT Preis
  FROM liefert
  WHERE LName='Suedmilch'
  AND Preis >= ALL (SELECT Preis FROM liefert WHERE
  LNme='Suedmilch');
  ```

  gibt den höchsten Preis aus, den ein von 'Suedmilch' geliefertes Produkt hat

- Mengenvergleiche müssen simuliert werden

SQL-89

Aggregatfunktionen, arithm. Operationen:

- auch: Build-In-Funktionen
  - count: Anzahl der Werte einer Spalte
  - count(*): Anzahl der Tupel einer Relation
  - sum: Summe der Werte einer Spalte (auf num. Domänen)
  - avg: arithmetisches Mittel der Werte einer Spalte (auf num. Domänen)
  - max: größter Wert einer Spalte
  - min: kleinster Wert einer Spalte
- bis auf count(*): ALL (Voreinstellung) oder DISTINCT möglich
- Nullwerte werden vorher eliminiert (bis auf count(*))
- Beispiele:
  - Gib die Anzahl aller Kunden aus:
    SELECT count(*) FROM Kunde;
  - Gib die Anzahl der liefernden Lieferanten aus:
    SELECT count(DISTINCT LName) FROM liefert;

- Forts. Beispiele

  – Gib den durchschnittlichen Lieferpreis für Vollmilch aus:

    ```
    SELCT avg(Lieferpreis)
    FROM liefert, Artikel
    WHERE liefert.Artnr=Artikel.Artnr AND Artikel.ArtName
    LIKE '%Vollmilch%';
    ```

- arithmetische Operationen: + (unär, binär),- (unär, binär),*,/

- in Oracle: -/+ als Vorzeichen, *, /, +, -, || (Konkatenation von char-Strings), +, !=, >, <, >=, <=, [NOT] IN, ANY, SOME, ALL, [NOT] BETWEEN x AND y, EXISTS, [NOT] LIKE, IS [NOT] NULL, AND, OR, NOT,
  numerische Funktionen wie ABS, COS, SIN, EXP, SQRT, ROUND,...
  Char/String-Funktionen wie CHR, CONCAT, LOWER, UPPER, ASCII, LENGTH, ...
  Datums-Funktionen wie ADD_MONTH, NEXT_DAY, SYSDATE, TO_DATE,...
  Gruppierungen wie AVG, COUNT, MAX, MIN, SUM, STDDEV, VARIANCE
  sonstige wie GREATEST, LEAST, NVL, UID, USER,...

SQL-89

Verbunde mit derselben Relation

- auch: Self-Join

- eine Anwendung: Arbeiten auf Datenmengen (mit IN, s. vorne)

- andere Anwendung:'Zählen' ohne count

  ```
  SELECT DISTINCT X.LName
  FROM liefert X
  WHERE X.Artnr IN (SELECT Y.Artnr FROM liefert Y
  WHERE Y.LName != X.LName);
  ```

  ohne IN und Schachtelung formulierbar

  ```
  SELECT DISTINCT X.LName
  FROM liefert X, liefert Y
  WHERE X.Artnr=Y.Artnr
  AND X.Lname != Y.Lname;
  ```

SQL-89

Sortierung

- ORDER BY attributliste [ASC | DESC]
- letzte Klausel nach SELECT,FROM,WHERE,GROUP BY, HAVING, UNION
- Beispiel:

```
SELECT Artnr, LName
FROM liefert
ORDER BY LName ASC;
```

- Sortierung aufsteigend: ASC (Voreinstellung), absteigend: DESC
- Sortierung nach mehreren Attributen möglich

```
SELECT Artnr, LName
FROM liefert
ORDER BY LName DESC, Artnr DESC;
```

- Sortierung wird auf Ergebnis der SFW-Anfrage angewendet, daher nicht erlaubt:

```
SELECT Artnr
FROM liefert
ORDER BY LName ASC;
```

- Hat eine Spalte keinen Namen: Spaltennummer verwenden

```
SELECT Artnr, Preis/1.65
FROM liefert
ORDER BY 2;
```