3.17 Zugriffskontrolle

- Datenbanken enthalten häufig vertrauliche Informationen, die nicht jedem Anwender zur Verfügung stehen dürfen.
- Außerdem wird man nicht allen Anwendern dieselben Möglichkeiten zur Verarbeitung der Daten einräumen wollen, da Änderungen der Daten unter Umständen kritisch sind, auch wenn die Daten an sich nicht vertraulich sind.
- Zugriffsrechte können nicht nur einzelnen Benutzern zugewiesen werden, sondern es können Zugriffsrechte auch an *Rollen* gebunden werden.

Rollen

- `CREATE ROLE <Rollenname>`
- `DROP ROLE <Rollenname>`
- `GRANT <Rollenname> TO <Benutzerliste>`
- `REVOKE <Rollenname> FROM <Benutzerliste>`

Benutzer und Objekte

Jeder Benutzer wird durch die spezielle Kennung `PUBLIC` identifiziert; `PUBLIC` erteilte Rechte sind automatisch für alle Benutzer gültig.

- Zugriffskontrolle mittels `GRANT` und `REVOKE`.
- Objekte, die mit Zugriffsrechten versehen werden können, sind unter anderem Tabellen, Spalten, Sichten, Wertebereiche (Domains) und Routinen (Funktionen und Prozeduren).

Rechte

- Die möglichen Rechte sind `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `REFERENCES`, `USAGE`, `TRIGGER` und `EXECUTE`, wobei nicht jedes Recht für jede Art von Objekten angewendet werden kann.
- Syntax:

```
GRANT <Liste von Rechten>
ON <Objekt>
TO <Liste von Benutzern> [WITH GRANT OPTION]

REVOKE [GRANT OPTION FOR] <Liste von Rechten>
ON <Objekt>
FROM <Liste von Benutzern> {RESTRICT | CASCADE}
```

- Mit `GRANT OPTION` erhaltene Rechte können weitergereicht werden.

Verwaltung von Rechten

Der Erzeuger einer Basistabelle, hat zu dieser Tabelle alle für eine Tabelle möglichen Rechte, d.h. die Rechte `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `REFERENCES` und `TRIGGER`.

Beispiel:

Angenommen der Benutzer `Admin` hat alle Tabellen der Mondial-Datenbank erzeugt und besitzt somit alle Rechte. Das Leserecht zu der Tabelle `Land` soll dem Benutzer `PUBLIC` erteilt werden.
Außerdem, sollen den Benutzern `Assistent` und `Tutor` die Rechte zum Lesen, Einfügen, Löschen und Ändern zugeteilt werden in der Weise, dass diese Benutzer diese Rechte auch anderen Benutzer erteilen dürfen.
Schließlich soll der Benutzer `SysProg` die Rechte `REFERENCES` und `TRIGGER` erhalten.

```
GRANT SELECT ON Land TO PUBLIC

GRANT SELECT, INSERT, DELETE, UPDATE
ON Land TO Assistent, Tutor WITH GRANT OPTION

GRANT REFERENCES, TRIGGER
ON Land TO SysProg
```

Bemerkungen

- Die Definition von Fremdschlüsseln, Integritätsbedingungen und Triggern darf nur bei Besitz entsprechender Rechte erlaubt sein, da sonst indirekt auf den Inhalt einer Tabelle geschlossen werden könnte.
- Jeder Benutzer, der ein `SELECT`-Recht besitzt, darf eine Sicht über dieser Tabelle definieren.
- Wird eine Sicht über mehreren Tabellen definiert, dann muss das `SELECT`-Recht zu allen diesen Tabellen zugeteilt sein. Weitere Rechte zu der Sicht existieren nur dann, wenn diese Rechte auch für alle der Sicht zugrunde liegenden Tabellen besessen werden.

zum REVOKE

Ein Recht R heißt *verlassen*, wenn das Recht, das für seine Zuteilung erforderlich war, zurückgezogen wurde und keine weitere Zuteilung von R vorgenommen wurde, deren erforderlichen Rechte noch existieren.

- Die Option `CASCADE` veranlaßt zusätzlich zu der Rücknahme des in der `REVOKE`-Klausel benannten Rechts auch die Zurücknahme aller verlassenen Rechte.
- die Option `RESTRICT` führt zum Abbruch der `REVOKE`-Anweisung, wenn verlassene Rechte resultieren.

Benutzer Assistent teilt dem Benutzer Tutor ein INSERT-Recht für Land zu.

```
GRANT INSERT ON Land TO Tutor
```

Es folgen eine Reihe von durch Benutzer Admin vorgenommene REVOKE-Anweisungen.

```
REVOKE INSERT ON Land FROM Tutor
```

Tutor behält das Recht, da er es unabhängig auch von Assistent erhielt.

```
REVOKE INSERT ON Land FROM Assistent CASCADE
```

Jetzt verliert sowohl Assistent, als auch Tutor das Recht.

Angenommen, Admin führt anstatt der letzten Anweisung

```
REVOKE GRANT OPTION FOR INSERT ON Land
FROM Assistent CASCADE
```

aus.

Jetzt behält Assistent das INSERT-Recht, jedoch Tutor verliert es, da die Erlaubnis für die Vergabe des Rechts an ihn zurückgezogen wurde.

3.18 Verschiedenes

ORA_HASH (Oracle Database SQL Language Reference 11g Release 1 (11.1))

ORA_HASH (expr [,max_bucket] [,seed_value])

- ORA_HASH is a function that computes a hash value for a given expression.
- The expr argument determines the data for which you want Oracle Database to compute a hash value.
- The optional max_bucket argument determines the maximum bucket value returned by the hash function. You can specify any value between 0 and 4294967295. The default is 4294967295.
- The optional seed_value argument enables Oracle to produce many different results for the same set of data.
- The function returns a NUMBER value.

Test auf Gleichheit von Tabellen - nur als notwendige Bedingung, garantiert korrekt, d.h. wenn die Summen der Hashs verschieden sind, dann sind auch die Tabellen verschieden.

```
SELECT DISTINCT LCode FROM Mitglied M WHERE
  (SELECT SUM(ORA_HASH(Organisation||Art)) FROM Mitglied WHERE LCode = M.LCode) =
  (SELECT SUM(ORA_HASH(Organisation||Art)) FROM Mitglied WHERE LCode = 'A')
```

In-line/temporary-table Sichten mittels WITH

Vergleiche:

```
SELECT DISTINCT S.LCode FROM Stadt S
WHERE (SELECT AVG(Einwohner) FROM Stadt WHERE LCode = S.LCode) =
      (SELECT MIN(Einw) FROM (SELECT AVG(Einwohner) AS Einw
                              FROM Stadt GROUP BY Lcode));
```

mit

```
WITH
avgEinwohner AS (SELECT LCode, AVG(Einwohner) AS Einw
                 FROM Stadt GROUP BY Lcode),
minEinwohner AS (SELECT MIN(Einw) as minE FROM avgEinwohner)
SELECT LCode FROM avgEinwohner S
WHERE S.Einw = (SELECT minE FROM minEinwohner);
```

WITH führt zu einer klareren Struktur,

verwendet wie lokal definierte virtuelle Sicht (*in-line view*), bzw. wie eine materialisierte Sicht (*temporäre Tabelle*).

FROM DUAL

In manchen Situationen ist das Resultat einer SFW-Anfrage unabhängig von den Tabellen der FROM-Klausel. Um der SFW-Syntax zu genügen verwende man dann als Tabelle eine Dummy-Tabelle (in Oracle existent mit Namen DUAL), die genau eine Zeile enthält.

```
SELECT (ln(50) * sin(300)) / tan(0.5) FROM DUAL;


SELECT (SELECT min(Einwohner) FROM Stadt) AS Klein,
       (SELECT max(Einwohner) FROM Stadt) AS Gross
FROM DUAL;
```

DESCRIBE liefert die Attribute einer Tabelle inklusive Typen.

```
DESCRIBE Stadt;
```

ROLLUP und CUBE

SALES

Model	Year	Color	Sales
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	blue	55
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	blue	39

$\Longrightarrow$ *ROLLUP*

ROLLUP

Model	Year	Color	Sales
Chevy	1990	blue	62
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	ALL	154
Chevy	1991	blue	49
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	ALL	198
Chevy	1992	blue	71
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	ALL	156
Chevy	ALL	ALL	508
Ford	1990	blue	63
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	ALL	189
Ford	1991	blue	55
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	ALL	116
Ford	1992	blue	39
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	ALL	128
Ford	ALL	ALL	433
ALL	ALL	ALL	941

SALES

Model	Year	Color	Sales
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	blue	55
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	blue	39

$\Longrightarrow$ *CUBE*

DATA CUBE

Model	Year	Color	Sales
Chevy	1990	blue	62
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	ALL	154
Chevy	1991	blue	49
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	ALL	198
Chevy	1992	blue	71
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	ALL	156
Chevy	ALL	blue	182
Chevy	ALL	red	90
Chevy	ALL	white	236
Chevy	ALL	ALL	508
Ford	1990	blue	63
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	ALL	189
Ford	1991	blue	55
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	ALL	116
Ford	1992	blue	39
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	ALL	128
Ford	ALL	blue	157
Ford	ALL	red	143
Ford	ALL	white	133
Ford	ALL	ALL	433
.	.	.	
.	.	.	
.	.	.	
.	.	.	
.	.	.	
.	.	.	
ALL	1990	blue	125
ALL	1990	red	69
ALL	1990	white	149
ALL	1990	ALL	343
ALL	1991	blue	106
ALL	1991	red	104
ALL	1991	white	110
ALL	1991	ALL	314
ALL	1992	blue	110
ALL	1992	red	58
ALL	1992	white	116
ALL	1992	ALL	284
ALL	ALL	blue	339
ALL	ALL	red	233
ALL	ALL	white	369
ALL	ALL	ALL	941

Online Analytical Processing (OLAP) mittels ROLLUP und CUBE

Datawarehouse-Anwendungen auf ausgelagerten typischerweise historischen Daten - im Unterschied zu *Online Transaction Processing* (OLTP), Anwendungen auf den aktuellen operationalen Daten.

```
SELECT Model, Year, Color, sum(Sales)
FROM SALES
GROUP BY ROLLUP(Model, Year, Color)
```

```
SELECT Model, Year, Color, sum(Sales)
FROM SALES
GROUP BY CUBE(Model, Year, Color)
```

3.19 empfohlene Lektüre

Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals*

JIM GRAY Gray@Microsoft.com
SURAJIT CHAUDHURI SurajitC@Microsoft.com
ADAM BOSWORTH AdamB@Microsoft.com
ANDREW LAYMAN AndrewL@Microsoft.com
DON REICHART DonRei@Microsoft.com
MURALI VENKATRAO MuraliV@Microsoft.com
Microsoft Research, Advanced Technology Division, Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

FRANK PELLOW Pellow@vnet.IBM.com
HAMID PIRAHESH Pirahesh@Almaden.IBM.com
IBM Research, 500 Harry Road, San Jose, CA 95120

Editor: Usama Fayyad

Received July 2, 1996; Revised November 5, 1996; Accepted November 6, 1996

Abstract. Data analysis applications typically aggregate data across many dimensions looking for anomalies or unusual patterns. The SQL aggregate functions and the GROUP BY operator produce zero-dimensional or one-dimensional aggregates. Applications need the *N*-dimensional generalization of these operators. This paper defines that operator, called the **data cube** or simply **cube**. The cube operator generalizes the histogram, cross-tabulation, roll-up, drill-down, and sub-total constructs found in most report writers. The novelty is that cubes are relations. Consequently, the cube operator can be imbedded in more complex non-procedural data analysis programs. The cube operator treats each of the *N* aggregation attributes as a dimension of *N*-space. The aggregate of a particular set of attribute values is a point in this space. The set of points forms an *N*-dimensional cube. Super-aggregates are computed by aggregating the *N*-cube to lower dimensional spaces. This paper (1) explains the cube and roll-up operators, (2) shows how they fit in SQL, (3) explains how users can define new aggregate functions for cubes, and (4) discusses efficient techniques to compute the cube. Many of these features are being added to the SQL Standard.

[1]

[1] In: Data Mining and Knowledge Discovery 1, 1997. Kann gegoogelt werden.