Can Türker

SQL:2003 - Was dürfen wir erwarten?

1 Einleitung

Kaum hat sich der aktuelle Datenbankstandard SQL:1999 ein wenig etabliert, da steht schon der Nachfolger SQL:2003 mit den folgenden Spezifikationsteilen zur Publikation (Ende 2003?) bereit:

- Part 1: Framework
- Part 2: Foundation
- Part 3: Call Level Interface
- Part 4: Persistent Stored Modules
- Part 9: Management of External Data
- Part 10: Object Language Binding
- Part 11: Information and Definition Schema
- Part 13: SQL Routines and Types Using the Java Programming Language
- Part14: XML-related Specifications

Im Vergleich zu SQL:1999 sind folgende Neuerungen in SQL:2003 vorgesehen:

- Part 2 enthält einige kleinere Erweiterungen des Datenmodells und der damit verbundenen Sprachen.
- SQL:1999's Part 5: Host Language Binding wird in den neuen Part 2 verschoben. Die Java-Anbindung bleibt weiterhin separat in Part 10 definiert.
- Die Sprachanbindungen werden um Konstrukte für die Handhabung der Datenmodellerweiterungen ergänzt.
- Die Meta-Tabellen werden von Part 2 in einem eigenen Part 11 ausgelagert.
- Part 14 ist mit der Verknüpfung von SQL und XML die spannendeste Neuheit von SQL:2003.

Die Parts 9, 10 (ehemals SQLJ Part 0) und 13 (ehemals SQLJ Part 1 und 2) wurden erst in den Jahren 2000 bis 2002 als Zusatz von SQL:1999 veröffentlicht.

2 SQL:2003-Erweiterungen

Part 2 definiert folgende Erweiterungen:

- *Basisdatentyp* **BIGINT**
- *Multimengen* (**MULTISET**)
- *Generierte Spalten*
- *Sequenzgeneratoren*
- *Identitätsspalten*
- **MERGE**

Part 14 bietet darüber hinaus einen neuen *Basisdatentyp* **XML**.

Multimengen

Eine *Multimenge* ist eine ungeordnete, unbeschränkte Kollektion von Werten. Multimengen werden im Vergleich zu Arrays mit einer Vielzahl typspezifischer Funktionen und Prädikate bereitgestellt:

- **MULTISET**() generiert eine leere Multimenge.
- **MULTISET**(<Werteliste>) erzeugt aus einer Werteliste eine Multimenge.
- **MULTISET**(<Unteranfrage>) konvertiert eine Tabelle in eine Multimenge.
- **UNNEST**(<M>) wandelt eine Multimenge in eine virtuelle Tabelle um.
- **Element**(<M>) liefert das Element einer einelementigen Multimenge M.
- **SET**(<M>) entfernt die Duplikate aus der Multimenge M.
- <M1> **MULTISET UNION** <M2> vereinigt die Multimengen M1 und M2.
- <M1> **MULTISET EXCEPT** <M2> bildet die Differenz zwischen M1 und M2.
- <M1> **MULTISET INTERSECT** <M2> berechnet den Durchschnitt.
- **CARDINALITY** (<M>) liefert die Anzahl der Elemente einer Multimenge M.
- <M> **IS A SET** überprüft, ob die Multimenge M Duplikate enthält.
- <W> **MEMBER** <M> testet, ob der Wert W in der Multimenge M enthalten ist.
- <M1> **SUBMULTISET** <M2> testet, ob M1 eine Teilmultimenge von M2 ist.

Neu sind ebenfalls die multimengenspezifischen Aggregatfunktionen:

- **COLLECT** *erzeugt eine Multimenge aus den Werten einer Gruppe.*
- **FUSION** *vereinigt die Multimengen einer Gruppe zu einer Multimenge.*
- **INTERSECTION** *bildet den Schnitt aus den Multimengen einer Gruppe.*

Funktionen zur Konvertierung zwischen Multimengen und Arrays fehlen jedoch. Erfreulicherweise erlaubt SQL:2003 eine orthogonale Schachtelung von Kollektionen.

Generierte Spalten

Ein *generierte Spalte* ist eine Spalte, deren Wert aus Werten anderer Spalten derselben Zeile berechnet werden. Eine generierte Spalte wird folgt definiert:

```
<Spaltenname> GENERATED ALWAYS
(<Wertausdruck>)
```

Die Variablen im Wertausdruck sind auf nicht-generierte Spalten derselben Zeile sowie Funktionen beschränkt, die weder SQL-Anfragen noch DML-Anweisungen enthalten.

Sequenzgeneratoren

Ein *Sequenzgenerator* erzeugt Sequenzen von numerischen Werten, z.B. für die Vergabe künstlicher Schlüsselwerte:

```
CREATE SEQUENCE <Sequenzname>
AS <Typname>
[START WITH <Wert>]
[INCREMENT BY <Wert>]
[NO MINVALUE | MINVALUE <Wert>]
[NO MAXVALUE | MAXVALUE <Wert>]
[NO CYCLE | CYCLE]
```

Die Angabe der Sequenzoptionen erfolgt reihenfolgeunabhängig. Ist der Inkrementwert negativ, erfolgt eine Dekrementierung. Wird beim Inkrementieren der Maximalwert (bzw. beim Dekrementieren der Minimalwert) erreicht, entscheidet die Cycle-Klausel über den nächsten Schritt. Ist **NO CYCLE** definiert, wird die Generierung mit dem Setzen einer Ausnahmebedingung beendet. **CYCLE** dagegen setzt die Generierung mit dem Minimalwert fort.

Der Zugriff auf den nächsten Wert einer Sequenz erfolgt mit dem Ausdruck **NEXT VALUE FOR** <Sequenzname>.

Identitätsspalten

Eine *Identitätsspalte* ermöglicht die automatische Generierung von Schlüsseln mit Hilfe eines impliziten Sequenzgenerators. Der »Typ« einer solchen Spalte wird folgenderweise angegeben:

```
GENERATED {ALWAYS | BY DEFAULT}
AS IDENTITY (<Sequenzoptionen>)
```

ALWAYS schliesst ein Update der Spalte aus. **BY DEFAULT** dagegen erlaubt Inserts und Updates.

Identitätsspalten sind nicht mit OID-Spalten zu verwechseln:

- OID-Spalten besitzen eindeutige, aber *unveränderbare* Werte, auf sich Referenzspalten beziehen können. Identitätsspalten sind nicht über Referenzspalten referenzierbar.
- Eine OID-Spalte kommt nur in *typisi-*

erten Tabellen vor, während eine Identitätsspalte auch in gewöhnlichen *Tupeltabellen* vorkommen kann.
- Eine Tabelle darf maximal eine Identitätspalte enthalten. Dasselbe gilt für eine OID-Spalte. Typisierte Tabellen können beide Spaltenarten besitzen.

Die MERGE-Anweisung

Die Merge-Anweisung steht für eine Kombination von mehreren Update- und Insert-Anweisungen, die ausgewählte Spaltenwerte bzw. Zeilen aus einer Tabelle in eine andere Tabelle übernimmt:

```
MERGE INTO <Zieltabellenname>
[AS <Korrelationsname>]
USING <Tabellenreferenz>
ON <Verbundbedingung>
WHEN
[MATCHED THEN SET <Spaltenzuweisung>]
[NOT MATCHED THEN
  INSERT [(<Spalten>)] VALUES (<Werte>)]
```

Mindestens eine der beiden »Matched-Klauseln« muss angegeben werden. Je nachdem, ob die Verbundbedingung erfüllt ist oder nicht, wird die jeweilige Klausel ausgeführt.

Der XML-Datentyp

SQL/XML [3] spezifiziert den Basisdatentyp **XML** und entsprechende Abbildungen zwischen SQL und XML.

SQL-Character-Sets	XML-Unicode
SQL-Identifiers	XML-Names
SQL-Datentypen	XML-Schema-Datentypen
SQL-Werte	XML-Werte
SQL-Tabellen, SQL-Schemata, SQL-Kataloge	XML-Dokumente (Daten) und XML-Schema-Dokumente (Meta-Daten)

Zum Erstellen und Bearbeiten von XML-Dokumenten stehen folgende Funktionen zur Verfügung:
- **XMLELEMENT** erzeugt ein XML-Element aus einer Werteliste; **XMLATTRIBUTES** erzeugt ein XML-Attribut.
- **XMLFOREST** erzeugt einen Wald von XML-Elementen aus einer beliebigen Werteliste.
- **XMLCONCAT** konkateniert XML-Elemente zu einem Wald.
- **XMLGEN** generiert ein XML-Element basierend auf einer XQuery.
- **XMLAGG** aggregriert XML-Elemente einer Gruppe.

Anfragen auf XML-Dokumenten sind über eine Kombination von SQL und XQuery 1.0 [4] möglich. Hier sind die Spezifikationsarbeiten noch nicht abgeschlossen. Es werden Funktionen und Prädikate wie **XMLEXTRACT** und **XMLEXISTS** erwartet, die auf XML-Werten operieren und SQL-Werte zurückliefern.

Die folgenden Beispiele illustrieren einige der neuen SQL:2003-Konstrukte:

```
CREATE TABLE Mitarbeiter (
  MNr        INTEGER
             GENERATED ALWAYS
             AS IDENTITY
             START WITH 100000
             INCREMENT 1
             MINVALUE 100000
             MINVALUE 999999
             CYCLE,
  Gehalt     DECIMAL(9,2),
  Bonus      DECIMAL(9,2),
  Total      GENERATED ALWAYS AS
             (Salary + Bonus),
  Bewerbung  XML);

SELECT XMLELEMENT(
             NAME "income",
             XMLATTRIBUTES(MNr),
             Total)
FROM Mitarbeiter;
```

Diese Anfrage liefert eine einspaltige Tabelle der Form:

```
<income MNR="100002">3500</income>
<income MNR="100005">4750</income>
```

3 SQL-Konformität

Die Tabelle unten gibt eine grobe Übersicht über die Unterstützung der neuen SQL:2003-Konstrukte in einigen ausgewählten Datenbanksystemen. Ein Plus bedeutet, dass das Konstrukt analog bzw. ähnlich in dem jeweiligen SQL-Dialekt enthalten ist. (+) kennzeichnet Lösungen, die syntaktisch und konzeptionell stark vom Standard abweichen.

Bei näherer Betrachtung der SQL-Konstrukte ist der Einfluss von Oracle und IBM auf den Standard unverkennbar. Das Sequenz-Konstrukt, die Merge-Anweisung und der XML-Datentyp stammen von Oracle, generierte Spalten und Identitätsspalten von IBM.

SQL:2003	Oracle9i	DB2 V7.2	Informix 9.3	Postgres 7.2	MSSQL 2000
BIGINT	-	+	-	+	+
MULTISET	-	-	+	-	-
Generierte Spalten	-	+	-	-	-
Sequenzgenerator	+	+	(+)2	+	-
Identitätsspalten	-	+	-	-	+
MERGE	+	-	-	-	-
Basisdatentyp XML	+	(+)1	-	-	-

1 DB2-XML-Extender-Datentypen (kein Basisdatentyp), 2 Informix-SERIAL-Datentyp

4 Fazit

Der zukünftige Standard SQL:2003 sieht im Vergleich zu SQL:1999 nur wenige, aber durchaus nützliche Neuerungen vor. Diese »Zurückhaltung« ist zu begrüssen, da die aktuellen Datenbanksysteme noch weit davon entfernt, die erweiterten Konstrukte von SQL:1999 umzusetzen. Eine Vergrösserung der Diskrepanz zwischen Standard und gängigen SQL-Dialekten erschwert die Motivation und das Vermitteln von »Advanced« SQL-Kenntnissen. Jedenfalls spricht die geringe Anzahl von Lehrveranstaltungen, die über den relationalen Teil von SQL hinausgehen, nicht dafür, dass die neueren SQL-Konzepte eine weite Verbreitung finden.

Literatur

[1] *ANSI/ISO/IEC9075-2:1999*. ISO International Standard: Database Language - SQL - Part 2: Foundation (SQL/Foundation), Sept. 1999.

[2] *ANSI/ISO/IEC9075-2:200x*. ISO Working Draft: Database Language - SQL - Part 2: Foundation (SQL/Foundation), August 2002.

[3] *ANSI/ISO/IEC9075-14:200x*. ISO Working Draft: Database Language - SQL - Part 14: XML-related Specifications (SQL/XML), August 2002.

[4] W3C Working Draft: XQuery 1.0: An XML Query Language, August 2002.

Dr. Can Türker ist Oberassistent in der Datenbankgruppe der ETH Zürich. Seine Forschung befasst sich mit der Integration und Mobilität von Informationssystemen. Er ist Koautor des Lehrbuchs »Objektdatenbanken« und Verfasser von Überblicksartikeln über SQL:1999 und SQL-Dialekte.

tuerker@inf.ethz.ch