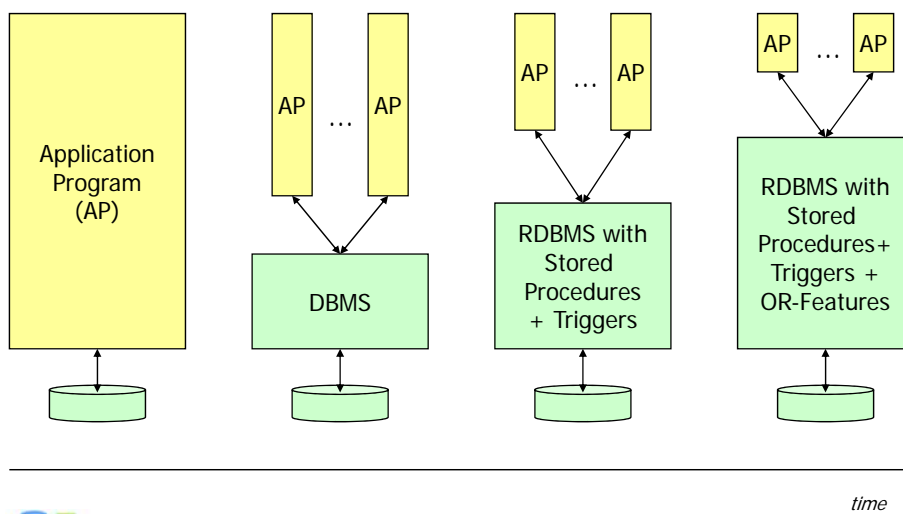


Overview



Recent Developments for Data Models

DBMS Tasks – Historic Development



Major Extensions in SQL:1999 and 2003

- Mechanism for "users" to extend the database with application "objects" (specific types and their behavior - functions/methods)
 - User Defined Types (UDTs): Text, Image, CAD/CAM Drawing, Video ...
 - User Defined Functions (UDFs): Contains, Display, Rotate, Play, ...
- Support for storage/manipulation of large data types
 - Large Object Support (LOBs): Binary, Character
- Mechanism to improve the DB integrity and to allow checking of business rules inside the DBMS
 - Triggers: Auditing, Cross-Referencing, Alerts ...
- Means to express complex data relationships such as hierarchies, bills-of-material, travel planning ...
 - Recursion, Common Table Expressions, ...
- Support for data analysis, online analytic processing
 - CUBE, ROLLUP, SQL Windows, ...
- XML support
 - XML data type, publishing functions, mapping, ...

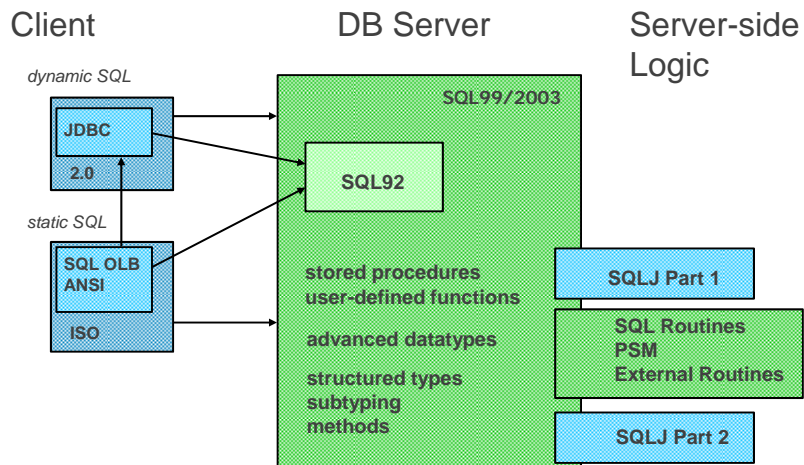


Object-Relational Support

- Major goals
 - support management of complex business objects
 - provide extensibility for defining new, complex data types and behavior
- Key features
 - Large Objects (LOBs)
 - Binary, Character
 - User-Defined Data Types
 - Distinct types, Structured types
 - Type Constructors
 - Row types, Reference types
 - Collection Types
 - Arrays, Multisets
 - User-Defined Methods, Functions, and Procedures
 - Typed tables and views
 - Table hierarchies, View hierarchies (object views)



The "Big Picture"



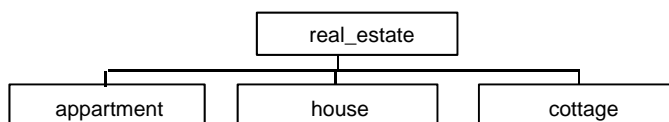
© Prof. Dr.-Ing. Stefan Deßloch

5

Recent Developments for Data Models

Subtyping and Inheritance

- Structured types can be a subtype of another UDT
- UDTs inherit structure (attributes) and behavior (methods) from their supertypes
- Example
 - CREATE TYPE real_estate ... NOT FINAL
 - CREATE TYPE apartment UNDER real_estate ... NOT FINAL
 - CREATE TYPE house UNDER real_estate ... NOT FINAL



© Prof. Dr.-Ing. Stefan Deßloch

6

Recent Developments for Data Models

Structured Types – Nesting and Behavior

```

CREATE TYPE envelope (
  xmin  INTEGER,
  ymin  INTEGER,
  xmax  INTEGER,
  ymax  INTEGER);

CREATE TYPE geometry (
  gtype  INTEGER,
  refsystem  INTEGER,
  tolerance  FLOAT,
  area  FLOAT,
  length  FLOAT,
  mbr  envelope,
  numparts  INTEGER,
  numpoints  INTEGER,
  points  BLOB(1m),
  zvalue  BLOB(500k),
  measure  BLOB(500k));

CREATE TYPE point UNDER geometry;
CREATE TYPE line UNDER geometry;
CREATE TYPE polygon UNDER geometry;

CREATE FUNCTION distance
(s1 geometry, s2 geometry)
RETURNS BOOLEAN
EXTERNAL NAME
'usr/lpp/db2se/gis!shapedist'
...

CREATE FUNCTION within
(s1 geometry, s2 geometry)
RETURNS BOOLEAN
EXTERNAL NAME
'usr/lpp/db2se/gis!shapewithin'
...

```



© Prof. Dr.-Ing. Stefan Deßloch

7

Recent Developments for Data Models

Structured Types as Column Types

```

CREATE TABLE customers (
  cid  INTEGER,
  name VARCHAR(20),
  income  INTEGER,
  addr  CHAR(20)
  loc  point);

CREATE TABLE stores (
  sid  INTEGER,
  name VARCHAR(20),
  addr  CHAR(20),
  loc  point,
  zone  polygon);

CREATE TABLE sales (
  sid  INTEGER,
  cid  INTEGER,
  amount  INTEGER);

```



© Prof. Dr.-Ing. Stefan Deßloch

8

Recent Developments for Data Models

Structured Types as Column Types

```
SELECT * FROM stores s, customers c
WHERE within(c.loc, s.zone)=1
    or distance(c.loc, s.loc)<100
ORDER BY s.name, c.name;
```

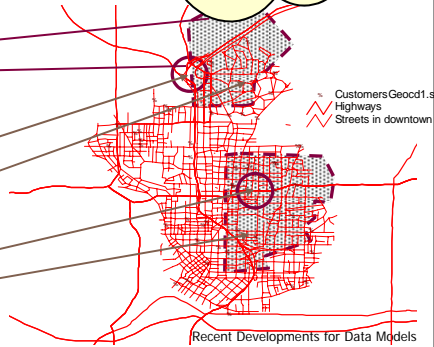
CUSTOMERS

CID	NAME	INCOME	ADDR	LOC

STORES

SID	NAME	ADDR	LOC	ZONE

"Tell me all the information I have about each customer who either lives within a stores' zone or within 100 miles of the store."



© Prof. Dr.-Ing. Stefan Deßloch

9

Recent Developments for Data Models

Structured Types as Row Types: Typed Tables

- Structured types can be used to define typed tables
 - Attributes of type become columns of table
 - Plus one column to define REF value for the row (object id)

```
CREATE TYPE real_estate AS
(owner                REF (person),
 price               money,
 rooms               INTEGER,
 size                 DECIMAL(8,2),
 location             address,
 text_description     text,
 front_view_image     bitmap,
 document             doc) NOT FINAL
```

```
CREATE TABLE properties OF real_estate
(REF IS oid USER GENERATED)
```



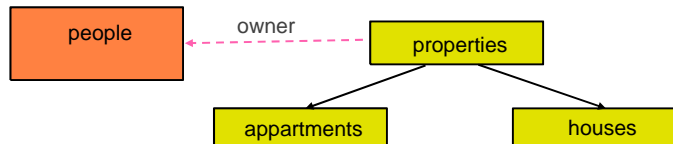
© Prof. Dr.-Ing. Stefan Deßloch

10

Recent Developments for Data Models

Subtables: Table Hierarchies

- Typed tables can have subtables
 - Inherit columns, constraints, triggers, ... from the supertable



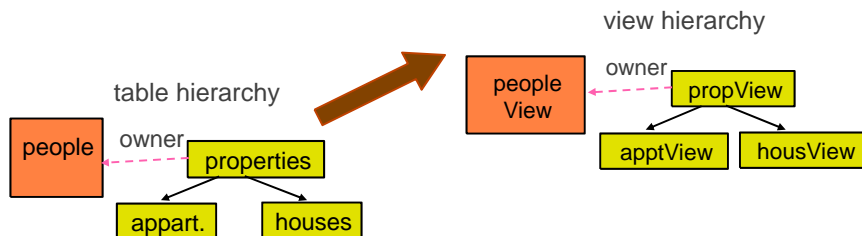
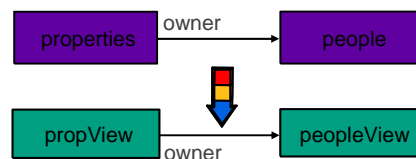
© Prof. Dr.-Ing. Stefan DeBloch

11

Recent Developments for Data Models

Object Views

- Views have been extended to support
 - Typed views
 - View hierarchies
 - References on base tables can be mapped to references on views



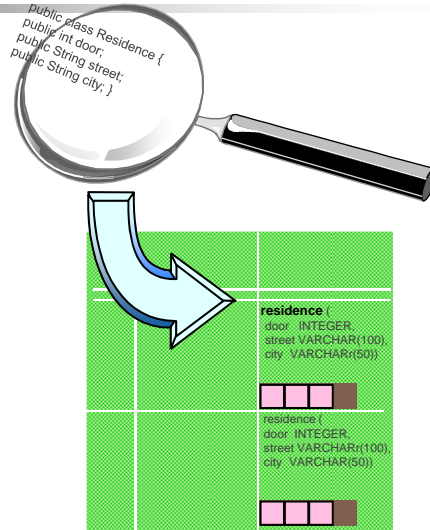
© Prof. Dr.-Ing. Stefan DeBloch

12

Recent Developments for Data Models

Mapping Java Objects to Structured Types

- Support built into the DBMS
- Very flexible
 - DB understands internal structure of type
 - Based on SQL type system
 - Client applications written in other programming languages are supported
 - Can be used to define row types/typed tables
 - DB functions/methods can be implemented in other programming language
- Potential for better performance
- Requires conversion (Java <-> SQL)



© Prof. Dr.-Ing. Stefan Deßloch

13

Recent Developments for Data Models

SQLJ/JRT

- SQL Types using the Java™ Programming Language
- Use of Java classes to define SQL types
 - Can be mapped to structured types or "native" Java types (blobs)
 - Can be used to define columns in tables
 - Can be used to define SQL99 tables (structured types)
- Mapping of object state and behavior
 - Java methods become SQL99 methods on SQL type
 - Java methods can be invoked in SQL statements
- Automatic mapping to Java object on fetch and method invocation
 - Java Serialization
 - JDBC 2.0 SQLData interface



© Prof. Dr.-Ing. Stefan Deßloch

14

Recent Developments for Data Models

Mapping Java Classes to SQL

- Described using extended CREATE TYPE syntax
 - DDL statement, or
 - Mapping description in the deployment descriptor
- Supported Mapping

Java	SQL
class	user-defined (structured) type
member variable	attribute
method	method
constructor	constructor method
static method	static method
static variable	static observer method

- SQL constructor methods
 - Have the same name as the type for which they are defined
 - Are invoked using the NEW operator (just like in Java)
- SQL does not know static member variables
 - Mapped to a static SQL method that returns the value of the static variable
 - No support for modifying the static variable

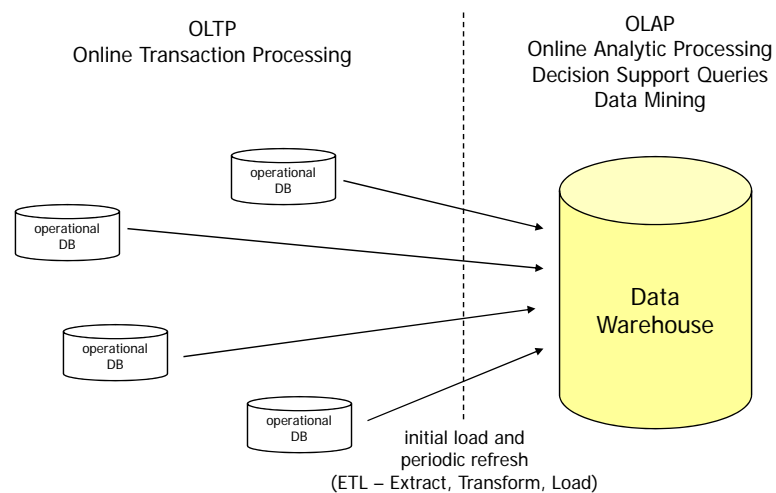


© Prof. Dr.-Ing. Stefan DeBöck

15

Recent Developments for Data Models

Business Intelligence, Online Analytic Processing



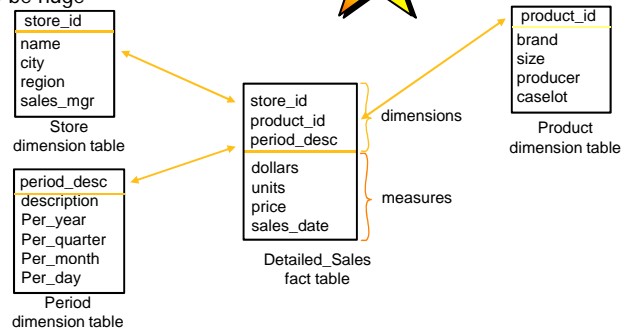
© Prof. Dr.-Ing. Stefan DeBöck

16

Recent Developments for Data Models

OLAP Schema

- Typically uses a "STAR" structure
 - Dimension tables tend to be small
 - Fact table tends to be huge



```
CREATE VIEW Sales AS
(SELECT ds.*, YEAR (sales_date) AS year, MONTH (sales_date) AS month, DAY (sales_date) AS day
FROM (Detailed_Sales NATURAL JOIN Store NATURAL JOIN Product NATURAL JOIN Period) ds
```



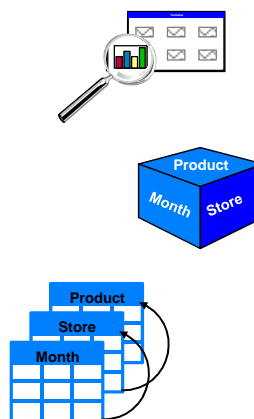
© Prof. Dr.-Ing. Stefan Deßloch

17

Recent Developments for Data Models

SQL99 OLAP SQL Extensions

- Extension to GROUP BY clause
- Produces "super aggregate" rows
- ROLLUP equivalent to "control breaks"
- CUBE equivalent to "cross tabulation"
- GROUPING SETS equivalent to multiple GROUP BYs
- Provides "data cube" collection capability
 - Often used with data visualization tool



© Prof. Dr.-Ing. Stefan Deßloch

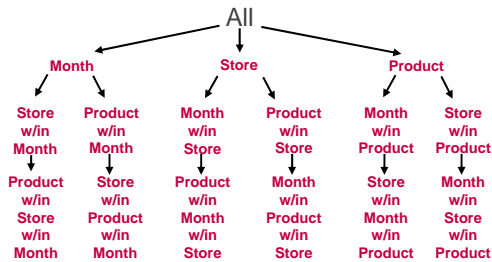
18

Recent Developments for Data Models

CUBE

- Extends grouping semantics to produce multidimensional grouping and "subtotal" rows
 - Produces "regular" grouped rows
 - Produces same groupings reapplied down to grand total
 - Produces additional groupings on all variants of the CUBE clause

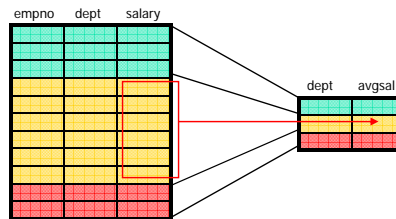
```
SELECT month, city, product_id, SUM(units)
FROM Sales
WHERE year = 1998
GROUP BY CUBE (month, city, product.id)
```



Windows in SQL

- Set functions (aggregate functions)

```
SELECT dept, AVG(salary) AS avgsal
FROM Employees
GROUP BY dept
```



- Windowed Table Functions

```
SELECT dept, empno, salary,
AVG(salary) OVER(
PARTITION BY dept
ORDER BY age
ROWS
BETWEEN 2 PRECEDING
AND 2 FOLLOWING)
AS c-avg
FROM Employees
```



XML Origin and Usages

- Defined by the WWW Consortium (W3C)
- Originally intended as a document markup language, not a database language
 - Documents have tags giving extra information about sections of the document
 - For example:
 - `<title> XML </title>`
 - `<slide> XML Origin and Usages </slide>`
- Derived from SGML (Standard Generalized Markup Language)
 - standard for document description
 - enables document interchange in publishing, office, engineering, ...
 - main idea: separate form from structure
- XML is simpler to use than SGML
 - roughly 20% complexity achieves 80% functionality



Why is XML Important?

- Exchanging data among different systems or applications
 - neutral, flexible format
 - self-describing
- Wide range of support for processing XML
 - free/open source
 - major industry vendors
- Business reasons
 - increased agility, flexibility through easier adaptability of IT infrastructure
 - economic advantages
 - wide range of support
 - service-oriented architectures (SOA) promise better interoperability, software reuse
 - cost savings over proprietary electronic data interchange solutions
 - regulatory requirements and industry-specific initiatives
 - capture, maintain, monitor, store electronic transactions
 - "vertical" standards for data exchange



Describing XML Data: XML Schema

- XML Schema is close to the general understanding of a (database) schema
- XML Schema supports
 - Typing of values
 - E.g. integer, string, etc
 - Constraints on min/max values
 - Typed references (for ID and IDREFS)
 - User defined types
 - Specified in XML syntax (unlike DTDs)
 - Integrated with namespaces
 - Many more features
 - List types, uniqueness and foreign key constraints, inheritance ..



XQuery

- XQuery is a general purpose query language for XML data
- Currently being standardized by the World Wide Web Consortium (W3C)
- Alpha version of XQuery engine available free from Microsoft
- XQuery is derived from
 - the **Quilt** ("Quilt" refers both to the origin of the language and to its use in "knitting" together heterogeneous data sources) query language, which itself borrows from
 - **XPath**: a concise language for navigating in trees
 - **XML-QL**: a powerful language for generating new structures
 - **SQL**: a database language based on a series of keyword-clauses: SELECT - FROM - WHERE
 - **OQL**: a functional language in which many kinds of expressions can be nested with full generality



XQuery: FLWOR Syntax

- XQuery is a functional language
 - Every query is an expression
 - Expressions can be nested with full generality:
 - XPath expressions
 - Element constructors
 - FLWR expressions
- Simple FLWOR expression in XQuery
 - Find all accounts with balance > 400, with each result enclosed in an <account-number> .. </account-number> tag

```
for   $x in /bank-2/account
let   $acctno := $x/@account-number
where $x/balance > 400
return <account-number> { $acctno } </account-number>
```



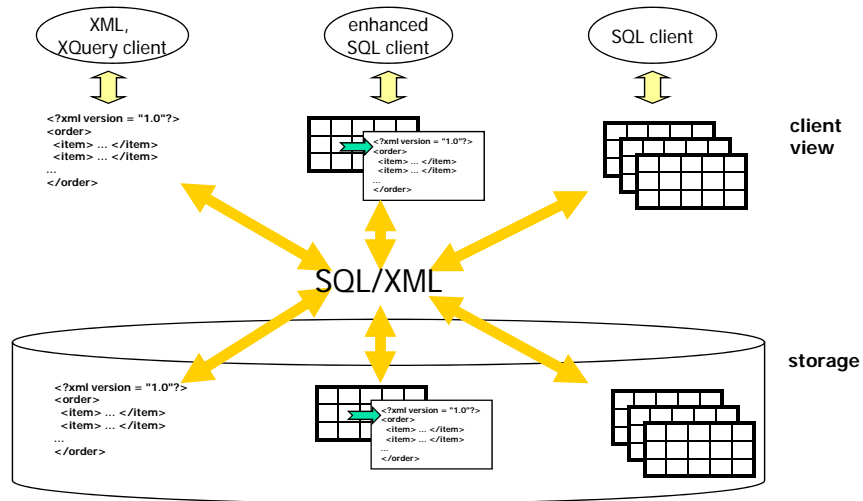
SQL and XML?!

- Two major perspectives
 - Flexible exchange of relational data using XML
 - publish relational as XML
 - decompose or "shred" XML into relational
 - Reliable XML data management
 - manage, search, maintain, update, ...
 - integrate with relational data
- Native-XML databases? No significant customer interest!
 - reluctance to introduce new DBMS environment
 - limited integration with relational DBMS products
 - lack of maturity (scalable, reliable, highly available, ...)
 - skill revolution (not evolution) required

Remember OO-DBMS?



SQL/XML Big Picture



© Prof. Dr.-Ing. Stefan Deßloch

27

Recent Developments for Data Models

SQL:2003 Parts and Packages

- Two major goals:
 - "Publish" SQL query results as XML documents
 - Ability to store and retrieve XML documents
- Rules for mapping SQL types, SQL identifiers and SQL data values to and from corresponding XML concepts
- A new built-in type *XML*
- A number of built-in operators that produce values of type *XML*

recent additions for SQL200n:

- Integration of the XQuery Data Model
- Additional XML Constructor Functions
- Querying XML values

optional features

(1) Enhanced Date/Time Fac.

(8) Active Databases

(6) Basic Objects

(10) OLAP

mandatory features

Core SQL



© Prof. Dr.-Ing. Stefan Deßloch

28

Recent Developments for Data Models

XML Data Type

```
CREATE TABLE employee
( id CHAR(6),
  lastname VARCHAR (30),
  ...,
  resume XML
)
```

ID	LASTNAME	...	RESUME
940401	Long	...	<?xml version="1.0"?> <resume xmlns="http://www.res.com/resume"> <name> ... </name> <address> ... </address> ... </resume>
862233	Nicks	...	null
766500	Banner	...	<resume ref="http://www.banner.com/resume.html"/>



XML Publishing Functions - Example

```
SELECT  e.id,
        XMLELEMENT (NAME "Emp",
                    e.fname || ' ' || e.lname)
        AS "result"
FROM    employees e
WHERE   ... ;
```

==>

ID	result
1001	<Emp>John Smith</Emp>
1006	<Emp>Mary Martin</Emp>



XMLQUERY

- Evaluates an XQuery or XPath expression
 - Provided as a character string literal

- Example

```
SELECT XMLQUERY('for $e in $dept[@count > 3]/emp
                where $e/hire > 2004-12-31 return $e/name'
                PASSING BY REF deptDoc AS "dept"
                RETURNING SEQUENCE) AS "Name_elements"
```

FROM XMLDept

=>

Name_elements
<name>Miller</name>
<name>Smith</name>
<name>Johnson</name>
<name>Martin</name>

