Object Relational DB

TOPICS

- Nested Relational Model
- Implementation and Related Issues for Extended Type Systems
- Comparison between RDBMS, OODBMS, ORDBMS.

Section 22.6
Nested Relational Model

- **Nested relational mode**:
  - Removes the restriction of the first normal form (1NF)
- No commercial database supports a nested relational model
- Visual representation:

Figure 22.1
Illustrating a nested relation. (a) DEPT schema. (b) Example of a Non-1NF tuple of DEPT. (c) Tree representation of DEPT schema.

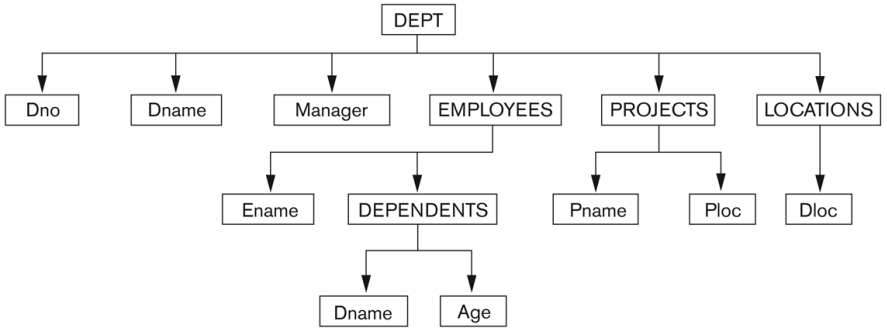
(a)

Dno	Dname	Manager	EMPLOYEES		PROJECTS		LOCATIONS
			Ename	DEPENDENTS	Pname	Ploc	Dloc
				Dname Age			

(b)

4	Administration	Wallace	Zelaya	Thomas	8	New benefits	Stafford	Stafford
				Jennifer	6	Computerization	Stafford	Greenway
			Wallace	Jack	18	Phone system	Greenway	
				Robert	15			
				Mary	10			
			Jabbar					

(c)



Attributes of Nested Relations

- **Simple value** attributes
- **Multi-valued simple** attributes
- **Multi-valued composite** attributes
- **Single-valued composite** attributes

Manipulating Nested Relations

- Extension made to
  - **Relational algebra**
  - **Relational calculus**
  - **SQL**
- Two operations for converting between nested and flat relations:
  - **NEST**
  - **UNNEST**

Example of NEST

- To nest un-nested attributes:

```
EMP_PROJ_FLAT ←
  Π SSN, ENAME, PNUMBER, HOURS (EMP_PRO)
EMP_PROJ_NESTED ←
  NEST PROJ = (PNUMBER,HOURS) (EMP_PROJ_FLAT)
```

- Nested relation **PROJS** within **EMP_PROJ_NESTED** groups together the tuples with the same value for the attributes that are not specified in the **NEST** operation

Example of UNNEST

- UNNEST operation is the inverse of NEST; thus we can recover **EMP_PROJ_FLAT**:

```
EMP_PROJ_FLAT ← UNNEST PROJS =
(PNUMBER,HOURS)  (EMP_PROJ_NESTED)
```

Implementation and Related Issues for Extended Type Systems

- The ORDBMS must dynamically link a user-defined function in its address space only when it is required
  - numerous functions are required to operate on two-or three-dimensional spatial data, images, text, and so on.
  - With a static linking of all function libraries, the DBMS address space may increase by an order of magnitude.

- Client-server issues deal with the placement and activation of functions
  - If the server needs to perform a function, it is best to do so in the DBMS address space rather than remotely, due to the large amount of overhead.
  - If the function demands computation that is too intensive or if the server is attending to a very large number of clients, the server may ship the function to a separate client machine.
  - For security reasons, it is better to run functions at the client using the user ID of the client.

- It should be possible to run queries inside functions.
  - A function must operate the same way whether it is used from an application using the application program interface (API), or whether it is invoked by the DBMS as a part of executing SQL with the function embedded in an SQL statement.
  - Systems should support a nesting of these "callbacks."

- Because of the variety in the data types in an ORDBMS and associated operators, efficient storage and access of the data is important.
  - For spatial data or multidimensional data, new storage structures such as R-trees, quad trees, or Grid files may be used.
  - The ORDBMS must allow new types to be defined with new access structures. Dealing with large text strings or binary files also opens up a number of storage and search options.
  - It should be possible to explore such new options by defining new data types within the ORDBMS.

- *Object-relational database design:*
  - Object-relational design is more complicated because we have to consider not only the underlying design considerations of application semantics and dependencies in the relational data model but also the object-oriented nature of the extended features that we have just discussed.
- *Query processing and optimization:*
  - By extending SQL with functions and rules, this problem is further compounded beyond the query optimization
- *Interaction of rules with transactions:*
  - Rule processing as implied in SQL3 covers more than just the update-update rules , which are implemented in RDBMSs as triggers. Moreover, RDBMSs currently implement only immediate execution of triggers. A deferred execution of triggers involves additional processing.

- Comparison
  - Self study