Object-Oriented Databases (OODB) Object Relational Databases (ORDBMS)

Douglas S. Kerr
CIS 671

Object Database Systems

(Elmasri & Navathe, Ch 11-13)

- Object-Oriented Database Systems (ODBMS)
- Object-Relational Database Systems (ORDBMS)

Why? Briefly

- Object-Oriented Databases (OODBMS)
  - Add persistence to object-oriented programming languages
- Object Relational Databases (ORDBMS)
  - Add user-defined methods to relational databases
  - Allow grouping of relations into more complex "objects"

Review of Object-Oriented Concepts

Object

- State (value)
- Behavior (operations)
  - *Signature* or *interface*: operation name and arguments
  - *Method* or *body*: implementation
- Identified by unique Object Identifier (OID)

Class

- Group of similar objects
- Class *hierarchies*
  - *Inheritance*
- *Persistence* must be specified explicitly
  - Via entry point - a *named* object
  - Via reachability - ∃ sequence of references from named persistent object

Type Hierarchies and Inheritance
(EER – superclass/subclass relationship)

PERSON: Name, Address, Birthdate, Age, SSN

EMPLOYEE: **subtype-of** PERSON: Salary, HireDate, Seniority
STUDENT: **subtype-of** PERSON: Major, GPA

EMPLOYEE: Name, Address, Birthdate, Age, SSN,
Salary, HireDate, Seniority
STUDENT: Name, Address, Birthdate, Age, SSN,
Major, GPA

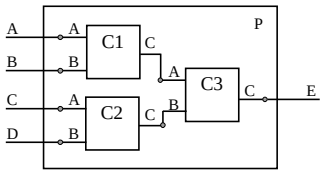
Object Data Management Group (ODMG)

- Object Model
- Object Definition Language (ODL)
- Object Query Language (OQL)
- Bindings to object-oriented programming languages
  - C++, Java, Smalltalk

ODMG Object Model

- Object
  - Identifier (unique OID)
  - Name (like primary key, optional)
  - Lifetime (persistent or transient)
  - Structure (atomic or collection object)
    - How to construct using type constructors

Example: 4-Input AND Gate Built From Three (3) 2-Input AND Gates



4-Input AND Gate in a Relational Database

GateType

GT*	Description	AbsLoc	PowerCons	Shape
2AND	"C=A & B"			
4AND	"E=A&B&C&D"			

PinType

GT*	PT*	I/O
2AND	A	I
2AND	B	I
2AND	C	O
4AND	A	I
4AND	B	I
4AND	C	I
4AND	D	I
4AND	E	O

GateInstance

GT	GI*	Parent
2AND	C1	4AND
2AND	C2	4AND
2AND	C3	4AND
4AND	P	-

WireInstance

WI*	GI1	Pin1	GI2	Pin2	Parent
W1	P	A	C1	A	4AND
W2	P	B	C1	B	4AND
W3	P	C	C2	A	4AND
W4	P	D	C2	B	4AND
W5	C1	C	C3	A	4AND
W6	C2	C	C3	B	4AND
W7	C3	C	P	E	4AND

Problems

- ***The relational model has no notion of a single object.***
- ***There is spatial data present,***
  e.g., location of a gate.
- ***There will be versions.***
- ***There may also be configurations,***
  i.e variations of the same design.

- ***Functions (or derived values) may be desired.***
  - Examples:

PowerCons(GateType) =
Σ PowerCons(ComponentGateInstances)

AbsLoc(GateInstance) =
transformation(RelLoc(GateInstance),
AbsLoc(GateType containing GateInstance))

Relational is Good For

- Fixed format data
- Transaction processing:
  simple short transactions
- Query processing
- Concurrency control
- Recovery from crashes

Problems With the Relational Model - 1

- Nonhomogeneous collection of design objects.
- Data Types: images, matrices, vectors; variable length.
- Temporal and/or spatial data.
- Many data types; few instances of each type.

Problems With the Relational Model - 2

- Schemas evolve during design.
- Long running transactions: "checkout a design".
- Versions; design log.

Problems With the Relational Model - 3

- Functions needed:
  - Making a design permanent,
  - Releasing design to production,
  - Archiving design,
  - etc.
- Library of design objects:
  - minimize redundancy.

Object-Oriented Concepts - 1

- Complex Objects:
  - sets,
  - bags,
  - lists,
  - arrays,
  - tuples.

Object-Oriented Concepts - 2

- Object Identity: identify by object id (oid) rather than by an attribute value, never changes.
- Encapsulation: Operations and Data available to user.
- Implementation hidden. No other operations available.

Object-Oriented Concepts - 3

- Types and/or Classes:
  - Type:
    - Interface and Implementation.
    - Types declared, checked at compile-time
  - Class:
    - Instances created at run-time

Object-Oriented Concepts - 4

- Class or Type Hierarchies:
  - Inheritance: both data and function

    example: student: undergraduate, graduate
- Overriding, overloading, and late binding:
  - Polymorphism

Object-Oriented Concepts - 5

- Computational Completeness:
- Extensibility:
  - Means to define new types.
  - User types not distinguished from system types.
- Persistence:
  - Not present in OO programming languages.
  - Should be implicit not explicit.

Object-Oriented Concepts - 6

- Secondary Storage Management:
- Concurrency:
- Recovery:
- Ad Hoc Query Facility:

Group of Closely Related Entities and Relationships

Text Example: Research Department with its locations, employees, and projects.

Departments

(DNAME,	DNUMBER*,	MGRSSN)
Research	5	333445555

Managers

(MGRSSN*,	MGRSTARTDATE)
333445555	22-May-78

Dept_Locations

DNUMBER*,	LOCATION*)
5	Bellaire
5	Sugarland
5	Houston

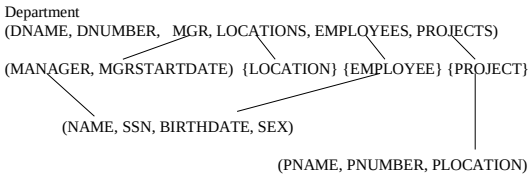
Projects

(PNAME,	PNUMBER*,	DNUM,	PLOCATION)
ProductX	1	5	Bellaire
ProductY	2	5	Sugarland
ProductZ	3	5	Houston

Employees

(NAME,	SSN*,	BIRTHDATE,	SEX,	DNUMBER)
Smith	123456789	9-Jan-55	M	5
Wong	333445555	8-Dec-45	M	5

OO View - schema



Instance (sort of) of Department

```
(Research  5  MGR LOCATIONS   EMPLOYEES  PROJECTS)

((Wong  333445555  08-Dec-45  M)  22-May-78)

                {Bellaire
                 Sugarland
                 Houston}

             {(Smith 123456789 09-Jan-55)
               (Wong 333445555 08-Dec-45)}

                       {(ProductX  1   Bellaire)
                         (ProductY  2   Sugarland)
                         (ProductZ   3  Houston)}
```


Operations

- NumberOfEmployeesInDepartment (d: Department): integer
- EmployeeAge(e: Employee): integer

Observations from Department Example

- Duplication of data
  - Employee “Wong” information
  - 3 locations “Bellaire”, “Sugarland”, “houston”
- Give each “object” an “objectID” or OID
- In some sense a reversion to navigation

Development of Object-based Systems

- Object-Oriented Database Systems
  - An alternative to relational systems
  - Application domains where objects play central role
  - Heavily influenced by object-oriented programming languages
  - An attempt to add DBMS functionality to a programming language environment

Development of Object-based Systems - cont.

- Object-Relational Database Systems
  - An attempt to extend relational databases
    - Broader set of applications
  - Provide bridge between relational and object-oriented systems