4541.564; Spring 2008

Prof. Sang-goo Lee

(14:30pm: Mon & Wed: Room 302-208)

ADVANCED DATABASES

Syllabus

- Text Books

  Database System Concepts, 5th Edition, A. Silberschatz, H. F. Korth, and S. Sudarshan, McGraw Hill, 2006.

  Relational Database Theory, Atzeni & De Antonellis, Benjamin/Cummings, 1993. (Chap 2)

  Database Systems, Atzeni, et al, McGraw Hill, 2000. (Chap 6~7)

- Lecture Notes
  - will be posted before class at http://europa.snu.ac.kr
  - username & password required
  - Please use only for personal use

- Exams (tentative dates)
  - Exam 1: 4/7 (Mon)
  - Exam 2: 5/19 (Mon)
  - Exam 3: 6/14 (Sat, 14:30)
- Term Project/Paper
  - 2~3 programming assignments
  - To be announced later
- Grades
  - Exams 1, 2, & 3: 20% each
  - Term Project/Paper: 30% total
  - Quizzes, Assignments, and others: 10%

**** A score of 0 in**
• any one of the exams,
• any one of your term project, term paper, or
• more than 50% of your assignments/quizzes
will result in F.

REVIEWS

Data, Database

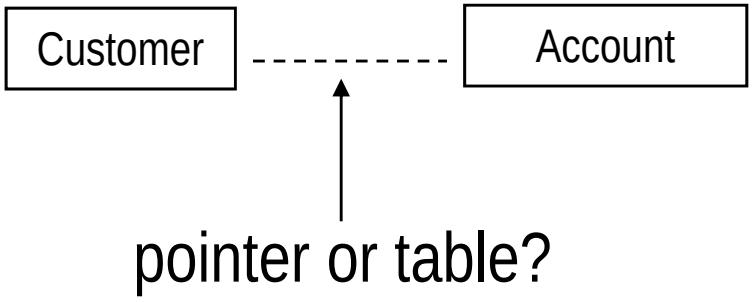
- Data
  - A *formal description* of
    - an entity, event, phenomena, or idea
    - that is worth recording
- Database
  - An integrated collection of
  - persistent data
  - representing the information of interest
  - for various programs that compose the computerized information system of an organization.
  - Data are separated from the programs that use them

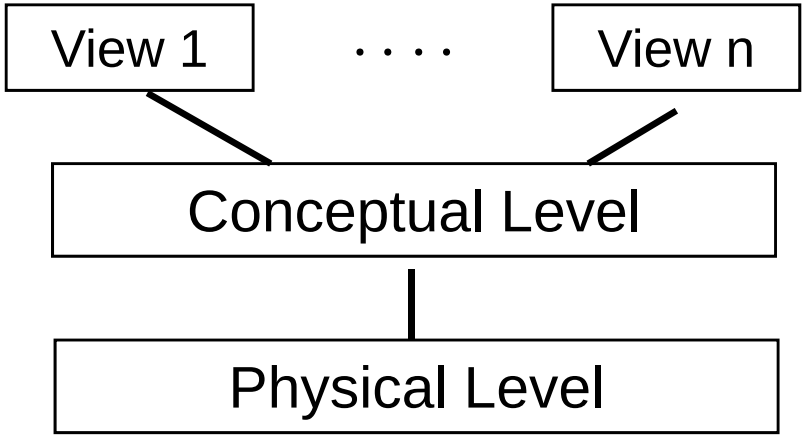
DBMS

- Database Management System
  - Collection of interrelated data and
  - a set of programs to access those data
- Information System
  - DB + DBMS + Application programs + utilities
- File System
  - Part of OS
  - Stores programs, data, documents, or anything
  - (in disk)

Data Independence

- ability to modify a schema in one level without affecting a schema definition in the next higher level
- physical data independence:
  - physical level - conceptual level
- logical data independence:
  - conceptual level - view level





Instances and Schemas

- Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them)
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable

Data Models

- The underlying structure of a database
- Collection of conceptual tools for describing
  - data
  - data relationships
  - data semantics
  - consistency constraints
- Entity Relationship Model
- Relational Model
- Object Oriented Model

Database Languages

- Data Definition Language (DDL)
  - Specifies the DB Schema
    - create table
    - drop column
- Data Manipulation Language (DML)
  - Operate on the contents of the DB
    - retrieve, insert, delete, change, etc.
- Query
  - a statement requesting the retrieval of information
  - query language: part of DML
  - data model dependent

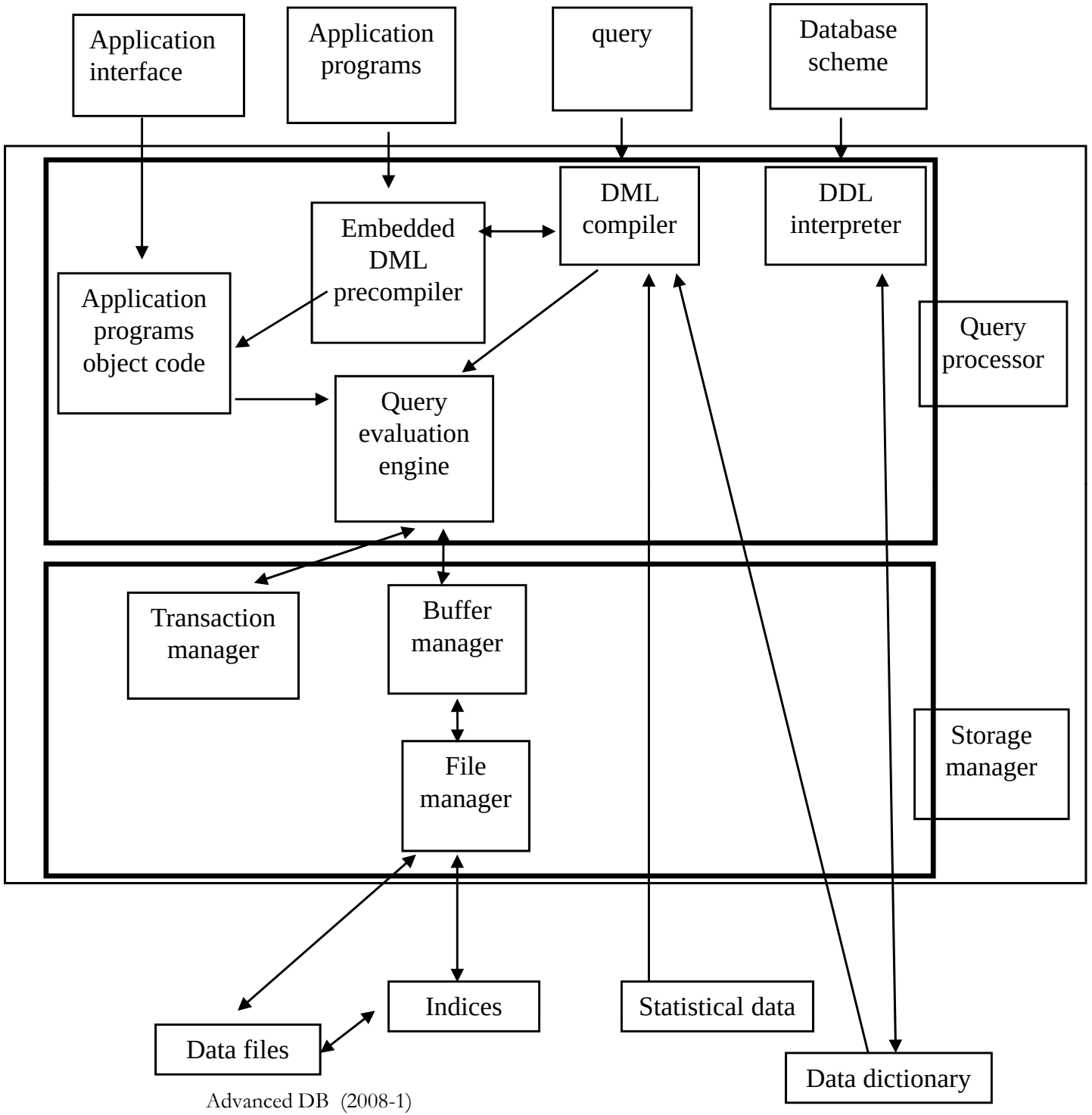
Storage Management

- DBMS must effectively and efficiently manage storage (disk) space
- Storage manager
  - a program module that
  - provides the interface between
  - the low-level data stored in the database and
  - the application programs and queries submitted to the system
- Physical Storage
  - Physical storage media hierarchy
  - RAID
  - Storage Access
  - File Organization
  - Storage Structures for Object-Oriented Databases

DB users

- DBA (DB Administrator)
  - schema definition
  - storage structure, access method definition
  - schema & physical organization
  - security & authorization
  - backup and recovery
- Application programmers
- Sophisticated users
  - use DML
- Naïve users
  - Use interfaces provided by application programs

Overall System Structure



Entity Relationship Model

- entity
  - entity set vs. entity (instance)
  - weak entity sets
- relationship
  - relationship cardinality
  - binary, ternary, n-ary
- attribute
  - multivalued attributes, derived attributes
- generalization/specialization
  - total-partial, exclusive-overlap
- aggregation

ER Diagram

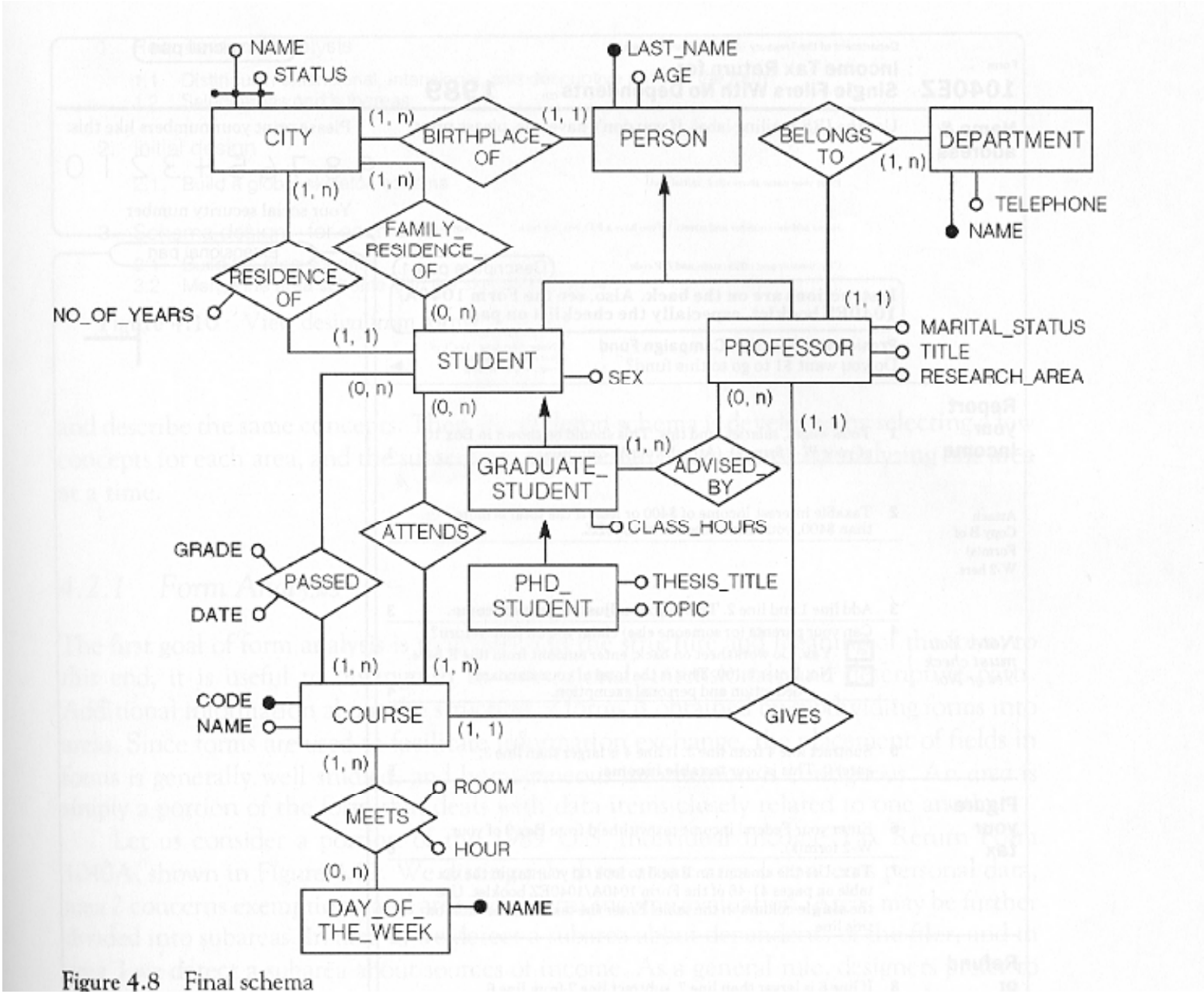


Figure 4.8 Final schema

Relational Model

- E. F. Codd, "a Relational Model of Data for Large Shared Data Banks," *Communications of the ACM*, June 1970, pp.377-387.
- Uses a single structure called relation
- Set (& math) oriented model
- Physical data independence
- Definition of a relation *R*:

  Let D_1, ..., D_n be domains, then

  $R \subset D_1$, X ….. X D_n

  $R = \{ <d_1, ..., d_n> \mid d_1 \in D_1, ..., d_n \in D_n \}$

Relations and Tables

- a tuple in a *relation* represents *relationship* among set of values
- Implemented as tables

Relation $R = \{ <a_1, b_1, c_1>, <a_2, b_2, c_2>, \quad , <a_n, b_n, c_n> \}$

=> Table *R*

column (field, attribute)

Name	Address	Telephone
HS Kim	Suwon	323-3232
KS Lee	Busan	323-5454
MH Choi	Seoul	553-3235
KH Na	Yongin	545-5488
...		

row (record, tuple)

Relational Database

- A relational database
  - a set of relations
  - a collection of tables

- Keys
  - superkey, candidate key, and primary key
  - keys are constraints on allowable relation instances for a given schema

Relational Algebra

- Query language
  - Network, Hierarchical : navigational language
  - Relational
    - relational algebra
    - relational calculus
    - SQL
    - QUEL
  - Algebra : operators and operands
  - Relational algebra
    - operands : relations
    - operators : fundamental operators + additional operators

Formal Definition

- A *basic expression in the relational algebra* consists of either one of the following:
  - A relation in the database
  - A constant relation
- Let E_1 and E_2 be relational-algebra expressions; the following are all *relational-algebra expressions*:
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - $E_1 \times E_2$
  - $\sigma_p (E_1)$, P is a predicate on attributes in E_1
  - $\prod_s (E_1)$, S is a list consisting of some of the attributes in E_1
  - $\rho_N (E_1)$, N is the new name for the result of E_1

Additional Operations

We define additional operations that do not add any power to the relational algebra, but that simplify common queries.

- Set intersection
- Natural join
- Division
- Assignment

SQL

- Structured Query Language
- IBM's System R project : Sequel
- RC-based: SQL is declarative
- DML & DDL
  - SELECT / UNION / INTERSECT / EXCEPT
  - INSERT / DELETE / UPDATE
  - CREATE / DROP / ADD

Integrity Constraints

- Integrity Constraints (IC) are rules that the data in the DB must abide by
- IC defines the semantics of the DB
- Domain Constraints
  - restricts the values of a column
- Referential Integrity
  - Foreign Key Constraint
  - Let *r1(R1), r2(R2)* be relations with primary keys *k1* & *k2*, respectively. $\alpha \subseteq R2$ is a foreign key referencing *k1* if it is required that for every $t2 \in r2$, there must be a tuple $t1 \in r1$ such that $t1[k1] = t2[\alpha]$
  - $\Pi_{\alpha}(r2) \subseteq \Pi_{k1}(r2)$

Integrity Constraints

- Assertion
  - a general constraint expressed as $\forall x\ P(x)$
    - but in SQL as $\exists x\ (\neg P(x))$

```
CREATE ASSERTION sum-constraint CHECK
  (NOT EXISTS (SELECT * FROM branch
               WHERE (SELECT SUM(amount) FROM loan
                      WHERE loan.b_name=branch.b_name)
                      >= (SELECT SUM(amount) FROM account
                      WHERE account.b_name=branch.b_name)))
```

- Trigger
  - Action tied to a DB event (insert/delete/update)

```
DEFINE TRIGGER overdraft ON UPDATE OF account T
               (IF NEW T.balance < 0
```

Functional Dependencies

- Basic Concept

  R: relation scheme. Let $\alpha \subseteq R$, $\beta \subseteq R$

  Functional dependency $\alpha \rightarrow \beta$ holds on R,

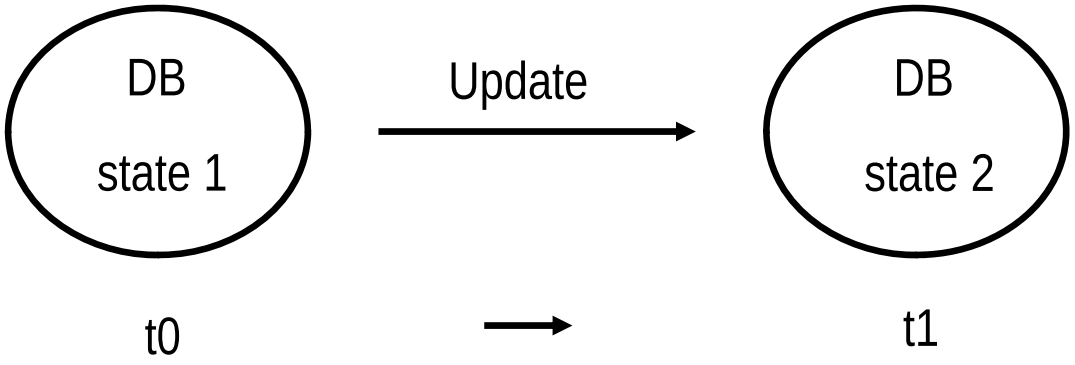
  if in any legal relation $r(R)$, for all pairs of tuples $t1$ and $t2$ in r,

  if $t1[\alpha] = t2[\alpha]$ then $t1[\beta] = t2[\beta]$

- Keys
- Trivial FD
- Inference Rules
  - Reflexivity, Transitivity, Augmentation
- Closure & Cover

Enforcing IC

- DB states
  - DB state = DB instance
  - DB is in a valid state if it satisfies all Integrity constraints
  - All assertions are tested when created
  - Modification to db is only allowed if it does not cause violation



Object-Oriented Data Model

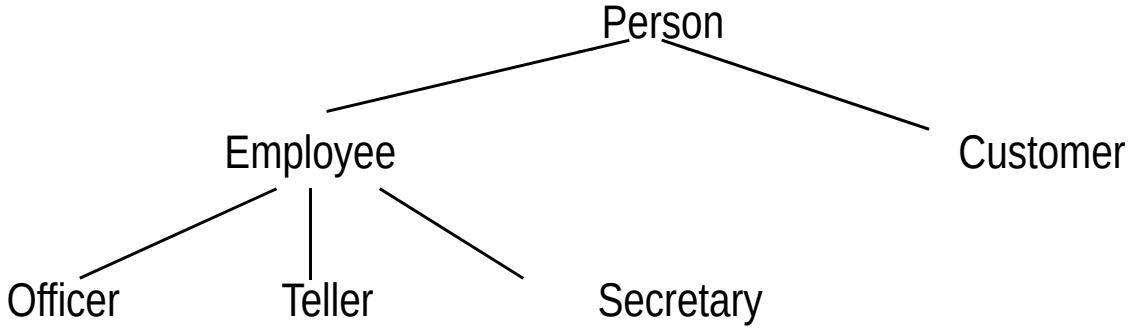
- Object Structure
  - an object corresponds to an entity in ER model
  - an object encapsulates data and behavior
  - all interactions to an object are made via messages.

  An object has
  - variables : contain data for object (attribute in ER model)
  - messages : interaction with rest of the world
  - methods : procedures that implement the messages

Classes & Inheritance

- Object Class
  - a group of similar objects
  - same messages, methods, variables

  ⇒ class : object (instance) = entity set : entity (instance)
- Inheritance



- Complex (Composite) Objects
  - Objects that contain other objects
  - Aggregation (containment) hierarchy
  - Use references (OID) as containment links

Object Identity & Persistence

- Object Identity
  - identity in relational model - key (by value)
  - identity in file system - file name (by name)
  - identity in OO system - file generated id (built-in)
    - An object retains its identity even if some or all variables or definitions of methods change over time
- Storage and access of persistent objects
  - How do you store methods?
  - How do you find objects in a database?
  - How do you store a composite object?

Object-Relational Database

- Extended relational model to support
  - Nested relations
  - Complex types
  - Object orientation
- Most commercial DBMS claim to be OR
  - Oracle, DB2, Informix, …

Nested Relation

- Relational model
  - First Normal Form: all attributes have atomic domains
- Nested relational model
  - Domains may be atomic or relation valued
    - tuple (complex structure)
    - set (multiset)
    - list (special set)

Querying with Complex Types

- Relation valued attributes
  - An expression evaluating to a relation
  - can appear anywhere that a relation name my appear

    Select *B.name*, *Y.name*
    from *docs* as *B*, *B.author_list* as *Y*

- Path expression (dot notation) is used
  - composite attributes / references

    create table *phd_students*
    (*advisor* ref(*people*)) under *people*

    select *phd_students.advisor.name*
    from *phd_students*

OO vs OR

- OR
  - declarative and limited power of (extended) SQL (compared to PL)
  - data protection and good optimization
  - extends the Rel. model to make modeling and querying easier
- OO
  - efficient in complex main mem. operations of persistant data
  - susceptible to data corruption
- Relational
  - simple data types, good query language, high protection

Indexing

- Ordered Indices
- B+-Tree Index Files
- B-Tree Index Files
- Static Hashing
- Dynamic Hashing
- Comparison of Ordered Indexing and Hashing
- Index Definition in SQL
- Multiple-Key Access

Query Processing & Optimization

- Measures of Query Cost
- Evaluation of individual operations
  - Select, sort, join
- Evaluation of Expressions
- Equivalence of expressions
- Cost based optimization

Transactions

- Transaction
  - a collection of operations that performs a single logical function in a database application
  - programmer is responsible for writing "correct" transactions
- DBMS must ensure the *atomicity* and *durability* of each transaction (at least)
  - Atomicity: all-or-nothing
  - Consistency: should not introduce inconsistencies
  - Isolation: should not interfere with each other
  - Durability: effect should be persistent

Concurrency Control & Recovery

- Serializability
- Lock-based protocols
- Time-stamp based protocols
- Multiple granularity
- Log-Based Recovery
- Shadow Paging