CS 411: Summary

Saurabh Sinha

Logistics

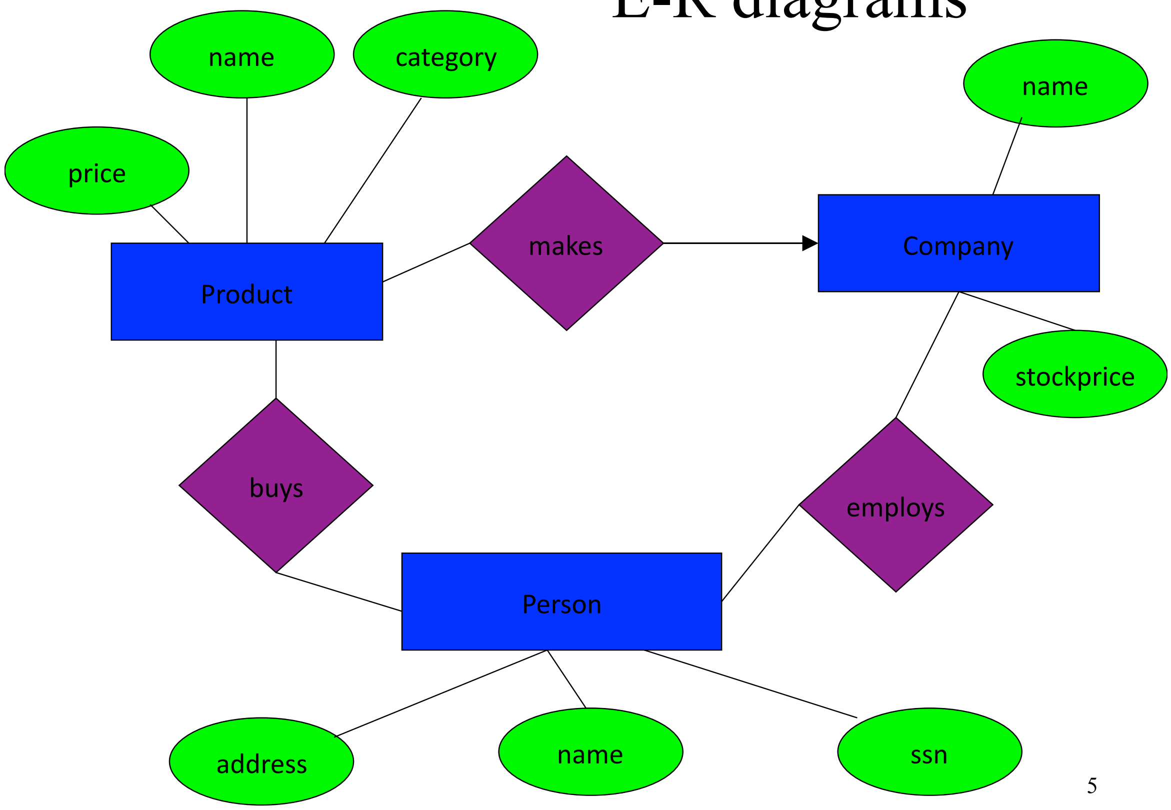
- Final exam on Thursday, 12/16 at 1:30 pm, in Siebel Center, rooms 1404 & 1105.
- Come to 1404, 10-15 mins before time.
- Closed book/note/computer.
- Bring your UIUC ID.
- Do not discuss exam on newsgroup.
- Syllabus: everything, with emphasis on 2^{nd} half.
- Final tutorial session: ??

Studying for finals …

- Go over the lecture slides. Study the textbook (chapter numbers indicated on course web page).
- Work on problems in hw/lectures.
- Work on sample exams (see course Web).
- Questions? Tutorial sessions if solved there.
- Questions? Office hours.

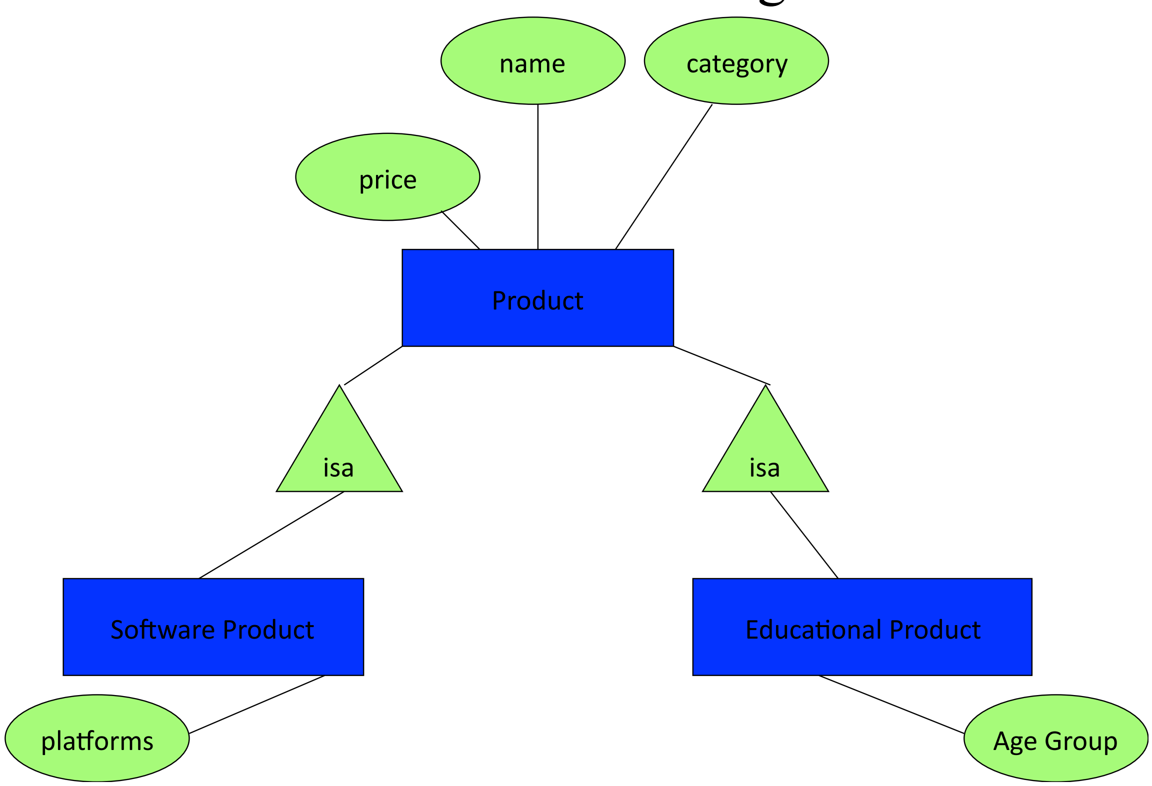
What did we do over the past four months?

E-R diagrams



5

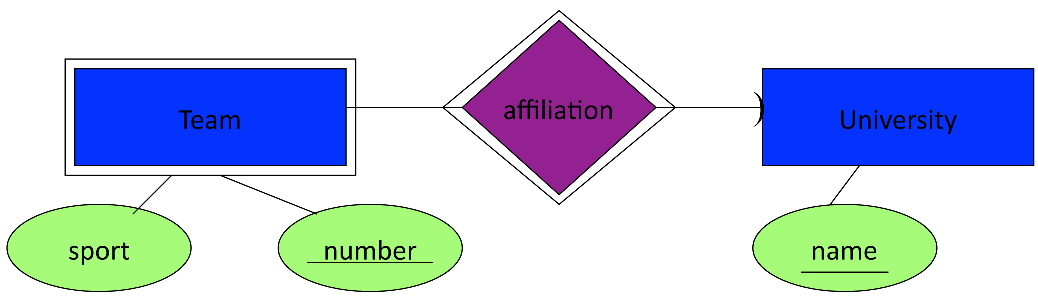
Subclasses in ER Diagrams



Weak Entity Sets

Entity sets are weak when their key attributes come from other classes to which they are related.

Football Team 1 (UIUC)



Design principles: Avoiding Redundancy

- Redundancy occurs when we say the same thing in two different ways.
- Redundancy wastes space and (more importantly) encourages inconsistency.
  - The two instances of the same fact may become inconsistent if we change one and forget to change the other, related version.

Relational Model

Table name

Attribute names

Products:

Name	Price	Category	Manufacturer
gizmo	$19.99	gadgets	GizmoWorks
Power gizmo	$29.99	gadgets	GizmoWorks
SingleTouch	$149.99	photography	Canon
MultiTouch	$203.99	household	Hitachi

tuples

Translating ER Diagram to Rel. Design

- Basic cases
  - entity set E = relation with attributes of E
  - relationship R = relation with attributes being keys of related entity sets + attributes of R
- And some special cases we'll cover afterwards

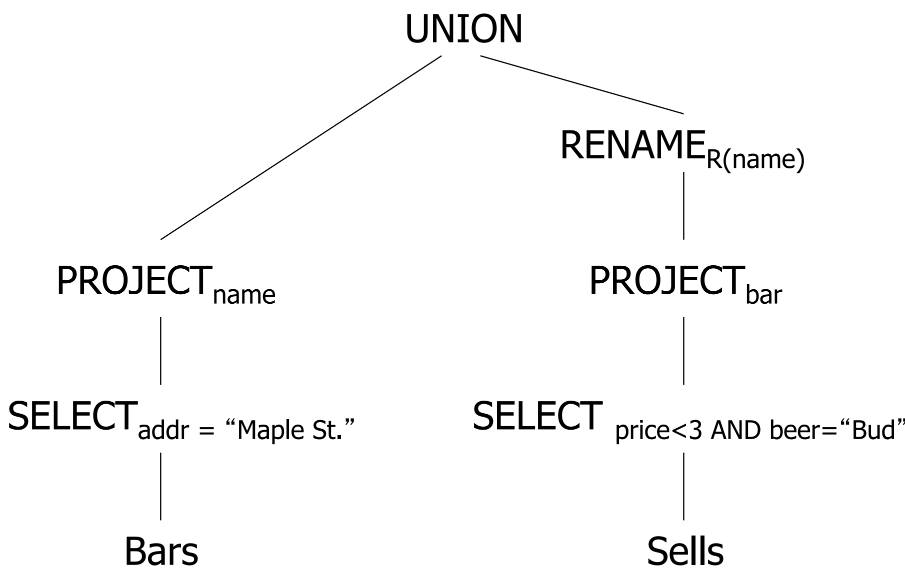
Relational Algebra at a Glance

- Operators: relations as input, new relation as output
- Five basic RA operations:
  - Basic Set Operations
    - union, difference (no intersection, no complement)
  - Selection: σ
  - Projection: π
  - Cartesian Product: X
- When our relations have attribute names:
  - Renaming: ρ
- Derived operations:
  - Intersection, complement
  - "Join"s (natural, equi-join, theta join)

Building Complex Expressions

- Algebras allow us to express sequences of operations in a natural way.
- Example
  - in arithmetic algebra: $(x + 4)*(y - 3)$
  - in stack "algebra": T.push(S.pop())
- Relational algebra allows the same.
- Three notations:
  1. Sequences of assignment statements.
  2. Expressions with several operators.
  3. Expression trees.

As a Tree:



Relation Design

- Start with the original db schema R
- Transform it until we get a good design R*

Desirable Properties of Schema Refinement

1) minimize redundancy
2) avoid info loss
3) preserve dependency
4) ensure good query performance

Normal Forms

First Normal Form = all attributes are atomic
Second Normal Form (2NF) = old and obsolete

Boyce Codd Normal Form (BCNF)
Third Normal Form (3NF)
Fourth Normal Form (4NF)

Others...

Desirable Properties of Schema Refinement (again)

1) **minimize redundancy**
2) **avoid info loss**
3) preserve dependency
4) ensure good query performance

BCNF

Desirable Properties of Schema Refinement (again)

1) minimize redundancy
2) **avoid info loss**
3) **preserve dependency**
4) ensure good query performance

3NF

SQL

- The principal form of a query is:

  SELECT   desired attributes
  FROM   one or more tables
  WHERE   condition about tuples of the tables

SQL: Aggregations

- SUM, AVG, COUNT, MIN, and MAX can be applied to a column in a SELECT clause to produce that aggregation on the column.

- Also, COUNT(*) counts the number of tuples.

SQL: Database Modifications

- A modification command does not return a result as a query does, but it changes the database in some way.
- There are three kinds of modifications:
  1. *Insert* a tuple or tuples.
  2. *Delete* a tuple or tuples.
  3. *Update* the value(s) of an existing tuple or tuples.

SQL: Views

- A view is a "virtual table," a relation that is defined in terms of the contents of other tables and views.
- Declare by:

  CREATE VIEW <name> AS <query>;
- Views are not stored in the database, but can be queried as if they existed.
- In contrast, a relation whose value is really stored in the database is called a *base table*.

SQL: Constraints and Triggers

- A *constraint* is a relationship among data elements that the DBMS is required to enforce.
  - Example: key constraints.

- *Triggers* are only executed when a specified condition occurs, e.g., insertion of a tuple.
  - Easier to implement than many constraints.

The Data Definition Language

- Simplest form is:

```
CREATE TABLE <name> (
    <list of elements>
);
```

- And you may remove a relation from the database schema by:

```
DROP TABLE <name>;
```

The user's perspective

- E/R diagrams: High level design
- Relation model: Low level (detail) design
- Schema refinement: Better design
- SQL: Implementation

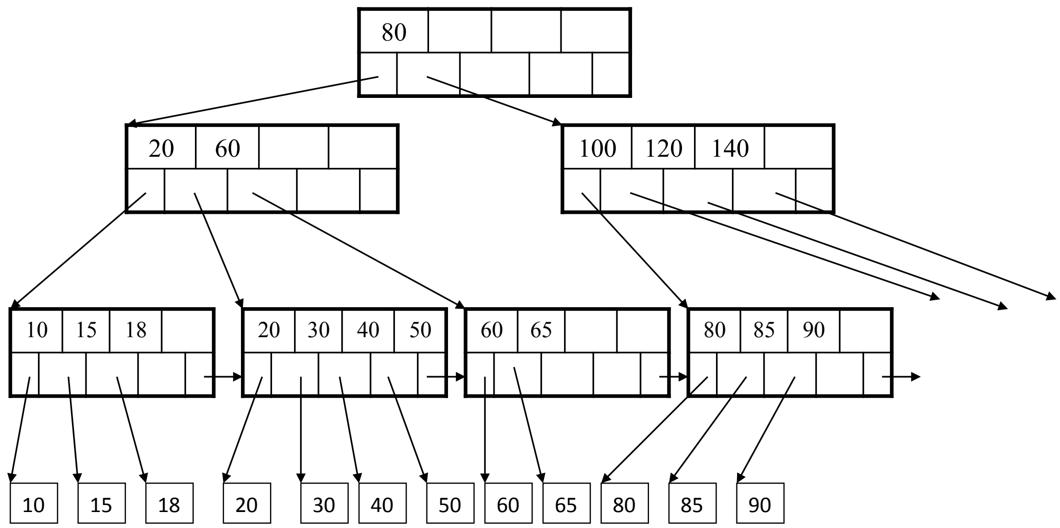
- Next up: the systems perspective

Indexes in databases

- An *index* on a file speeds up selections on the *search key field(s)*
- Search key = any subset of the fields of a relation
  - *Search key* is not the same as *key* (minimal set of fields that uniquely identify a record in a relation).
- Entries in an index: (k, r), where:
  - k = the key
  - r = the record OR record id OR record ids

B+ Tree Example

d = 2



Hash Tables

- Secondary storage hash tables are much like main memory ones
- Recall basics:
  - There are n *buckets*
  - A hash function f(k) maps a key k to {0, 1, …, n-1}
  - Store in bucket f(k) a pointer to record with key k
- Secondary storage: bucket = block
  - Store in bucket f(k) any record with key k
  - use overflow blocks when needed

Query Processing

- Logical operators
  - *what* they do
  - e.g., union, selection, project, join, grouping
- Physical operators
  - *how* they do it
  - e.g., nested loop join, sort-merge join, hash join, index join
  - In other words, physical operators are particular implementations of relational algebra operations
  - Physical operators also pertain to non RA operations, such as "scanning" a table.

Cost Parameters (or "statistics")

- Estimating the cost:
  - Important in optimization (next lecture)
  - Compute disk I/O cost only
  - We compute the cost to *read* the arguments of the operator
  - We don't compute the cost to *write* the result

- Cost parameters
  - M = number of blocks that fit in main memory
  - B(R) = number of blocks needed to hold R
  - T(R) = number of tuples in R
  - V(R,a) = number of distinct values of the attribute a

One pass algorithms

Nested Loop Joins (“one and a half pass” algorithms)

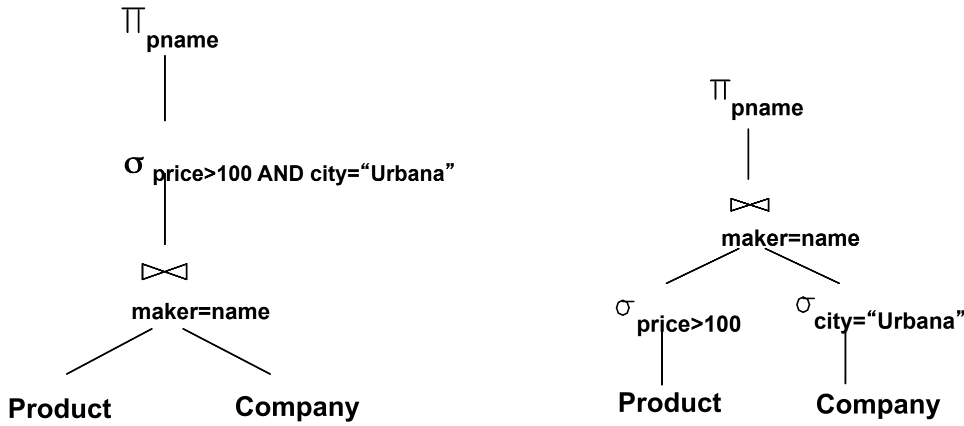
Two pass algorithms

Index-based algorithms

Query Optimization

- At the heart of the database engine
- Step 1: convert the SQL query to a logical plan
- Step 2: find a better logical plan, find an associated physical plan
- (Feed the physical plan into the query processor.)

Pushing selection down



The earlier we process selections, less tuples we need to manipulate higher up in the tree (but may cause us to lose an important ordering of the tuples, if we use indexes).

Cost-based Optimizations

- Main idea: apply algebraic laws, until estimated cost is minimal
- Start from partial plans, build more complete plans
  - Will see in a few slides
- Problem: there are too many ways to apply the laws, hence too many (partial) plans

16.5.4

Optimal Join Trees

- Given: a query R1 ⋈ R2 ⋈ … ⋈ Rn
- Assume we have a function cost() that gives us the cost of every join tree
- Find the best join tree for the query

Completing the Physical Query Plan

- Choose algorithm to implement each operator
  - Need to account for more than cost:
    - How much memory do we have ?
    - Are the input operand(s) sorted ?

- Decide for each intermediate result:
  - To materialize: create entirely and store on disk
  - To pipeline: create in parts and move on to next operation; entire result may never be available at the same time, not stored on disk.

Estimating Sizes

Simplifying assumptions:

- *Containment of values*: if V(R,A) <= V(S,A), then the set of A values of R is included in the set of A values of S
  - Note: this holds when A is a foreign key in R, and a key in S

- *Preservation of values sets*: for any other attribute B, $V(R \bowtie_A S, B) = V(R, B)$ (or $V(S, B)$)

Transaction manager and Recovery

Undo Recovery with Checkpointing

During recovery,
Can stop at first
<CKPT>

…
…
<T9,X9,v9>
…
…
(all completed)
<CKPT>
<START T2>
<START T3
<START T5>
<START T4>
<T1,X1,v1>
<T5,X5,v5>
<T4,X4,v4>
<COMMIT T5>
<T3,X3,v3>
<T2,X2,v2>

other transactions

transactions T2,T3,T4,T5

Redo Recovery with Nonquiescent Checkpointing

Step 1: look for
The last
<END CKPT>

…
<START T1>
…
<COMMIT T1>
…
…
<START CKPT T4, T5, T6>
…
…
…
…
<END CKPT>
…
…
…
<START CKPT T9, T10>
…

Step 2: redo
from <START Ti> for
Ti in {T4, T5, T6}.

All OUTPUTs
of T1 are
known to be on disk

The System Perspective

- Indexing
- Query Processing
- Query Optimization
- Transaction Management and Recovery

THANK YOU AND BEST WISHES !