CS 133: Databases

Fall 2016
Lec 20 – 11/16
Prof. Beth Trushkowsky

Lab 4:
Buffer Pool and Deadlock detection

• Buffer pool policy: NO STEAL, FORCE
  – (no logging)

• Deadlock detection
  – Timeout-based approach
  – Waits-for-graph
    • Idea: if xact were to acquireLock() would deadlock happen? If so, abort
  – How to abort? Throw DeadlockException

Goals for Today

• Learn about the process of designing a database to model a real-world application

• Understand how to encode an application in an entity-relationship (ER) diagram

• Reason about translating an ER model to a relational model

Database Design

• Requirements Analysis
  – user needs; what must database do?

• Conceptual Design
  – high level description (often done w/ ER model)

• Logical Design
  – translate ER into DBMS data model

• Schema Refinement
  – consistency, normalization

• Physical Design - indexes, disk layout

• Security Design - who accesses what
Data Models – Describing Data

- A *Database design* encodes some portion of the real world

- A *Data Model* is a set of concepts for thinking about this encoding

Helpful to start with a graphical representation: the *Entity-Relationship* model!

Integrity Constraints (ICs)

- Remember the “C” in ACID (Consistency)

- **Integrity Constraint (IC):** condition that must be true for any instance of the database
  - e.g., *domain constraints, keys and foreign key*
  - ICs are specified when schema is defined.
  - ICs are checked when relations are modified.

- **Come from semantics of the real world!**
  - Should be determined during Requirements Analysis and/or Conceptual Design phases

Entity-Relationship (ER) Model Basics

- **Entity:** Real-world object, distinguishable from other objects. An can have a set of *attributes.*

- **Entity Set:** A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. *(Until we consider hierarchies, anyway)*
  - Each entity set has a *key (underlined).*
  - Each attribute has a *domain.*

ER Model Basics (Contd.)

- **Relationship:** Association among two or more entities. E.g., Alice *works in* Pharmacy department.
  - Relationships can have their own attributes
  - Relationships uniquely identified only by participating entities, excluding attributes

- **Relationship Set:** Collection of similar relationships
  - An $n$-ary relationship set $R$ relates $n$ entity sets $E_1 ... E_n$; each relationship in $R$ involves entities $e_1 \in E_1, ..., e_n \in E_n$
• Same entity set can participate in different relationship sets, or in different “roles” in the same set.

### Key Constraints

An employee can work in many departments; a dept can have many employees.

In contrast, each dept has at most one manager, according to the *key constraint* on Manages.

### Weak Entities

A *weak entity* can be identified uniquely only with the primary key of another (*owner*) entity.

- Owner entity set and weak entity set must participate in a *one-to-many* relationship set (one owner, many weak entities).
- Weak entity set must have total participation in this *identifying* relationship set.

Weak entities have only a “partial key” (dashed underline)
ISA (‘is a’) Hierarchies

- **Overlap constraints:** Can Alice be an Hourly_Emps as well as a Contract_Emps entity? (Allowed/disallowed)

- **Covering constraints:** Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? (Yes/no)

- **Reasons for using ISA:**
  - To add descriptive attributes specific to a subclass.
    - i.e. not appropriate for all entities in the superclass
  - To identify entities that participate in a particular relationship
    - i.e., not all superclass entities participate

Conceptual Design Using the ER Model

- ER modeling can get tricky!
- **Example design choices:**
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?

- Note constraints of the ER Model:
  - A lot of data semantics can (and should) be captured
  - But some constraints cannot be captured in ER diagrams
    - We’ll refine things in our logical (relational) design

Entity vs. Attribute

- E.g., capturing employee “Address“:

  - Employees **vs.** Employees has Address

- It depends! Semantics and usage
  - Several addresses per employee?
    - must be an entity!
    - atomic attribute types (no set-valued attributes!)
  - Care about structure? (city, street, etc.)
    - must be an entity!
    - atomic attribute types (no tuple-valued attributes!)

Example: Entity vs. Attribute

- Works_In2: employee cannot work in a department for >1 period.

- Like multiple addresses per employee!
Example: Entity vs. Relationship

- Implies separate discretionary budget \( (dbudget) \) for each dept

- What if manager’s \( dbudget \) covers all managed depts
  - Could repeat value (Confusing)
  - Also redundancy problems

Better design:

Relationship Sets to Tables

- In translating a many-to-many relationship set to a relation, attributes of the relation must include:
  1) Keys for each participating entity set (as foreign keys).

  This set of attributes forms a super key for the relation.

  2) All descriptive attributes.

In general, schema will include keys from entity sets (but not necessarily all will form primary key)

Logical DB Design: ER to Relational

Entity sets to tables

CREATE TABLE Employees
(ssn CHAR(11),
 name CHAR(20),
 lot INTEGER,
 PRIMARY KEY (ssn))

CREATE TABLE Works_In
(ssn CHAR(1),
did INTEGER,
since DATE,
PRIMARY KEY (ssn, did),
FOREIGN KEY (ssn)
REFERENCES Employees,
FOREIGN KEY (did)
REFERENCES Departments)

Translating ER with Key Constraints

One way to translate the Manages Relationship:

CREATE TABLE Manages
(ssn CHAR(11),
did INTEGER,
since DATE,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees,
FOREIGN KEY (did) REFERENCES Departments)
Translating ER with Key Constraints (2)

Since each department has a unique manager, we could instead combine Manages and Departments.

```
CREATE TABLE Dept_Mgr(
    did INTEGER,
    dname CHAR(20),
    budget REAL,
    ssn CHAR(11),
    since DATE,
    PRIMARY KEY (did),
    FOREIGN KEY (ssn) REFERENCES Employees
)
```

Translating Weak Entity Sets

- Weak entity set and identifying relationship set are translated into a single table.
  - When the owner entity is deleted, all owned weak entities must also be deleted.

```
CREATE TABLE Dep_Policy ( 
    pname CHAR(20),
    age INTEGER,
    cost REAL,
    ssn CHAR(11) NOT NULL,
    PRIMARY KEY (pname, ssn),
    FOREIGN KEY (ssn) REFERENCES Employees,
    ON DELETE CASCADE 
)
```

Participation Constraints in SQL

- We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).

```
CREATE TABLE Dept_Mgr( 
    did INTEGER,
    dname CHAR(20),
    budget REAL,
    ssn CHAR(11) NOT NULL,
    since DATE,
    PRIMARY KEY (did),
    FOREIGN KEY (ssn) REFERENCES Employees,
    ON DELETE NO ACTION 
)
```

ISA Hierarchies to Relations

- **Three relations**
  - Employees(ssn, name, lot)
  - Contract_emps(ssn, contractid)
  - Hourly_Emps(ssn, hourly_wages, rating, hours_worked)

- **Alternative (assuming covering)**
  - Employees(ssn, name, lot, contractid)
  - Hourly_Emps(ssn, name, lot, hourly_wages, rating, hours_worked)

Query to get:
Names of all Employees?
Names of just Hourly Emps?
CREATE TABLE Owns
(
    ssn CHAR(11),
    bar_name CHAR(20),
    PRIMARY KEY (bar_name),
    FOREIGN KEY (ssn)
    REFERENCES Drinkers,
    FOREIGN KEY (bar_name)
    REFERENCES Bars(name))