Database Topic Due Monday! (11:59pm)

- Brief paragraph explaining the topic, goals, and users of the database (½ page at most)

- List of data (nouns) you will store
  - does not need to be complete

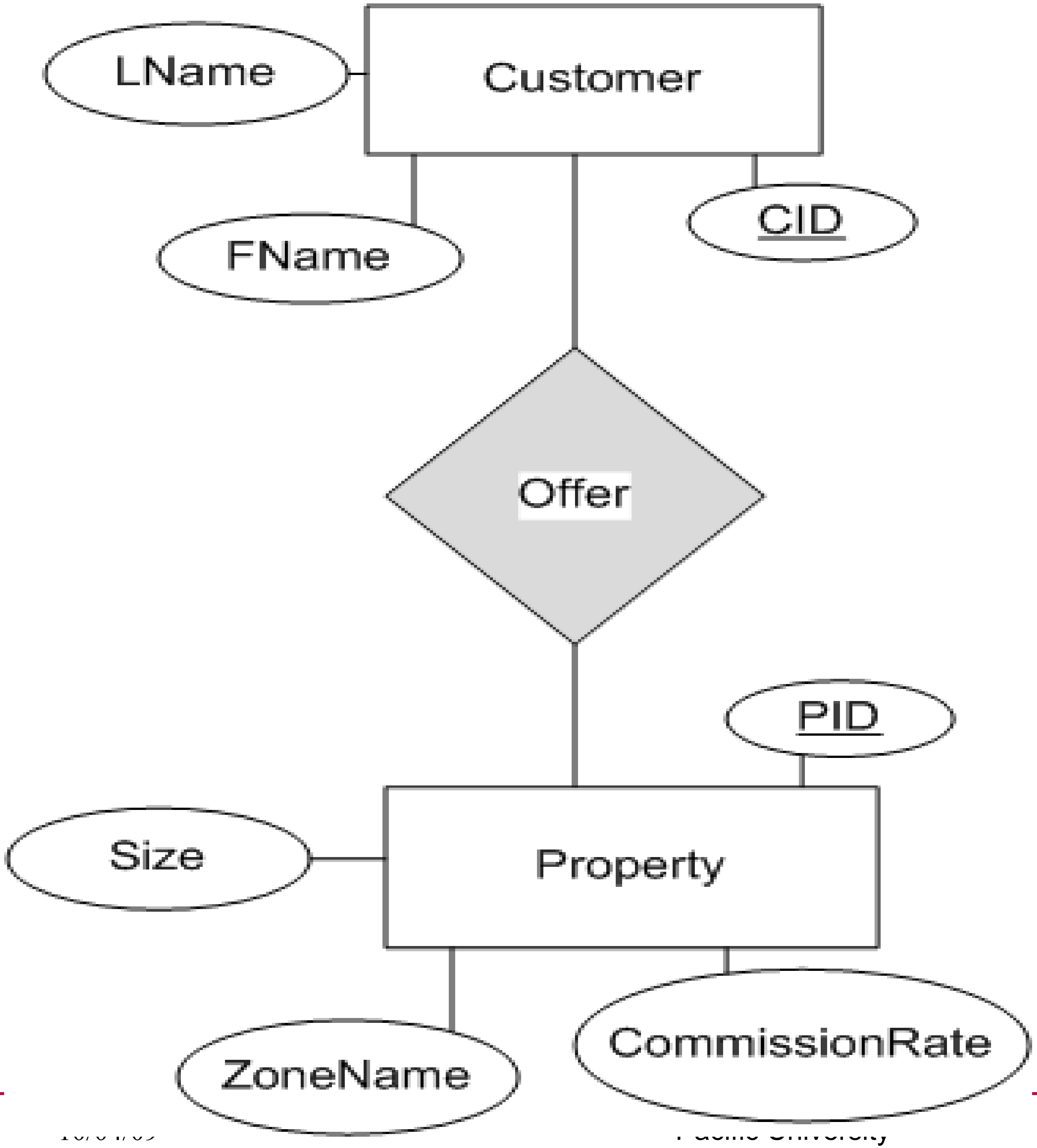
- Three queries that your users will want to answer

- Worth 5 points (of 25 points for Design Documents)
- Submit electronically (GoogleDoc)
  - CS445_DBTopic_PUNetID
- If I have any concerns, I'll ask you to schedule an appointment with me

Normalization

Oct 5, 2009

Chapter 19



What does this look like in the database?

How could this cause us problems?

Problems

- Redundant Storage

- Update Anomalies

- Insertion Anomalies

- Deletion Anomolies

Solutions

- Get rid of redundancy!

- Identify *functional dependencies*

- Decompose Relations
  - Must preserve semantics of relations (don't lose data)
    - and by lose we may mean gain
  - Must preserve all dependencies (constraints)

Function Dependency

- FD:

- Key
  "If X -> Y holds, where Y is the set of all attributes, and there is no proper subset V of X such that V -> Y holds, then X is a key." [1]
- Superkey
  "If X -> Y holds, where Y is the set of all attributes, then X is a superkey."[1]

- A key is also a superkey

[1]http://www.imada.sdu.dk/~meer/dm26/

Error on page 612, top paragraph

Set of FDs

- Closure:
  - F is a set of FDs for Relation R, closure of F is F^+
- Armstrong's Axioms:
  - Reflexivity:
  - Augmentation:
  - Transitivity:
  - Sound
  - Complete

Additional Rules

- Union:

- Decomposition:

- Trivial FD
  - X -> Y: all attributes in Y are in X
  - {SID, Major, Name} -> { Major, Name }

Normal Forms

- Boyce-Codd Normal Form (BCNF)

  if there is an FD B->a in relation R then

      B -> a is trivial (a ϵ B)
      or
      B is a superkey

      From the Assignments:

      FD {ZoneName} -> {CommisionRate}

      Is Property in BCNF? Why or why not?

3rd Normal Form

if there is an FD B->a in relation R then

B -> a is trivial (a ϵ B)

or

B is a superkey

or

a is part of some key for R

Less restrictive (weaker) than BCNF. More practical, easier to preserve dependencies.

- Possible violations: X -> A
  - X is a proper subset of some key K
    - partial dependency
  - X is not a proper subset of any key
    - transitive dependency
- Everything in BCNF is in 3NF, everything not in 3NF is not in BCNF

Example 3NF

- BoatReservation (page 633 section 19.7.4)

  (SailorID, BoatID, Date, CreditCard)

  Key: (SailorID, BoatID, Date)

  What type of relationship is this?

  FD: {SailorID} -> {CreditCard}

  What does this FD mean?

  Is this in 3NF?

  Is this in BCNF?

Example 3NF

- BoatReservation (page 619)

  (SailorID, BoatID, Date, CreditCard)

  Key: (SailorID, BoatID, Date)

  FD: {SailorID} -> {CreditCard}

  If we also have FD {CreditCard}->{SailorID}

  what does this FD mean?

  Is this in 3NF?

  Is this in BCNF?

Decompositions

- To put a Relation R in BCNF:
  - if R is not in BCNF then there must be at least one nontrivial FD a -> B such that a is not a superkey for R.
  - Rewrite R as two schemas:
    - (a U B)
    - (R – (B - a))

Lossy Decomposition

S	P	D
s1	p1	d1
s2	p2	d2
s3	p1	d3

Original Relation

S	P
s1	p1
s2	p2
s3	p1

P	D
p1	d1
p2	d2
p1	d3

Decomposed Relations

What data was lost?

Test to determine losslessness:

When R is decomposed into R1 and R2, the attributes common to R1 and R2 must contain a key for either R1 or R2.

Formally:

F^+ (of R) contains either FD R1 ∩ R2 -> R1 or FD R1 ∩ R2 ->R2

s	p	d
s1	p1	d1
s2	p2	d2
s3	p1	d3
s1	p1	d3
s3	p1	d1

New Relation

page 620

Dependency Preservation

- "Allow us to enforce all FDs by examining a single relation instance" on each change of that relation instance
- Enforcing an FD across relations instances is expensive (if possible)

- If we decompose relation R down in to X and Y, the dependencies are preserved if $(F_x \cup F_y)^+ = F^+$
  - if we insert/delete/update into/from X or Y, we only need to examine the respective relation to check constraints

Page 621 (with edits for clarity)

Decomposition

- Relation (C,S,J,D,P,V,P)
  - FD: {C}->{C,S,J,D,P,V}, {J,P} ->{C}, {S,D} -> {P}
    What FDs can we infer?

    What are keys?

    SuperKeys?

    What violates BCNF?

    How do we decompose this?

    What dependency is not preserved?

Page 621 (with edits for clarity)

Normalization

- The process of putting a schema in a particular normal form
  - BCNF
    - may not be a be able to create a dependency-preserving decomposition in BCNF
  - 3NF
    - can always create a lossless, dependency-preserving decomposition in 3NF

Normalization to BCNF

- If R is not in BCNF there must be at least one FD X->Y such that Y is a single attribute and X->Y violates BCNF.
- Decompose R into R-Y and XY
- Repeat while R is not in BCNF
  {CSJDPQV}   FDs: {JP}->{C} ; {SD}->{P}

- To preserve dependencies in BCNF, we may store some redundant information
  - still can't always preserve dependencies, however
    {CSP} FDs: {CS}->{P} ; {P}->{C} ; KEYs: {CS}, {PS}

Normalization to 3NF

- We can use the method above to get a lossless decomposition in BCNF (hence it is in 3NF)
- This does not ensure dependency preservation
  - we need to add that for a 3NF normalization
- Minimal Cover set for FDs
  - given a set of FDs F, a minimal cover set of FDs G is
    - X->A is in G, and A is a single attribute
    - F^+ is equal to G^+
    - if any FDs are deleted from G to form set H, $H^+ \neq F^+$

Minimal Cover, example

- FDs {A}->{B} {ABCD}->{E} {EF}->{G} {EF}->{H} {ACDF}->{EG}
- Single attribute on Right:
- Minimize Left Side
- Remove redundant FDs

Decomposition into 3NF

- R is a relation with a set of FDs F where F is a minimal cover
- Produce a lossless decomposition as per BCNF
  - produce relations D = $\{R_1, R_2, \ldots, R_n\}$
- Identify FDs in F not preserved in the closure of the FDs in $R_1 \ldots R_n$
  - for each non-preserved FD {X}->{A}, add relation XA to D