CS186 Discussion 6

(Relational Algebra, ER Diagrams, Functional Dependencies)

Matthew Deng

Relational Algebra

Relational Algebra

- Set of operators that map relations to relations
- Set semantics
  - Does not include duplicates
  - SQL has multiset semantics

Relational Operators

- Selection (σ) Selects a subset of rows (horizontal)
  - $\sigma_{age>20}(R)$
- Projection (π) Selects a subset of columns (vertical)
  - $\pi_{name,\ age}(R)$
- Cross-product (×) Allows us to combine two relations.
  - $R \times S$
- Set-difference (—) Tuples in r1, but not in r2.
  - $R - S$
- Union (U) Tuples in r1 or in r2.
  - $R \cup S$

Compound Operators

- Intersection ($\cap$) Tuples in r1 and r2
  - $R \cap S = R - (R - S)$
- Join ($\bowtie$) joins r1 and r2 on common attributes
  - Compute $R \times S$
  - Select rows where attributes appearing in both relations have equal values
  - Project onto all unique attributes and one copy of each of the common ones

Relational Algebra Exercises

Relational Algebra Exercises

1. What are the 6 basic operators in relational algebra, and what does each one do?

Relational Algebra Exercises

1. What are the 6 basic operators in relational algebra, and what does each one do?

Selection – filters on a predicate

Projection – outputs subset of fields

Cross-product – returns cartesian product of 2 input relations

Union – returns tuples that appear in either R or S for (R U S) [must have the same schema]

Set Difference – returns tuples in R that are not in S for (R - S) [must have the same schema]

Rename – Renames a field e.g. C(1-> sid

Relational Algebra Exercises

Consider the schema:

`Songs (song_id, song_name, album_id, weeks_in_top_40)`

`Artists(artist_id, artist_name, first_year_active)`

`Albums (album_id, album_name, artist_id, year_released, genre)`

Write relational algebra expressions for the following queries:

1. Find the name of the artists who have albums with a genre of either ‘pop’ or ‘rock’.

Relational Algebra Exercises

Consider the schema:

```
Songs (song_id, song_name, album_id,
weeks_in_top_40)
Artists(artist_id, artist_name, first_year_active)
Albums (album_id, album_name, artist_id, year_released,
genre)
```

Write relational algebra expressions for the following queries:

1. Find the name of the artists who have albums with a genre of either ‘pop’ or ‘rock’.

$$\pi_{\text{artists.artist\_name}}$$
$$(\text{Artists} \bowtie (\sigma_{\text{albums.genre = ‘pop’} \vee \text{albums.genre = ‘rock’}} \text{Albums}))$$

Relational Algebra Exercises

Consider the schema:

```
Songs (song_id, song_name, album_id, weeks_in_top_40)
Artists(artist_id, artist_name, first_year_active)
Albums (album_id, album_name, artist_id, year_released, genre)
```

Write relational algebra expressions for the following queries:

2. Find the name of the artists who have albums of genre ‘pop’ and ‘rock’.

Relational Algebra Exercises

Consider the schema:

```
Songs (song_id, song_name, album_id, weeks_in_top_40)
Artists(artist_id, artist_name, first_year_active)
Albums (album_id, album_name, artist_id, year_released, genre)
```

Write relational algebra expressions for the following queries:

2. Find the name of the artists who have albums of genre ‘pop’ and ‘rock’.

$$\pi_{artists.artist\_name}((\sigma_{albums.genre\,=\,\text{‘pop’}}Albums) \bowtie Artists) \cap$$
$$\pi_{artists.artist\_name}((\sigma_{albums.genre\,=\,\text{‘rock’}}Albums) \bowtie Artists)$$

Relational Algebra Exercises

Consider the schema:

`Songs (song_id, song_name, album_id, weeks_in_top_40)`

`Artists(artist_id, artist_name, first_year_active)`

`Albums (album_id, album_name, artist_id, year_released, genre)`

Write relational algebra expressions for the following queries:

3. Find the id of the artists who have albums of genre ‘pop’ or have spent over 10 weeks in the top 40.

Relational Algebra Exercises

Consider the schema:

`Songs (song_id, song_name, album_id, weeks_in_top_40)`

`Artists(artist_id, artist_name, first_year_active)`

`Albums (album_id, album_name, artist_id, year_released, genre)`

Write relational algebra expressions for the following queries:

3. Find the id of the artists who have albums of genre ‘pop’ or have spent over 10 weeks in the top 40.

$$\pi_{artists.artist\_id}(\text{Artists} \bowtie (\sigma_{albums.genre\ =\ 'pop'}\text{Albums})) \cup$$
$$\pi_{albums.artist\_id}(\text{Albums} \bowtie (\sigma_{songs.weeks\_in\_top\_40\ >\ 10}\text{Songs}))$$

Relational Algebra Exercises

Consider the schema:

`Songs (song_id, song_name, album_id, weeks_in_top_40)`

`Artists(artist_id, artist_name, first_year_active)`

`Albums (album_id, album_name, artist_id, year_released, genre)`

Write relational algebra expressions for the following queries:

4. Find the names of all artists who do not have any albums.

Relational Algebra Exercises

Consider the schema:

```
Songs (song_id, song_name, album_id,
weeks_in_top_40)
Artists(artist_id, artist_name, first_year_active)
Albums (album_id, album_name, artist_id, year_released,
genre)
```

Write relational algebra expressions for the following queries:

4. Find the names of all artists who do not have any albums.

$$\pi_{artists.artist\_name}$$

$$(Artists \bowtie ((\pi_{artists.artist\_id} Artists) - (\pi_{albums.artist\_id} Albums)))$$

ER Diagrams

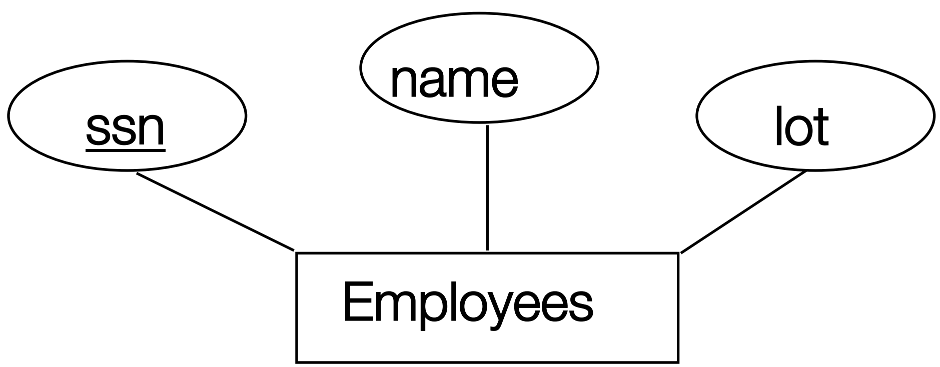
ER Diagrams

- Conceptual design of relations
  - Entities
  - Relationships
  - Attributes

Entities

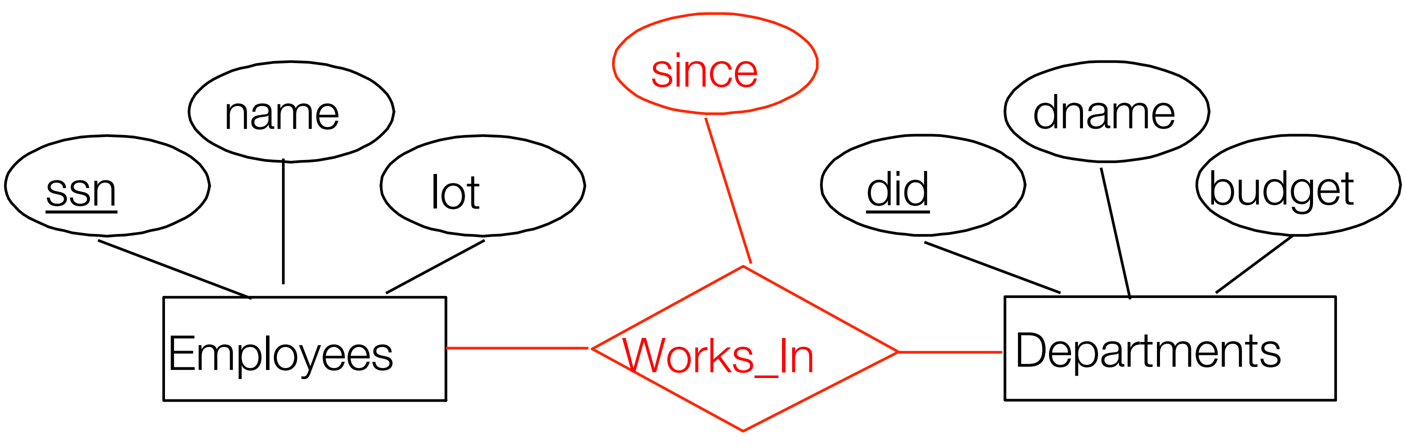
- Entity: a real world object described by a set of attributes

- Entity set: a collection of similar entities
  - E.g. all employees
  - Entities in entity set have same attributes
  - Has a key attribute
  - Each attribute has a domain

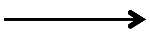


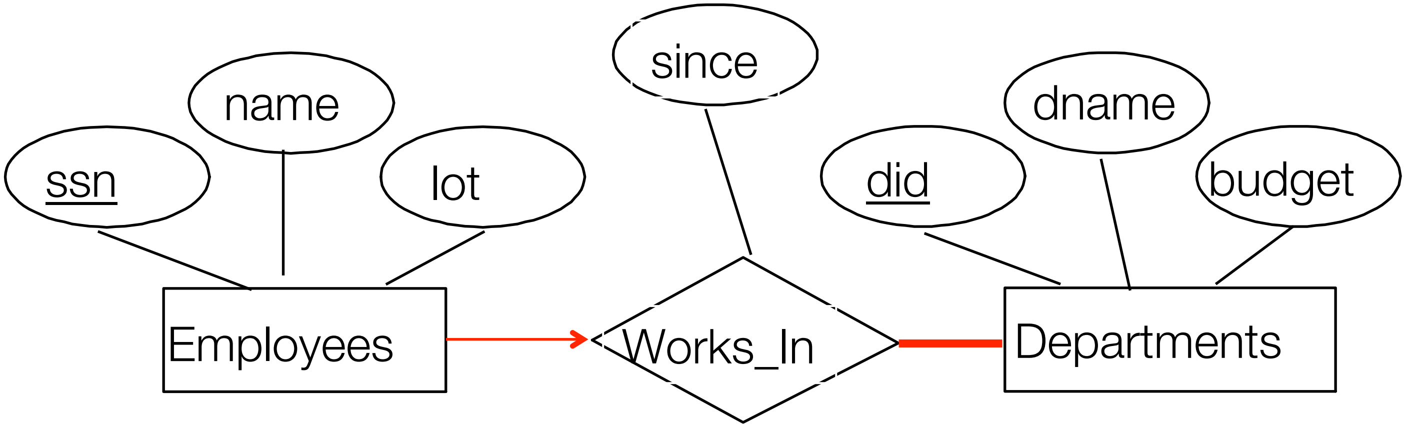
Relationships

- Relationship: association among two or more entities
  - Can have own attributes
- Relationship set: collection of similar relationships



Constraints

- Key constraint: entity participates at most once
  - Key, non-key
  - Represented by 
- Participation constraint: entity participates at least once
  - Total, partial
  - Represented by



Constraints

- Key constraint: entity participates at most once
  - Key, non-key
  - Represented by
- Participation constraint: entity participates at least once
  - Total, partial
  - Represented by

	Partial Participation	Total Participation
Non-Key	0 or more	1 or more
Key	0 or 1	1

Ternary Relations

- 3 entities connected to a relationship instead of 2

Weak Entity

- An entity that only makes sense in the context of another entity (its parent)
  - Has a partial key (dashed underline)

- E.g. there can be two songs with the same name
  - Key is (Artists.artist_id, Songs.song_name)

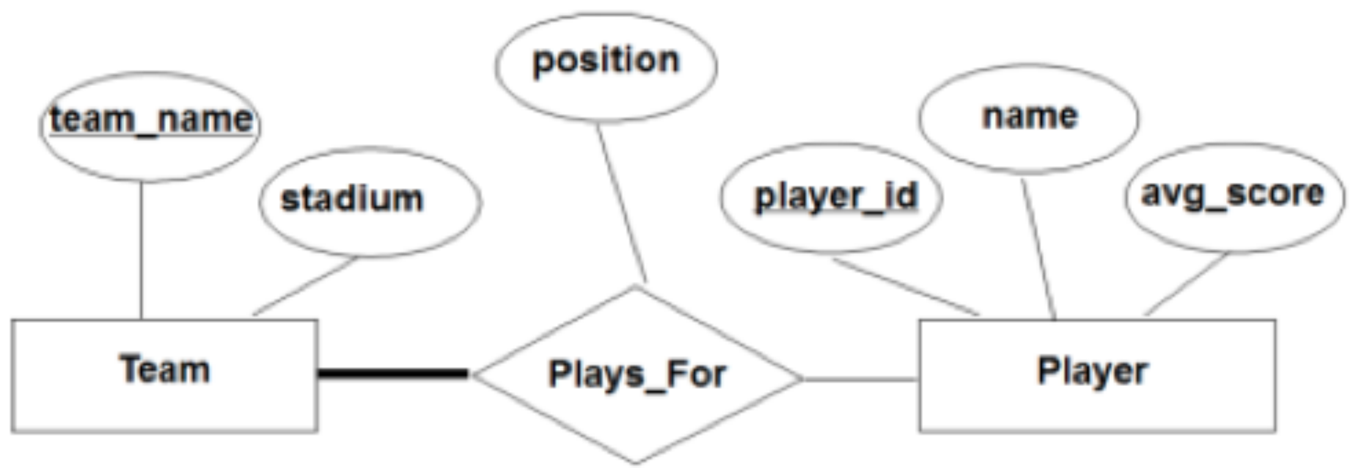
ER Diagram Exercises

ER Diagrams Exercises

1. Assume that a player can play in more than one team (Yes, our league has different rules!) and that a team needs at least one player. Draw an ER diagram for our database.

ER Diagrams Exercises

1. Assume that a player can play in more than one team (Yes, our league has different rules!) and that a team needs at least one player. Draw an ER diagram for our database.

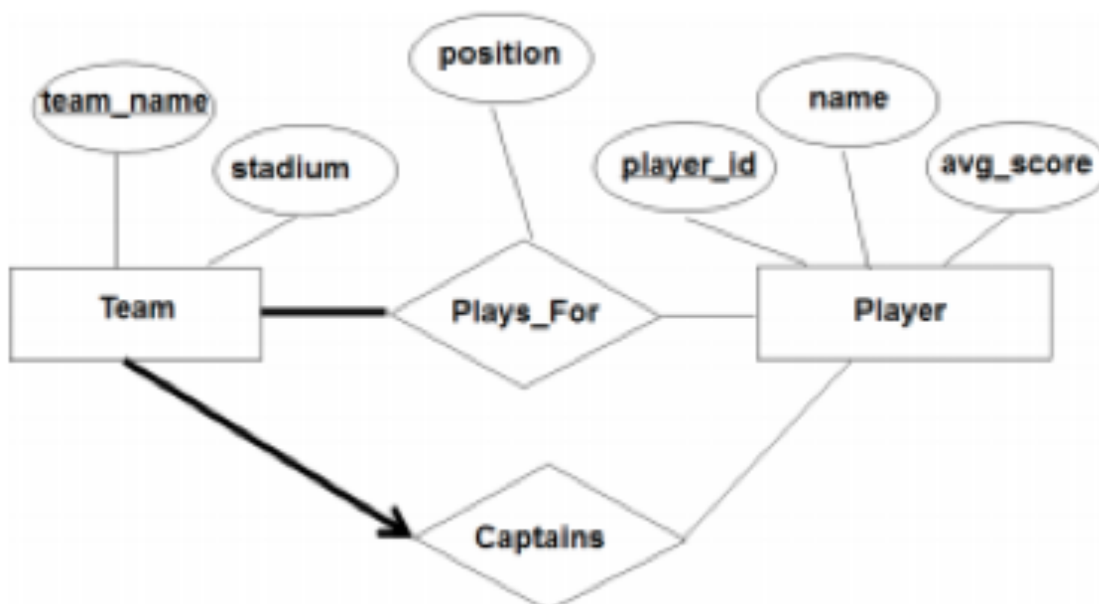


ER Diagrams Exercises

2. Now let’s say we want to also track who is the captain of every team. How will the ER diagram change from the previous case? Note: Every team needs exactly one captain!

ER Diagrams Exercises

2. Now let's say we want to also track who is the captain of every team. How will the ER diagram change from the previous case? Note: Every team needs exactly one captain!



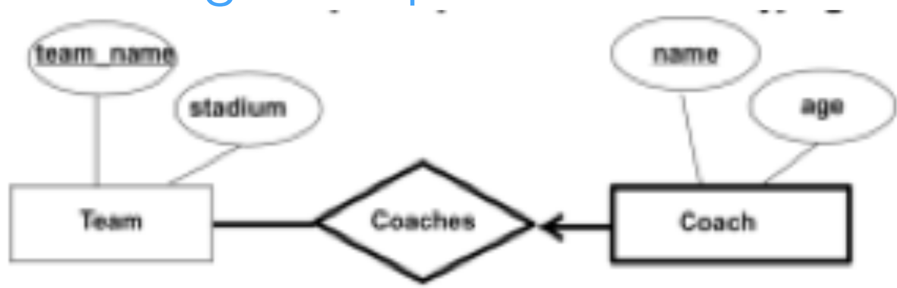
ER Diagrams Exercises

3. Are there any weak-entity relationships in either of our ER diagrams?

ER Diagrams Exercises

3. Are there any weak-entity relationships in either of our ER diagrams?

No. A weak entity can be identified uniquely only by considering the primary key of another (owner) entity. Consider the following example:



A team can have many coaches, but each coach exactly coaches one team. Coach is a weak-entity set and can be identified by its partial key “name”.

Functional Dependencies

Functional Dependencies

- Used to identify redundancy in schemas and suggest refinement
- $X \rightarrow Y$
  - X determines Y
- $K \rightarrow$ {all attributes of R}
  - K is a superkey of R

Armstrong's Axioms

- **Reflexivity:** If $X \supseteq Y$, then $X \rightarrow Y$
- **Augmentation:** If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - $XZ \rightarrow YZ$ does not mean $X \rightarrow Y$
- **Transitivity:** If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- **Union:** If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
- **Decomposition:** If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

Closures

- X+: set of all FDs implied by X, including trivial dependencies

```
X+ := X
while not done:
   for U→V in F:
      if U in X+:
         add V to X+
```

Boyce-Codd Normal Form (BCNF)

- R is in BCNF if the only non-trivial FDs over R are key constraints

- R with FDs F is in BCNF if

  for all $X \to A$ in F+:

  - $A \subseteq X$ (called a trivial FD), or
  - X is a superkey for R

BCNF Decomposition

- For R, if $X \rightarrow A$ violates BCNF, decompose R into $R - A$ and XA

Decomposition

- When decomposing R into X and Y…

- Lossless decomposition
  - $\pi_X(R) \bowtie \pi_Y(R) == R$
  - $X \cap Y \rightarrow X$ or $X \cap Y \rightarrow Y$

- Dependency preservation:
  - $F_X \cup F_Y = F+$

Minimal Cover

- Lossless join and dependency preserving decomposition
- G for a set of FDs F:
  - Closure of F = closure of G
  - Right hand side of each FD in G is a single attribute
  - As small as possible

Functional Dependencies Exercises

Functional Dependencies Exercises

1. Flight schema

Flights(<u>**F**light_no, **D**ate</u>, f**R**om, **T**o, **P**lane_id),
ForeignKey(**P**lane_id)

Planes(<u>**P**lane_id</u>, t**Y**pe)

Seat(<u>**S**eat_no, **P**lane_id</u>, **L**egroom), ForeignKey(**P**lane_id)

Find the set of functional dependencies.

Functional Dependencies Exercises

1. Flight schema

Flights(**F**light_no, **D**ate, f**R**om, **T**o, **P**lane_id),
ForeignKey(**P**lane_id)

Planes(**P**lane_id, t**Y**pe)

Seat(**S**eat_no, **P**lane_id, **L**egroom), ForeignKey(**P**lane_id)

Find the set of functional dependencies.

$FD \rightarrow RTP$

$P \rightarrow Y$

$SP \rightarrow L$

Functional Dependencies Exercises

2. Now consider the attribute set $R = ABCDE$ and the FD set $F = \{AB \rightarrow C, A \rightarrow D, D \rightarrow E, AC \rightarrow B\}$. Compute the closure for the following attributes.

A:

AB:

B:

D:

Functional Dependencies Exercises

2. Now consider the attribute set R = ABCDE and the FD set F = {$AB \rightarrow C$, $A \rightarrow D$, $D \rightarrow E$, $AC \rightarrow B$}. Compute the closure for the following attributes.

A: {A, D, E}

AB: {A, B, C, D, E}

B: {B}

D: {D, E}

Functional Dependencies Exercises

3. Decompose R = ABCDEFG into BCNF, given the functional dependency set:

F = {AB → CD, C → EF, G → A, G → F, CE → F}

Functional Dependencies Exercises

3. Decompose R = ABCDEFG into BCNF, given the functional dependency set:

F = {AB → CD, C → EF, G → A, G → F, CE → F}

AB→CD => decompose ABCDEFG into ABCD, ABEFG

G→A => decompose ABEFG into AG, BEFG

G→F => decompose BEFG into FG, BEG

Final relations: ABCD, AG, FG, BEG.

Functional Dependencies Exercises

4. Does the above decomposition preserve dependencies?

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$

Final relations: ABCD, AG, FG, BEG

Functional Dependencies Exercises

4. Does the above decomposition preserve dependencies?
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$

Final relations: ABCD, AG, FG, BEG

No, $C \rightarrow EF$ and $CE \rightarrow F$ are not represented in the closure of the union of each subrelation's dependencies

Functional Dependencies Exercises

5. Give a minimal cover for the original functional dependencies given: $F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$

Functional Dependencies Exercises

5. Give a minimal cover for the original functional dependencies given: $F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$

$AB \rightarrow C$

$AB \rightarrow D$

$C \rightarrow F$

$C \rightarrow E$

$G \rightarrow A$

$G \rightarrow F$