COL 362/632
Introduction To Database Management Systems

Data Modelling — Relational Model

Kaustubh Beedkar

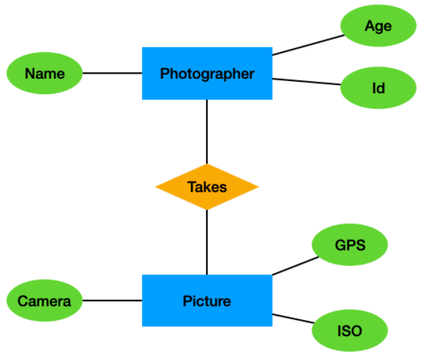
Administrivia

- No class on Friday 10.01.2025!
- We'll take attendance starting next week
  - Look out for announcement on Piazza tomorrow
  - Give your seating preference (we'll try out best to accommodate all requests)

Syllabus

Overview

1. Design and Programming
   - Data Modelling
     - ER Model
     - Relational Model
     - Schema Design
   - Relational Algebra
   - SQL (Basic + Advanced)
2. Implementation and Internals
3. Misc. Topics



From RealWorld to Database Schema

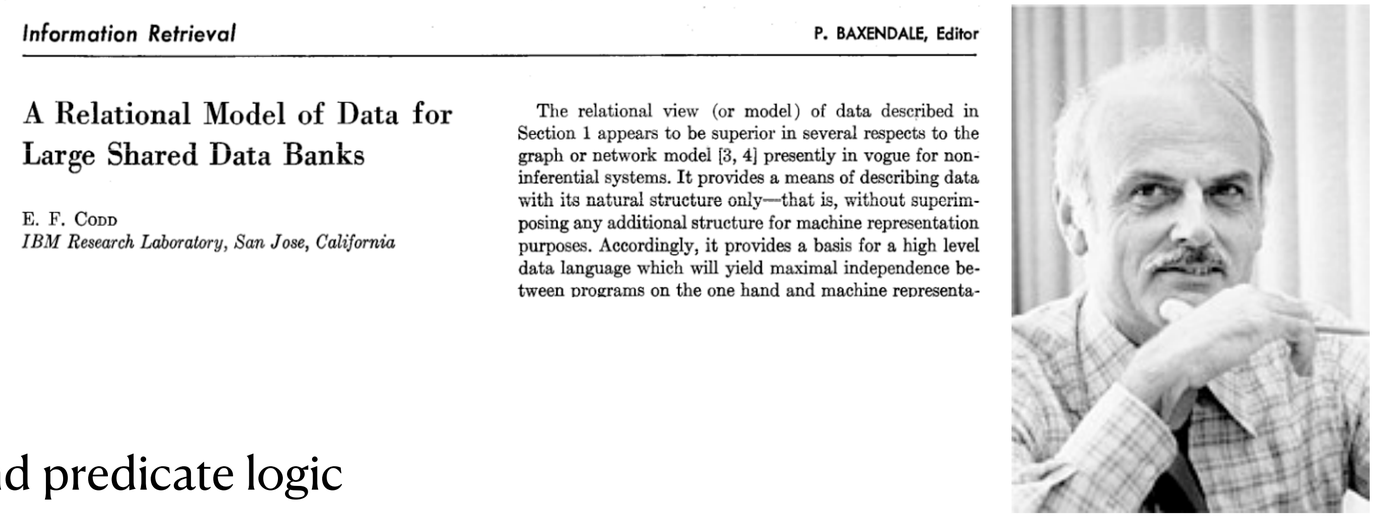
Database Schema

Photographer				Picture			
First Name	Last Name	Age	ID	Camera	GPS	ISO	Apperture

Photographer = {[Name: String, Age: Integer, ID: Integer]}
Picture = {[Camera: String, GPS: Point]}

Relational Model

F.Codd and the Relational Model



Information Retrieval P. BAXENDALE, Editor

A Relational Model of Data for Large Shared Data Banks

E. F. CODD
IBM Research Laboratory, San Jose, California

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representa-

Turing Award (1981)

- Proposed by Edgar Frank Codd in 1970
- Has just one concept that of **relation**
  - Mathematical concept based on set theory and predicate logic
  - — a set of tuples (rows) with defined attributes (columns)
  - — provides strong theoretical foundations
- A database table represents a relation, with rows as instances and columns as attributes
- Simple but very powerful concept
- Applicable across domains and forms the basis of modern database management systems

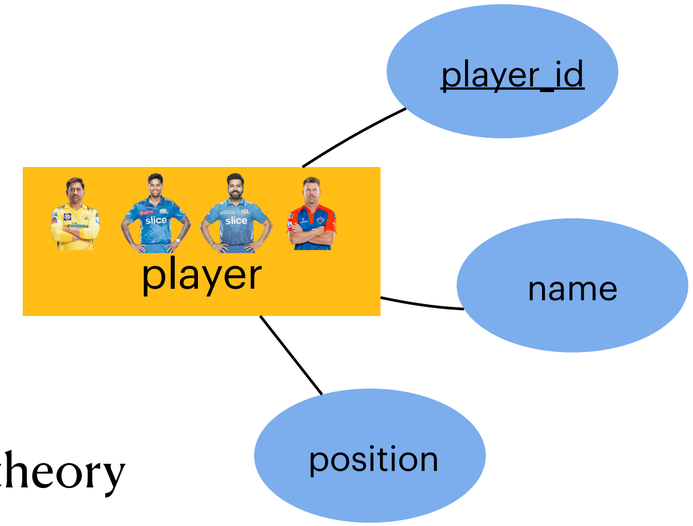
"The relational model is not just a way of organizing data; it's a way of thinking about data."

Relational Model and Its Mathematical Foundations

- $R \subseteq D_1 \times D_2 \times \ldots \times D_n$
- D_i is domain or range of values
  - Specifies the set of values that an attribute can take
- R is the expression or instance
- Example: Player ⊆ int x string x int
- **Relational Algebra:** formal query language for relational model based on set theory and predicate logic — more on this in the next lecture
- Schema of a relation = relation name + attribute name
- $[R] : \{[\text{attribute}_1 : D_1, \ldots, \text{attribute}_n : D_n]\}$
- $D_i = \text{dom}(\text{attribute}_i)$
- Example

[Player] : {[player_id:int, name:string, position:int]}

Player ⊆ dom(player_id) x dom(name) x dom(position)



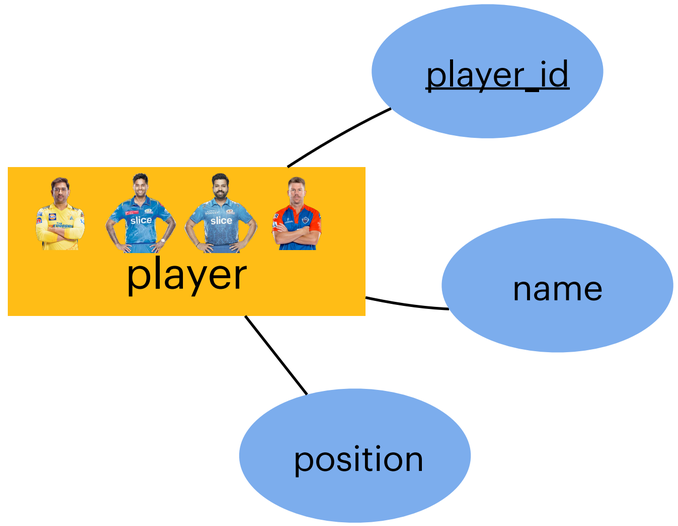
PlayerId	Name	Position
123	MS Dhoni	7
131	Rohit Sharma	1
134	SK Yadav	5
154	David Warner	3

Relational Model and Its Mathematical Foundations

Instance vs Schema

- Relation Player is an instance of the relation schema [Player]
- Instance is a set of records
- Example [Player] : {[player_id:int, name:string, position:int]}

  Player = {(123, MSD, 7), (134, SKY, 5), (131, RS, 1), (154, DW, 3)}



Tuple

- Also known as **record**
- $t \in R$
- Example Player = {(123, MSD, 7), (134, SKY, 5), (131, RS, 1), (154, DW, 3)}
  t = (134, SKY, 5)

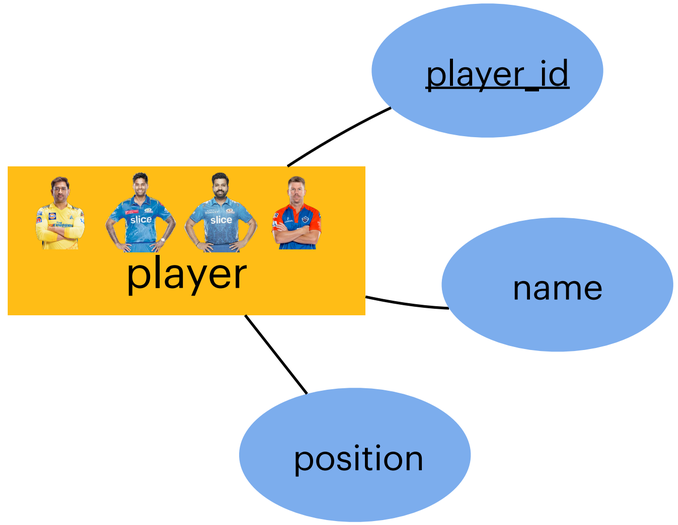
PlayerId	Name	Position
123	MS Dhoni	7
131	Rohit Sharma	1
134	SK Yadav	5
154	David Warner	3

ER Model to Relational Model

Mapping Entity set

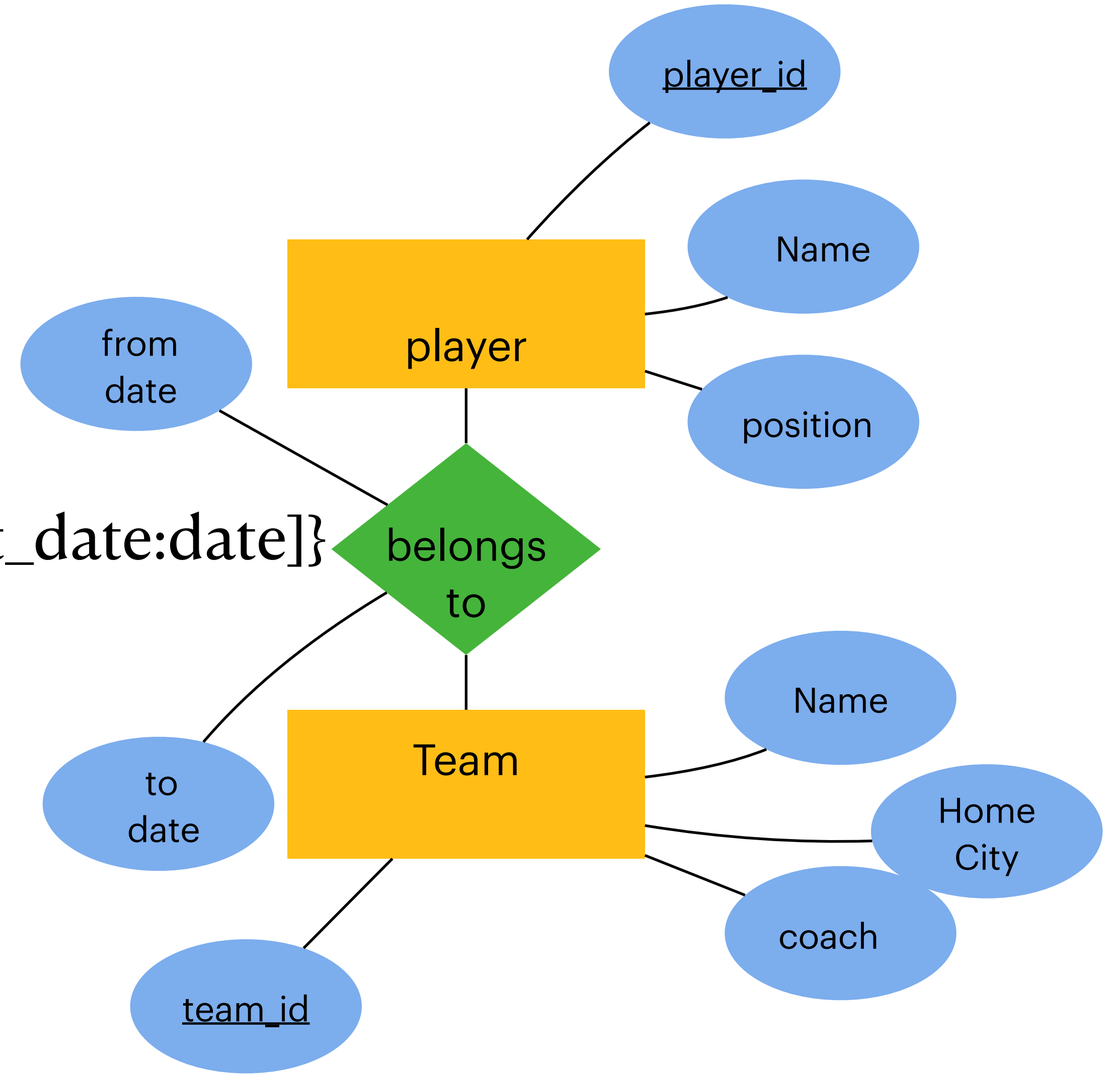
- Convert each strong entity type to a relation
- Include attributes and specify the (primary) key

- [Player]: {[player_id:int, name:string, position:int]}



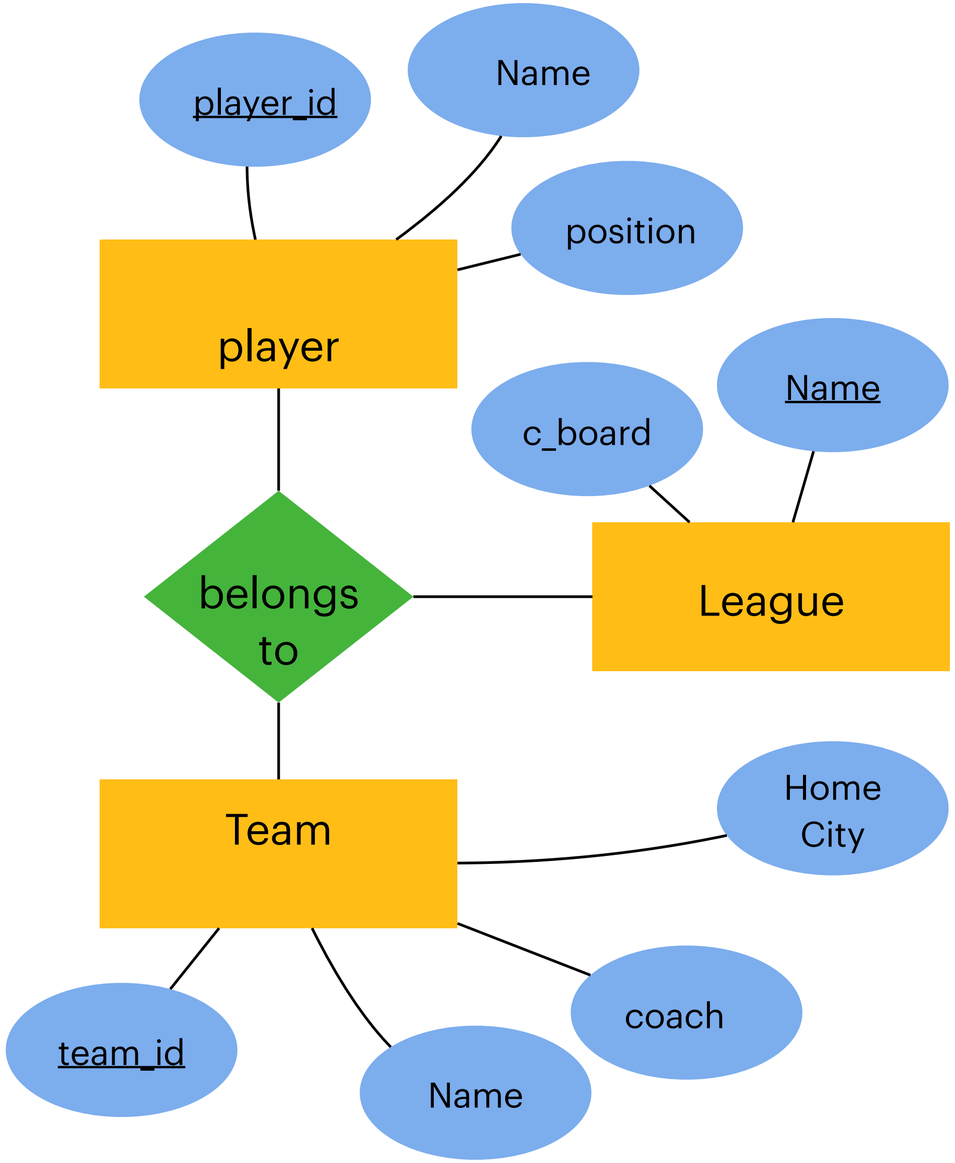
Mapping Relationship

- [Belongs]: {[player_id:int, team_id:int]}
- [Belongs]: {[player_id:int, team_id:int, f_date:date, t_date:date]}



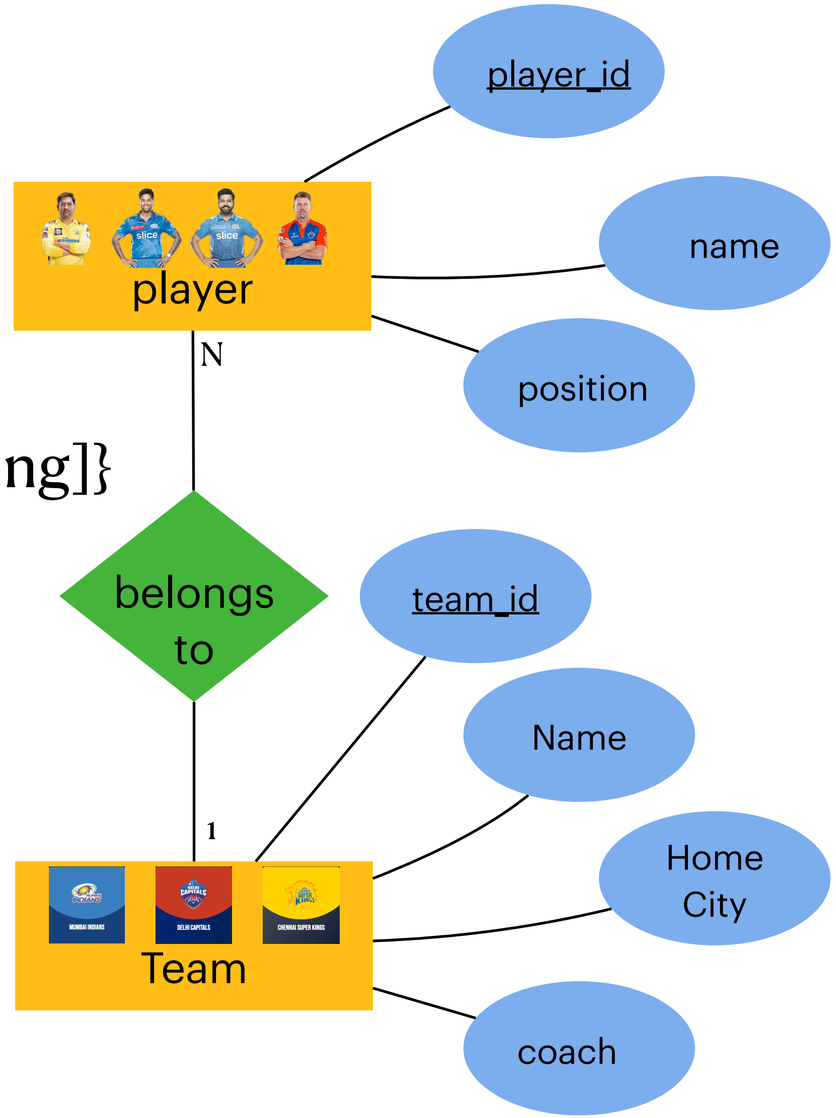
Mapping Multiway Relationship

- Create a new relation with (foreign) keys of all participating entities
- [Belongs]: {[player_id:int, team_id:int, l_name:string]}



1:N (Many-One) Relationship

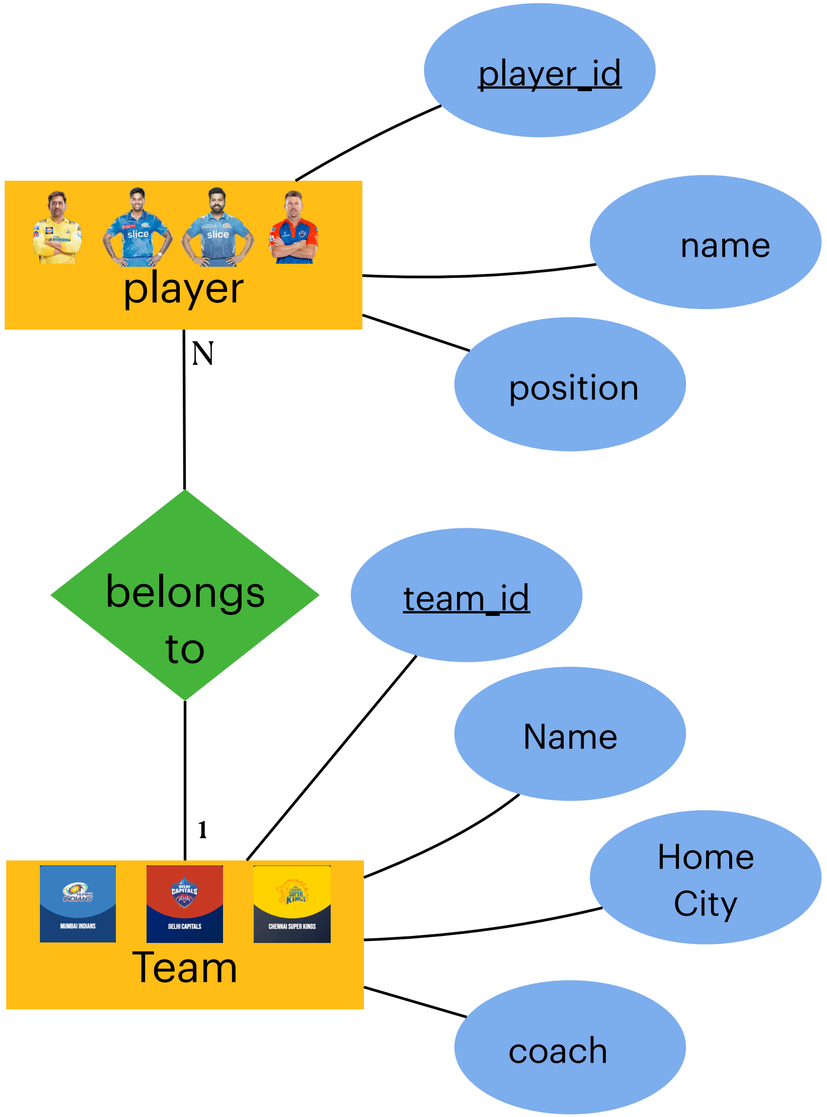
- [Player]: {[player_id:int, name:string, position:int]}
- [Belongs]: {[player_id:int, team_id:int]}
- [Team]: {[team_id:int, name:string, home:string, coach:string]}



1:N (Many-One) Relationship

- Modify the Player relation
- Add a (foreign) key to one of the participating relations

- [Player]: {[player_id:int, name:string, position:int, plays_for: int]}
- ~~[Belongs]: {[player_id:int, team_id:int]}~~
- [Team]: {[team_id:int, name:string, home:string, coach:string]}



Can we modify the Team relation instead?

Always add a (foreign) key to the relation on the “many” side

1:1 (One-One) Relationship

Add a (foreign) key to one of the participating relations

Option-1

[Player]: {[player_id:int, name:string, position:int, captain_of:int]}

[Team]: {[team_id, name:string, home:string, coach:string]}

Option-2

[Player]: {[player_id:int, name:string, position:int]}

[Team]: {[team_id:int, name:string, home:string, coach:string, captain_id:int]}

Which is better? (Option-2 is better as the option-1 will have a lot of null values)

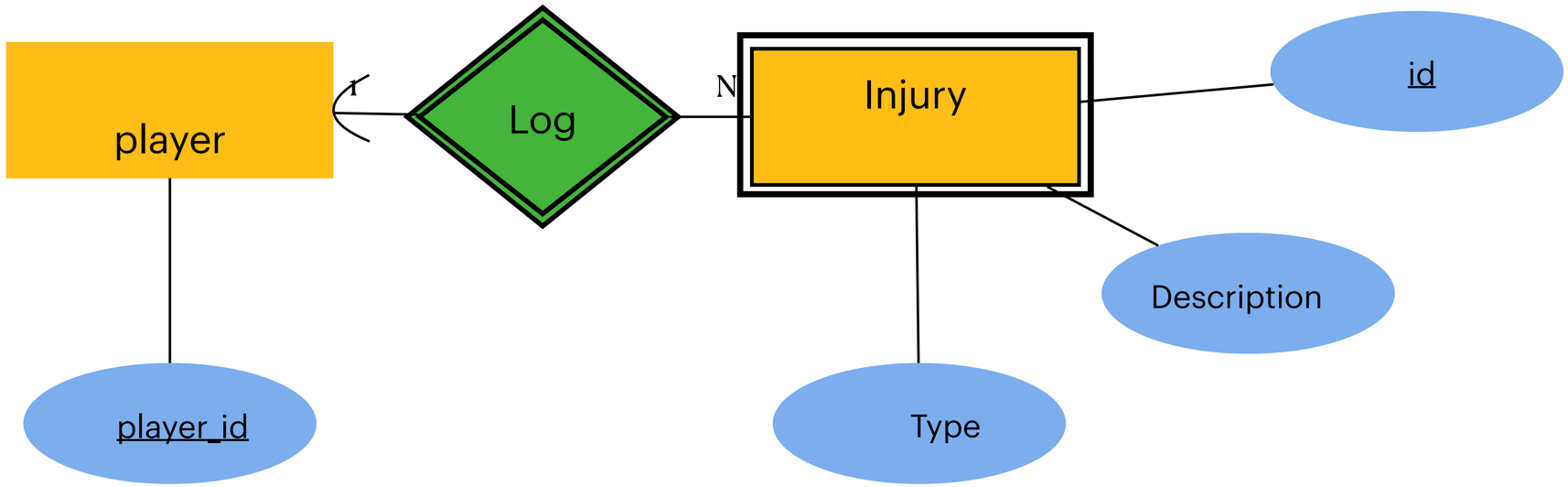
M:N (Many-Many) Relationship

- [Player]: {[player_id:int, name:string, position:int, plays_for:int, sponsored_by:int]}

Whats wrong here?

- **Always create a new relation with (foreign) keys referencing the participating entities**
- [Player]: {[player_id:int, name:string, position:int, plays_for:int]}
- [Sponsor]: {[player_id:int, company_id:int]}
- [Company]: {[company_id:int, name:string, manager_id:int]}

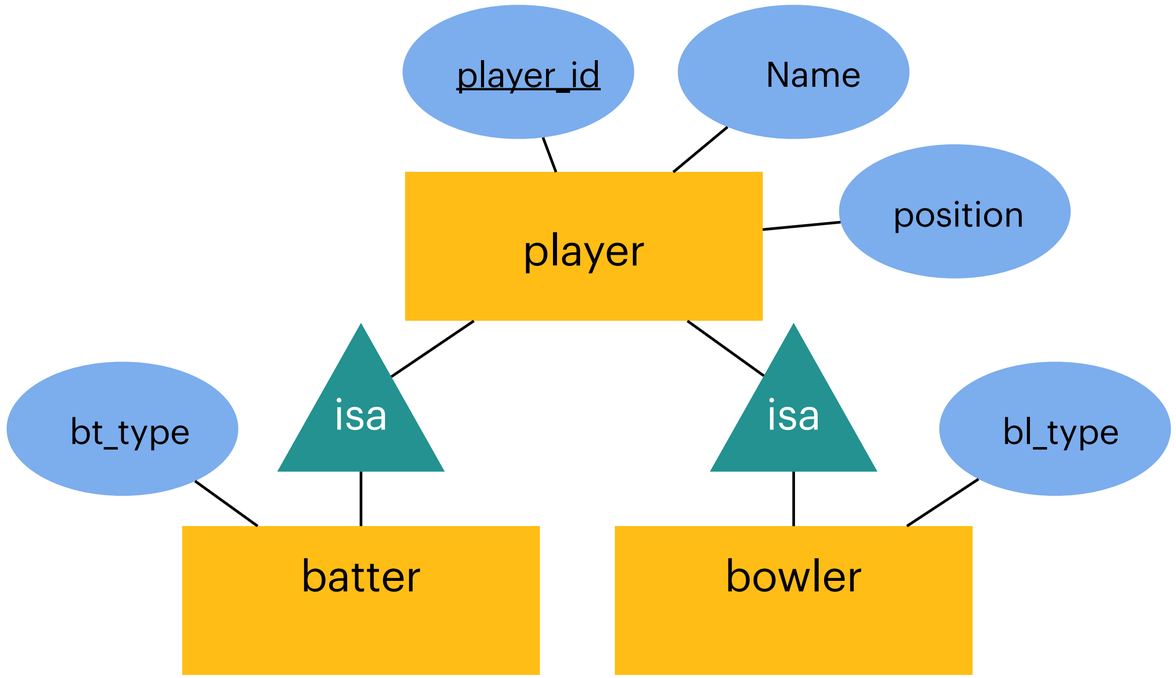
Mapping Weak Entity Sets



- Include (foreign) key referencing the related strong entity

- Combine primary key of the strong entity with weak entity discriminator
- [Injury]: {[id:int, player_id:int, type:string: description:string]}

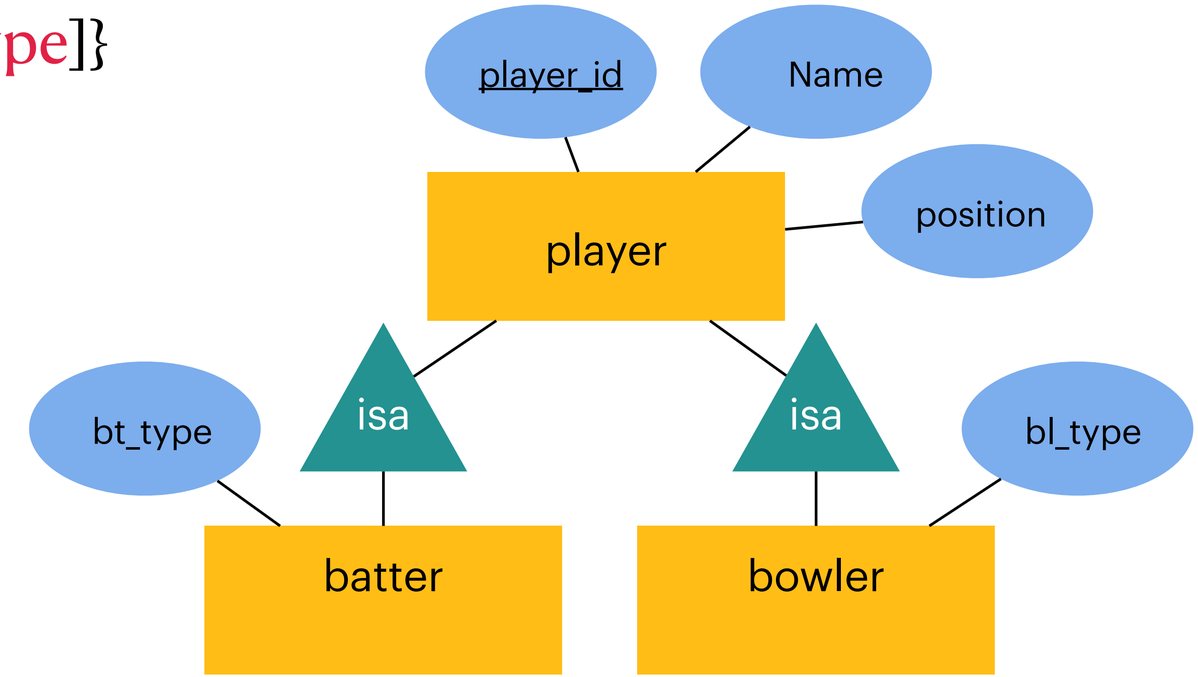
Mapping IsA Relationship



IsA Relationship

Option-1 (create one relation)

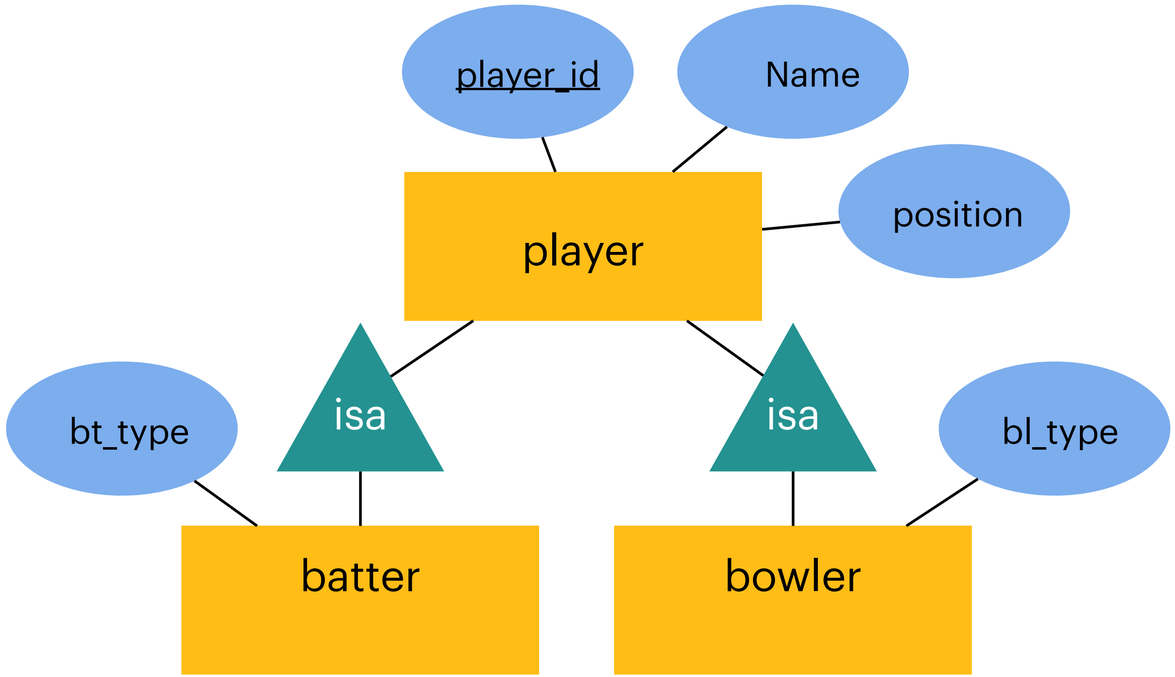
- [Player]: {[player_id, name, pos, bt_type, bl_type]}



IsA Relationship

Option-2

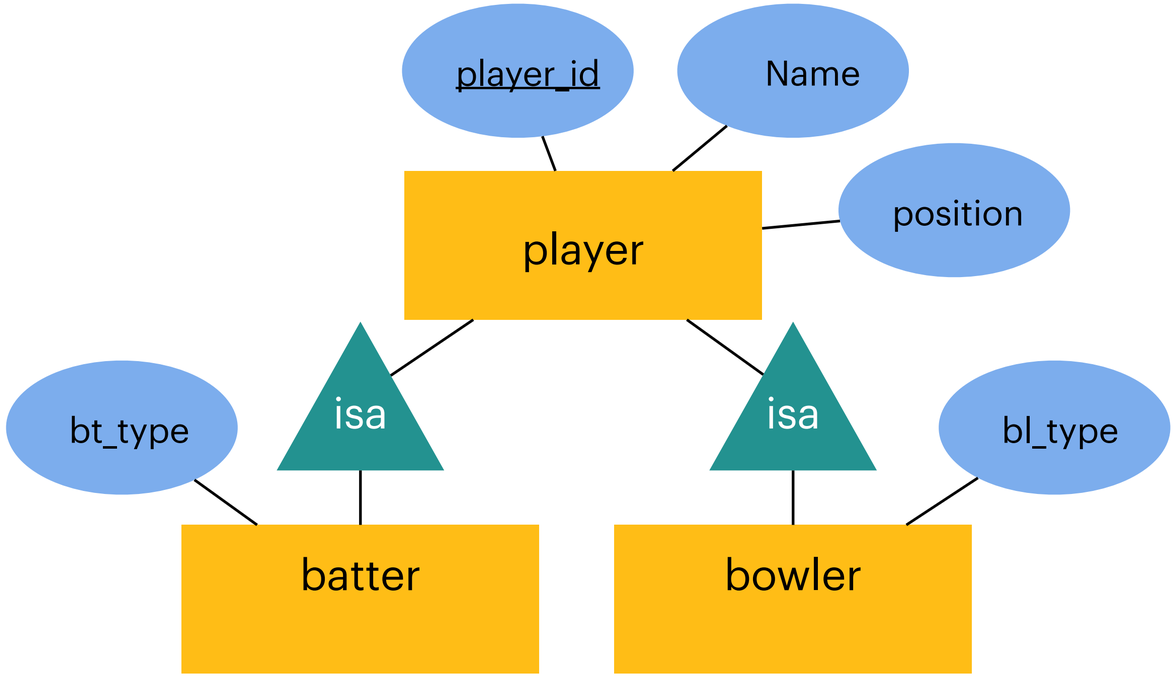
- [Player]: {[ ]}
- [Batter]: {[player_id, name, pos, bt_type]}
- [Bowler]: {[player_id, name, pos, bl_type]}



IsA Relationship

Option-3

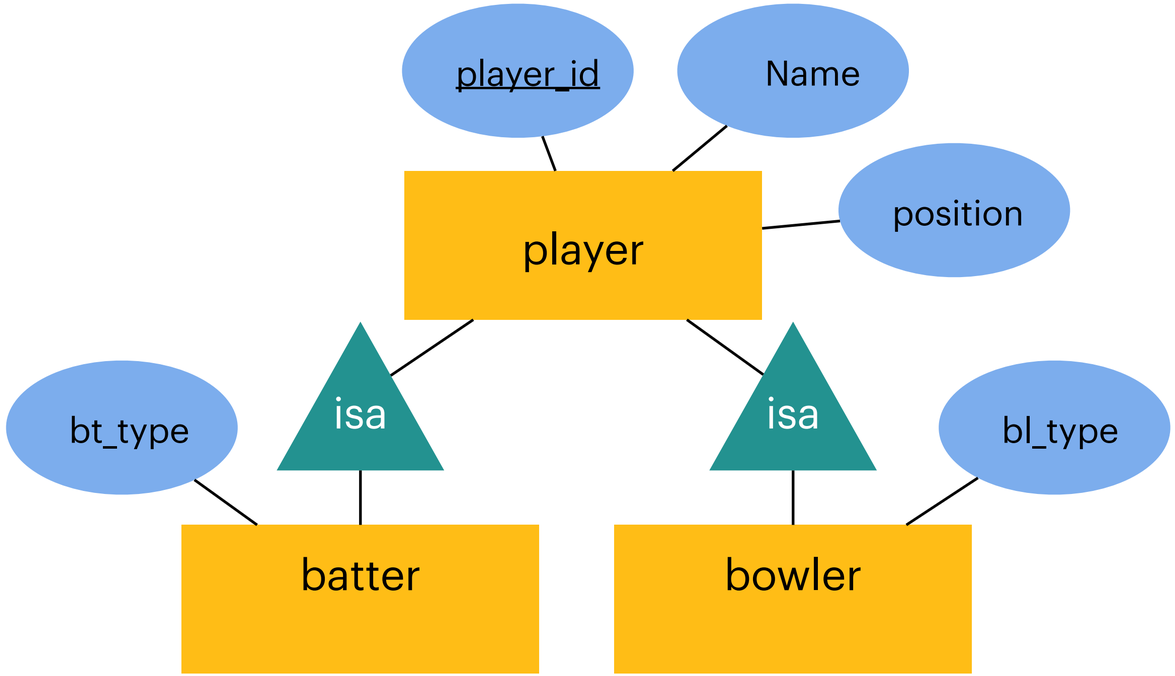
- [Player]: {[player_id, name, pos]}
- [Batter]: {[player_id, name, pos, bt_type]}
- [Bowler]: {[player_id, name, pos, bl_type]}



IsA Relationship

Option-4

- [Player]: {[player_id, name, pos]}
- [Batter]: {[player_id, bt_type]}
- [Bowler]: {[player_id, bl_type]}



Map to separate relations for the superclass and subclasses (option-4) or a single relation using a type attribute (option-1)

Relation vs Tables

Player =

(154, David Warner, 3, DC)
(123, MS Dhoni, 7, CSK)
(134, SK Yadav, 5, MI)
(131, Rohit Sharma, 1, MI)
(193, Virat Kohli, 3, RCB)

Player

PlayerId	Name	Position	PlaysFor
123	MS Dhoni	7	CSK
193	Virat Kohli	3	RCB
131	Rohit Sharma	1	MI
134	SK Yadav	5	MI
154	David Warner	3	DC

- Mathematical concept
- Tuples have no inherent order (unordered set)
- No duplicates (its a set!)
- Does not formally account for nulls

- Structure in a database to store data
- Rows have order (but does affect results)
- May allow duplicates (unless constrained)
- Often allow null values (missing data)

Summary

- Relational model fundamentals
- ER to relational mapping
  - Entities become relations with their attributes
  - Relationships are translated based on their cardinalities and participation
- Relational Model underpins modern databases, enabling efficient querying, data consistency, and scalability
- Next Lecture: Relational algebra — the query language of the relational model