ARIES

1. Your database server has just crashed due to a power outage. You boot it up, find the following log and checkpoint information on disk, and begin the recovery process. Assume we use a STEAL/NO FORCE recovery policy.

LSN	Record	prevLSN
30	update: T3 writes P5	null
40	update: T4 writes P1	null
50	update: T4 writes P5	40
60	update: T2 writes P5	null
70	update: T1 writes P2	null
80	begin checkpoint	-
90	update: T1 writes P3	70
100	end checkpoint	-
110	update: T2 writes P3	60
120	T2 commit	110
130	update: T4 writes P1	50
140	T2 end	120
150	T4 abort	130
160	update: T5 writes P2	null
180	CLR: undo T4 LSN 130	150

Transaction table at time of checkpoint

Transaction	lastLSN	Status
T1	70	Running
T2	60	Running
T3	30	Running
T4	50	Running

Dirty Page Table at time of checkpoint

Page ID	recLSN
P5	50
P1	40

(a) The log record at LSN 60 says that transaction 2 updated page 5. Was this update to page 5 successfully written to disk? The log record at LSN 70 says that transaction 1 updated page 2. Was this update to page 2 successfully written to disk?

Update at LSN 60 MAY have been written to disk. The log entry was flushed before the write itself. IT was not yet flushed at the time of the checkpoint, but may have flushed later. Update at LSN 70 was flushed to disk because P2 was not in the dirty page table at the time of the checkpoint.

(b) At the end of the analysis phase, what transactions will be in the transaction table, and what pages will be in the dirty page table?

Transaction Table			**Dirty Page Table**	
Transaction	**lastLSN**	**Status**	**Page ID**	**recLSN**
T1	90	Running	P1	40
T3	30	Running	P2	160
T4	180	Aborting	P3	90
T5	160	Running	P5	50

(c) At which LSN in the log should redo begin? Which log records will be redone (list their LSNs)? All other log records will be skipped.

Start from 40, the smallest LSN in the dirty page table. The following should be redone: 40, 50, 60, [80], 90, [100], 110, [120], 130, [140], [150], 160, 180 30 is skipped because 30 < 40. 70 is skipped because P2.recLSN = 160 > 70.

2. Consider the execution of the ARIES recovery algorithm given the following log (assume a checkpoint is completed before LSN 0 and the Dirty Page Table and Transaction Table for that checkpoint are empty):

LSN	**Log Record**
10	update: T1 writes P1
20	update: T2 writes P3
30	T1 commit
40	update: T3 writes P4
50	update: T2 writes P1
60	T1 end
70	update: T3 writes P2
80	T2 abort
X - crash, restart	

(a) During Analysis phase, what log records are read? What are the contents of the dirty page table and the translation table at the end of the analysis stage?

All records (since last checkpoint) are read.

Transaction Table			**Dirty Page Table**	
TX	**lastLSN**	**Status**	**Page ID**	**recLSN**
T3	70	Running	P1	10
T2	80	Aborting	P2	70
			P3	20
			P4	40

(b) During Redo, what log records are read? What data pages are read? What operations are redone (assuming no updates made it out to disk before the crash)?

Redo starts at LSN 10 (smallest recLSN). All pages in the dirty page table are read from disk (i.e. P1, P2, P3, P4). Assuming no updates made it to disk, all updates are redone. So 10, 20, 40, 50, 70.

(c) During Undo, what log records are read? What operations are undone? Show any new log records that are written for CLRs. Be sure to show the undoNextLSN?

Starting from 80 and 70, we will read 80, 70, 50, 40, 20.

Undone operation will be 70, 50, 40, 20.

New log records are:

100 CLR T3 LSN = 70; undoNextLSN = 40

110 CLR T2 LSN = 50; undoNextLSN = 20

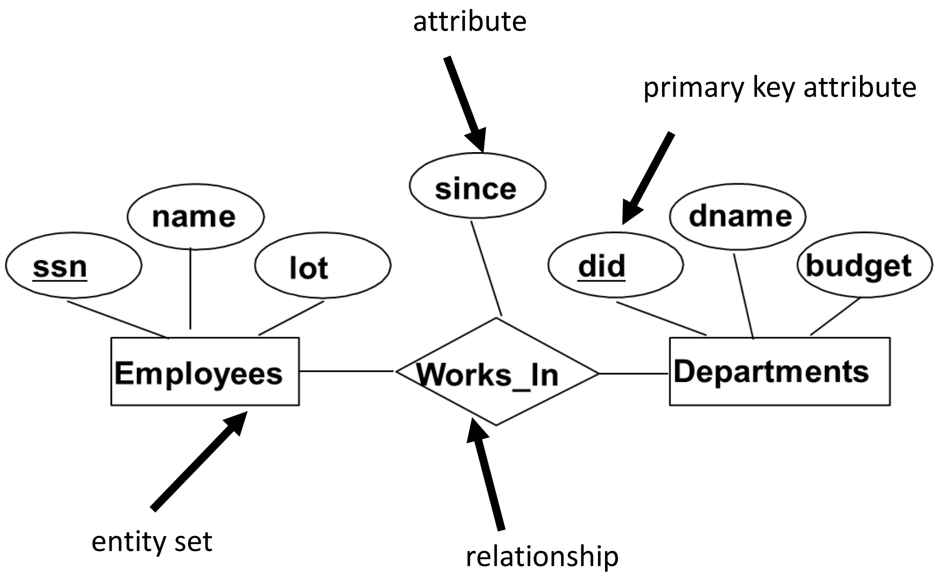
120 CLR T3 LSN = 40; undoNextLSN = null

130 T3 end

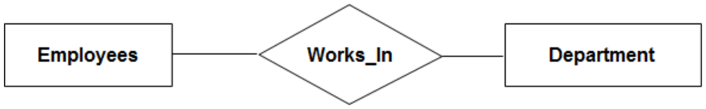
140 CLR T2 LSN = 20; undoNextLSN = null

150 T2 end

Entity Relationship Diagrams

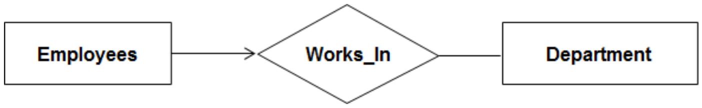


1. Normal Line ——— *Entity may participate in a relationship zero or more times*



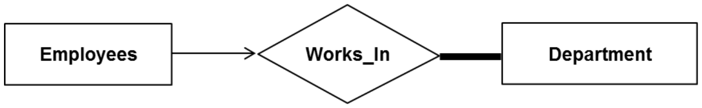
An employee may work in zero or more departments. A department may have zero or more employees in it.

2. Normal arrow (aka key constraint) ⟶ *Entity may participate in relationship at most one time*



An employee may work in zero or exactly one department.

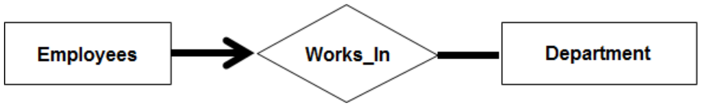
3. Bolded line (aka total participation constraint) ▬ *Entity must participate in relationship at least once*



Each department must have at least one employee (but may have several)

4. Bolded arrow (aka total participation with key constraint) ➔

Entity must participate in relationship exactly once



Each employee must work in exactly one department

Design a schema!

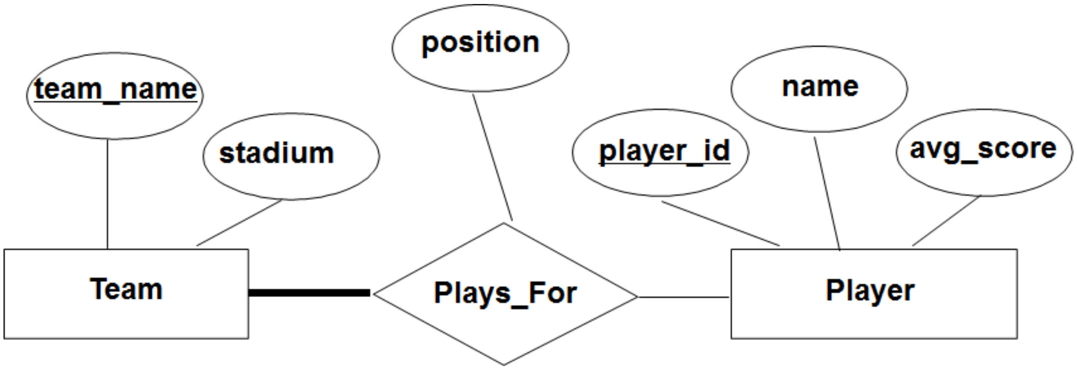
1. We want to store sports teams and their players in our database. So let's first make an Entity-Relationship diagram!

Every **Team** in our database will have a team_name and a stadium where they play their games.

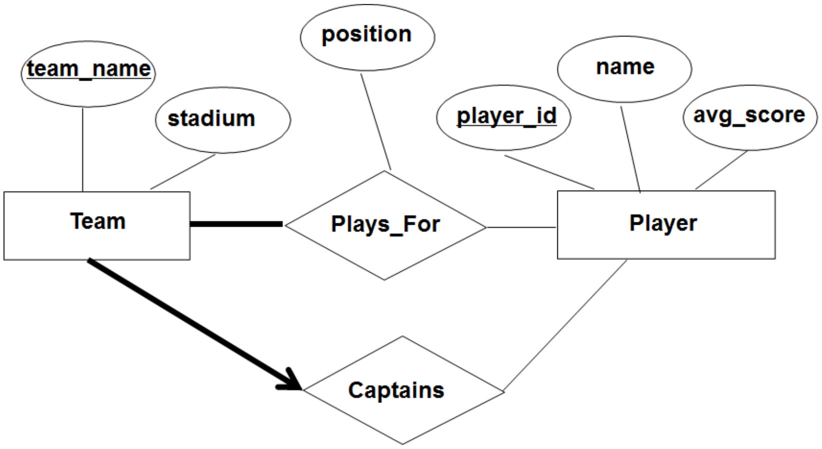
Each **Player** will have a player_id, name and their average score (This can be used for any sport !).

Finally our database will contain who **Plays_For** which team and also the "position" that the player plays in.

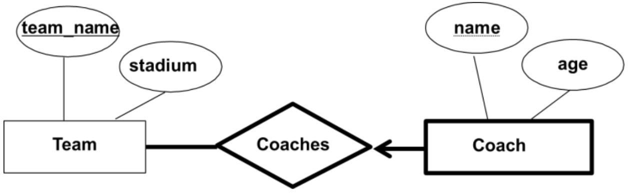
a. Assume that a player can play in any number of teams (Yes, our league has different rules !) and that a team should have **at least** one player. Draw an ER diagram for our database



b. 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 to have exactly one captain!



c. Are there are any weak-entity relationships in either of our ER diagrams?
No. Definition: A *weak entity* can be identified uniquely only with the primary key of *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.



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"