Section A:

1)

1. // lol u wat? football? I don’t football the fk is football
   1. Put circle that has the player age, dunno what people give 2 fks about in football matches. Salary maybe? how they make 103013i5015M for just running behind a ball?   
      for participates put a circle that says
   2. Maximum\_capacity, and set constraint that attendance can’t be bigger than maximum\_capcity.
   3. put circle on top of the playsorplayedfor relationship and say duration which is derived from start date and end date
   4. put a relationship between team and stadium that says owns.   
      DONE
2. 1. not relevant anymore
   2. not relevant anymore
3. 1. I agree with aht she is trying to say but she probably meant to say scaling out as opposite to scaling up. No need to improve on current hardware just buy more of the same and plug it in, since our DB designed well with good backend, of course since you are on the team, then plugging more computers to this cluster is matter of seconds
   2. not relevant anymore

Section B:

2)

1. not relevant anymore. But statement is true, and that is due to how IMDBs are constructed they use less CPUs cycle thus computing faster
2. false, heap is great for quick inserts, but updating and deleting are really slow since we need to retrieve the record by going linearly through the input and update it, for deleting we need to retrieve delete it and mark the space as empty, thus creating run times of O(N)
3. Designing a trigger requires to specify an event yes, no need for condition and can do no action but print results so it is only require a trigger to specify an event.
4. Yes. as it keeps all records directly references so accessing queries will be just simple dereferencing the pointers to the physical location of the record
5. Those information are stored in the log file which is used by some of the recovery mechanism to ensure that recovery is done properly without putting the db in inconsistent state.

Section C:

3)

1. 1. SELECT DISTINCT c.dept\_name  
      FROM course c
   2. Missing GROUP BY name to make sure all
   3. SELECT d.dept\_name, bugdet  
      FROM department d  
      ORDERED BY budget DESC
   4. solution := π(student.name, course.course\_id, takes.grade)(σ(student.ID=takes.ID, course.course\_id = takes.course\_id, takes.year = 2010)(student x course x takes))
   5. SELECT insct.name, std.name  
      FROM instructor insct   
      JOIN teaches tech  
       on insct.ID = tech.course\_id  
      JOIN takes tak  
       on tech.course\_id = tak.ID  
      JOIN student std  
       on tak.ID = std.ID
   6. relationship type, between student and instructor, using lookup table (relationship relation), need refining I don’t know what else he wants
2. [Picture](https://drive.google.com/open?id=0BxSfLz74ghGpeFdsYnpBS2R2LUk)
3. reading week mandatory reading go enjoy reading that for those 2 marks meanwhile I will be doing 3 other past papers :)

4)

1. 1. no, in table T2 reads A after T1 modifies it, if T1 doesn’t commit due to abort and rollback to its previous state the result in T2 is still the old one thus creating inconsistent database.
   2. Same issue as i, T2 reds A after it was written by T1, when a process writes on a data that other process needs to read it the first process needs to commit else we will end up with inconsistent and not recoverable state
   3. Serializability is the process of finding non-serial schedules that allow transactions to execute concurrently without interfering with one another.
   4. with two-phase locking protocol it deny access to other transactions and prevent incorrect updates thus resulting in serializability (because if it a transaction is denied access it cannot interfere with another one)   
      [Picture](https://drive.google.com/open?id=0BxSfLz74ghGpbXV0WHhadEJfSTQ) NOTE: I made a mistake you should request write\_lock at t1 no need for read lock
   5. Deadlock might happen, livelock, and cascading rollbacks.   
      Example are in lecture slide 48 of lec 10
2. Author (name, URL)  
   Book (ISBN, title, year, price)  
   Warehouse(Code, address, phone)   
   Shopping\_basket(basket\_id)  
   Customer(e-mail, name, address, phone)  
   written\_by(authour\_name [fk: Author.name), book\_isbn[fk:Book.ISBN])  
   Stocks(book\_isbn[fk:Book.ISBN], warehouse\_code [fk:Warehouse.code], number)  
   Contains (shopping\_id [fk:shopping\_basket.basket\_id], book\_id [fk:Book.ISBN], number)  
   basket\_of (customer\_email[fk:customer.email], shopping\_id [fk:shopping\_basket.basket\_id])
3. Good quality ER model will eliminate a lot of redundancy in the database, thus leaving us to use normalization just to ensure that it is of best possible quality.  
   deriving a relational schema from scratch would require various steps of decomposing from 1NF -> 2NF -> 3NF and possibly BCNF. normalization will result in best quality representation of the database as it is follow logical approach while ER model is conceptual and relies on the good design and decision making of the designer.