**In your own words answer the following questions:**

1. What is a (database) transaction?
2. A Database transaction a single unit of actions or multiple logic processes that can possibly modify or retrieve data from a database.
3. Why do we need (database) transactions?
4. Transactions are useful in databases when there is a need to store or modify data in multiple rows of a table or multiple tables. Using a transaction to perform all these processes in a single action allows for data to only be changed or modified if all the processes complete successfully.
5. What needs to happen when (whether due to a technical error, database constraint, business rule or other reason) a transaction can only be partially completed and why does this need to happen?
6. When a Transaction fails to complete all its processes due to any error these errors are caught using a Try Catch block of logical code. If by chance an error is caught the transaction can then Rollback any changes made.
7. What keyword/ action tells a database that a transaction has been successfully completed and the results of the transaction can be made permanent in the database?
8. The keyword to make any changes a transaction has a applied to a database is COMMIT TRANS.
9. Give an example of a business operation that might require a database transaction (other than those given in the lecture) and explain WHY it would need the transaction
10. An example of this would be if a sport club were to remove a Player from a team. The transaction would have to update multiple tables. The player would need to be removed from the members table also the Teams table. The players playing number would also have to be marked as available so would his playing position. Due to the need to modify data in multiple tables it would suit being done in a single transaction.
11. What does the acronym ACID stand for (in database terms)?

Provide a brief explanation of the meaning of each word in the Acronym.

1. The Acronym ACID in Database terms stands for Atomic, Consistent, Isolated and Durable.

* **Atomicity** ensures that each Transaction is handled as a Single Unit that either fails or succeeds. If one of the statements happens to fail, then they all fail.
* **Consistency** ensures that a database can only go from one State to Another. Any data or modification to data in a database must adhere to all the rules and constraints pre-defined within the database.
* **Isolation** means that a transaction is isolated from another transaction that is being performed concurrently. Isolation ensures that the database is in the same state when concurrent transactions are being processed as it would have been if the transactions had been executed sequentially.
* **Durability** ensures that once a transaction has been COMMITED it will remain in that state. This means that the changes made have been written to Non-Volatile memory and will remain that way until another transaction attempts to make any changes.