Disaster Recovery Key takeaways

- Disaster Recovery core concepts you've learned in this lesson:
  - Most of enterprises that suffer a disaster, not having implemented a disaster recovery plan, are doomed to disappear. That's why a DR plan is always needed.
  - When thinking about disasters, don't think only in headline-grabbing events. The most likely to occur (and they will) are hardware faults, data corruptions, power outages, etc ...
  - DR is part of the Business Continuity plan, that covers DR plans, business recovery plan and emergency response plans.
  - When elaborating a DR plan, we should pay attention to: RTO/RPO, **Cost**, Application transparency, Performance impact and administrative complexity. As RTO and RPO values have a direct impact on the costs of the DR plan, they should be carefully fixed in each case. Administrative complexity must be minimized, as it impacts costs as well, and as a high administrative complexity will make things difficult to test and implement.
  - Technical concerns: pay attention on slide 13
  - Business concerns: pay attention on slide 14

- Disaster recovery implementing a standby database:
  - Data changes are propagated from a primary to a standby database through redologs. Therefore the standby database is a *physical clone of the primary database*. The way we will propagate the changes will determine how synchronized will be the standby database at any moment.
  - To ensure a real DR plan, the standby database must be at a different location than the primary database, usually in **a second data center**. In Europe, companies usually use data centers in the same metropolitan area, sometimes in distinct metropolitan areas, distant of a few hundreds of kilometers.
  - For companies that cannot afford a second Data Centers, a DR solution in the Cloud might be a good solution. It addresses the need of a second (or third) data center, without the huge infrastructure costs associated.
  - There are three protection mode in this DR architecture:
    - SYNC: data written on the primary database is not committed until it has been written to the remote (standby) redologs. This protection mode is the only one that guarantees **RPO=0**, *as primary database will shutdown if it cannot propagate the changes to the standby site*. This protection mode is very sensitive to network latency between primary and standby databases, hence can't be implemented if the two sites are not in the same metropolitan area (latency between the sites <= 5ms).
    - ASYNC: data written on the primary is committed, and the data is propagated asynchronously to the standby redologs. If the changes can't be propagated for any reason, it will be re-intended later, and a lag appears between the primary and the standby database. This means that we can't guarantee RPO=0 with ASYNC protection mode. ASYNC mode is not sensitive to network latency, but we must be aware that high latencies between the primary and the standby sites may cause a lag that will increase RPO.
    - HYBRID: SYNC by default, ASYNC whenever SYNC is not possible for any reason. If the protection mode has been degraded to ASYNC, the system tries to get back to SYNC mode.
    - Be aware of the distinct RPO and RTO that we can expect from each protection mode (slide 19).

- Implementing and using a DR based on a primary-standby database architecture:
  - Three possible actions:
    - *Switchover:* this is a role permutation, where primary and standby databases permute their respective roles. It's an ordered maneuver, used for planned maintenance on the primary site. **No data is lost during a switchover maneuver**.
    - *Failover*: is the *emergency maneuver* done when primary database has been lost or is unusable. The standby database becomes the new primary, and the former primary needs to be re-constructed: that's the *re-instate* action, discussed below. Data might be lost during a failover operation, unless DR was configured in SYNC mode.
    - *Re-instate:* this is the reconstruction of the standby database from a primary database. This must be done to re-create a standby database after a failover maneuver.
- Disaster recovery best practices:
  - You need to think bigger than the database: think about *Business Continuity*, and design a DR for your apps as well.
  - Design your DR strategy to meet RTO/RPO: *not all the databases need the same DR solution*.
  - Automate as much as possible the maneuver steps. Implement and test, ensure you'll meet RTO and RPO in any case.
  - In production, plan and execute switchover maneuvers from time to time to ensure your Business Continuity plan still works.