Caching:

- Storing frequently accessed data in a faster cache to reduce access latency.
- Enhances system performance by minimizing data retrieval time and redundancy.

Cache Miss:

- Occurs when a requested piece of data is not found in the cache.
- Leads to fetching the data from a slower main memory or storage.
- Cache misses result in increased access latency and reduced performance.

Cache Hit:

- Happens when a requested data item is found in the cache.
- Allows the system to quickly retrieve data without accessing slower storage.
- Cache hits enhance system performance by reducing data access time.

Cache Invalidation:

- Process of marking cached data as invalid or outdated.
- Needed to ensure that stale data isn't used from the cache.
- Occurs when data is modified in main memory or when it becomes irrelevant due to changes in the system.

Cache Eviction Policy:

- Determines which item to remove when the cache is full.
- Needed due to limited cache size and impacts cache hit rates and performance.

Cache Eviction Policies:

- **LRU** (Least Recently Used): Evicts least recently accessed item. Simple but may not predict future access patterns.
- **LFU** (Least Frequently Used): Evicts least frequently accessed item. Suitable for varying access frequencies.
- **FIFO** (First-In-First-Out): Evicts oldest item. Simple but may ignore access frequencies.
- **Random Replacement:** Evicts randomly. Simple but not always optimal.
- **MRU** (Most Recently Used): Evicts most recently accessed. Prioritizes recent data.

Cache Write Strategy:

- Defines how write operations are managed between cache and main memory.
- Ensures data consistency and impacts performance and data availability.

Cache Write Strategies:

- **Write Through:**
  - Writes to both cache and main memory. (go through cache)
  - Ensures data consistency but may lead to higher latency.
  - Advantages: Data consistency, lower risk of loss.
  - Disadvantages: Higher latency, performance impact.

- **Write Back:**
  - Writes to cache, updates main memory later.
  - Lowers write latency but risks data inconsistency.

  - Advantages: Lower latency, improved performance.
  - Disadvantages: Data inconsistency, complexity.

- **Write Around:**
  - Data is written directly to the main memory, bypassing the cache.
  - Useful for data that is not frequently accessed.
  - May result in longer read times for recently written data.
  - Effective in preventing unnecessary caching of less important or temporary data.

Difference between Cache and Databases:
Purpose: Caching for performance; databases for structured storage.
Data Durability: Caching sacrifices persistence; databases prioritize it.
Latency: Caching minimizes latency; databases might have higher latency.
Data Management: Caching uses lightweight structures; databases offer complex management.
Lifespan: Caching is short-term; databases offer long-term data storage.

Distributed Cache:
Cache system spanning multiple networked nodes for faster data access by distributing cached data and improving scalability, availability, and performance.

Redis:
- In-Memory Data Store: RAM-based, enabling rapid data access.
- Key-Value Store: Stores data as key-value pairs for quick retrieval.
- Versatile Data Structures: Supports strings, hashes, lists, etc.
- Persistence: Offers snapshotting, append-only files for durability.
- Caching: High-speed caching for frequently used data.
- Scalability: Supports clustering for distributed data.

Memcached:
- In-Memory Caching: Stores frequently used data in RAM.
- Key-Value Store: Optimized for quick key-value lookups.
- Distributed Architecture: Scalable with data spread across servers.
- No Data Persistence: Data not persisted, suited for non-critical info.
- Efficient Caching: Reduces database load, minimizing latency.

Content Delivery Network (CDN)

A Content Delivery Network (CDN) is a network of distributed servers strategically placed across various geographic locations to efficiently deliver digital content to users.

Why it's Needed:
- **Latency Reduction**: CDNs minimize the distance data travels, reducing loading times.
- **Global Reach**: Ensures consistent content access for users worldwide.

- **Scalability**: Handles traffic spikes and varying user demands effectively.
- **Load Balancing**: Distributes traffic to prevent server overloads.
- **Content Caching**: Stores copies of static content for faster retrieval.
- **Dynamic Content Acceleration**: Optimizes dynamic content delivery.
- **Security and Reliability**: Offers DDoS protection and security features.
- **Improved User Experience**: Faster content loading and smoother streaming.
- **Bandwidth Savings**: Offloads traffic from origin servers, reducing costs.

Examples of Commonly Used CDNs: Akamai, Cloudflare, Amazon CloudFront, Microsoft Azure CDN, KeyCDN, Cachefly