大题回答

Question 1

1. Purpose of Continuous Consistency

English Answer: The core purpose of continuous consistency is to allow controllable deviations (in numerical values, timeliness, or update order) between replicas. It reduces the high coordination costs of strong consistency while meeting the practical consistency requirements of applications.

中文答案：连续一致性（Continuous Consistency）的核心目的是允许副本之间在数值、时效性或更新顺序上存在可控偏差，从而降低强一致性带来的高昂协调成本，同时满足应用对一致性的实际需求。

2. Quantitative Observation Methods

English Answer: Three key quantitative dimensions are used: ① Numerical deviation (absolute/relative differences between replica values, e.g., stock price differences ≤ $0.02); ② Staleness deviation (how old replica data is, e.g., weather report validity ≤ 2 hours); ③ Order deviation (number of out-of-order updates between replicas).

中文答案：量化观测主要通过三个维度实现：① 数值偏差（副本间数据的绝对/相对差异，如股价差异≤0.02美元）；② 时效性偏差（副本数据的陈旧程度，如天气预报有效期≤2小时）；③ 顺序偏差（副本间更新操作的乱序数量）。

3. Implementation of Client-Centric Consistency Models

English Answer: Client-centric consistency is implemented by tracking two sets for each client: Read set (identifiers of relevant writes) and Write set (identifiers of the client's own writes). For specific models: ① Monotonic Reads: Ensure subsequent reads get the same or newer values by syncing the read set with the server; ② Monotonic Writes: Enforce write order by completing previous writes before new ones; ③ Read-Your-Writes: Ensure the server has executed all writes in the client's write set before reading; ④ Writes-Follow-Reads: Sync the read set to the server before new writes.

中文答案：用户中心一致性模型通过为每个客户端维护两组数据实现：读取集（Read set，相关写入操作的标识符）和写入集（Write set，客户端自身写入操作的标识符）。具体模型实现：① 单调读：读取前同步读取集至服务器，确保后续读取获取相同或更新的值；② 单调写：新写入前完成之前的写入操作，保证写入顺序；③ 读己写：读取前确保服务器已执行客户端写入集中的所有操作；④ 写跟随读：新写入前将读取集同步至服务器，再执行写入。

4. Replica Operation Ordering & Sequential Consistency Scenario

English Answer:

- **Configuration Scenario**: A distributed database with 3 replicas (R1, R2, R3) handling client writes (W1: x=1, W2: x=2) and reads. Assume operations execute as: R1 executes W1→W2, R2 executes W2→W1, R3 executes W1→W2.
- **Analysis**: Sequential consistency requires all processes to see operations in the same global order (matching each process's local order). Here, R2's operation order (W2→W1) conflicts with R1/R3's order. Thus, sequential consistency is NOT guaranteed. For guarantee, all replicas must execute operations in a globally consistent order (e.g., via primary-based protocols or total-order multicast).

中文答案:

- **配置场景**：分布式数据库包含3个副本（R1、R2、R3），处理客户端写入操作（W1：x=1，W2：x=2）和读取操作。假设副本执行步骤：R1执行W1→W2，R2执行W2→W1，R3执行W1→W2。
- **分析**：顺序一致性（Sequential Consistency）要求所有进程看到的操作遵循相同的全局顺序，且与每个进程的本地顺序一致。此处R2的操作顺序（W2→W1）与R1、R3的顺序冲突，因此**无法保证顺序一致性**。若要保证，需通过主副本协议或全序多播等机制，使所有副本按全局一致的顺序执行操作。

5. Implementation of Read-Your-Writes Consistency in Browser Caching

English Answer:

Read-Your-Writes Consistency in browser caching ensures clients always read their own latest writes, and it's implemented through the following coordinated mechanisms (based on the content):

1. **Client-side write set management**: The browser maintains a "write set" (record of the client's own write operations). When initiating a read request, the browser sends this write set to the target server to inform it of all writes the client has performed.
2. **Server-side data synchronization**: Before responding to the read request, the server checks if it has executed all operations in the client's write set. If any writes are missing, the server fetches the latest data corresponding to these writes from other servers (to ensure it holds the up-to-date content).
3. **Target server selection optimization**: Instead of synchronizing data, the client can directly select a server that has already completed all writes in its write set (avoiding extra synchronization delays). After the read is executed, the server updates the client's "read set" (record of reads associated with writes) to maintain consistency for subsequent operations.
4. **Web cache freshness control**: To prevent stale cached data from breaking consistency, two strategies are used:

- **Active validity verification (Option 1)**: The cache contacts the origin server to confirm if the cached content is still up-to-date before returning it to the client.
- **Expiration time-based validity (Option 2)**: The cache assigns an expiration time (T_{expire}) to cached content (calculated based on the last modification time of the document and the caching time). Content is considered valid only before T_{expire}, forcing a fresh request afterward.

中文答案：

浏览器缓存中的读己写一致性，核心是确保客户端能读取到自己写入的最新数据，其实现基于以下协同机制（结合内容）：

1. **客户端追踪写入集**：浏览器维护“写入集”（记录客户端自身的写入操作），发起读取请求时，将该写入集发送给目标服务器，告知其客户端已执行的所有写入操作。
2. **服务器同步写入操作**：服务器在响应读取请求前，会校验是否已执行客户端写入集中的所有操作；若存在缺失的写入，服务器会从其他服务器拉取这些写入对应的最新数据，确保自身持有最新内容。
3. **优化目标服务器选择**：客户端可直接选择已完成其写入集中所有操作的服务器进行读取（避免额外的同步延迟）；读取执行后，服务器会更新客户端的“读取集”（关联写入的读取记录），为后续操作维持一致性。
4. **保障缓存新鲜度**：为避免陈旧缓存数据破坏一致性，采用两种策略：
   - **主动验证有效性（方案1）**：缓存向源服务器确认缓存内容是否仍为最新版本，再返回给客户端。
   - **基于过期时间的有效性（方案2）**：为缓存内容分配过期时间（T_{expire}），该时间由文档最后修改时间与缓存时间计算得出；内容仅在T_{expire}前有效，超时后会触发新的请求以获取最新数据。

Question 2

1. Replica Quantity for Fault Tolerance in Stateless Applications

English Answer: For stateless applications relying on replication, the required replica quantity depends on fault types: ① For halting failures (crash, latency, failure to send responses), at least k+1 replicas are needed (non-faulty replicas provide correct services directly); ② For arbitrary failures (e.g., incorrect responses), at least 2k+1 replicas are needed (correct results are obtained via majority voting).

中文答案：无状态应用依赖复制容错时，副本数量取决于故障类型：① 针对停止类故障（崩溃、延迟、无法发送响应），至少需要 k+1 个副本（非故障副本直接提供正确服务）；② 针对任意故障（如返回错误响应），至少需要 2k+1 个副本（通过多数投票获取正确结果）。

2. Replica Quantity for Byzantine Fault Tolerance (Compromised Replicas)

English Answer:
To tolerate k compromised (Byzantine) replicas, **at least 3k+1 replicas are required** (instead of 2k+1 for non-Byzantine faults).
The key reason is that Byzantine replicas can exhibit "dishonest" behavior: they may send conflicting messages to different honest replicas (e.g., a malicious replica sends value X to one honest node and Y to another). This leads to honest nodes perceiving different "majority" results, causing system divergence.

When the total replica count is 3k+1:

- The number of honest replicas is 3k+1 - k = 2k+1.
- To reach consensus, a majority (≥ 2k+1 replicas) is required (since 2k+1 > (3k+1)/2, satisfying the majority condition).
- Any two such majorities will intersect by at least (2k+1) + (2k+1) - (3k+1) = k+1 nodes. Since there are only k malicious replicas, this intersection must include at least 1 honest node—ensuring honest replicas can align on a consistent result and avoid divergence.

中文答案：

要容忍 k 个被攻破的（拜占庭）副本，**至少需要 3k+1 个副本**（区别于非拜占庭故障的 2k+1 个副本）。
核心原因是拜占庭副本会表现出“不诚实”行为：它们可能向不同诚实副本发送矛盾消息（例如，恶意副本向一个诚实节点发值X，向另一个发值Y），导致诚实节点感知到不同的“多数派”结果，进而引发系统分叉。

当总副本数为 3k+1 时：

- 诚实副本数为 3k+1 - k = 2k+1；
- 共识需满足多数派条件（≥ 2k+1个副本），而 2k+1 > (3k+1)/2，符合多数派的数学要求；
- 任意两个这样的多数派交集至少包含 (2k+1)+(2k+1)-(3k+1) = k+1 个节点。由于恶意副本仅 (k) 个，该交集必然包含至少1个诚实节点——确保诚实副本能对齐一致结果，避免系统分叉。

3. CAP Theory Application: Prioritizing Availability During Partition

English Answer: According to the CAP theory, when network partition (P) occurs and availability (A) is prioritized over consistency (C), the system should: ① Allow each partition to continue independent operations (e.g., processing local reads/writes) instead of blocking; ② Use eventual consistency protocols (e.g., Paxos with leader election in each partition) to sync data after the partition heals; ③ Avoid global consistency coordination during partition to ensure immediate responses to client requests.

中文答案：根据CAP理论，当出现网络分区（P）且可用性（A）优先于一致性（C）时，系统应：① 允许每个分区独立运行（如处理本地读写操作），而非阻塞；② 采用最终一致性协议（如分区内选举leader的Paxos协议），在分区恢复后同步数据；③ 分区期间避免全局一致性协调，确保客户端请求获得即时响应。