NoSQL ve CAP Teoremi

CAP Theorem

- Dağıtık sistemlerin consistency, availability ve partition toleration kabiliyetlerini belirler
- CAP teoremine göre pratikte dağıtık sistemlerdeki olası partition problemleri durumunda consistency ve availability arasında bir tercihte bulunmak gerekecektir
- Bu tercih ya hep ya hiç olmak zorunda değildir

CAP Theorem

- Consistency'den biraz feragat ederek, availability'den biraz daha kazanmak söz konusu olabilir
- Sonuç olarak tam manası ile ne tam consistent ne de tam available bir sistem ortaya çıkar
- Ama mevcut durum kullanıcılar için kabul edilebilir bir seviyede olabilir

Consistency

- Okunan veya yazılan verinin tutarlılığının sağlanmasıdır
- ACID prensiplerinden birisidir
- İlişkisel veritabanları muhtemel inconsistent durumlardan sakınarak “strong consistency” sağlamaya çalışırlar
- NoSQL veritabanlarında ise “eventual consistency” sağlamaya yönelik işlem yapılır

Update Consistency

- Verinin tutarlı biçimde güncellenmesine odaklanır
- Aynı veri üzerinde eş zamanlı birden fazla yazma (write) işlemi gerçekleşirse “write-write conflict” ortaya çıkar
- Eşzamanlı erişim denetimi yoksa “lost update” problemi ortaya çıkar
- Çözümü için pessimistic veya optimistic yöntemlerden birisi kullanılabilir

Update Consistency

- Pessimistic yöntem conflict'in önüne geçmeye çalışır
  - Kayıt veya tablo düzeyinde kilitler
- Optimistic yöntem ise conflict'i bir an evvel tespit etmeye odaklanır
  - Versiyon kontrolü
  - Şartlı güncelleme (conditional update)
  - Merge

Read Consistency

- Yazılan verinin tutarlı biçimde okunmasına (read) odaklanır
- Bir grup veri üzerinde eş zamanlı işlem yapan birden fazla transaction varsa “read-write conflict” ortaya çıkabilir
- Birden fazla tabloya yazma işi yapan bir transaction’ın ortasında diğer bir transaction bu tablolardan okuma yapabilir
- Buna “logical inconsistency” adı verilir

Read Consistency

- Logical inconsistency belirli bir süre devam edebilir
- Buna “inconsistency window” adı verilir
- Belirli bir süre sonra bu inconsistency sona erecektir
- Bu duruma “eventually consistent” adı verilir

Read Consistency

- Cluster ortamlarda “replication consistency” de önem arz eder
- Inconsistency window, iki node’a yazma işlemi sırasında da söz konusudur
- Bu durumda en azından transaction’ın kendi yazdığını okuyabilmesi (read your writes consistency) önemlidir
- Buna “session consistency” adı verilir

Read Consistency

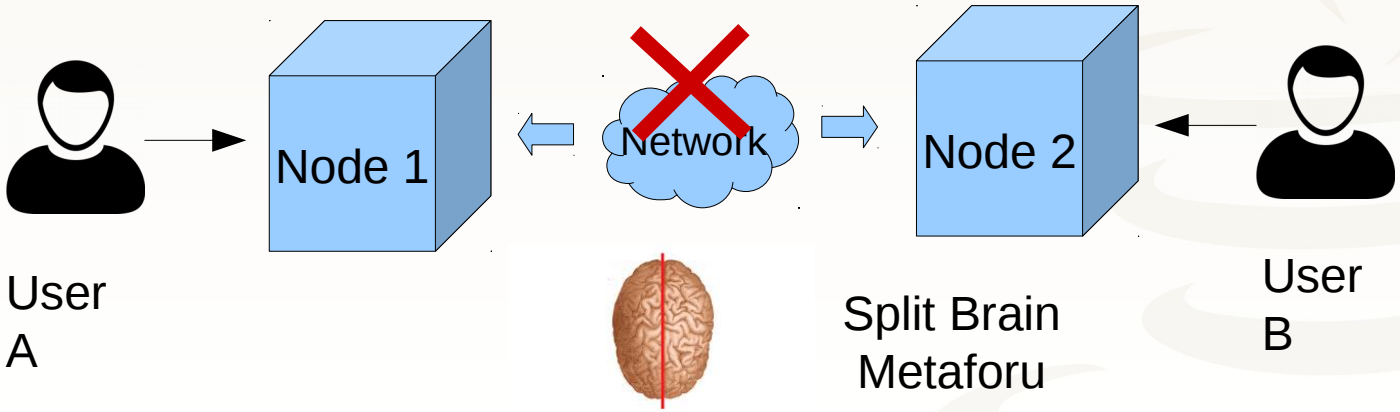
- Session consistency'i sağlamak için şu yöntemler kullanılabilir
  - Sticky session (Session affinity)
  - Version timestamp

Availability

- CAP teoremi kapsamında ayrı bir anlama sahiptir
- Eğer cluster ortamdaki bir node konuşulabilir durumda ise bu node üzerinden okuma ve yazma (read & write) yapılabilir

Partition Tolerance

- Cluster'ın birbirinden bağımsız kümeleri arasında meydana gelen iletişim problemlerinden cluster'ın etkilenmemesi demektir
- Tek sunucu durumu -> C A ~~P~~



Durability

- Verinin kalıcı biçimde saklanmasıdır
- ACID prensiplerinin temellerinden biridir
- Bazı durumlarda yüksek performans kazancı karşılığında durability'den bir miktar feragat edilebilir
- Bunun için veri önce hafızada saklanır ve hafızadan erişilir
- Bu durumda veri zaman zaman diske flush edilerek durability sağlanır

Durability

- Verinin diske yazılmadığı sırada meydana gelebilecek bir problemde bir miktar veri kaybı söz konusu olur
- Örneğin, kullanıcı session verisi'nin hafızada saklanması ve hafızadan erişilmesi, zaman zaman diske yazılması

Quorum

- Consistency ve durability'den belirli oranlarda feragat edip toplamda kabul edilebilir bir sistem davranışı elde edilebilir
- Veri işlemine daha fazla node dahil ederek inconsistency'den biraz daha kaçınmak mümkündür
- Böylece strong consistency durumuna biraz daha yaklaşılır
- Quorum, strong consistency elde etmek için kaç node dahil edilmesi gerektiğinin hesabıdır

Quorum ve Distribution Model

- Read ve Write şeklinde iki tür Quorum hesabı söz konusudur
- Read ve Write Quorum formülleri peer-to-peer model için geçerlidir
- Master-slave model'inde write consistency için sadece master'a yazmak yeterlidir
- Read-write conflict olmaması için de sadece master'dan okumak gerekir

Write Quorum

- Yazma işleminde strong consistency için bütün node'lardan ACK almak gerekli değildir
- Write node'larının sayısının, replikasyona katılan node'ların yarısından fazla olması yeterlidir
- Write Quorum = W > N / 2
  - W : yazmaya katılan node sayısı
  - N : replikasyona katılan node sayısı (replication factor)

Read Quorum

- Okuma işleminde de strong consistency elde etmek için her zaman bütün node'lardan ACK şart olmayabilir
- Read quorum'un değeri write quorum'a bağlıdır
- Read Quorum = R + W > N
  - R : okumaya katılan node sayısı
  - W : yazmaya katılan node sayısı
  - N : replikasyona katılan node sayısı

Quorum ve Replication Factor

- Çoğu otoriteye göre N = 3 şeklinde bir replication factor değeri resillience için gayet yeterlidir
- Böyle bir durumda bir node başarısız olsa bile write ve read quora'lar sağlanmaya devam edecektir
- Eğer hızlı ve strongly consistent okuma isteniyorsa N = 3, W = 3, R =1 olmalıdır
- Böyle bir durumda yazma işlemleri doğal olarak yavaş olacaktır

Business/System Transactions ve Consistency

- Kullanıcı açısından anlam ifade eden ve bir bütün olarak ele alınan işlemlere business transaction adı verilir
- Örneğin, ürün seçme, ödeme ve teslimat bilgilerini girme adımları bir bütün olarak ürün siparişini oluşturur
- Ürün siparişi business transaction'dır
- Bir business transaction sırasında birden fazla tablo veya kayıt üzerinde işlem yapılır

Business/System Transactions ve Consistency

- System transaction'lar ise DB düzeyinde tablo veya kayıtlara atılan kilitleri ifade eder
- Bu kilitlerin business transaction'lar boyunca açık kalması performans açısından uygun değildir
- Dolayısı ile system transaction'lar çoğunlukla business transaction'ların son aşamasında oluşturularak işlemler gerçekleştirilir

Business/System Transactions ve Consistency

- Bu durumda da business transaction içerisinde güncel olmayan veriye dayanarak işlem yapılması söz konusu olabilir
- Bu tür güncel olmayan veri üzerinde işlem yapma durumlarını ele almak için genellikle offline optimistic concurrency tekniğinden yararlanılır
- Version stamp kullanarak offline optimistic concurrency kabiliyeti sağlanabilir

Version Stamps

- Update ve session consistency sağlamak için kullanılan bir yöntemdir
- Veri ile tutulan bir değerdir ve veri her değiştiğinde version stamp değeri de güncellenir
- Version stamp türleri
  - Counter
  - GUID
  - Content hash
  - Timestamp

Version Stamps

- Counter, birer birer artırılan bir değerdir, hangi verinin daha güncel olduğunu tespit etmeye yarar, ancak culuster genelinde counter'ı yöneten bir master'a ihtiyaç duyulur
- GUID, rastgele üretilen büyük cluster genelinde benzersiz bir değerdir, ancak hangi değerin daha güncel olduğunu tespit etmeye yardımcı olmaz

Version Stamps

- Content hash, verinin içeriğinden üretilen bir hash değeridir, GUID gibi iki işlemden hangisinin daha güncel olduğunun tespitine yardımcı olmaz
- Timestamp, güncelleme zaman bilgisidir ve hangi verinin daha güncel olduğunu tespit etmeye yarar, ancak cluster genelinde clock senkronizasyonu ve timestamp değerinin hassasiyeti (millisec/nanosec) de önemlidir

Version Stamps

- Composite stamp ile, birden fazla yöntem birleştirilir
- Örneğin CouchDB, counter + content hash birlikte kullanır
- Böylece version stamp değeri hangi verinin daha güncel olduğunu tespit etmeyi sağlar
- İki peer node aynı anda fakat farklı content hash değeri ile yazma yapmaya çalışırsa write conflict olduğu tespit edilebilir

Version Stamps

- Peer-to-peer NoSQL sistemlerinde yaygın biçimde kullanılan çözüm vector stamp (vector clock/version vector) yöntemidir
- Her node için ayrı bir counter değeri vardır
- Peer node üzerinde bir update gerçekleşirse vector içindeki kendi counter değeri de artırılır
- İki peer node birbirleri ile senkronize olurken vector stamp'lerini de senkronize ederler

Version Stamps

- Vector stamp kullanımı inconsistent durumları tespit etmek için yararlıdır
- Ancak tespit edilen inconsistent durumu çözüme kavuşturmak uygulama domain'ine göre değişiklik gösterecektir
- Sonuç olarak ya network partition'ları sisteminizi unavailable yapacaktır, ya da inconsistent durumları tespit edip, bunlarla uğraşılacaktır

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com