Discussion of Projects

• These slides suffer from my having spent the past couple of days preparing my presentation for the NSF site visit that is going on right now :-(

(New) Project Requirements

- 3-Tier
- Message-based, J2EE (JMS)
- Scalability/failover in middle tier
- Scalable replication/hot standby at bottom tier (MySQL)
(New) Project Requirements

- Transaction Mix:
  - Frequent small txns
  - Infrequent very large txns
- Consistency, isolation models, performance tradeoffs in each class of txn
(New) Project Requirements

- User identity, authentication
  - Stored user preferences
- User query capability
  - DB queries or search engine
- Some form of subscription / notification
  - Performance?
  - Scalability?
Replication / Hot Standby

- Replication:
  - Multiple instances visible to mid tier
  - Replicated updates e.g. triggers + distributed SQL
  - Mid tier handles failover
Replication / Hot Standby

- Hot Standby
  - One logical DB instance
  - Primary + “hot” standby
  - Propagate updates (via DB log) to standby
  - Failure: fail over and recover standby
Small vs Large Transactions

• How much data is locked => how much concurrency is available

• Examples:
  • Small: update my account at BoA
  • Large: how much $ on deposit at BoA

• Tweaks to get performance:
  • Consistency model
  • Serialization order
Example 1

- Music/content auction site
- Like eBay, but different auction model:
  - Artist proposes to release work
  - Potential buyer bids “I will buy for at most $7.50”, puts bid in escrow
  - Buyer can change / revoke bid at any time
  - Seller can choose price at any time to maximize revenue
- Sell once!
Example 1 (continued)

- Capability for users to search offerings
  - e.g. by artist, genre, title
- Query capability for sellers
  - revenue as function of price
  - price for max revenue
Example 1 (continued)

- Search need not be atomic
- Deciding to sell is a big (atomic) query
- Sale is a big transaction