Key-Value SQL DBMSs: Rise of the Hybrids

Shahram Ghandeharizadeh
Computer Science Department
University of Southern California

Outline

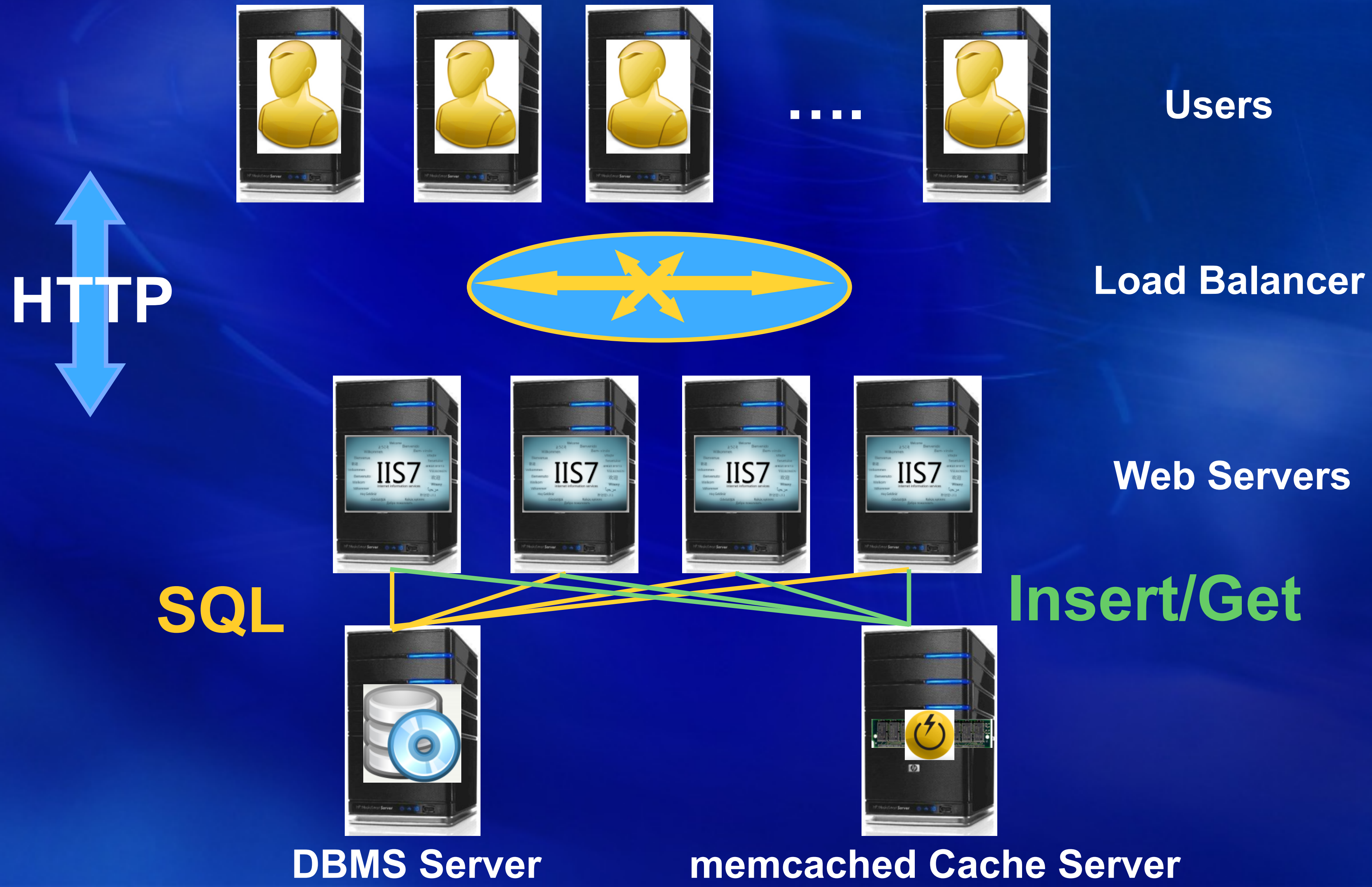
- Motivation
- Architecture
  - Classic
  - KV-SQL
- Research Challenges

Social Networking

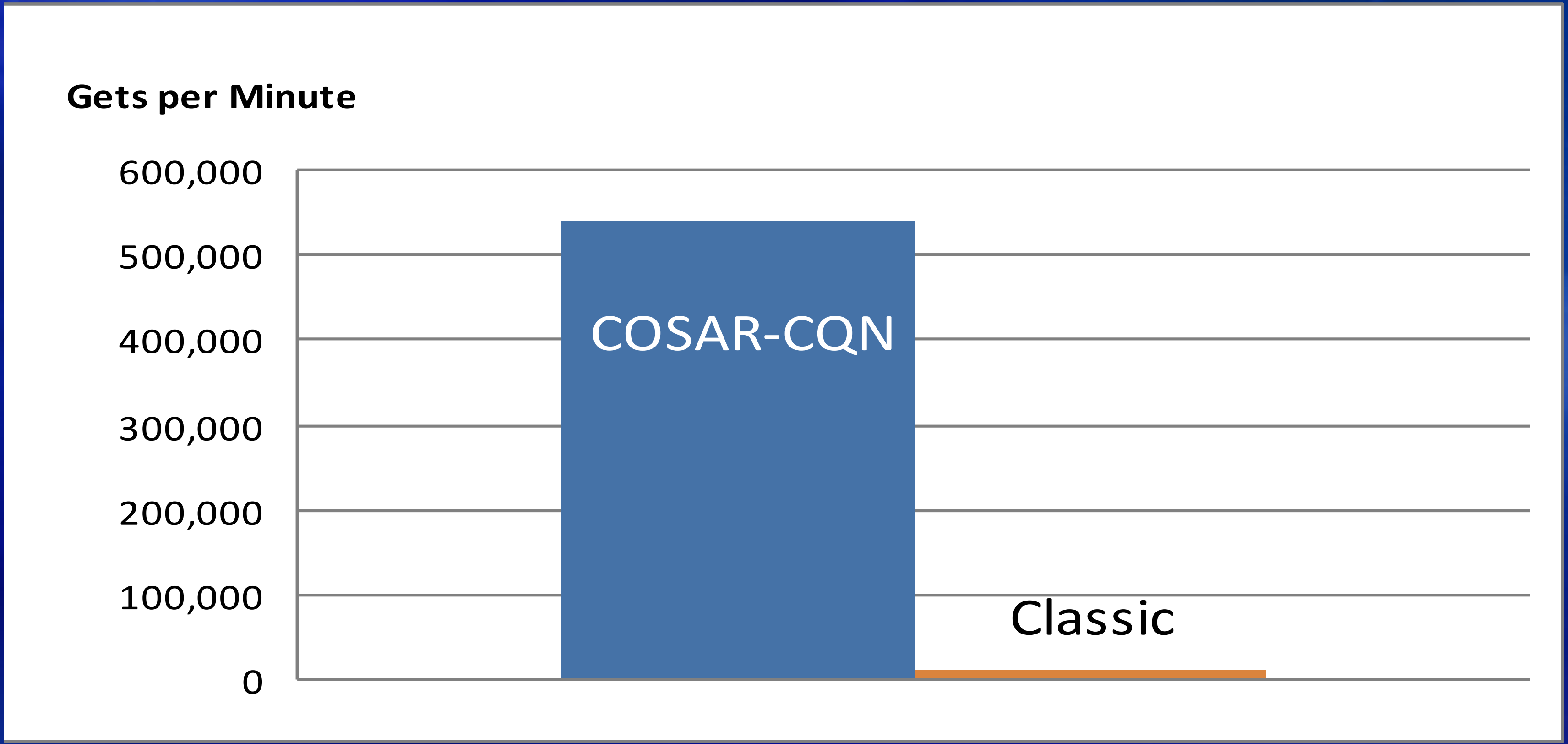
- Read intensive workloads. Queries that compute the same result repeatedly.
  - Same result because the database changes infrequently, e.g., your Facebook account:
    - How often do you visit your Facebook profile page?
      - Once a second, every minute, every hour, once a week?
    - How often do you add and drop friends/ change your profile?

Classical Architecture

KV-SQL Architecture



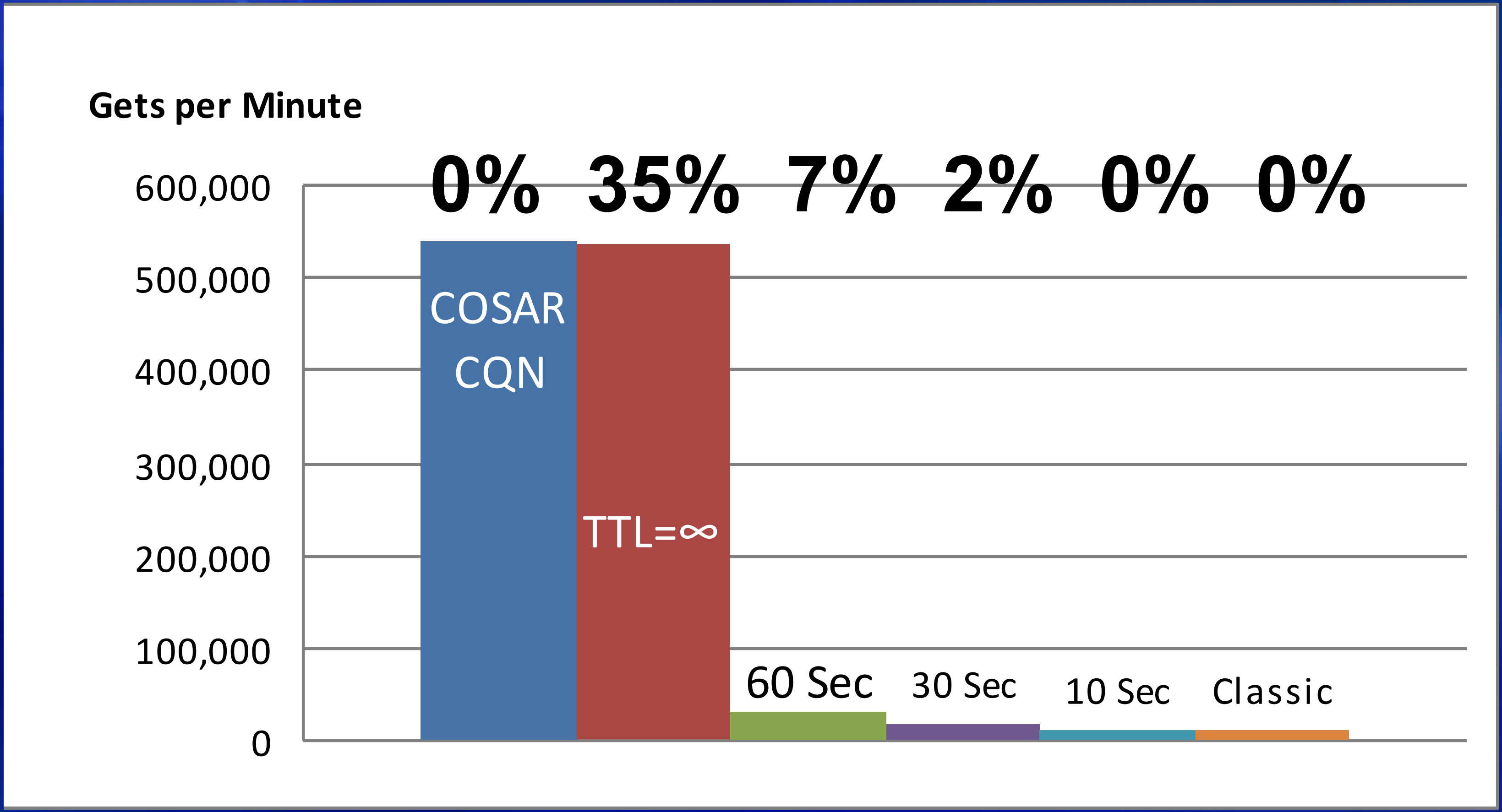
Enhances Performance



Cache Consistency

- When database changes, the application must maintain the cache consistent with the DBMS. How?
  - Application driven
  - DBMS driven
  - Synthetic, TTL

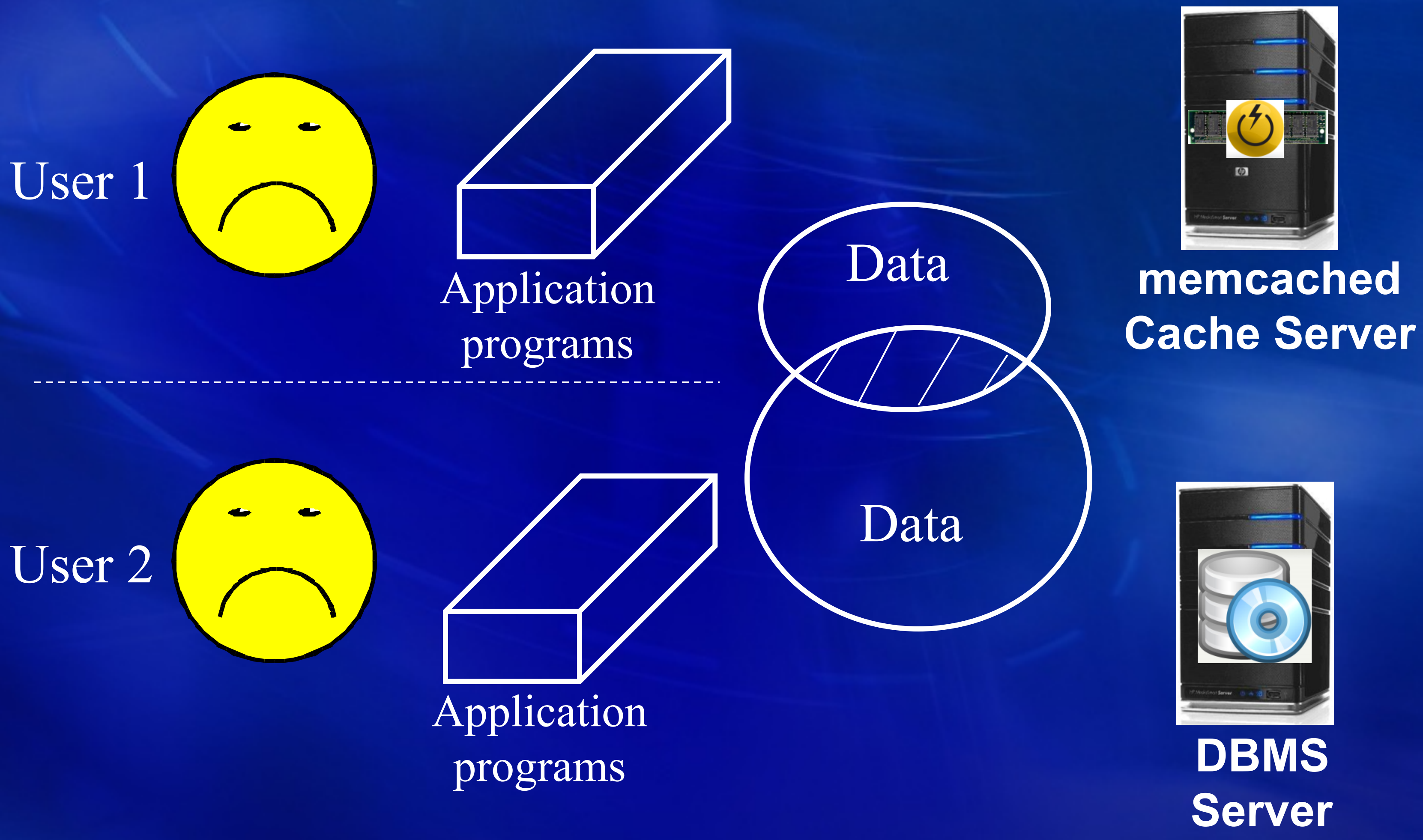
Synthetic: TTL Value?



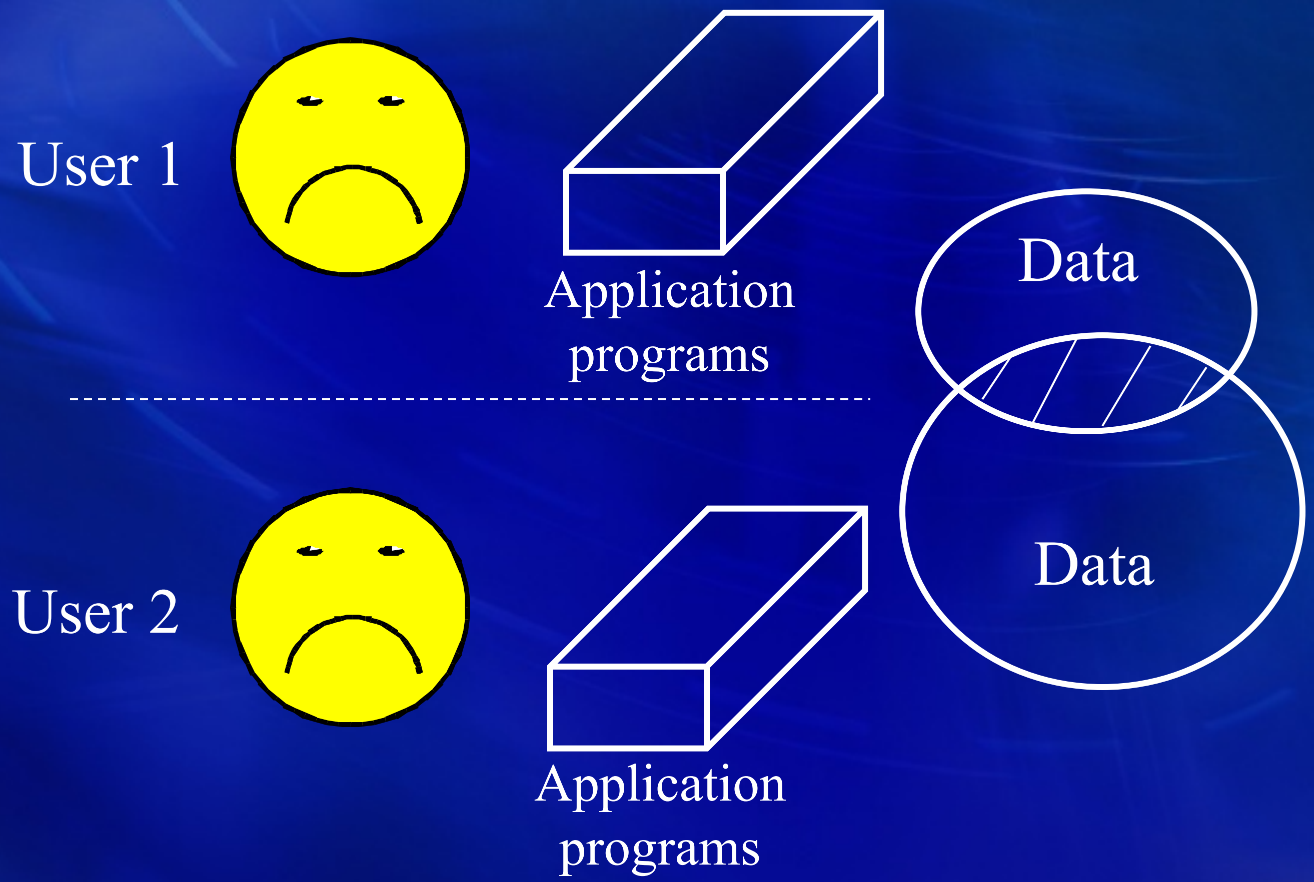
ACID Properties?

- No. To illustrate, consider the following two transactions:
  - T1 retrieves Key 1, K1
  - T2 deletes Key 1, K1
- Possible Serial Schedules: (T1, T2), and (T2, T1). With both schedules, a subsequent transaction T3 that references K1 finds no results.
- Consider the following schedule:
  - T1 observes a cache miss for K1
  - T1 issues queries to DBMS to construct K1-V1
  - T2 deletes K1 from the cache and DBMS
  - T1 inserts K1-V1 into the cache
- Final system state: K1-V1 in the cache with no copy in the DBMS. Subsequent transactions referencing K1 will find it in the cache.
- The application must implement the concept of transactions.

Oct 2011


User 1
User 2
Application programs
Application programs
Data
Data
memcached Cache Server
DBMS Server

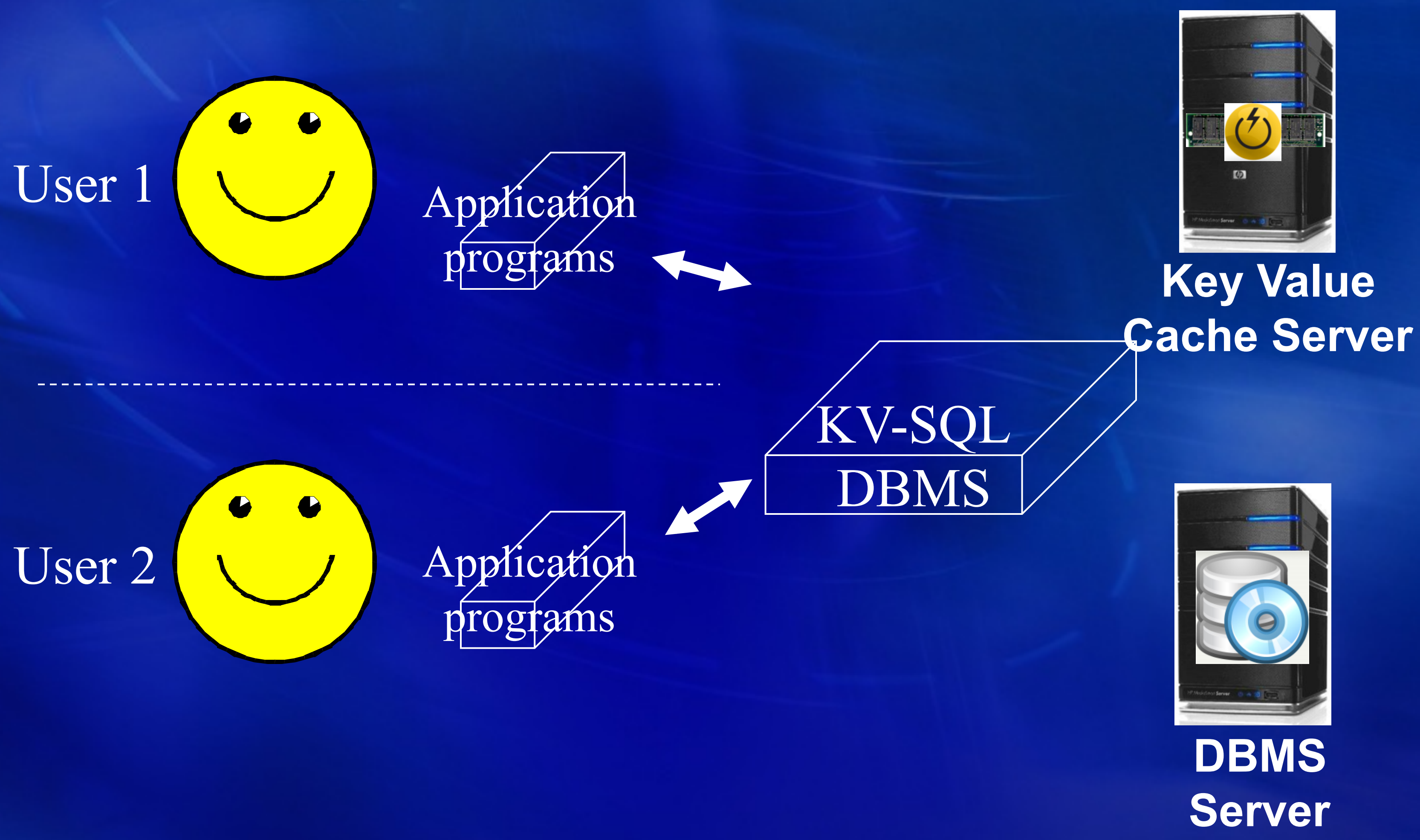
BEFORE DBMS: 1960/70s



Oct 2011

- To put people back to work, the government stops using electronic DBMSs.

KV-SQL DBMS


User 1
Application programs
User 2
Application programs
KV-SQL DBMS
Key Value Cache Server
DBMS Server

Stonebraker/Cattel’s 10 Rules

- Approximates each rule.

Research Challenges

- High level query languages:
  - SQL extended with KV
  - Object relational mapping, e.g., Djanga, Rails.
- Consistency and its definition.
- Physical data independence:
  - Application transparent approach to cache consistency.
- Granularity of cached data: structured versus semi-structured.
- Architecture and KV cache:
  - Server/infrastructure side:
    - Inside the DBMS,
    - middleware:
      - in a data center,
      - across a wide geographical area.
  - Client side.