Big Data Systems

Djellel Difallah

Spring 2023

Lecture 5 (cont.) –

Caching + Memcached + LRU Cache

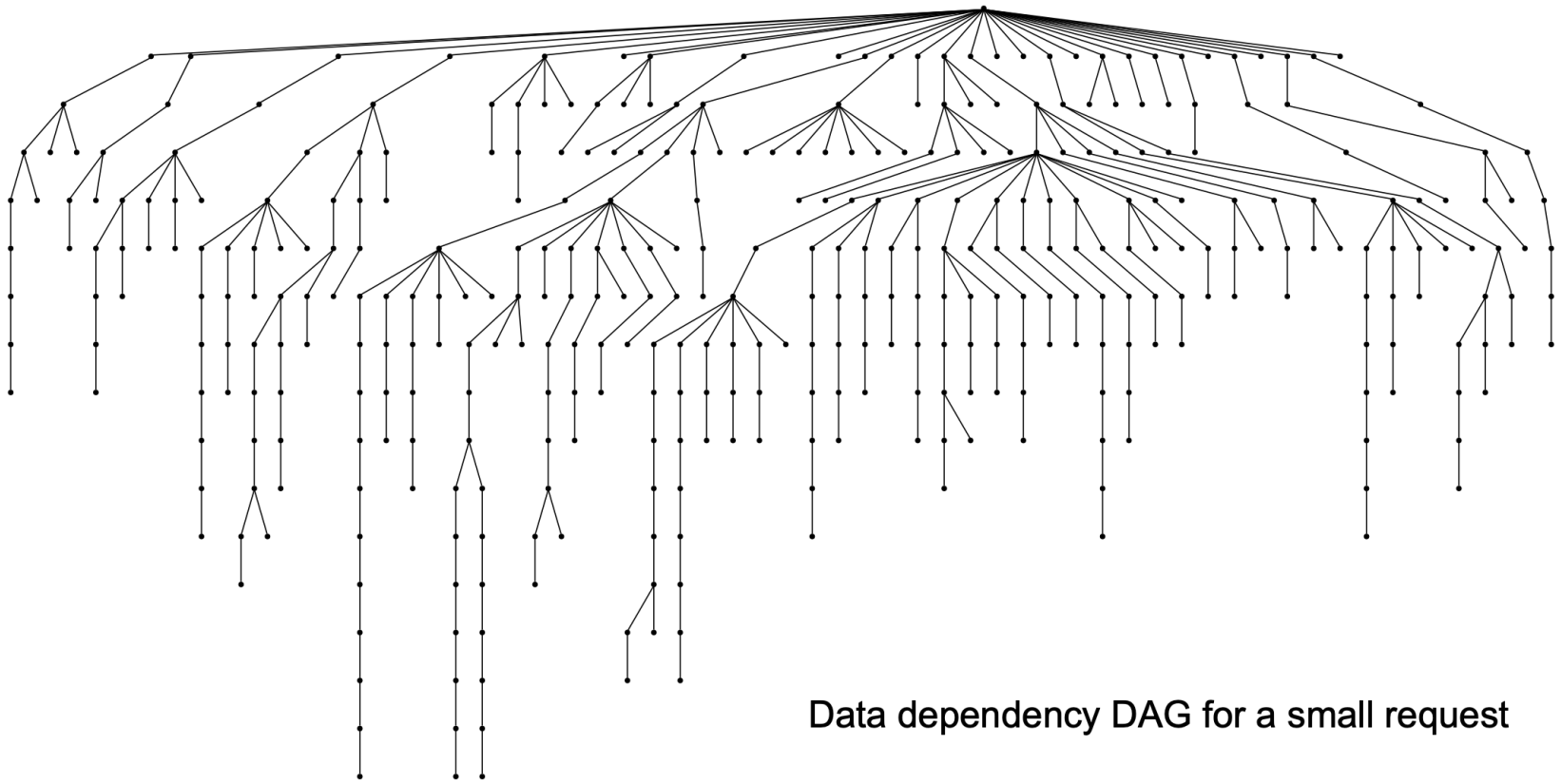
System Design Requirements (from Facebook)

- Support a very heavy read load
  - Over 1 billion reads / second
  - Insulate backend services (DB) from high read rates
- Geographically Distributed
- System must be flexible enough to support a variety of use cases
  - Support rapid deployment of new features
- **Persistence handled separately**
  - Support mechanisms to refresh content after updates

Caching

- Memory caching is essential to Web-scale Application Performance.
- Improves:
  - Read latency
  - Availability (When distributed)
  - Reduce the strain on the overall distribute system architecture
- In-memory Key Value Stores are used by virtually every large web platform: Facebook, Netflix, Pinterest, Wikipedia etc

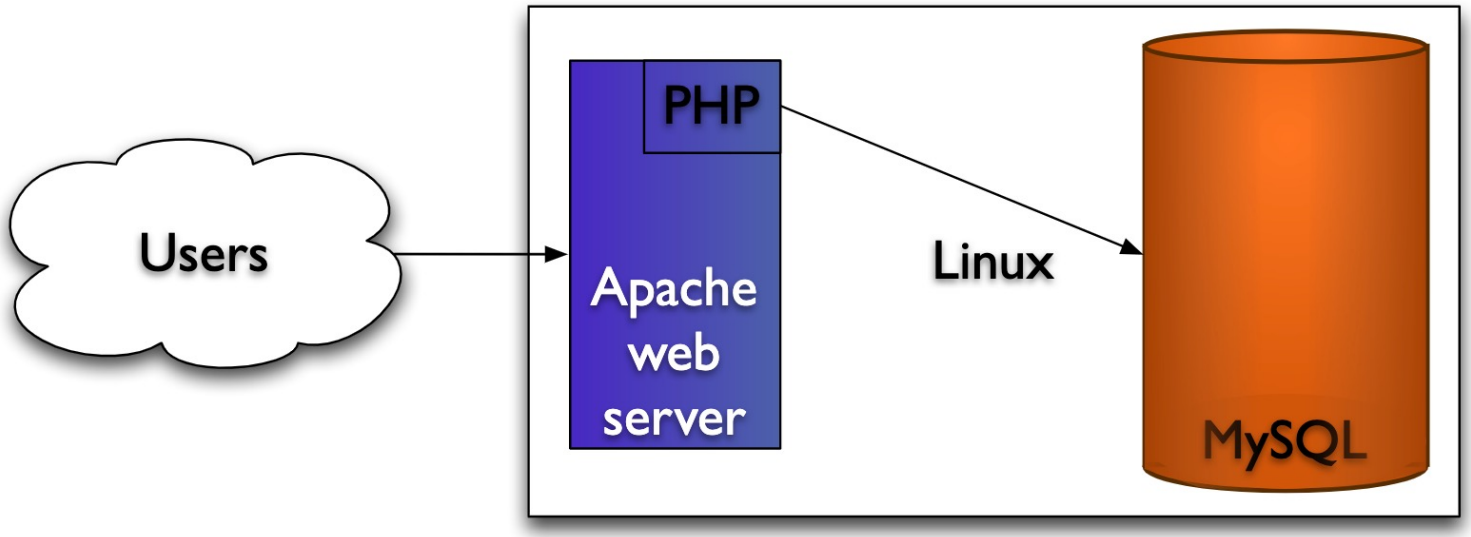
Why Caching Improves Reads?



Data dependency DAG for a small request

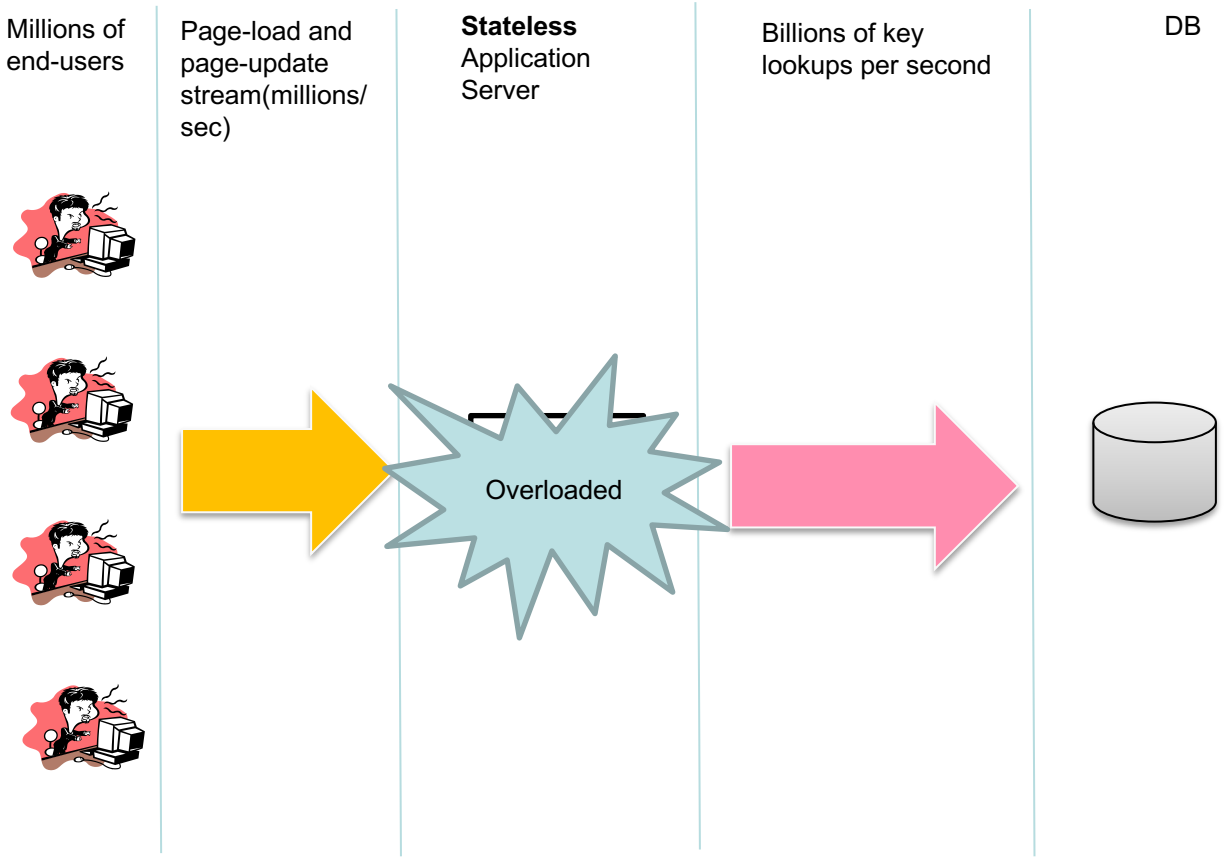
From "Scaling Memcache at Facebook" NSDI'13

Web Stack Architecture (LAMP)

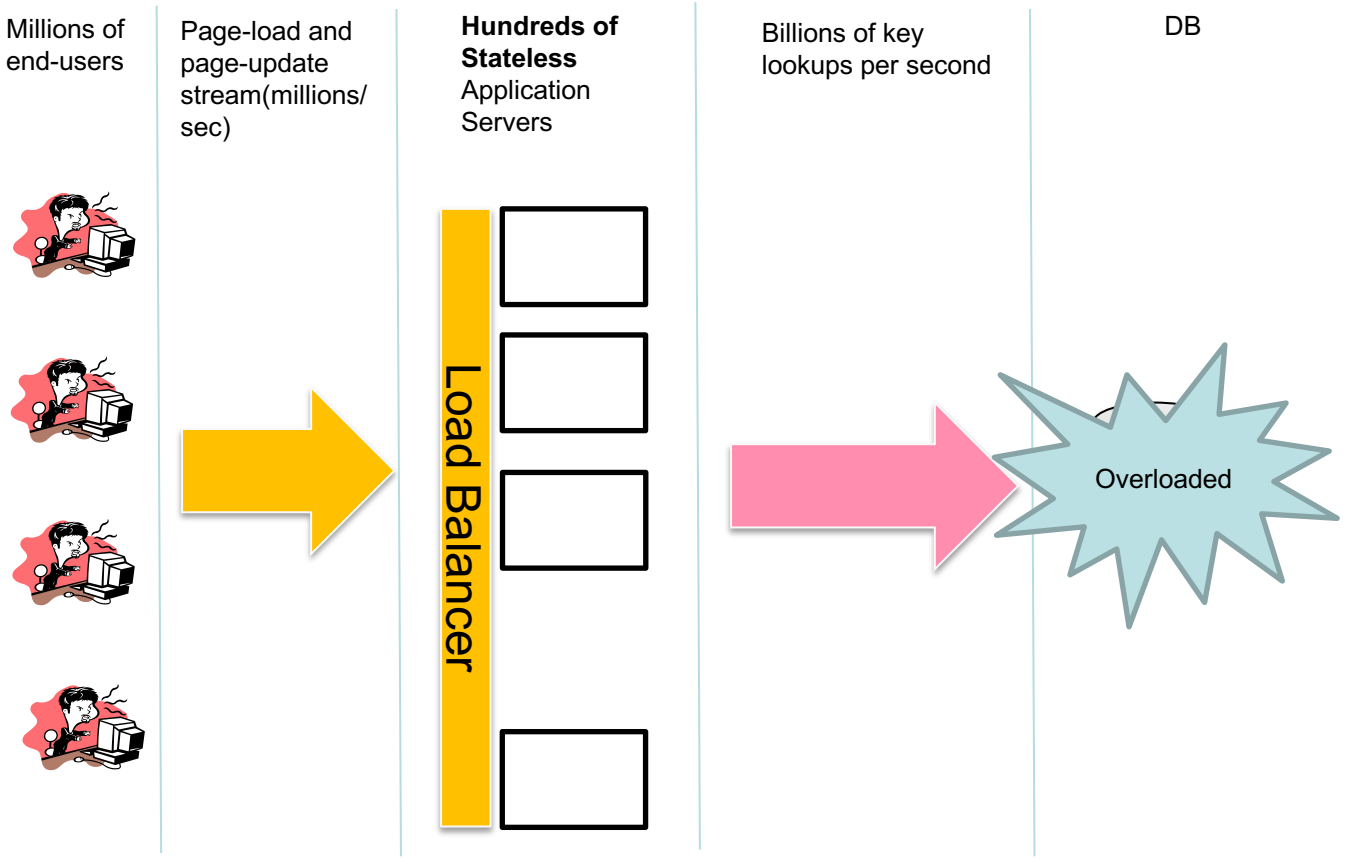


Wikimedia architecture by Mark Bergsma

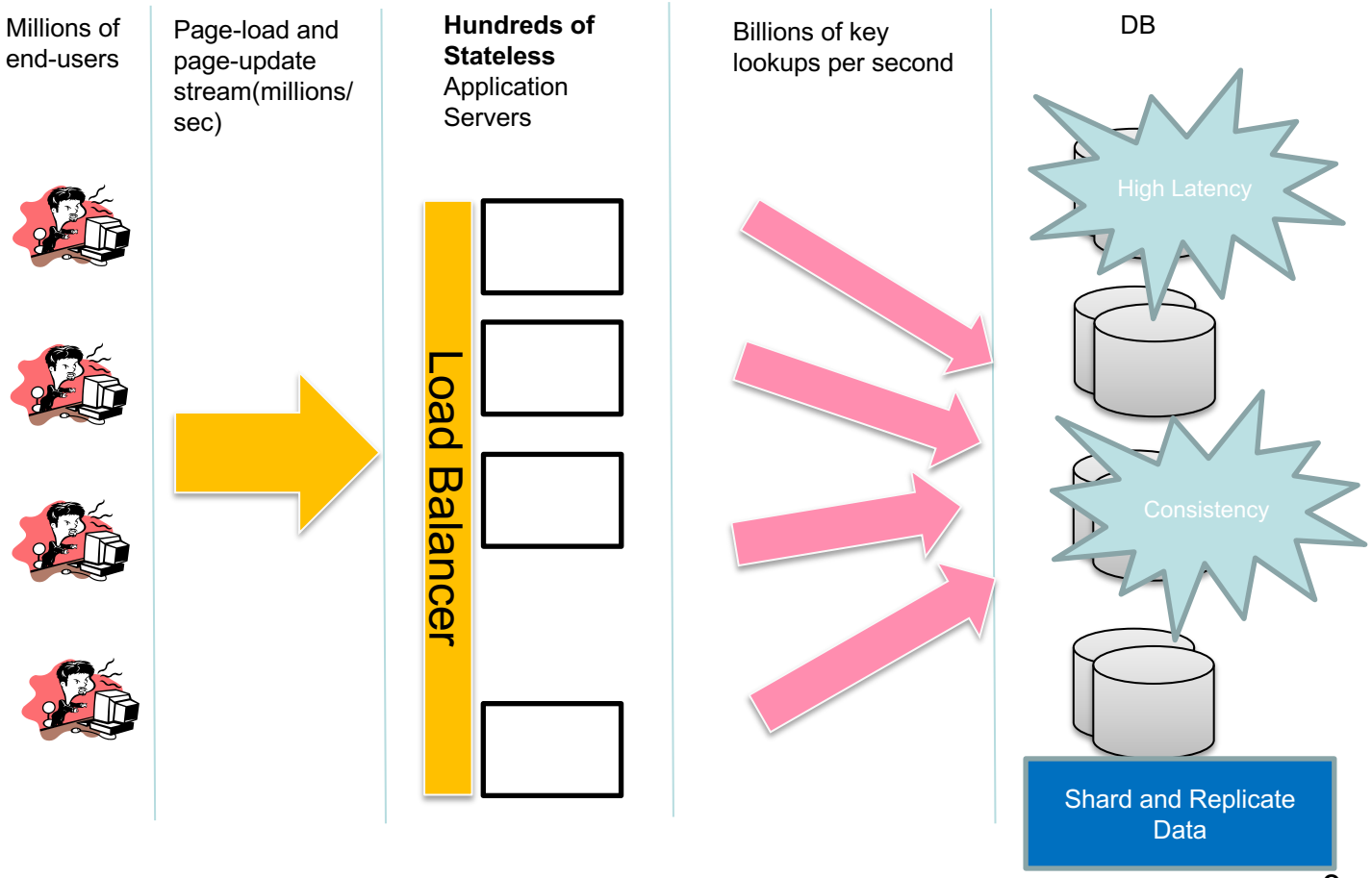
Why Separate Caching?



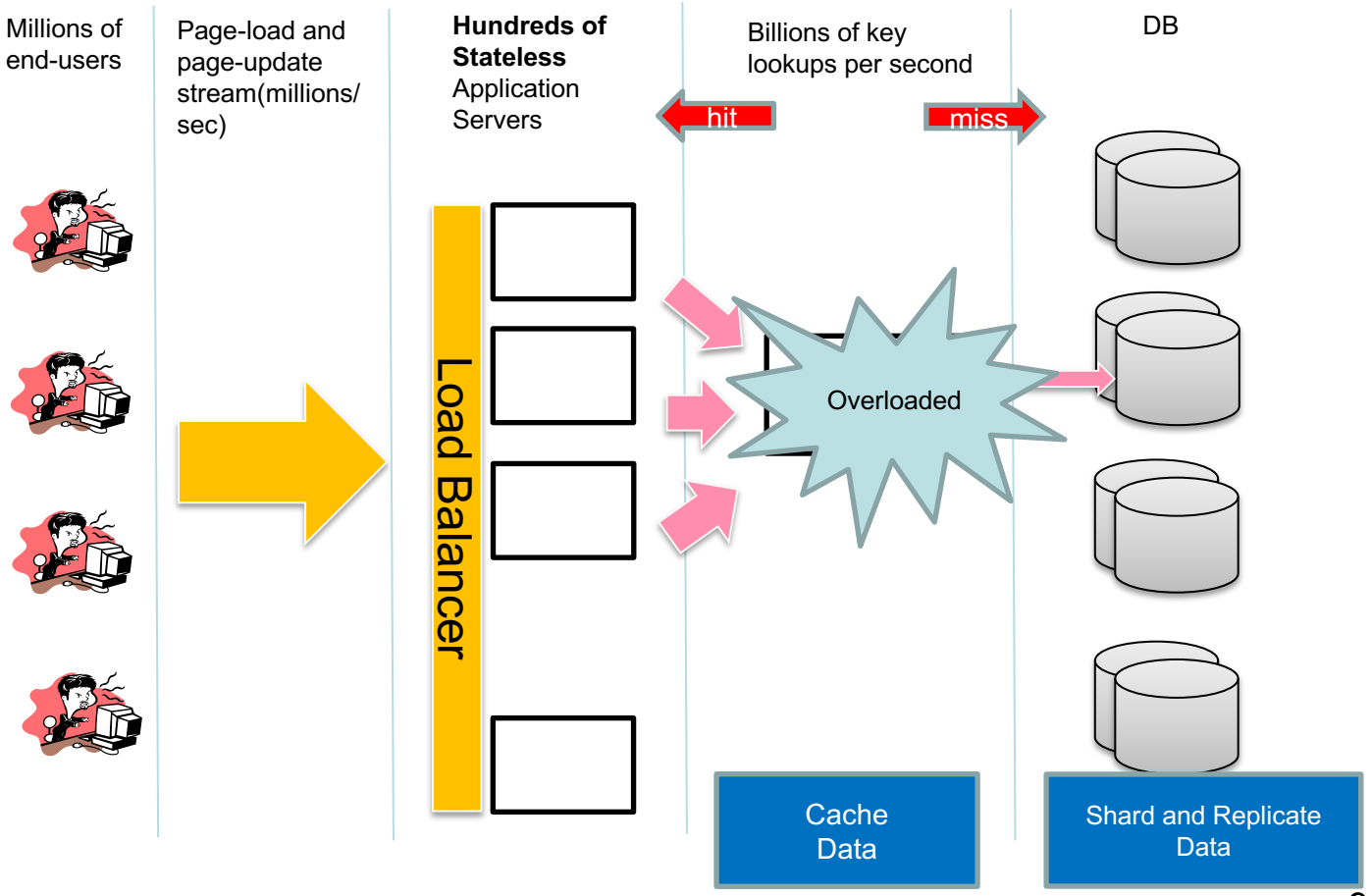
Why Separate Caching?



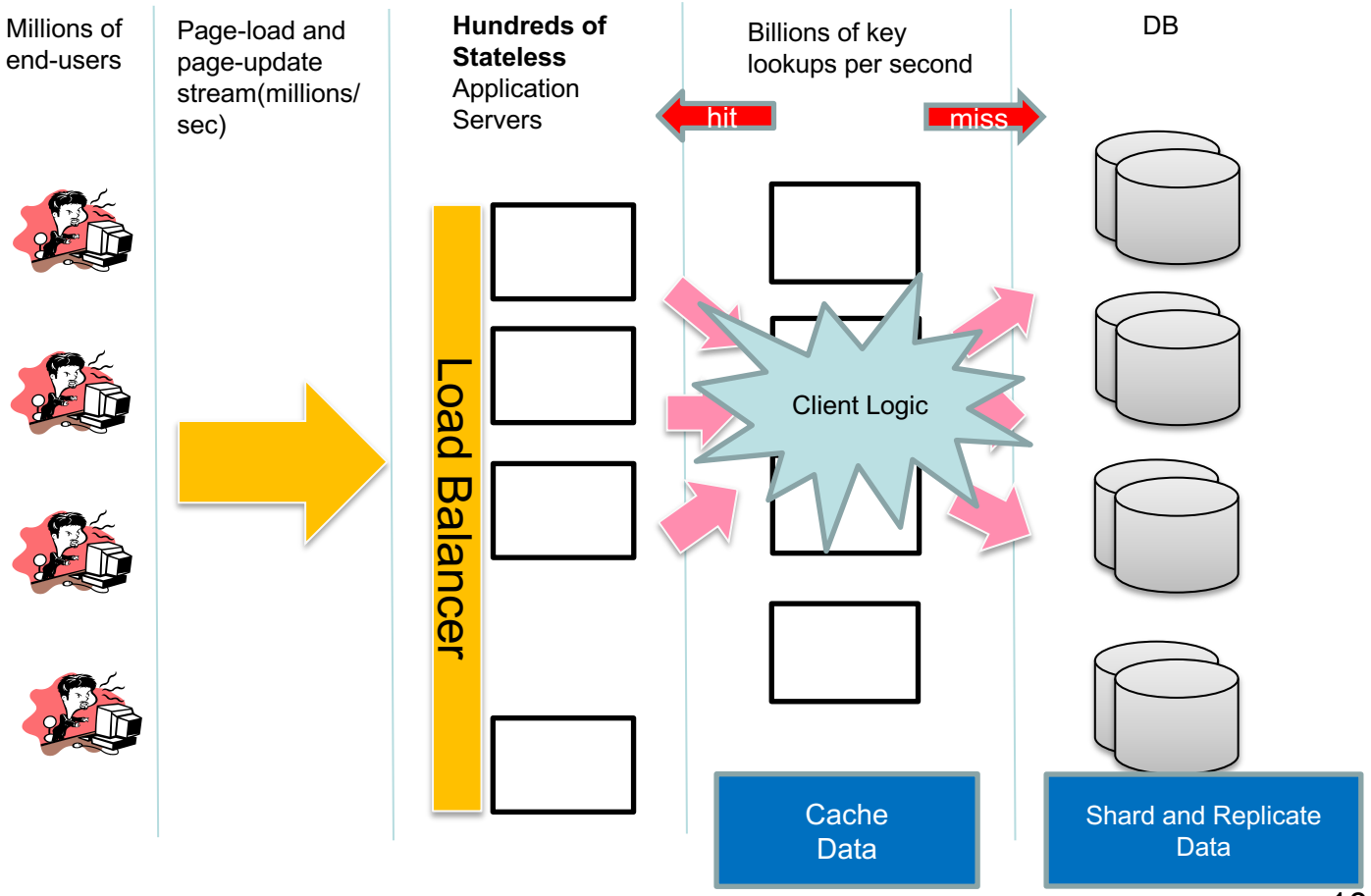
Why Separate Caching?



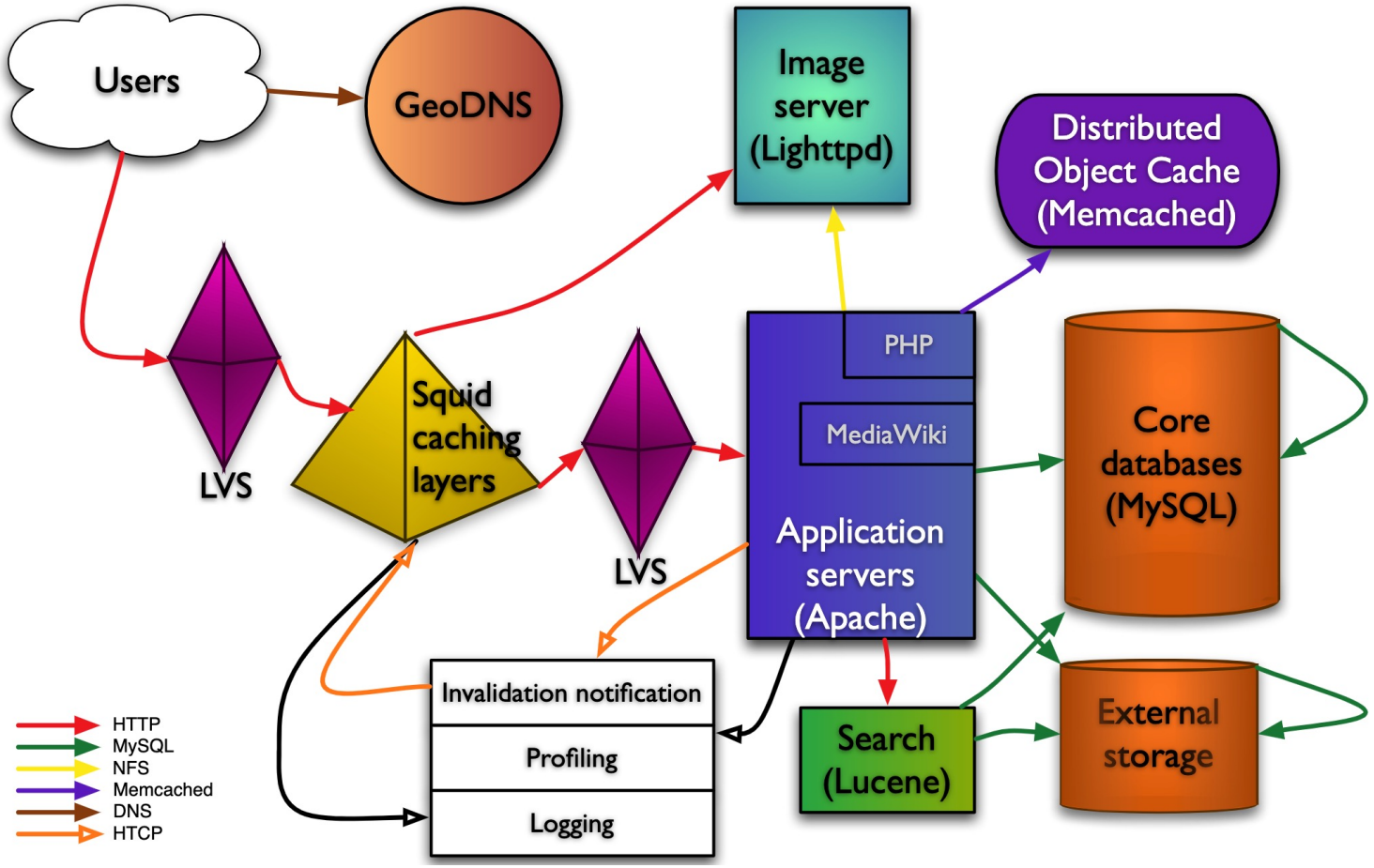
Why Separate Caching?



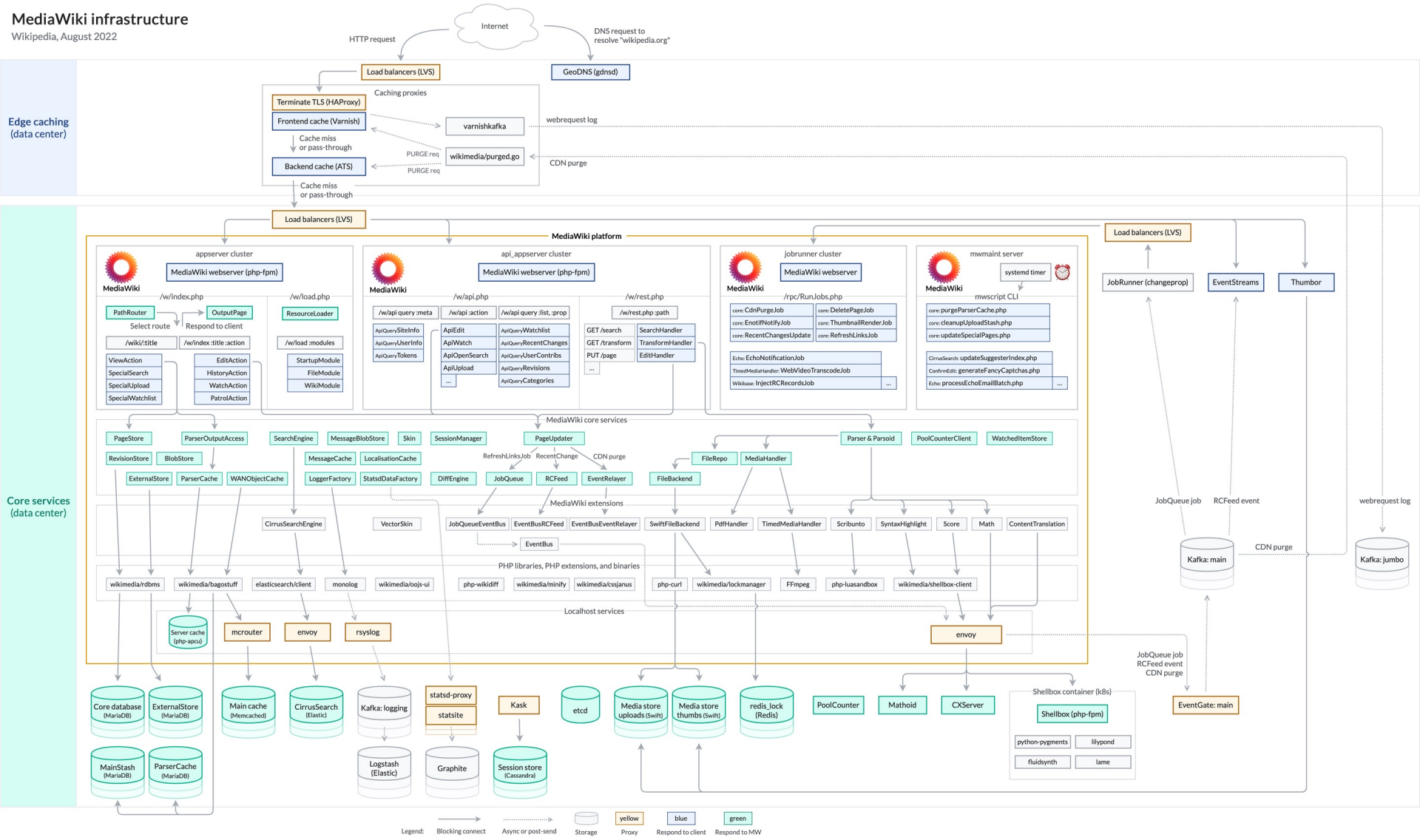
Why Separate Caching?



Wikipedia Architecture

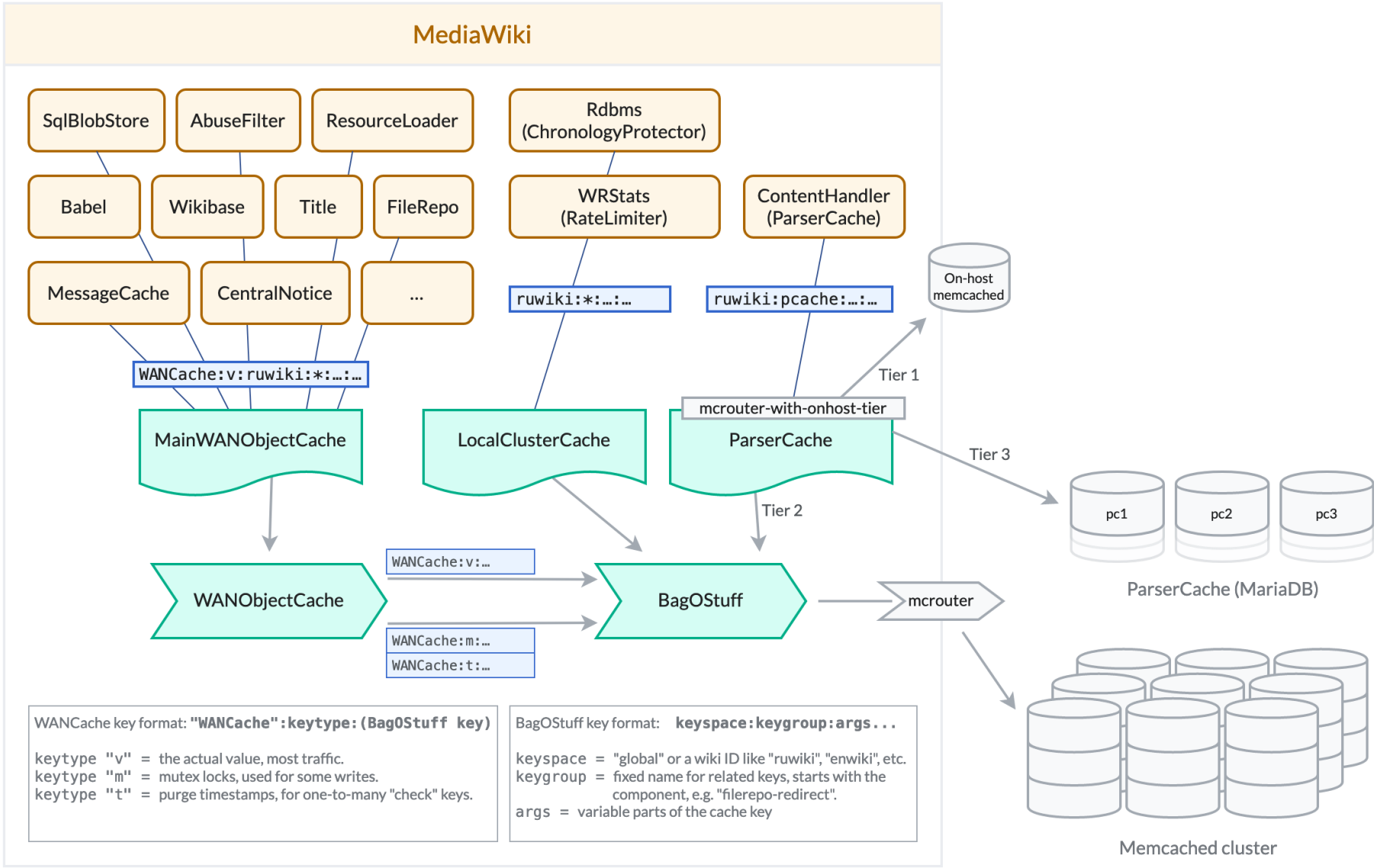


Wikimedia architecture by Mark Bergsma (2007)

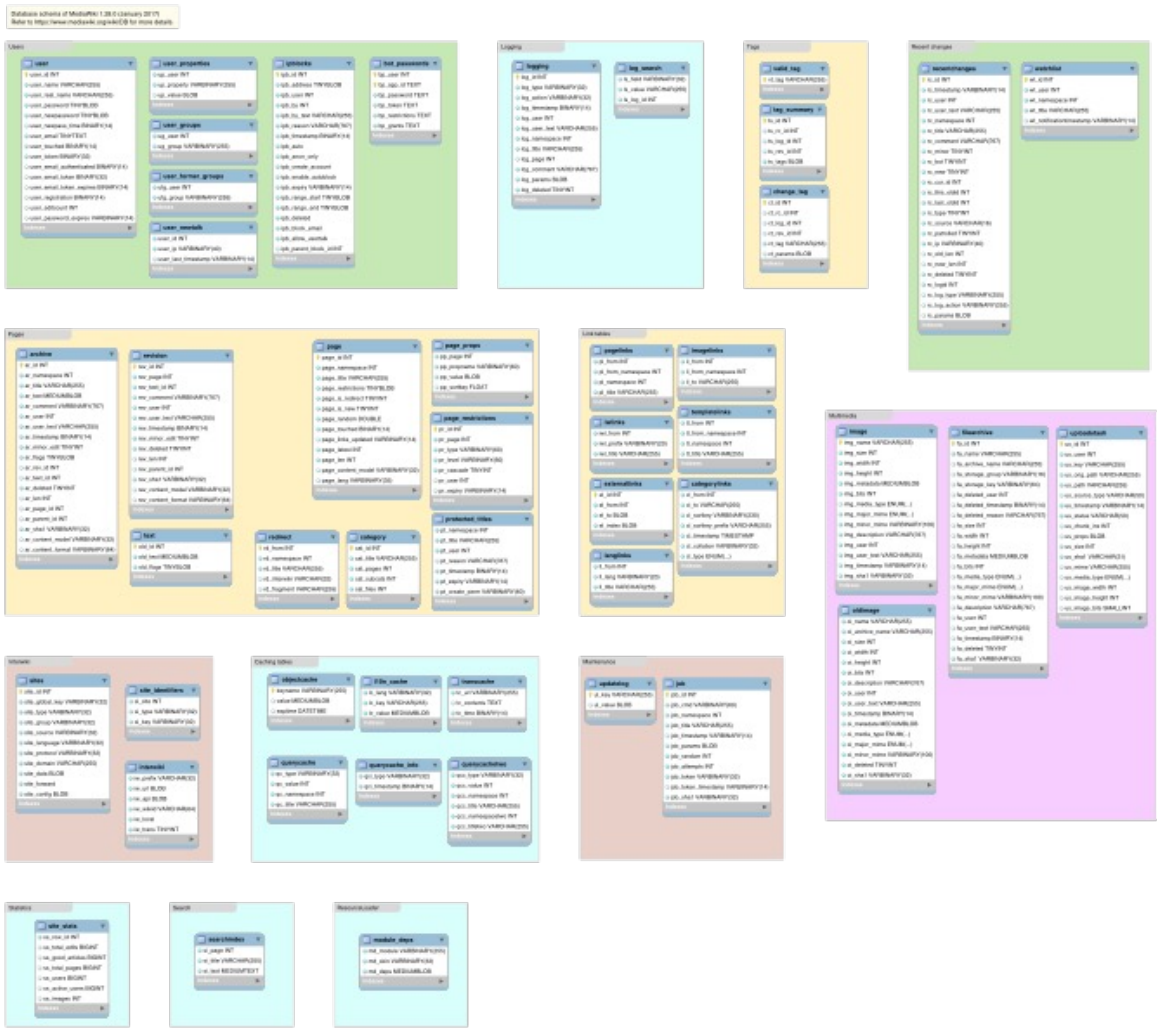

MediaWiki infrastructure
Wikipedia, August 2022
Internet
HTTP request
DNS request to resolve "wikipedia.org"
Load balancers (LVS)
GeoDNS (gdnsd)
Edge caching (data center)
Caching proxies
Terminate TLS (HAProxy)
Frontend cache (Varnish)
Cache miss or pass-through
Backend cache (ATS)
varnishkafka
wikimedia/purged.go
PURGE req
webrequest log
CDN purge
Core services (data center)
MediaWiki platform
appserver cluster
api_appserver cluster
jobrunner cluster
mwmaint server
MediaWiki webserver (php-fpm)
MediaWiki webserver
systemd timer
/w/index.php
/w/load.php
/w/api.php
/w/rest.php
/rpc/RunJobs.php
mwscript CLI
PathRouter
OutputPage
Select route
Respond to client
ResourceLoader
/wiki/:title
/w/index :title :action
/w/load :modules
/w/api query :meta
/w/api :action
/w/api query :list, :prop
/w/rest.php :path
ViewAction
SpecialSearch
SpecialUpload
SpecialWatchlist
EditAction
HistoryAction
WatchAction
PatrolAction
StartupModule
FileModule
WikiModule
ApiQuerySiteInfo
ApiQueryUserInfo
ApiQueryTokens
ApiEdit
ApiWatch
ApiOpenSearch
ApiUpload
ApiQueryWatchlist
ApiQueryRecentChanges
ApiQueryUserContribs
ApiQueryRevisions
ApiQueryCategories
GET /search
GET /transform
PUT /page
SearchHandler
TransformHandler
EditHandler
core: CdnPurgeJob
core: EnotifNotifyJob
core: RecentChangesUpdate
core: DeletePageJob
core: ThumbnailRenderJob
core: RefreshLinksJob
Echo: EchoNotificationJob
TimedMediaHandler: WebVideoTranscodeJob
Wikibase: InjectRCRecordsJob
core: purgeParserCache.php
core: cleanupUploadStash.php
core: updateSpecialPages.php
CirrusSearch: updateSuggesterIndex.php
ConfirmEdit: generateFancyCaptchas.php
Echo: processEchoEmailBatch.php
MediaWiki core services
PageStore
RevisionStore
BlobStore
ExternalStore
ParserCache
WANObjectCache
ParserOutputAccess
SearchEngine
MessageBlobStore
Skin
SessionManager
MessageCache
LocalisationCache
LoggerFactory
StatsdDataFactory
DiffEngine
PageUpdater
RefreshLinksJob
RecentChange
CDN purge
JobQueue
RCFeed
EventRelayer
FileRepo
MediaHandler
FileBackend
Parser & Parsoid
PoolCounterClient
WatchedItemStore
MediaWiki extensions
CirrusSearchEngine
VectorSkin
JobQueueEventBus
EventBusRCFeed
EventBusEventRelayer
SwiftFileBackend
PdfHandler
TimedMediaHandler
Scribunto
SyntaxHighlight
Score
Math
ContentTranslation
EventBus
PHP libraries, PHP extensions, and binaries
wikimedia/rdbms
wikimedia/bagostuff
elasticsearch/client
monolog
wikimedia/oojs-ui
php-wikidiff
wikimedia/minify
wikimedia/cssjanus
php-curl
wikimedia/lockmanager
FFmpeg
php-luasandbox
wikimedia/shellbox-client
Localhost services
Server cache (php-apcu)
mcrouter
envoy
rsyslog
envoy
Load balancers (LVS)
JobRunner (changeprop)
EventStreams
Thumbor
JobQueue job
RCFeed event
webrequest log
Kafka: main
CDN purge
Kafka: jumbo
JobQueue job
RCFeed event
CDN purge
EventGate: main
Core database (MariaDB)
ExternalStore (MariaDB)
MainStash (MariaDB)
ParserCache (MariaDB)
Main cache (Memcached)
CirrusSearch (Elastic)
Kafka: logging
Logstash (Elastic)
statsd-proxy
statsite
Graphite
Kask
Session store (Cassandra)
etcd
Media store uploads (Swift)
Media store thumbs (Swift)
redis_lock (Redis)
PoolCounter
Mathoid
CXServer
Shellbox container (k8s)
Shellbox (php-fpm)
python-pygments
lilypond
fluidsynth
lame
Legend:
Blocking connect
Async or post-send
Storage
yellow
Proxy
blue
Respond to client
green
Respond to MW

Memcached flow from MediaWiki

Wikipedia, August 2022

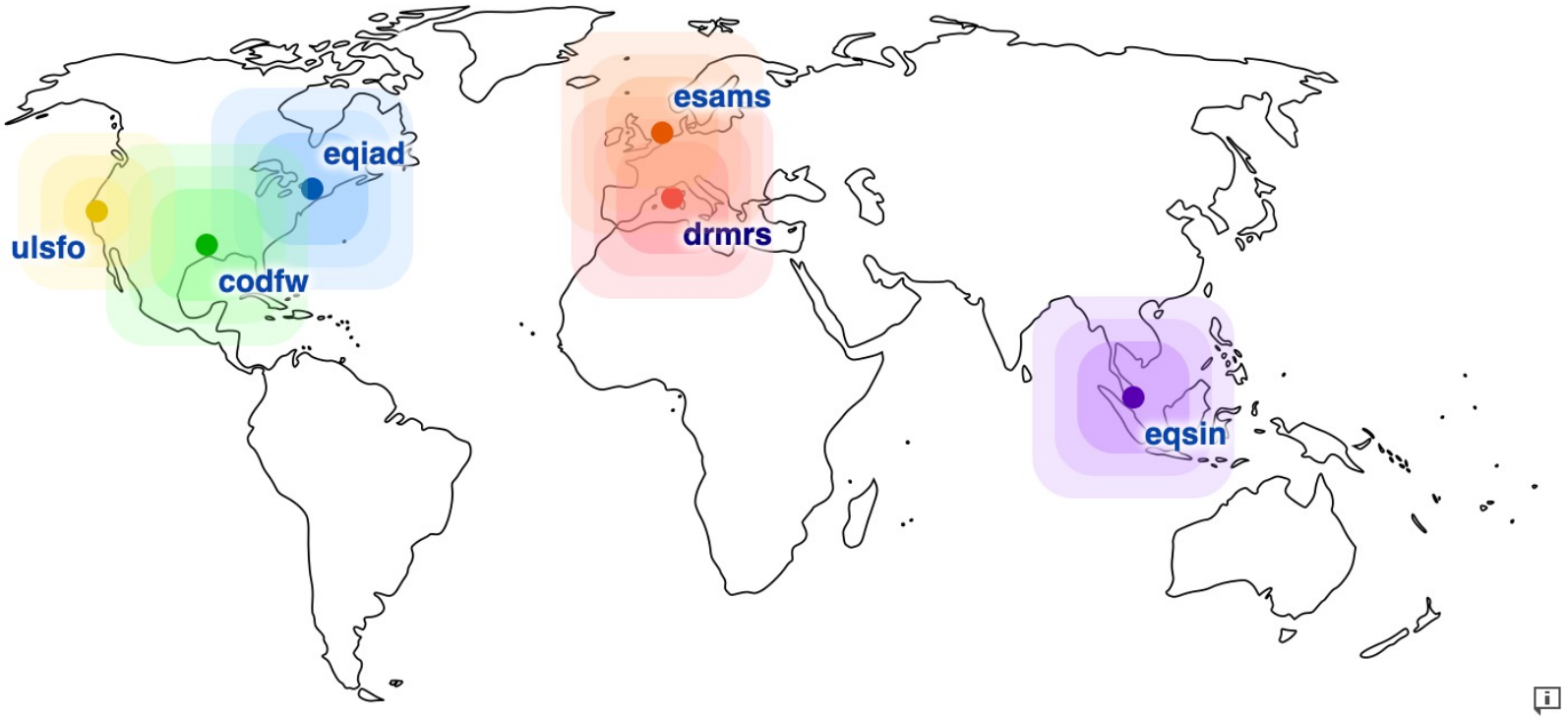


MediaWiki Database Schema



https://www.mediawiki.org/w/index.php?title=Manual:Database_layout/diagram&action=render

Map of Wikmiedia Data Centers



https://wikitech.wikimedia.org/wiki/Data_centers

Introduction to Memcached

- What is Memcached?
  - High-performance, distributed memory object caching system.
- When can we us it?
  - Anywhere where there is RAM
  - Used to cache *objects*
- Why should we us it?
  - If we have a high-traffic site that is dynamically generated with a high database load that contains mostly read threads

Introduction (cont.)

- Memcached is not:
  - A persistent data store
  - A database
  - Application-specific
  - A large object cache
  - Fault-tolerant or highly available

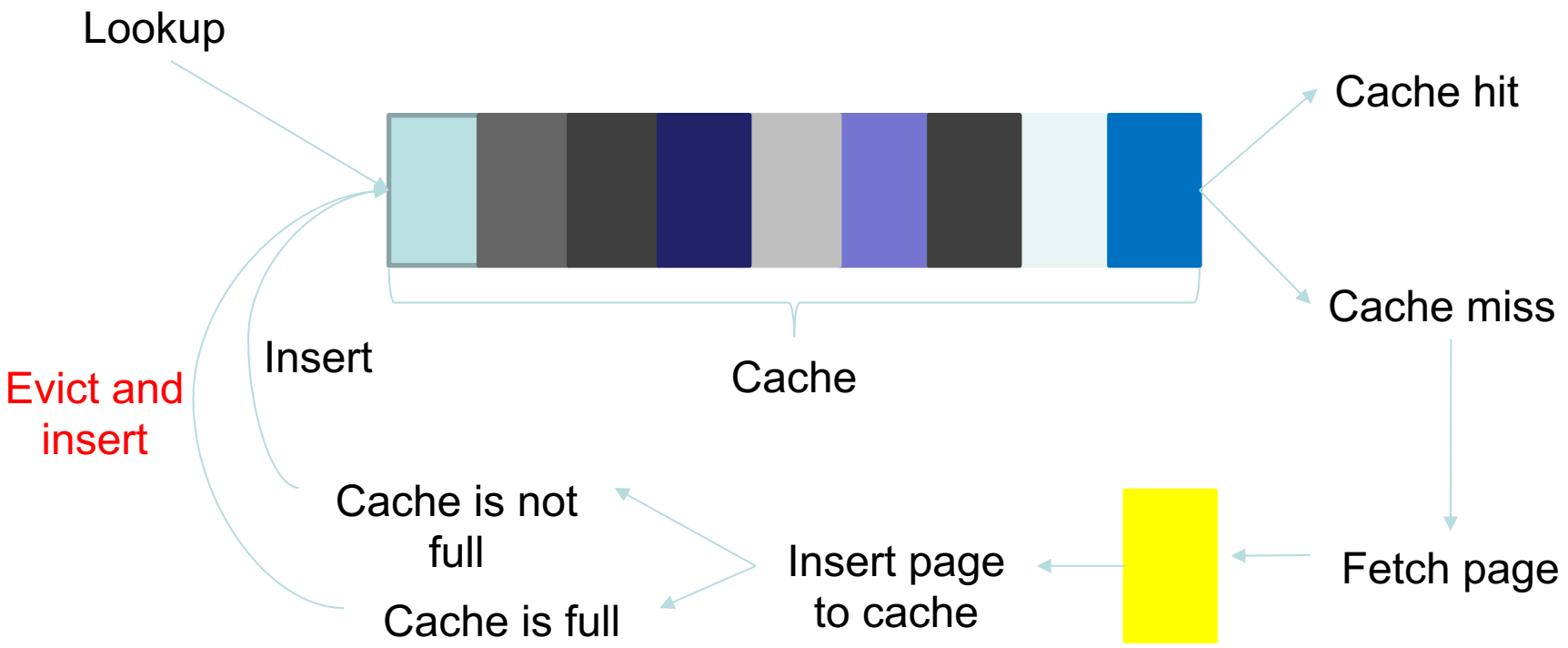
Introduction (cont.)

- Memcached is
  - Pure single-node
  - Key-value store (simple set/get operations)
  - Optimized for multithreading (compared to Redis)
- Limits
    - Key size = 250 bytes
    - Item size: The default value is 1 MB

Use cases for Caching

Site Type	Repeatable Use
Social Networking	Profile caching
Content Aggregation	HTML/page caching
Ad Targeting	Cookie/Profile tracking
Location-based services	DB Query scaling
E-Commerce	Session caching

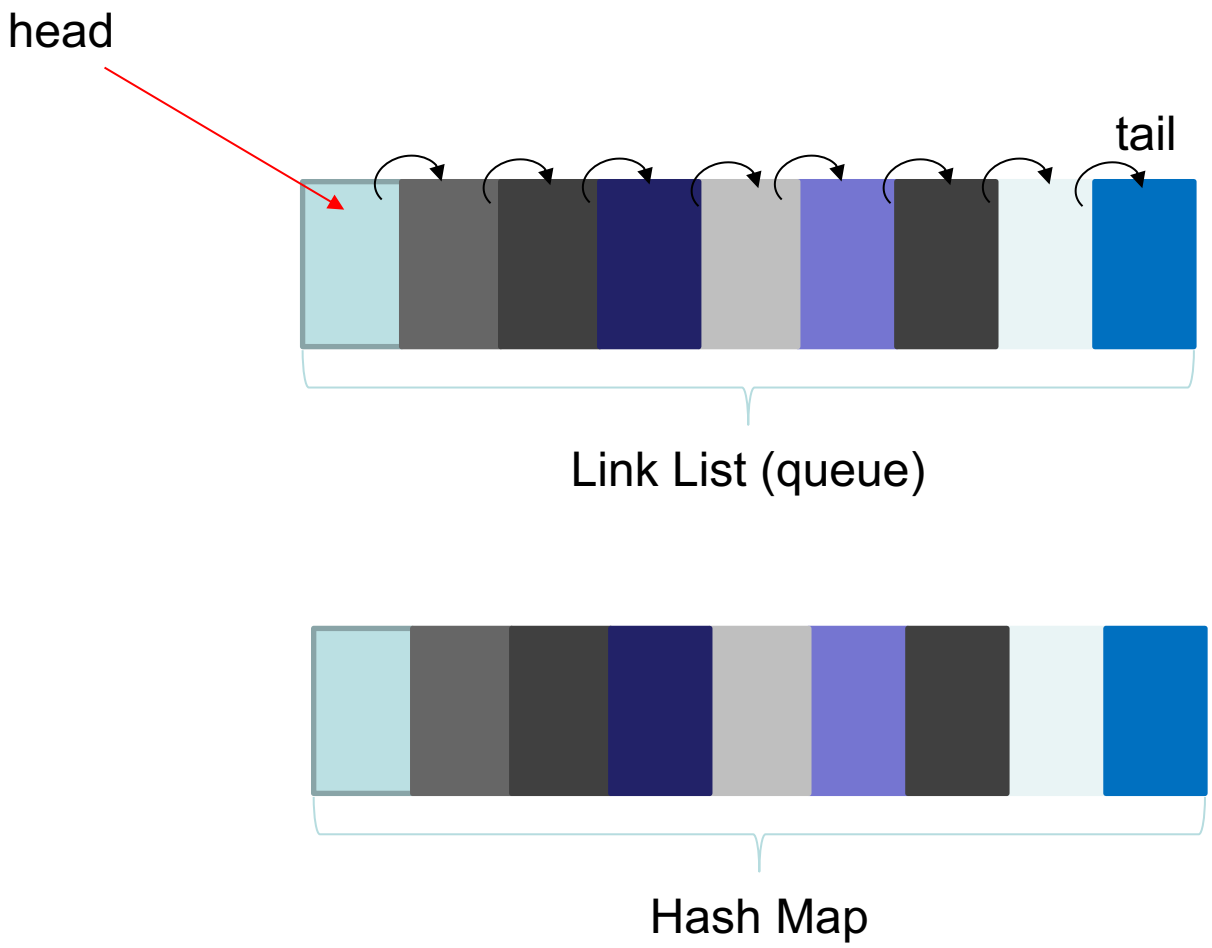
Cache Management



Cache Eviction
LRU Cache

- Least recently accessed items are cycled out
- Memcached deals with memory fragmentation using “slab” memory allocation. i.e., different categories of object size are grouped in similar memory allocated areas.
- One LRU exists per “slab class”
- LRU “evictions” need not be common
- Keys expiration time (exptime), however the LRU algorithm may remove expired keys before they are accessed.

Cache Eviction
LRU Cache



Install Memcached

- MacOS
  - brew install Memcached
- Linux
  - apt-get install memcached
- Windows:

  https://github.com/jefyt/memcached-windows

- libMemcache for C++
  - Adds many funcationalities, including consistent hashing
  - Recall, memcached is single-node
- Most libs are wrappers.

Using Memcached in Python

- https://github.com/pinterest/pymemcache
- Quick Start

Basic usage

```
from pymemcache.client.base import Client

client = Client('localhost')
client.set('some_key', 'some_value')
result = client.get('some_key')
```

Memcached cluster

```
from pymemcache.client.hash import HashClient

client = HashClient([
    '127.0.0.1:11211',
    '127.0.0.1:11212',
])
client.set('some_key', 'some value')
result = client.get('some_key')
```

Demo on telnet ☺

```
telnet 127.0.0.1 11211
set v0 0 10 1 h
set v1 0 0 10 TestValue1
get v1
get v0
delete v1
stats items
stats cachedump 1 10
```