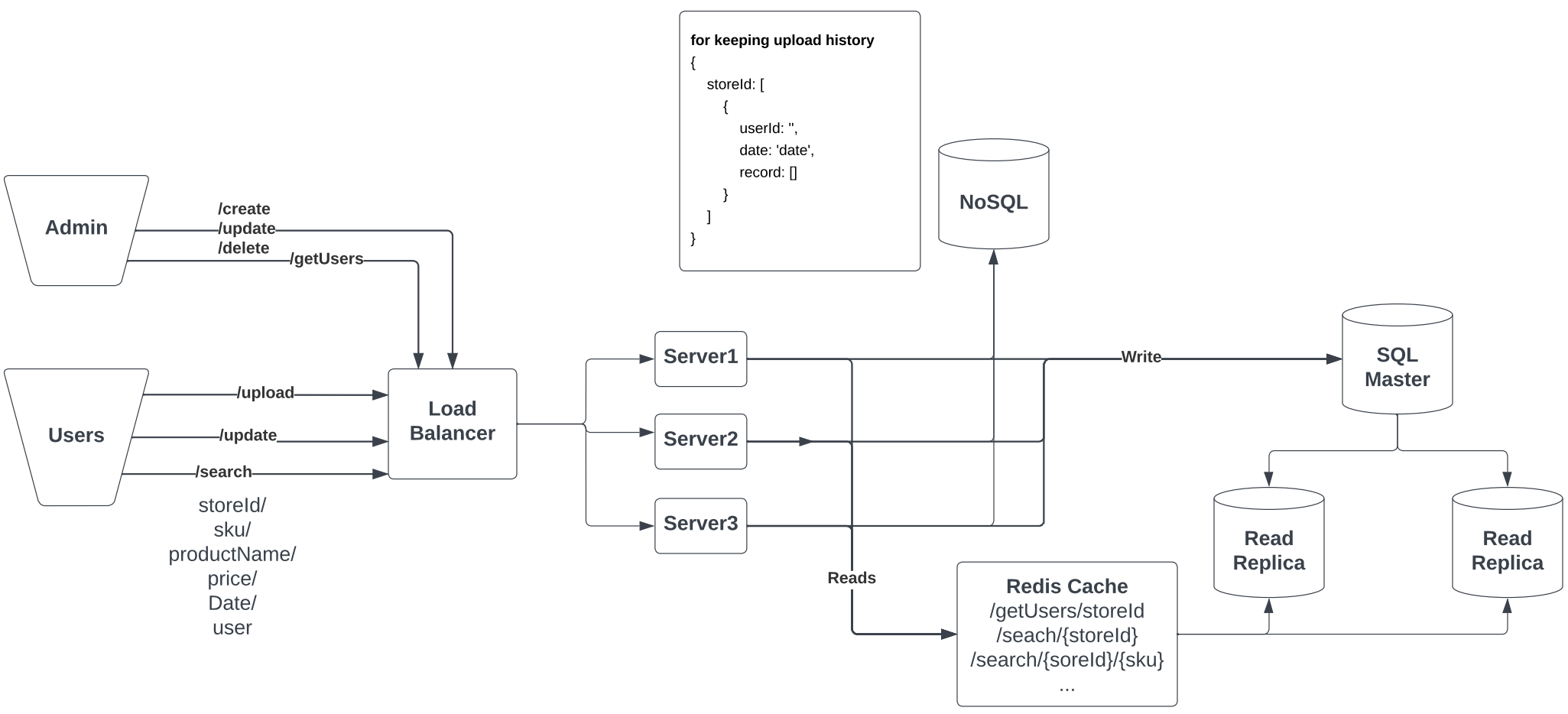
High Level Design

1. Context Diagram
2. Solution Architecture
3. Design Decisions:
   a. users will be created by admin. lb to avoid server failiure
   b. We choose SQL db
      i. schema is predfined
      ii. need ACID properties for the records.
      iii. data is not going to grow after a scale and we can verticlly scale or shard based on the storeId key
   c. will keep record for uploads and edits happening for tracking
   d. will keep the option available for upload as bulk edit.
4. Non-functional requirements considered and how the design addresses them
   a. we will https/tcp potocol:
      i. for file upload api we will use tcp: as we need to send data in chunks
      ii. for other apis https as request/response has to be at once.
   b. handle single point of failure: load balancer in front of multiple server
   c. handle large data: indexing && verticle scale && sharding
   d. consistency is more important then availabilty
   e. db read replicas to handle db failure and data backup
   f. cache to remember search records as its going to be heavy: redis/memcash with Proper TTL Policy
5. Assumptions
   a. csv file size is not big enough
   b. Users will be manged by admin for stores
   c. StoreId can not be edited as its unique for the store
   d. only users mapped to perticular store can edit records for that store
   e. search will return recrods only for that store which user is mapped to.
   f. history need not be maintained. so cost estimation is...
   g. only users mapped to store can upload, search store records
   h. uploaded csv is an latest pricing feed.
6. Source for the implementation

QUESTIONS

1. CSV file size limitaions?
2. new csv is an update list or latest list for store feed?
   a. if its an update list then we should update the matching rows and add new rows and not delete missing rows.
   b. if its an lates list then we should delete all rows for the store and add new rows as per the new csv.
3. Does history needs to maintained?
   a. for old csv
   b. for fields update
4. How many products one store can have
   a. avarage would tell us how storage we need for handle db scaling



Low Level Design

Users

id,
name,
store ID,
createdDate
createdByUserId
designation
refreshToken
email
password
userType
isActive

PricingFeed

createdByUserId,
editedByUserId,
storeId (Primary),
sku,
productName (Primary),
Price,
currency,
createdDate,
editedDate