Load can be described using a load parameter. A load parameter can be something like requests per second to a web server, the ratio of reads to writes in a database, the number of simultaneously active users in a chat room, the hit rate on a cache, etc...

Let us consider Twitter. Two of Twitter's main operations are:

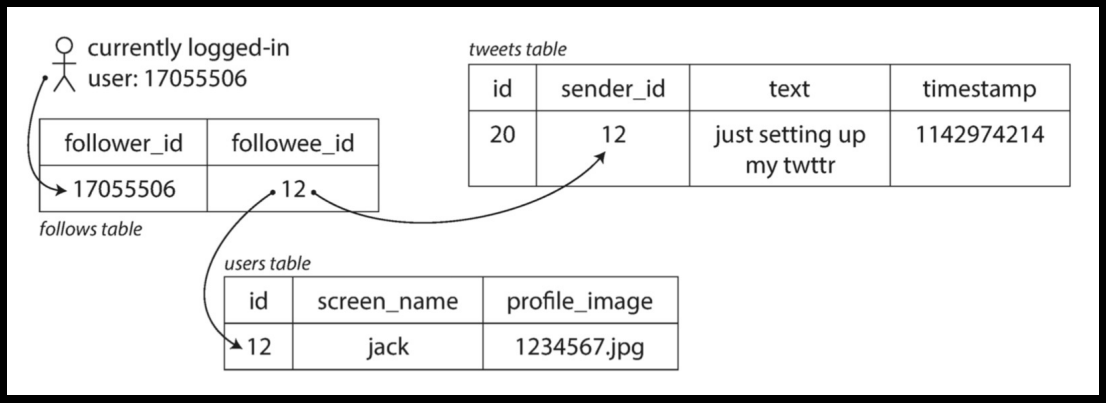
Post tweet:
A user can publish a new message to their followers (4.6k requests/sec on average, over 12k requests/sec at peak).

Home timeline:
A user can view tweets posted by the people they follow(300k requests/sec)

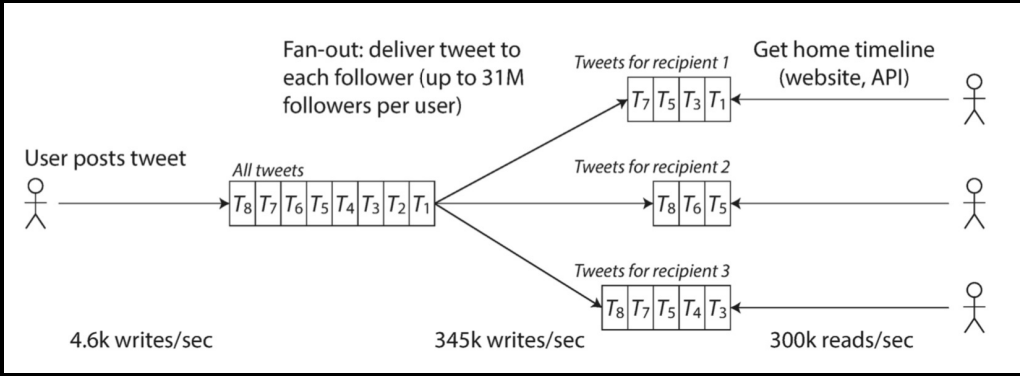
Twitter's scaling challenge is not primarily due to tweet volume, but due to fan-out- each user follows many people, and each user is followed by many people.

There are 2 ways of implementing these 2 operations:

1. Posting a tweet simply inserts the new tweet into a global collection of tweets. When a user requests their home timeline, look up all the people they follow, find all the tweets for each of those users, and merge them sorted by time (followee - the person being followed)



2. Maintain a cache for each user's home timeline-like a mailbox of tweets for each recipient user. When a user posts a tweet, look up all the people who follow that user, and insert the new tweet into each of their home timeline caches. This way then, the request to read the home timeline is then cheap, because its result has been computed ahead of time.



The first version of Twitter used approach 1, but the systems struggled to keep up with the load of home timeline queries, so the company switched to approach 2. This works better because the average rate of published tweets is almost two orders of magnitude (100x) lower than the rate of home timeline reads, and so in this case it's preferable to do more work at write time and less at read time.

The downside of approach 2 is that posting a tweet now requires a lot of extra work. On average, a tweet is delivered to about 75 followers. So:

75 x 4.6k tweets/sec = 345 k writes/sec.

...345k writes per second to the home timeline caches. Some users have over 30 million followers, so a single tweet may result in over 30 million writes to home timelines. Since Twitter tries to deliver tweets to followers within 5 seconds, this can become a challenge.

This makes the distribution of followers per user a key load parameter for discussing scalability, since it determines the fan-out load.