Q1.

Part 1: Building

Let’s say a developer wants to deploy code corresponding to a commit SHA.  
What seems to be a right choice is a Queuing mechanism to queue all the requests. And worker pool building the code commits in FIFO manner.  
Workers will then store the binary in blob store.

Since we want to persist the history of jobs. We can think of a SQL table for storing the entries (this can represent our queue).

How to work this as a queue ?  
We can select the jobs where status = QUEUED and has the oldest created\_at timestamp.  
Hence the index on status and created\_at will improve performance.

The fact that we have SQL database, we have ACID transactions. This enables our X number of workers to query and update the jobs as every run is a transaction and hence concurrency safe.

**What if our worker crashes while building code ?**

We can tackle it by adding a column last\_heartbeatIt will continuously send heartbeat signals to the job table. Let’s say a build takes around 15 mins, the workers will send this heartbeat every 3 mins.

**How many workers are actually needed ?**

Assuming 5000 builds/day. And 15 mins/build → ~ 100 builds/day that a single worker will perform. Hence we can say that 5000/100 = 50 workers are required on average. We can horizontally scale the number of workers on peak hours and vice versa.

**What if we want the developers to be able to deploy the build only if the binary is replicated to all the regions ?**

To tackle this requirement we can have a really simple service which polls the master blob store for any new binary and tracks the status of that binary by polling to all the regional blob stores.

The build is only deployable if the replication\_status in the table is “COMPLETED”.

## Part 2: Deploying

We assumed earlier that the build takes 15 mins and let’s say replication takes another 5 mins. So for us to meet the 30 min deadline for entire deployment, we are left with 10 mins. For 100K machines to download a 10 GB file from the blob store over network seems unreasonable. Hence we can create a Peer-to-Peer network. All the machines in a region are part of a Peer to Peer network, this will enable them to download multiple such binaries really fast.