SuperFast PostgreSQL

Don't bother with a RAM disk. Using following settings you'll end up not using disk at all.

No reliability on cluster level

- Use `fsync=off` setting. Whole cluster becomes completely unreliable and very fast.
- Now set `synchronous_commit=off`, you won't need it anyway when `fsync` is turned off.
- Set `full_page_writes=off` too, it won't do much difference now.
- Create `UNLOGGED` tables only, it will remove the last "durability" option.

Use such settings only for testing, never for production.

Partial reliability

- Set `synchronous_commit=off` and `commit_delay` to few seconds, it will give you performance almost the same as `fsync=off`, but won't endanger data that much. You can lose a commit, not more.

Real tuning

- Set `max_connections` to exact number of clients expected, if it is over 250, consider using a pooler.
- Set `shared_buffers` to 512MB in Windows, ¼ RAM or more on *nix.
- Set `checkpoint_segments` to 32 (checkpoint every 512MB of WAL segments) for basic write-heavy system, and continue increasing it up to 256 (every 4GB).
- Set `checkpoint_completion_target` to 0.75, or even 0.9
- Set `effective_cache_size` to value that system can spare freely for PostgreSQL as a disk cache.
- Set `work_mem` to such value, that `max_connections*work_mem*max-query-table-count` won't exceed free RAM left. Few large queries can consume some GBs of RAM so use it carefully.
- Set `random_page_cost` to 4.0 on generic hardware, 2.0 on fast RAID of SCSI disks.
- Set `seq_page_cost` to 0.7 on fast disks or leave it the default 1.0
- Turn `auto_explain` on to catch slow queries.
- Always `TRUNCATE` tables instead of `DELETE`, it works way faster when you need just dump all the data in the table.