Postmortem: delayed payroll exports on October 14, 2026

Harbor & Pine Payroll is a fictional company. This sample document was written for pdfconversionapi.com and is released under CC0 1.0.

Field	Detail
Status	Resolved
Severity	SEV-2 (customer-facing delay, no data loss)
Duration	2 h 47 min (09:12 to 11:59 Pacific)
Authors	Platform on-call, Payroll Exports team

Summary

On Tuesday, October 14, the nightly ACH export for 312 employer accounts finished **2 hours 47 minutes late**. Employees were paid on time because the bank cutoff is 14:00 Pacific, but 41 employers opened support tickets when the export dashboard showed `PENDING` past 09:00.

No payroll data was lost or changed. The root cause was a database migration that added an index without `CONCURRENTLY`, which locked the `pay_runs` table while the export job waited.

Timeline (Pacific)

1. **02:00** The scheduled migration `2026_10_14_add_run_status_idx` starts.
2. **02:05** The nightly export job starts and blocks on the table lock.
3. **09:12** Support escalates the first ticket to on-call.
4. **09:40** On-call finds the lock with `pg_locks` and cancels the migration.
5. **11:59** The export queue drains and the last file reaches the bank.

What went well

- Alerting on the bank cutoff gave us a clear deadline.
- The runbook for stuck exports was current and worked as written.
  - Step 3 (find blocking queries) took under five minutes.
  - Step 5 (re-queue failed files) needed no changes.

What went wrong

The migration ran in the same window as the export job, and nothing in our deploy checks looks at table locks.

Our migration linter checks naming and reversibility but not lock behavior. The export job also had no timeout on its first query, so it waited for seven hours instead of failing loudly.

Root cause

The migration ran this statement:

```
CREATE INDEX idx_pay_runs_status
    ON pay_runs (status, scheduled_for);
```

Without `CONCURRENTLY`, PostgreSQL takes a `SHARE` lock on `pay_runs` for the whole build. On a table with 48 million rows, that took longer than the export window.

Action items

- [x] Re-run the index build with `CREATE INDEX CONCURRENTLY` (done October 15)

- [x] Add a 60-second `statement_timeout` to the export job's first query
- [ ] Teach the migration linter to reject non-concurrent index builds on tables over 1 million rows
- [ ] Move scheduled migrations out of the 01:00 to 06:00 export window

---

Questions about this postmortem go to the `#payroll-platform` channel. See the PostgreSQL documentation on building indexes concurrently for background.