RLS capability-check obrazac za migracije — V152 hardening V144/V147 (MC #106695, 2026-08-02)

MC #106695 — RLS hardening for V144/V147 — BUILD EVIDENCE

Agent: bm-106695-build (Bruce Momjian) Date: 2026-08-02 Repo: /Users/makinja/business/ALAI-Holding-AS/products/Bilko Worktree: `.claude/worktrees/bm-106695` (new, created this session) Branch: `fix/106695-rls-migration-hardening` Base: `azdo/main` @ `e81db44ab943076d7cc2b6e4fbb0326993eef2ff` (cached ref, 2026-07-29 12:00:34 UTC — `git fetch azdo` fails headlessly in this environment, reproduced live this session: `fatal: could not read Username for 'https://dev.azure.com/alai-holding/Bilko/_git/Bilko': Device not configured`)

NOT marked done. Per the team-lead's dispatch, Mehanik requires an independent peer review for this RLS-scope task before ready/done. No merge, no deploy, no push performed by this agent.

Sources read in full before building: `~/system/prompts/forged/106695.md` (285 lines), `~/system/evidence/106342/audit-2026-08-02.md`, `~/system/evidence/106302/fix-verdict.md`, and the live repo files V17, V30, V66, V111, V144, V147, V148, V149 in the new worktree.

D1 — Live-state verification (read-only, before writing SQL)

Access chain (identical method to `~/system/evidence/106342/audit-2026-08-02.md` §1): Vault entry `Azure Service Principal — alai-cli-deployer` (Contributor @ subscription `5b0b4d9b-e677-464e-abf0-5170cbce3b8e`), logged in via an **isolated** `AZURE_CONFIG_DIR` (scratchpad-local — this session's Azure CLI state never touched the shared `~/.azure` profile other concurrent agents may depend on). `az containerapp secret show -g rg-bilko-demo -n bilko-api-demo --secret-name db-password` → live `bilko_admin` password. Session logged out immediately after the two read-only queries below; the fetched password was written to a scratchpad-local file with `chmod 600` and deleted after use.

1. Was V147 actually applied? (unresolved from the 2026-08-02 audit — resolved here)

```
psql "host=bilko-demo-pg.postgres.database.azure.com port=5432 dbname=bilko user=bilko_admin
sslmode=require" \
  -tAc "SELECT version, description, installed_on FROM flyway_schema_history WHERE version IN
('144','147') ORDER BY version;"

144|seed ci tenant chart of accounts|2026-07-23 13:43:48.231407
147|seed e2e admin user            |2026-07-24 18:14:51.597826
```

RESOLVED: V147 IS applied, `installed_on 2026-07-24 18:14:51.597826`. The prior audit's `flyway_schema_history` dump omitted V147 from its result set without explanation while its migration table reviewed V147 as though live — that omission appears to have been a query/scope artifact of that session, not evidence of non-application. Both V144 and V147 are confirmed live on `bilko-demo-pg`, and both are therefore treated as **applied** and are **not edited** (Dev Rule #6 / Flyway `validateOnMigrate=true` checksum enforcement).

2. Re-confirmed role attributes and FORCE RLS flags (unchanged from the audit)

```
psql ... -tAc "SELECT rolname, rolsuper, rolbypassrls, rolcanlogin FROM pg_roles WHERE
rolname='bilko_admin';"
bilko_admin|f|t|t

psql ... -tAc "SELECT c.relname, c.relrowsecurity, c.relforcerowsecurity,
pg_get_userbyid(c.relowner) AS owner
  FROM pg_class c JOIN pg_namespace n ON n.oid=c.relnamespace
  WHERE n.nspname='public' AND c.relname IN ('accounts','users','entra_external_identities')
ORDER BY c.relname;"
accounts                |t|t|bilko_admin
```

```
entra_external_identities|t|t|bilko_admin
users                    |t|t|bilko_admin
```

Confirmed live, 2026-08-02, this session: `bilko_admin` (`rolsuper=f`, `rolbypassrls=t`, `rolcanlogin=t`) still owns all three FORCE-RLS target tables, and `BYPASSRLS` is still the Azure-provisioned admin-login attribute described in the audit — not set by any migration. This is exactly why V152 (below) does not rely on that fact being permanent.

D2 — New migration

File: `apps/api/src/main/resources/db/migration/V152__harden_ci_and_e2e_seed_rls_safety.sql`

V-number selection (collision-avoidance procedure, run twice)

Enumerated **every** worktree on disk (~140 checked via `git worktree list` + `find ... -path "*/db/migration/V1*.sql"`) **and** every `azdo` remote branch's migration tree (`git for-each-ref refs/remotes/azdo/` + `git ls-tree <branch> -- apps/api/.../db/migration/`), not just `azdo/main` (whose cached tip here is only V149). `git fetch azdo` was attempted directly and failed immediately and reproducibly: `fatal: could not read Username for 'https://dev.azure.com/alai-holding/Bilko/_git/Bilko': Device not configured`.

Highest V-number found anywhere, **both times** (once before creating the worktree, once immediately before writing the file):

```
V150__capture_financial_audit_ap_tables.sql
V151__purchase_invoices_oib_identity_and_document_type.sql
```

— both present, identically, on two branches: `azdo/feat/106632-phase0-schema-spike` and `azdo/feat/106632-phase1-read-swap`. No V152 found anywhere on either enumeration pass.

V152 used. Flagged PROVISIONAL — re-verify against all remote branches again before merge, per the exact V148/V149 collision this repo hit twice already (`~/system/evidence/106302/fix-verdict.md`).

Shape — matches the D2 spec exactly

- **Header**: cites MC #106695, the audit, explains why V144/V147 are not edited (checksum + Dev Rule #6, same rationale V148's revision header uses), and includes the D1 verification output verbatim.

- **Block A** (CI tenant chart-of-accounts, `public.accounts`): capability check (`rolsuper OR rolbypassrls`) → BYPASSRLS/superuser direct INSERT, or owner (`NO FORCE` → INSERT → restore `FORCE` → assert restored) → payload byte-identical to V144's INSERT (same CI org `2e852173-170e-5f18-adf9-253ce4922444`, same 5 rows, same `ON CONFLICT (organization_id, code) DO NOTHING`) → else `RAISE EXCEPTION`.
- **Block B** (E2E admin user, `public.users` UPDATE + `public.entra_external_identities` INSERT): same capability shape, but ownership is verified **per table** in a loop (`users` and `entra_external_identities` are independently FORCE-RLS with independent policies — V30, V66 — so one check does not clear both), `NO FORCE` is applied per table via a single dynamic `EXECUTE format(...)` (both tables relaxed/restored from one source line), repairs exactly V147's existing-user UPDATE + the entra INSERT that follows it (both previously covered only by a comment, not a runtime check), else `RAISE EXCEPTION` per table.
- `schema_version` registration at the end, same convention as V144's tail.
- **DOWN script** as a trailing comment (ZAKON PI2).

Acceptance signal — verified against the final file

```
grep -c "SELECT rolsuper OR rolbypassrls" V152...sql   -> 2   (expect 2)
grep -c "RAISE EXCEPTION" V152...sql                    -> 5   (expect >= 2)
grep -c "NO FORCE ROW LEVEL SECURITY" V152...sql        -> 2   (expect 2)
grep -c "INSERT INTO schema_version" V152...sql         -> 1   (expect 1)
```

All four acceptance signals from the forged spec match exactly (one iteration was needed: the first draft used `SELECT (rolsuper OR rolbypassrls)` with parentheses, matching V149's own style but not the literal grep string the spec tests against — fixed to the unparenthesized form in both blocks).

D3 — psql-provable proof (local scratch PostgreSQL, no Gradle)

PostgreSQL version note: the spec asks for PostgreSQL 15; only PostgreSQL 18.3 (Homebrew, `/opt/homebrew/opt/postgresql@18`) is installed on this machine — PostgreSQL 15 is not available locally and was not installed for this task. `ENABLE`/`FORCE ROW LEVEL SECURITY`, `BYPASSRLS`, and RLS policy evaluation are unchanged in behavior between PG15 and PG18 (stable since RLS's introduction in PG9.5); this is flagged as a version deviation from the spec, not concealed.

Fixture (`fixture-106695.sql`, scratchpad-local): faithful re-creation of `accounts` (V17), `users` (V30), `entra_external_identities` (V66) — same columns needed for the migration's writes, same `ENABLE ROW LEVEL SECURITY` + `CREATE POLICY org_isolation`/`entra_org_isolation ... TO bilko_app` (predicate

text copied verbatim from the live migrations) + `FORCE ROW LEVEL SECURITY`, tables owned by a non-superuser, non-BYPASSRLS role (`flyway_sim`, matching bilko_admin's real Azure shape: `rolsuper=f`). Four roles created: `flyway_sim` (owner, no BYPASSRLS/superuser — **the actual production shape**), `admin_sim` (`BYPASSRLS`), `stranger_sim` (neither, no table ownership), `bilko_app` (the RLS policy's `TO` target, unused directly by this migration but created for fixture completeness). Cluster: `pg_ctl` scratch instance, Unix socket only (`/tmp/pg106695-sock`, port 5599), no shared/demo database touched. Four **separate** databases (`rlsfix_run1`..`rlsfix_run4`) — fresh fixture state per run, as required.

Run 1 — BYPASSRLS/superuser branch (`admin_sim`)

Pre-seeded (as superuser) one existing, unpromoted e2e admin user row so Block B's UPDATE has a real target to prove against (not just a trivial 0-row no-op):

```
$ psql -h /tmp/pg106695-sock -p 5599 -U admin_sim -d rlsfix_run1 -1 -v ON_ERROR_STOP=1 -f
V152__harden_ci_and_e2e_seed_rls_safety.sql
NOTICE:  V152 Block A: current_user=admin_sim bypasses RLS (superuser/BYPASSRLS) — inserting
directly.
NOTICE:  V152 Block A (MC #106695): CI tenant chart-of-accounts repair complete for org
2e852173-170e-5f18-adf9-253ce4922444.
DO
NOTICE:  V152 Block B: current_user=admin_sim bypasses RLS (superuser/BYPASSRLS) — updating
directly.
NOTICE:  V152 Block B (MC #106695): 1 users row(s) promoted to is_platform_admin=TRUE;
entra_external_identities link ensured for bilko-e2e-admin@bilkociam.onmicrosoft.com.
DO
INSERT 0 1
$ echo exit=$?
exit=0
```

Direct verification (as superuser, not trusting the migration's own NOTICE):

```
accounts (CI org)               : 5
users.is_platform_admin         : bilko-e2e-admin@bilkociam.onmicrosoft.com | t
entra_external_identities (match): 1
relforcerowsecurity accounts/entra_external_identities/users: t | t | t   (untouched
throughout — bypass branch never calls NO FORCE)
```

Run 2 — Owner branch (`flyway_sim` — the actual production shape)

Pre-seeded the same unpromoted user **as superuser** (seeding it as `flyway_sim` itself was attempted first and correctly **failed** — `ERROR: new row violates row-level security policy for table "users"` — proving the fixture's FORCE RLS genuinely blocks the owner too, before any migration logic runs). Confirmed pre-run that `flyway_sim` cannot see the seeded row at all:

```
$ psql -h /tmp/pg106695-sock -p 5599 -U flyway_sim -d rlsfix_run2 -tAc "SELECT count(*) FROM users;"
0
```

Migration run:

```
$ psql -h /tmp/pg106695-sock -p 5599 -U flyway_sim -d rlsfix_run2 -1 -v ON_ERROR_STOP=1 -f V152__harden_ci_and_e2e_seed_rls_safety.sql
NOTICE:  V152 Block A: current_user=flyway_sim owns accounts — FORCE RLS relaxed for this transaction.
NOTICE:  V152 Block A (MC #106695): CI tenant chart-of-accounts repair complete for org 2e852173-170e-5f18-adf9-253ce4922444.
DO
NOTICE:  V152 Block B: current_user=flyway_sim owns users/entra_external_identities — FORCE RLS relaxed for this transaction (tables: {users,entra_external_identities}).
NOTICE:  V152 Block B (MC #106695): 1 users row(s) promoted to is_platform_admin=TRUE; entra_external_identities link ensured for bilko-e2e-admin@bilkociam.onmicrosoft.com.
DO
INSERT 0 1
$ echo exit=$?
exit=0
```

Direct verification (as superuser):

```
accounts (CI org)               : 5
users.is_platform_admin         : bilko-e2e-admin@bilkociam.onmicrosoft.com | t
entra_external_identities (match): 1
relforcerowsecurity, AFTER (direct query, not the migration's NOTICE): accounts=t entra_external_identities=t users=t
```

Tenant isolation confirmed restored, not left weakened — re-checked as `flyway_sim` with no `app.current_org_id` set, post-migration:

```
$ psql -h /tmp/pg106695-sock -p 5599 -U flyway_sim -d rlsfix_run2 -tAc "SELECT count(*) FROM users;"
0
```

Same result as the pre-run check (0) — FORCE RLS is genuinely back in effect, not merely reported as such.

Run 3 — Neither branch (`stranger_sim`) — the OCD-5 failure mode this migration exists to prevent

```
$ psql -h /tmp/pg106695-sock -p 5599 -U stranger_sim -d rlsfix_run3 -1 -v ON_ERROR_STOP=1 -f V152__harden_ci_and_e2e_seed_rls_safety.sql
ERROR:  V152 (MC #106695) Block A refuses to run silently: current_user=stranger_sim is neither superuser nor BYPASSRLS, and does not own public.accounts (owner=flyway_sim). accounts has FORCE ROW LEVEL SECURITY with a fail-closed org_isolation policy, so this INSERT would silently apply to no visible target and Flyway would still record success — the BUG-005/V65 failure mode (MC #103001), the same class V144 itself was exposed to. Run Flyway as the accounts owner or as a BYPASSRLS role.
CONTEXT:  PL/pgSQL function inline_code_block line 31 at RAISE
$ echo exit=$?
exit=3
```

Rollback verified — nothing partially applied:

```
$ psql -h /tmp/pg106695-sock -p 5599 -U postgres -d rlsfix_run3 -tAc "SELECT count(*) FROM accounts WHERE organization_id='2e852173-170e-5f18-adf9-253ce4922444';"
0
$ psql -h /tmp/pg106695-sock -p 5599 -U postgres -d rlsfix_run3 -tAc "SELECT count(*) FROM schema_version WHERE version='152.0.0';"
0
```

Non-zero psql exit (3), row count 0 both before and after — matches the spec's acceptance signal exactly. This is the proof that turns BUG-005/V65-class silent success into a loud, catchable deploy failure.

Run 4 — Idempotency / no-op-on-already-applied-data

Seeded, as superuser, the **exact rows** V144/V147 already wrote live (simulating today's `bilko-demo-pg` state): the same 5 `accounts` rows at the same IDs, the e2e admin user already `is_platform_admin = TRUE`, and the entra identity link already present.

```
BEFORE (as superuser): accounts=5  users=1  entra_external_identities=1

$ psql -h /tmp/pg106695-sock -p 5599 -U flyway_sim -d rlsfix_run4 -1 -v ON_ERROR_STOP=1 -f
V152__harden_ci_and_e2e_seed_rls_safety.sql
NOTICE:  V152 Block A: current_user=flyway_sim owns accounts — FORCE RLS relaxed for this
transaction.
NOTICE:  V152 Block A (MC #106695): CI tenant chart-of-accounts repair complete for org
2e852173-170e-5f18-adf9-253ce4922444.
DO
NOTICE:  V152 Block B: current_user=flyway_sim owns users/entra_external_identities — FORCE
RLS relaxed for this transaction (tables: {users,entra_external_identities}).
NOTICE:  V152 Block B (MC #106695): 0 users row(s) promoted to is_platform_admin=TRUE;
entra_external_identities link ensured for bilko-e2e-admin@bilkociam.onmicrosoft.com.
DO
INSERT 0 1
$ echo exit=$?
exit=0

AFTER (as superuser): accounts=5  users=1  entra_external_identities=1   -- identical to
BEFORE
relforcerowsecurity: accounts=t entra_external_identities=t users=t     -- restored
```

`0 users row(s) promoted` (correctly — the seeded row was already `is_platform_admin=TRUE`, so the `IS DISTINCT FROM TRUE` guard excludes it), no unique-constraint errors on either the `ON CONFLICT` accounts INSERT or the `NOT EXISTS`-guarded entra INSERT, exit 0, row counts identical before and after. Idempotency confirmed on the exact already-applied shape.

Summary

Run	Role	Capability	Expected	Observed
1	admin_sim	BYPASSRLS	direct writes, FORCE untouched	5 accounts, user promoted, 1 entra row, FORCE=t throughout — MATCH

Run	Role	Capability	Expected	Observed
2	flyway_sim	owner only	NO FORCE → write → restore FORCE, isolation intact after	5 accounts, user promoted, 1 entra row, FORCE=t after + isolation re-verified (0 visible rows without org context) — MATCH
3	stranger_sim	neither	RAISE EXCEPTION, full rollback, nonzero exit	exit=3, 0 rows in accounts and schema_version — MATCH
4	flyway_sim	owner, already-applied data	no-op, identical counts, exit 0	5/1/1 before and after, exit=0 — MATCH

All four required runs pass. D1 additionally **resolves** the audit's open question: V147 is confirmed applied live (`installed_on 2026-07-24 18:14:51.597826`), so both V144 and V147 are correctly left untouched and only the new V152 file changes.

Correction from independent peer verification (received during this build)

An independent peer verifier reviewing the underlying `~/system/evidence/106342/audit-2026-08-02.md` audit re-enumerated all 149 migrations (cross-checked against when each table actually got FORCE RLS, to exclude legitimate pre-RLS cases) and found **at least 7 additional unguarded migrations** the audit's "9-row table, complete" framing missed: V38 (INSERT `users`), V65 (UPDATE `invoices`), V67 (bare `UPDATE users SET role='viewer'` — no guard code at all), V92, V98, V107, V135 — all `flyway_schema_history.success = t`. This was reported to me by the team lead mid-build, not independently re-verified here; scope for this task stays fixed to V144/V147 per the original gate approval — none of those 7 are touched by V152 or by this evidence file's D1/D3 work.

The same message also refined the D1 finding on V147: the **live `is_platform_admin=true` row on the seeded e2e admin user proves only that V147's INSERT branch (new-user path, which V147 itself guards with `SET LOCAL app.current_org_id`) executed correctly.** The UPDATE/promote branch — the one with the comment-only guard, and the one V152 Block A repairs — was **never actually exercised** on `bilko-demo-pg` (the user row did not pre-exist when V147 ran, so the `IF v_existing IS NOT NULL` branch never fired). This risk is therefore real but **not yet materialized as a data defect** — Block B's fix is preventive/forward-looking for that branch, not a repair of already-corrupted data, which matches how it is written (idempotent guard, `IS DISTINCT FROM TRUE`) but is worth stating plainly for the reviewer rather than leaving implicit.

REUSE — primjena obrasca na sljede? u migraciju

V152's two blocks are a **directly copyable pattern**, not a one-off simplified for these two tables. This section exists specifically for whoever picks up V38/V65/V67/V92/V98/V107/V135 (MC #106719) so they copy the shape instead of reinventing it — divergence here is exactly the error class this whole effort exists to prevent.

1. Minimal copy-paste skeleton (single FORCE-RLS table)

This is Block A stripped to its structural skeleton. Everything marked `-- CHANGE:` is the only thing that should differ between migrations; everything else — branch order, the post-restore assert, the exception wording style — should not:

```
DO $$
DECLARE
    tbl_owner      name;
    was_forced     boolean;
    can_bypass_rls boolean;
    relaxed_force  boolean := false;
BEGIN
    SELECT pg_get_userbyid(c.relowner), c.relforcerowsecurity
      INTO tbl_owner, was_forced
      FROM pg_class c JOIN pg_namespace n ON n.oid = c.relnamespace
     WHERE n.nspname = 'public' AND c.relname = 'TARGET_TABLE';        -- CHANGE: table name

    SELECT rolsuper OR rolbypassrls
      INTO can_bypass_rls FROM pg_roles WHERE rolname = current_user;

    IF can_bypass_rls THEN
        RAISE NOTICE 'V<N> Block: current_user=% bypasses RLS — writing directly.',
current_user;
    ELSIF tbl_owner = current_user THEN
        IF was_forced THEN
            EXECUTE 'ALTER TABLE public.TARGET_TABLE NO FORCE ROW LEVEL SECURITY';  -- CHANGE:
table name
            relaxed_force := true;
```

```
        END IF;
    ELSE
        RAISE EXCEPTION
            'V<N> (MC #<TASK>) refuses to run silently: current_user=% is neither superuser
nor '
            'BYPASSRLS, and does not own public.TARGET_TABLE (owner=%). ...',
-- CHANGE: table name + task no.
            current_user, tbl_owner;
    END IF;

    -- CHANGE: the guarded DML — copy the ORIGINAL unguarded migration's INSERT/UPDATE payload
    -- byte-identical from `git show V<orig>:.../V<orig>__*.sql`, not from V152. Do not invent
a
    -- new payload; this block re-asserts what the original migration already claims to do.
    -- <original DML here>

    IF relaxed_force THEN
        EXECUTE 'ALTER TABLE public.TARGET_TABLE FORCE ROW LEVEL SECURITY';   -- CHANGE: table
name

        SELECT c.relforcerowsecurity INTO relaxed_force
          FROM pg_class c JOIN pg_namespace n ON n.oid = c.relnamespace
         WHERE n.nspname = 'public' AND c.relname = 'TARGET_TABLE';    -- CHANGE: table name

        IF NOT relaxed_force THEN
            RAISE EXCEPTION 'V<N> post-condition failed: public.TARGET_TABLE did not get FORCE
'
                'ROW LEVEL SECURITY back. Aborting so the whole migration rolls back.';
        END IF;
    END IF;
END $$;
```

For a table already covered by this file's fixture (`accounts`, `users`, `entra_external_identities` — relevant if a later migration in the 7 touches one of these same tables again), the RLS policy/FORCE setup in `fixture-106695.sql` can be copied as-is; for any other table, its `ENABLE`/`FORCE ROW LEVEL SECURITY` + policy predicate must be read from the migration that actually created it (same method used here: read V17/V30/V66) and copied verbatim into the new fixture — do not approximate the predicate, per the V149 postmortem below.

2. What MUST be adapted per table — per-table ownership verification, not one check

`rolsuper`/`rolbypassrls` are **role attributes** — checked once, globally, regardless of how many tables a block writes to. **Table ownership is a per-table fact** and must be checked once per FORCE-RLS table the block touches, in a loop (Block B's shape), never assumed from checking just one table. Concretely, in this build `users` and `entra_external_identities` are independently FORCE-RLS (V30, V66) with independent policies, and I verified both separately rather than checking `users` and assuming `entra_external_identities` was the same — today they share an owner (`bilko_admin`, confirmed live in D1), so the distinction doesn't currently change any outcome, but nothing in Postgres or in Bilko's migration history guarantees that stays true (a future migration could `ALTER TABLE ... OWNER TO` one table without the other, or OCD-5's `flyway_ci` role could be granted ownership unevenly during a phased rollout). If any of the 7 target migrations write to more than one FORCE-RLS table, use Block B's per-table loop, not a single check against the first table.

3. Traps — hit or deliberately avoided in this build, so the next author doesn't repeat them

- **FORCE relax → DML → FORCE restore → assert must all be in the same transaction/DO block.** Flyway already wraps the whole migration file in one transaction (matches V149's own note), so a mid-block failure rolls everything back including the relaxed FORCE state — but if this pattern is ever run manually outside Flyway (e.g. via bare `psql -f`, not `psql -1 -f`), a crash between the relax and the restore statements would commit with FORCE left off. Always test with `psql -1 -v ON_ERROR_STOP=1 -f ...` (single transaction), exactly like Flyway does — Run 3 below only proves rollback-on-failure because of `-1`.
- **Assert the restore against the state observed at entry (`was_forced`), never hardcode `true`.** An environment that legitimately has FORCE off should be left exactly as found, not force-fail the deploy for the wrong reason. This is inherited caution from V149's own postmortem (`~/system/evidence/106302/fix-verdict.md`) rather than something I personally got wrong this build, but it is the single easiest thing to regress when copy-pasting under time pressure.
- **A trap I did personally hit in this build:** Run 1 first failed with `permission denied for table account_types`, exit 3 — the fixture had granted the simulated roles `SELECT` only on `account_types`, not `INSERT`. The migration's own guard correctly rolled back and reported clearly (not a silent failure — the fail-loud behavior working as intended, just tripped by a fixture gap, not a migration bug), but it costs a full fixture-reload cycle each time a GRANT is missing. **Grant every privilege the guarded DML actually needs, on every simulated role, before the first proof run** — read the DML first, list every table/column it touches, grant all of them up front, rather than discovering gaps one exit-code-3 at a time.

- **A trap avoided by design:** seeding "before" state as the role under test (e.g. `flyway_sim`) rather than as the fixture superuser fails with "new row violates row-level security policy" — because that role has neither BYPASSRLS nor an `app.current_org_id` context, exactly like production. This is a useful confirmation the fixture is faithful (see Run 2's transcript, which deliberately shows this failure once on purpose), but always seed "before" state as `postgres`/superuser in the actual proof runs — that mirrors how the real data got there in production (through whatever role/context had access at the time), not through the locked-down role you are now testing against.
- **Multiple owners across the tables one block touches:** not the case today for any table this migration writes to (D1 re-confirmed `bilko_admin` owns `accounts`, `users`, and `entra_external_identities`), but the per-table loop exists precisely because it *could* become true later. A role that owns table A but not table B under the loop shape will `RAISE EXCEPTION` naming table B specifically — it will not silently "half-succeed" on table A alone. Do not "simplify" the loop into a single check against the first table in the array; that would silently reintroduce the exact per-table blind spot the loop exists to close.

4. Which of the 4 D3 runs must repeat per new migration vs. which are one-time

Must repeat, in full, for every one of V38/V65/V67/V92/V98/V107/V135 individually — each guarded table/DML pair is a materially different code path and none of these four runs can be skipped or assumed from V152's result:

- **Run 1 (BYPASSRLS)** — cheap to run, but still required: a copy-paste error in the bypass branch is exactly as capable of a silent no-op as an unguarded migration would be.
- **Run 2 (owner)** — the most important repeat. This is the real production shape (`bilko_admin` today), and the FORCE-relax/restore/per-table-loop logic is the part most likely to carry a new bug when adapted to a different table shape.
- **Run 3 (fail-loud)** — must repeat; this is the entire reason the hardening effort exists, and the branch most likely to be silently broken by a careless copy-paste (e.g., the `RAISE EXCEPTION` accidentally left after the DML instead of before it — that single ordering mistake would turn a fail-loud migration back into a silent one, undetectable by Runs 1/2/4 alone).
- **Run 4 (idempotency)** — must repeat, seeded with *that* migration's own already-applied-data shape (not V144/V147's rows) — the `ON CONFLICT`/`WHERE NOT EXISTS`/`IS DISTINCT FROM` guard predicate is specific to each table's DML and has to be proven idempotent on its own terms.

Can be reused / is effectively one-time across the whole 7-migration pass:

- **Role setup** (`flyway_sim`, `admin_sim`, `stranger_sim`, `bilko_app`) — Postgres roles are cluster-wide, so the same three roles can be reused against a fresh database per migration without re-creating them; only the per-migration `CREATE DATABASE` + fixture-table load needs to be fresh.

- **Fixture table/policy definitions for any of the three tables already covered here** (`accounts`, `users`, `entra_external_identities`) — if one of the 7 target migrations touches one of these same tables again, its RLS setup does not need to be re-derived from V17/V30/V66; it can be copied from `fixture-106695.sql` directly. In practice most of the 7 likely touch *different* tables (V65 → `invoices` per the audit's one-line description, others unconfirmed without reading each file), so expect to add new table/policy definitions to the fixture per migration rather than treating the whole fixture file as reusable as-is.

What was deliberately NOT done here

No shared SQL function/library was extracted (e.g. a `bilko_auth.assert_rls_capability(text[])` helper) — V148, V149, and V152 all inline the check independently. Extracting one would reduce duplication across the next 7-file pass, at the cost of adding a new shared object every future migration then depends on (and a new thing that itself needs RLS/ownership reasoning about who can call it). That tradeoff belongs to whoever picks up MC #106719, not to this task — deciding it here would expand V152's scope beyond V144/V147, which the gate did not approve.

Scope discipline

- New worktree only (`.claude/worktrees/bm-106695`), no changes to any other worktree.
- No `git push`, no PR opened, no `mc.js done`/`ready` called by this agent.
- Azure session (isolated `AZURE_CONFIG_DIR`) logged out immediately after the two D1 queries; the fetched DB password was scratchpad-local, `chmod 600`, and deleted after use.
- Scratch PostgreSQL cluster stopped (`pg_ctl ... stop -m fast`) after all four runs completed; no shared/demo database was touched by any D3 query.
- **Not marked done.** Per the team-lead's dispatch, this task requires an independent peer (RLS scope) before ready/done — handing off for that review now.

Open items for the reviewer / orchestrator

1. **V152 is provisional.** Re-run the same collision-avoidance enumeration (worktrees + all `azdo` remote branches) immediately before merge — `git fetch azdo` could not be tested as fixed in this environment, so the possibility of a fresh authenticated fetch surfacing a different tip is real and unverified from here.
2. **PostgreSQL 18.3 was used for D3, not PostgreSQL 15** (not installed on this machine). RLS/ FORCE RLS/BYPASSRLS semantics are unchanged across that version range, but a reviewer with PG15 available may wish to re-run the same four scenarios for full spec conformance.
3. The audit's §4 proposed CI lint and companion runtime guard (post-`Flyway_Migrate` BYPASSRLS assertion) remain out of scope here, per the forged spec's OPEN CEO

DECISIONS #3 — not built, not implied to be built by this evidence.

Revision #2
Created 2026-08-02 16:16:40 UTC by John
Updated 2026-08-10 07:38:01 UTC by John