```sql
-- PostGIS
CREATE EXTENSION IF NOT EXISTS postgis;

-- Locations for ALL users (1:1 with users)
CREATE TABLE IF NOT EXISTS virginia_dev_saayam_rdbms.locations (
  user_id VARCHAR(255) NOT NULL PRIMARY KEY,  -- natural key
  prev_loc geography(Point, 4326),
  curr_loc geography(Point, 4326),
  updated_at timestamptz NOT NULL DEFAULT now(),
  FOREIGN KEY (user_id)
    REFERENCES virginia_dev_saayam_rdbms.users (user_id)
    ON DELETE CASCADE
);
-- Spatial indexes
CREATE INDEX IF NOT EXISTS idx_locations_curr_gist
    ON virginia_dev_saayam_rdbms.locations USING GIST (curr_loc);
CREATE INDEX IF NOT EXISTS idx_locations_prev_gist
    ON virginia_dev_saayam_rdbms.locations USING GIST (prev_loc);

-- Trigger function to shift old curr_loc -> prev_loc on update
CREATE OR REPLACE FUNCTION virginia_dev_saayam_rdbms.fn_shift_prev_loc()
RETURNS trigger
LANGUAGE plpgsql
AS $$
BEGIN
    IF TG_OP = 'UPDATE' AND NEW.curr_loc IS DISTINCT FROM OLD.curr_loc THEN
        NEW.prev_loc := OLD.curr_loc;
        NEW.updated_at := now();
    END IF;
    RETURN NEW;
END;
$$;

DROP TRIGGER IF EXISTS trg_shift_prev_loc ON virginia_dev_saayam_rdbms.locations;
CREATE TRIGGER trg_shift_prev_loc
BEFORE UPDATE ON virginia_dev_saayam_rdbms.locations
FOR EACH ROW
EXECUTE FUNCTION virginia_dev_saayam_rdbms.fn_shift_prev_loc();

--Trigger function for INSERT ... (user_id, curr_loc); if the row exists, we update instead (and also do the prev→curr
shift). We can also add a per-user advisory lock to prevent rare race conditions with concurrent inserts.

CREATE OR REPLACE FUNCTION virginia_dev_saayam_rdbms.fn_locations_insert_as_upsert()
RETURNS trigger
LANGUAGE plpgsql AS $$
DECLARE
  k BIGINT;
BEGIN
  -- Serialize concurrent inserts for the same user
  k := hashtextextended(NEW.user_id, 0);
  PERFORM pg_advisory_xact_lock(k);

  -- If row exists, UPDATE only curr_loc (shift handled by BEFORE UPDATE trigger)
  UPDATE virginia_dev_saayam_rdbms.locations l
     SET curr_loc = NEW.curr_loc
   WHERE l.user_id = NEW.user_id;

  IF FOUND THEN
    RETURN NULL;      -- suppress the original INSERT
  END IF;

  -- No row yet → proceed with INSERT
  RETURN NEW;
END;
$$;

DROP TRIGGER IF EXISTS trg_locations_insert_as_upsert ON virginia_dev_saayam_rdbms.locations;
CREATE TRIGGER trg_locations_insert_as_upsert
BEFORE INSERT ON virginia_dev_saayam_rdbms.locations
```

```
FOR EACH ROW
EXECUTE FUNCTION virginia_dev_saayam_rdbms.fn_locations_insert_as_upsert();
```

-- To ensure each volunteer is unique by user_id as we don’t plan to allow multiple volunteer rows per user

```
DO $$
BEGIN
    IF NOT EXISTS (
        SELECT 1 FROM pg_constraint
        WHERE conname = 'volunteer_details_user_uk'
          AND conrelid = 'virginia_dev_saayam_rdbms.volunteer_details'::regclass
    ) THEN
        ALTER TABLE virginia_dev_saayam_rdbms.volunteer_details
            ADD CONSTRAINT volunteer_details_user_uk UNIQUE (user_id);
    END IF;
END;
$$;
```