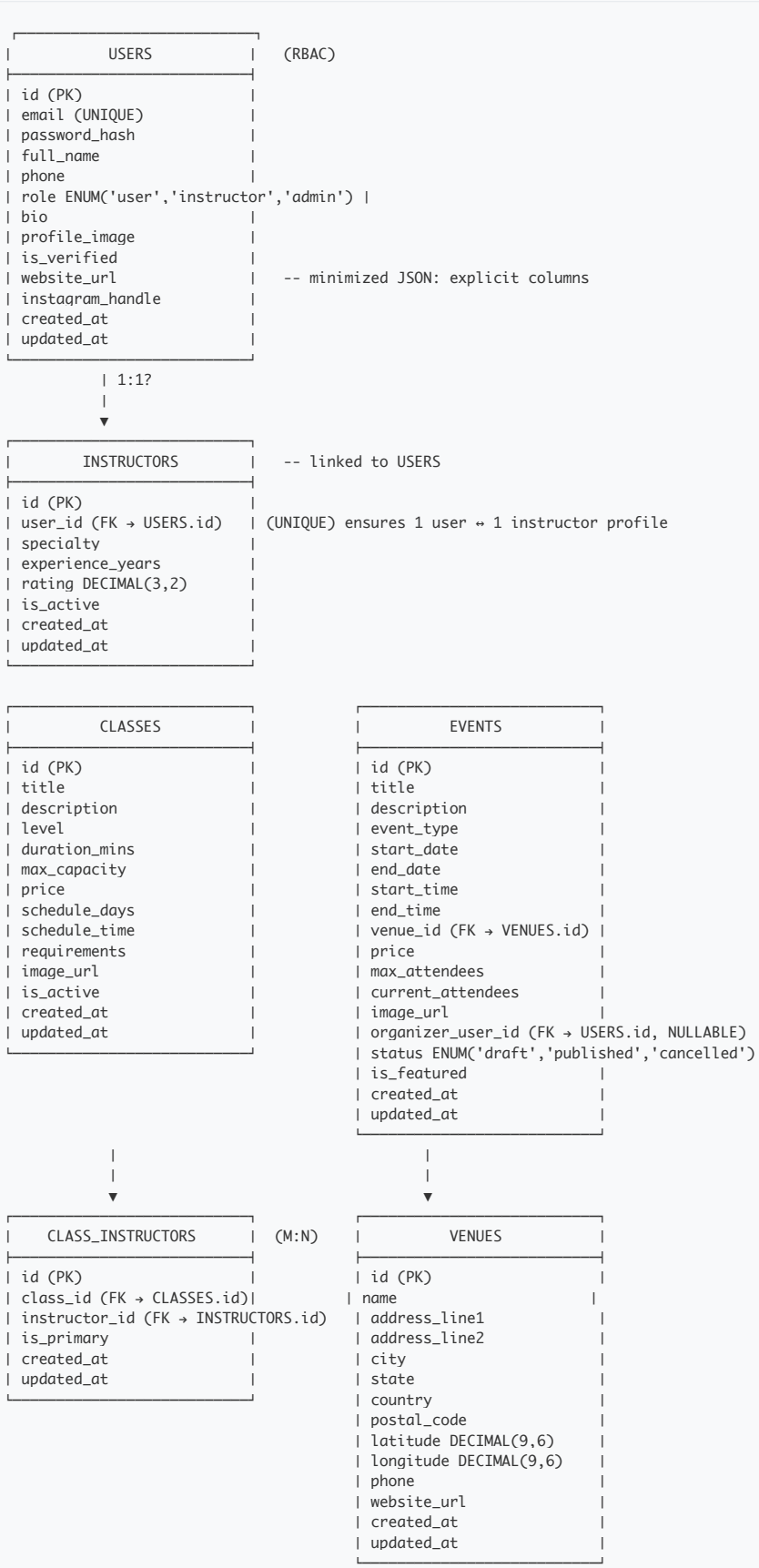
Dance Website - ERD v2 (RBAC, Venues, Styles, Unified Bookings, Transactions)

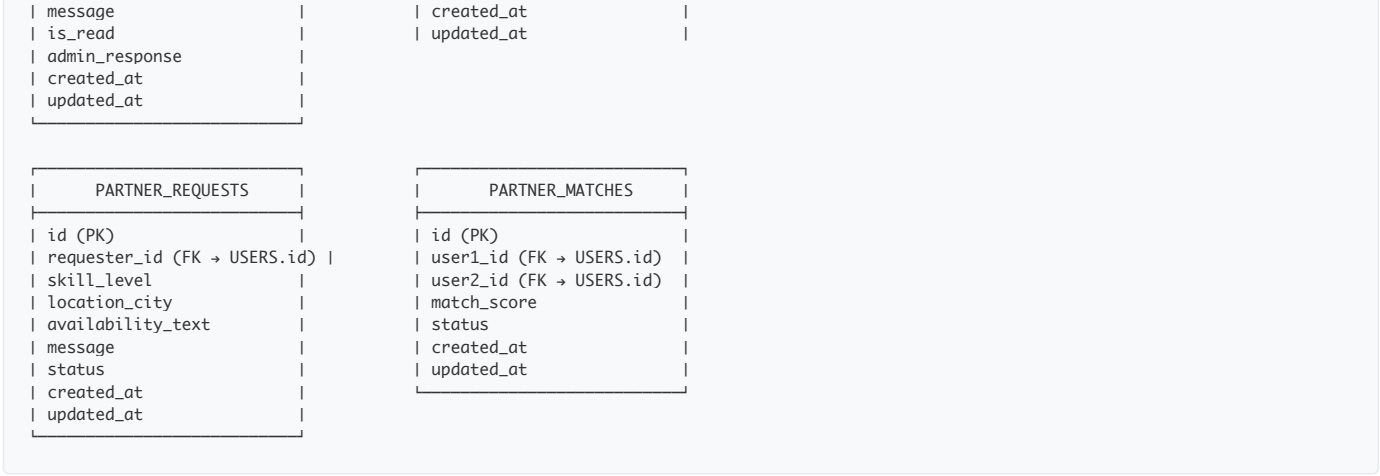
Overview

This ERD updates the prior schema to: - Merge ADMINS into USERS via RBAC (no ADMINS table) - Link INSTRUCTORS to USERS for unified accounts - Add VENUES and reference from EVENTS via venue_id - Normalize dance styles via mapping tables (EVENT_STYLES, CLASS_STYLES, USER_STYLES) - Unify BOOKINGS across classes/events with duplicate-prevention - Add TRANSACTIONS for payment/refund logging - Remove duplicated forum tables and add ON DELETE/UPDATE rules - Minimize JSON fields for MVP; prefer normalized columns - Add indexes/uniques for FKs, timestamps, start_time, and style mappings

Core Entities and Relationships


USERS
(RBAC)
id (PK)
email (UNIQUE)
password_hash
full_name
phone
role ENUM('user','instructor','admin')
bio
profile_image
is_verified
website_url
instagram_handle
created_at
updated_at
-- minimized JSON: explicit columns
1:1?
INSTRUCTORS
-- linked to USERS
id (PK)
user_id (FK → USERS.id)
(UNIQUE) ensures 1 user ↔ 1 instructor profile
specialty
experience_years
rating DECIMAL(3,2)
is_active
created_at
updated_at
CLASSES
id (PK)
title
description
level
duration_mins
max_capacity
price
schedule_days
schedule_time
requirements
image_url
is_active
created_at
updated_at
EVENTS
id (PK)
title
description
event_type
start_date
end_date
start_time
end_time
venue_id (FK → VENUES.id)
price
max_attendees
current_attendees
image_url
organizer_user_id (FK → USERS.id, NULLABLE)
status ENUM('draft','published','cancelled')
is_featured
created_at
updated_at
CLASS_INSTRUCTORS
(M:N)
id (PK)
class_id (FK → CLASSES.id)
instructor_id (FK → INSTRUCTORS.id)
is_primary
created_at
updated_at
VENUES
id (PK)
name
address_line1
address_line2
city
state
country
postal_code
latitude DECIMAL(9,6)
longitude DECIMAL(9,6)
phone
website_url
created_at
updated_at
BOOKINGS
-- unified for class/event
id (PK)
user_id (FK → USERS.id)
class_id (FK → CLASSES.id, NULLABLE)
event_id (FK → EVENTS.id, NULLABLE)
booking_date
status ENUM('pending','confirmed','cancelled','completed','refunded')
amount_paid
payment_method
notes
created_at
updated_at

```
└──────────────────────────────┘
Constraints:
- CHECK ((class_id IS NOT NULL) <> (event_id IS NOT NULL))  -- exactly one set
- UNIQUE (user_id, class_id) WHERE class_id IS NOT NULL     -- prevent duplicates
- UNIQUE (user_id, event_id) WHERE event_id IS NOT NULL


┌──────────────────────────────┐
|       TRANSACTIONS           |   -- payment/refund logs
├──────────────────────────────┤
| id (PK)                      |
| booking_id (FK → BOOKINGS.id, NULLABLE) |
| user_id (FK → USERS.id)      |
| provider ENUM('stripe','paypal','other') |
| provider_payment_id          |
| provider_refund_id           | (NULLABLE)
| type ENUM('payment','refund','adjustment') |
| status ENUM('created','succeeded','failed','refunded','cancelled') |
| amount                       |
| currency                     |
| payload TEXT                 |  -- raw provider payload (minimize JSON usage)
| created_at                   |
| updated_at                   |
└──────────────────────────────┘


┌──────────────────────────────┐            ┌──────────────────────────────┐
|        DANCE_STYLES          |            |       USER_STYLES            |
├──────────────────────────────┤            ├──────────────────────────────┤
| id (PK)                      |            | id (PK)                      |
| name (UNIQUE)                |            | user_id (FK → USERS.id)      |
| category                     |            | style_id (FK → DANCE_STYLES.id) |
| is_active                    |            | proficiency ENUM('beginner','intermediate','advanced') |
| created_at                   |            | created_at                   |
| updated_at                   |            | updated_at                   |
└──────────────────────────────┘            └──────────────────────────────┘

┌──────────────────────────────┐            ┌──────────────────────────────┐
|       CLASS_STYLES           |            |       EVENT_STYLES           |
├──────────────────────────────┤            ├──────────────────────────────┤
| id (PK)                      |            | id (PK)                      |
| class_id (FK → CLASSES.id)|          | event_id (FK → EVENTS.id) |
| style_id (FK → DANCE_STYLES.id) |     | style_id (FK → DANCE_STYLES.id) |
| created_at                   |            | created_at                   |
| updated_at                   |            | updated_at                   |
└──────────────────────────────┘            └──────────────────────────────┘
Constraints:
- UNIQUE (class_id, style_id)
- UNIQUE (event_id, style_id)
- UNIQUE (user_id, style_id) in USER_STYLES


┌──────────────────────────────┐            ┌──────────────────────────────┐
|        FORUM_POSTS           |            |        FORUM_REPLIES         |
├──────────────────────────────┤            ├──────────────────────────────┤
| id (PK)                      |            | id (PK)                      |
| user_id (FK → USERS.id)      |            | post_id (FK → FORUM_POSTS.id)
| category                     |            | user_id (FK → USERS.id)      |
| title                        |            | parent_id (FK → FORUM_REPLIES.id, NULLABLE)
| content                      |            | content                      |
| views_count                  |            | likes_count                  |
| likes_count                  |            | is_solution                  |
| replies_count                |            | created_at                   |
| is_pinned                    |            | updated_at                   |
| is_locked                    |            └──────────────────────────────┘
| created_at                   |
| updated_at                   |
└──────────────────────────────┘

┌──────────────────────────────┐            ┌──────────────────────────────┐
|       NOTIFICATIONS          |            |         AUDIT_LOGS           |
├──────────────────────────────┤            ├──────────────────────────────┤
| id (PK)                      |            | id (PK)                      |
| user_id (FK → USERS.id)      |            | user_id (FK → USERS.id)      |  -- replaces admin_id
| type                         |            | action                       |
| title                        |            | table_name                   |
| message                      |            | record_id                    |
| is_read                      |            | old_values TEXT              |
| priority                     |            | new_values TEXT              |
| action_url                   |            | ip_address                   |
| created_at                   |            | user_agent                   |
| updated_at                   |            | created_at                   |
└──────────────────────────────┘            | updated_at                   |
                                            └──────────────────────────────┘

┌──────────────────────────────┐            ┌──────────────────────────────┐
|      CONTACT_MESSAGES        |            |        TESTIMONIALS          |
├──────────────────────────────┤            ├──────────────────────────────┤
| id (PK)                      |            | id (PK)                      |
| name                         |            | user_id (FK → USERS.id)      |
| email                        |            | rating                       |
| phone                        |            | message                      |
| subject                      |            | is_featured                  |
```



Key Changes vs Previous ERD

1. Removed ADMINS table. USERS has role and all admin actions reference USERS.
2. INSTRUCTORS has user_id (unique), removing duplicate identity fields from instructors; social links are simplified to explicit columns on USERS.
3. Added VENUES and referenced by EVENTS via venue_id. Removed free-text location/address from EVENTS (moved to VENUES).
4. Normalized dance styles with mapping tables: USER_STYLES, CLASS_STYLES, EVENT_STYLES.
5. Unified BOOKINGS with constraints to prevent duplicate bookings per user per class/event.
6. Added TRANSACTIONS to log payments/refunds with provider IDs and status.
7. Forum tables are singular (FORUM_POSTS, FORUM_REPLIES). All FKs include ON DELETE/UPDATE rules below.
8. Minimized JSON fields: replaced preferences/tags/social_links with normalized columns or TEXT where appropriate.
9. Added indexes and unique constraints for performance and data integrity.

Foreign Keys and Referential Actions

Unless otherwise noted, FKs use: - ON DELETE CASCADE where child records have no meaning without parent (e.g., CLASS_INSTRUCTORS, STYLES mappings, BOOKINGS, TRANSACTIONS linked to BOOKINGS) - ON UPDATE CASCADE for id changes (rare but safe) - ON DELETE SET NULL where history should be retained without a parent (e.g., TRANSACTIONS.booking_id, EVENTS.organizer_user_id)

Examples: - INSTRUCTORS.user_id → USERS.id ON DELETE CASCADE ON UPDATE CASCADE - CLASS_INSTRUCTORS.class_id → CLASSES.id ON DELETE CASCADE ON UPDATE CASCADE - CLASS_INSTRUCTORS.instructor_id → INSTRUCTORS.id ON DELETE CASCADE ON UPDATE CASCADE - EVENTS.venue_id → VENUES.id ON DELETE RESTRICT ON UPDATE CASCADE - BOOKINGS.user_id → USERS.id ON DELETE CASCADE ON UPDATE CASCADE - BOOKINGS.class_id → CLASSES.id ON DELETE CASCADE ON UPDATE CASCADE - BOOKINGS.event_id → EVENTS.id ON DELETE CASCADE ON UPDATE CASCADE - TRANSACTIONS.booking_id → BOOKINGS.id ON DELETE SET NULL ON UPDATE CASCADE - TRANSACTIONS.user_id → USERS.id ON DELETE CASCADE ON UPDATE CASCADE - USER_STYLES.user_id → USERS.id ON DELETE CASCADE; style_id → DANCE_STYLES.id ON DELETE RESTRICT - CLASS_STYLES.class_id → CLASSES.id ON DELETE CASCADE; style_id → DANCE_STYLES.id ON DELETE RESTRICT - EVENT_STYLES.event_id → EVENTS.id ON DELETE CASCADE; style_id → DANCE_STYLES.id ON DELETE RESTRICT - FORUM_POSTS.user_id → USERS.id ON DELETE SET NULL (optional, if you want to retain posts) or CASCADE if you prefer deletion - FORUM_REPLIES.user_id → USERS.id ON DELETE SET NULL; post_id → FORUM_POSTS.id ON DELETE CASCADE; parent_id → FORUM_REPLIES.id ON DELETE CASCADE

Indexes and Constraints

- USERS(email) UNIQUE
- USERS(role), USERS(created_at) INDEX
- INSTRUCTORS(user_id) UNIQUE, INDEX
- CLASSES(created_at), CLASSES(is_active) INDEX
- EVENTS(venue_id) INDEX, EVENTS(start_date, start_time) INDEX, EVENTS(status) INDEX
- VENUES(name), VENUES(city), VENUES(country) INDEX
- BOOKINGS(user_id) INDEX, BOOKINGS(created_at) INDEX, BOOKINGS(status) INDEX
- Unique constraints:
- USER_STYLES(user_id, style_id)
- CLASS_STYLES(class_id, style_id)
- EVENT_STYLES(event_id, style_id)
- BOOKINGS(user_id, class_id) where class_id not null
- BOOKINGS(user_id, event_id) where event_id not null
- TRANSACTIONS(provider, provider_payment_id) UNIQUE (nullable provider_refund_id can be separate unique when present)
- FORUM_POSTS(created_at), FORUM_REPLIES(created_at) INDEX

Notes on JSON Minimization

- Replaced JSON social_links with explicit columns on USERS (website_url, instagram_handle). Add more as needed.
- Replaced EVENT.tags JSON with normalized styles mapping; additional tagging can be added later as TAGS/TAG_MAPPINGS if needed.
- AUDIT_LOGS uses TEXT for old/new values to avoid complex JSON; can move to JSON later as needed.

Suggested SQL Migration Outline (RDBMS-agnostic, adjust types)

- Add USERS.role, website_url, instagram_handle
- Drop ADMINS; migrate admin rows into USERS with role='admin'
- INSTRUCTORS: add user_id UNIQUE NOT NULL; backfill from existing email/user mapping; drop duplicate identity fields if present
- Create VENUES; add EVENTS.venue_id; backfill from existing location/address; drop EVENTS.location/address
- Create DANCE_STYLES, USER_STYLES, CLASS_STYLES, EVENT_STYLES; backfill from prior dance_style columns then drop those columns
- Create unified BOOKINGS; migrate CLASS_BOOKINGS and EVENT_BOOKINGS into BOOKINGS; drop old tables
- Create TRANSACTIONS; link to BOOKINGS where applicable
- Ensure ON DELETE/UPDATE rules per above
- Create/adjust indexes and uniques per above

Open Decisions

- FORUM_POSTS/REPLIES ON DELETE behavior: choose SET NULL vs CASCADE to align with content retention policy.
- EVENTS.organizer_user_id optional; you may also want ORGANIZERS table linked to USERS similar to INSTRUCTORS.
- If you need additional user preferences later, consider a USER_PREFERENCES table rather than JSON.