Here’s a concise, normalized ERD design for ThesisBoard that aligns with your flows, roles, and Auth0 integration. It separates Pre‑Thesis and Thesis, captures semester snapshot data, enforces teacher slots/capacity, and supports announcements/notifications.

Core design principles

* Auth0 is authoritative for authN/authZ; DB stores shadow users keyed by auth0\_user\_id and derived user\_type for convenience.
* Semester-driven data: student GPA/credits tracked per semester; “current” is the latest active semester.
* Capacity and eligibility enforced via constraints and service logic.
* Grading is evaluator-based (per evaluator/role) with an aggregate per project.
* Everything auditable with created\_by/updated\_by timestamps; soft-delete where appropriate.

Entities (grouped)

1. Identity and People

* users
  + id (PK), auth0\_user\_id (unique), user\_type enum [admin, moderator, teacher, student]
  + username (unique; for students = student\_id), email, full\_name, status, last\_login\_at
  + created\_at, updated\_at
* students
  + id (PK, FK users.id unique), student\_code (unique; student ID), cohort, program, class\_name
  + gpa\_current, credits\_current
  + phone
* teachers
  + id (PK, FK users.id unique), academic\_title, department, bio
* student\_semesters
  + id (PK), student\_id (FK students), semester\_id (FK semesters), gpa, credits\_earned, status
  + unique(student\_id, semester\_id)

1. Academic Structure

* semesters
  + id (PK), code (e.g., 2024-2025-Fall, unique), year, term enum [Spring, Summer, Fall], start\_date, end\_date, is\_active
* teacher\_semester\_capacity
  + id (PK), teacher\_id (FK teachers), semester\_id (FK semesters)
  + prethesis\_topic\_slots (number of concurrent topics teacher may open)
  + prethesis\_student\_capacity (sum of accepted students across topics)
  + thesis\_supervision\_capacity (active theses supervised)
  + unique(teacher\_id, semester\_id)

1. Pre‑Thesis

* prethesis\_topics
  + id (PK), teacher\_id (FK teachers), semester\_id (FK semesters)
  + title, description, capacity, prerequisites, skills, tags\_json, status enum [draft, open, closed, archived]
  + unique(teacher\_id, semester\_id, title)
* prethesis\_applications
  + id (PK), topic\_id (FK prethesis\_topics), student\_id (FK students), semester\_id (FK semesters)
  + status enum [pending, accepted, rejected, withdrawn], priority smallint null
  + submitted\_at, decided\_by (FK users), decided\_at, reason
  + unique(student\_id, semester\_id) with partial index for active apps
* prethesis\_enrollments
  + id (PK), topic\_id (FK prethesis\_topics), student\_id (FK students), supervisor\_id (FK teachers), semester\_id (FK semesters)
  + status enum [active, completed, cancelled], approved\_by (FK users), approved\_at
  + final\_grade numeric(5,2) null
  + unique(student\_id, semester\_id) for accepted records
* prethesis\_grades
  + id (PK), enrollment\_id (FK prethesis\_enrollments), graded\_by (FK teachers), score numeric(5,2), rubric\_json, graded\_at
  + unique(enrollment\_id, graded\_by)

1. Thesis

* thesis\_registrations
  + id (PK), student\_id (FK students), semester\_id (FK semesters)
  + submitted\_by\_teacher\_id (FK teachers), title\_proposed, abstract\_proposed
  + status enum [pending, approved, rejected], decided\_by (FK users), decided\_at, notes
  + unique(student\_id, semester\_id) for active requests
* theses
  + id (PK), semester\_id (FK semesters), created\_from\_registration\_id (FK thesis\_registrations)
  + title, abstract, status enum [active, defense\_scheduled, defended, completed, cancelled]
  + repository\_url, final\_grade numeric(5,2), defense\_date, defense\_room
  + unique(title, semester\_id) optional
* thesis\_students
  + thesis\_id (FK theses), student\_id (FK students)
  + PK(thesis\_id, student\_id) // allows team if needed; in CSE typically size 1
* thesis\_members
  + id (PK), thesis\_id (FK theses), teacher\_id (FK teachers)
  + role enum [Supervisor, Reviewer, Committee], is\_chair boolean default false
  + unique(thesis\_id, teacher\_id, role)
* thesis\_evaluations
  + id (PK), thesis\_id (FK theses), teacher\_id (FK teachers), role enum [Supervisor, Reviewer, Committee]
  + score numeric(5,2), weight numeric(5,2), comments, submitted\_at
  + unique(thesis\_id, teacher\_id, role)
* grading\_schemes
  + id (PK), scope enum [prethesis, thesis], semester\_id (FK semesters), name
* grading\_scheme\_components
  + id (PK), scheme\_id (FK grading\_schemes), role enum [Supervisor, Reviewer, Committee]
  + weight numeric(5,2) // e.g., Supervisor 30, Reviewer 30, Committee 40
  + unique(scheme\_id, role)
* defense\_events
  + id (PK), semester\_id (FK semesters), schedule\_at, room, status enum [planned, ongoing, completed]
* defense\_event\_members
  + event\_id (FK defense\_events), teacher\_id (FK teachers), is\_chair boolean
  + PK(event\_id, teacher\_id)
* defense\_event\_theses
  + event\_id (FK defense\_events), thesis\_id (FK theses), order\_in\_session smallint
  + PK(event\_id, thesis\_id)

1. Deliverables and Materials

* milestones
  + id (PK), scope\_type enum [prethesis, thesis], scope\_id (FK to prethesis\_enrollments or theses)
  + title, description, due\_at, weight numeric(5,2) null
* submissions
  + id (PK), milestone\_id (FK milestones), student\_id (FK students)
  + submitted\_at, status enum [submitted, graded, resubmission\_requested], grade numeric(5,2) null, feedback
* files
  + id (PK), storage\_key, file\_name, mime\_type, size\_bytes, uploaded\_by (FK users), uploaded\_at
* submission\_files
  + submission\_id (FK submissions), file\_id (FK files)
  + PK(submission\_id, file\_id)

1. Communication

* announcements
  + id (PK), title, content, visible\_from, visible\_to, created\_by (FK users), created\_at
  + scope enum [all, role, semester, teacher, student]
* announcement\_audience\_roles
  + announcement\_id (FK announcements), role enum [admin, moderator, teacher, student]
  + PK(announcement\_id, role)
* announcement\_audience\_semesters
  + announcement\_id (FK announcements), semester\_id (FK semesters)
  + PK(announcement\_id, semester\_id)
* announcement\_reads
  + announcement\_id (FK announcements), user\_id (FK users), read\_at
  + PK(announcement\_id, user\_id)
* notifications
  + id (PK), type, entity\_type, entity\_id, message, created\_at, sender\_id (FK users null for system)
* notification\_recipients
  + id (PK), notification\_id (FK notifications), user\_id (FK users)
  + delivered\_at, read\_at, action\_url

1. Common/Audit

* Every main table: created\_at, updated\_at, created\_by (FK users), updated\_by (FK users), deleted\_at (soft delete).

Key relationships and constraints

* users 1–1 students; users 1–1 teachers.
* semesters 1–\* student\_semesters; semesters 1–\* prethesis\_topics; semesters 1–\* thesis\_registrations; semesters 1–\* theses; semesters 1–\* defense\_events.
* teachers 1–\* prethesis\_topics; teachers 1–\* thesis\_members; teachers 1–\* thesis\_evaluations.
* prethesis\_topics 1–\* prethesis\_applications; acceptance creates 1 prethesis\_enrollments per student per semester.
* Capacity rules:
  + prethesis\_topics.capacity limits accepted enrollments per topic.
  + Sum of active enrollments under a teacher in a semester ≤ teacher\_semester\_capacity.prethesis\_student\_capacity.
  + Active theses supervised by a teacher ≤ thesis\_supervision\_capacity.
* Eligibility rules:
  + A student may have at most one active prethesis\_enrollments per semester.
  + A student may have at most one active thesis.
  + GPA/credits checks use the latest student\_semesters entry of the semester in question.
* Grading:
  + Pre‑Thesis: one or more prethesis\_grades by supervisor; final stored on enrollment or computed.
  + Thesis: multiple thesis\_evaluations by role; aggregate using grading\_schemes/components per semester to compute theses.final\_grade.
* Reviewer/committee constraints:
  + Reviewer(s) and committee members added by moderator.
  + Reviewer cannot be the same teacher as a supervisor (enforce by constraint or service logic).
  + Committee has optional chair (is\_chair).

Auth0 integration notes

* Store users.auth0\_user\_id as the unique external key. Do not store passwords.
* Mirror the effective role into users.user\_type for querying/UI; treat it as cached/derived. Authoritative role checks come from Auth0 JWT claims.
* On login, upsert shadow user with Auth0 profile; keep username = student\_code for students, and provision usernames for admin/mod/teacher.

Announcements/Notifications

* Announcements are broadcast, filterable by role/semester; read receipts tracked.
* Notifications are event-driven (e.g., student applied, teacher accepted, moderator approved, defense scheduled) with per-recipient read state.

Indexes to add

* users: unique(auth0\_user\_id), unique(username), index(user\_type)
* students: unique(student\_code)
* student\_semesters: unique(student\_id, semester\_id)
* teacher\_semester\_capacity: unique(teacher\_id, semester\_id)
* prethesis\_applications: unique(student\_id, semester\_id) partial for active
* prethesis\_enrollments: unique(student\_id, semester\_id)
* thesis\_registrations: unique(student\_id, semester\_id)
* thesis\_members: unique(thesis\_id, teacher\_id, role)
* thesis\_evaluations: unique(thesis\_id, teacher\_id, role)
* announcements: index(scope, visible\_from, visible\_to)
* notification\_recipients: index(user\_id, read\_at)

Typical flows this design supports

* Moderator uploads/updates student semester data; eligibility computed from student\_semesters.
* Teacher opens Pre‑Thesis topics (bounded by teacher\_semester\_capacity); students apply; teacher accepts within topic capacity; enrollment created; supervisor grades.
* Teacher submits Thesis registration; moderator approves; thesis created; reviewers/committee assigned; defense scheduled via defense\_events; evaluators grade; scheme aggregates to final.
* Announcements visible by role/semester; notifications for all key actions with read receipts.