UACS ER → Relational Mapping (HW02)

Global choices

- **Naming:** Avoid SQL keywords. Use `UserAccount` for `User`, `SystemRes` for `System`, and `time_stamp` for the event time attribute.
- **Keys:** Surrogate `BIGINT AUTO_INCREMENT` primary keys for all supertypes.
- **ISA strategy:** Class-table inheritance with shared PK (one table per subtype; PK = FK to supertype). Deletes cascade from supertype to subtype.
- **Cardinality enforcement:** All 1:N relationships implemented via `NOT NULL` foreign keys on the N-side.
- **Referential actions:** `ON DELETE CASCADE` only on subtype links; relationship FKs use default `RESTRICT` to preserve history.
- **Domains (ENUMs):**
  - `Request.status` ∈ {`PENDING`, `APPROVED`, `DENIED`}
  - `Access_Event.outcome` ∈ {`SUCCESS`, `FAILED`}

Entities → Tables (attributes & constraints)

1.1 User → UserAccount

- **Columns**
  - `user_id BIGINT` PK `AUTO_INCREMENT`
  - `full_name VARCHAR(100) NOT NULL` (from ER attribute name)
  - `email VARCHAR(120) NOT NULL UNIQUE`

1.2 Resource → Resource

- **Columns**
  - `resource_id BIGINT` PK `AUTO_INCREMENT`
  - `name VARCHAR(120) NOT NULL`
  - `relation VARCHAR(120)` (from ER attribute *relation*)

1.3 Request → Request

- **Columns**
  - `request_id BIGINT` PK `AUTO_INCREMENT`
  - `status ENUM('PENDING','APPROVED','DENIED') NOT NULL DEFAULT 'PENDING'`
  - *Folded relationship FKs (see §2)*
  - `made_by_user_id BIGINT NOT NULL` (Makes: User → Request)
  - `resource_id BIGINT NOT NULL` (Targets: Request → Resource)

1.4 Access Event → Access_Event

- **Columns**
  - `event_id BIGINT` PK `AUTO_INCREMENT`
  - `time_stamp DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP` (ER time stamp)
  - `outcome ENUM('SUCCESS','FAILED') NOT NULL`
  - *Folded relationship FKs (see §2)*
  - `user_id BIGINT NOT NULL` (Trigger: User → Access_Event)
  - `resource_id BIGINT NOT NULL` (ON: Resource → Access_Event)

Relationship sets → Foreign keys

2.1 Makes (User 1:N Request)

- **Implementation:** `Request.made_by_user_id` → `UserAccount(user_id)` (`NOT NULL`).

- **Semantics:** Each request is made by exactly one user; a user may make many requests.

2.2 Targets (Request N:1 Resource)

- **Implementation:** `Request.resource_id` → `Resource(resource_id)` (`NOT NULL`).
- **Semantics:** Each request targets exactly one resource; a resource can be targeted by many requests.

2.3 ON (Resource 1:N Access Event)

- **Implementation:** `Access_Event.resource_id` → `Resource(resource_id)` (`NOT NULL`).
- **Semantics:** Each access event occurs on exactly one resource; a resource can have many events.

2.4 Trigger (User 1:N Access Event)

- **Implementation:** `Access_Event.user_id` → `UserAccount(user_id)` (`NOT NULL`).
- **Semantics:** Each access event is triggered by exactly one user; a user can trigger many events.

Note: The ERD does not include an Event↔Request relationship, so there is no join table for that.

ISA hierarchies (class-table, shared PK)

3.1 User ISA {Student, Staff, Admin, Security, Visitor}

- **Tables:** `Student`, `Staff`, `AdminUser`, `SecurityUser`, `Visitor`
- **Shape:** Each has `user_id BIGINT` PK with `FOREIGN KEY ... REFERENCES UserAccount(user_id) ON DELETE CASCADE`.
- **Semantics:** Membership in a subtype table encodes the subtype. (ERD doesn't specify disjointness/totality; not enforced.)

3.2 Request ISA {Access_Request, Extension_Request}

- **Tables:**
  - `Access_Request(request_id` PK FK→`Request)`
  - `Extension_Request(request_id` PK FK→`Request, extend_from DATETIME, extend_until DATETIME)`
- **Optional integrity:** Could add `CHECK (extend_from < extend_until)`.

3.3 Resource ISA {Room, SystemRes}

- **Tables:**
  - `Room(resource_id` PK FK→`Resource, building VARCHAR(64), room_no VARCHAR(16), capacity INT)`
  - `SystemRes(resource_id` PK FK→`Resource, hostname VARCHAR(120))`
- **Optional integrity:** e.g., `CHECK (capacity >= 0)`.

3.4 Access_Event ISA {Success_Event, Failed_Event}

- **Tables:**
  - `Success_Event(event_id` PK FK→`Access_Event)`
  - `Failed_Event(event_id` PK FK→`Access_Event, fail_reason VARCHAR(255))`
- **Consistency with supertype:** `Access_Event.outcome` records `SUCCESS` vs `FAILED`. (Can add `CHECK`s or triggers so subtypes match this value.)

Keys & Constraints (quick list)

- **PKs:** `UserAccount.user_id`, `Resource.resource_id`, `Request.request_id`, `Access_Event.event_id`, and each subtype's id column.
- **FKs:**
  - `Request.made_by_user_id` → `UserAccount.user_id` (`NOT NULL`)
  - `Request.resource_id` → `Resource.resource_id` (`NOT NULL`)
  - `Access_Event.user_id` → `UserAccount.user_id` (`NOT NULL`)
  - `Access_Event.resource_id` → `Resource.resource_id` (`NOT NULL`)
  - Each subtype id → its supertype id (`ON DELETE CASCADE`)

- **UNIQUE:** `UserAccount.email`
- **ENUMs:** `Request.status`, `Access_Event.outcome`
- **Referential actions:** Cascade only on ISA links; otherwise default `RESTRICT`.

Assumptions & deviations from ERD

- Pure renamings for SQL compatibility: `User`→`UserAccount`, `System`→`SystemRes`, `time stamp`→`time_stamp`.
- No extra attributes/relationships added beyond the ERD.
- ISA disjointness/totality not specified in ERD, so not enforced in the schema.