EDABA project report:
High-security facility access control system

Task 1: Design and implementation of database

Michał Szopiński 300182

November 25, 2021

1 Overview

This document describes a database system which could be implemented to manage and monitor personnel activity within a high-security facility, e.g. a military establishment, a research laboratory or a storage vault.

1.1 Facility layout and organization

A closed-access building houses a high-security facility where personnel activity must be strictly controlled and monitored. The building is divided into several rooms, the access to which is guarded by a turnstile and a card reader device placed on both sides of the doorway. Each employee is given an ID card to authenticate themselves and request permission from the system to enter an area. If the permission is granted, the turnstile becomes temporarily unlocked. If the turnstile is turned, the activity is registered in the system. The position of an employee is tracked at all times and nobody may enter or leave the facility without the proper permissions.

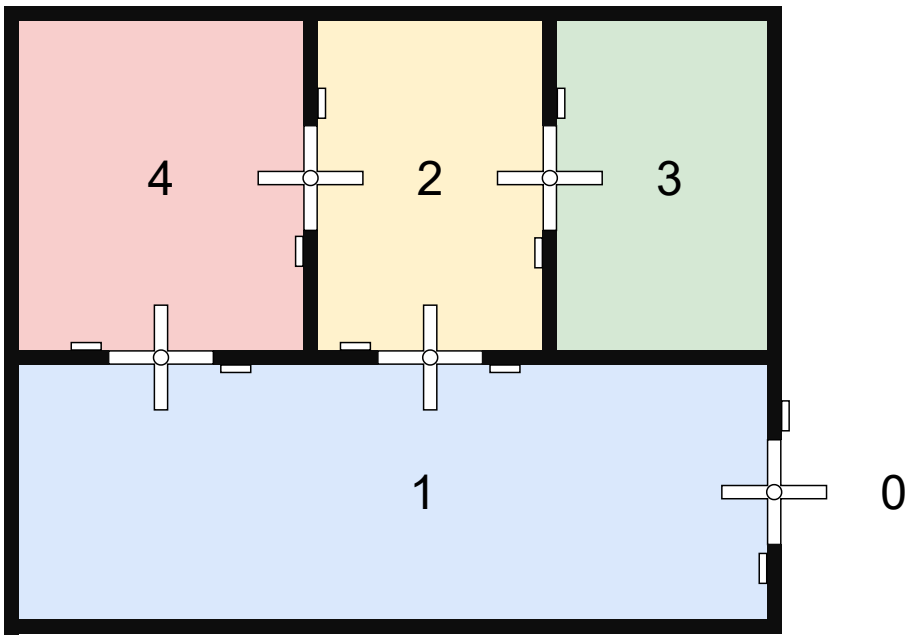


Figure 1: A top-down view of the facility with access areas marked with numbers. 0 denotes the "outside" access area.

1.2 System objectives

1. **Access control** – the administrator is able to configure which employees may enter which areas
2. **Activity tracking and reports** – the administrator is able to review the location of an employee as it changes with time and receive reports on how much time employees spend within an area
3. **Access violation detection** – the administrator is notified when an access violation occurs (e.g. the same ID card was used to enter the same room twice)

2 Database entities and relations

2.1 Schedule

Describes the work hours of an employee. Attributes include the start time and end time for each weekday. Used to detect when an employee is present in the building outside of work hours.

2.2 Employee

Basic entity describing an employee. Actions are associated with employees based on employee IDs encoded on ID cards.

2.2.1 Relation with schedule

Each employee has a schedule describing their work hours. Employees may share schedules. A schedule may be associated with no employees.

2.2.2 Relation with room

An employee may be permitted to enter zero or more rooms. A room may be entered by zero or more employees. Permissions may be temporary or permanent.

2.3 Room

Describes an access area employees may be permitted to enter.

2.4 Gate

Describes a connection between rooms. Together with the room entity it forms a graph describing the building.

2.4.1 Relation with room

A room may be connected to other rooms (or may contain an entrance to the building) through gates. Each gate connects two rooms.

2.5 Activity

Describes an action performed by an employee.

Type	Name	Description
0	Unlocked	Unlocked a turnstile, but has yet to turn it.
1	Passed	Turned the previously unlocked turnstile.
2	Abandoned	The previously unlocked turnstile timed out and became locked again.

2.5.1 Relation with employee, room and gate

The activity is associated with the employee who performed it. It is also described by the room where it happened and the gate that triggered it.

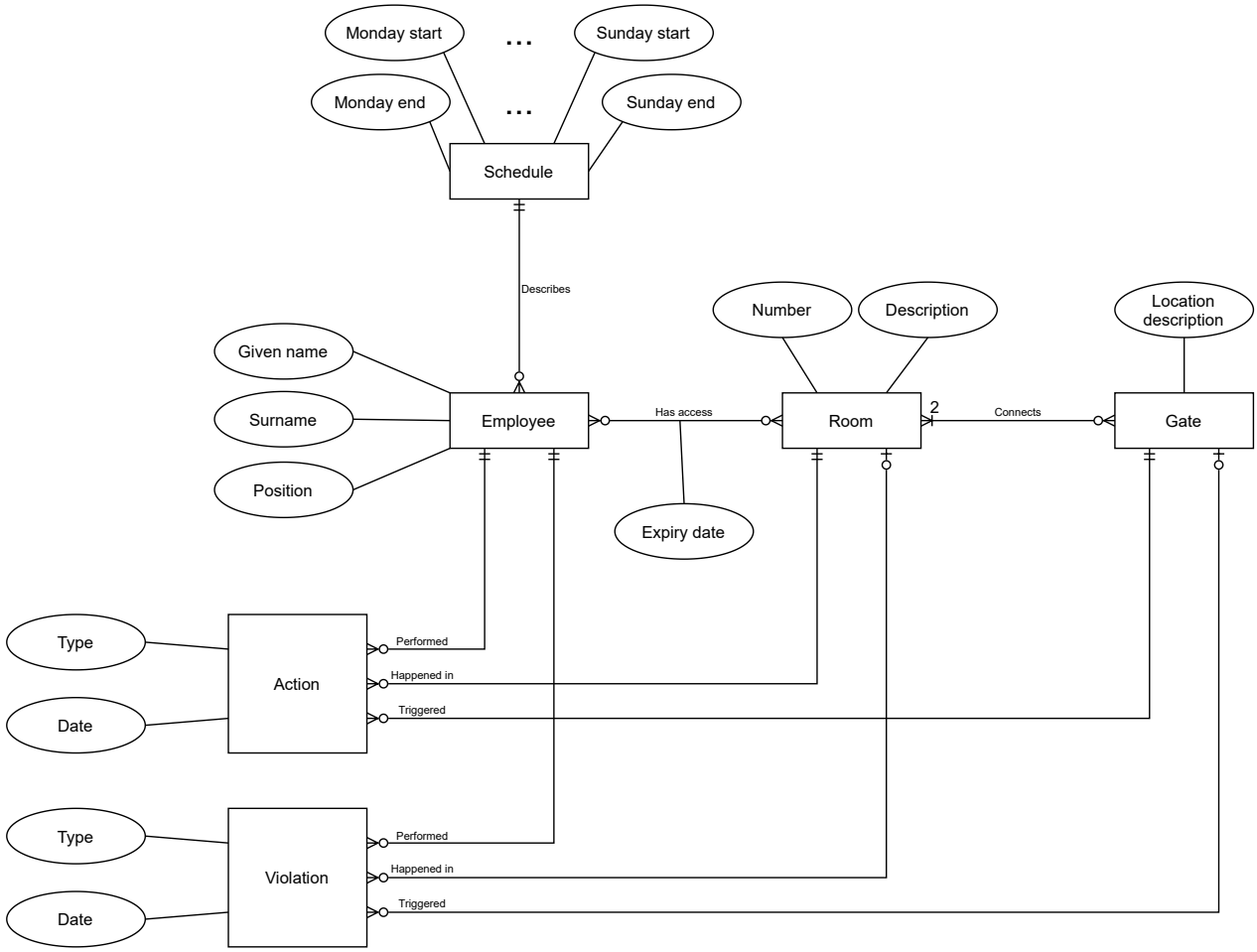
2.6 Violation

Describes an illegal action (security incident) performed by an employee.

Type	Name	Description
0	Absent holder	The recorded location of the holder doesn't match the location of the gate.
1	Unauthorized	The holder requested access to a room they are not permitted to enter.
2	Overstayed	The holder was present in the facility outside of their work hours.
3	Late entry	The holder attempted to enter the facility outside of their work hours.

2.6.1 Relation with employee, room and gate

The violation is associated with the employee who performed it. The location and gate attributes apply if relevant.


Monday start
...
Sunday start
Monday end
...
Sunday end
Schedule
Describes
Number
Description
Location description
Given name
Surname
Position
Employee
Has access
Room
2
Connects
Gate
Expiry date
Type
Date
Action
Performed
Happened in
Triggered
Type
Date
Violation
Performed
Happened in
Triggered

Schedules	
PK	**ScheduleID: int**
	MondayStart: int
	MondayEnd: int
	...10 more columns...
	SundayStart: int
	SundayEnd: int

Employees	
PK	**EmployeeID: int**
	GivenName: varchar
	Surname: varchar
	Position: varchar
FK	Schedule: int

Permissions	
PK	**PermissionID: int**
	Expires: date
FK	Employee: int
FK	Room: int

Rooms	
PK	**RoomID: int**
	Number: varchar
	Description: varchar

Gates	
PK	**GateId: int**
	Location: varchar
FK	RoomA: int
FK	RoomB: int

Actions	
PK	**ActionID: int**
	ActionType: int
	ActionDate: date
FK	Employee: int
FK	Room: int
FK	Gate: int

Violations	
PK	**ViolationID: int**
	ViolationType: int
	ViolationDate: date
FK	Employee: int
FK	Room: int or null
FK	Gate: int or null

```sql
CREATE TABLE Schedules (
    ScheduleID INTEGER PRIMARY KEY NOT NULL,
    MondayStart INTEGER NOT NULL,
    MondayEnd INTEGER NOT NULL,
    TuesdayStart INTEGER NOT NULL,
    TuesdayEnd INTEGER NOT NULL,
    WednesdayStart INTEGER NOT NULL,
    WednesdayEnd INTEGER NOT NULL,
    ThursdayStart INTEGER NOT NULL,
    ThursdayEnd INTEGER NOT NULL,
    FridayStart INTEGER NOT NULL,
    FridayEnd INTEGER NOT NULL,
    SaturdayStart INTEGER NOT NULL,
    SaturdayEnd INTEGER NOT NULL,
    SundayStart INTEGER NOT NULL,
    SundayEnd INTEGER NOT NULL
);

CREATE TABLE Employees (
    EmployeeID INTEGER PRIMARY KEY NOT NULL,
    GivenName VARCHAR(64) NOT NULL,
    Surname VARCHAR(64) NOT NULL,
    Position VARCHAR(128) NOT NULL,
    Schedule INTEGER NOT NULL,

    CONSTRAINT employee_has_schedule
    FOREIGN KEY (Schedule) REFERENCES Schedules(ScheduleID)
);

CREATE TABLE Rooms (
    RoomID INTEGER PRIMARY KEY NOT NULL,
    RoomINTEGER VARCHAR(16) NOT NULL,
    Description VARCHAR(64) NOT NULL
);

CREATE TABLE Permissions (
    PermissionID INTEGER PRIMARY KEY NOT NULL,
    Expires DATE NOT NULL,
    Employee INTEGER NOT NULL,
    Room INTEGER NOT NULL,

    CONSTRAINT permission_employee
    FOREIGN KEY (Employee) REFERENCES Employees(EmployeeID)
    ON DELETE CASCADE,

    CONSTRAINT permission_room
    FOREIGN KEY (Room) REFERENCES Rooms(RoomID)
    ON DELETE CASCADE
);

CREATE TABLE Gates (
    GateID INTEGER PRIMARY KEY NOT NULL,
    Location VARCHAR(64) NOT NULL,
    RoomA INTEGER NOT NULL,
    RoomB INTEGER NOT NULL,

    CONSTRAINT gate_room_a
    FOREIGN KEY (RoomA) REFERENCES Rooms(RoomID)
    ON DELETE CASCADE,
    CONSTRAINT gate_room_b
    FOREIGN KEY (RoomB) REFERENCES Rooms(RoomID)
    ON DELETE CASCADE,

    CONSTRAINT gate_room_order CHECK (RoomA < RoomB)
```

```
);

CREATE TABLE Actions (
    ActionID INTEGER PRIMARY KEY NOT NULL,
    ActionType INTEGER NOT NULL,
    ActionDate DATE NOT NULL,
    Employee INTEGER NOT NULL,
    Room INTEGER NOT NULL,
    Gate INTEGER NOT NULL,

    CONSTRAINT action_employee
    FOREIGN KEY (Employee) REFERENCES Employees(EmployeeID),
    CONSTRAINT action_room
    FOREIGN KEY (Room) REFERENCES Rooms(RoomID),
    CONSTRAINT action_gate
    FOREIGN KEY (Gate) REFERENCES Gates(GateID),

    CONSTRAINT a_type_in_range CHECK (ActionType BETWEEN 0 AND 2)
);

CREATE TABLE Violations (
    ViolationID INTEGER PRIMARY KEY NOT NULL,
    ViolationType INTEGER NOT NULL,
    ViolationDate DATE NOT NULL,
    Employee INTEGER NOT NULL,
    Room INTEGER,
    Gate INTEGER,

    CONSTRAINT violation_employee
    FOREIGN KEY (Employee) REFERENCES Employees(EmployeeID),
    CONSTRAINT violation_room
    FOREIGN KEY (Room) REFERENCES Rooms(RoomID),
    CONSTRAINT violation_gate
    FOREIGN KEY (Gate) REFERENCES Gates(GateID),

    CONSTRAINT v_type_in_range CHECK (ViolationType BETWEEN 0 AND 3)
);
```