National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Faculty of Computing
CS220: Database Systems

Class: BSDS-02

Person	Assigned Sections & Tasks
Momin	- Describe **System Architecture** and Modules - Justify **DBMS selection** (SQL vs NoSQL) - Explain **Normalization approach** (1NF → 3NF) - made final erd diagram
Ali	- Write **Relational Algebra Expressions** (π, σ, ⋈, etc.) for key queries - Justify **Schema Design** - Explain how schema supports normalization and requirements - Write **SQL Queries**: INSERT, UPDATE, JOINs, WHERE, aggregation (e.g., COUNT, GROUP BY)
Gulwarina	- Define **Project Objectives and Scope**: Real-world problem, system goals and functionalities - Identify and apply **Indexes** where necessary - Provide **Analysis and Conclusion**: Strengths, limitations, design reflection, future improvements - compiled report
All 3	- Design **Schema (ERD)**: Entities, relationships, attributes - Define **Constraints**: Primary key, foreign key, NOT NULL, UNIQUE, etc. - Define **Attribute data types (** the data gathering, mock erd schema, constraints, indexes,dbms selection, project objectives and requirement were all decided on call/in university, the work distribution mainly applies to how they were documented in the report)

Project Report

The sql script is included in the zip file

Instructor: Dr. Fahad Satti

TABLE OF CONTENTS

1. Project Objectives and Scope

1.1 Real-World Problem Definition

At present, NUST's gym and sports complex relies almost entirely on paper registers, printed forms, and ad-hoc email or phone communications to manage:

- **User registrations & renewals** (students, faculty, alumni)
- **Attendance tracking** at gym, pool, bowling alley, and other facilities
- **Fee collection & reconciliation** via physical receipts
- **Equipment inventory, damage reporting, and repair approvals**
- **Facility bookings** for inter-departmental events and external organizations
- **Feedback and issue logging** in separate complaint books

This manual approach leads to:

- **Time-consuming record retrieval**, especially for old or cross-facility queries
- **Surprise mis-charges** (e.g. Ramadan pause not recorded) and payment inconsistencies
- **Difficulty enforcing access rules** for alumni, gender-segregated slots, or token limits
- **Delayed maintenance**, because damage reports live on paper and approvals can be delayed
- **Congestion at peak hours** (5 PM–9 PM) with no real-time visibility into facility usage
- **Lack of accountability** and traceability for who made changes or approved requests

Together, these issues impact:

- **Gym & sports staff**, who spend large portions of their day on paperwork instead of training or supervision

- **Students & faculty**, who experience registration delays, broken equipment, and slow feedback processes
- **DD Sports & administration**, who lack timely, accurate data for proper planning and event coordination

1.2 Goals of the Proposed System

1. **Digital Onboarding & Verification**
   - Reduce new-member registration time through an online form and document upload portal.
2. **Real-Time Attendance & Billing**
   - Achieve quick response times for attendance check-ins and payment status lookups.
3. **Centralized Equipment & Maintenance Workflow**
   - equipment-repair via an online request/approval system with automatic notifications.
4. **Unified Facility Booking & Reporting**
   - Enable DD Sports to view and schedule all 147 sports facilities in one interface, reducing booking conflicts
5. **Effective Feedback & Analytics**
   - Launch a single feedback portal with status tracking; generate weekly usage and satisfaction dashboards.
6. **Data Security & Retention**
   - Protect all personal and medical data at rest and ongoing; retain atleast five years of records with daily backups.

1.3 Scope of the System

- **User Management Module**
    - Online registration, profile editing, document uploads, alumni access flags
- **Membership & Payment Module**
    - Subscription plans (single/multi-facility), automated billing cycles, reminder notifications
- **Attendance Module**
    - Facility-specific check-in/out, staff shifts, overtime tracking, reporting exports
- **Equipment & Maintenance Module**
    - Inventory catalog, damage reports, vendor tracking
- **Booking & Scheduling Module**
    - Central portal for paid/free facility reservations, pool gender-blocks + individual slots, event workflows
- **Feedback & Issue Tracker**
    - Unified online feedback submission, status updates, category tagging
- **Analytics & Reporting Module**
    - Real-time dashboards: daily footfall, peak congestion, maintenance KPIs, user satisfaction

2. Requirements Gathering

2.1 Stakeholder Identification:

Role / Stakeholder	Goals	System Interactions
Student Member	• Quickly register or renew gym/pool membership • Check in/out easily • Track their own bookings and feedback status	• Fill online registration form, upload documents • View membership status & pay fees • Self-serve check-in • Submit feedback or equipment requests • Book pool slots or bowling games
Faculty Member	• Maintain active access to facilities • View usage history • Provide feedback	• Same as Student Member, plus view reports of their own past attendances and payments
Alumni Member	• Gain/retain access according to policy • See available plans and fees	• Request alumni activation via portal • View and pay alumni-specific membership • Check in with alumni flag enforcement
Gym Trainer / Frontline Staff	• Register new users accurately • Record attendance & equipment issues • Review member payment status	• Use registration wizard to onboard users • Log attendance for members/staff • Enter maintenance requests • Verify payments and view member profiles
Maintenance Staff	• Receive and act on repair approvals • Track equipment inventory & vendor orders	• View open maintenance requests assigned to them • Update request status (in-progress, completed) • Access equipment and vendor catalogs
DD Sports Administrator	• Oversee all facility bookings & schedules • Manage high-level reports & budgets • Approve event requests and maintenance plans	• Use centralized booking dashboard to approve or adjust facility reservations • Generate and export attendance, usage, and maintenance reports • Approve recurring maintenance plans
System Administrator	• Keep system secure • Manage user roles and data backups	• Define Roles and permissions • Monitor performance metrics • Schedule and verify backups

Finance / Accounts	• Reconcile payments and billing cycles • Send payment reminders and reports	• View and export payment records via the Payment Module • Trigger reminder notifications • Access past transactions

2.2 Functional Requirements

ID	Requirement	Justification (source)	ERD Mapping / Key Tables
F1	**User & Membership Registration + Document Upload** Register new users via form and upload docs.	Paper registers & Qalam forms with photos, IDs, challans across multiple staff (Female Q3–Q5, Sara Q3, Abbas Q3)	USER + MEMBERSHIP
F2	**CRUD on Memberships & Plans (including bundled plans)** Create, update/pause, cancel subscriptions.	Mischarges for Ramadan pause & multi-facility plans (Abbas Q14–Q16; Umer/Mohammad multi-plan)	MEMBERSHIP,
F3	**Payment Processing & Reminders** Record payments, send due-date reminders, enforce grace period.	Physical receipts & ad-hoc reminders, 10-day payment window (Sara Q3; Mohammad “25th of every month + 10 days”)	PAYMENT
F4	**Facility-Specific Attendance Tracking** Log daily in/out per user or staff per facility.	Paper registers per facility (Hamza Q2; Habib Q2; Mahtab Q2, Q7; Umer/Sara manual registry)	ATTENDANCE (user_id/staff_id, facility_id, time_in, time_out)
F5	**Attendance Reporting & Export** Generate/send reports to DD Sports/Rector.	Habib sends registers to DD Sports → Rector (Habib Q2)	Queries on ATTENDANCE + USER/STAFF/FACILITY

F6	**Alumni Access Control** Flag allowed/inactive alumni and enforce at check-in.	Selective alumni allowed (Habib Q2)	ALUMINI
F7	**Role-Based Access** Enforce roles (trainer, supervisor, admin)	Supervisors approve maintenance; AD/DD view data; trainers enter data (Hamza Q6; Female Q6; Habib Q7)	
F8	**Equipment Inventory & Damage Logging** Track items, flag damage, record status.	Trainers patrol & report damage via forms; stock replacements (Hamza Q4–Q5; Habib Q7; Mahtab Q10)	EQUIPMENT, MAINTENANCEREQUEST
F9	**Maintenance-Approval Workflow** Submit requests → supervisor/AD/DD approval → log repair.	Paper approval sheets (Hamza Q5; Female Q7)	MAINTENANCEREQUEST
F10	**Vendor & Procurement Management** Track vendors, costs, delivery statuses.	Local & imported equipment, multi-vendor sourcing (DD Q14–Q15)	VENDOR links to EQUIPMENTTYPEVENDOR
F11	**Recurring Maintenance & Budget Planning** Define tasks (chlorination, lighting), budgets, schedule.	Annual routine visits & budget approvals (DD Q7–Q8)	MAINTENANCEREQUEST
F12	**Centralized Booking & Scheduling Portal** Master interface for all facility reservations.	Multi-facility ION/email/call requests handled manually today (DD Q9–Q10)	EVENTBOOKING + FACILITYBOOKING

F13	**Pool Block & Individual Slot Scheduling** Gender blocks + 45 min individual sub-slots.	Gender-segmented blocks & per-person 45 min slots (Umer)	TIMESLOT
F14	**Token Management for Pay-Per-Use** Issue & limit tokens (e.g. bowling max 4), track issuance.	Bowling tokens, CMS ID + color-coded games, limit 4 (Mahtab Q2–Q4, Q8)	TOKEN (user_id, facility_id, serial_no, game_number)
F15	**Event Booking Workflow via DD Sports(similar to f12)** Submit events → DD approval → auto-schedule & notify.	SEECES/org events via DD Sports & email confirmations (Habib Q8; Mahtab Q11; Female Q14)	
F16	**Feedback & Issue Tracking** Unified portal for complaints, suggestions, status tracking.	No online feedback or status tracking today (Sara Q11–Q12; Abbas Q18; Female Q15; DD Q11–Q12)	FEEDBACK (user_id, facility_id, category, status)
F17	**On-Demand Record Retrieval Dashboard** AD/DD view & export any member's full history.	Random AD/DD requests for paper records; hard to find student files (Female Q8–Q12)	Aggregated views: USER + ATTENDANCE + PAYMENT
F18	**Search Across All Records** Fast lookup by name, CMS ID, membershipid, equipmentid, etc.	Paper-search hard & need for quick access (Female Q12; general interviews)	Indexed fields in USER, ATTENDANCE, PAYMENT, EQUIPMENT, FEEDBACK
F19	**Staff Work-Hours & Overtime Tracking** Capture clock-in/out, calculate normal/overtime hours.	12 hr days vs 8 hr shifts, low OT pay (Habib Q4; Mahtab Q13)	SPORTSCOMPLEXSTAFF+ derived OVERTIME

F20	**Usage & Performance Analytics Dashboard** Real-time footfall, peak congestion, satisfaction.	DD Sports KPIs: daily footfall (1,000–3,000), user feedback, peak-hour rush (DD Q6, Q17)	Aggregates on ATTENDANCE, EVENTBOOKING, FEEDBACK

2.3 Non-Functional Requirements

ID	Requirement	Justification	ERD Implementation
N1	**Responsiveness**	Staff and students shouldn't wait, especially at peak times when 2,500 – 3,000 people check in daily (DD Q17).	Adding indexes on search-heavy columns (e.g. FACILITY.facility_id, USER.cms_id) and ensuring lookups use those keys.
N2	**Simplicity & Usability**	Many staff are used to paper registers; clear, step-by-step, mobile-friendly forms help them switch without training (Female Q3–Q5).	Keep entity and attribute names in the ERD human-readable (e.g. user.first_name not usr_fn)
N3	**Maintainability**	Future student-developers need to add new sports or fix bugs without going through messy code	Use clear naming conventions in the ERD (e.g. FACILITY_TYPE, MAINTENANCE_REQUEST)
N4	**Scalability (Growth-Ready)**	The university may add new facilities or see membership grow; the system must handle that without a full redesign.	Model FACILITY and FACILITY_TYPE as extensible lookup tables, and avoid hard-coding facility lists in entity attributes.

3. System Architecture and Modules

3.1 Modules Description

Module Name	Functionality	Detail
User Management	Manages user identity, profiles, and role types such as student, faculty, alumni, and staff.Profile-specific tables like STUDENT and FACULTYMEMBER extend USER.	Inputs: USER - first_name, last_name, email, phone, cnic etc. Outputs: user_id, cms_id, facultymember_id, alumni_id
Membership Management	Handles membership records, plan types, freeze periods, and violation tracking. Each membership is linked to a user and a membership type. Freeze history and violations are stored in separate tables.	Inputs: user_id, membership_type_id, start_date, end_date, status from MEMBERSHIP. Outputs: membership_id, freeze_id, violation_id
Facility Booking	Manages booking of facilities by users. Schedule and access are controlled via supporting tables.	Inputs: facility_id, day_of_week, start_time, end_time from FACILITYSCHEDULE. Outputs: booking_id, available time slot
Attendance Tracking	Records check-in and check-out activity per facility for users and staff. Tracks usage frequency and helps generate reports.	Inputs: user_id, facility_id, check_in_time, check_out_time. Outputs: attendance_id

Payment Management	Records all payment transactions for memberships or facility use. Links payments to memberships and validates payment status.	Inputs: membership_id, amount, payment_type, payment_date, is_fee_charged. Outputs: payment_id, status
Feedback Management	Collects user comments or complaints about facilities. Feedback entries are linked to the user and facility involved and stored with their current status.	Inputs: user_id, facility_id, date_submitted, feedback_type, description. Outputs: feedback_id, feedback status
Maintenance Module	Logs issues related to facilities and equipment. Each maintenance request is linked to a user, equipment item, and optionally a staff member.	Inputs: equipment_id, user_id, report_date, description, staff_id, status. Outputs: request_id, resolution status
Token Management	Tracks usage of game tokens issued to users for pay-per-use facilities like bowling. Records are linked to users and facilities.	Inputs: user_id, game_number, issue_date. Outputs: token_id
Equipment Management	Stores data about facility equipment, types, condition, and suppliers. Also tracks the last maintenance date.	Inputs: facility_id, name, status, purchase_date, equipment_type_id. Outputs: equipment_id, vendor links
Vendor Management	Records details of vendors and their services. Connects vendors to equipment types and procurement orders.	Inputs: name, service_type, contact_type, contact_value.

		Outputs: vendor_id, contact_id
Reporting and Analytics	Provides summaries and insights about attendance, payments, facility usage, and user activity. Built using joins and aggregations on normalized data.	Inputs: filters such as user, facility, date, metrics. Outputs: summaries, trends, downloadable reports

4. Database Design Decisions and Justifications

4.1 DBMS Type Selection

- **Chosen:** SQL
- **Justification:**

The DBMS (Database Management System) chosen for the presented system is MySQL, and Structured Query Language (SQL) has been utilized which is a declarative language used to manage and manipulate relational databases. It provides powerful capabilities for querying, updating, and organizing structured data stored in tables with predefined schemas.

1) Structured and Relational Data

The system is highly relational. All entities are closely linked to at least one other entity through various relationships with the user of foreign keys, which are required for this system to function properly. SQL is specifically designed to efficiently manage such relational and structured data.

2) Data Integrity and Consistency

SQL databases assist in enforcing data consistency through various sorts of constraints such as foreign keys, primary keys and NOT NULL constraints. These ascertain that the data is consistent and relationships remain valid across numerous entities. This prevents occurrances such as orphaned or duplicate entities, which may prove to be critical issues for the architecture.

3) Complex Queries and Joins

This system has a recurring need to retrieve data from multiple tables to solve our problems. For Example, we may need to retrieve the data about students who have not paid their monthly dues. In order to achieve such data, we have to make use of “Joins” and conditional filtering using “where”. All these functional operations are provided by SQL which makes it more convenient for us to implement complex business logic.

4) Transaction Support

SQL databases provide ACID (Atomicity, Consistency, Isolation, Durability) compliant transactions. This is necessary when multiple related operations are performed together. SQL ensures that these actions either complete successfully as a unit or are rolled back to avoid partial updates, maintaining a reliable state of data.

5) Data Aggregation and Reporting

SQL allows you to group or aggregate data and retrieve it. Say you have to make a report on data such as the total payments in a certain month, or the most attended facility. To achieve this, you can utilize aggregate functions such as “COUNT()” alongside “Group By” which are provided by SQL, making the system ideal for analytics and reporting directly from the database.

6) Security and Access Control

SQL provides us with role based access control. Different roles (such as admin, staff, student etc) have different levels of access to the database. This ensures protection of sensitive data and system integrity, as it restricts the users that can read, insert or update data.

4.2 Data Types for Attributes with justification

Table	Attribute	Data Type And Constraints	Justification
User	user_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	first_name	VARCHAR(255) NOT NULL	Names are textual and are better saved as varchar. User must have a name
	last_name	VARCHAR(255) NOT NULL	Names are textual and are better saved as varchar. User must have a name
	dob	DATE NOT NULL	Stores year-month-day cleanly; allows age calculations. User must have a dob.
	gender	ENUM('Male','Female','Other') NOT NULL	Fixed set of valid options as genders are limited,small storage overhead. User must provide gender information.
	email	VARCHAR(255) UNIQUE NOT NULL	Variable length and varchar saves space, enforces uniqueness as each address is unique and every user must provide an email.
	phone	VARCHAR(255) NOT NULL	Not purely numeric as they can include “+” or “-”. Every user must provide a phone number as well.
	cnic	VARCHAR(255) UNIQUE NOT NULL	Not purely numerical as it has “-”. Every person has a unique CNIC and it must be provided for identification.

	membership_status	ENUM('Active','Inactive','Suspended')	Only a limited options are available for this attribute so an ENUM is suitable.
	address	TEXT	Unpredictable length and may include numbers and hyphens.
	emergency_contact	VARCHAR(255)	Phone number or other contact options; same format as phone.
user_medical_info	user_medical_info_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Cannot be null as it medical info requires for a user to be related to it
	medical_condition	VARCHAR(255) NOT NULL	Short description of condition. Cannot be null because medical info is required in this table
	details	TEXT	Details can vary in length.
STUDENT	cms_id	INT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the user table has attributes linked to a student.
	program_id	INT NOT NULL FOREIGN KEY	Must be provided as each student is enrolled in a program.
	Residency_Type	ENUM('Day','Hostel','Other')	Limited valid values; an ENUM is suitable.

	batch_year	YEAR	Four-digit year; YEAR type enforces valid range.
FACULTYMEMBER	facultymember_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the user table has attributes linked to a faculty member.
	department_id	INT NOT NULL FOREIGN KEY	Must be provided as each faculty member works in a department
	rank	VARCHAR(255) NOT NULL	Each faculty member has a rank; cannot be null
FAMILYMEMBER	familymember_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the user table has attributes linked to a family member.
	facultymember_id	INT NOT NULL FOREIGN KEY	Must be provided as a family member must be linked to a faculty member to avail the facilities.
ALUMNI	alumni_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL	Must be provided as the user table has attributes linked to an alumni.

	graduation_year	YEAR NOT NULL	Year of graduation; valid four-digit years. Not null as an alumnus have to provide their graduation year.
PROGRAM	program_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	program_name	ENUM (BSDS,BSAI,BSDSCS) NOT NULL	Limited name opt; it cannot be null as each program must have a name.
	department_id	INT NOT NULL FOREIGN KEY	Must be provided as each program is a part of a certain department.
	degree_level	ENUM('BSc','MSc','PhD')	Fixed number of options.
	duration_years	TINYINT NOT NULL	Small integer for program length in years. Every program has a certain duration so it must have a value
DEPARTMENT	department_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	name	ENUM (SEECS,SMME,SNS…) NOT NULL	Limited name options; It cannot be null as each department has a name.
	contact_email	VARCHAR(255) NOT NULL	Email must be a varchar as it can have special characters. It cannot be NULL so that a department can be contacted.

MEMBERSHIP	membership_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the user table has attributes linked to a member.
	membership_type_id	INT NOT NULL FOREIGN KEY	Member must have a certain type of membership.
	start_date	DATE NOT NULL	Date for the initiation of membership; Has to be filled to keep record.
	end_date	DATE NOT NULL	Date for the termination of membership; Has to be filled to keep record.
	status	ENUM('Active','Expired','Cancelled') NOT NULL	Limited options for status of membership. Must have a status.
	card_status	ENUM(‘Applied, Active, Lost’) NOT NULL	Limited options for card status. Must have a status.
	freeze_months_used	TINYINT DEFAULT 0	Count of freeze months; small integer.
	last_intimation_date	DATE	Date type to keep record for the visit.
MEMBERSHIP TYPE	membership_type_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.

	type	ENUM (Regular, Gold etc) NOT NULL	Limited options for type. Must have a type of membership.
	registration_fee	DECIMAL(8,2) NOT NULL	Currency can be up to 2 decimal places for accuracy. Membership type must have a registration fee.
	monthly_fee	DECIMAL(8,2) NOT NULL	Currency can be up to 2 decimal places for accuracy. Membership type must have a monthly fee.
MEMBERSHIPFREEZE	freeze_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	membership_id	INT NOT NULL FOREIGN KEY	In order to freeze, there must be a membership in the first place.
	start_date	DATE NOT NULL	Date for the initiation of freezing; Has to be filled to keep record.
	end_date	DATE NOT NULL	Date for the initiation of membership freeze; Has to be filled to keep record.
	request_date	DATE NOT NULL	Date for the initiation of membership freeze; Has to be filled to keep record.
MEMBERSHIP_VIOLATION	violation_id	INT NOT NULL FOREIGN KEY	Must be added for the relationship.
	membership_id	INT NOT NULL FOREIGN KEY	Must be added for the relationship.

LOCKERRENTAL	locker_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	membership_id	INT NOT NULL FOREIGN KEY	Locker must be rented by a member.
	issue_date	DATE NOT NULL	Date for the initiation of locker rental; Has to be filled to keep record.
	return_date	DATE NOT NULL	Date for the termination of locker rental; Has to be filled to keep record.
PAYMENT	payment_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	membership_id	INT NOT NULL FOREIGN KEY	A member must exist to make a payment.
	payment_date	DATETIME NOT NULL	Timestamp of payment required to keep the record.
	amount	DECIMAL(10,2) NOT NULL	Currency can be up to 2 decimal places for accuracy. Amount must be specified for a payment.
	payment_type	ENUM('Cash','Card','Online') NOT NULL	Limited options for payment. Payment must have a type.
	status	ENUM('Pending','Completed','Failed') NOT NULL	Limited options for status of payment. Payment must have a status.

	is_fee_charged	BOOLEAN DEFAULT FALSE NOT NULL	Either true or false. False by default as fee isn’t charged without a reason. Has to be specified, cannot be NULL.
NSTPTENANT	nstpid	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the user table has attributes linked to an NSTP Tenant.
	tenant_start_date	DATE NOT NULL	Date for the initiation of tenancy; Has to be filled to keep record.
	tenant_end_date	DATE NOT NULL	Date for the completion of tenancy; Has to be filled to keep record.
FACILITY	facility_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	name	VARCHAR(255) NOT NULL	Name must be a varchar, and a facility must have a name.
	location	TEXT NOT NULL	Location must be a text, and a facility must have a location.
	max_capacity	INT NOT NULL	Number of people that a capacity can facilitate, hence an integer. Each facility has a capacity; cannot be NULL.
FACILITY_BOOKING	booking_id	INT NOT NULL FOREIGN KEY	Required for the current relationship.

	facility_id	INT NOT NULL FOREIGN KEY	Required for the current relationship.
FACILITYSCHEDULE	schedule_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	facility_id	INT NOT NULL FOREIGN KEY	A schedule must be linked to a facility.
	day_of_week	ENUM('Mon','Tue','Wed','Thu','Fri','Sat','Sun') NOT NULL	Constrained list of weekdays. A day must be chosen for a schedule.
	open_time	TIME NOT NULL	Facility must have a time it opens at.
	close_time	TIME NOT NULL	Facility must have a time it closes at.
FACILITY_MEMBERSHIPTYPE	facility_membershiptype_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	facility_id	INT NOT NULL FOREIGN KEY	Must be there for the relationship to exist.
	membership_type_id	INT NOT NULL FOREIGN KEY	Must be there for the relationship to exist.
STAFF_WORKING_SCHEDULE	schedule_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	staff_id	INT NOT NULL FOREIGN KEY	Schedule must be made only if there is a staff member; cannot be null.

	day_of_week	ENUM('Mon','Tue','Wed','Thu','Fri','Sat','Sun') NOT NULL	Limited options for weekdays. A day must be chosen for a schedule.
	start_time	TIME NOT NULL	Shift start time must be provided for a schedule.
	end_time	TIME NOT NULL	Shift end time must be provided for a schedule.
STAFF_DAYS_OFF	days_off_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	staff_id	INT NOT NULL FOREIGN KEY	Days off only applicable when a staff member exists; cannot be null.
	date	DATE NOT NULL	Day to be off must be mentioned.
	reason	VARCHAR(255)	Optional description, varchar is suitable for this attribute.
SPORTS_COMPLEX_STAFF	staff_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	role	VARCHAR(255) NOT NULL	Roles are best defined as varchars. Each staff member must have a role.
	job_title	VARCHAR(255) NOT NULL	More specific title for the job. Title cannot be null.

	supervisor_id	INT FOREIGN KEY (self-reference)	Integer is suitable for ID’s. May be null if a staff member does not have a supervisor.
	overtime_hours	DECIMAL(5,2) DEFAULT 0	Hours of overtime; decimal for partial hours.
	user_id	INT NOT NULL FOREIGN KEY	Must be provided as the staff member has attributes linked to user.
ATTENDANCE RECORD	attendance_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Required, as attendance can only be taken if a user is involved.
	facility_id	INT NOT NULL FOREIGN KEY	Required, as attendance can only be taken if a facility is involved.
	check_in_time	DATETIME NOT NULL	Timestamp of check-in; cannot be null to track the time the user checked in.
	check_out_time	DATETIME	Timestamp of check-out; nullable if not yet checked out.
FEEDBACK	feedback_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	user_id	INT NOT NULL FOREIGN KEY	Feedback can only be provided by a user.
	facility_id	INT NOT NULL FOREIGN KEY	Feedback has to be related to a certain facility.

	date_submitted	DATETIME NOT NULL	Needed to track the time when the feedback was submitted.
	feedback_type	ENUM('Complaint','Suggestion','Praise') NOT NULL	Limited feedback categories; feedback has to fall under one of these categories.
	description	TEXT NOT NULL	Detailed feedback. Must be provided to send the feedback.
	status	ENUM('Open','In Progress','Closed') DEFAULT 'Open' NOT NULL	Limited status options. A status has to exist for a feedback.
EQUIPMENT	equipment_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	facility_id	INT NOT NULL FOREIGN KEY	Equipment has to be part of a facility.
	name	VARCHAR(255) NOT NULL	Equipment name should be a varchar; It must have a name.
	status	ENUM('Operational','Maintenance','Out of Service') NOT NULL	Limited status options; equipment must have a status.
	purchase_date	DATE NOT NULL	Date of purchase; must be noted to keep a record of the order.
	last_maintenance_date	DATE	Date of maintenance must be recorded as a date; may be null if it never went through maintenance.

EQUIPMENT_TYPE	equipment_type_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	name	VARCHAR(255) NOT NULL	Names are best stored as varchars; cannot be null as it must have a name.
EQUIPMENT_TYPE_VENDOR	equipment_type_vendor_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	equipment_type_id	INT NOT NULL FOREIGN KEY	Required to form this relationship.
	vendor_id	INT NOT NULL FOREIGN KEY	Required to form this relationship.
MAINTENANCEREQUEST	request_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	equipment_id	INT NOT NULL FOREIGN KEY	Request can only be made if an equipment needs maintenance.
	user_id	INT NOT NULL FOREIGN KEY	Request has to be made by a user.
	report_date	DATETIME NOT NULL	Date and Time of issue must be recorded in DATETIME format; It cannot be null as it is important to keep track of reported date and time.
	issue_description	TEXT NOT NULL	Description can be of variable length; it must be described for the request.

	status	ENUM('Open','Assigned','Resolved','Closed') NOT NULL	Limited status options; Request must have a status.
	resolution_date	DATETIME	Date for being resolved; may be null if it hasn’t been resolved
	staff_id	INT NOT NULL	A staff member must be assigned to resolve the issue.
PROCUREMEN TORDER	order_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification
	vendor_id	INT NOT NULL FOREIGN KEY	Order must involve a vendor; cannot be null.
	equipment_id	INT NOT NULL FOREIGN KEY	Order must involve an equipment; cannot be null.
	order_date	DATE NOT NULL	Date that the order was placed on must be recorded as a DATE; must not be null to keep the record of order.
	expected_delivery_ date	DATE	DATE type for the date order is expected to be delivered on
	status	ENUM('Pending','Delivered','Cancelled') NOT NULL	Limited status options; order must have a status.
	cost	DECIMAL(10,2) NOT NULL	Currency can be up to 2 decimal places for accuracy. Ordering an equipment must have a cost so it cannot be null.

VENDOR	vendor_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	name	VARCHAR(255) NOT NULL	Name must be saved as a varchar; vendor must have a name so it cannot be null.
	service_type	VARCHAR(255)	Brief elaboration of the service_type provided by the vendor.
VENDOR_CONTACT	contact_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	vendor_id	INT NOT NULL FOREIGN KEY	Contact can only exist if there is a vendor.
	contact_type	ENUM('Email','Phone','Fax','Other') NOT NULL	Limited contact options; contact must have a type.
	contact_value	VARCHAR(255) NOT NULL	Contact can be an email or phone number which may include special characters; Contact must exist.
TOKEN	token_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	serial_number	VARCHAR(255) NOT NULL	Serial number must be a varchar as it can have letters or ‘#’; Cannot be null as each token must have a serial number.
	game_number	INT NOT NULL	The game number should be an integer as it is numerical.

	issue_date	DATETIME NOT NULL	Has to be saved as a DATETIME; must be saved to keep the record of the time when the token was issued.
	user_id	INT NOT NULL FOREIGN KEY	Token must be issued for a user.
VIOLATION	violation_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	action_taken	ENUM('Warning','Fine','Suspension','Other') NOT NULL	Limited action options; Violation must have an action taken against it.
	status	ENUM('Reported','Under Review','Resolved') NOT NULL	Limited status options; Violation must have a status;
	reported_by	VARCHAR(255) NOT NULL	Name of the person it has been reported by; Violation must be reported by someone.
	description	VARCHAR(255) NOT NULL	Violation must be described as a varchar; There must be a description for violation.
	severity_level	ENUM('Low','Medium','High','Critical') NOT NULL	Limited severity options; must have a severity;
INCIDENT	incident_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	date_reported	DATETIME NOT NULL	DATETIME to record when the incident occurred; Must be noted to investigate into the matter according to the time.

	description	TEXT NOT NULL	Detailed description in text form; There must be description for it;
	violation_id	INT FOREIGN KEY	It can be a violation but not necessarily, not all incidents are violations.
USER_INCIDENT	incident_id	INT NOT NULL FOREIGN KEY	Required to make this relationship.
	user_id	INT NOT NULL FOREIGN KEY	Required to make this relationship.
EVENTBOOKING	booking_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	date	DATE NOT NULL	Must be saved as a DATE; Must be noted as an event must have a date.
	start_time	TIME NOT NULL	Start time has to be recorded as time; Event must have a time it starts at.
	end_time	TIME NOT NULL	End time has to be recorded as time; Event must have a time it ends at.
	booking_status	ENUM('Booked','Cancelled','Completed') NOT NULL	Limited status options; Booking must have a status.
TIMESLOT	slot_id	INT AUTO_INCREMENT PRIMARY KEY	Simple integer ID which is good for fast indexing, auto-incremented for unique user identification.
	facility_id	INT NOT NULL FOREIGN KEY	Timeslot must be related to a facility.

	start_time	TIME NOT NULL	Must be saved as TIME; cannot be null as a slot must have a start time.
	end_time	TIME NOT NULL	Must be saved as TIME; cannot be null as a slot must have an end time;
	membership_id	INT NOT NULL FOREIGN KEY	Timeslot must be catering members to fulfill its purpose.

5. Normalization Strategy

We normalized the database to 3NF but left it there. Beyond 3NF would result in creating many more entities. That would complicate the design. In actual systems, too many entities imply more joins. More joins make queries slower and burn more resources.

Change Log Table 1NF

Table Name	Field	Original Value	New Value	Reason for Change
USER	department	VARCHAR(255)	department_id (INT, FK)	To establish proper relationship with DEPARTMENT table
USER	medical_info	ENUM	[Removed]	Non-atomic data, moved to USER_MEDICAL_INFO table
USER_MEDICAL_INFO	[New Table]	-	Created new table	To store atomic medical information per user
FACILITY	operating_hours	TEXT	[Removed]	Non-atomic data, already represented in FACILITYSCHEDULE table
STAFF	working_hours	VARCHAR(255)	[Removed]	Non-atomic data, moved to STAFF_WORKING_SCHEDULE table
STAFF	days_off	VARCHAR(255)	[Removed]	Non-atomic data, moved to STAFF_DAYS_OFF table
STAFF_WORKING_SCHEDULE	[New Table]	-	Created new table	To store atomic working hours information

STAFF_DAYS_OFF	[New Table]	-	Created new table	To store atomic days off information
FAMILYMEMBER	familymember_id	[Not present]	Added as PK	To ensure every table has a dedicated primary key
EVENTBOOKING	time_slot	VARCHAR(255)	Replaced with start_time & end_time	To ensure atomic time values
MEMBERSHIPFREEZE	freeze_id	[Not present]	Added as PK	To ensure every table has a dedicated primary key
ALUMNI	alumni_id	[Not present]	Added as PK	To ensure every table has a dedicated primary key

SCHEMA in 1NF:

Entity: USER

- user_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- dob (DATE)
- gender (ENUM)
- email (VARCHAR(255))
- phone (VARCHAR(255))
- cnic (VARCHAR(255))
- user_type_id (INT, Foreign Key to USERTYPE)
- membership_status (ENUM)
- address (TEXT)
- department_id (INT, Foreign Key to DEPARTMENT)
- emergency_contact (VARCHAR(255))

Entity: USER_MEDICAL_INFO

- user_medical_info_id (INT, Primary Key)

- user_id (INT, Foreign Key to USER)
- medical_condition (VARCHAR(255))
- details (TEXT)

Entity: USERTYPE

- user_type_id (INT, Primary Key)
- type_name (VARCHAR(255))

Entity: MEMBERSHIP

- membership_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)
- start_date (DATE)
- end_date (DATE)
- status (ENUM)
- freeze_months_used (INT)
- last_intimation_date (DATE)

Entity: MEMBERSHIPTYPE

- membership_type_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))
- registration_fee (DECIMAL(10,2))
- monthly_fee (DECIMAL(10,2))

Entity: MEMBERSHIPCARD

- membership_id (INT, Foreign Key to MEMBERSHIP)
- issued_date (DATE)
- surrendered_date (DATE)
- lost_replacement (TINYINT(1))

Entity: MEMBERSHIPFREEZE

- freeze_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- start_date (DATE)
- end_date (DATE)
- request_date (DATE)

Entity: FACILITY

- facility_id (INT, Primary Key)
- name (VARCHAR(255))
- location (TEXT)

Entity: FACILITYSCHEDULE

- schedule_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- day_of_week (ENUM)
- open_time (TIME)
- close_time (TIME)

Entity: TIMESLOT

- slot_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- start_time (TIME)
- end_time (TIME)
- max_capacity (INT)

Entity: BOOKING

- booking_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- slot_id (INT, Foreign Key to TIMESLOT)
- booking_date (DATE)
- duration_mins (INT)

Entity: EVENTBOOKING

- booking_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- user_id (INT, Foreign Key to USER)
- date (DATE)
- start_time (TIME)
- end_time (TIME)
- booking_status (ENUM)

Entity: ATTENDANCERECORD

- attendance_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- check_in_time (TIMESTAMP)

- check_out_time (TIMESTAMP)
- recorded_by (INT, Foreign Key to STAFF)

Entity: EQUIPMENT

- equipment_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))
- status (ENUM)
- purchase_date (DATE)
- last_maintenance_date (DATE)

Entity: MAINTENANCEREQUEST

- request_id (INT, Primary Key)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- reported_by (INT, Foreign Key to USER)
- report_date (DATE)
- issue_description (TEXT)
- status (ENUM)
- resolution_date (DATE)
- handled_by (INT, Foreign Key to STAFF)

Entity: PROCUREMENTORDER

- order_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- order_date (DATE)
- expected_delivery_date (DATE)
- status (ENUM)
- cost (DECIMAL(10,2))

Entity: VENDOR

- vendor_id (INT, Primary Key)
- name (VARCHAR(255))
- service_type (VARCHAR(255))
- contact_info (TEXT)

Entity: FEEDBACK

- feedback_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)

- facility_id (INT, Foreign Key to FACILITY)
- date_submitted (DATE)
- feedback_type (ENUM)
- description (TEXT)
- status (ENUM)

Entity: INCIDENT

- incident_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- date_reported (DATE)
- description (TEXT)

Entity: VIOLATIONTYPE

- violation_type_id (INT, Primary Key)
- name (VARCHAR(255))
- description (TEXT)
- action_taken (ENUM)
- severity_level (ENUM)
- status (ENUM)
- reported_by (VARCHAR(255))

Entity: PAYMENT

- payment_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- payment_date (DATE)
- amount (DECIMAL(10,2))
- payment_type (ENUM)
- status (ENUM)
- is_fee_charged (TINYINT(1))

Entity: STAFF

- staff_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- role (VARCHAR(255))
- contact (VARCHAR(255))
- job_title (VARCHAR(255))
- supervisor_id (INT, Self-reference to STAFF)
- overtime_hours (INT)

Entity: STAFF_WORKING_SCHEDULE

- schedule_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- day_of_week (ENUM)
- start_time (TIME)
- end_time (TIME)

Entity: STAFF_DAYS_OFF

- days_off_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- date (DATE)
- reason (VARCHAR(255))

Entity: DEPARTMENT

- department_id (INT, Primary Key)
- name (VARCHAR(255))
- head_staff_id (INT, Foreign Key to STAFF)
- contact_email (VARCHAR(255))

Entity: STUDENT

- cms_id (VARCHAR(255), Primary Key)
- user_id (INT, Foreign Key to USER)
- ug_pg_status (VARCHAR(255))
- hostelOrDayscholar (VARCHAR(255))
- program (VARCHAR(255))
- year (INT)

Entity: TOKEN

- token_id (INT, Primary Key)
- cms_id (VARCHAR(255), Foreign Key to STUDENT)
- serial_number (VARCHAR(255))
- game_number (INT)
- issue_date (DATE)

Entity: ALUMNI

- alumni_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- graduation_year (INT)

Entity: FACULTYMEMBER

- facultymember_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- department (VARCHAR(255))
- rank (VARCHAR(255))

Entity: FAMILYMEMBER

- familymember_id (INT, Primary Key)
- staffid (INT, Foreign Key to STAFF)
- user_id (INT, Foreign Key to USER)

Entity: NSTPTENANT

- nstpid (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- tenant_start_date (DATE)
- tenant_end_date (DATE)

Entity: LOCKERRENTAL

- locker_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- issue_date (DATE)
- return_date (DATE)

Change Log Table 2NF

Table Name	Field	Original Value	New Value	Justification	Impact
VENDOR	contact_info	TEXT	[Removed]	The contact_info field contained non-atomic data that could include multiple contact points (phone, email, address) in a single text field. This violates 2NF principles as it's not fully dependent on the primary key and could lead to data redundancy and update anomalies.	Removed field entirely. Contact information now managed in the dedicated VENDOR_CONTACT table with proper structure.
VENDOR_CONTACT	[New Table]	-	Created new table with fields: contact_id (INT, PK), vendor_id (INT, FK), contact_type (ENUM), contact_value (VARCHAR(255)), is_primary (BOOLEAN)	Created to properly store vendor contact information in an atomic way. Each contact point (phone, email, address) now has its own record with a type indicator. This resolves the non-atomic data issue in the original VENDOR table and ensures 2NF compliance.	New table allows for multiple contacts per vendor with proper categorization and primary contact identification.

PROGRAM	[New Table]	-	Created new table with fields: program_id (INT, PK), program_name (VARCHAR(255)), department_id (INT, FK), level (ENUM), duration_years (INT)	The original schema had program information embedded in the STUDENT table, creating partial dependencies. By extracting this into a dedicated PROGRAM table, we ensure information about academic programs is stored once and referenced by students.	New table normalizes program data, avoiding redundancy and ensuring consistency across student records.
STUDENT	program	VARCHAR(255)	program_id (INT, FK)	The program field was storing the name of the academic program as text, potentially leading to inconsistencies and redundancy. Converting it to a foreign key reference ensures data integrity and proper normalization.	Field now references program_id in the new PROGRAM table, establishing a proper relationship.
STUDENT	ug_pg_status	VARCHAR(255)	[Removed]	This field indicated undergraduate/postgraduate status but is actually an attribute of the program itself, not directly of the student. This created a partial dependency where attributes were dependent on part of the composite key.	Field removed as this information is now stored in the PROGRAM table's level field, eliminating redundancy.

FACULTYMEMBER	department	VARCHAR(255)	department_id (INT, FK)	The department was stored as text, creating potential inconsistencies with the DEPARTMENT table that already exists in the schema. This violated 2NF as it created a partial dependency.	Field changed to properly reference the department_id in the DEPARTMENT table, ensuring referential integrity and proper normalization.

2NF Schema:

Entity: USER

- user_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- dob (DATE)
- gender (ENUM)
- email (VARCHAR(255))
- phone (VARCHAR(255))
- cnic (VARCHAR(255))
- user_type_id (INT, Foreign Key to USERTYPE)
- membership_status (ENUM)
- address (TEXT)
- department_id (INT, Foreign Key to DEPARTMENT)
- emergency_contact (VARCHAR(255))

Entity: USER_MEDICAL_INFO

- user_medical_info_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- medical_condition (VARCHAR(255))
- details (TEXT)

Entity: USERTYPE

- user_type_id (INT, Primary Key)
- type_name (VARCHAR(255))

Entity: MEMBERSHIP

- membership_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)
- start_date (DATE)
- end_date (DATE)
- status (ENUM)
- freeze_months_used (INT)
- last_intimation_date (DATE)

Entity: MEMBERSHIPTYPE

- membership_type_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))
- registration_fee (DECIMAL(10,2))
- monthly_fee (DECIMAL(10,2))

Entity: MEMBERSHIPCARD

- membership_id (INT, Foreign Key to MEMBERSHIP)
- issued_date (DATE)
- surrendered_date (DATE)
- lost_replacement (TINYINT(1))

Entity: MEMBERSHIPFREEZE

- freeze_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- start_date (DATE)
- end_date (DATE)
- request_date (DATE)

Entity: FACILITY

- facility_id (INT, Primary Key)
- name (VARCHAR(255))
- location (TEXT)

Entity: FACILITYSCHEDULE

- schedule_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)

- day_of_week (ENUM)
- open_time (TIME)
- close_time (TIME)

Entity: TIMESLOT

- slot_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- start_time (TIME)
- end_time (TIME)
- max_capacity (INT)

Entity: BOOKING

- booking_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- slot_id (INT, Foreign Key to TIMESLOT)
- booking_date (DATE)
- duration_mins (INT)

Entity: EVENTBOOKING

- booking_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- user_id (INT, Foreign Key to USER)
- date (DATE)
- start_time (TIME)
- end_time (TIME)
- booking_status (ENUM)

Entity: ATTENDANCERECORD

- attendance_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- check_in_time (TIMESTAMP)
- check_out_time (TIMESTAMP)
- recorded_by (INT, Foreign Key to STAFF)

Entity: EQUIPMENT

- equipment_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))

- status (ENUM)
- purchase_date (DATE)
- last_maintenance_date (DATE)

Entity: MAINTENANCEREQUEST

- request_id (INT, Primary Key)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- reported_by (INT, Foreign Key to USER)
- report_date (DATE)
- issue_description (TEXT)
- status (ENUM)
- resolution_date (DATE)
- handled_by (INT, Foreign Key to STAFF)

Entity: PROCUREMENTORDER

- order_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- order_date (DATE)
- expected_delivery_date (DATE)
- status (ENUM)
- cost (DECIMAL(10,2))

Entity: VENDOR

- vendor_id (INT, Primary Key)
- name (VARCHAR(255))
- service_type (VARCHAR(255))

Entity: VENDOR_CONTACT

- contact_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- contact_type (ENUM)
- contact_value (VARCHAR(255))
- is_primary (BOOLEAN)

Entity: FEEDBACK

- feedback_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)

- date_submitted (DATE)
- feedback_type (ENUM)
- description (TEXT)
- status (ENUM)

Entity: INCIDENT

- incident_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- date_reported (DATE)
- description (TEXT)

Entity: VIOLATIONTYPE

- violation_type_id (INT, Primary Key)
- name (VARCHAR(255))
- description (TEXT)
- action_taken (ENUM)
- severity_level (ENUM)
- status (ENUM)
- reported_by (VARCHAR(255))

Entity: PAYMENT

- payment_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- payment_date (DATE)
- amount (DECIMAL(10,2))
- payment_type (ENUM)
- status (ENUM)
- is_fee_charged (TINYINT(1))

Entity: STAFF

- staff_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- role (VARCHAR(255))
- contact (VARCHAR(255))
- job_title (VARCHAR(255))
- supervisor_id (INT, Self-reference to STAFF)
- overtime_hours (INT)

Entity: STAFF_WORKING_SCHEDULE

- schedule_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- day_of_week (ENUM)
- start_time (TIME)
- end_time (TIME)

Entity: STAFF_DAYS_OFF

- days_off_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- date (DATE)
- reason (VARCHAR(255))

Entity: DEPARTMENT

- department_id (INT, Primary Key)
- name (VARCHAR(255))
- head_staff_id (INT, Foreign Key to STAFF)
- contact_email (VARCHAR(255))

Entity: PROGRAM

- program_id (INT, Primary Key)
- program_name (VARCHAR(255))
- department_id (INT, Foreign Key to DEPARTMENT)
- level (ENUM)
- duration_years (INT)

Entity: STUDENT

- cms_id (VARCHAR(255), Primary Key)
- user_id (INT, Foreign Key to USER)
- program_id (INT, Foreign Key to PROGRAM)
- hostelOrDayscholar (VARCHAR(255))
- year (INT)

Entity: TOKEN

- token_id (INT, Primary Key)
- cms_id (VARCHAR(255), Foreign Key to STUDENT)
- serial_number (VARCHAR(255))
- game_number (INT)
- issue_date (DATE)

Entity: ALUMNI

- alumni_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- graduation_year (INT)

Entity: FACULTYMEMBER

- facultymember_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- department_id (INT, Foreign Key to DEPARTMENT)
- rank (VARCHAR(255))

Entity: FAMILYMEMBER

- familymember_id (INT, Primary Key)
- staffid (INT, Foreign Key to STAFF)
- user_id (INT, Foreign Key to USER)

Entity: NSTPTENANT

- nstpid (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- tenant_start_date (DATE)
- tenant_end_date (DATE)

Entity: LOCKERRENTAL

- locker_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- issue_date (DATE)
- return_date (DATE)

3NF Changes Log

Entity	Change Made	Justification	3NF Violation Fixed
MEMBERSHIPTYPE	Removed facility_id foreign key	In 2NF, a membership type had a direct link to a facility. This creates a transitive dependency, as a membership type can be associated with multiple facilities.	Removed transitive dependency where non-key attribute facility_id depends on another non-key attribute (membership type fields).
EQUIPMENT_TYPE	Created new entity to separate equipment categories	Equipment types can have relationships with multiple vendors	Normalized many-to-many relationship between equipment types and vendors
EQUIPMENT_TYPE_VENDOR	Created junction table for the many-to-many relationship	Links equipment types to vendors	Properly represents many-to-many relationship between equipment types and vendors

3NF Schema

Entity: USER

- user_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- dob (DATE)
- gender (ENUM)
- email (VARCHAR(255))
- phone (VARCHAR(255))
- cnic (VARCHAR(255))
- user_type_id (INT, Foreign Key to USERTYPE)
- membership_status (ENUM)
- address (TEXT)
- department_id (INT, Foreign Key to DEPARTMENT)
- emergency_contact (VARCHAR(255))

Entity: USER_MEDICAL_INFO

- user_medical_info_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- medical_condition (VARCHAR(255))
- details (TEXT)

Entity: USERTYPE

- user_type_id (INT, Primary Key)
- type_name (VARCHAR(255))

Entity: MEMBERSHIP

- membership_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)
- start_date (DATE)
- end_date (DATE)
- status (ENUM)
- freeze_months_used (INT)
- last_intimation_date (DATE)

Entity: MEMBERSHIPTYPE

- membership_type_id (INT, Primary Key)
- name (VARCHAR(255))
- registration_fee (DECIMAL(10,2))
- monthly_fee (DECIMAL(10,2))

Entity: FACILITY_MEMBERSHIPTYPE

- facility_membershiptype_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)

Entity: MEMBERSHIPCARD

- membership_id (INT, Foreign Key to MEMBERSHIP)
- issued_date (DATE)
- surrendered_date (DATE)
- lost_replacement (TINYINT(1))

Entity: MEMBERSHIPFREEZE

- freeze_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- start_date (DATE)
- end_date (DATE)
- request_date (DATE)

Entity: FACILITY

- facility_id (INT, Primary Key)
- name (VARCHAR(255))
- location (TEXT)

Entity: FACILITYSCHEDULE

- schedule_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- day_of_week (ENUM)
- open_time (TIME)
- close_time (TIME)

Entity: TIMESLOT

- slot_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- start_time (TIME)
- end_time (TIME)
- max_capacity (INT)

Entity: BOOKING

- booking_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- slot_id (INT, Foreign Key to TIMESLOT)
- booking_date (DATE)
- duration_mins (INT)

Entity: EVENTBOOKING

- booking_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- user_id (INT, Foreign Key to USER)
- date (DATE)
- start_time (TIME)
- end_time (TIME)
- booking_status (ENUM)

Entity: ATTENDANCERECORD

- attendance_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- check_in_time (TIMESTAMP)
- check_out_time (TIMESTAMP)
- recorded_by (INT, Foreign Key to STAFF)

Entity: EQUIPMENT

- equipment_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))
- status (ENUM)

- purchase_date (DATE)
- last_maintenance_date (DATE)

Entity: EQUIPMENT_TYPE

- equipment_type_id (INT, Primary Key)
- name (VARCHAR(255))

Entity: EQUIPMENT_TYPE_VENDOR

- equipment_type_vendor_id (INT, Primary Key)
- equipment_type_id (INT, Foreign Key to EQUIPMENT_TYPE)
- vendor_id (INT, Foreign Key to VENDOR)

Entity: MAINTENANCEREQUEST

- request_id (INT, Primary Key)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- reported_by (INT, Foreign Key to USER)
- report_date (DATE)
- issue_description (TEXT)
- status (ENUM)
- resolution_date (DATE)
- handled_by (INT, Foreign Key to STAFF)

Entity: PROCUREMENTORDER

- order_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- order_date (DATE)
- expected_delivery_date (DATE)
- status (ENUM)
- cost (DECIMAL(10,2))

Entity: VENDOR

- vendor_id (INT, Primary Key)
- name (VARCHAR(255))
- service_type (VARCHAR(255))

Entity: VENDOR_CONTACT

- contact_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- contact_type (ENUM)
- contact_value (VARCHAR(255))
- is_primary (BOOLEAN)

Entity: FEEDBACK

- feedback_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- date_submitted (DATE)
- feedback_type (ENUM)
- description (TEXT)
- status (ENUM)

Entity: INCIDENT

- incident_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- date_reported (DATE)
- description (TEXT)

Entity: VIOLATION

- violation_type_id (INT, Primary Key)
- name (VARCHAR(255))
- description (TEXT)
- severity_level (ENUM)
- action_taken (ENUM)
- status (ENUM)
- reported_by (VARCHAR(255))

Entity: PAYMENT

- payment_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)

- payment_date (DATE)
- amount (DECIMAL(10,2))
- payment_type (ENUM)
- status (ENUM)
- is_fee_charged (TINYINT(1))

Entity: STAFF

- staff_id (INT, Primary Key)
- first_name (VARCHAR(255))
- last_name (VARCHAR(255))
- role (VARCHAR(255))
- contact (VARCHAR(255))
- job_title (VARCHAR(255))
- supervisor_id (INT, Self-reference to STAFF)
- overtime_hours (INT)

Entity: STAFF_WORKING_SCHEDULE

- schedule_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- day_of_week (ENUM)
- start_time (TIME)
- end_time (TIME)

Entity: STAFF_DAYS_OFF

- days_off_id (INT, Primary Key)
- staff_id (INT, Foreign Key to STAFF)
- date (DATE)
- reason (VARCHAR(255))

Entity: DEPARTMENT

- department_id (INT, Primary Key)
- name (VARCHAR(255))
- head_staff_id (INT, Foreign Key to STAFF)
- contact_email (VARCHAR(255))

Entity: PROGRAM

- program_id (INT, Primary Key)
- program_name (VARCHAR(255))
- department_id (INT, Foreign Key to DEPARTMENT)
- level (ENUM)
- duration_years (INT)

Entity: STUDENT

- cms_id (VARCHAR(255), Primary Key)
- user_id (INT, Foreign Key to USER)
- program_id (INT, Foreign Key to PROGRAM)
- hostelOrDayscholar (VARCHAR(255))
- year (INT)

Entity: TOKEN

- token_id (INT, Primary Key)
- cms_id (VARCHAR(255), Foreign Key to STUDENT)
- serial_number (VARCHAR(255))
- game_number (INT)
- issue_date (DATE)

Entity: ALUMNI

- alumni_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- graduation_year (INT)

Entity: FACULTYMEMBER

- facultymember_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- department_id (INT, Foreign Key to DEPARTMENT)
- rank (VARCHAR(255))

Entity: FAMILYMEMBER

- familymember_id (INT, Primary Key)
- staffid (INT, Foreign Key to STAFF)
- user_id (INT, Foreign Key to USER)

Entity: NSTPTENANT

- nstpid (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- tenant_start_date (DATE)
- tenant_end_date (DATE)

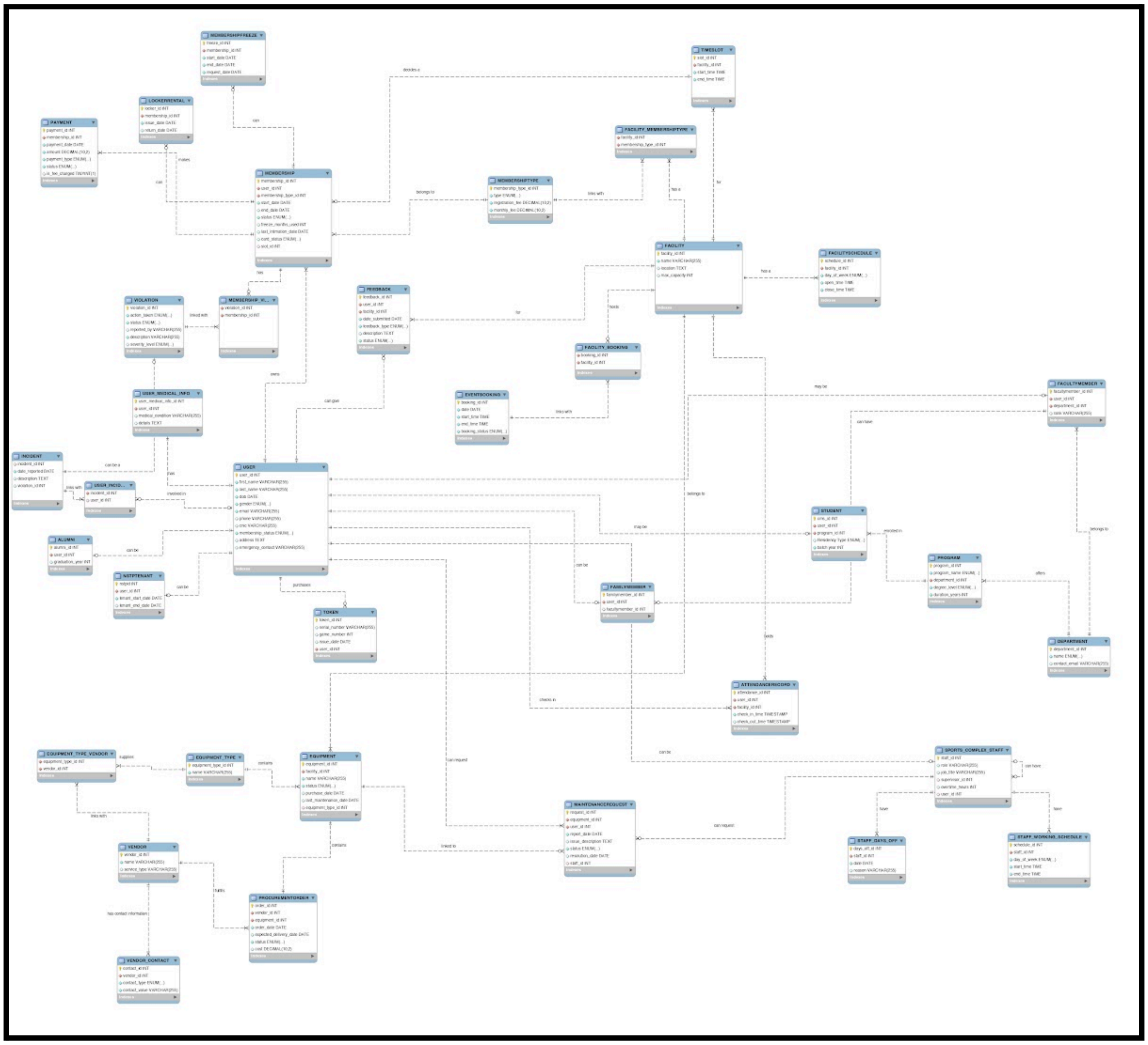
Entity: LOCKERRENTAL

- locker_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- issue_date (DATE)
- return_date (DATE)

6. ER Diagram and Schema Design

6.1 ER Diagram

(click to zoom)



6.2 ERD Schema

ALUMNI

- alumni_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- graduation_year (YEAR)

ATTENDANCERECORD

- attendance_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- check_in_time (DATETIME)
- check_out_time (DATETIME)

DEPARTMENT

- department_id (INT, Primary Key)
- name (VARCHAR(255))
- contact_email (VARCHAR(255))

EQUIPMENT

- equipment_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- name (VARCHAR(255))

- status (ENUM: 'Functional', 'Under Maintenance', 'Out of Service', 'Working')
- purchase_date (DATE)
- last_maintenance_date (DATE)
- equipment_type_id (INT, Foreign Key to EQUIPMENT_TYPE)

EQUIPMENT_TYPE

- equipment_type_id (INT, Primary Key)
- name (VARCHAR(255))

EQUIPMENT_TYPE_VENDOR

- equipment_type_vendor_id (INT, Primary Key)
- equipment_type_id (INT, Foreign Key to EQUIPMENT_TYPE)
- vendor_id (INT, Foreign Key to VENDOR)

EVENTBOOKING

- booking_id (INT, Primary Key)
- date (DATE)
- start_time (TIME)
- end_time (TIME)
- booking_status (ENUM: 'Booked', 'Cancelled', 'Completed')

FACILITY

- facility_id (INT, Primary Key)
- name (VARCHAR(255))
- location (TEXT)
- max_capacity (INT)

FACILITY_BOOKING

- booking_id (INT, Foreign Key to EVENTBOOKING)
- facility_id (INT, Foreign Key to FACILITY)

FACILITY_MEMBERSHIPTYPE

- facility_membershiptype_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)

FACILITYSCHEDULE

- schedule_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- day_of_week (ENUM: 'Monday' to 'Sunday')
- open_time (TIME)
- close_time (TIME)

FACULTYMEMBER

- facultymember_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- department_id (INT, Foreign Key to DEPARTMENT)
- rank (VARCHAR(255))

FAMILYMEMBER

- familymember_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facultymember_id (INT, Foreign Key to FACULTYMEMBER)

FEEDBACK

- feedback_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- facility_id (INT, Foreign Key to FACILITY)
- date_submitted (DATETIME)
- feedback_type (ENUM: 'Complaint', 'Suggestion', 'Praise', 'Other')
- description (TEXT)
- status (ENUM: 'New', 'Reviewed', 'Closed')

INCIDENT

- incident_id (INT, Primary Key)
- date_reported (DATETIME)
- description (TEXT)
- violation_id (INT, Foreign Key to VIOLATION)

LOCKERRENTAL

- locker_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- issue_date (DATE)
- return_date (DATE)

MAINTENANCEREQUEST

- request_id (INT, Primary Key)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- user_id (INT, Foreign Key to USER)
- report_date (DATETIME)
- issue_description (TEXT)
- status (ENUM: 'Open', 'In Progress', 'Closed')
- resolution_date (DATE)
- staff_id (INT, Foreign Key to SPORTS_COMPLEX_STAFF)

MEMBERSHIP

- membership_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- membership_type_id (INT, Foreign Key to MEMBERSHIPTYPE)
- start_date (DATE)
- end_date (DATE)
- status (ENUM: 'Active', 'Suspended', 'Expired')
- freeze_months_used (INT)
- last_intimation_date (DATE)
- card_status (VARCHAR(50))
- slot_id (INT, Foreign Key to TIMESLOT)

MEMBERSHIP_VIOLATION

- violation_id (INT, Foreign Key to VIOLATION)
- membership_id (INT, Foreign Key to MEMBERSHIP)

MEMBERSHIPFREEZE

- freeze_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- start_date (DATE)
- end_date (DATE)

- request_date (DATE)

MEMBERSHIPTYPE

- membership_type_id (INT, Primary Key)
- type (VARCHAR(50))
- registration_fee (DECIMAL(10,2))
- monthly_fee (DECIMAL(10,2))

NSTPTENANT

- nstpid (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- tenant_start_date (DATE)
- tenant_end_date (DATE)

PAYMENT

- payment_id (INT, Primary Key)
- membership_id (INT, Foreign Key to MEMBERSHIP)
- payment_date (DATETIME)
- amount (DECIMAL(10,2))
- payment_type (ENUM: 'Cash', 'Credit', 'Debit', 'Online')
- status (ENUM: 'Pending', 'Completed', 'Failed', 'Done')

- is_fee_charged (TINYINT(1))

PROCUREMENTORDER

- order_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- equipment_id (INT, Foreign Key to EQUIPMENT)
- order_date (DATE)
- expected_delivery_date (DATE)
- status (ENUM: 'Ordered', 'Received', 'Cancelled')
- cost (DECIMAL(10,2))

PROGRAM

- program_id (INT, Primary Key)
- program_name (ENUM with multiple program names)
- department_id (INT, Foreign Key to DEPARTMENT)
- degree_level (VARCHAR(50))
- duration_years (TINYINT)

SPORTS_COMPLEX_STAFF

- staff_id (INT, Primary Key)
- role (VARCHAR(255))

- job_title (VARCHAR(255))
- supervisor_id (INT, Foreign Key to SPORTS_COMPLEX_STAFF)
- overtime_hours (DECIMAL(5,2))
- user_id (INT, Foreign Key to USER)

STAFF_DAYS_OFF

- days_off_id (INT, Primary Key)
- staff_id (INT, Foreign Key to SPORTS_COMPLEX_STAFF)
- date (DATE)
- reason (VARCHAR(255))

STAFF_WORKING_SCHEDULE

- schedule_id (INT, Primary Key)
- staff_id (INT, Foreign Key to SPORTS_COMPLEX_STAFF)
- day_of_week (ENUM: 'Monday' to 'Sunday')
- start_time (TIME)
- end_time (TIME)

STUDENT

- cms_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)

- program_id (INT, Foreign Key to PROGRAM)
- Residency Type (ENUM: 'Hostellite', 'Day Scholor', 'Day Scholar')
- batch year (YEAR)

TIMESLOT

- slot_id (INT, Primary Key)
- facility_id (INT, Foreign Key to FACILITY)
- start_time (TIME)
- end_time (TIME)

TOKEN

- token_id (INT, Primary Key)
- serial_number (VARCHAR(255))
- game_number (INT)
- issue_date (DATE)
- user_id (INT, Foreign Key to USER)

USER

- user_id (INT, Primary Key)
- first_name (VARCHAR(255))

- last_name (VARCHAR(255))
- dob (DATE)
- gender (ENUM: 'Male', 'Female', 'Other')
- email (VARCHAR(255))
- phone (VARCHAR(255))
- cnic (VARCHAR(255))
- membership_status (ENUM: 'Active', 'Inactive', 'Suspended')
- address (TEXT)
- emergency_contact (VARCHAR(255))

USER_INCIDENT

- incident_id (INT, Foreign Key to INCIDENT)
- user_id (INT, Foreign Key to USER)

USER_MEDICAL_INFO

- user_medical_info_id (INT, Primary Key)
- user_id (INT, Foreign Key to USER)
- medical_condition (VARCHAR(255))
- details (TEXT)

VENDOR

- vendor_id (INT, Primary Key)
- name (VARCHAR(255))
- service_type (VARCHAR(255))

VENDOR_CONTACT

- contact_id (INT, Primary Key)
- vendor_id (INT, Foreign Key to VENDOR)
- contact_type (ENUM: 'Email', 'Phone', 'Fax', 'Other')
- contact_value (VARCHAR(255))

VIOLATION

- violation_id (INT, Primary Key)
- action_taken (ENUM: 'Warning', 'Suspension', 'Termination', 'Other')
- status (ENUM: 'Open', 'Resolved', 'Closed')
- reported_by (VARCHAR(255))
- description (VARCHAR(255))
- severity_level (ENUM: 'Information', 'Warning', 'Critical')

6.3 Schema Justification

Relationships between entities

Entities	Relationship	Explanation
User to Alumni	one to zero or one	Each alumni record references exactly one user, but not every user has an alumni record.
User to Facultymember	one to zero or one	Every facultymember entry ties to one user, yet most users are not faculty, so a user may have none.
User to Familymember	one to zero or one	Every familymember references one user, but not every user references a familymember.
User to Membership	one to zero or many	Each membership belongs to one user, but a user can hold multiple memberships over time or none.
User to Nstptenant	one to zero or one	The nstptenant record is for one user only, and most users won't have any record for nstptenant.
User to Attendancerecord	one to zero or many	Each attendance record logs one user's check-in/out, and a user may have many or no records.
User to Maintenancerequest	one to zero or many	Maintenance requests have to be filed by a user, but a user can submit many requests or none.
User to Feedback	one to zero or many	Each Feedback entry is tied to one user, and a user may submit multiple feedbacks or none.
User to Token	one to zero or many	Each token issued belongs to one user, but a user can hold many tokens or none.

User to Usermedicalinfo	one to zero or many	Each medical info entry is for one user, and users may have multiple entries or none.
User to Student	one to zero or one	Each student is a user in this case, but not every user is a student.
Student to Program	many to one	Each student enrolls in one program, but a program can have many students.
Program to Department	many to one	Each program belongs to one department, while a department may offer multiple programs.
Membership to Membershiptype	many to one	Every membership refers to one membershiptype, and each type can be used by many memberships.
Facilitymembershiptype to Facility	many to one	Each facilitymembershiptype record must have one facility, but a single facility can be in multiple records.
Facilitymembershiptype to Membershiptype	many to one	Each facilitymembershiptype record must have one membershiptype, but a single membershiptype can be in multiple records.
Membership to Lockerrental	one to zero or many	Each lockerrental ties to a membership, but a membership may rent zero or several lockers.
Membership to Membershipfreeze	one to zero or many	Each freeze entry belongs to one membership, and a membership can be frozen multiple times or never.
Membership to Payment	one to zero or many	Payments are recorded per membership, and a membership can have several payments or none.

Membership to Timeslot	Zero or Many to one	A time slot can be assigned to Zero or Many memberships, but one membership must have only one time slot.
Membership to Membershipviolation	one to zero or many	Each membershipviolation record must have one membership, but a single membership can be in multiple records.
Facility to Attendancerecord	one to zero or many	Each attendance log references one facility, and a facility can have many or no attendance records.
Facility to Feedback	one to zero or many	Each Feedback entry belongs to one facility, and a facility may receive many or no feedback entries.
Facility to Equipment	one to zero or many	Each equipment item is placed in one facility, and a facility can house many or no items.
Facility to Facilityschedule	one to zero or many	Each schedule entry belongs to only one facility, and a facility can have multiple or no schedules.
Facility to Timeslot	one to zero or many	Each Timeslot is for one facility, and a facility can offer many or no slots.
Facility to Facilitybooking	one to zero or many	Each booking record references one facility, and a facility can have many or no bookings.
Equipment to Maintenancerequest	one to zero or many	Each Maintenance request targets one equipment item, and an item can have many or no requests.

Equipmenttype to Equipmenttypevendor	one to zero or many	Each Equipmenttypevendor record must have one Equipmenttype, but a single Equipmenttype can be in multiple records.
Vendor to Equipmenttypevendor	one to zero or many	Each Equipmenttypevendor record must have one vendor, but a single vendor can be in multiple records.
Procurementorder to Equipment	one to one	Each Order is placed for one equipment item, and a certain item can appear only in one order.
Procurementorder to Vendor	many to one	Each procurement order goes to one vendor, and a vendor can fulfill many orders.
Vendor to Vendorcontact	one to many	Each contact entry references one vendor, and a vendor can have multiple contact records.
Facultymember to Department	many to one	Each faculty member is assigned to one department, while a department may include many faculty.
Familymember to Facultymember	Zero or many to one	Each Family member can be referenced to one faculty member, but one faculty member may have zero or many family members.
Sports_complex_staff to User	Zero or one to one	Each staff member references one user ,but one user may reference to zero or one staff member.
Sports_complex_staff to Staffdaysoff	one to zero or many	Each days-off entry belongs to one staff member, and a staff member can request many or no days off.

Sports_complex_staff to Staffworkingschedule	one to many	Each Working-schedule entry references one staff member, and a staff member can have many schedules.
Sports_complex_staff to Maintenancerequest	one to zero or many	Each Maintenance request may be assigned to one staff member, and a member can handle many or no requests.
Sports_complex_staff to Sports_complex_staff	One or Zero to Many	A staff member can be working under zero or one staff member (zero only if they work at the top), and a staff member can have many staff members working under them.
Incident to Violation	one to zero or one	Each incident cites zero or one violation, while a violation can be referenced to one incident.
User to Userincident	one to zero or many	Each UserIncident record must have one User, but a single User can be in multiple records.
Incident to Userincident	one to zero or many	Each UserIncident record must have one incident, but a single Incident can be in multiple records.
Membershipviolation to Violation	many to one	Each membershipviolation record must have one violation, but a single facility can be in multiple records.
Membershipviolation to Membership	Zero or many to one	Each membershipviolation record must have one membership, but a single membership can be in zero or many records.

Eventbooking to Facilitybooking	one to many	Each FacilityBooking record must have one event booking, but a single event booking can be in multiple records.
Facility to Facilitybooking	one to many or zero	Each FacilityBooking record must have one facility, but a single facility can be in zero or many records.

7. Relational Algebra Expressions

Query	Description	Expression	Functional Requirement(s)

RA001	Active memberships expiring before 2025-06-01	π user_id, membership_id (σ end_date < '2025-06-01' ∧ status = 'Active' (MEMBERSHIP))	F2
RA002	Users diagnosed with “Diabetes” who have active memberships	π user_id (σ medical_condition = 'Diabetes' (USER_MEDICAL_INFO) ⋈ σ status = 'Active' (MEMBERSHIP) ⋈ USER)	F1
RA003	Students in BSDS program for batch year 2024	π cms_id (σ program_id = (π program_id (σ program_name = 'BSDS' (PROGRAM))) ∧ batch_year = 2024 (STUDENT))	F18
RA004	Users with overdue payments (due date passed and status still Pending)	π membership_id (σ payment_date < today ∧ status = 'Pending' (PAYMENT) ⋈ MEMBERSHIP)	F3
RA005	Facility-specific attendance on Mondays between 08:00–10:00	π user_id, facility_id (σ day_of_week = 'Monday' ∧ check_in_time >= '08:00' ∧ check_in_time < '10:00' (ATTENDANCERECORD))	F4
RA006	Users who have both feedback and maintenance requests outstanding	π user_id (σ status = 'Open' (FEEDBACK)) ∩ π user_id (σ status = 'Pending' (MAINTENANCEREQUEST))	F9, F16
RA007	Alumni who have never used any facility in 2025	π user_id (ALUMNI) - π user_id (σ check_in_time >= ‘2025-01-01’(ATTENDANCERECORD))	F6
RA008	Trainers (role='Trainer') with no days-off scheduled	π staff_id (σ role = 'Trainer' (SPORTS_COMPLEX_STAFF)) - π staff_id (STAFF_DAYS_OFF)	F7, F19

RA009	Facilities that require maintenance this month	π facility_id (month(report_date)=month(today) (MAINTENANCEREQUEST))	F11
RA0010	Memberships that have never incurred a violation	π membership_id (MEMBERSHIP) - π membership_id (MEMBERSHIP_VIOLATION)	F8
RA0011	Equipment never serviced (no maintenance request)	π equipment_id (EQUIPMENT) - π equipment_id (MAINTENANCEREQUEST)	F8
RA0012	Vendor contacts for vendors supplying “Treadmill” equipment	π contact_value (σ name = 'Treadmill' (EQUIPMENT_TYPE) ⋈ EQUIPMENT_TYPE_VENDOR etv ⋈ VENDOR_CONTACT vc)	F10
RA0013	Users with at least two freezes recorded	π membership_id (σ freeze_months_used >= 2(MEMBERSHIP))	F2
RA0014	Find all tokens issued for the game number 2 at the bowling facility (facility_id = 1)	π user_id, serial_number (σ facility_id = 1 ∧ game_number = 2 (TOKEN))	F14
RA0015	Events scheduled but not yet completed	π booking_id, date (σ booking_status = 'Booked' (EVENTBOOKING))	F12,F15
RA0016	On-demand full history for user 1 (attendance, payments, feedback)	(π attendance_id (σ user_id = 1 (ATTENDANCERECORD))) ∪ (π payment_id (σ user_id = 1 (PAYMENT))) ∪ (π feedback_id (σ user_id = 1 (FEEDBACK)))	F17

RA0017	Find users who visited during both peak hours (17:00-18:00 and 18:00-19:00)	π user_id (σ check_in_time >= '17:00' ∧ check_in_time < '18:00' (ATTENDANCERECORD)) ∩ π user_id (σ check_in_time >= '18:00' ∧ check_in_time < '19:00' (ATTENDANCERECORD))	F20

8. SQL Query Implementation

8.1 DDL: Table Creation Scripts

Query	Description	SQL Query
DDL01	Create STUDENT table	CREATE TABLE user (user_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, first_name VARCHAR(255) NOT NULL, last_name VARCHAR(255) NOT NULL, dob DATE NOT NULL, gender ENUM('Male', 'Female', 'Other') NOT NULL, email VARCHAR(255) NOT NULL UNIQUE, phone VARCHAR(255) NOT NULL, cnic VARCHAR(255) NOT NULL UNIQUE, membership_status ENUM('Active', 'Inactive', 'Suspended') NULL, address TEXT NULL, emergency_contact VARCHAR(255) NULL)
DDL02	Create FACULTYMEMBER table	CREATE TABLE FacultyMember (facultymember_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, user_id INT NOT NULL, department_id INT NOT NULL, rank VARCHAR(255) NOT NULL, FOREIGN KEY (user_id) REFERENCES User(user_id), FOREIGN KEY (department_id) REFERENCES Department(department_id))
DDL03	Create MEMBERSHIP table	CREATE TABLE Membership (membership_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, user_id INT NOT NULL, membership_type_id INT NOT NULL, start_date DATE NOT NULL, end_date DATE NOT NULL, status ENUM('Active','Suspended','Expired') NOT NULL, freeze_months_used INT DEFAULT 0, last_intimation_date DATE, card_status VARCHAR(50) NOT NULL DEFAULT 'Issued', slot_id INT, FOREIGN KEY (user_id) REFERENCES User(user_id), FOREIGN KEY (membership_type_id) REFERENCES MembershipType(membership_type_id), FOREIGN KEY (slot_id) REFERENCES Timeslot(slot_id))

8.2 DML: Sample Insert, Update, Delete

Query	Description	SQL Query
DML01	Insert a new STUDENT	INSERT INTO STUDENT (cms_id, user_id, program_id, \`Residency Type\`, \`batch year\`) VALUES (1, 500230, 1, 'Hostellite', 2024)
DML02	Update batch_year for all students in program 2 to 2026	UPDATE STUDENT SET batch_year = 2026 WHERE program_id = 2
DML03	Delete expired memberships	DELETE FROM MEMBERSHIP WHERE end_date < CURDATE() AND status = 'Expired'

8.3 Joins & Filtering Queries

Query	Description	SQL Query
JOIN01	List user names and their membership status	SELECT first_name, last_name, status FROM User NATURAL JOIN MEMBERSHIP
JOIN02	Find all bookings on 2025-05-20 with facility name	SELECT f.facultymember_id, u.first_name, u.last_name, d.department_id, f.rank FROM FACULTYMEMBER f JOIN USER u ON f.user_id = u.user_id JOIN DEPARTMENT d ON f.department_id = d.department_id
JOIN03	Show maintenance requests assigned to staff 2	SELECT R.request_id, R.report_date, S.role FROM MAINTENANCEREQUEST R JOIN SPORTS_COMPLEX_STAFF S ON R.staff_id = S.staff_id WHERE R.staff_id = 2

8.4 Aggregations & Reporting

Query	Description	SQL Query
AGG01	Count students per program	SELECT program_id, COUNT(*) AS std_num FROM STUDENT GROUP BY program_id
AGG02	List membership types with at least 1 active member	SELECT membership_type_id, COUNT() AS active_count FROM MEMBERSHIP WHERE status = 'Active' GROUP BY membership_type_id HAVING COUNT() >= 1;
AGG03	Total payments collected per user	SELECT membership_id, SUM(amount) AS total_paid FROM PAYMENT GROUP BY membership_id

9.INDEXING AND JUSTIFICATION

Table	Column(s) to Index	Index Type	Justification
USER	email	UNIQUE INDEX	Fast lookup on login/verification
USER	cnic	UNIQUE INDEX	Quick identity checks
USER_MEDICAL_INFO	user_id	INDEX	JOINs to USER for health data retrieval
STUDENT	user_id, program_id	COMPOSITE INDEX	Filter or JOINs by student's user record and program
FACULTYMEMBER	user_id, department_id	COMPOSITE INDEX	Filter by faculty user and department
FAMILYMEMBER	user_id, facultymember_id	COMPOSITE INDEX	joins between users and their faculty relations

ALUMNI	user_id	INDEX	Lookup alumni activation and membership
PROGRAM	department_id	INDEX	JOINs to DEPARTMENT
DEPARTMENT	—	—	Primary key auto-indexed
MEMBERSHIP	user_id, membership_type_id, status	COMPOSITE INDEX	Commonly filter by member and plan, and by active/inactive status
MEMBERSHIPTYPE	—	—	Primary key auto-indexed
MEMBERSHIPFREEZE	membership_id	INDEX	Track freeze history per membership
MEMBERSHIP_VIOLATION	membership_id	INDEX	Quick lookup of violations per membership
LOCKERRENTAL	membership_id	INDEX	Lookup locker assignments per member
PAYMENT	membership_id, payment_date	COMPOSITE INDEX	JOINs/payments by membership, filter by date ranges
NSTPTENANT	user_id	INDEX	Lookup tenant periods by user
FACILITY	—	—	Primary key auto-indexed
FACILITY_BOOKING	facility_id	INDEX	Retrieve all bookings for a facility
FACILITYSCHEDULE	facility_id	INDEX	Filter or JOIN schedules by facility
FACILITY_MEMBERSHIPTYPE	facility_id, membership_type_id	COMPOSITE INDEX	Determine which plans apply to which facilities
STAFF_WORKING_SCHEDULE	staff_id	INDEX	Retrieve schedules for individual staff
STAFF_DAYS_OFF	staff_id	INDEX	Lookup days off by staff member

SPORTS_COMPLEX_STAFF	user_id, supervisor_id	COMPOSITE INDEX	Navigate staff hierarchy and link to user profiles
ATTENDANCERECORD	user_id, facility_id, check_in_time	COMPOSITE INDEX	Query attendance by user/facility and order chronologically
FEEDBACK	user_id, facility_id, date_submitted	COMPOSITE INDEX	Filter feedback by user, facility, or date
EQUIPMENT	facility_id	INDEX	Lookup equipment by location
EQUIPMENT_TYPE_VENDOR	equipment_type_id, vendor_id	COMPOSITE INDEX	Many-to-many mapping lookups
MAINTENANCEREQUEST	equipment_id, user_id, staff_id	COMPOSITE INDEX	queries on maintenance logs
PROCUREMENTORDER	vendor_id, equipment_id	COMPOSITE INDEX	Query orders by vendor or equipment
VENDOR_CONTACT	vendor_id	INDEX	Retrieve all contact points for a vendor
TOKEN	Token_id, user_id	COMPOSITE INDEX	Track issued tokens by user
VIOLATION	—	—	Primary key auto-indexed; lookups via junction tables
INCIDENT	violation_id	INDEX	Link incidents to their violation records
EVENTBOOKING	date, booking_status	COMPOSITE INDEX	Filter event bookings by date and status
TIMESLOT	facility_id	INDEX	Determine available slots for a facility

10. Analysis and Conclusion

Strengths

1. Clean, Organized Data
    - We've split information into focused tables (Users, Memberships, Facilities, Bookings, Payments, etc.), so each piece of data lives in just one place. That makes it easy to keep everything accurate.
2. Easy to Grow
    - Important lists like programs or membership types live in their own tables instead of hard-coded lists. Adding a new program or plan won't require huge change
3. Good Performance for Checks and Reports
    - Key fields (user IDs, dates) are indexed, so looking up daily check-ins or payments is fast.

Limitations

1. Multiple Tables for User Roles
    - Students, faculty, alumni, and tenants each have their own table. Pulling a "combined" list of all users means writing more complex queries.
2. Busy Check-In Times
    - Morning rushes (hundreds of check-ins at once) can slow down inserts. We'll need to consider batching inserts.

Future Improvements

1. One-Place User Roles
   Make it easy to list every user: student, staff, alumni by adding a single "role" field in the main user table.
2. Speed Up Big Tables
   Break attendance and payment records into monthly chunks so daily operations stay fast.
3. Quick Daily Summaries
   Keep simple tables that store each day's total (check-ins, payments, feedback), so dashboards load instantly.

11. Appendices

INTERVIEWS

Interview 1: Umer (SEECS, Electrical Engineering, Batch 2023)

Q: Assalamualaikum, who am I here with?
A: Umer

Q: And what department are you from?
A: I am from SEECS, electrical engineering.

Q: What batch?
A: 2023

Q: So do you hold a membership for the gym? If so, what sort of package or plans do you follow?
A: Yup, I hold a membership for the gym and as per the plan I pay 1000 per month just for the gym.

Q: Only for the gym?
A: Ok.

Q: How often do you visit the gym?
A: 4 days a week.

Q: And how does the staff keep track of your payment?
A: Basically, we submit the fees at the bank via some online app and then we submit the receipt at the reception desk.

Q: Right, so you submit a physical receipt at the reception?
A: Yup.

Q: So that's how they know, because if you submit your payment from a different account, like the holder name is something else from yours, so then you have to submit the physical receipt.
A: No, no, no, even if you are paying from the account which is on your own name, you still have to submit a physical receipt. And if it is not on your name, then you have to go to the bank and submit it there.

Q: Oh, right, ok. Wait, submit the fees there physically?
A: You have to submit the fees there physically and they give you the receipt and you have to submit it here. If you have your own account, you submit it via online app and then print the printout here.

Q: Ok, got it. And how do the staff keep a record of your attendance?
A: As for the attendance, they kind of don't really care. There's a registry and whenever you come, you have to enter your name and gym card number and time. And even if you don't, then there's no problem.

Q: Right, so it's totally manual?
A: Yup.

Q: Ok, and what facilities do you often use in the gym?
A: Most of all, the equipment as well as the washrooms and the showers.

Q: Could you elaborate on what sort of equipment and the showers, like the lockers and all that?
A: As for the lockers, we usually just keep our bags there, but it is very rare. Mostly we just keep it outside because it is easier to access there. As for the equipment, we use dumbbells and the cables, mainly cables, and the barbell.

Q: And the cardio machines?
A: And the cardio machines.

Q: Ok, so are you satisfied with your experience at the gym?
A: I was, but I think there should be more cables because you can do every exercise on it and it is always under use. Most of the time, there's a huge line for you, you have to wait for your turn and there are 7 people after you.

Q: Any other improvements you'd like to suggest?
A: Not really.

Q: Ok, now for the bowling alley, have you ever been there?
A: No.

Q: Ok, so we're going to skip that for now. Now for the swimming pool, have you ever been in the swimming pool?
A: Yeah.

Q: And what sort of plans do they follow, as for the payment?
A: As for the payment, there is a combined plan that you can pay 1500 per month for the both swimming pool and the gym. So I follow that plan.

Q: Ok, right. So how often do you visit the swimming pool complex?
A: Only in the summer, like June, July, August.

Q: And how do they keep track of your payment? Is it, wait, the payment is the same as the gym?
A: Yes, same as the gym. But there's a difference, for example, if you submit via the bank, you only get one receipt. You cannot submit to two places at the same time. So you show it here, they enter your details and then you submit a physical receipt at the swimming pool.

Q: Oh, ok, that makes sense. And how do they keep a record of your attendance?
A: Just in the same way, but the only difference is there is always someone at the reception. So they are kind of a bit strict about the attendance and that slot they gave you about the swimming pool.

Q: Ok, and what's your experience in the gym, sorry, in the swimming pool? Has it been good?
A: Yes, I guess. They don't really clean the water. To be honest, it takes months, 2-3 months. Sometimes some senior says it takes a year. They clean it once a year.

Q: Ok, and what facilities do they provide and that you utilize there?
A: Just the swimming pool, I take a shower back at the hostel. I don't take a shower here.

Q: Ok, so what improvements would you suggest for the swimming pool complex?
A: Maybe the timing slots. They assign you a slot because there are too many people in the swimming pool. So they give you slots and the slots are not actually suitable for students sometimes. Sometimes they tell you to come from 6 to 7. Now sometimes it is not possible to come from 6 to 7. You have semester projects, presentation next day, you have dayscholar friends. So you are supposed to prepare it with them. But at the same time, if you are with your friends or some work, you cannot come to the swimming pool.

Q: Ok, so they provide specific slots for individuals as opposed to the gender?
A: Genders have a specific, females have a specific time slot during which each individual has a specific time, like 45 minutes. For example, mine was back in the summer, 8.50 to 9.

Q: In the morning or evening?
A: Evening, 8.50 to 9. And the girls, I think they have 5 in the evening.

Q: So if I am getting it right, basically there are slots for genders and in those slots, each individual is assigned a certain slot.
A: And the faculty have their own slot.

Q: How does that work?
A: So basically, the current slot for the faculty is 8 to 9. And during that time, only male faculty members can come.

Q: Male faculty members and their male kids.
A: Ok, wow, I did not know about that.

Q: Ok, thank you so much. What was your name?
A: Umar.

Q: Thank you so much Umar for your time.
A: -

Great! Here's the **Q&A format for the third interview** with **DD Sports (Sports Complex In-charge)**:

Interview 3: DD Sports (NUST Sports Complex In-Charge)

Q: Assalamualaikum, who am I here with?
A: I am DD sports

Q: So I had a few questions from you regarding sports complex. First of all, what are the main responsibilities of sports complex at University level?
A: At University level we try to provide the healthy sports activities to the students and promote the sports culture in students. We do not aim to produce professional sportsmen, but sports is a culture and healthy mind and healthy brain is supported by such sports. This is our priority and responsibility.

Q: Okay so, what departments and facilities are under your direct supervision?
A: See, all the sports facilities we have in NUST, among them we have 84 indoor sports facilities and 63 outdoor sports facilities, which includes bowling alley, saddle club, swimming pool and gymnasium. These are the main facilities. All these are supervised by us.

Q: How does a regular user access your facilities such as gym, swimming, bowling alley etc?
A: You have to apply for a membership on the website. The link for website is available and all the SOPs and forms are available there. You have to provide your documentation. You have to also submit your fee form. You can start accessing them from that very day.

Q: And are the records of these facilities kept in a centralized system or individually?
A: In our record we have a manual. We only check the attendance of the students. Students manually enter their attendance. They enter their name themselves and that's how we get their data. They also mention their time in and time out for the sports facility they use.

Q: Okay for example a student goes to a swimming pool and other goes to the gym, their data is recorded on separate registers. But when that data comes to you, do you deal with them separately or collectively?
A: Okay so for our paid facilities, as I told you 84+62 facilities, there are only 3 paid facilities, which includes saddle club, gym and swimming pool, the rest are free. For these 3, we have data in physical form and as well as a database on computer. On that we ensure that every member pays their monthly dues, and accordingly they can avail the facilities. So it is a dual system.

Q: So as far as I understand, payment system is digitalized and attendance system is manual?
A: Yes you're right.

Q: Other than that for inter departmental events, say between SEECS and S3H, how would you manage them?
A: For the sports calendar we take guidance from the HEC. Every year HEC wants a lot of events, around 35+. In them for the events we think we have talent in, we participate in those events. There are

around 15 to 20 such events for male and female. First it's inter school, or inter NUST, then it's inter collegiate, and then we play on HEC level. So we have plans the whole year as to what events we have to hold. Other than that schools have their own sports gala and sports week. In that they need some assistance, like ground, sports facilities, referees and equipment. They request for it and we allot these in regards to availability.

Q: As there are multiple sports facilities, and you handle them simultaneously, what challenges do you face here?
A: Now see from the point of view of students, there is a challenge that the H-12 campus has become very crowded. Some facilities are very popular like swimming and gym, and they're stacked. Regulating them is a challenge. Everyone wants to go from 5 to 9, although our facilities are open from the morning, but students have their classes. So everyone wants to visit from specifically 7 to 8. Regulating them is challenging.

Q: Okay so let's say you want to upgrade or replace damaged equipment, what's the process of such decisions?
A: See these replacements keep going round the year. Making and establishing these facilities is indeed a task, but maintaining them is a bigger task. So in our routine sports visit which is annual, we prioritize certain requirements, and our system of office with competent authorities take such decisions for routine maintenance. For example in the swimming pool we have to regulate chlorination and other chemicals. Obviously there's a budget for it, so we take approval for it and maintain them regularly. We have support from the work directorate as well. We get a lot of things done with their assistance. If not, we have to outsource them.

Q: My next question is, for the current system you have related to the sports complex, how much role does technology play?
A: Some facilities like bowling, they require a lot of technology. The whole scoring system is cloud based. And you can access them from the internet. So in that our firm burnstreet (?) whom we have taken the technology from, we take their assistance. Our own staff also looks after the minor maintenance. Even in sports a lot of technology is required such as camera and lights. We do not have that, as we cannot afford it. We don't have a digital scoreboard, only manual. We have referees that don't have technological assistance. A dispute for decision cannot be decided easily as there are no camera recordings.

Q: Is there a centralized system for booking and scheduling as in an online system? And alongside that is there a system for user feedback?
A: For feedback, we have 2 ways. Firstly, you can say it's manual. Each facility has its own complaint register. Users can write in them. Secondly, they can always talk to me through phone, email, send an ION or can even visit my office themselves.

Q: Right. Also does the booking and scheduling has a system that does not require you being physically there?
A: It can be done through a call, an ION request can also be sent. As I said that sports gala and sports week happen, so their sports office or DD admin can write an ION or call that they have sports week and

they need to book this ground. And for that they coordinate with a playzone (?) officer who they give requirements to and get their bookings.

Q: What improvements would you suggest in your current issues, disregarding financial issues for now?
A: The sky is not the limit. Sports equipment is very expensive nowadays. Our infrastructure and setup is humongous for sports. Which needs maintenance accordingly. There's a multi purpose hall in old gym complex. That hall has basketball, volleyball, table tennis and badminton facilities. Other than that other events also take place. So the basic thing that needs fixing is its floors and lights because they're unplayable. Half the lights aren't working as they're very old. With the passage of time these things keep on going. There can be so many improvements in the cricket ground. Pitch covers, lights, boundary ropes, rollers all can be added as improvements, but we have to keep our budget in mind.

Q: Other than that, are there external bodies and vendors that collaborate with sports complex?
A: Can you elaborate?

Q: So what I'm asking is are there vendors outside NUST whom you get equipment from, and what's the process. Also if an organization outside NUST wants an event inside regarding sports, what's the process for that?
A: Recently we had a blind women cricket championship here. Blind cricket board requested us in collaboration with Australian high commission they were organizing it. They requested us for a ground and we let them know the charges for a ground and they agreed. Hence they were able to arrange their event here. A volleyball event also took place regarding women empowerment. A Karachi based women empowerment club made it happen here. So different organizations keep coming. It's not fixed, anyone can approach. And for accessing equipment, we check the local market. Even in Sialkot. Some of our equipment is imported especially for bowling alley.

Q: My last question is, how do you measure the efficiency and performance of your sports complex?
A: It's based on user feedbacks as to how satisfied they are. The footfall is as to how many members attend it daily. Normally we have 1000 to 1500 members daily, and in peak season it can reach up to 2500 to 3000 people. So we get the feedback from them.

Gym staff

Q1: Tell me your name and what's your responsibility?
A1: My name is Hamza Rauf, and I'm a gym trainer.

Q2: How do you track attendance of students and staff?
A2: We have a proper register, and we maintain a daily record for attendance.

Q3: The staff that comes to the gym, do they have any attendance system?
A3: Yes, staff that comes regularly have their attendance marked on a regular basis.

Q4: In the gym, how do you ensure equipment is being used properly and not being damaged?
A4: I keep an eye everywhere, and if I see a student doesn't know how to use a machine, I tell them how to use it or what to do. I keep circulating and check if students are doing things right or not. If not, I guide them on what to do.

Q5: If equipment is damaged, what's the process?
A5: We inform the senior staff supervisor, and then the relevant people come to check the equipment. They fill out an approval sheet, and they assess whether the equipment is broken and if it can be fixed.

Q6: You mentioned you have staff at higher positions above you?
A6: Yes, above me is the supervisor, and we report to them.

Q7: Is there the same supervisor for everyone?
A7: Yes, the gym and pool share the same supervisor.

Q8: Are there any issues that come up?
A8: If someone forgets to mark attendance, we just ask them to come back and do it.

Q9: Anything related to your job, any issues?
A9: No, there are no issues related to my job.

Q10: What are your working hours?
A10: I work from 1 PM to 9 PM.

Q11: Is it every day?
A11: I have Fridays and Sundays off.

Old gym staff

Q1: Name and occupation
ANS: Habib ur Rehman, sports branch ground staff.

Q2: How do you guys make entries of students?
ANS: We have a proper register for male and female students in the sports hall. We make entries with the in and out timings, and then we send the details to the DD sports, who forward them to the rector. The purpose of entering this information is so we know who's coming — whether it's a student, faculty, or alumni. We don't allow all alumni; some are allowed, some are not.

Q3: Is there separate attendance for everything?
ANS: Yes, everything has a separate register.

Q4: While you work here, is there any issue you face on a daily basis?
ANS: No major issues. We work for 8 hours a day. If there are days where we have to work for 12 hours, it's a challenge due to limited staff. Sometimes, we end up working late, and on some days, one person does the work while the other person covers the rest.

Q5: Are there only 2 staff members?
ANS: Yes.

Q6: What are the working hours?
ANS: The working hours are from the morning until 9 PM. In the morning, we have sessions from 5 to 6 AM, or 7 to 8 AM, and then we continue until 10 PM at night.

Q7: The equipment you have, like basketball or gym equipment, what happens when something is damaged? How do you guys fix it?
ANS: If anything gets damaged, we take it and issue a new one that we already have in stock. In the gym, we haven't faced many cases of equipment damage. If someone does damage it, they have to fill out a form.

Q8: For events, how is the booking or time slot arranged?
ANS: Whoever is hosting the event, whether it's for the sports hall or another facility, if it's from an organization like SEECES, they send the request to our DD Sports. Based on that, we adjust the time and email them with the confirmed slot. The sports ground or any other ground is booked for them accordingly.

Q9: In this attendance system, do faculty and students share one register?
ANS: Yes, they share one register. They write down their details themselves, indicating whether they are a student or faculty member.

Q10: Is there anything here that can be improved, or anything you would suggest?
ANS: There's nothing major to improve, but overtime work is very difficult. If staff work for 4 additional hours, they only make 40 rupees more. Either the staff should be increased, or overtime pay should be increased

Bowling staff

Q1: Tell your name and occupation.
ANS: My name is Mahtab Akhtar, and I'm a machine operator here in the bowling alley.

Q2: Do you have to do attendance, and how do you do it?
ANS: Yes, we record their name, school, CMS ID, and token serial number.

Q3: Is ticketing done on paper?
ANS: No, the tokens are already made. Four tokens are given, and each is issued under one person's name.

Q4: Can one person take as many tokens as possible?
ANS: If it's one, they can take one. If it's four, they can take four.

Q5: Is it mandatory for you to be a student?
ANS: Yes.

Q6: Is the process the same for students?
ANS: Yes, it's the same.

Q7: How is your attendance done?
ANS: Our supervisor marks attendance daily.

Q8: How is the equipment and tokens checked here?
ANS: For example, if there's a line, the first game has a certain color token, and the second game has a different color. That's how we can tell which game a person is playing.

Q9: Any issues with staff or students?
ANS: No, no problems.

Q10: If any equipment is damaged, either by staff or students, what's the process?
ANS: We inform the senior staff. Sometimes, if there's a mistake, we just tell the senior and don't take any action on our side. We also note down the CMS ID of the student if the supervisor asks.

Q11: If there are events, what is the process?
ANS: We talk to the DD sports about it.

Q12: Is there anything about your job that you dislike?
ANS: No, nothing. Alhamdulillah, everything is good. If there's something, we talk to our senior about it.

Q13: How many hours do you work, and is it regular? Is there a shift?
ANS: We have two days off. I take Saturday and Sunday off, while the other person with me takes Friday and Saturday. I come early because of transport issues, but others arrive by 1 PM.

Female staff

Q1: Who are you here with?
ANS: I'm a physiotherapist and gym instructor here, and I keep the register.

Q2: What are your main responsibilities in the gym?
ANS: The entire female gym is under my responsibility because I give training, provide instruction on machines, handle registration, and manage everything related to it.

Q3: How is the registration done?
ANS: The registration is done by downloading a form from Qalam, filling it out, attaching two pictures, a student card copy, and a bank challan. Once all the documents are collected, they bring it to me, and then the registration is completed.

Q4: Do you enter the data, or how is this done?
ANS: Yes, I have a register where I enter their name, department, school, and whether they are UG or PG.

Q5: What kind of information do you store about them?
ANS: I store their name, address, whether they are a day scholar or in the hostel, emergency contact details, and any medical issues. There is also a small interview after which the registration is completed.

Q6: Are these records maintained by you alone, and do you have to submit them?
ANS: Yes, I maintain these records alone. The AD/DD can ask to see the records whenever needed.

Q7: If you need to order new equipment, how is that done?
ANS: First, we inform our supervisor, who tells the AD, and then we get a recommendation from the DD. Once that's done, we get the machinery or equipment fixed.

Q8: What other things do you keep records of on paper?
ANS: Almost everything. For example, if a student registers but doesn't come for a year, the AD/DD might contact me and say, "This student has been registered; is there a problem?" I immediately open my register, write the student's record on a separate page, and send it to them.

Q9: So, all student records are on paper?
ANS: Yes, all records are maintained on paper.

Q10: Even the equipment records, everything?
ANS: Yes, all of that is on paper as well.

Q11: Is there a set frequency for how often the AD/DD contacts you?
ANS: No, it's random. They can contact me whenever they need something.

Q12: Are there any issues you face?
ANS: For example, if there are 2,000 registrations done in a year and I need to find a specific student, it becomes very hard. If there was a computerized system, it would make things so much easier. Right now, I have to open last year's records, give the gym card number, and compare it with everything else to find the record. It's time-consuming. There's been so much scientific advancement, but we're still working with paperwork.

Q13: What happens if equipment is damaged?
ANS: It's not a big issue. We don't ask the student to pay for it; we just inform the superiors, and they get it fixed.

Q14: Do you need approval for events or anything like that?
ANS: Yes, for events like the sports gala, those are handled separately from here.

Q15: Do you have any suggestions?
ANS: As a trainer, most of my time is spent on paperwork, it would be much better if everything was digitized. It would give me more time to spend with the students.

Student 2-Sara

Q1: Assalamu Alaikum, who am I here with?
ANS: Walaikum Assalam, I'm Sarah Fawad. From? From SEECES, DS 509615.

Q2: Okay, so I had a few questions regarding the gym, the sports complex, and the swimming complex, if you've been there.
ANS: Sure!

Q3: First of all, how did you register for the gym?
ANS: There is a form on Qalam. You have to print it, fill it out, pay online, and then give the form to the staff over there.

Q4: Right, so have you ever been to another gym, and if you have, what was the process of registration there?
ANS: Yes, I have been to another gym, but there was no form. You had to go to the registration desk, where they gave you a form and a slip. Then you had to pay, and that's it

Q5: So, it was manual as well?
ANS: Yes, it was manual as well

Q6: Does the number of students make the registration inconvenient at NUST?
ANS: No, I don't think so. Since it's a manual form, you just need to print it, fill it out, and submit it to the staff. There is no issue with that.

Q7: Do you have to pay separately for the services provided by the gym, or is it in a single plan?
ANS: No, it's a single plan. You pay for the registration, and then there are monthly fees. There are no extra fees for any other services.

Q8: Okay, if you've ever participated in any sports events, what was the registration process for that?
ANS: They give you an online form where you have to fill in your details. After submitting the form, they contact you to confirm your participation.

Q9: So, basically, they have no way of saving your data in a formal way, like in a database or something like that? It's more of a list?
ANS: I think they do save the data online since they use an online form, but I don't know more than that.

Q10: Have you ever requested new equipment or reported broken equipment? If so, what was the process?
ANS: Not yet, but I heard that some equipment was missing. Students reported it to the staff, and they said they would arrange it. I'm not sure if it's been arranged, though.

Q11: So, basically, whoever reports the issue can report it directly to the staff, but there's no official feedback system online?
ANS: Yes, that's correct. There's no system for that.

Q12: There's no way to know if the issue has been approved or not?
ANS: Yes, exactly. There's no way to track it.

Q13: Have you ever been mischarged?
ANS: No, not in that sense.

Q14: As in, have they charged you more money than you thought was fair?
ANS: No, I think the charges are completely fair.

Q15: Any improvements you would suggest for the gym?
ANS: Yes, I would suggest having a snacks corner for shakes, protein, and such. Also, more equipment for the girls' gym because I think the boys' gym has more equipment. The girls' gym could use more options since girls don't usually do gym workouts as much, and sometimes our complaints aren't addressed properly.

Q16: Right, so you don't know if your complaint has been approved or not?
ANS: Yes, that's correct.

Q17: How do they keep the record of your attendance?
ANS: There's a register where you have to write your name and sign in. It's all manual, there's no online system.

Q18: So, if it's manual, does that mean a person can just sign in for someone else?
ANS: Yes, they can. But I don't think that affects much because it's up to you if you want to come regularly or not. There's no strict rule for attendance.

Q19: This can cause multiple students to apply for the gym membership under the same card since they can't track who has applied and who has not, right?
ANS: I don't think so, because when you come, you have to sign the manual register and show your membership card. The card has your picture, so they know it's for one person. I don't think someone else can join in your name.

Q20: Right, thank you so much.
ANS: You're welcome!

Student1- ABBAS

Q1: Assalamualaikum, who am I here with?
ANS: You're here with Abbas.

Q2: Abbas from?
ANS: S3H, the Public Administration.

Q3: Okay, so I had a few questions I wanted to ask regarding the gym. How did you register for NUST Gym?
ANS: Basically, there's a website where you find a document. You print the document, fill out the details, get printouts of your passport-size pictures, and then submit them along with a copy of the fee receipt.

Q4: Have you been to another gym? And what was the process of registration there?
ANS: I've been to another gym, and the process was just you go to the reception, they write your name, you pay the fee, and that's it.

Q5: So that's pretty much manual?
ANS: Yes, manually. I've also been to a gym where it was automatic. They just type your name in a computer, and that's pretty much it. You just pay the fee at the reception, and that's it.

Q6: Do you feel like a digitized system would be more convenient compared to something like just writing your name on paper?
ANS: I would love for there to be a digitalized system. There's a lot of problems with paperwork in the gym. For example, if you're paying fees, you show them the receipt, and then they say, "Okay, now get this printed out and bring it back." So if it was digitalized, that would be a lot less hassle.

Q7: Does the number of students make the registration process inconvenient for the gym here?
ANS: For the gym here, no, I don't think so. A lot of people have already registered, so you just go up to them, and they do it on the spot. But the process before that is pretty trivial.

Q8: Do you have to pay separately for services provided by the gym?
ANS: No, there are no separate charges for services.

Q9: So it's in one single plan?
ANS: Yes, everything is included in one single plan.

Q10: If you've ever participated in any sports events, what was the registration process for that?
ANS: If there's an assigned team captain, you go up to the team captain, give trials, and then they add you to the team after you give them your details. I think that's pretty much it.

Q11: So they save your data in a manual form as well?
ANS: Yes, in a notebook.

Q12: If you need to request new equipment or report broken equipment, what's the process?
ANS: There have been instances where equipment was broken. For example, the cable was broken for an entire month. Usually, the process is that you tell the staff, and then they do the rest. But recently, the cable was broken for an entire month. Today, I went, and there were dumbbells that were broken or about to break. There are a lot of broken ones there, and they've been there for a long time, and they're not fixing them yet. So I don't think it's a very efficient process, even if there is one.

Q13: So there's no record of your feedback?
ANS: No, not really. The staff walks around the gym and if they see something broken, they'll fix it eventually. But for example, the dumbbells I picked up today were moving so much that it was almost scary because they were about to fall. And the broken dumbbells have never been fixed.

Q14: Have you ever been mischarged for the gym?
ANS: Yes, because they have this policy that if you go to the gym for 11 months but don't go during Ramadan, and you don't inform them to stop your gym membership, they'll still charge you for that month, even if you didn't go. You have to tell them beforehand, and I never knew I had to do that. So, they had to charge me for that, even though I didn't go.

Q15: So, you feel like you were misinformed?
ANS: Yes.

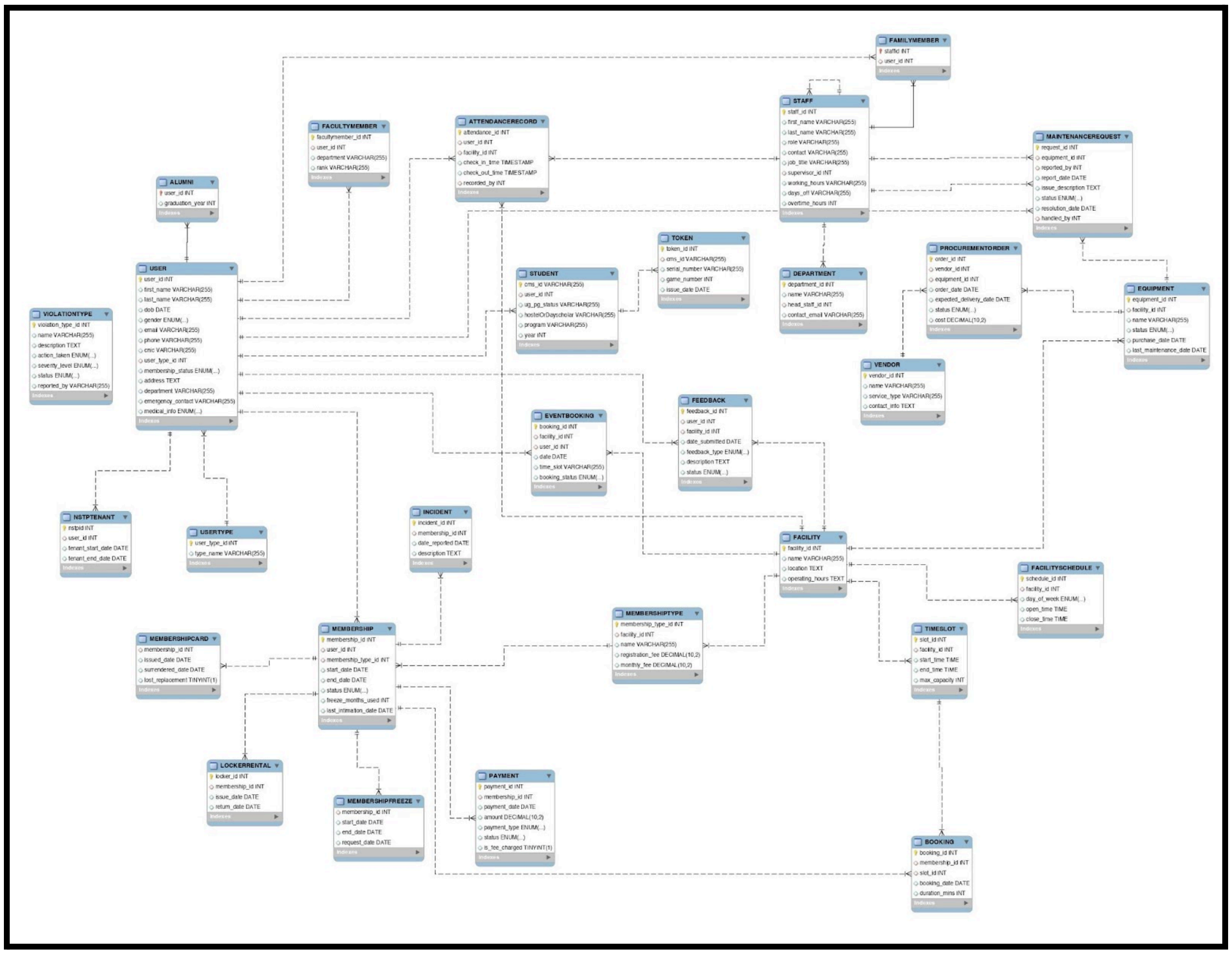
Q16: So there could be a better system?
ANS: Yes, there could be a better system.

Q17: Any improvements you would suggest for the gym?
ANS: Improvements, I think they could change some machines because there are a lot of useless machines. Maybe they could add a punching bag or something for cardio and such. And as we discussed, if they digitize the registration system, especially the fee payment process, it would be a lot easier. Because right now, it's a hassle to pay the fee online, show the receipt, and then they send you back to the printer to print it out again. Then you have to go back and show them again before they let you in. It's a pretty long process.

Q18: And you mentioned that if you want to request something, you just verbally say it. Would it be better if there was an online system so you could know if your request has been approved or not?
ANS: Yes, I think if there was a feedback system, maybe every month or every two months, where you could tell them what's wrong with the gym or what you want in the gym, I think that would improve a lot.

Q19: Right, that's about it. Thank you so much.
ANS: You're welcome!

MOCK ERD


FAMILYMEMBER
staffid INT
user_id INT
STAFF
staff_id INT
first_name VARCHAR(255)
last_name VARCHAR(255)
role VARCHAR(255)
contact VARCHAR(255)
job_title VARCHAR(255)
supervisor_id INT
working_hours VARCHAR(255)
days_off VARCHAR(255)
overtime_hours INT
MAINTENANCEREQUEST
request_id INT
equipment_id INT
reported_by INT
report_date DATE
issue_description TEXT
status ENUM(...)
resolution_date DATE
handled_by INT
FACULTYMEMBER
facultymember_id INT
user_id INT
department VARCHAR(255)
rank VARCHAR(255)
ATTENDANCERECORD
attendance_id INT
user_id INT
facility_id INT
check_in_time TIMESTAMP
check_out_time TIMESTAMP
recorded_by INT
ALUMNI
user_id INT
graduation_year INT
TOKEN
token_id INT
cms_id VARCHAR(255)
serial_number VARCHAR(255)
game_number INT
issue_date DATE
PROCUREMENTORDER
order_id INT
vendor_id INT
equipment_id INT
order_date DATE
expected_delivery_date DATE
status ENUM(...)
cost DECIMAL(10,2)
USER
user_id INT
first_name VARCHAR(255)
last_name VARCHAR(255)
dob DATE
gender ENUM(...)
email VARCHAR(255)
phone VARCHAR(255)
cnic VARCHAR(255)
user_type_id INT
membership_status ENUM(...)
address TEXT
department VARCHAR(255)
emergency_contact VARCHAR(255)
medical_info ENUM(...)
STUDENT
cms_id VARCHAR(255)
user_id INT
ug_pg_status VARCHAR(255)
hostelOrDayscholar VARCHAR(255)
program VARCHAR(255)
year INT
DEPARTMENT
department_id INT
name VARCHAR(255)
head_staff_id INT
contact_email VARCHAR(255)
EQUIPMENT
equipment_id INT
facility_id INT
name VARCHAR(255)
status ENUM(...)
purchase_date DATE
last_maintenance_date DATE
VIOLATIONTYPE
violation_type_id INT
name VARCHAR(255)
description TEXT
action_taken ENUM(...)
severity_level ENUM(...)
status ENUM(...)
reported_by VARCHAR(255)
VENDOR
vendor_id INT
name VARCHAR(255)
service_type VARCHAR(255)
contact_info TEXT
FEEDBACK
feedback_id INT
user_id INT
facility_id INT
date_submitted DATE
feedback_type ENUM(...)
description TEXT
status ENUM(...)
EVENTBOOKING
booking_id INT
facility_id INT
user_id INT
date DATE
time_slot VARCHAR(255)
booking_status ENUM(...)
NSTPTENANT
nstpid INT
user_id INT
tenant_start_date DATE
tenant_end_date DATE
USERTYPE
user_type_id INT
type_name VARCHAR(255)
INCIDENT
incident_id INT
membership_id INT
date_reported DATE
description TEXT
FACILITY
facility_id INT
name VARCHAR(255)
location TEXT
operating_hours TEXT
FACILITYSCHEDULE
schedule_id INT
facility_id INT
day_of_week ENUM(...)
open_time TIME
close_time TIME
MEMBERSHIPTYPE
membership_type_id INT
facility_id INT
name VARCHAR(255)
registration_fee DECIMAL(10,2)
monthly_fee DECIMAL(10,2)
MEMBERSHIPCARD
membership_id INT
issued_date DATE
surrendered_date DATE
lost_replacement TINYINT(1)
MEMBERSHIP
membership_id INT
user_id INT
membership_type_id INT
start_date DATE
end_date DATE
status ENUM(...)
freeze_months_used INT
last_intimation_date DATE
TIMESLOT
slot_id INT
facility_id INT
start_time TIME
end_time TIME
max_capacity INT
LOCKERRENTAL
locker_id INT
membership_id INT
issue_date DATE
return_date DATE
MEMBERSHIPFREEZE
membership_id INT
start_date DATE
end_date DATE
request_date DATE
PAYMENT
payment_id INT
membership_id INT
payment_date DATE
amount DECIMAL(10,2)
payment_type ENUM(...)
status ENUM(...)
is_fee_charged TINYINT(1)
BOOKING
booking_id INT
membership_id INT
slot_id INT
booking_date DATE
duration_mins INT