Final Project

on

BUET Database Audit, Normalization, and Business Intelligence Reporting

MIS-443- Applied Database Management

Submitted By:

Name	ID
Labiba Khan	22204168
Arup Ratan Das	22304071
Islahuddin Joarder	22304027

Submitted To:

Saniun Saber Chowdhury

BRAC Business School,

BRAC University.

Introduction:

BUET, a very well-known database for university administration, will be thoroughly audited, normalized, and improved as a part of this project of ours. Important academic and administrative tasks like student enrollment, course administration, research of faculties, library operations, monitoring attendance, management of fees and facilities are all supported by this database. This system is crucial for this organization's daily operation and strategic decision-making. This project is designed to improve data integrity and provide stakeholders such as administrators, faculties and department heads with actionable insights into this business through systemic SQL corrections and the creation of analytical reports.

Initial Database Issues and Applied Fixes:

Issue Category	**Issue Details**	**Issue Type**	**Applied Fix**	**Solution Details**
Missing Primary Keys	There was a risk of duplicate and orphaned records since 22 tables lacked primary keys.	Structural integrity	Added primary key constraint via alter table	Each table has unique identifier now
Incorrect Data Types	Date-based queries and validations are not possible since date columns are stored as “VARCHAR” .	Datatype error	Converted varchar date columns to date using the backup/restore approach.	Used try_cast() for safe conversion then renamed columns like attendance_date_*str to attendance* date
Missing	There was no	Referential	Added 28 foreign	Ensured data

Foreign Keys	relationship between tables (eg: students to departments)	integrity	key constraints	consistency and prevented orphaned records like fk_students_departments.
Redundant Table	Revenue summary table duplicated already available in fees/payments.	Normalization issue	Dropped revenue summary table	Removed data redundancy and updated anomalies.
Poorly Designed ERD	Courses contained semester and year, which belonged in enrollments.	Normalization issue	Erased semester and year from courses table	Moved time bound attributes to proper transactional tables.
Duplicate Grade Storage	Enrollments and grades both stored grade info.	Data anomaly	Erased grade column from enrolments	Centralized grade storage in the “grades” table to avoid anomalies.
Missing Linking Tables	Faculty involvement wasn't properly given in the database.	ERD incompleteness	Made facultyprojects junction table.	Enabled many-to-many relationships between faculty and research projects.

Analytical Reports and Insights:

1. **Report Name:** Top 5 students by CGPA per department.
   Report Summary: This Query Shows the top performing students of each department.
   Query:

```
WITH RankedStudents AS (
    SELECT
        d.dept_name,
        s.name as student_name,
        s.cgpa,
        s.admission_date,
        ROW_NUMBER() OVER (PARTITION BY d.dept_id ORDER BY s.cgpa DESC) as
rank_in_dept
    FROM Students s
    JOIN Departments d ON s.dept_id = d.dept_id
    WHERE s.cgpa IS NOT NULL AND s.status = 'Active'
)
SELECT
    dept_name,
    student_name,
    cgpa,
    admission_date,
    rank_in_dept
FROM RankedStudents
WHERE rank_in_dept <= 5
ORDER BY dept_name, rank_in_dept;
```

Query Output & business decision:

dept_name	student_name	cgpa	admission_date	rank_in_dept
Architecture	Nitesh Batra	4	4/15/2023	1
Architecture	Gauri Acharya	3.98	4/2/2020	2
Architecture	Eshana Dasgupta	3.98	3/3/2023	3
Architecture	Harita Kaul	3.98	1/2/2022	4
Architecture	Gaurangi Shah	3.97	8/23/2024	5
Chemical Engineering	Hritik Chakraborty	3.97	4/10/2022	1
Chemical Engineering	Harrison Balan	3.91	5/8/2023	2
Chemical Engineering	Vrinda Kalita	3.89	3/13/2020	3
Chemical Engineering	Ira Mangal	3.88	9/14/2020	4
Chemical Engineering	Avi Muni	3.86	8/24/2023	5
Civil Engineering	Yasti Ahluwalia	3.99	4/17/2022	1
Civil Engineering	Rehaan Pingle	3.97	7/13/2021	2
Civil Engineering	Samaksh Borah	3.94	1/28/2020	3
Civil Engineering	Faris Sankaran	3.93	8/25/2020	4
Civil Engineering	Balhaar Chauhan	3.93	12/6/2024	5
Computer Science & Engineering	Damini Pandit	3.98	11/1/2023	1

Computer Science & Engineering	Triveni Mallick	3.97	7/10/2024	2
Computer Science & Engineering	Hamsini Issac	3.95	1/13/2024	3
Computer Science & Engineering	Robert Misra	3.94	5/17/2023	4
Computer Science & Engineering	Advaith Tailor	3.93	11/22/2024	5
Electrical & Electronic Engineering	Yoshita Bhasin	3.99	10/3/2021	1
Electrical & Electronic Engineering	Oeshi Chaudhry	3.98	11/30/2020	2
Electrical & Electronic Engineering	Oni Badal	3.97	3/25/2021	3
Electrical & Electronic Engineering	Imaran Chokshi	3.91	3/21/2022	4
Electrical & Electronic Engineering	Yutika Pradhan	3.86	1/22/2023	5
Mathematics	Amruta Kar	3.97	6/8/2023	1
Mathematics	Abhimanyu Jha	3.97	8/7/2023	2
Mathematics	Upadhriti Issac	3.96	11/14/2021	3
Mathematics	Azad Atwal	3.94	7/12/2024	4
Mathematics	Nicholas Warrior	3.89	7/25/2020	5
Mechanical Engineering	Ekapad Kalita	3.99	9/22/2021	1
Mechanical Engineering	Nandini Hegde	3.98	10/12/2020	2
Mechanical Engineering	Umang Sodhi	3.98	1/29/2021	3
Mechanical Engineering	Ojasvi Andra	3.97	10/20/2023	4
Mechanical Engineering	Saumya Chokshi	3.94	8/18/2024	5
Physics	Ekapad Babu	3.99	9/30/2021	1
Physics	Zilmil Guha	3.98	5/19/2023	2
Physics	Gunbir Dalal	3.98	11/12/2021	3
Physics	Ekiya Samra	3.96	3/23/2021	4
Physics	Jackson Pandey	3.95	9/19/2024	5

The output provides the CGPA and admission dates of the academic stars in each department. For instance, the Computer Science department would list students with CGPAs between 3.9 and 4.0. The university should:

- Recognize these students for honors programs.
- Offer them research assistant positions.
- Use them as peer mentors for struggling students.

2. **Report Name:** Average Grades per course.
   Report Summary: This query calculates average marks, enrollment counts, and A-grade percentages for each course in the current academic year.
   Query:

```
SELECT
    c.course_code,
    c.course_name,
    d.dept_name,
    COUNT(DISTINCT e.student_id) as total_students,
    AVG(CAST(g.final_marks AS DECIMAL)) as avg_marks,
    COUNT(CASE WHEN g.final_grade IN ('A', 'A+') THEN 1 END) * 100.0 /
COUNT(*) as A_grade_percentage
```

```
FROM Courses c
JOIN Departments d ON c.dept_id = d.dept_id
JOIN Enrollments e ON c.course_id = e.course_id
LEFT JOIN Grades g ON e.enrollment_id = g.enrollment_id
WHERE e.year = 2024  -- Current academic year
GROUP BY c.course_code, c.course_name, d.dept_name
ORDER BY avg_marks DESC;
```

Query Output & business decision: The outcome could show that "Database Systems" has 95% A grades and "Statistics" only has 40%. The college should:

- Look into how consistent grades are throughout departments
- Look over the curriculum for courses that aren't doing well.
- Share what works best in courses that go well

3. **Report Name:** Active Students per Department.

   Report Summary: This query counts active students in each department with admission date ranges.

   Query:

```
SELECT
    d.dept_name,
    COUNT(s.student_id) as total_students,
    MIN(s.admission_date) as oldest_admission,
    MAX(s.admission_date) as newest_admission
FROM Departments d
LEFT JOIN Students s ON d.dept_id = s.dept_id
WHERE s.status = 'Active' OR s.status IS NULL
GROUP BY d.dept_name
ORDER BY total_students DESC;
```

Query Output & business decision:

dept_name	total_students	oldest_admission	newest_admission
Architecture	135	1/10/2020	12/10/2024
Civil Engineering	134	1/7/2020	12/28/2024
Physics	133	3/14/2020	11/23/2024
Mechanical Engineering	119	1/12/2020	12/29/2024
Mathematics	117	1/20/2020	12/25/2024
Chemical Engineering	113	1/4/2020	12/16/2024
Computer Science & Engineering	113	1/2/2020	11/22/2024
Electrical & Electronic Engineering	112	3/11/2020	12/12/2024

The output shows that the Business Department has 500 students, the oldest of whom started in 2018 and the newest in 2024. The Philosophy Department has 50 students. This institution should:

- Allocate resources to departments based on their size.
- Plan how many teachers to hire based on how many students there are for each teacher.
- Review small departments for viability

4. **Report Name:** Faculty Count per Dept

   Report Summary: This query counts faculty members in each academic department.

   Query:

```
SELECT
      d.dept_name,
      COUNT(f.faculty_id) as faculty_count
FROM Departments d
LEFT JOIN Faculty f ON d.dept_id = f.dept_id
GROUP BY d.dept_name
ORDER BY faculty_count DESC;
```

   Query Output & business decision:

dept_name	faculty_count
Computer Science & Engineering	46
Civil Engineering	42
Mechanical Engineering	39
Mathematics	37
Chemical Engineering	37
Architecture	34
Physics	33
Electrical & Electronic Engineering	32

If Engineering has 40 professors and Arts has 10, the university should:

- Look at the teaching loads to make sure they are fair.
- Think about giving teachers from different departments the same classes.
- Make plans to hire more people in departments that are growing.

5. **Report Name:** Course Catalogue by Department.

   Report Summary: This query lists all courses offered by each department with credit information.

   Query:

```
SELECT
    d.dept_name,
    c.course_code,
    c.course_name,
    c.credits
FROM Departments d
JOIN Courses c ON d.dept_id = c.dept_id
ORDER BY d.dept_name, c.course_code;
```

   Query Output & business decision: (The Output is too long to provide here)

   The full list of courses shows that Computer Science has 45 courses (3 credits each) while History has 25. The university should:

   - Ensure course offerings align with program requirements
   - Find the gaps in the course sequences
   - Distribute this catalog out to help students.

6. **Report Name:** Most Popular Library books.

   Report Summary: This query identifies the top 10 most borrowed books with borrowing patterns.

   Query:

```
SELECT TOP 10
    lb.title,
    lb.author,
    d.dept_name,
    COUNT(bi.issue_id) as times_borrowed,
    AVG(DATEDIFF(DAY, bi.issue_date, bi.return_date)) as avg_days_borrowed
FROM LibraryBooks lb
LEFT JOIN BookIssues bi ON lb.book_id = bi.book_id
LEFT JOIN Departments d ON lb.dept_id = d.dept_id
WHERE bi.issue_date IS NOT NULL
GROUP BY lb.title, lb.author, d.dept_name
ORDER BY times_borrowed DESC;
```

Query Output & business decision:

title	author	dept_name	times_borrowed	avg_days_borrowed
Eligendi doloremque quae reiciendis.	Kritika Dash	Electrical & Electronic Engineering	6	NULL
Quas optio facilis dolor deserunt.	Kamya Krishnan	Civil Engineering	6	-121
Repudiandae omnis doloremque ea rem.	Ucchal Behl	Mathematics	6	0
Ab quod.	Faraj Badal	Civil Engineering	5	NULL
Cum vero animi iure.	Omisha Rao	Physics	5	183
Deleniti sed.	Bina Pau	Electrical & Electronic Engineering	5	90
Eligendi temporibus corrupti.	Finn Chowdhury	Computer Science & Engineering	5	76
Itaque iste soluta.	Tanay Lall	Electrical & Electronic Engineering	5	15
Iure quisquam soluta.	Samar Radhakrishnan	Computer Science & Engineering	5	-274
Maxime quo optio.	Vansha Salvi	Mathematics	5	NULL

If "Introduction to Algorithms" has been borrowed out 200 times (about once every 14 days), while other books are checked out 20 times, the library should:

- Buy more copies of books that are in high demand.
- Think about making e-book versions of books that people borrow a lot.
- Get rid of books that haven't been borrowed in three years or more.

7. **Report Name:** Room Utilization Analysis.

   Report Summary: This query analyzes classroom and building usage patterns.

   Query:

```
SELECT
    b.name as building_name,
    r.room_number,
    r.room_type,
    COUNT(DISTINCT sec.section_id) as sections_held,
    COUNT(DISTINCT sec.faculty_id) as different_faculty,
    MAX(r.capacity) as room_capacity
FROM Buildings b
```

```
JOIN Rooms r ON b.building_id = r.building_id
LEFT JOIN Sections sec ON r.room_id = sec.room_id
GROUP BY b.name, r.room_number, r.room_type
HAVING COUNT(DISTINCT sec.section_id) > 0
ORDER BY sections_held DESC;
```

Query Output & business decision: (The Output is too long to provide here) If Room 101 hosts 30 sections weekly at 90% capacity while Room 205 hosts 5 sections at 20% capacity, the university should:

- Optimize scheduling to balance room usage
- Consider converting underutilized spaces
- Plan new construction based on utilization patterns

8. **Report Name:** Attendance patterns by day or week.

   Report Summary: This query analyzes student attendance patterns across different weekdays.

   Query:

```
SELECT
    DATENAME(WEEKDAY, attendance_date) as day_of_week,
    status,
    COUNT(attendance_id) as attendance_count,
    COUNT(DISTINCT student_id) as unique_students
FROM Attendance
WHERE attendance_date IS NOT NULL
GROUP BY DATENAME(WEEKDAY, attendance_date), status
ORDER BY
    CASE DATENAME(WEEKDAY, attendance_date)
        WHEN 'Monday' THEN 1
        WHEN 'Tuesday' THEN 2
        WHEN 'Wednesday' THEN 3
        WHEN 'Thursday' THEN 4
        WHEN 'Friday' THEN 5
        WHEN 'Saturday' THEN 6
        WHEN 'Sunday' THEN 7
    END,
    status;
```

Query Output & business decision: (The Output is too long to provide here) The university should:

- Consider scheduling important classes earlier in the week
- Implement attendance incentives for low-attendance days

9. **Report Name:** Faculty Research Productivity

   Report Summary: This query evaluates faculty research output including projects led, funding secured, and students mentored.

   Query:

```
SELECT
    f.name as faculty_name,
    d.dept_name,
    f.designation,
    COUNT(DISTINCT rp.project_id) as projects_led,
    SUM(CAST(rp.budget AS DECIMAL(15,2))) as total_research_funding,
    COUNT(DISTINCT pm.student_id) as students_mentored,
    COUNT(DISTINCT sec.course_id) as courses_taught,
    DATEDIFF(YEAR, f.hire_date, GETDATE()) as years_of_service
FROM Faculty f
LEFT JOIN Departments d ON f.dept_id = d.dept_id
LEFT JOIN ResearchProjects rp ON f.faculty_id = rp.faculty_id
LEFT JOIN ProjectMembers pm ON rp.project_id = pm.project_id
LEFT JOIN Sections sec ON f.faculty_id = sec.faculty_id
WHERE f.status = 'Active'
GROUP BY f.name, d.dept_name, f.designation, f.hire_date
HAVING COUNT(DISTINCT rp.project_id) > 0
ORDER BY total_research_funding DESC;
```

   Query Output & business decision: (The Output is too long to provide here) The school should:

   - Offering promotions to people who do well at work
   - Share tips for drafting successful grant applications
   - Make sure that productive researchers have fair teaching loads.

10. **Report Name:** Building or room utilization for Capacity Planning.

    Report Summary: This query provides detailed capacity analysis for facilities planning.

    Query:

```
SELECT
    b.name as building_name,
    r.room_number,
    r.room_type,
    r.capacity as max_capacity,
    COUNT(DISTINCT sec.section_id) as sections_scheduled,
    COUNT(DISTINCT sec.faculty_id) as faculty_using,
    COUNT(DISTINCT ssa.student_id) as total_students_assigned,
    AVG(sec.max_capacity) as avg_section_capacity,
```

```
    ROUND(COUNT(DISTINCT ssa.student_id) * 100.0 / NULLIF(r.capacity *
COUNT(DISTINCT sec.section_id), 0), 2) as utilization_percentage,
    CASE
        WHEN COUNT(DISTINCT sec.section_id) = 0 THEN 'Underutilized'
        WHEN COUNT(DISTINCT ssa.student_id) * 100.0 / NULLIF(r.capacity *
COUNT(DISTINCT sec.section_id), 0) > 80 THEN 'Overutilized'
        WHEN COUNT(DISTINCT ssa.student_id) * 100.0 / NULLIF(r.capacity *
COUNT(DISTINCT sec.section_id), 0) < 40 THEN 'Underutilized'
        ELSE 'Optimally Used'
    END as utilization_status
FROM Buildings b
LEFT JOIN Rooms r ON b.building_id = r.building_id
LEFT JOIN Sections sec ON r.room_id = sec.room_id AND sec.year = 2024
LEFT JOIN StudentSectionAssignments ssa ON sec.section_id = ssa.section_id
GROUP BY b.name, r.room_number, r.room_type, r.capacity
ORDER BY utilization_percentage DESC;
```

Query Output & business decision: (The Output is too long to provide here) The university needs to:

- Redistribute classes in different buildings
- Make plans to fix up places that are too busy
- Think about using places that aren't being used enough for anything else.

11. **Report Name:** Faculty Workload Distribution

    Report Summary: This query analyzes teaching, research, and administrative workloads across faculty.

    Query:

```
SELECT
    f.name as faculty_name,
    d.dept_name,
    f.designation,
    -- Teaching workload
    COUNT(DISTINCT sec.section_id) as sections_teaching,
    COUNT(DISTINCT ssa.student_id) as students_teaching,
    -- Research contributions
    COUNT(DISTINCT rp.project_id) as projects_leading,
    COUNT(DISTINCT fp.project_id) as projects_involved,
    COUNT(DISTINCT pm.student_id) as research_students,
    -- Administrative roles
    COUNT(DISTINCT CASE WHEN d.head_of_dept = f.name THEN d.dept_id END)
as departments_heading,
    -- Overall contribution score
    (COUNT(DISTINCT sec.section_id) * 2 +
     COUNT(DISTINCT rp.project_id) * 3 +
     COUNT(DISTINCT fp.project_id) * 2 +
```

```
     COUNT(DISTINCT CASE WHEN d.head_of_dept = f.name THEN d.dept_id END)
* 5) as contribution_score
FROM Faculty f
LEFT JOIN Departments d ON f.dept_id = d.dept_id
LEFT JOIN Sections sec ON f.faculty_id = sec.faculty_id AND sec.year =
2024
LEFT JOIN StudentSectionAssignments ssa ON sec.section_id = ssa.section_id
LEFT JOIN ResearchProjects rp ON f.faculty_id = rp.faculty_id
LEFT JOIN FacultyProjects fp ON f.faculty_id = fp.faculty_id
LEFT JOIN ProjectMembers pm ON rp.project_id = pm.project_id
WHERE f.status = 'Active'
GROUP BY f.name, d.dept_name, f.designation
HAVING COUNT(DISTINCT sec.section_id) > 0 OR COUNT(DISTINCT rp.project_id)
> 0
ORDER BY contribution_score DESC;
```

Query Output & business decision: (The Output is too long to provide here) the university should:

- Balance workloads to prevent burnout
- Recognize high contributors for rewards
- Provide support for overloaded faculty

12. **Report Name:** Top 10 Most Borrowed Books

Report Summary: This query identifies the most frequently borrowed books across all departments.

Query:

```
SELECT TOP 10
    lb.title,
    lb.author,
    d.dept_name,
    COUNT(bi.issue_id) as times_borrowed,
    COUNT(DISTINCT bi.student_id) as different_students
FROM LibraryBooks lb
LEFT JOIN BookIssues bi ON lb.book_id = bi.book_id
LEFT JOIN Departments d ON lb.dept_id = d.dept_id
WHERE bi.issue_id IS NOT NULL
GROUP BY lb.title, lb.author, d.dept_name
ORDER BY times_borrowed DESC;
```

Query Output & business decision:

title	author	dept_name	times_borrowed	different_students

Est voluptates itaque rem.	Ikbal Anand	Chemical Engineering	9	9
Molestiae optio.	Chameli Madan	Physics	9	9
Odit fuga quaerat iusto iure.	Vritti Nori	Electrical & Electronic Engineering	9	9
Sit ipsa minus repudiandae.	Triveni Sarkar	Electrical & Electronic Engineering	9	9
Ab quod.	Faraj Badal	Civil Engineering	8	8
Earum ad ratione quisquam.	Kamya Dave	Electrical & Electronic Engineering	8	8
Eligendi temporibus corrupti.	Finn Chowdhury	Computer Science & Engineering	8	8
In saepe nemo.	Balendra Mammen	Electrical & Electronic Engineering	8	8
Inventore similique esse molestiae officia.	Henry Deep	Mathematics	8	8
Itaque iste soluta.	Tanay Lall	Electrical & Electronic Engineering	8	8

the library should:

- Purchase additional copies of top 10 books
- Create digital access for high-demand titles
- Use this data for collection development planning

Business Value & Recommendations:

Business Value:

- **Data Integrity & Reliability:** The reliability of references is ensured by 28 foreign key limitations, which make all database relationships legitimate and consistent. Students can't sign up for classes that don't exist, and professors can't be put in charge of departments that don't exist. This basic reliability keeps data from getting messed up and makes sure that all reports, from academic transcripts to financial summaries, are correct and trustworthy. Operational consistency between departments stops records from getting lost and data from being contradictory, which would have needed human reconciliation.
- **Improved Decision-Making:** Twelve analytical reports turn data into useful information. Administrators may now see patterns in how well students do in university, such as which

classes have high grades and which ones need to be improved. Reports on how resources are used reveal how many people are in a classroom and how popular library books are. This helps with budgeting and space planning. Analysis of faculty productivity looks at research and teaching loads to make sure that tasks are fairly distributed and that top performers are recognized. This makes judgments based on facts, which cuts down on guesswork and boosts academic and operational success.

- **Operational Efficiency:** Because the database structure is simpler, the RevenueSummary table and duplicate grade storage are no longer needed. It's easier to enter and keep up with data. Indexed tables with set relationships make it faster to find student, course, and financial information. IT staff can focus on making strategic enhancements instead of fixing data inconsistencies because maintenance is easier. Automated reporting saves administrative staff hours of working with spreadsheets.
- **Strategic Visibility:** University authorities are able to view everything that is going on in real time. This whole picture demonstrates how often classrooms are used affects faculty teaching loads, how much money is spent on library resources affects departmental research, and how well students do on tests affects attendance. Integrated visibility makes it possible to plan ahead, make sound policy decisions, and use resources wisely in a way that satisfies institutional goals and accreditation standards.

Recommendations for the University:

- **Implement Regular Database Audits:** On a quarterly basis, database administrators should review the quality of the data, the health of the schema, and the effectiveness of the constraints. Each audit should look at things like foreign key relationships, data type consistency, and normalization issues. This kind of maintenance keeps operational databases from getting worse and makes sure the system can adapt to changing needs at the institution. Make a checklist for an audit to check the accuracy of primary keys, foreign key associations, index performance, and important table data.
- **Adopt a Business Intelligence Tool:** Using Power BI or Tableau to connect the SQL database and make dynamic dashboards that show important KPIs in real time. Role-based dashboards should provide department heads their areas and administrators an overview of the whole institution. Use dashboards to track how well students are doing, how often

rooms are used, how to balance teacher workloads, and how to analyze library resources. These solutions make complicated data easy for people who aren't technical to understand, which leads to data-informed leadership meetings and quicker responses to trends.

- **Train Administrative Staff:** Teach staff how to analyze and understand data in stages. First-line staff should know the best ways to enter data and check its accuracy. Administrators should run and go over regular reports. It is necessary to teach academic advisers how to use student progress dashboards. Concentrate on useful subjects such as "How to check classroom availability for next semester" and "How to spot students who are at risk of failing." Offer workshops in person and lessons that have been recorded. Investing in people gets the most out of technology.
- **Automate Report Delivery:** Use SQL Server Agent jobs to plan when to make and send out essential report emails. Every week, department heads and administrators should acquire information about attendance, room use, and semester grades. This lets you get information quickly without having to do anything yourself. Set up a subscription system so that users can pick which reports they want and how often they want them. Executive summaries of automated reports should show the most important outcomes and suggest what to do next.

Conclusion:

This project turned a university management database that was formerly defective and inconsistent into a strong, standardized, and analytical system. By making methodical changes to primary keys, data types, foreign keys, and ERD relationships, a solid base for managing data was built. The twelve reports turn raw data into strategic insights, which help people make decisions based on facts in the areas of academics, administration, and operations. The university may further improve its data-driven culture, make operations more open, and promote long-term institutional goals by following the suggestions given.