DATABASE DESIGN CASE-STUDY

DATE

3RD FEBRUARY 2025

CREATED BY

EMMANUEL FALONIPE

TABLE OF CONTENT

INTRODUCTION

The Smart Academy for Innovative Technology (SAIT) is a premier e-learning platform dedicated to providing students with access to exceptional educational opportunities in a digitally advanced environment. Our programs leverage cutting-edge online learning systems, enabling seamless connections between students and specialist instructors. Through meticulously designed courses and robust evaluation processes, SAIT fosters an engaging and effective digital learning experience tailored to meet diverse academic needs.

MISSION

Our mission is to develop an intelligent and integrated data management system that enhances institutional operations and academic efficiency. This system will:

- Automate student enrollment and administrative processes.
- Manage payments securely and systematically.
- Utilize real-time analytics for examinations, enabling students to track their progress through live monitoring.

Through these innovations, we aim to foster precision, streamline workflows, and elevate the academic standards of the institution.

OBJECTIVE

To design and implement an intelligent institutional system that enhances operational efficiency and academic outcomes by:

- Establishing precise and accurate attendance tracking mechanisms.
- Automating and streamlining payment management processes.
- Providing tools to monitor and improve student academic performance through data-driven insights.

PRELIMINARY LIST

~~STUDENT~~
DEPARTMENT
CERTIFICATE
~~ENROLLMENT~~
~~COURSE~~
~~INSTRUCTOR~~
~~ATTENDANCE~~
PAYMENT
~~EXAM~~

SUBJECT LIST

STUDENT
INSTRUCTOR
ENROLLMENT
PAYMENT METHOD
ATTENDANCE
COURSE
EXAM

ENTITY DESCRIPTION

Student Table

Field	Data Type	Description
Student ID (PK)	Integer	Unique student identifier
Full Name	String	Student's full name
Address	String	Residential address
Date of Birth (DOB)	Date	Birthdate of the student
Gender	String	Gender information
Email	String	Email address

Attendance Table

Field	Data Type	Description
Attendance ID (PK)	Integer	Unique attendance record ID
Enroll ID (FK)	Integer	References Enrollment Table
Course ID (FK)	Integer	References Course Table
Date	Date	Date of attendance record
Status	String	Present, Absent, Late, etc.

Enrollment Table

Field	Data Type	Description
Enroll ID (PK)	Integer	Unique enrollment identifier
Student ID (FK)	Integer	References Student Table
Payment ID (FK)	Integer	References Payment Table
Instructor ID (FK)	Integer	References Instructor Table
Enroll Date	Date	Enrollment date
Status	String	Enrollment status
Course ID (FK)	Integer	References Course Table

Course Table

Field	Data Type	Description
Course ID (PK)	Integer	Unique course identifier
Name	String	Course name
Description	String	Course details
Price	Float	Course fee
Duration	String	Course duration

Exam Table

Field	Data Type	Description
Exam ID (PK)	Integer	Unique exam identifier
Course ID (FK)	Integer	References Course Table
Exam Type	String	Exam format (MCQ, written, etc.)
Exam Date	Date	Scheduled exam date
Grades	Float	Performance score

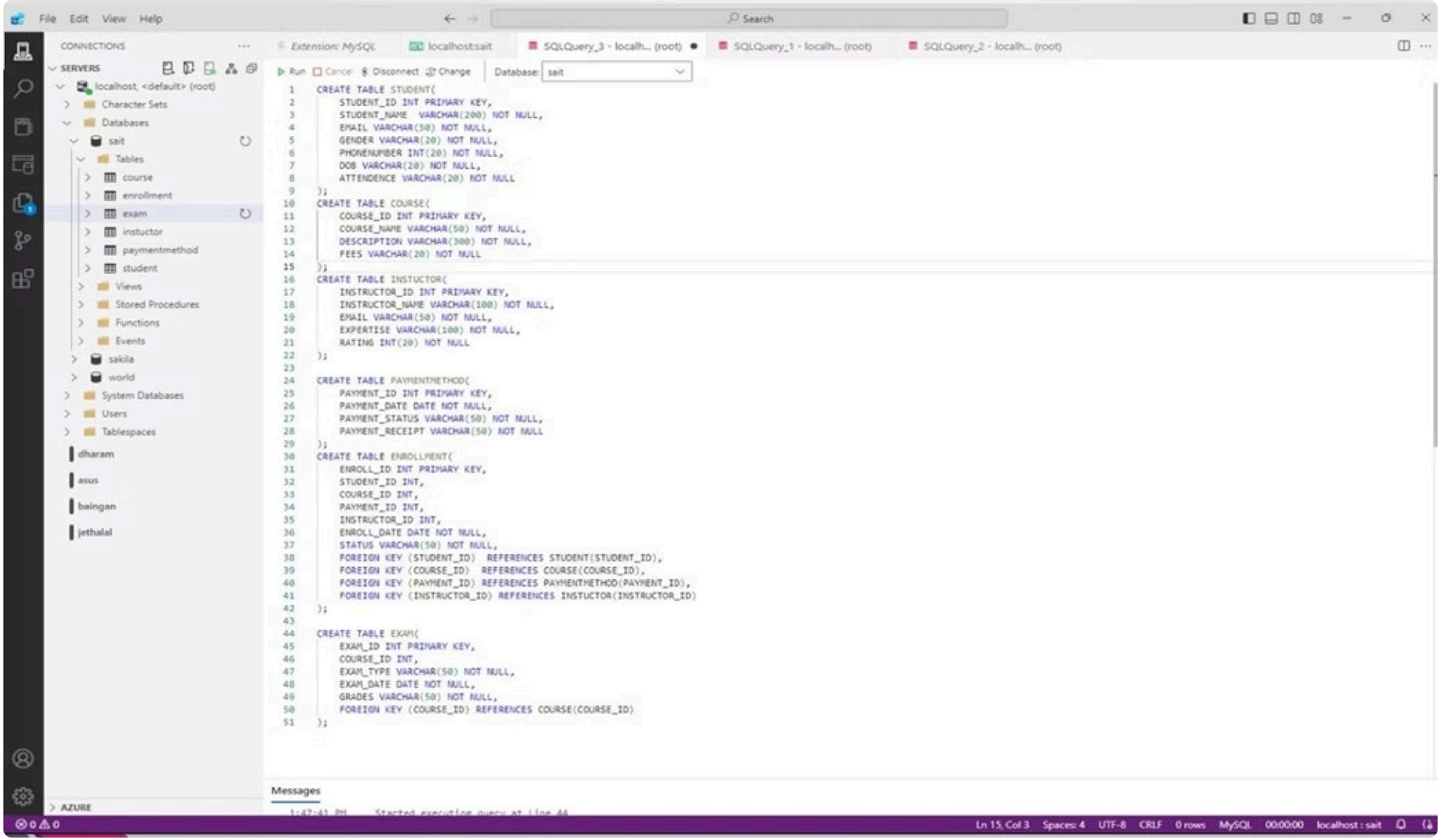
Payment Management

Field	Data Type	Description
Payment ID (PK)	Integer	Unique payment identifier
Payment Date	Date	Date of payment
Payment Status	String	Payment confirmation status
Payment_method	ENUM('credit card','paypal','bank transfer')	The mode of payment used to make a payment

Attendance Table

Field Name	Data Type	Description
Attendance_ID	INT PRIMARY KEY AUTO_INCREMENT	Unique identifier for each attendance record
Course_ID	INT (Foreign Key)	References the Course table to track which course the attendance is for
Date	DATE	The date when attendance was recorded
Status	ENUM('Present', 'Absent', 'Excuse')	Status of attendance for the course

DATABASE CREATION



```sql
CREATE TABLE STUDENT(
    STUDENT_ID INT PRIMARY KEY,
    STUDENT_NAME  VARCHAR(200) NOT NULL,
    EMAIL VARCHAR(50) NOT NULL,
    GENDER VARCHAR(20) NOT NULL,
    PHONENUMBER INT(20) NOT NULL,
    DOB VARCHAR(20) NOT NULL,
    ATTENDENCE VARCHAR(20) NOT NULL
);
CREATE TABLE COURSE(
    COURSE_ID INT PRIMARY KEY,
    COURSE_NAME VARCHAR(50) NOT NULL,
    DESCRIPTION VARCHAR(300) NOT NULL,
    FEES VARCHAR(20) NOT NULL
);
CREATE TABLE INSTUCTOR(
    INSTRUCTOR_ID INT PRIMARY KEY,
    INSTRUCTOR_NAME VARCHAR(100) NOT NULL,
    EMAIL VARCHAR(50) NOT NULL,
    EXPERTISE VARCHAR(100) NOT NULL,
    RATING INT(20) NOT NULL
);

CREATE TABLE PAYMENTMETHOD(
    PAYMENT_ID INT PRIMARY KEY,
    PAYMENT_DATE DATE NOT NULL,
    PAYMENT_STATUS VARCHAR(50) NOT NULL,
    PAYMENT_RECEIPT VARCHAR(50) NOT NULL
);
CREATE TABLE ENROLLMENT(
    ENROLL_ID INT PRIMARY KEY,
    STUDENT_ID INT,
    COURSE_ID INT,
    PAYMENT_ID INT,
    INSTRUCTOR_ID INT,
    ENROLL_DATE DATE NOT NULL,
    STATUS VARCHAR(50) NOT NULL,
    FOREIGN KEY (STUDENT_ID)  REFERENCES STUDENT(STUDENT_ID),
    FOREIGN KEY (COURSE_ID)  REFERENCES COURSE(COURSE_ID),
    FOREIGN KEY (PAYMENT_ID) REFERENCES PAYMENTMETHOD(PAYMENT_ID),
    FOREIGN KEY (INSTRUCTOR_ID) REFERENCES INSTUCTOR(INSTRUCTOR_ID)
);

CREATE TABLE EXAM(
    EXAM_ID INT PRIMARY KEY,
    COURSE_ID INT,
    EXAM_TYPE VARCHAR(50) NOT NULL,
    EXAM_DATE DATE NOT NULL,
    GRADES VARCHAR(50) NOT NULL,
    FOREIGN KEY (COURSE_ID) REFERENCES COURSE(COURSE_ID)
);
```

```
1:47:41 PM     Started executing query at Line 44
```

ENTITY RELATIONSHIP

· Student → Enrollment (one-to-many): A single student can register for multiple courses.

· Course → Enrollment (one-to-many): Multiple students can enroll in a single course.

· Enrollment → Attendance (one-to-many): Each enrollment is associated with multiple attendance records.

· Course → Attendance (one-to-many): Attendance data can be recorded for multiple sessions within a single course.

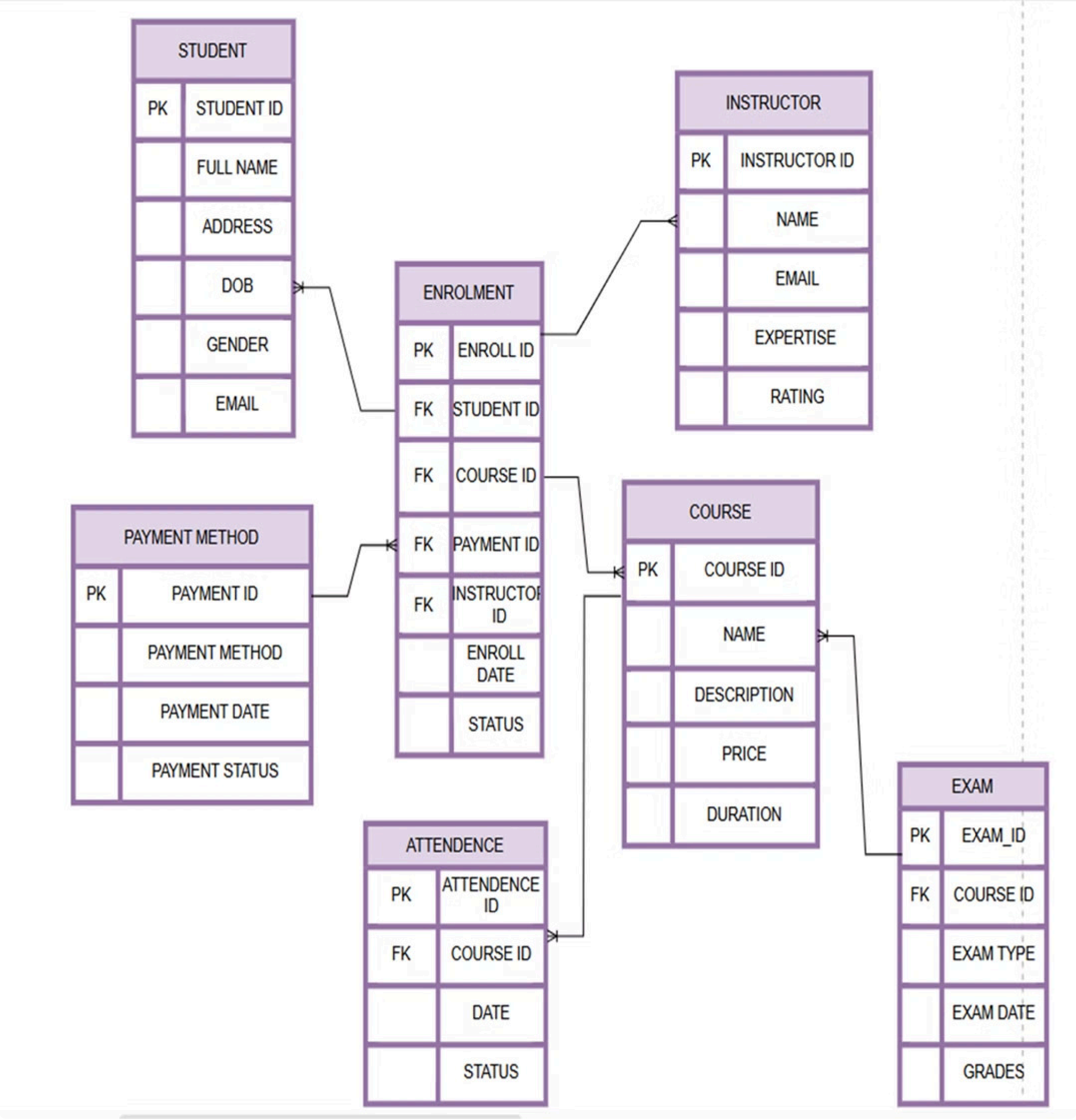
· Payment Method → Enrollment (one-to-many): Each enrollment is tied to one specific payment method.

· Instructor → Enrollment (one-to-many): An instructor manages several student enrollments.

· Course → Exam (one-to-many): A course can include various assessments.

· Course → Attendance (one-to-many): A course contains multiple attendance entries for its sessions.

ER DIAGRAM



STUDENT	
PK	STUDENT ID
	FULL NAME
	ADDRESS
	DOB
	GENDER
	EMAIL

INSTRUCTOR	
PK	INSTRUCTOR ID
	NAME
	EMAIL
	EXPERTISE
	RATING

ENROLMENT	
PK	ENROLL ID
FK	STUDENT ID
FK	COURSE ID
FK	PAYMENT ID
FK	INSTRUCTOR ID
	ENROLL DATE
	STATUS

PAYMENT METHOD	
PK	PAYMENT ID
	PAYMENT METHOD
	PAYMENT DATE
	PAYMENT STATUS

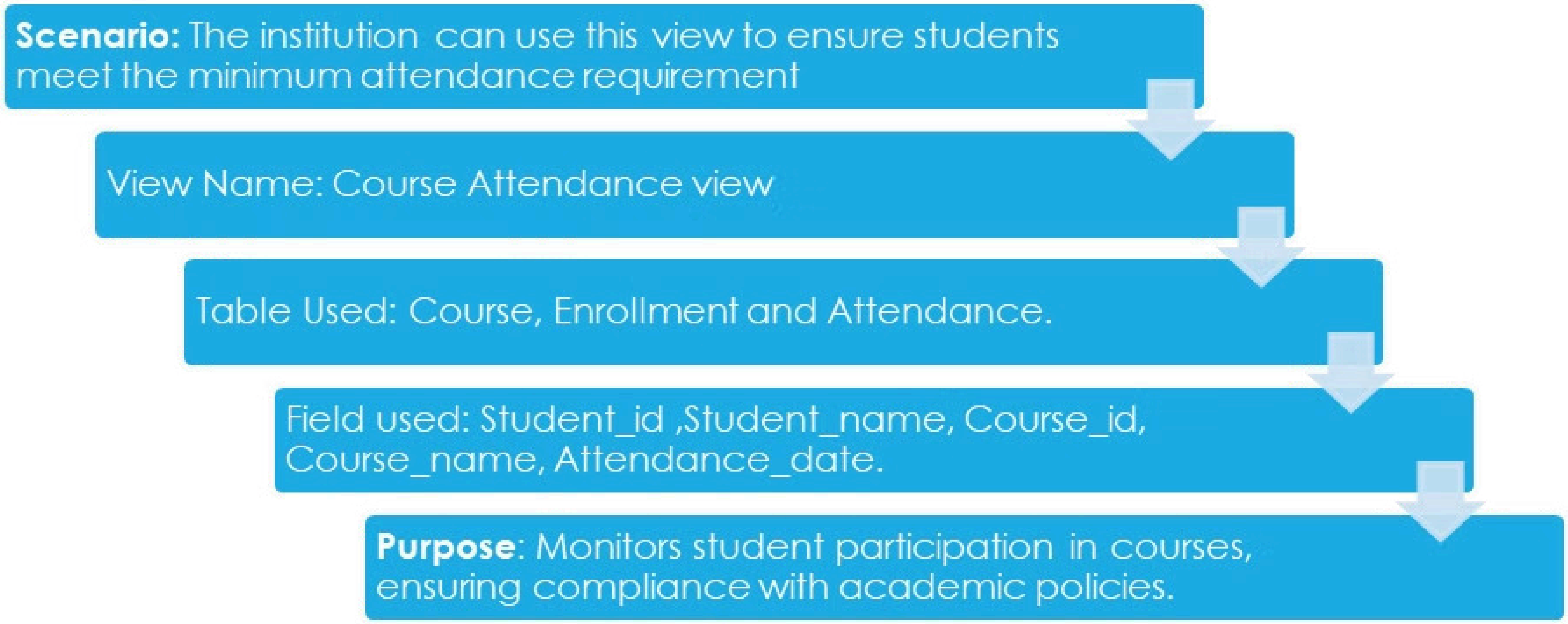
COURSE	
PK	COURSE ID
	NAME
	DESCRIPTION
	PRICE
	DURATION

ATTENDENCE	
PK	ATTENDENCE ID
FK	COURSE ID
	DATE
	STATUS

EXAM	
PK	EXAM_ID
FK	COURSE ID
	EXAM TYPE
	EXAM DATE
	GRADES

Data View and Significance

1. Course Attendance View



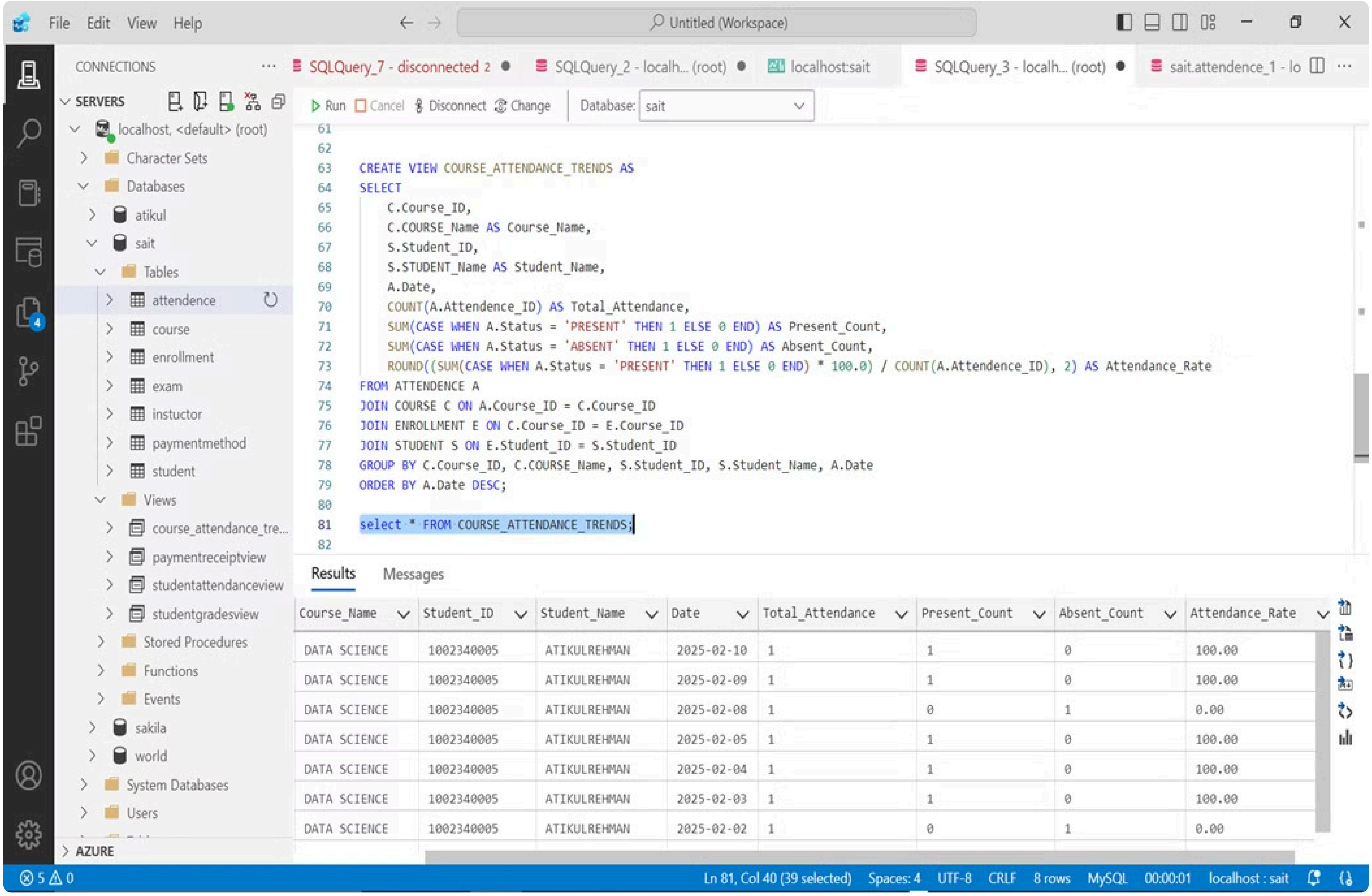
Scenario: The institution can use this view to ensure students meet the minimum attendance requirement

View Name: Course Attendance view

Table Used: Course, Enrollment and Attendance.

Field used: Student_id ,Student_name, Course_id, Course_name, Attendance_date.

Purpose: Monitors student participation in courses, ensuring compliance with academic policies.



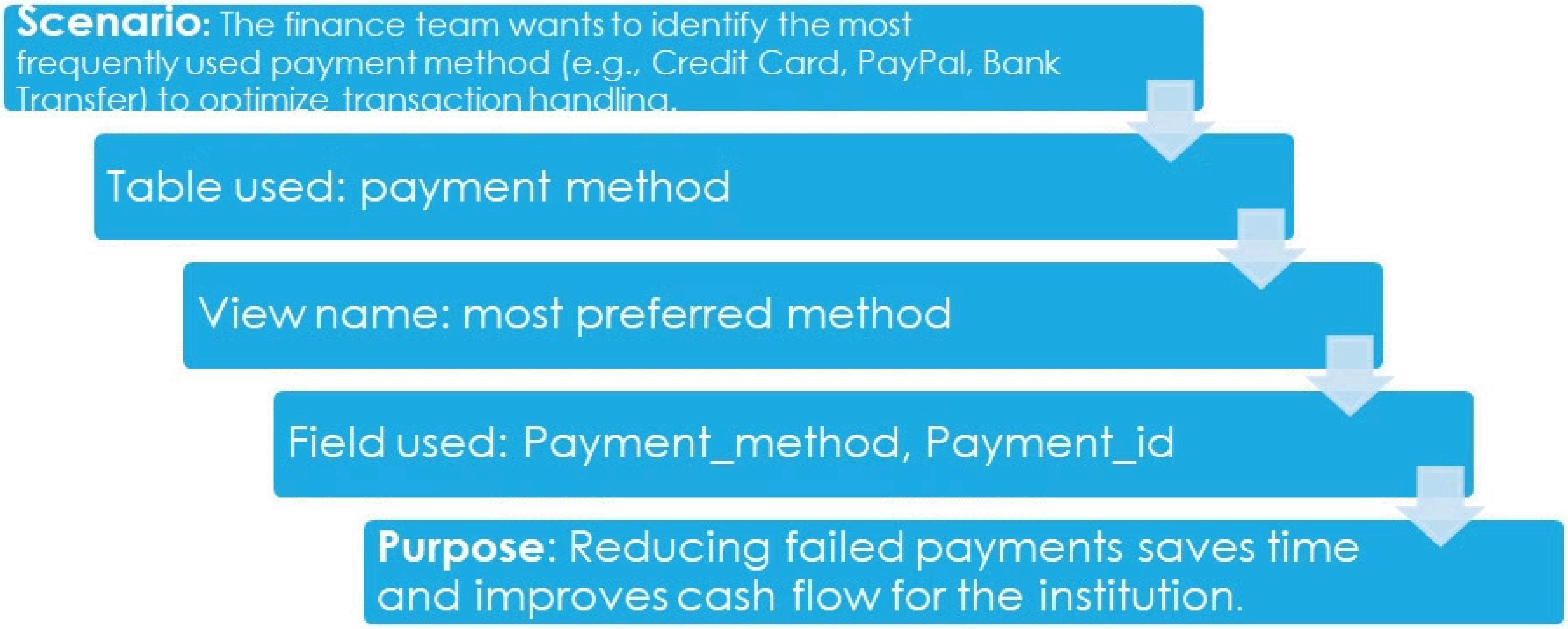
```sql

CREATE VIEW COURSE_ATTENDANCE_TRENDS AS
SELECT
    C.Course_ID,
    C.COURSE_Name AS Course_Name,
    S.Student_ID,
    S.STUDENT_Name AS Student_Name,
    A.Date,
    COUNT(A.Attendence_ID) AS Total_Attendance,
    SUM(CASE WHEN A.Status = 'PRESENT' THEN 1 ELSE 0 END) AS Present_Count,
    SUM(CASE WHEN A.Status = 'ABSENT' THEN 1 ELSE 0 END) AS Absent_Count,
    ROUND((SUM(CASE WHEN A.Status = 'PRESENT' THEN 1 ELSE 0 END) * 100.0) / COUNT(A.Attendence_ID), 2) AS Attendance_Rate
FROM ATTENDENCE A
JOIN COURSE C ON A.Course_ID = C.Course_ID
JOIN ENROLLMENT E ON C.Course_ID = E.Course_ID
JOIN STUDENT S ON E.Student_ID = S.Student_ID
GROUP BY C.Course_ID, C.COURSE_Name, S.Student_ID, S.Student_Name, A.Date
ORDER BY A.Date DESC;

select * FROM COURSE_ATTENDANCE_TRENDS;
```

Course_Name	Student_ID	Student_Name	Date	Total_Attendance	Present_Count	Absent_Count	Attendance_Rate
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-10	1	1	0	100.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-09	1	1	0	100.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-08	1	0	1	0.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-05	1	1	0	100.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-04	1	1	0	100.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-03	1	1	0	100.00
DATA SCIENCE	1002340005	ATIKULREHMAN	2025-02-02	1	0	1	0.00

2. Most preferred Payment Method



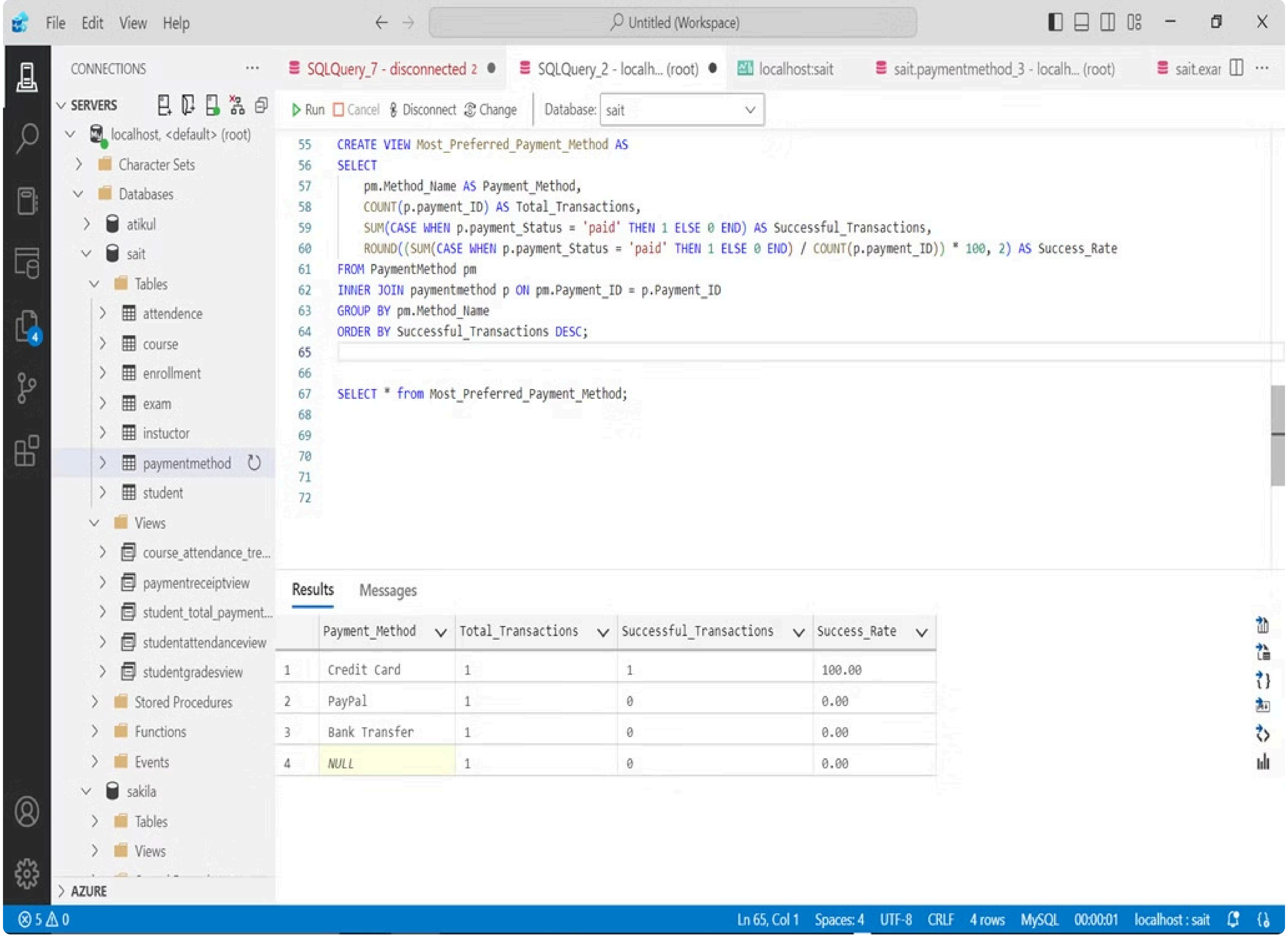
Scenario: The finance team wants to identify the most frequently used payment method (e.g., Credit Card, PayPal, Bank Transfer) to optimize transaction handling.

Table used: payment method

View name: most preferred method

Field used: Payment_method, Payment_id

Purpose: Reducing failed payments saves time and improves cash flow for the institution.

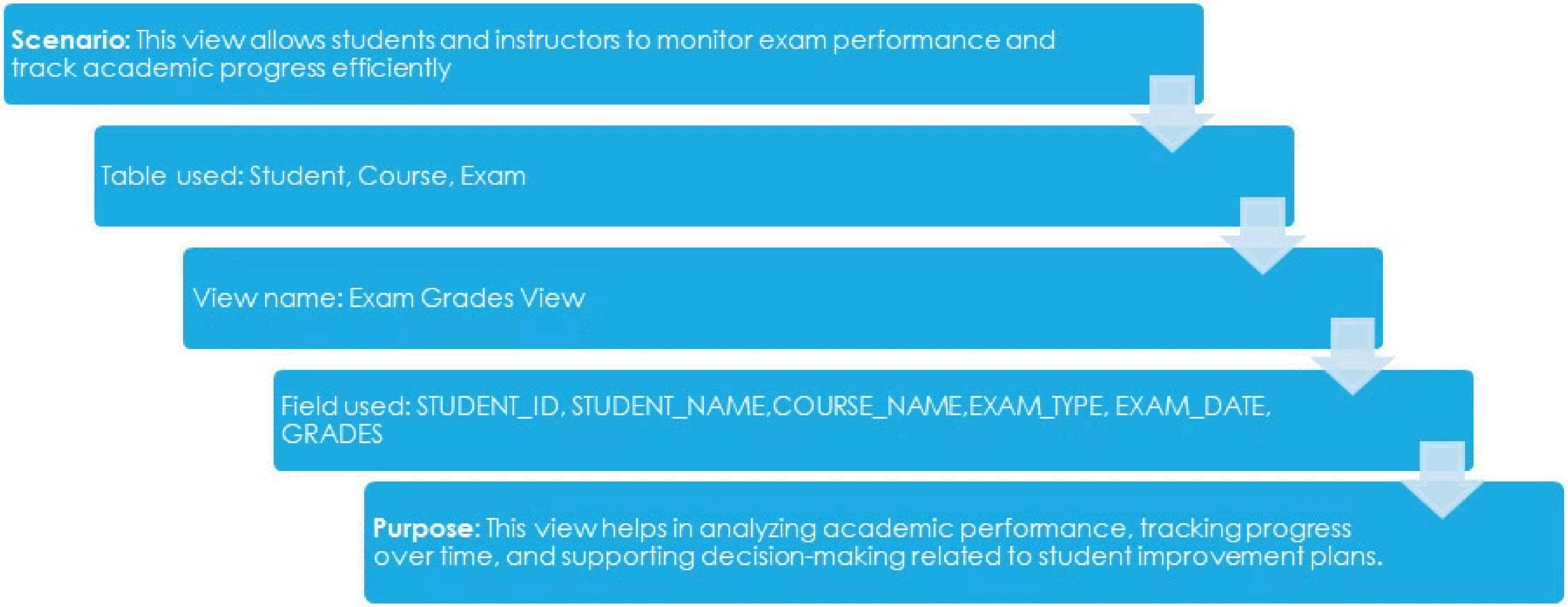


```sql
CREATE VIEW Most_Preferred_Payment_Method AS
SELECT
    pm.Method_Name AS Payment_Method,
    COUNT(p.payment_ID) AS Total_Transactions,
    SUM(CASE WHEN p.payment_Status = 'paid' THEN 1 ELSE 0 END) AS Successful_Transactions,
    ROUND((SUM(CASE WHEN p.payment_Status = 'paid' THEN 1 ELSE 0 END) / COUNT(p.payment_ID)) * 100, 2) AS Success_Rate
FROM PaymentMethod pm
INNER JOIN paymentmethod p ON pm.Payment_ID = p.Payment_ID
GROUP BY pm.Method_Name
ORDER BY Successful_Transactions DESC;


SELECT * from Most_Preferred_Payment_Method;
```

	Payment_Method	Total_Transactions	Successful_Transactions	Success_Rate
1	Credit Card	1	1	100.00
2	PayPal	1	0	0.00
3	Bank Transfer	1	0	0.00
4	NULL	1	0	0.00

3. Exam Grade View



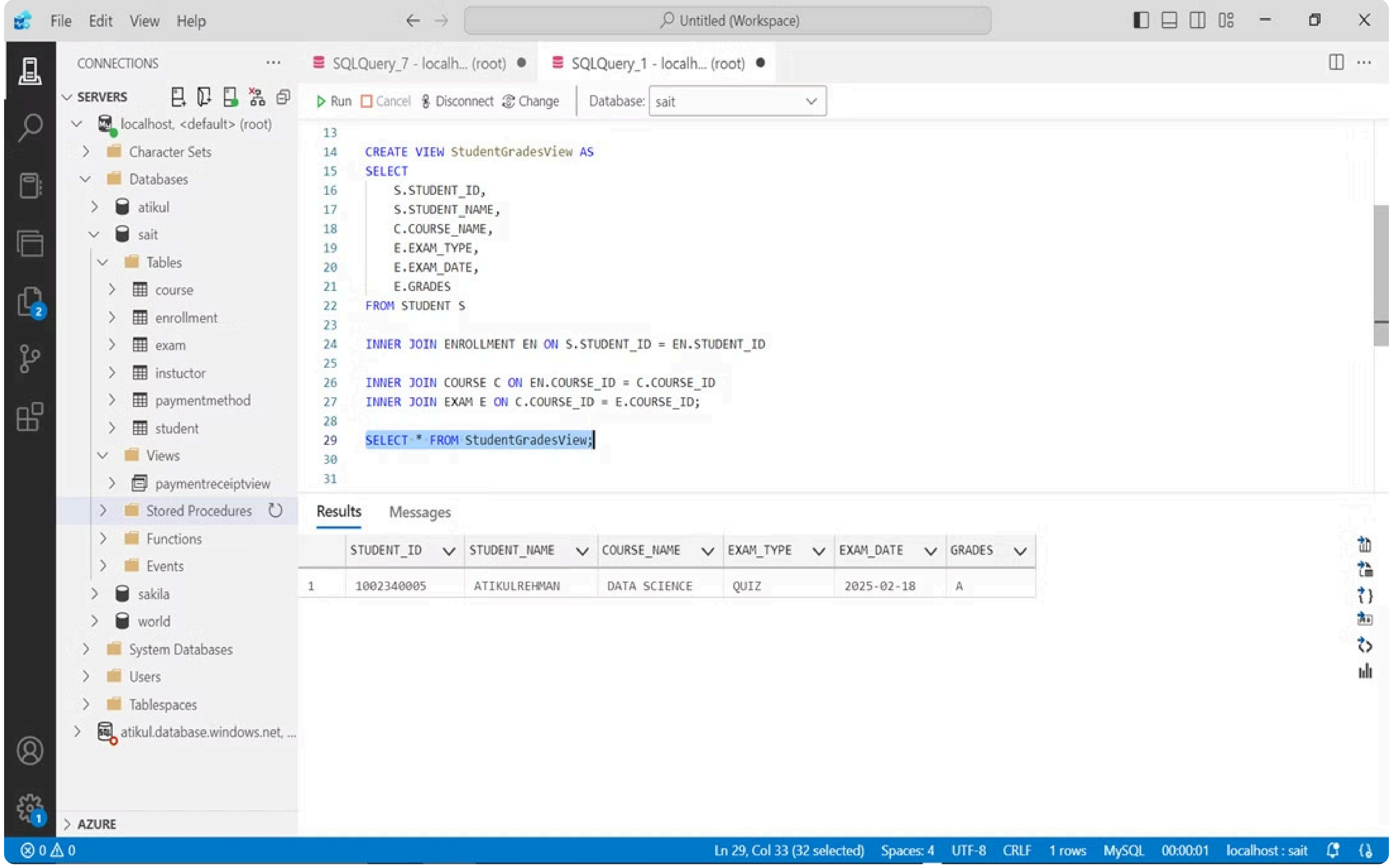
Scenario: This view allows students and instructors to monitor exam performance and track academic progress efficiently

Table used: Student, Course, Exam

View name: Exam Grades View

Field used: STUDENT_ID, STUDENT_NAME,COURSE_NAME,EXAM_TYPE, EXAM_DATE, GRADES

Purpose: This view helps in analyzing academic performance, tracking progress over time, and supporting decision-making related to student improvement plans.



```sql
CREATE VIEW StudentGradesView AS
SELECT
    S.STUDENT_ID,
    S.STUDENT_NAME,
    C.COURSE_NAME,
    E.EXAM_TYPE,
    E.EXAM_DATE,
    E.GRADES
FROM STUDENT S

INNER JOIN ENROLLMENT EN ON S.STUDENT_ID = EN.STUDENT_ID

INNER JOIN COURSE C ON EN.COURSE_ID = C.COURSE_ID
INNER JOIN EXAM E ON C.COURSE_ID = E.COURSE_ID;

SELECT * FROM StudentGradesView;
```

	STUDENT_ID	STUDENT_NAME	COURSE_NAME	EXAM_TYPE	EXAM_DATE	GRADES
1	1002340005	ATIKULREHMAN	DATA SCIENCE	QUIZ	2025-02-18	A

Key Benefits

The system provides effective control over student records that include maintaining attendance registers along with grades and financial payments.

- Accurate and real-time data tracking.
- Simplified financial transaction processing.
- The solution provides educational institutions with both transparent operations and efficient administrative practices.
- The institutional data supply enables better instructor-student relationship by providing organized analytical findings.

Future Enhancement

- Mobile Accessibility: Develop a mobile-friendly interface for seamless access.
- The system provides automatic notification services for important educational and financial information.
- The system gives users access to advanced analytics which help them create detailed performance and institutional development reports.

Conclusion

The SAIT system operates as a strong solution for making e-learning more efficient by optimizing data management. Our team continually works on enhancements because our main objective includes improving both institutional efficiency and academic performance tracking capabilities as well as financial transparency. We acknowledge your interest in SAIT software. Future system enhancements together with updates will be available to users.

THANK YOU
SMART
THANK YOU
SMART ACADEMY
FOR INNOVATIVE TECHNOGY
THANK YOU
THANK YOU
SMART ACADEMY
FOR INNOVATIVE TEVOLOGY