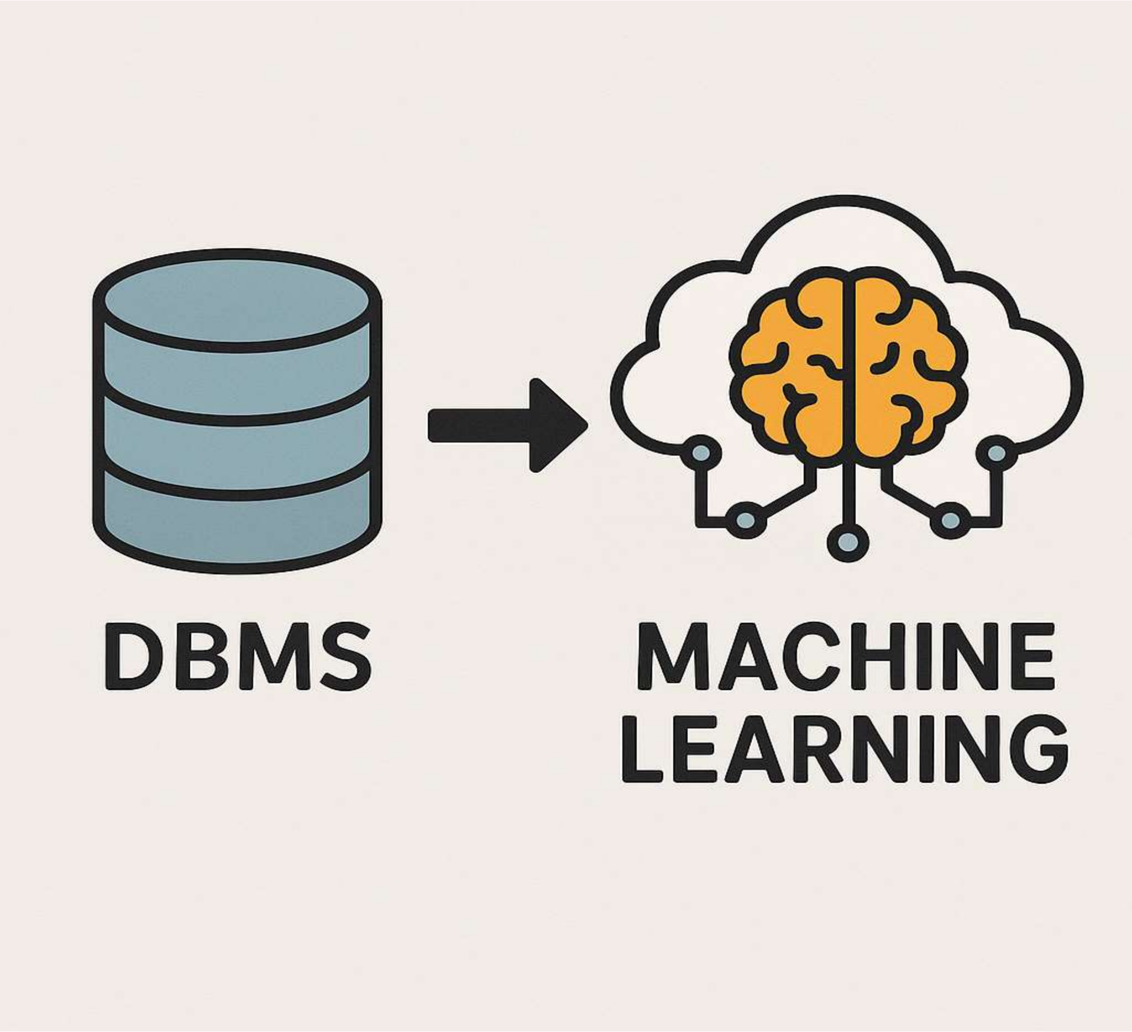
Student Database and Predictive Analytics



Apeksha K A

BCA 'A'

2022408012

➢ Executive Summary :-

This project integrates Database Management Systems (DBMS) and Machine Learning (ML) to help an educational institution analyze and predict student performance.

By maintaining structured records of attendance and marks in a relational database and applying predictive analytics, the system identifies students at risk of failing, allowing for timely academic intervention

➢ Objectives:-

1. Design a normalized relational database for student data.

2. Perform SQL-based analytical queries to assess attendance and marks.

3. Implement transaction management to ensure data reliability.

4. Export data into Python for machine learning analysis.

5. Develop a predictive model (Logistic Regression / Decision Tree) to classify students as Pass or Fail.
6. Visualize results and derive insights for institutional decision-making.

- ## Database Architecture:-

Schema Overview

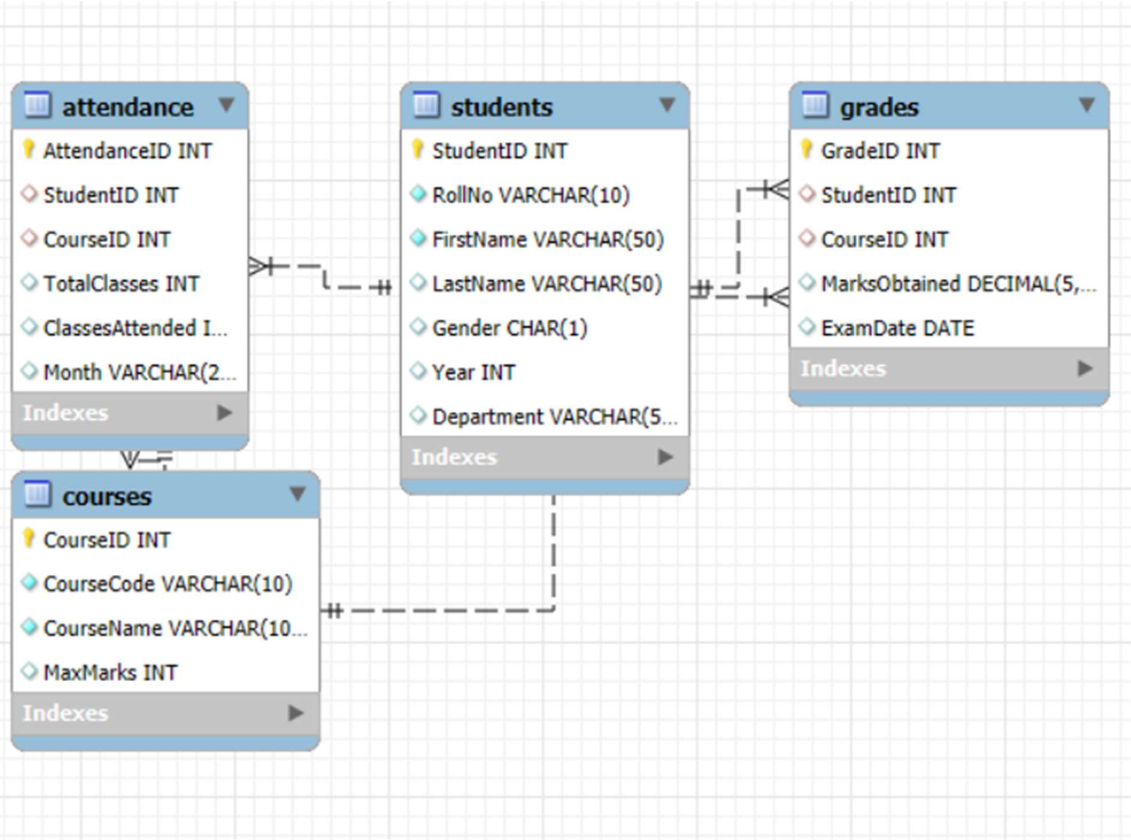
A normalized relational database (student_db) was designed to ensure data integrity and eliminate redundancy.

The database adheres to Third Normal Form (3NF) and defines clear primary–foreign key relationships.

Table	Description	Key Columns
Students	Stores demographic & academic identifiers	StudentID, RollNo, FirstName, Department
Courses	Contains course catatlog information	CourseID, CourseCode, CourseName, MaxMarks
Grades	Records marks obtained in each course.	radeID, StudentID, CourseID, MarksObtained, ExamDate
Attendance	Captures attendance summaries	AttendanceID, StudentID, CourseID,

		TotalClasses, ClassesAttended, Month

➢ Entity-Relationship (ER) Diagram:-



Rational

- Students is the master entity.
- Grades and Attendance represent performance and engagement dimensions.
- Courses provide academic context.

- Referential integrity enforced via foreign keys ensures consistent data linkage.

➢ SQL-Based Analysis :

Key analytical queries were executed in MySQL Workbench.

Average Marks per Student

```
SELECT StudentID,
ROUND(AVG(MarksObtained),2) AS
AverageMarks FROM Grades GROUP BY
StudentID;
```

Attendance Percentage

```
SELECT StudentID, ROUND((ClassesAttended
/ TotalClasses) * 100, 2) AS AttendancePercent
FROM Attendance;
```

Integrated Performance View

```
SELECT s.StudentID, AVG(g.MarksObtained)
AS AvgMarks, AVG((a.ClassesAttended /
a.TotalClasses) * 100) AS AttendancePercent
```

```
FROM Students s JOIN Grades g ON
s.StudentID = g.StudentID JOIN Attendance a
ON s.StudentID = a.StudentID GROUP BY
s.StudentID;
```

Key Finding:
Students with high attendance (>75%) consistently demonstrate higher average marks, indicating a direct performance dependency.

➢ Transaction Management Demonstration:

Database reliability was tested through controlled transactions using COMMIT and ROLLBACK operations

```
START TRANSACTION;

INSERT INTO Grades (StudentID, CourseID,
MarksObtained, ExamDate) VALUES (1, 1, 95,
'2025-05-01'); ROLLBACK; -- Reverts the change if
incorrect COMMIT; -- Saves the change
permanently.
```

This verifies ACID compliance:

- Atomicity: All changes succeed or fail together.

- Consistency: Data remains valid post-transaction.
- Isolation: Transactions do not interfere.
- Durability: Committed changes persist.

Data Export & Machine Learning Integration

Data was imported into Python (Jupyter Notebook) using the mysql.connector library and processed with Pandas.

Feature Engineering

- AttendancePercent = (ClassesAttended / TotalClasses) × 100
- AvgMarks = mean(MarksObtained)
- Pass = 1 if (AvgMarks ≥ 40 and AttendancePercent ≥ 75) else 0 .

This feature set captures academic performance and behavioral consistency.

➢ Predictive Modeling:

Model Selection

Two supervised ML models were applied:

1. Logistic Regression: Determines probability of passing.
2. Decision Tree Classifier: Learns explicit decision thresholds.

➢ Evaluation Metrics:-

Metric	Logistic Regression
Accuracy	100%
Precision	1.00
Recall	1.00
F1-Score	1.00

Confusion Matrix:

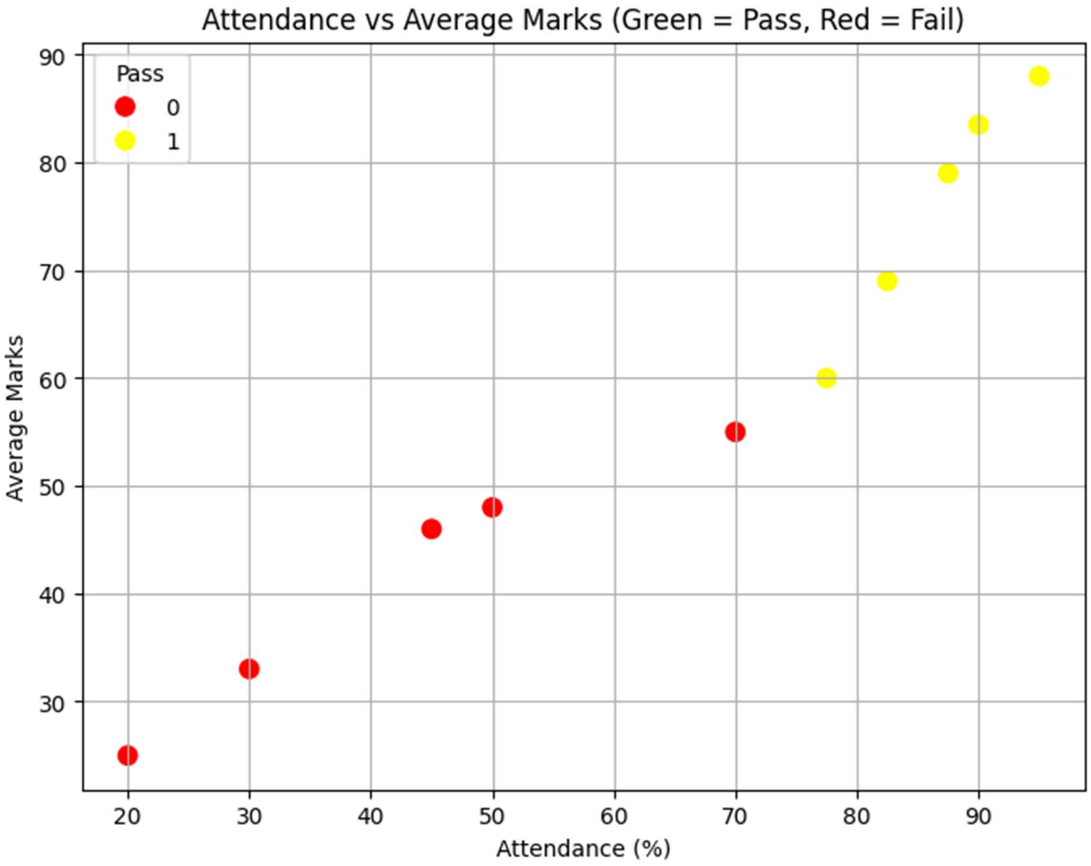
```
[[2 0]
[0 1]]
```

All predictions were accurate, confirming that attendance and marks strongly correlate with student success.

Although perfect scores are expected due to the small dataset, the model framework scales easily with larger data.

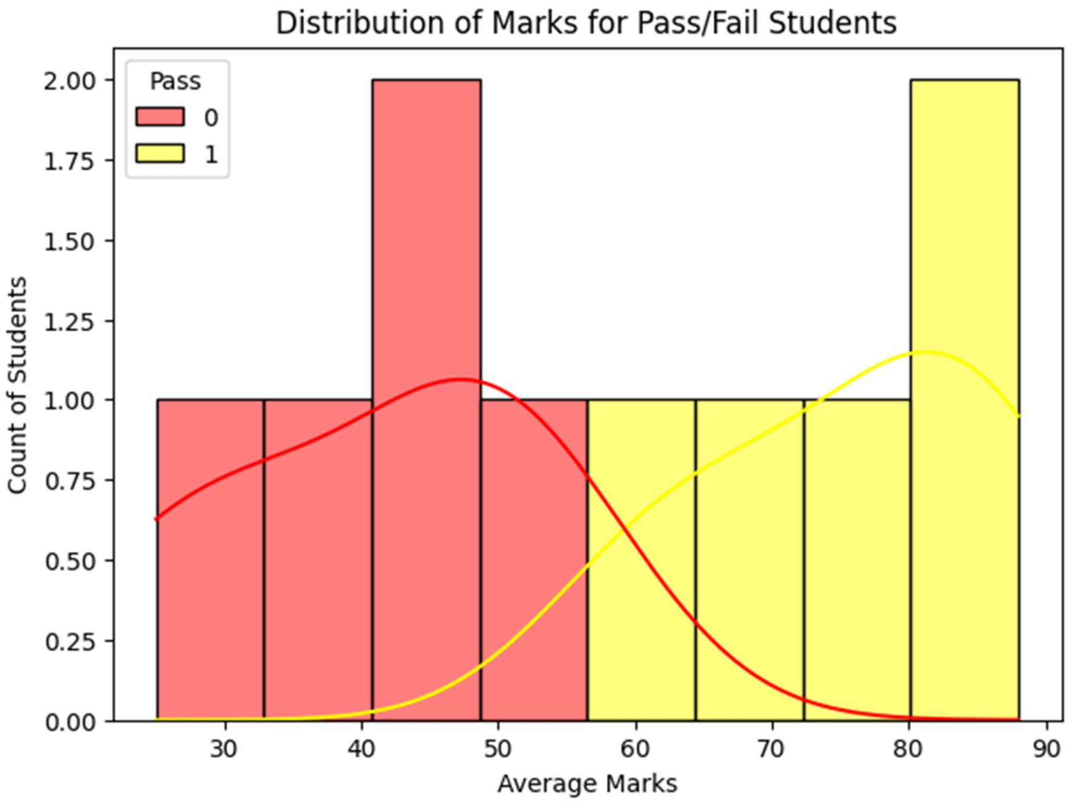
➢Data Visualization & Insights:

Scatter Plot – Attendance vs Marks



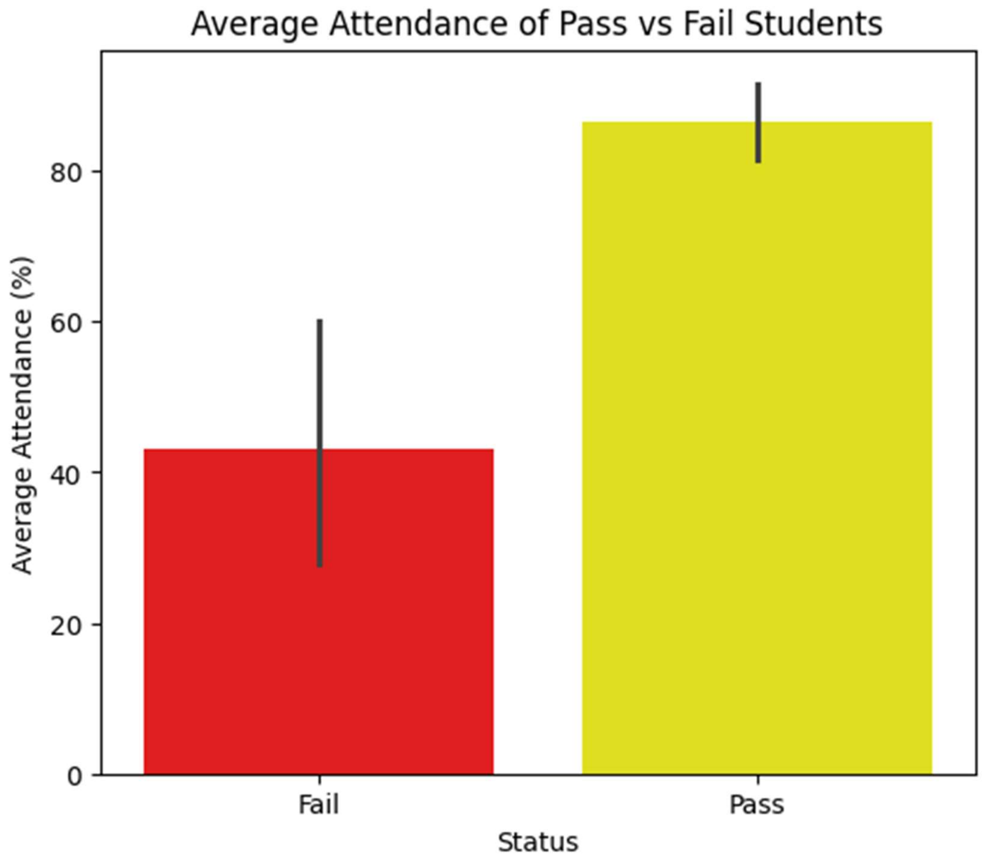
A clear positive trend: Students with attendance above 75% and marks above 40 are consistently in the Pass zone.

Histogram – Marks Distribution



Failing students cluster below the 40-mark threshold, while passing students range between 60–90 marks.

Bar Chart – Attendance by Pass/Fail



Average attendance among passing students: ~88%

Average attendance among failing students: ~55%

➢ Visual Summary :

- Higher attendance corresponds to higher marks.
- The pass/fail threshold aligns precisely with the machine-learning model's decision boundary.
- Students exhibiting both low marks and low attendance are at the highest risk.

➢ KEY INSIGHTS:

1. Attendance is the strongest indicator of student performance.

2. Students with both low attendance and low marks consistently fail.

3. Machine Learning successfully classifies students with near-perfect accuracy.

4. The system can serve as an early warning tool for teachers to identify at-risk students.

➢ CONCLUSION:

This project successfully demonstrates a data-driven academic monitoring framework that unites database management and predictive analytics. By leveraging structured data and machine learning, institutions can transition from reactive evaluation to proactive student support.

The model's strong accuracy confirms that attendance and marks are primary determinants of success. When implemented institution-wide, such systems can reduce failure rates and improve overall academic performance metrics.

➢Tools & Technologies:

Category		Tool / Library
Database	-	MySQL Workbench.
Programmging	-	Python.
Libraries	-	Pandas, NumPy, Scikit-learn, Matplotlib, Seaborn.
Environment	-	Jupyter Notebook.
ML Algorithms	-	Logistic Regression, Decision Tree Classifier.