رواد مصر الرقمية

Final Project Documentation

StudentSight

Student Performance Dashboard

Team Members:

Abdelrahman Mohamed Fathi

Sabry Tarek Sabry

Ahmed Osama Ahmed

Ahmed Moustafa Mahmoud

Yasser Elsayed Mohamed

Esraa Mahmoud Abdelrahman

Table of Contents

1.Project Planning & Overview

1.1 Project Name:

StudentSight

We chose StudentSight because it merges our project's "Student"-centric data focus with the "Sight" (or *insight*) our dashboard provides, enabling educators to clearly "see" performance trends and make informed decisions.

1.2 Executive Summary

The StudentSight project aims to solve the problem of fragmented and underutilized student academic data. It establishes a complete data pipeline using Python (Pandas) for data preprocessing and cleaning, SQL for relational data storage and structured querying, and visualization tools for reporting. The final dashboard centralizes information on scores, attendance, and overall performance, transforming raw data into clear, actionable insights. This system empowers teachers and administrators to make data-driven decisions, enabling proactive interventions and ultimately improving student outcomes.

1.3 Project Scope & Problem Statement

Problem Statement

Schools and universities often store critical student performance data in scattered, unstructured formats (e.g., local spreadsheets or manual records). This leads to several inefficiencies:

- Delayed Identification: Teachers struggle to track performance trends over time, delaying the identification of students requiring support.
- Lack of Correlation: Administrators lack structured data to analyze the relationship between factors like attendance patterns and academic performance.
- Untimely Feedback: Students receive feedback too late to take effective corrective action.

The lack of a centralized, analytical system limits opportunities to improve learning outcomes.

Project Scope

The project covers the development of a complete ETL (Extract, Transform, Load) and reporting pipeline:

1. Extraction & Transformation: Collecting raw student datasets and using Python/Pandas for cleaning, transformation, and feature engineering.
2. Loading & Storage: Designing a normalized relational database schema and loading the cleaned data into an SQL environment.
3. Analysis & Visualization: Developing key SQL queries to generate insights and creating a reporting dashboard to visualize trends.

1.4 Key Project Goals

1. Establish a scalable and maintainable data pipeline from raw data to actionable insights.
2. Normalize and centralize student records into a secure, queryable SQL database.
3. Generate specific analytical reports on performance trends, top performers, and attendance correlations.
4. Develop a user-friendly dashboard to empower teachers and administrators with timely, data-driven feedback.

1.5 Technologies Used

1. Data ingestion and Cleaning: pyspark
2. Database and Storage: MySQL and PostgreSQL
3. Containerization**:** Docker
4. Visualization and Reporting: Power BI
5. Version Control: Git and GitHub

2. Stakeholder Analysis & Team Structure

2.1 Project Team Roles and Responsibilities

Team Member	Role	Responsibilities
Abdelrahman Mohamed	Team Leader	Coordinates team activities, delegates responsibilities, tracks progress against deadlines, and facilitates clear communication with all stakeholders.
Sabry Tarek	PySpark ETL Developer& Containerization Lead	Developed and implemented PySpark ETL scripts for large-scale data cleaning and transformation, utilizing Docker for hosting database services.
Ahmed Osama Sabry Tarek	SQL Database Architect	Designing the normalized database schema in both MySQL and PostgreSQL, and ensuring data migration integrity.
Ahmed Mostafa Abdelrahman Mohamed Esraa Mahmoud	Data Visualization	Implementing the final dashboard using Power BI, focusing on report design and data connection to PostgreSQL.
Esraa Mahmoud yasser elsayed	Documentation and persentation	Developing and finalizing all project documentation (reports, summaries), creating presentation materials (slides)

2.2 External Stakeholders

Stakeholders	**Role in the Project**
Data Organization (school)	Source of the raw student performance data.
Teachers & Instructors	Primary end-users; providing feedback on the utility and design of performance reports.
School Administrators	Users of aggregated data; evaluating the dashboard's effectiveness for curriculum and resource planning.
End Users (Students)	Indirectly benefiting from the system; gaining access to timely, personalized feedback.

3. Database Design

3.1 Overview

The PostgreSQL database is structured using a heavily normalized Dimensional Model optimized for analytical reporting. This approach separates descriptive data (Dimension Tables) from measurable data (Fact Tables), allowing for fast aggregation and complex querying (e.g., slice-and-dice analysis in Power BI).

3.2 Database Structure

Main Collections: Dimension Tables (Descriptive Data)

dimstudent Stores student demographics and identity. Fields:

- student_id (Character varying(100) - Primary Key)
- student_name (Character varying(100))
- gender (Character varying(50))
- grade_character (Character varying(50))
- city (Character varying(100))
- family_id (Character varying(100) - Foreign Key to dimfamily)

dimteacher Stores instructor details and employment status. Fields:

- teacher_id (Character varying(100) - Primary Key)
- teacher_name (Character varying(500))
- phone_character (Character varying(50))
- hire_date (Date)
- department (Character varying(100))

dimcourse Stores course details and associated instructor/semester. Fields:

- course_id (Character varying(100) - Primary Key)
- course_name (Character varying(200))
- semester_character (Character varying(50))
- teacher_id (Character varying(100) - Foreign Key to dimteacher)

Fact Tables (Measured Data)

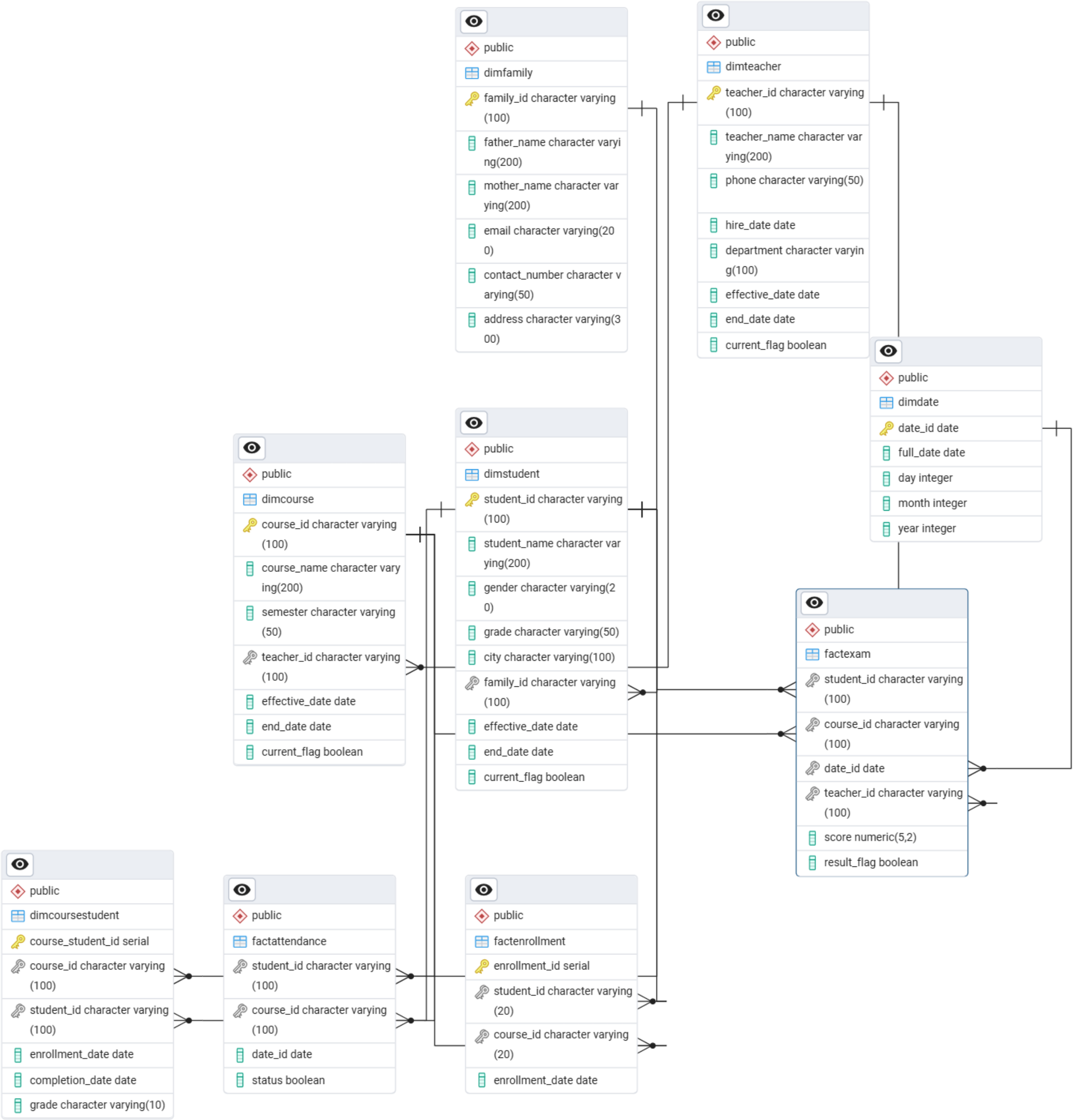
factexam Records of all exam results and scores. Fields:

- student_id (Character varying(100) - Composite Key/FK to dimstudent)
- course_id (Character varying(100) - Composite Key/FK to dimcourse)
- date_id (Date - Composite Key/FK to dimdate)
- score (Numeric(5,2) - Measured Metric)
- result_flag (Boolean - e.g., Passed/Failed)

factattendance Records of daily student attendance. Fields:

- student_id (Character varying(100) - Composite Key/FK to dimstudent)

3.3 ERD (Entity Relationship Diagram)


public
dimfamily
family_id character varying(100)
father_name character varying(200)
mother_name character varying(200)
email character varying(200)
contact_number character varying(50)
address character varying(300)
public
dimteacher
teacher_id character varying(100)
teacher_name character varying(200)
phone character varying(50)
hire_date date
department character varying(100)
effective_date date
end_date date
current_flag boolean
public
dimdate
date_id date
full_date date
day integer
month integer
year integer
public
dimcourse
course_id character varying(100)
course_name character varying(200)
semester character varying(50)
teacher_id character varying(100)
effective_date date
end_date date
current_flag boolean
public
dimstudent
student_id character varying(100)
student_name character varying(200)
gender character varying(20)
grade character varying(50)
city character varying(100)
family_id character varying(100)
effective_date date
end_date date
current_flag boolean
public
factexam
student_id character varying(100)
course_id character varying(100)
date_id date
teacher_id character varying(100)
score numeric(5,2)
result_flag boolean
public
dimcoursestudent
course_student_id serial
course_id character varying(100)
student_id character varying(100)
enrollment_date date
completion_date date
grade character varying(10)
public
factattendance
student_id character varying(100)
course_id character varying(100)
date_id date
status boolean
public
factenrollment
enrollment_id serial
student_id character varying(20)
course_id character varying(20)
enrollment_date date

4. System Architecture

4.1 Overview

The system utilizes a scalable, multi-stage architecture designed for large-volume educational data. The process begins with raw data extraction, is processed by PySpark for scalability, moves through a two-tier database system hosted in Docker containers, and concludes with visualization in Power BI.

4.2 Architecture Layers and Flow

1. **Extraction & Ingestion:** Raw student performance data is extracted from the source organization.
2. **Transformation Layer (PySpark):** The raw data is loaded into a **PySpark** environment. This scalable, distributed framework performs the comprehensive data cleansing, feature engineering (e.g., calculating attendance rates, overall performance categories), and validation.
3. **Staging Layer (MySQL):** The cleaned, structured output from PySpark is loaded into **MySQL** for initial staging and basic historical retention.
4. **Analytical Loading (ETL):** A subsequent ETL process moves the validated data from MySQL into **PostgreSQL**.
5. **Analytical Warehouse (PostgreSQL): PostgreSQL** serves as the final, optimized analytical data store (Data Warehouse). This database is designed for complex relational querying and reporting.
6. **Analysis & Visualization Layer (Power BI): Power BI** connects directly to the **PostgreSQL** database, executing analytical queries and transforming the results into the final interactive dashboard for end-users.

5. Reporting and Visualization Design

5.1 Key Visualizations

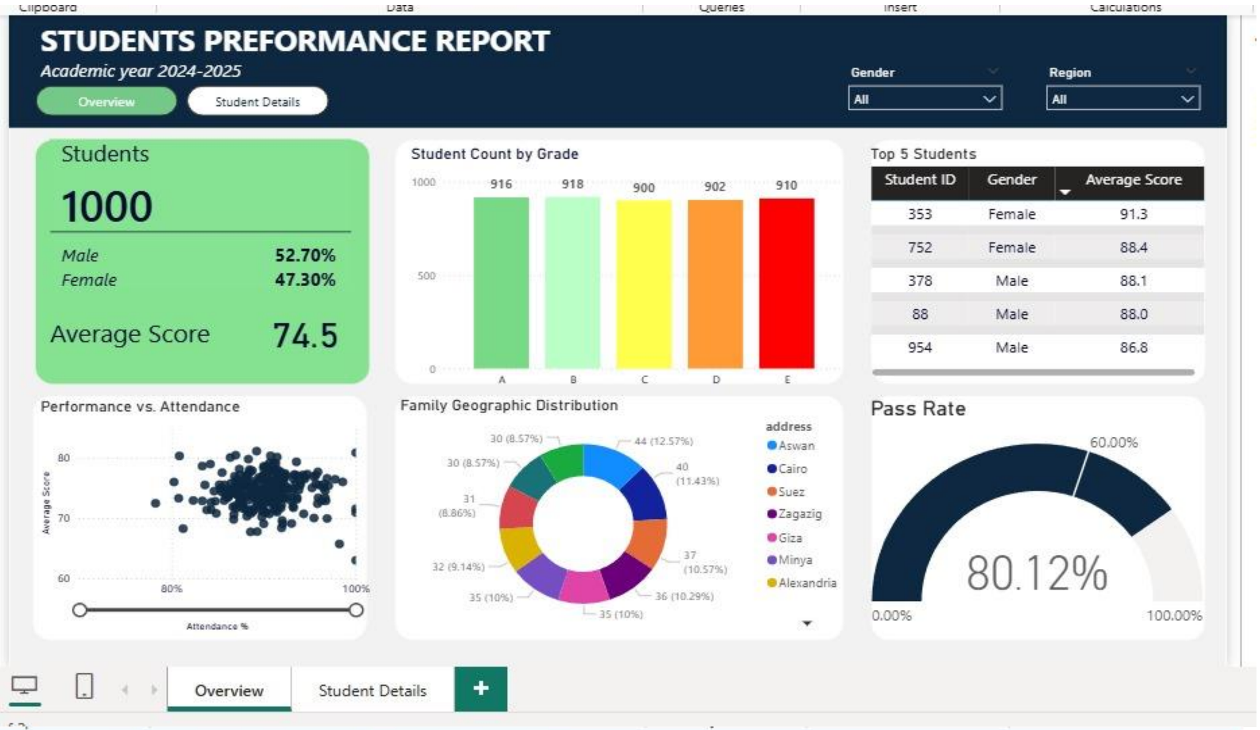
The main dashboard provides a comprehensive overview of system-wide performance and key indicators, broken down into the following visualization types:

1. **KPI Scorecards:** Displaying high-impact metrics such as **Total Students**, **Average Score.**
2. **Gender Distribution:** A metric display showing the breakdown of students by gender.
3. **Pass Rate Gauge:** A gauge highlighting the percentage of students meeting the pass threshold.
4. **Top Performers:** A **Table** listing the **Top 5 Students** based on Student ID, Gender, and their **Average Score**.
5. **Count by Grade Distribution:** A **Bar Chart** showing the **Student Count by Grade** (e.g., Grades A, B, C, D, E) to visualize enrollment distribution and performance clustering.
6. **Performance vs. Attendance Correlation:** A **Scatter Plot** showing the relationship between **Average Score** and **Attendance %** to identify potential correlations between presence and academic success.
7. **Family Geographic Distribution:** A **Pie Chart** visualizing the student population distribution across different **Geographic Regions**
8. **Detailed Student Details Page:** A comprehensive **Table/Matrix** view listing individual students, filterable by Student ID, Gender, and Region. This report details key metrics for each student, including Student Name, Family ID, City, Attendance %, Average Score, and assigned Grade, with data bars for quick visual comparison.

5.2 Dashboard Utility

The dashboard acts as the single source of truth, facilitating:

- Teacher Drill-Down: The ability to filter reports by individual student, subject, or assessment type.
- Administrator Overview: Aggregated views of performance across grade levels and sections.
- Proactive Alerts: Visual flags to identify students whose scores are below a set threshold .



STUDENTS PREFORMANCE REPORT

Academic year 2024-2025

Students

1000

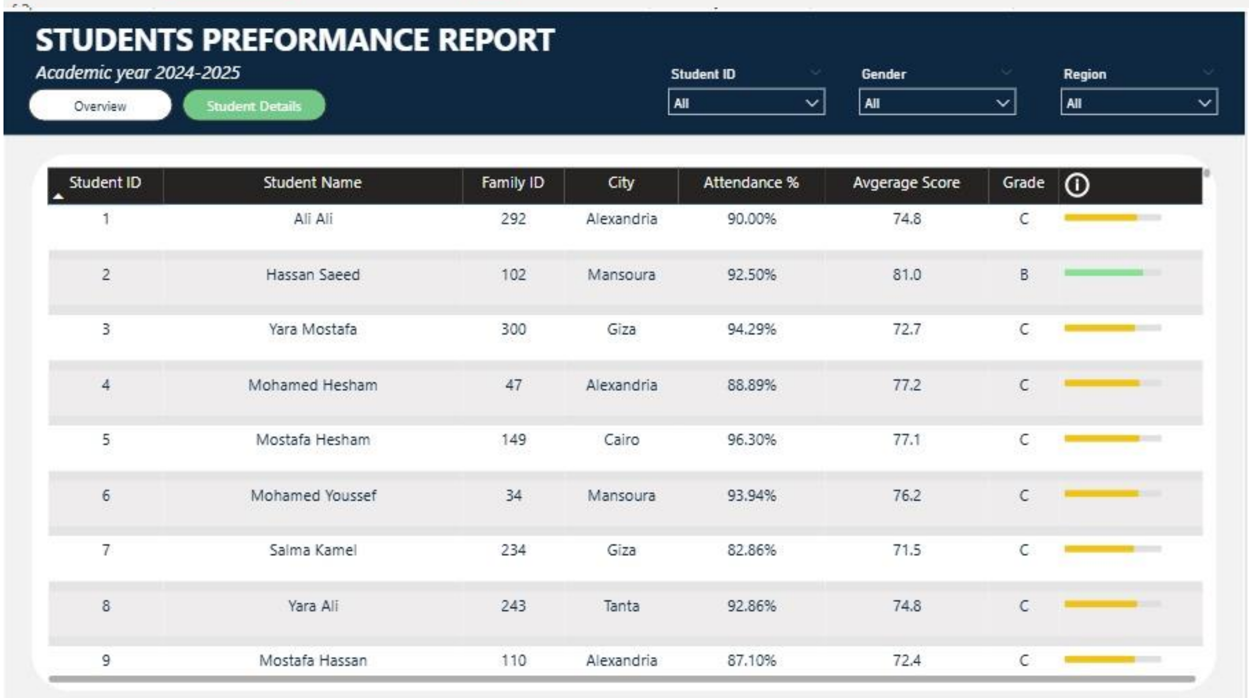
Male 52.70%

Female 47.30%

Average Score 74.5

Top 5 Students

Student ID	Gender	Average Score
353	Female	91.3
752	Female	88.4
378	Male	88.1
88	Male	88.0
954	Male	86.8



STUDENTS PREFORMANCE REPORT

Academic year 2024-2025

Student ID	Student Name	Family ID	City	Attendance %	Avgerage Score	Grade	ⓘ
1	Ali Ali	292	Alexandria	90.00%	74.8	C	
2	Hassan Saeed	102	Mansoura	92.50%	81.0	B	
3	Yara Mostafa	300	Giza	94.29%	72.7	C	
4	Mohamed Hesham	47	Alexandria	88.89%	77.2	C	
5	Mostafa Hesham	149	Cairo	96.30%	77.1	C	
6	Mohamed Youssef	34	Mansoura	93.94%	76.2	C	
7	Salma Kamel	234	Giza	82.86%	71.5	C	
8	Yara Ali	243	Tanta	92.86%	74.8	C	
9	Mostafa Hassan	110	Alexandria	87.10%	72.4	C	

6. Project Timeline

Phase	Duration	Core Deliverable
Data Collection & Preprocessing	Week 1 - 2	PySpark ETL Script, Cleaned Staged Data (in MySQL)
SQL Integration & Querying	Week 3 - 4	ER Diagram, PostgreSQL Database Schema Scripts, Core Analytical Query Results
Visualization & Reporting	Week 5 - 6	Final Power BI Dashboard Prototype (Charts and Layout)
Final Documentation & Presentation	Week 7	Final Report (PDF), Presentation

7. Conclusion & Future Enhancements

Conclusion

The StudentSight project successfully implements a complete, scalable data pipeline that centralizes, cleans, and analyzes student performance metrics. By leveraging the distributed processing power of PySpark and the analytical querying capabilities of PostgreSQL and Power BI, the system moves educational reporting from static record-keeping to dynamic, real-time intelligence. The core objective of providing actionable, data-driven feedback to improve student outcomes has been met through the development of robust analytical queries and intuitive visualizations.

Future Enhancements

The current project establishes a strong foundation. Potential future enhancements could include:

- **Personalized Student Dashboards:** Developing a dedicated, individual-level dashboard interface to provide students with **direct, secure access** to their personal scores, attendance logs, and progress against class averages, promoting self-correction and accountability.
- **Machine Learning Integration:** Using **PySpark MLlib** to integrate predictive models for identifying students at high risk of failure based on early-term data (scores and attendance).
- **Automated Data Ingestion:** Setting up a dedicated orchestration tool (e.g., Apache Airflow) to manage the scheduled, automated flow of data from the source organization through PySpark, MySQL, and PostgreSQL.
- **Web Application Interface:** Developing a dedicated web front-end to embed Power BI reports and provide custom data interaction capabilities.

GitHub Link:

https://github.com/nhahub/NHA-163