

SQL School

Database Project

(MySQL)

Project Description

This project demonstrates the design and analysis of a **School Management Database** using **MySQL**.

The database models real-world academic entities such as students, courses, enrolments, and progress tracking.

Using SQL joins, aggregation functions, and filtering techniques, meaningful insights are derived from relational data.

Database Schema

Schema Name: school

Tables Used:

- student
- course
- enrollment
- progress

Key Relationships:

- One student → multiple enrollments
 - One course → multiple students
 - Enrollment → linked to progress data
-

Tools & Technologies

- **Database:** MySQL
 - **IDE:** MySQL Workbench
 - **Concepts:**
 - Joins (INNER JOIN)
 - Aggregate Functions (SUM, AVG)
 - GROUP BY
 - WHERE & Date Filters
 - Relational Modeling
-

Steps Followed

1. Designed relational database schema

2. Created tables with primary & foreign keys
 3. Inserted sample data
 4. Established relationships between tables
 5. Wrote SQL queries for analysis
 6. Validated results using MySQL Workbench
-

SQL Queries Performed

- Total course price per student
 - Average course completion percentage per student
 - Students enrolled in multiple courses
 - Course-wise enrollment analysis
 - Students accessing courses after a specific date
 - Revenue analysis using enrollments
-

Sample Query

```
SELECT
    s.student_id,
    s.student_name,
    SUM(c.price) AS total_course_price
FROM student s
JOIN enrollment e ON s.student_id = e.student_id
JOIN course c ON e.course_id = c.course_id
GROUP BY s.student_id, s.student_name;
```

Key Insights

- Students often enroll in multiple courses, increasing total revenue
 - Average completion percentage helps identify engaged learners
 - Course pricing directly impacts total student spending
 - Enrollment data helps track course popularity
-

Conclusions

This project showcases strong understanding of:

- Relational database design
- SQL joins & aggregations
- Real-world data analysis using SQL

da 1 evening ×

File Edit View Query Database Server Tools

Navigator

SCHEMAS

Filter objects

school

Tables

course

enrollment

Columns

Indexes

Foreign Keys

Triggers

progress

student

Views

Stored Procedures

Administration Schemas

Information

Table: enrollment

Columns:

enrollment_id	int PK
student_id	varchar(40)
course_id	varchar(40)
enroll_date	date

MySQL Workbench - da1 evening

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

SCHEMAS: Filter objects

- north_region
- sales
- school
 - Tables: course, enrollment, progress, student
 - Views
 - Stored Procedures
 - Functions

Administration Schemas Information Schema: employees

Result Grid: Filter Rows Export: Wrap Cell Content

```
141 -- Q7: Find the total price of all courses each student is enrolled
142 •   SELECT
143     s.student_id,
144     s.student_name,
145     SUM(c.price) AS total_course_price
146   FROM student s
147   JOIN enrollment e
148     ON s.student_id = e.student_id
149   JOIN course c
```

student_id	student_name	total_course_price
1	Alice Johnson	4498
2	Bob Smith	4499
3	Charlie Brown	2999
4	Diana Prince	1799
5	Ethan Lee	4998

Result 2: Action Output

#	Time	Action	Message	Duration / Fetch
1	03:25:23	use school	0 row(s) affected	0.016 sec
2	03:25:30	SELECT s.student_name, s.student_email, c.course_title, e.enroll_date FROM Enrollment e...	7 row(s) returned	0.296 sec / 0.000 sec
3	03:26:36	SELECT s.student_id, s.student_name, SUM(c.price) AS total_course_price FROM student s...	5 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

16°C Clear

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

SCHEMAS: Filter objects

- north_region
- sales
- school
 - Tables: course, enrollment, progress, student
 - Views
 - Stored Procedures
 - Functions

Administration Schemas Information Schema: employees

Result Grid: Filter Rows Export: Wrap Cell Content

```
116 -- Q5: Find the average completion percentage for each student.
117
118 •   SELECT
119     s.student_id,
120     s.student_name,
121     AVG(p.completion_percent) AS avg_completion
122   FROM student s
123   JOIN enrollment e
124     ON s.student_id = e.student_id
125   JOIN progress p
126     ON e.enrollment_id = p.enrollment_id
127   GROUP BY s.student_id, s.student_name;
128
129 -- Q6: List students who accessed their course progress after '2025-08-04'.
130 •   SELECT
131     s.student_id,
132     s.student_name,
133     p.last_accessed_date
134   FROM student s
135   JOIN enrollment e
136     ON s.student_id = e.student_id
```

Action Output

#	Time	Action	Message	Duration / Fetch
1	03:25:23	use school	0 row(s) affected	0.016 sec
2	03:25:30	SELECT s.student_name, s.student_email, c.course_title, e.enroll_date FROM Enrollment e...	7 row(s) returned	0.296 sec / 0.000 sec

Object Info Session

16°C Clear

The screenshot displays two instances of MySQL Workbench, each showing a session named 'da1 evening'. Both sessions are connected to the same database, 'SQL School database project', and are executing the same SQL script.

Session 1 (Top):

- Information:** Shows the schema 'employees' is selected.
- Action Output:** Displays the results of the executed SQL script. The output shows two rows of data from the 'Enrolment' table.
- Object Info:** Shows the current object is 'Action Output'.

Session 2 (Bottom):

- Information:** Shows the schema 'employees' is selected.
- Action Output:** Displays the results of the executed SQL script. The output shows the same two rows of data from the 'Enrolment' table.
- Object Info:** Shows the current object is 'Result Grid'.

Script Content (Both Sessions):

```
87 -- Q2: Show the names of students from the city 'Mumbai' who are enrolled in any course.
88
89 •    SELECT DISTINCT s.student_name
90   FROM Student s
91   JOIN Enrollment e
92     ON s.student_id = e.student_id
93   WHERE s.city = 'Mumbai';
94
95
96 -- Q3: Count how many students are enrolled in each course.
97
98 •    SELECT
99       c.course_title,
100      COUNT(e.student_id) AS total_students
101     FROM course c
102     LEFT JOIN enrollment e
103       ON c.course_id = e.course_id
104     GROUP BY c.course_title;
105
106 -- Q4: Find all courses with more than 1 student enrolled.
107 •    SELECT
```