

Analyzing E-Learning Platform Purchases using MySQL

1. Project Overview:

The database captures key entities such as learners, courses, and purchases, enabling structured storage and efficient querying of transactional data. By leveraging SQL queries, the project aims to uncover valuable insights related to sales performance, learner purchasing behavior, and popular course categories across different countries.

Objective:

- Design a normalized relational database schema for learners, courses, and purchases using MySQL.
- Store and manage learner information across multiple countries.
- Track course purchases and analyze sales trends over time.
- Identify top-selling courses and popular course categories.
- Analyze learner behavior, including purchase frequency and spending patterns.
- Generate country-wise and category-wise revenue insights.
- Support business decision-making through meaningful SQL queries and reports.

2. Data Sources:

- **Source Description and Timeline:** online purchase by learners in 2026
 - **Domain:** Education
-

3. Problem Statement:

An online learning platform sells various digital courses to learners across different countries. Each learner can purchase multiple courses, and each course belongs to a specific category.

The management team wants to analyze purchase data to understand **sales trends, learner behavior, and popular course categories**.

Your task is to use **MySQL** to design and query a relational database that helps derive meaningful insights from this e-learning purchase data.

4. Attribute (Column /Features) Details:

Attribute Name	Data Type	Description
Learner_id	Integer	Unique identifier for each learner
Full_name	String (Text)	Name of the learner
country	String(Text)	Country of each learner

Course_id	Integer	Unique identifier for each course
Course_name	String(Text)	Name of the course
Category	String(Text)	Category of courses
Unit_price	Decimal	Price of the each course
Purchase_id	Integer	Unique identifier for each course purchased
Quantity	Integer	Quantity of each course purchased
Purchase_date	Date	Purchased date of each course

5. Tools & Technologies:

MySQL Workbench 8.0 CE

6. Data Exploration Using Join:

Data Presentation Guidelines for the following query

- Format currency values to **2 decimal places**.

→select round(unit_price,2) from courses;

- Use **aliases** for column names (e.g., AS total_revenue).

→select sum(unit_price) as total_revenue from courses;

- Sort results appropriately (e.g., highest total_spent first).

→select unit_price as highest_total_spend_first from courses order by unit_price

desc limit 1;

Use SQL INNER JOIN, LEFT JOIN, and RIGHT JOIN to:

- Combine learner, course, and purchase data.

→select * from purchases p

join learners l on p.learner_id=l.learner_id

join courses c on p.course_id=c.course_id;

- Display each learner's purchase details (course name, category, quantity, total amount, and purchase date).
→ select full_name, course_name, category, Quantity, unit_price, purchase_date from purchases p
left join courses c on p.course_id=c.course_id
left join learners l on p.learner_id=l.learner_id;

8. Analytical Queries:

- * Display each learner's total spending (quantity × unit_price) along with their country.
→ select l.learner_id, l.full_name, l.country, SUM(p.quantity * c.unit_price) as total_spending from learners l
join purchases p on l.learner_id = p.learner_id
join courses c on p.course_id = c.course_id
group by l.learner_id, l.full_name, l.country;
- * Find the top 3 most purchased courses based on total quantity sold.
→ select c.course_id, c.course_name, SUM(p.quantity) as total_quantity_sold from courses c
join purchases p on c.course_id = p.course_id
group by c.course_id, c.course_name
order by total_quantity_sold desc limit 3;
- * Show each course category's total revenue and the number of unique learners who purchased from that category.

```
→ select c.category,SUM(p.quantity * c.unit_price) as total_revenue,COUNT(DISTINCT p.learner_id) as unique_learners  
from courses c  
  
join purchases p on c.course_id = p.course_id  
  
group by c.category;
```

* **List all learners who have purchased courses from more than one category.**

```
→ select l.learner_id,l.full_name,COUNT(DISTINCT c.category) as category_count  
from learners l  
  
join purchases p on l.learner_id = p.learner_id  
  
join courses c on p.course_id = c.course_id  
  
group by l.learner_id,l.full_name  
  
having category_count > 1;
```

* **Identify courses that have not been purchased at all.**

```
→ select c.course_id,c.course_name,c.category  
from courses c  
  
left join purchases p on c.course_id = p.course_id  
  
where p.course_id is null;
```

9. Key Findings:

- A few top courses generate the **majority of total revenue**.
- **Professional & technical categories** are the most popular.
- Learners often purchase **multiple courses**, indicating repeat behavior.
- **Moderately priced courses** achieve the highest sales volume.
- Certain countries contribute **significantly more revenue** than others.
- Beginner-level courses attract **more enrollments** than advanced ones.
- Repeat learners have **higher lifetime value**.
- Category-wise performance varies **by country**.
- Cross-selling opportunities exist based on **purchase patterns**.
- Database structure supports **scalable sales and behavior analysis**.

- **Analysis insights:**

- **Descriptive:** Technical and beginner-level courses generate the highest sales and enrollments.
 - **Diagnosis:** High demand for job-oriented skills and affordable pricing drives purchases.
 - **Predictive:** Skill-based categories and repeat learners will continue to fuel revenue growth.
 - **Prescriptive:** Focus on top categories, bundle courses, and target high-performing countries.
-

10. Conclusions:

This SQL-based analysis provides clear insights into learner behavior, course performance, and sales trends on an e-learning platform. The results highlight key revenue-driving categories and high-value learners, enabling data-driven decisions for course planning, pricing, and targeted marketing.