**Experiment No. 2**

**Experiment Title: Channel & Pricing Analytics with Data Integrity (MySQL)**

**Problem Statement:**

A retail company wants consistent payment methods and sales channels, reliable line-item pricing, and quick analytics on revenue, AOV, and price deviations. Using the existing tables—customers, products, sales—extend the model minimally, enforce integrity in MySQL, and produce analytics that management can run monthly.

**Create Table**

create database manager;

use manager;

CREATE TABLE customers (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE,

phone VARCHAR(15),

created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE products (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

sku VARCHAR(50) UNIQUE NOT NULL,

category VARCHAR(50),

base\_price DECIMAL(10,2) CHECK (base\_price > 0),

active BOOLEAN DEFAULT TRUE

);

CREATE TABLE sales\_channels (

id INT AUTO\_INCREMENT PRIMARY KEY,

channel\_name VARCHAR(50) NOT NULL UNIQUE,

active BOOLEAN DEFAULT TRUE

);

CREATE TABLE payment\_methods (

id INT AUTO\_INCREMENT PRIMARY KEY,

method\_name VARCHAR(50) NOT NULL UNIQUE,

active BOOLEAN DEFAULT TRUE

);

CREATE TABLE sales (

id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT NOT NULL,

sale\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,

payment\_method\_id INT NOT NULL,

sales\_channel\_id INT NOT NULL,

total\_amount DECIMAL(12,2),

FOREIGN KEY (customer\_id) REFERENCES customers(id),

FOREIGN KEY (payment\_method\_id) REFERENCES payment\_methods(id),

FOREIGN KEY (sales\_channel\_id) REFERENCES sales\_channels(id)

);

CREATE TABLE sale\_line\_items (

id INT AUTO\_INCREMENT PRIMARY KEY,

sale\_id INT NOT NULL,

product\_id INT NOT NULL,

quantity INT CHECK (quantity > 0),

unit\_price DECIMAL(10,2) CHECK (unit\_price >= 0),

line\_total DECIMAL(12,2),

FOREIGN KEY (sale\_id) REFERENCES sales(id),

FOREIGN KEY (product\_id) REFERENCES products(id)

);

**Verify integrity using Trigger**

DELIMITER //

CREATE TRIGGER trg\_line\_total\_before\_insert

BEFORE INSERT ON sale\_line\_items

FOR EACH ROW

BEGIN

SET NEW.line\_total = NEW.quantity \* NEW.unit\_price;

END;

//

DELIMITER ;

**Insert Values into Tables**

INSERT INTO customers (name, email, phone, created\_at) VALUES

('Alice Johnson', 'alice.johnson@example.com', '9876543210', '2025-01-15 10:30:00'),

('Bob Smith', 'bob.smith@example.com', '9123456780', '2025-02-10 14:45:00'),

('Charlie Brown', 'charlie.brown@example.com', '9988776655', '2025-03-05 09:20:00'),

('Diana Prince', 'diana.prince@example.com', '9871203456', '2025-03-25 16:40:00'),

('Ethan Hunt', 'ethan.hunt@example.com', '9765432109', '2025-04-01 11:05:00');

INSERT INTO products (name, sku, category, base\_price, active) VALUES

('Wireless Mouse', 'SKU1001', 'Electronics', 800.00, TRUE),

('Gaming Keyboard', 'SKU1002', 'Electronics', 2500.00, TRUE),

('Office Chair', 'SKU1003', 'Furniture', 4500.00, TRUE),

('Water Bottle', 'SKU1004', 'Accessories', 300.00, TRUE),

('Bluetooth Speaker', 'SKU1005', 'Electronics', 1500.00, TRUE);

INSERT INTO sales\_channels (channel\_name, active) VALUES

('Online Store', TRUE),

('Retail Outlet', TRUE),

('Mobile App', TRUE),

('Wholesale', TRUE),

('Partner Store', TRUE);

INSERT INTO payment\_methods (method\_name, active) VALUES

('Credit Card', TRUE),

('Debit Card', TRUE),

('UPI', TRUE),

('Cash', TRUE),

('Net Banking', TRUE);

INSERT INTO sales (customer\_id, sale\_date, payment\_method\_id, sales\_channel\_id, total\_amount) VALUES

(1, '2025-03-10 12:15:00', 1, 1, 3300.00), -- Alice via Credit Card, Online

(2, '2025-03-15 15:45:00', 2, 2, 5000.00), -- Bob via Debit Card, Retail

(3, '2025-04-01 10:30:00', 3, 3, 800.00), -- Charlie via UPI, Mobile App

(4, '2025-04-05 18:20:00', 4, 2, 4800.00), -- Diana via Cash, Retail

(5, '2025-04-10 09:50:00', 5, 1, 1500.00); -- Ethan via Net Banking, Online

INSERT INTO sale\_line\_items (sale\_id, product\_id, quantity, unit\_price, line\_total) VALUES

(1, 1, 2, 800.00, 1600.00), -- Wireless Mouse x2

(1, 5, 1, 1700.00, 1700.00), -- Bluetooth Speaker

(2, 3, 1, 5000.00, 5000.00), -- Office Chair

(3, 1, 1, 800.00, 800.00), -- Wireless Mouse

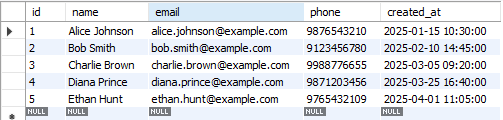
(4, 2, 1, 2500.00, 2500.00), -- Gaming Keyboard

(4, 4, 2, 1150.00, 2300.00), -- Water Bottle (special price)

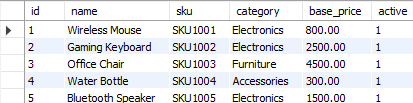
(5, 5, 1, 1500.00, 1500.00); -- Bluetooth Speaker

**Quick Checks**

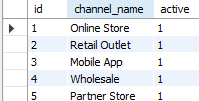
Select \* from customers;

****

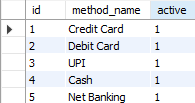
Select \* from products;



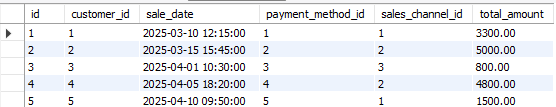
Select \* from sales\_channels;



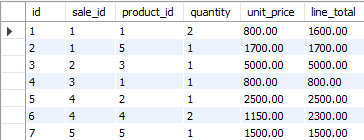
Select \* from payment\_methods;



Select \* from sales;



Select \* from sale\_line\_items;



**Analytics queries (Monthly Management Reports)**

**Total Revenue (Per Month)**

SELECT DATE\_FORMAT(sale\_date, '%Y-%m') AS month,

SUM(total\_amount) AS total\_revenue

FROM sales

GROUP BY month

ORDER BY month;



**Average Order Value**

SELECT DATE\_FORMAT(sale\_date, '%Y-%m') AS month,

AVG(total\_amount) AS avg\_order\_value

FROM sales

GROUP BY month

ORDER BY month;



**Price Deviations (Actual vs base price)**

SELECT p.name,

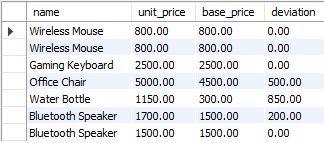
sli.unit\_price,

p.base\_price,

(sli.unit\_price - p.base\_price) AS deviation

FROM sale\_line\_items sli

JOIN products p ON sli.product\_id = [p.id](http://p.id);



**Revenue by Channel**

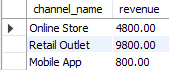
SELECT sc.channel\_name,

SUM(s.total\_amount) AS revenue

FROM sales s

JOIN sales\_channels sc ON s.sales\_channel\_id = sc.id

GROUP BY sc.channel\_name;



**Revenue by Payment Method**

SELECT pm.method\_name,

SUM(s.total\_amount) AS revenue

FROM sales s

JOIN payment\_methods pm ON s.payment\_method\_id = pm.id

GROUP BY pm.method\_name;

