Taller eventos

```
CREATE TABLE IF NOT EXISTS ingrediente (
  id              INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre          VARCHAR(100) NOT NULL,
  unidad_medida   VARCHAR(20) NOT NULL,
  stock_actual    INT NOT NULL DEFAULT 0,
  stock_minimo    INT NOT NULL DEFAULT 0,
  precio_unitario DECIMAL(10,2) NOT NULL,
  creado_en       DATETIME DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS resumen_ventas (
  fecha           DATE PRIMARY KEY,
  total_pedidos   INT,
  total_ingresos  DECIMAL(12,2),
  creado_en       DATETIME DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS alerta_stock (
  id              INT AUTO_INCREMENT PRIMARY KEY,
  ingrediente_id  INT UNSIGNED NOT NULL,
  stock_actual    INT NOT NULL,
  fecha_alerta    DATETIME NOT NULL,
  creado_en       DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (ingrediente_id) REFERENCES ingrediente(id)
);

INSERT INTO ingrediente (nombre, unidad_medida, stock_actual, stock_minimo,
precio_unitario)
VALUES
('Queso Mozzarella', 'kg', 3, 5, 12.50),
('Harina', 'kg', 20, 10, 1.80),
('Salsa de Tomate', 'litros', 9, 10, 2.30),
('Pepperoni', 'kg', 2, 5, 15.00),
('Aceitunas', 'kg', 8, 5, 6.00);

INSERT INTO resumen_ventas (fecha, total_pedidos, total_ingresos)
VALUES
(CURDATE() - INTERVAL 1 DAY, 15, 350.00),
(CURDATE() - INTERVAL 2 DAY, 12, 280.00),
(CURDATE() - INTERVAL 370 DAY, 10, 250.00);
```

```
DELIMITER //
CREATE PROCEDURE sp_generar_resumen_diario()
BEGIN
  INSERT INTO resumen_ventas (fecha, total_pedidos, total_ingresos)
  SELECT CURDATE() - INTERVAL 1 DAY,
         FLOOR(RAND() * 20 + 10),
         ROUND(RAND() * 500 + 200, 2);
END;
//
DELIMITER ;

DELIMITER //
CREATE PROCEDURE sp_generar_resumen_semanal()
BEGIN
  INSERT INTO resumen_ventas (fecha, total_pedidos, total_ingresos)
  SELECT CURDATE() - INTERVAL WEEKDAY(CURDATE()) + 7 DAY,
         FLOOR(RAND() * 100 + 50),
         ROUND(RAND() * 2500 + 1000, 2);
END;
//
DELIMITER ;

DELIMITER //
CREATE PROCEDURE sp_generar_alertas_criticas()
BEGIN
  INSERT INTO alerta_stock (ingrediente_id, stock_actual, fecha_alerta)
  SELECT id, stock_actual, NOW()
  FROM ingrediente
  WHERE stock_actual < 5;
END;
//
DELIMITER ;

DELIMITER //
CREATE PROCEDURE sp_monitoreo_stock_continuo()
BEGIN
  INSERT INTO alerta_stock (ingrediente_id, stock_actual, fecha_alerta)
  SELECT id, stock_actual, NOW()
  FROM ingrediente
  WHERE stock_actual < 10;
END;
//
DELIMITER ;

DELIMITER //
CREATE PROCEDURE sp_eliminar_resumenes_antiguos()
```

```
BEGIN
  DELETE FROM resumen_ventas
  WHERE fecha < CURDATE() - INTERVAL 365 DAY;
END;
//
DELIMITER ;

DROP EVENT IF EXISTS ev_resumen_diario_unico;
CREATE EVENT ev_resumen_diario_unico
ON SCHEDULE AT CURRENT_DATE + INTERVAL 1 HOUR
ON COMPLETION NOT PRESERVE
DO
  CALL sp_generar_resumen_diario();

DROP EVENT IF EXISTS ev_resumen_semanal;
CREATE EVENT ev_resumen_semanal
ON SCHEDULE EVERY 1 WEEK
STARTS (CURRENT_DATE + INTERVAL ((8 - DAYOFWEEK(CURRENT_DATE)) % 7) DAY +
INTERVAL 1 HOUR)
ON COMPLETION PRESERVE
DO
  CALL sp_generar_resumen_semanal();

DROP EVENT IF EXISTS ev_alerta_stock_unica;
CREATE EVENT ev_alerta_stock_unica
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 2 MINUTE
ON COMPLETION NOT PRESERVE
DO
  CALL sp_generar_alertas_criticas();

DROP EVENT IF EXISTS ev_purgar_resumen_antiguo;
CREATE EVENT ev_purgar_resumen_antiguo
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 MINUTE
ON COMPLETION NOT PRESERVE
DO
  CALL sp_eliminar_resumenes_antiguos();

SELECT * FROM alerta_stock;
SELECT * FROM resumen_ventas;
SHOW EVENTS;
```