

A. Normalización y DER

```
[PACIENTE]
* PK id_paciente (INT)
- apellido_nombre (VARCHAR) NOT NULL
- fecha_nacimiento (DATE) NOT NULL
- dni (VARCHAR) NOT NULL UNIQUE
- direccion (VARCHAR) NOT NULL
- email (VARCHAR) NULL
- telefono (VARCHAR) NOT NULL
    |
    | 1:N
    v

[INGRESO_LABORATORIO] <----- (1:N) ----- [PERSONAL_MEDICO]
* PK id_ingreso (INT)                * PK id_medico (INT)
- fecha_ingreso (DATE) NOT NULL        - apellido_nombre (VARCHAR) NOT NULL
- fecha_retiro (DATE) NOT NULL         - matricula (VARCHAR) NOT NULL UNIQUE
- FK id_paciente (INT) NOT NULL        - servicio (VARCHAR) NOT NULL
- FK id_medico (INT) NOT NULL
    |
    | N:M (a través de INGRESO_PRACTICA)
    v

[PRACTICA] <----- (N:1) ----- [ESPECIALIDAD]
* PK id_practica (INT)                * PK id_especialidad (INT)
- nombre (VARCHAR) NOT NULL           - nombre (VARCHAR) NOT NULL UNIQUE
- tipo_muestra (VARCHAR) NOT NULL     - FK id_jefe (INT) NOT NULL
- tiempo_procesamiento (INT) NOT NULL
- FK id_especialidad (INT) NOT NULL
    |
    | 1:N
    v

[EMPLEADO_LABORATORIO]
* PK id_empleado (INT)
- apellido_nombre (VARCHAR) NOT NULL
- categoria (VARCHAR) NOT NULL
- FK id_especialidad (INT) NOT NULL
```

```
[INGRESO_PRACTICA] (Tabla intermedia N:M)
* PK,FK id_ingreso (INT)
* PK,FK id_practica (INT)

Leyenda:
* PK: Clave Primaria
- FK: Clave Foránea
- : Atributo regular
1:N : Relación uno a muchos
N:M : Relación muchos a muchos
...
```

B. SQL e INTEGRIDAD REFERENCIAL

1. La respuesta correcta es la d y la b que son lo mismo. Establecer un valor predeterminado para una tabla, en caso de que no se especifique un valor durante la inserción.
2. La respuesta correcta es la c: `SELECT * FROM RUBROS WHERE NOMBRE LIKE 'C%'`
3. La respuesta correcta es la c: Se inserta un registro con el nombre vacío (ya que un string vacío no es lo mismo que NULL)
4. La respuesta correcta es la a: No se puede eliminar un registro de la tabla RUBROS si hay registros en la tabla PRODUCTOS con un ID_RUBRO existente, debido a la restricción de clave foránea.
5. Consulta para obtener las ventas del mes y año corriente:

```
SELECT * FROM VENTAS
WHERE MES = MONTH(CURRENT_DATE())
AND ANIO = YEAR(CURRENT_DATE());
```

6. Consulta para mostrar ventas ordenadas descendientemente sin repetir razones sociales:

```
SELECT DISTINCT RAZON_SOCIAL
FROM VENTAS
ORDER BY RAZON_SOCIAL DESC;
```

7. Consulta para establecer el stock_minimo en al menos 10 artículos:

```
ALTER TABLE STOCK_PRODUCTOS
ADD CONSTRAINT CHK_STOCK_MINIMO CHECK (STOCK_MINIMO >= 10);
```

8. Consulta para mostrar ventas de la primera quincena de septiembre del año actual:

```
SELECT RAZON_SOCIAL
FROM VENTAS
WHERE MES = 9
AND ANIO = YEAR(CURRENT_DATE())
AND DIA <= 15;
```

9. Consulta para productos con precio específico y nombre que comienza con "imp":

```
SELECT NOMBRE
FROM PRODUCTOS
WHERE (PRECIO_UNITARIO > 10000 OR PRECIO_UNITARIO < 5000)
```

```
AND NOMBRE LIKE 'imp%';
```