

Proyecto Final de Base de Datos

Carátula

Institución: Instituto Nacional Superior del Profesorado Técnico

Asignatura: Estructura y Base de Datos

Título del Proyecto: Gestión Hotel

Comisión: 2.601

Grupo N°: 7

Integrantes: Gian Franco Crovetto y Pablo Alexis González Román

Repositorio Git: <https://github.com/Crove7/BaseDeDatos>

Docentes Responsable: Alejandro Behringer y Juan Carlos Capia

Año Lectivo: 2025

índice

1.Resumen.....	3
2.Introducción	4
3.Objetivos	5
4.Alcance y Limitaciones.....	6
5.Análisis de Requerimientos	7
5.1Requerimientos Funcionales Clave	7
5.2 Requerimientos No Funcionales	7
6.Diseño de la Base de Datos.....	8
6.1. Diseño Conceptual	8
6.2. Diseño Lógico	8
7.Implementación.....	9
7.1. Definición de Estructura (DDL)	9
7.2. Manipulación de Datos (DML).....	9
7.3. Triggers (Lógica de Negocio Automática)	9
8.Consultas de Prueba.....	10
9.Resultados y Discusión	13
9.1 Resultados de Consultas	13
11. Bibliografía / Referencias	15

1. Resumen

El presente informe describe el diseño e implementación conceptual y lógica de una base de datos destinada a la gestión integral de un hotel.

El sistema tiene como finalidad administrar de manera eficiente la información relacionada con las reservas, los clientes, los empleados, las habitaciones y los servicios ofrecidos por el establecimiento.

El diseño propuesto busca garantizar la integridad de los datos, reducir la redundancia y optimizar las operaciones comunes del sistema, tales como el registro de reservas, la asignación de habitaciones, el control de servicios y la supervisión del personal.

2.Introducción

En la actualidad, los sistemas de gestión hotelera requieren bases de datos robustas que permitan almacenar, procesar y consultar información de forma precisa y ágil.

La base de datos diseñada tiene como propósito centralizar toda la información vinculada con el funcionamiento de los hoteles: clientes, reservas, habitaciones, servicios, empleados y supervisores.

De esta manera, se facilita la toma de decisiones administrativas y se mejora la experiencia del cliente al automatizar procesos que tradicionalmente se realizaban de forma manual o dispersa.

3.Objetivos

El objetivo general:

Es diseñar una base de datos relacional que permita gestionar los procesos fundamentales de los hoteles, incluyendo la administración de reservas, clientes, habitaciones, servicios y empleados encargados de las reservas.

Objetivos específicos:

- Modelar las entidades y relaciones que representan los elementos principales del negocio de hoteles.
- Asegurar la integridad referencial mediante el uso de claves primarias y foráneas.
- Permitir la trazabilidad de las reservas y los servicios asociados a cada habitación.
- Optimizar la estructura para futuras consultas y reportes administrativos.
- Establecer una base sólida para una posible implementación en un sistema de gestión real.

4.Alcance y Limitaciones

La base de datos cubre los siguientes aspectos del sistema hoteles:

- Registro y gestión de **clientes**.
- Administración de **habitaciones**, incluyendo tipo, estado, piso y precio diario.
- Registro de **reservas**, con fechas, montos, y asociación a clientes y empleados.
- Asociación de **servicios** adicionales a las habitaciones (por ejemplo: limpieza, minibar, spa).
- Control de **empleados** y **supervisores** a cargo de los hoteles.
- Asociación de empleados a reservas para reflejar atención o gestión administrativa.

Limitaciones

- No se incluyen funcionalidades de facturación ni pagos electrónicos.
- No contempla gestión de inventarios, mantenimiento ni proveedores.
- Las tarifas y servicios se manejan de forma estática (sin variaciones por temporada).
- El sistema no implementa control de usuarios ni niveles de acceso (solo nivel de datos).

5. Análisis de Requerimientos

El análisis de requerimientos se centró en identificar las necesidades funcionales y no funcionales esenciales para la operación de un hotel, asegurando la trazabilidad y la integridad de los datos transaccionales.

5.1 Requerimientos Funcionales Clave

- **Gestión Centralizada de Reservas:** El sistema debe permitir el registro de una reserva vinculada a un único cliente y una habitación específica, con fechas de inicio y fin claras.
- **Control de Servicios Adicionales:** Capacidad para asociar múltiples servicios (minibar, spa, etc.) a cada habitación, permitiendo calcular los costos totales.
- **Trazabilidad de Atención:** El sistema debe registrar qué empleados estuvieron involucrados en la gestión o atención de una reserva específica (relación M:N).
- **Cálculo de Tarifas:** El sistema debe facilitar el cálculo del costo total de una reserva basado en el precio diario de la habitación y la duración de la estadía ($\text{fecha_final} - \text{fecha_inicio}$).
- **Generación de Reportes:** Se requiere la capacidad de generar reportes administrativos clave, como la recaudación mensual por hotel y el listado de huéspedes activos o próximos a finalizar su estadía.

5.2 Requerimientos No Funcionales

- **Integridad de Datos:** Garantizar que las relaciones entre tablas sean válidas y consistentes (FOREIGN KEY).
- **Automatización de Reglas:** Implementar reglas de negocio complejas mediante triggers (ej. control de salarios o estados de habitación).
- **Rendimiento:** Optimizar la estructura para facilitar consultas de agregación y filtrado (ej. uso de INDEX).

6. Diseño de la Base de Datos

6.1. Diseño Conceptual

El diseño conceptual se representa mediante un Diagrama Entidad-Relación (DER) que modela los objetos y sus relaciones dentro del dominio hotelero.

Principales entidades:

- Cliente: contiene los datos personales de cada huésped (nombre, apellido, email, teléfono, país).
- Reserva: representa la acción de reservar una habitación, incluyendo fechas, monto y relación con cliente, habitación y empleado.
- Habitación: almacena información sobre número, estado, precio por día, piso y tipo.
- Hotel: identifica al hotel, su dirección, calificación y supervisor responsable.
- Empleado: registra los datos del personal, su salario y nombre/apellido.
- Supervisor: administra la relación jerárquica con los hoteles.
- Servicio: lista los servicios adicionales y su costo.
- Tipo: clasifica las habitaciones según su tipo (simple, doble, suite, etc.).

Relaciones principales:

- **Los hoteles tienen** muchas **habitaciones** (relación 1:N), y cada habitación pertenece a un solo hotel.
- Una habitación puede tener varios servicios (relación muchos a muchos mediante la tabla habitacion_servicio).
- Un cliente puede realizar varias reservas, pero cada reserva pertenece a un solo cliente.
- Una reserva puede estar atendida por uno o varios empleados (relación muchos a muchos mediante empleado_reserva).
- Cada hotel tiene un supervisor asignado.
- Cada habitación se clasifica con un tipo, si es vip, normal, etc.

6.2. Diseño Lógico

A partir del modelo conceptual, se construye el modelo lógico representado por las siguientes tablas:

- Supervisor (id_supervisor, DNI, fecha_asignacion, telefono, estado_laboral, salario, nombre, apellido)
- Hotel (id_hotel, nombre, direccion, calificacion, id_supervisor)
- Habitacion (id_habitacion, numero, precio_días, estado, piso, id_hotel)
- Tipo (id_tipo, descripcion, nombre_tipo, id_habitacion)
- Servicio (id_servicio, descripcion, nombre_servicio, costo)
- Habitacion_servicio (id, id_servicio, id_habitacion)
- Cliente (id_cliente, nombre, apellido, email, telefono, pais)
- Reserva (id_reserva, fecha_pago, monto, fecha_inicio, fecha_final, id_cliente, id_habitacion)
- Empleado (id_empleado, nombre, apellido, salario)
- Empleado_reserva (id, id_reserva, id_empleado)

7.Implementación

La implementación se llevó a cabo utilizando el lenguaje **SQL**, orientado a la plataforma **MySQL**.

7.1. Definición de Estructura (DDL)

Se definieron las **10 tablas** necesarias para el modelo lógico:

- **Claves y Tipos:** Todas las claves primarias (PK) fueron definidas con INT NOT NULL AUTO_INCREMENT para asegurar la unicidad y la generación automática de identificadores. Se utilizaron tipos de datos precisos como DECIMAL(11,2) para costos y salarios, y CHAR(8) para campos como el DNI y teléfono.
- **Restricciones de Integridad:** Se establecieron **12** restricciones FOREIGN KEY (FK) para garantizar la integridad referencial.

7.2. Manipulación de Datos (DML)

Se insertó un conjunto de datos de prueba (Inserts.sql) para poblar todas las tablas con información consistente. Esto incluyó la inserción de 20 reservas y la debida asociación en las tablas de enlace (empleado_reserva y habitacion_servicio).

7.3. Triggers (Lógica de Negocio Automática)

Se implementaron tres triggers para automatizar procesos clave en el servidor de base de datos, con un enfoque en la **prevención de datos inválidos** y el cumplimiento de reglas de negocio complejas:

Trigger	Evento	Función
empleado_aumento	BEFORE INSERT en Empleado	Aplica un aumento del 10% si el salario inicial de un nuevo empleado es menor a 2200.
TR_Reserva_Fechas_Validas	BEFORE INSERT en Reserva	Asegura que la fecha_final de la reserva sea siempre posterior a la fecha_inicio, evitando errores lógicos de dominio (Integridad Temporal).
TR_Reserva_Concurrencia	BEFORE INSERT en Reserva	Impide que se registre una reserva si las fechas solicitadas se solapan con una reserva existente en la misma habitación (Control de Concurrencia), utilizando la cláusula SIGNAL SQLSTATE para notificar el error.

8.Consultas de Prueba

A) Averiguo cuanto gasto cada cliente por sus reservas hechas, mostrando el total por días de estadía, los servicios usados y el total de todo lo gastado.

```
SELECT r.id_reserva, c.nombre, c.apellido, r.fecha_pago, h.numero AS habitacion, h.precio_dias,
DATEDIFF(r.fecha_final, r.fecha_inicio) AS dias_estadia,
(h.precio_dias * DATEDIFF(r.fecha_final, r.fecha_inicio)) AS total_habitacion,
IFNULL(SUM(s.costo), 0) AS total_servicios,
(h.precio_dias * DATEDIFF(r.fecha_final, r.fecha_inicio)) + IFNULL(SUM(s.costo), 0) AS
total_pagar
FROM Reserva r
JOIN Habitacion h ON r.id_habitacion = h.id_habitacion JOIN cliente c ON r.id_cliente = c.id_cliente LEFT
JOIN habitacion_servicio hs ON h.id_habitacion = hs.id_habitacion LEFT JOIN Servicio s ON hs.id_servicio
= s.id_servicio GROUP BY r.id_reserva, c.id_cliente, c.nombre, c.apellido, r.fecha_pago, h.numero, h.precio_dias
ORDER BY r.id_reserva;
```

id_reserva	1	nombre	apellido	fecha_pago	habitacion	precio_dias	dias_estadia	total_habitacion	total_servicios	total_pagar
	1	Santiago	Rivas	2025-11-01	101	180.00	5	900.00	160.00	1060.00
	2	Valentina	Suárez	2025-02-02	102	250.00	2	500.00	35.00	535.00
	3	Mateo	Ortega	2025-03-03	201	300.00	5	1500.00	25.00	1525.00
	4	Camila	Lagos	2025-09-04	202	220.00	5	1100.00	0.00	1100.00
	5	Ignacio	Torres	2025-12-05	301	400.00	3	1200.00	60.00	1260.00
	6	Santiago	Rivas	2025-11-06	102	250.00	4	1000.00	35.00	1035.00
	7	Valentina	Suárez	2025-06-07	201	300.00	5	1500.00	25.00	1525.00
	8	Mateo	Ortega	2025-02-08	202	220.00	5	1100.00	0.00	1100.00
	9	Camila	Lagos	2025-03-09	301	400.00	5	2000.00	60.00	2060.00
	10	Ignacio	Torres	2025-02-10	101	180.00	5	900.00	160.00	1060.00
	11	Santiago	Rivas	2025-03-11	201	300.00	5	1500.00	25.00	1525.00
	12	Valentina	Suárez	2025-10-12	202	220.00	5	1100.00	0.00	1100.00
	13	Mateo	Ortega	2025-12-13	301	400.00	5	2000.00	60.00	2060.00
	14	Camila	Lagos	2025-08-14	101	180.00	5	900.00	160.00	1060.00
	15	Ignacio	Torres	2025-07-15	102	250.00	5	1250.00	35.00	1285.00
	16	Santiago	Rivas	2025-03-16	202	220.00	5	1100.00	0.00	1100.00
	17	Valentina	Suárez	2025-05-17	301	400.00	5	2000.00	60.00	2060.00
	18	Mateo	Ortega	2025-04-18	101	180.00	5	900.00	160.00	1060.00
	19	Camila	Lagos	2025-09-19	102	250.00	5	1250.00	35.00	1285.00
	20	Ignacio	Torres	2025-11-20	201	300.00	5	1500.00	25.00	1525.00

B) Los clientes que siguen hospedándose en la fecha hoy que siguen con su reserva activa.

```
SELECT c.nombre, c.apellido, r.fecha_inicio AS Entrada, r.fecha_final AS Salida, CURDATE() AS
Fecha_Actual FROM Cliente c JOIN Reserva r ON c.id_cliente = r.id_cliente WHERE DATE(r.fecha_inicio)
<= CURDATE() AND DATE(r.fecha_final) > CURDATE();
```

nombre	apellido	Entrada	Salida	Fecha_Actual
Santiago	Rivas	2025-11-08	2025-11-13	2025-11-11
Santiago	Rivas	2025-11-11	2025-11-16	2025-11-11
Mateo	Ortega	2025-11-09	2025-11-14	2025-11-11

C) Los clientes que su reserva esta por vencer ya sea hoy o en un periodo de 48 horas.

```
SELECT c.nombre, c.apellido, r.fecha_final AS Fecha_Salida, h.numero AS Habitacion_Numero,
NOW() AS Ahora, DATE_ADD(NOW(), INTERVAL 48 HOUR) AS Limite_48hs FROM Reserva r JOIN
Cliente c ON r.id_cliente = c.id_cliente JOIN Habitacion h ON r.id_habitacion = h.id_habitacion WHERE
r.fecha_final > CURDATE() AND r.fecha_final <= DATE_ADD(NOW(), INTERVAL 48 HOUR) ORDER
BY r.fecha_final;
```

nombre	apellido	Fecha_Salida	Habitacion_Numero	Ahora	Limite_48hs
Santiago	Rivas	2025-11-13	101	2025-11-11 23:43:48	2025-11-13 23:43:48

D) Cuenta las Reservas que maneja cada empleado, Ordenados de manera descendente el que tiene mayor cantidad primero y el que tiene menor ultimo.

```
SELECT e.id_empleado,
e.nombre,
e.apellido,
COUNT(DISTINCT er.id_reserva) AS reservas_manejadas
FROM Empleado e JOIN empleado_reserva er ON e.id_empleado =
er.id_empleado
GROUP BY e.id_empleado, e.nombre, e.apellido
ORDER BY reservas_manejadas DESC;
```

id_empleado	nombre	apellido	reservas_manejadas	▼ 1
2	Bruno	Salazar	6	
1	Julieta	Mansilla	5	
3	Lucía	Herrera	5	
5	Martina	Roldán	4	
4	Tomás	Vega	4	

E) Muestro lo recaudado en cada mes por cada hotel durante todo el año que estuvo.

```
SELECT h.id_hotel, YEAR(r.fecha_inicio) AS Anio, MONTH(r.fecha_inicio) AS Mes, SUM(r.monto) AS  
Total_Recaudado_Por_Hotel FROM Reserva r JOIN Habitacion h ON r.id_habitacion = h.id_habitacion  
WHERE YEAR(r.fecha_inicio) = YEAR(CURDATE()) GROUP BY h.id_hotel, Anio, Mes ORDER BY  
h.id_hotel;
```

id_hotel	1	Anio	Mes	Total_Recaudado_Por_Hotel
	1	2025	2	1050.00
	1	2025	11	2050.00
	1	2025	12	950.00
	2	2025	9	500.00
	2	2025	10	1150.00
	2	2025	11	1200.00
	2	2025	12	1200.00
	3	2025	5	800.00
	3	2025	11	950.00
	3	2025	12	2600.00
	4	2025	2	1100.00
	4	2025	10	1100.00
	4	2025	11	1100.00
	4	2025	12	1150.00
	5	2025	7	1250.00
	5	2025	11	1250.00
	5	2025	12	1900.00

9.Resultados y Discusión

9.1 Resultados de Consultas

Las consultas de prueba demostraron la capacidad del modelo para responder a las necesidades administrativas complejas:

- **Cálculo Financiero Dinámico:** La Consulta 1 mostró la correcta aplicación de funciones de agregación (SUM, IFNULL) y tiempo (DATEDIFF) para calcular costos de estadía y servicios asociados, validando la atomicidad de las tablas de enlace.
- **Gestión Temporal:** Las Consultas 2 y 3 validaron el uso de funciones de fecha (CURDATE, NOW, DATE_ADD) para filtrar registros basados en la línea de tiempo actual, esencial para la planificación operativa y logística de *check-out*.
- **Eficiencia en Relaciones M:N:** La Consulta 4 empleó COUNT(DISTINCT `er.id_reserva`) en la tabla de enlace `empleado_reserva`, la técnica correcta para contar las reservas gestionadas por cada empleado sin duplicar los conteos.

9.2 Discusión Crítica de Integridad y Lógica

El diseño implementado garantiza un alto nivel de **Integridad de Datos** gracias a las siguientes decisiones:

- **Integridad Referencial Fuerte:** Uso correcto de FOREIGN KEY en 12 puntos clave.
- **Control de Superposición:** La implementación de los triggers `TR_Reserva_Fechas_Validas` y `TR_Reserva_Concurrencia` eleva la calidad del sistema. Estos triggers utilizan la lógica **BEFORE INSERT** y la instrucción **SIGNAL SQLSTATE** para **prevenir** que se inserten datos que violen las reglas de concurrencia y de dominio temporal (ej., fecha de fin anterior a fecha de inicio), lo cual es el método más robusto en SQL para proteger la base de datos de errores de aplicación.
- **Manejo de Lógica de Negocio:** El trigger `empleado_aumento` garantiza el cumplimiento de la política salarial de la empresa, automatizando la corrección de datos en el servidor.

10. Conclusiones

El diseño desarrollado permite gestionar de forma integral los principales procesos de los hoteles mediante una base de datos relacional bien estructurada.

La organización de las tablas y las relaciones establecidas aseguran consistencia, escalabilidad y la posibilidad de ampliar el sistema con módulos adicionales, como facturación o control de inventarios, en futuras versiones.

11. Bibliografía / Referencias

Alura Cursos. (s.f.). *¿Qué es y cómo funciona un trigger en SQL?*

<https://www.aluracursos.com/blog/que-es-y-como-trigger-en-sql>