Evaluación de portafolio[Actividad Evaluada]

Pablo Varas Salamanca

21 de julio de 2025, 20:35:00 h

Evaluación del portafolio

Instrucciones

En función de tu proyecto personal previamente establecido, deberás implementar clase a clase las diferentes tecnologías y competencias técnicas adquiridas a lo largo del curso.

Recuerda que este proyecto irá directamente al registro de evidencia de tu portafolio, el cual deberá demostrar el dominio, competencias técnicas y diferentes habilidades relacionadas con la gestión de bases de datos relacionales.

En una empresa de ventas, es necesario gestionar un inventario de productos que se van almacenando en diferentes proveedores, y a su vez, realizar transacciones de compra y venta. Para esto, se utiliza una base de datos relacional (RDBMS) para organizar toda la información de manera estructurada. El sistema debe permitir agregar productos, actualizar el inventario, registrar compras y ventas, y consultar los datos de manera eficiente.

Requerimientos Funcionales Mínimos Esperados

1. Distinguir las características, rol y elementos fundamentales de una base de datos relacional para la gestión de la información en una organización.

- Describir los componentes básicos de una base de datos relacional: tablas, registros, campos, claves primarias y foráneas. Elementos básicos

Una **tabla** organiza la información en **filas** (registros) y **columnas** (campos); cada fila representa una instancia de la entidad y cada columna un atributo.

Una **clave primaria** (PK) identifica de forma única cada registro dentro de su tabla, garantizando que no existan duplicados.

Una **clave foránea** (FK) referencia la PK de otra tabla y crea la relación lógica entre ambas, de modo que los valores almacenados en la FK sólo pueden existir si ya existen en la tabla referenciada.

- Explicar cómo se gestionan y almacenan los datos en tablas y cómo se establece la relación entre ellas para satisfacer necesidades organizacionales.

    - Las FKs admiten acciones que definen qué ocurre cuando la fila padre se actualiza o elimina:
        - `ON UPDATE CASCADE` propaga al hijo la modificación de la `PK`.
        - `ON DELETE RESTRICT` impide eliminar la fila padre si existen hijas relacionadas, evitando “huérfanos”.

- Ejemplo: Crear una tabla de clientes y otra de pedidos, relacionándolas por una clave foránea. Adaptamos el ejemplo a nuestro proyecto.

```sql
/* ================================
   0) Creación y selección de BD
   ================================*/
CREATE DATABASE IF NOT EXISTS epja_control;
USE epja_control;

/* ================================
   1) TABLAS PRINCIPALES
   ================================*/

/* 1.1 Funcionarios (profesores, psicólogos, etc.) */
CREATE TABLE funcionarios (
  id_funcionario INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(80) NOT NULL,
  rol    ENUM('profesor','psicologo','administrativo') NOT NULL, -- ENUM restringe valores
  ↪  :contentReference[oaicite:5]{index=5}
  email  VARCHAR(120)
);

/* 1.2 Cursos, cada uno con su profesor a cargo */
CREATE TABLE cursos (
  id_curso   INT AUTO_INCREMENT PRIMARY KEY,
  nombre     VARCHAR(40) NOT NULL,
  id_profesor INT,
  CONSTRAINT fk_profesor
    FOREIGN KEY (id_profesor)
      REFERENCES funcionarios(id_funcionario)
      ON UPDATE CASCADE
      ON DELETE RESTRICT
);

/* 1.3 Alumnos */
CREATE TABLE alumnos (
  id_alumno INT AUTO_INCREMENT PRIMARY KEY,
  nombre    VARCHAR(80) NOT NULL,
  fecha_nac DATE,
  telefono  VARCHAR(20),
  id_curso  INT,
  CONSTRAINT fk_curso
    FOREIGN KEY (id_curso)
      REFERENCES cursos(id_curso)
      ON UPDATE CASCADE
      ON DELETE RESTRICT
);

/* 1.4 Apoderados */
CREATE TABLE apoderados (
  id_apoderado INT AUTO_INCREMENT PRIMARY KEY,
  nombre  VARCHAR(80),
  telefono VARCHAR(20),
```

```sql
  email    VARCHAR(120)
);

/* 1.5 Relación N:M alumno-apoderado (clave compuesta) */
CREATE TABLE alumno_apoderado (
  id_alumno     INT,
  id_apoderado  INT,
  PRIMARY KEY (id_alumno, id_apoderado),               -- patrón recomendado
  ↪ :contentReference[oaicite:6]{index=6}
  FOREIGN KEY (id_alumno)    REFERENCES alumnos(id_alumno)
        ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (id_apoderado) REFERENCES apoderados(id_apoderado)
        ON UPDATE CASCADE ON DELETE CASCADE
);

/* 1.6 Acciones pedagógicas (responsable = funcionario) */
CREATE TABLE acciones_pedagogicas (
  id_accion      INT AUTO_INCREMENT PRIMARY KEY,
  descripcion    TEXT NOT NULL,
  fecha          DATETIME DEFAULT CURRENT_TIMESTAMP,
  id_alumno      INT,
  id_funcionario INT,
  FOREIGN KEY (id_alumno)     REFERENCES alumnos(id_alumno)
        ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY (id_funcionario) REFERENCES funcionarios(id_funcionario)
        ON UPDATE CASCADE ON DELETE RESTRICT
);

/* ================================
   2) DATOS DE EJEMPLO
   ================================*/
INSERT INTO funcionarios(nombre,rol,email) VALUES
  ('Patricia Ríos','profesor','patricia.rios@colegio.cl'),
  ('Carlos Salas','profesor','carlos.salas@colegio.cl'),
  ('María Godoy','psicologo','m.godoy@colegio.cl');

INSERT INTO cursos(nombre,id_profesor) VALUES
  ('1° Básico A',1),
  ('1° Básico B',2);

INSERT INTO alumnos(nombre,fecha_nac,telefono,id_curso) VALUES
  ('Ana Pérez','2015-03-04','+569123',1),
  ('Luis Soto','2015-11-20','+569456',2);

INSERT INTO apoderados(nombre,telefono,email) VALUES
  ('Fernando Pérez','+569777','fperez@familia.cl'),
  ('Laura Díaz','+569888','ldiaz@familia.cl');

INSERT INTO alumno_apoderado VALUES
  (1,1),(1,2),(2,2);

INSERT INTO acciones_pedagogicas(descripcion,id_alumno,id_funcionario)
VALUES ('Refuerzo lectura',1,3);
```

```sql
/* ================================
   3) TRANSACCIÓN ACID DE DEMO
   ================================*/
SET autocommit = 0;                        -- aislar la transacción
↪  :contentReference[oaicite:7]{index=7}
START TRANSACTION;
  INSERT INTO alumnos(nombre,id_curso) VALUES ('Marcela Núñez',1);
  UPDATE cursos
    SET nombre = CONCAT(nombre,' (actualizado)')
    WHERE id_curso = 1;
COMMIT;      -- usa ROLLBACK; para deshacer
```

Este patrón sustituye al clásico clientes / pedidos por el dominio escolar solicitado.

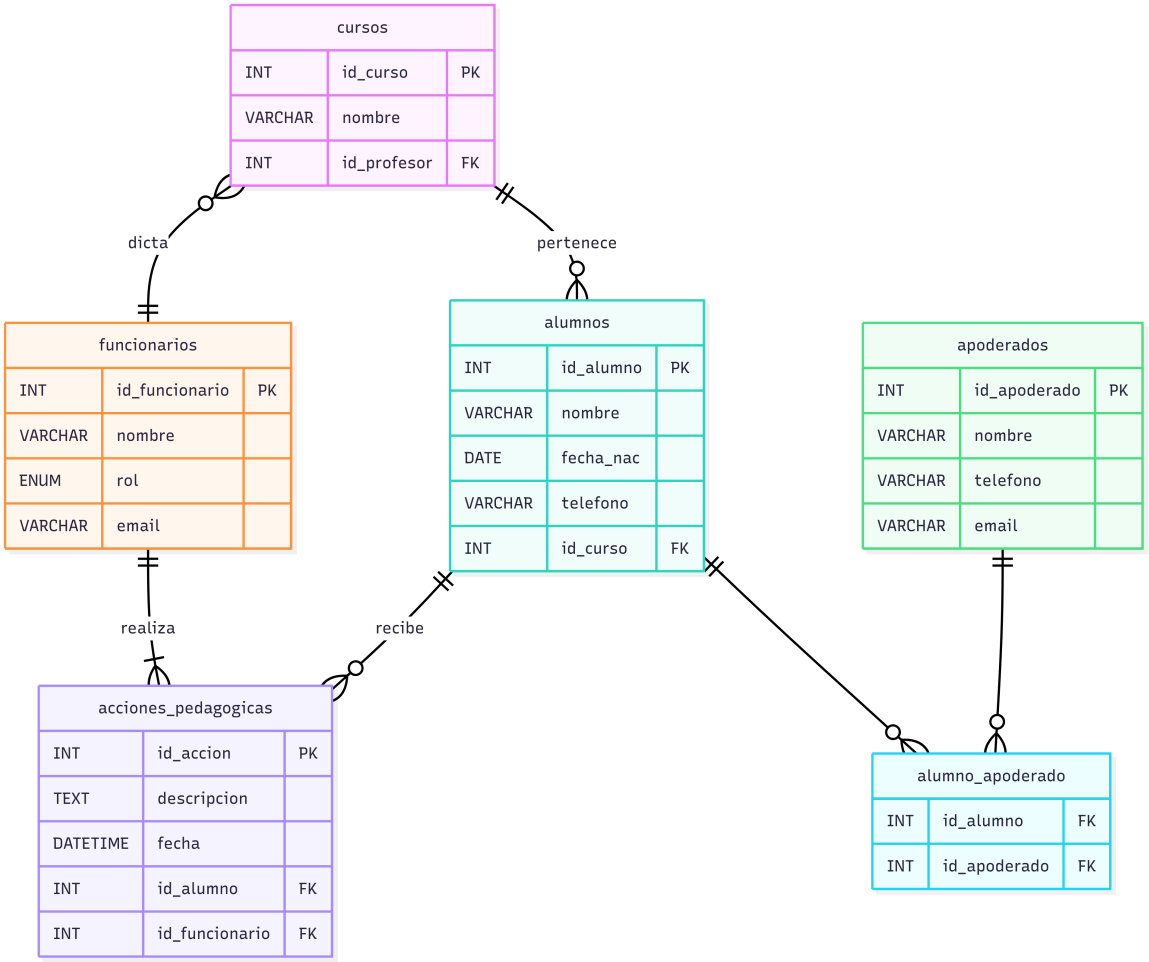

Figura 1: Diagrama inicial de inicio de la base de datos.

2. Utilizar Lenguaje Estructurado de Consultas (SQL) para la obtención de información que satisface los requerimientos planteados a partir de un modelo de datos dado.

- Desarrollar consultas `SQL` para obtener información específica de las tablas, utilizando cláusulas como `SELECT`, `WHERE`, `JOIN`, `GROUP BY`, entre otras.

```sql
/* Profesor responsable de cada curso */
SELECT c.nombre   AS curso,
       f.nombre   AS profesor
FROM   cursos c
JOIN   funcionarios f ON f.id_funcionario = c.id_profesor;

/* Total de acciones pedagógicas por curso */
SELECT c.nombre, COUNT(ac.id_accion) AS total_acciones
FROM   cursos  c
JOIN   alumnos a               USING(id_curso)
LEFT   JOIN acciones_pedagogicas ac ON ac.id_alumno = a.id_alumno
GROUP  BY c.id_curso;
```

- Ejemplo: Crear una consulta que obtenga todos los pedidos realizados por un cliente específico.

  En nuestro esquema escolar el análogo natural es obtener todas las acciones pedagógicas (pedidos) asociadas a un alumno (cliente) específico.

```sql
/* Acciones pedagógicas realizadas a un alumno concreto (id = :alumno_id) */
SELECT ac.id_accion,
       ac.descripcion,
       ac.fecha,
       f.nombre AS responsable,
       f.rol
FROM   acciones_pedagogicas  ac
JOIN   funcionarios          f  ON f.id_funcionario = ac.id_funcionario
WHERE  ac.id_alumno = :alumno_id;   -- sustituye :alumno_id por el número deseado
/* Cursos que actualmente no tienen alumnos */
SELECT *
FROM cursos
WHERE id_curso NOT IN (
  SELECT DISTINCT id_curso FROM alumnos
);
```

Cómo funciona

- `acciones_pedagogicas` contiene la PK $\text{id}_{\text{accion}}$, la descripción, la fecha y la FK $\text{id}_{\text{alumno}}$.
- Se realiza un `JOIN` con funcionarios para añadir el nombre y rol del responsable de cada acción.
- El filtro `WHERE ac.id_alumno = ...` limita el resultado al alumno solicitado (equivalente al «cliente específico» del ejemplo original).

  Si se prefiere usar el nombre del alumno en lugar de su ID:

```sql
SELECT ac.id_accion,
       ac.descripcion,
       ac.fecha,
       f.nombre AS responsable
FROM   alumnos              a
JOIN   acciones_pedagogicas ac ON ac.id_alumno = a.id_alumno
JOIN   funcionarios         f  ON f.id_funcionario = ac.id_funcionario
WHERE  a.nombre = 'Ana Pérez';
```

3. Utilizar lenguaje de manipulación de datos (DML) para la modificación de los datos existentes en una base de datos dando solución a un problema planteado.

- Implementar consultas de inserción (`INSERT`), actualización (`UPDATE`) y eliminación (`DELETE`) para modificar los datos dentro de las tablas.

  Lo adaptamos a nuestra base de datos:

```sql
/* INSERT: nuevo apoderado */
INSERT INTO apoderados(nombre,telefono,email)
VALUES ('Laura Díaz','+569888','ldiaz@familia.cl');

/* UPDATE: cambia teléfono del alumno */
UPDATE alumnos
SET telefono = '+569999999'
WHERE id_alumno = 2;
```

- Ejemplo: Crear una consulta que actualice la dirección de un cliente en la base de datos o elimine un pedido que no fue procesado.

  En nuestro caso adaptamos la consulta a nuestra base de datos: Eliminar una acción pedagógica que quedó pendiente (no realizada)

```sql
DELETE FROM acciones_pedagogicas
WHERE id_accion = 7          -- identifica la acción
  AND estado = 'sin realizar';
```

Las claves foráneas con `ON DELETE CASCADE` o `RESTRICT` garantizan que solo se borren filas cuando es seguro hacerlo, evitando registros huérfanos o borrados en cascada inesperados

4. Implementar estructuras de datos relacionales utilizando lenguaje de definición de datos (DDL) a partir de un modelo de datos para la creación y mantención de las definiciones de los objetos de una base de datos.

- Utilizar el lenguaje DDL para crear, modificar y eliminar tablas, índices y otros objetos dentro de una base de datos.

```sql
CREATE DATABASE IF NOT EXISTS epja_control;
USE epja_control;

/* Funcionario (profesor u otro rol) */
CREATE TABLE funcionarios (
  id_funcionario INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(80) NOT NULL,
  rol ENUM('profesor','psicologo','administrativo') NOT NULL,  -- ENUM restringe valores
  ↪ :contentReference[oaicite:7]{index=7}
  email VARCHAR(120)
```

```
);

/* Curso con su profesor */
CREATE TABLE cursos (
  id_curso INT PRIMARY KEY AUTO_INCREMENT,
  nombre   VARCHAR(30) NOT NULL,
  id_profesor INT,
  CONSTRAINT fk_profesor
    FOREIGN KEY (id_profesor)
    REFERENCES funcionarios(id_funcionario)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);

/* Alumno */
CREATE TABLE alumnos (
  id_alumno INT PRIMARY KEY AUTO_INCREMENT,
  nombre    VARCHAR(80) NOT NULL,
  fecha_nac DATE,
  telefono  VARCHAR(20),
  id_curso  INT,
  CONSTRAINT fk_curso
    FOREIGN KEY (id_curso)
    REFERENCES cursos(id_curso)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);

/* Apoderado y tabla puente N:M */
CREATE TABLE apoderados (
  id_apoderado INT AUTO_INCREMENT PRIMARY KEY,
  nombre  VARCHAR(80),
  telefono VARCHAR(20),
  email   VARCHAR(120)
);
CREATE TABLE alumno_apoderado (
  id_alumno INT,
  id_apoderado INT,
  PRIMARY KEY(id_alumno,id_apoderado),                -- tabla "junction"
  ↪  :contentReference[oaicite:8]{index=8}
  FOREIGN KEY(id_alumno)    REFERENCES alumnos(id_alumno)
        ON UPDATE CASCADE ON DELETE CASCADE,
  FOREIGN KEY(id_apoderado) REFERENCES apoderados(id_apoderado)
        ON UPDATE CASCADE ON DELETE CASCADE
);
```

- Ejemplo: Crear una tabla para almacenar información de empleados, con las columnas correspondientes como nombre, salario y fecha de ingreso.

  Lo adaptamos a nuestra base de datos:

```
/* Acciones pedagógicas */
CREATE TABLE acciones_pedagogicas (
```

```
    id_accion      INT AUTO_INCREMENT PRIMARY KEY,
    descripcion    TEXT NOT NULL,
    fecha          DATETIME DEFAULT CURRENT_TIMESTAMP,
    id_alumno      INT,
    id_funcionario INT,
    FOREIGN KEY(id_alumno)     REFERENCES alumnos(id_alumno)
          ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY(id_funcionario) REFERENCES funcionarios(id_funcionario)
          ON UPDATE CASCADE ON DELETE RESTRICT
);
```

---

5. Elaborar un modelo de datos de acuerdo a los estándares de modelamiento para resolver un problema de baja complejidad.

- Crear un diagrama entidad-relación (ER) para representar el modelo de datos antes de implementarlo en una base de datos.
    - 1.ª Forma Normal: todos los atributos contienen valores atómicos (sin listas ni multivalores).
    - 2.ª Forma Normal: cada atributo no-clave depende de la totalidad de su `PK` (las únicas `PK` compuestas están en la tabla puente).
    - 3.ª Forma Normal: no hay dependencias transitivas; por ejemplo, el rol depende sólo de `id_funcionario`, no de otro campo.

- Ejemplo: Crear un modelo de datos para una tienda en línea, que incluya entidades como productos, clientes, pedidos y métodos de pago, y sus respectivas relaciones.

  Lo adaptamos a nuestra base de datos:

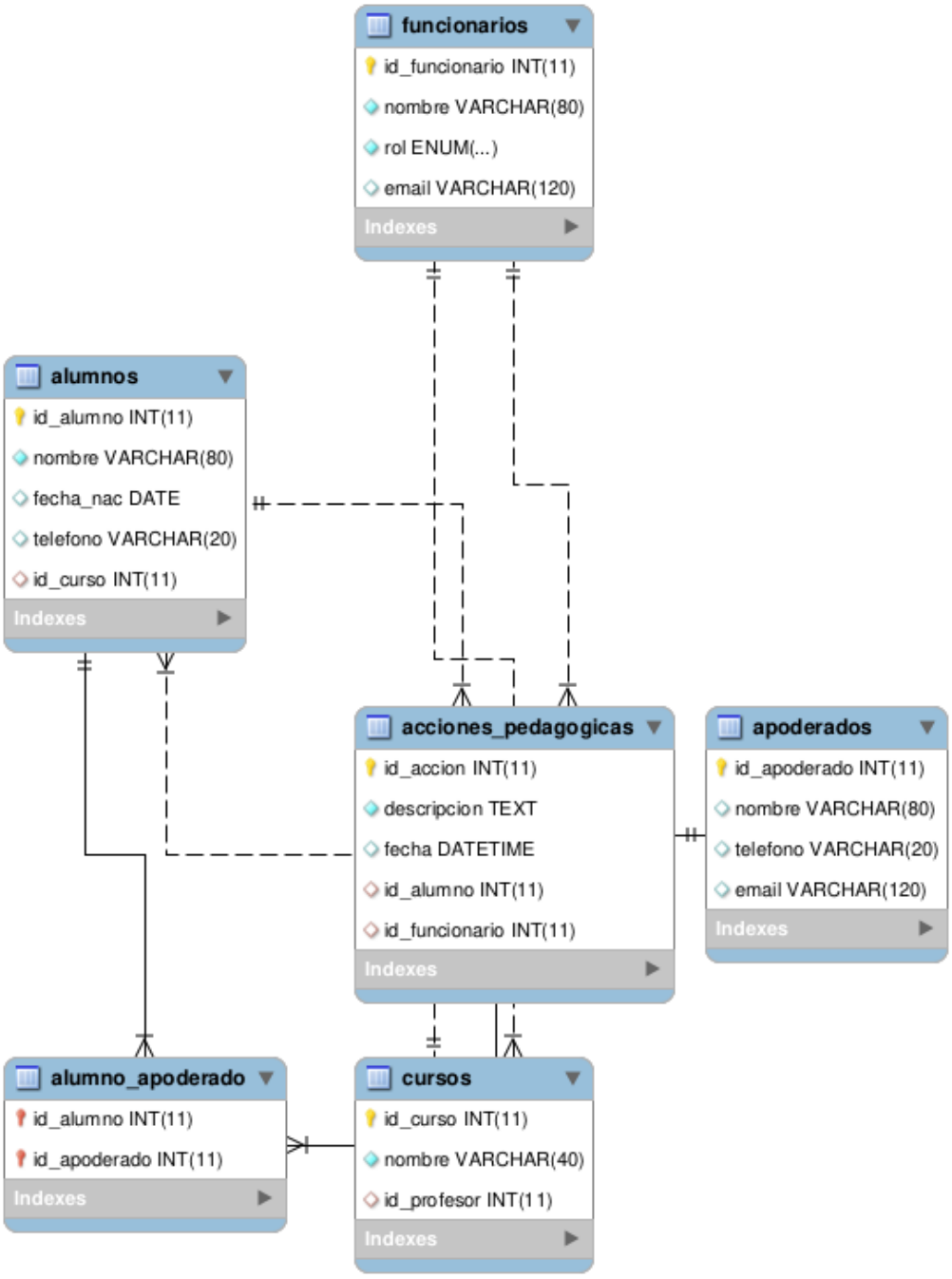

Figura 2: Diagrama Entidad-Relación de la base de datos.

Referencias

Skillnest. (2025a). *AE5_Elaboración de un modelo ERD [Material Lectivo]* [Última modificación: lunes, 05 de mayo de 2025, 11:40]. Consultado el 20 de junio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3002

Skillnest. (2025b). *AE5_Modelos de entidad relación (ERD) [Material Lectivo]* [ultima modificacin: lunes, 12 de mayo de 2025, 11:43]. Consultado el 18 de julio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3093

Skillnest. (2025c). *AE5_Relaciones muchos a muchos [Material Lectivo]* [Última modificación: lunes, 12 de mayo de 2025, 19:20]. Consultado el 20 de junio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3097

Skillnest. (2025d). *AE5_Relaciones uno a muchos [Material Lectivo]* [Última modificación: lunes, 12 de mayo de 2025, 19:12]. Consultado el 20 de junio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3096

Skillnest. (2025e). *AE5_Relaciones uno a uno [Material Lectivo]* [Última modificación: lunes, 12 de mayo de 2025, 19:08]. Consultado el 20 de junio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3095

Skillnest. (2025f). *Resumen del módulo [Video] [Módulo 3]* [Última modificación: jueves, 01 de mayo de 2025, 19:39]. Consultado el 20 de julio de 2025, desde https://learning-pro.skillnest.com/mod/page/view.php?id=3104

The Conventional Commits. (2025). *Commits Convencionales: Especificación para dar significado a los mensajes de los commits haciéndolos legibles para máquinas y humanos*. Consultado el 20 de julio de 2025, desde https://www.conventionalcommits.org/es/v1.0.0-beta.3/