Universidad Nacional Mayor de San Marcos
Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática Escuela Profesional de Ingeniería de Sistemas

Sistema de Gestión de Evaluaciones Virtuales

INFORME TÉCNICO

Curso:

Base de Datos

Docente:

Jorge Luis Chávez Soto

Grupo: 7

Alumnos:

21200125 - Ruiz Molina, Rafael Iván

22200052 - Uribe Mejía Guillermo César

20200154 - Auccaise Huallpa, Victor Alberto

22200003 - Aliaga Tolentino Fabrisio

Lima, Perú

2024

1. Presentación técnica

La visión general de la empresa EducaTech Perú es ser una empresa peruana líder en la transformación digital del sector educativo. Se especializa en el desarrollo de soluciones tecnológicas innovadoras que facilitan el aprendizaje y la evaluación en línea, brindando herramientas personalizadas y eficientes para instituciones educativas. La empresa se distingue por su enfoque en la mejora continua, la adaptabilidad de sus servicios y la satisfacción de las necesidades específicas de sus clientes.

Para ello se emplea la utilización de una base de datos en el programa de base de datos Oracle Database, con el lenguaje de PL/SQL para la gestión de proyectos.

La base de datos está diseñada para manejar procesos críticos relacionados con la evaluación educativa:

Esquema de Base de Datos

Tablas Principales:

1. Estudiante: Almacena información de los estudiantes registrados.
2. Examen: Registra información de exámenes asignados a estudiantes.
3. Resultado_Examen: Guarda las puntuaciones obtenidas y retroalimentación automatizada.
4. Plantilla: Define las características de los exámenes (número de preguntas según dificultad).
5. Pregunta: Contiene el contenido de las preguntas categorizadas por materia y dificultad.
6. Dificultad: Clasifica las preguntas en Fácil, Intermedio y Difícil.
7. Materia: Identifica las materias asociadas a preguntas y exámenes.
8. Examen_Pregunta: Relaciona preguntas con exámenes y estudiantes.

Relaciones Clave:

- Estudiante → Examen (1:N).
- Examen → Resultado_Examen (1:1).
- Examen → Plantilla (N:1).
- Pregunta → Dificultad y Materia (N:1).
- Examen_Pregunta: Tabla intermedia para gestionar preguntas asignadas a cada examen.

Optimizaciones:

- Claves primarias y foráneas: Garantizan integridad referencial.
- Índices: Se crearon índices en columnas claves (e.g., estudiante_ID, plantilla_ID, examen_ID) para mejorar el rendimiento de consultas.

2. Objetivos técnicos del trabajo final

El objetivo principal de este proyecto es implementar una base de datos optimizada y funcional que soporte evaluaciones virtuales, generando exámenes únicos para cada estudiante, con retroalimentación automatizada, permitiendo el manejo de múltiples monedas.

Los objetivos específicos son los siguientes:

2.1. Diseño y optimización de la base de datos.

2.1.1. Crear una estructura de base de datos relacional robusta que garantice integridad referencial y un rendimiento eficiente.

2.1.2. Implementar índices en tablas clave (como `estudiante_ID`, `examen_ID` y `plantilla_ID`) para optimizar consultas y mejorar tiempos de respuesta.

2.2. Generación dinámica de exámenes.

2.2.1. Desarrollar un sistema automatizado que genere exámenes únicos y personalizados para cada estudiante, utilizando criterios como nivel de dificultad y asignatura.

2.2.2. Crear algoritmos eficientes para la selección aleatoria de preguntas desde la tabla **Pregunta**, basándose en las plantillas definidas.

2.3. Retroalimentación automatizada

2.3.1. Implementar una funcionalidad que registre resultados y genere retroalimentación personalizada automáticamente, utilizando condiciones predefinidas.

2.3.2. Almacenar los resultados de cada evaluación en la tabla **Resultado_Examen** para futuras consultas y reportes.

2.4. Reportes automatizados y análisis de datos

2.4.1. Desarrollar consultas avanzadas y reportes personalizados que permitan:

- Analizar el desempeño de estudiantes por materia, nivel de dificultad o grupo.
- Evaluar métricas clave de rendimiento académico.

2.4.2. Crear visualizaciones y reportes dinámicos para clientes institucionales.

2.5. Escalabilidad y rendimientos

2.5.1. Diseñar la base de datos con un enfoque escalable, asegurando que soporte un alto volumen de estudiantes y evaluaciones simultáneas.

2.5.2. Utilizar infraestructura en la nube para almacenamiento, procesamiento y gestión eficiente de grandes volúmenes de datos.

2.6. Seguridad de los datos

2.6.1. Implementar medidas de seguridad robustas para proteger la integridad y confidencialidad de los datos (encriptación de información sensible, accesos controlados, respaldos periódicos).

2.6.2. Asegurar el cumplimiento de normativas internacionales de protección de datos (como GDPR).

2.7. Compatibilidad e integración

2.7.1. Asegurar que la base de datos sea compatible con otras plataformas tecnológicas existentes dentro de EducaTech Perú.

2.7.2. Facilitar la integración con herramientas externas para la facturación, gestión de pagos y generación de reportes.

3. Resumen de funcionalidades, alcances y limitaciones de la Base de Datos

Funcionalidades de la Base de Datos

1. **Gestión de Estudiantes**
    - Registro de información básica de los estudiantes (nombre, correo, fecha de registro)..
    - Garantía de correos únicos para evitar duplicidad.
2. **Gestión de Materias y Preguntas**
    - Almacenamiento de materias disponibles.
    - Registro de preguntas con niveles de dificultad (fácil, intermedio, difícil) y asociadas a materias.
3. **Configuración de Plantillas de Exámenes**
    - Creación de plantillas que determinan la cantidad de preguntas por nivel de dificultad.
    - Restricción para que la suma de preguntas no exceda las 100.
4. **Aplicación y Resultados de Exámenes**

- Registro de exámenes realizados por los estudiantes, asociándose a una plantilla y fecha.
- Almacenamiento de resultados con puntuación y retroalimentación.

5. **Consultas y reportes**
    - Obtener detalles de los exámenes realizados por un estudiante..
    - Calcular el promedio de puntuación de un estudiante.
    - Visualización de los resultados y desempeño de los exámenes.
6. **Optimización y restricciones**
    - Uso de índices para mejorar la velocidad de búsqueda en consultas frecuentes.
    - Restricciones mediante triggers para validar datos (puntuación, duplicados y límites de preguntas).
7. **Procedimientos y funciones**
    - Procedimiento para registrar un examen junto con sus resultados.
    - Función para calcular el promedio de puntuación de un estudiante.
    - Procedimiento para mostrar los detalles de los exámenes realizados por un estudiante.

Alcances de la Base de datos

1. **Escalabilidad**
    - La estructura permite agregar nuevas materias, preguntas y estudiantes sin afectar las relaciones existentes.
2. **Integridad referencial**
    - Se aseguran las relaciones entre tablas mediante claves foráneas.
    - Se evitan errores como duplicar resultados con triggers.
3. **Rendimiento optimizado**

- Índices definidos para columnas clave mejoran el rendimiento de consultas complejas.
- Procedimientos almacenados permiten automatizar tareas repetitivas.

4. **Validaciones robustas**
    - Triggers para asegurar que los datos ingresados sean correctos (por ejemplo, puntuaciones entre 0 y 100).
    - Restricciones para mantener límites en configuraciones de exámenes.
5. **Adaptabilidad:**
    - Permite la creación de nuevas plantillas y exámenes, adaptándose a diferentes materias y niveles de dificultad.
6. **Reportes detallados**
    - Generación de reportes del desempeño de los estudiantes, promedio de calificaciones y detalles de exámenes realizados.

Limitaciones de la Base de datos

1. **Dependencia del esquema rígido**
    - Si se requiere añadir atributos adicionales a alguna tabla (como más detalles de estudiantes o preguntas), será necesario modificar el esquema y scripts existentes.
2. **Límite de validaciones en triggers**
    - Los triggers no pueden realizar validaciones complejas como conteos o cálculos sobre grandes volúmenes de datos en tiempo real debido a restricciones de rendimiento.
3. **Carga de datos limitada**

- La base de datos no incluye mecanismos automatizados de carga masiva de datos externos, lo que puede ralentizar su implementación inicial.

4. **Capacidad de personalización**
    - No se consideran campos como “tiempo de respuesta de preguntas” o “estado del examen” (por ejemplo, completado, pendiente), lo que podría ser requerido en sistemas más avanzados.
5. **Mantenimiento de datos históricos**
    - La estructura no contempla un manejo detallado de versiones de preguntas, plantillas o exámenes, lo que podría ser útil para cambios futuros.
6. **Almacenamiento y rendimiento**
    - En grandes volúmenes de datos, el rendimiento de algunos procedimientos o consultas podría degradarse si no se optimizan correctamente.
7. **Seguridad básica**
    - Aunque el esquema valida restricciones lógicas, no contempla medidas avanzadas de seguridad (como cifrado de datos sensibles o control de accesos a nivel de usuario).

4. Procesos de negocio

4.1. Registro de Estudiantes

4.1.1. **Inicio del Proceso:** El estudiante desea registrarse en la plataforma.

4.1.2. **Actividades Principales:**

4.1.2.1. **Solicitud de Registro:** El estudiante ingresa su información básica (nombre, correo electrónico, etc.).

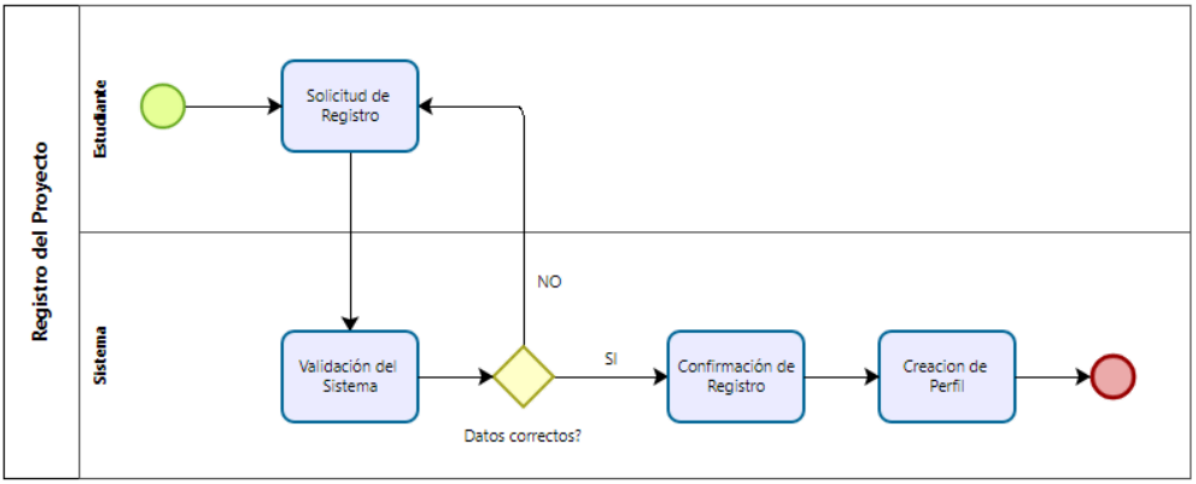
4.1.2.2. **Validación de Datos:** El sistema valida los datos del estudiante.

4.1.2.3. **Confirmación de Registro:** El estudiante recibe un correo de confirmación.

4.1.2.4. **Creación de Perfil:** El sistema genera un perfil único para el estudiante.

4.1.3. **Fin del Proceso:** El estudiante queda registrado y puede acceder al sistema.

4.1.4. **Actores:** Estudiante, Sistema.



4.2. Creacion y Gestion del Banco de Preguntas

4.2.1. **Inicio del Proceso:** El administrador o profesor desea crear preguntas para el banco.

4.2.2. **Actividades Principales:**

4.2.2.1. **Ingreso de Pregunta:** El administrador introduce una nueva pregunta.

4.2.2.2. **Asignación de Materia:** La pregunta se asocia a una materia específica.

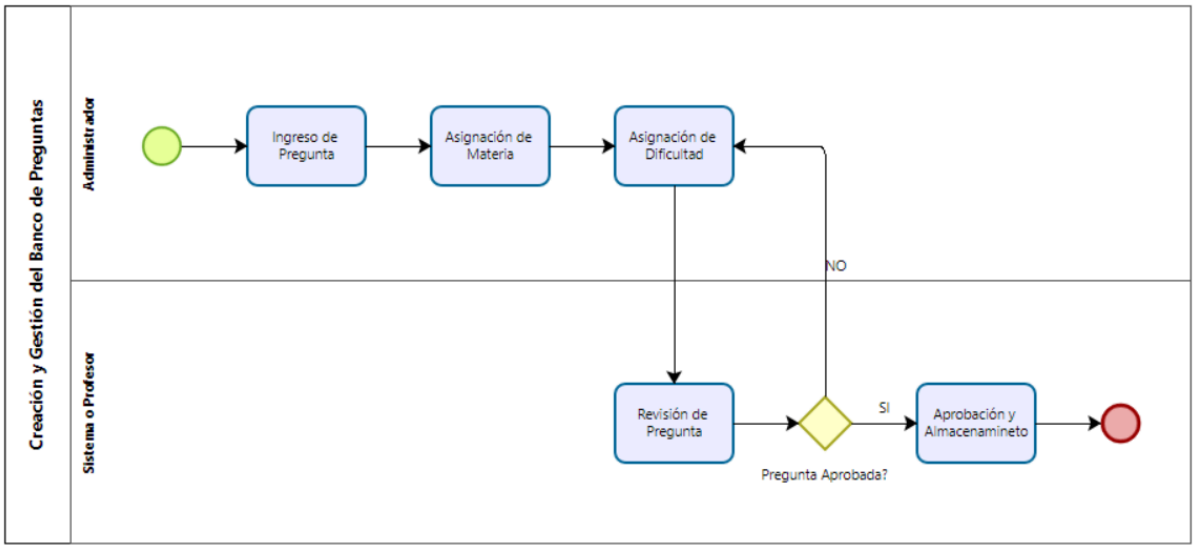
4.2.2.3. **Asignación de Dificultad:** Se asigna el nivel de dificultad (fácil, intermedio, difícil).

4.2.2.4. **Revisión de Pregunta:** El sistema o un profesor revisa la pregunta para asegurar su calidad.

4.2.2.5. **Aprobación y Almacenamiento:** La pregunta se aprueba y se almacena en el banco.

4.2.3. **Fin del Proceso:** La pregunta queda disponible para usarse en los exámenes.

4.2.4. **Actores:** Administrador, Sistema, Profesor.



4.3. Creación de Plantilla de Evaluación

4.3.1. **Inicio del Proceso:** El profesor desea crear una plantilla para una evaluación.

4.3.2. **Actividades Principales:**

4.3.2.1. **Definición de Materias:** El profesor selecciona las materias para la evaluación.

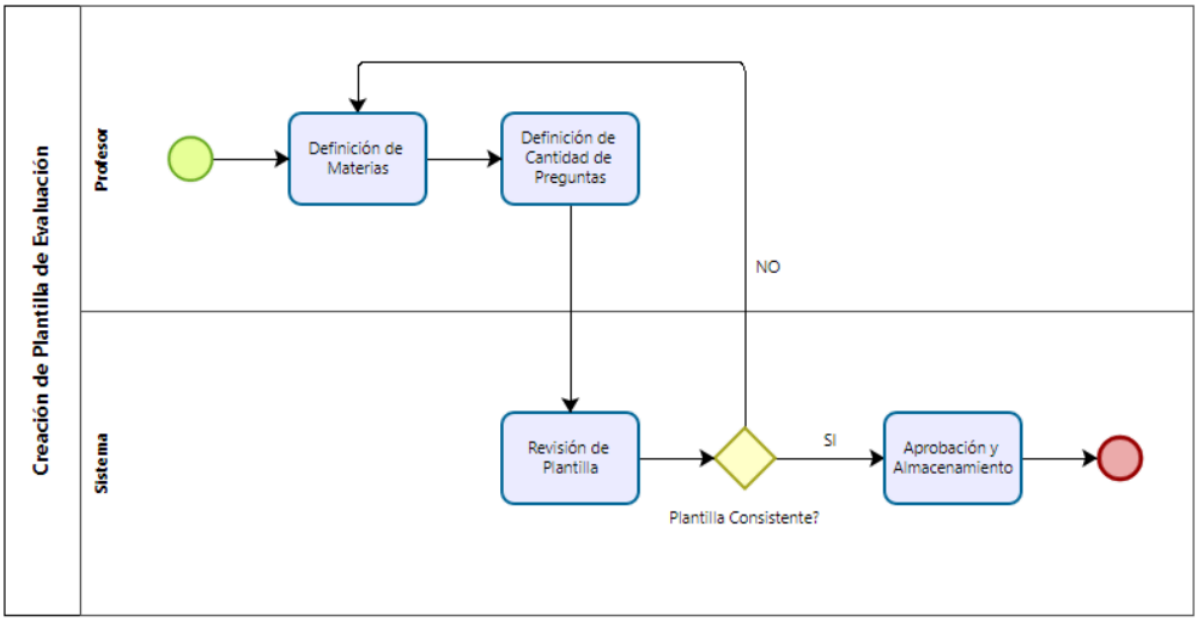
4.3.2.2. **Definición de Cantidad de Preguntas:** El profesor establece la cantidad de preguntas por nivel de dificultad (fácil, intermedio, difícil).

4.3.2.3. **Revisión de Plantilla:** El sistema revisa la plantilla para verificar consistencia.

4.3.2.4. **Aprobación de Plantilla:** La plantilla es aprobada y almacenada.

4.3.3. **Fin del Proceso:** La plantilla está lista para ser usada en exámenes.

4.3.4. **Actores:** Profesor, Sistema.



4.4. Generación de Examen Único

4.4.1. **Inicio del Proceso:** El estudiante inicia una evaluación en la plataforma.

4.4.2. **Actividades Principales:**

4.4.2.1. **Selección de Plantilla:** El sistema selecciona la plantilla correspondiente para el examen.

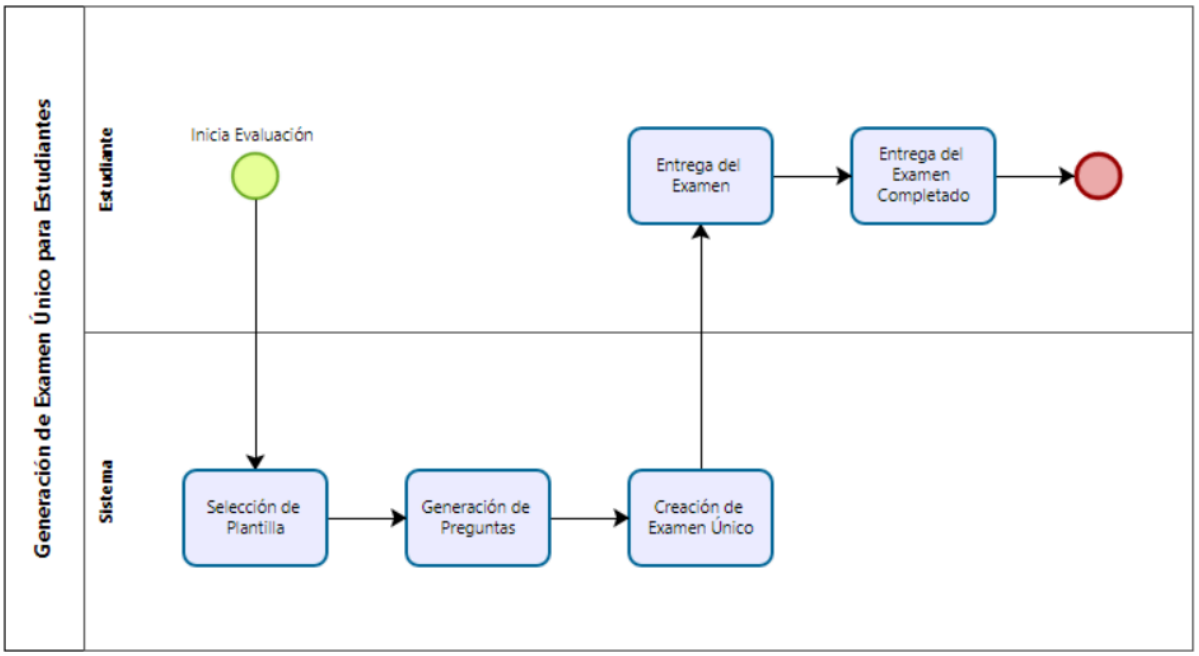
4.4.2.2. **Generación de Preguntas:** El sistema selecciona las preguntas de acuerdo a la plantilla (tomando en cuenta materia y nivel de dificultad).

4.4.2.3. **Creación de Examen Único:** El sistema genera un examen único para el estudiante.

4.4.2.4. **Entrega del Examen:** El estudiante recibe el examen y comienza a resolverlo.

4.4.3. **Fin del Proceso:** El estudiante completa y entrega el examen.

4.4.4. **Actores:** Estudiante, Sistema.



4.5. Calificación y retroalimentación

4.5.1. **Inicio del Proceso:** El examen ha sido completado por el estudiante.

4.5.2. **Actividades Principales:**

4.5.2.1. **Calificación Automática:** El sistema califica automáticamente el examen (si las preguntas son de opción múltiple o similares).

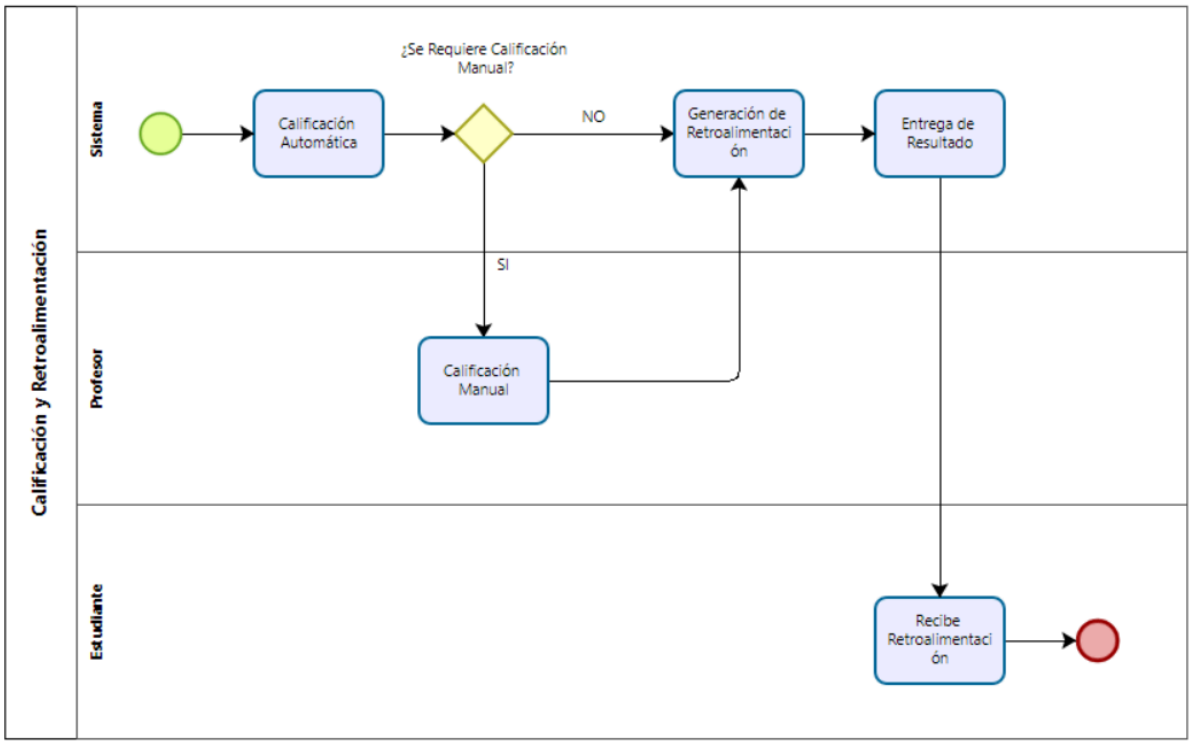
4.5.2.2. **Calificación Manual:** Si es necesario, un profesor revisa las preguntas que requieren evaluación manual.

4.5.2.3. **Generación de Retroalimentación:** El sistema genera la retroalimentación para el estudiante (puntuación y comentarios).

4.5.2.4. **Entrega de Resultado:** El sistema notifica al estudiante sobre sus resultados y retroalimentación.

4.5.3. **Fin del Proceso:** El estudiante recibe la retroalimentación y calificación final.

4.5.4. **Actores:** Estudiante, Sistema, Profesor (en caso de evaluación manual).



4.6. Respaldo y seguridad de la Base de Datos

4.6.1. **Inicio del Proceso:** Se ejecuta una rutina de respaldo y seguridad de la base de datos.

4.6.2. **Actividades Principales:**

4.6.2.1. **Generación de Copia de Seguridad:** El sistema genera una copia de la base de datos.

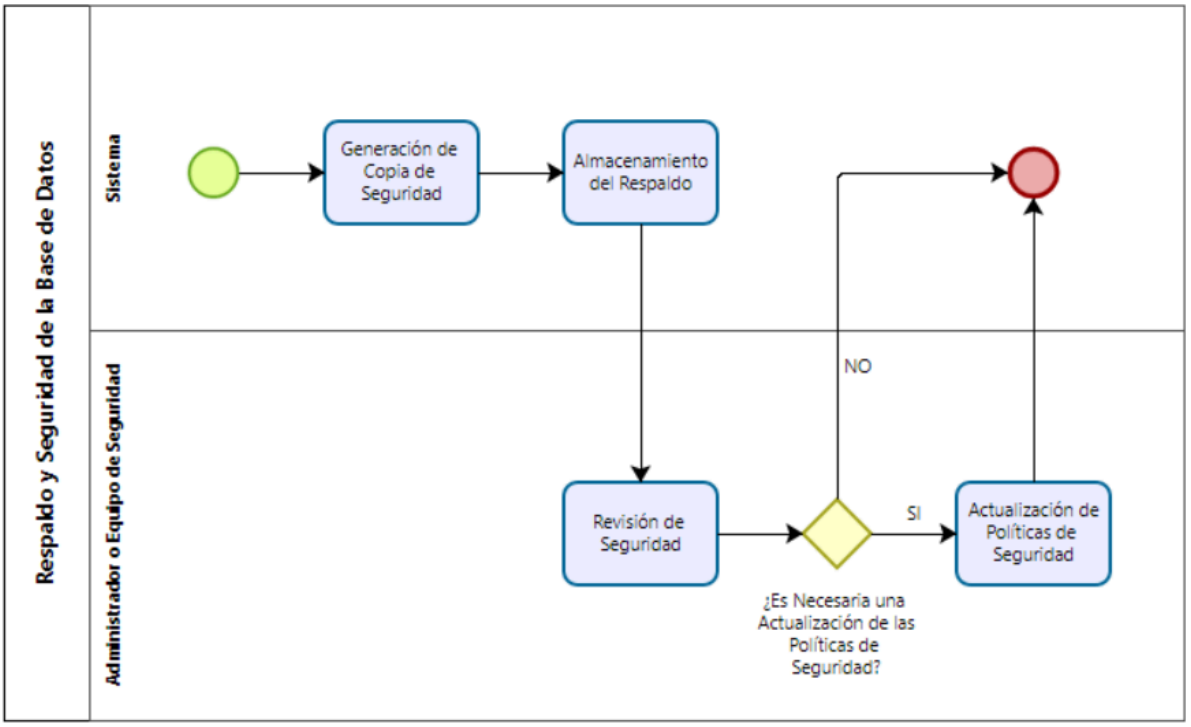
4.6.2.2. **Almacenamiento del Respaldo:** El respaldo se almacena en una ubicación segura (en la nube o local).

4.6.2.3. **Revisión de Seguridad:** Se ejecutan auditorías de seguridad en el sistema.

4.6.2.4. **Actualización de Políticas de Seguridad:** Las políticas de seguridad se actualizan de acuerdo con los resultados de las auditorías.

4.6.3. **Fin del Proceso:** El sistema queda respaldado y protegido.

4.6.4. **Actores:** Administrador, Sistema.



5. Reglas de negocio

Las reglas de negocio se derivan directamente de los procesos y aseguran que el sistema opere de manera coherente con los objetivos de EducaTech:

5.1. **Cada examen es único:** Los exámenes se generan automáticamente para que no haya dos estudiantes con las mismas preguntas, garantizando la originalidad.

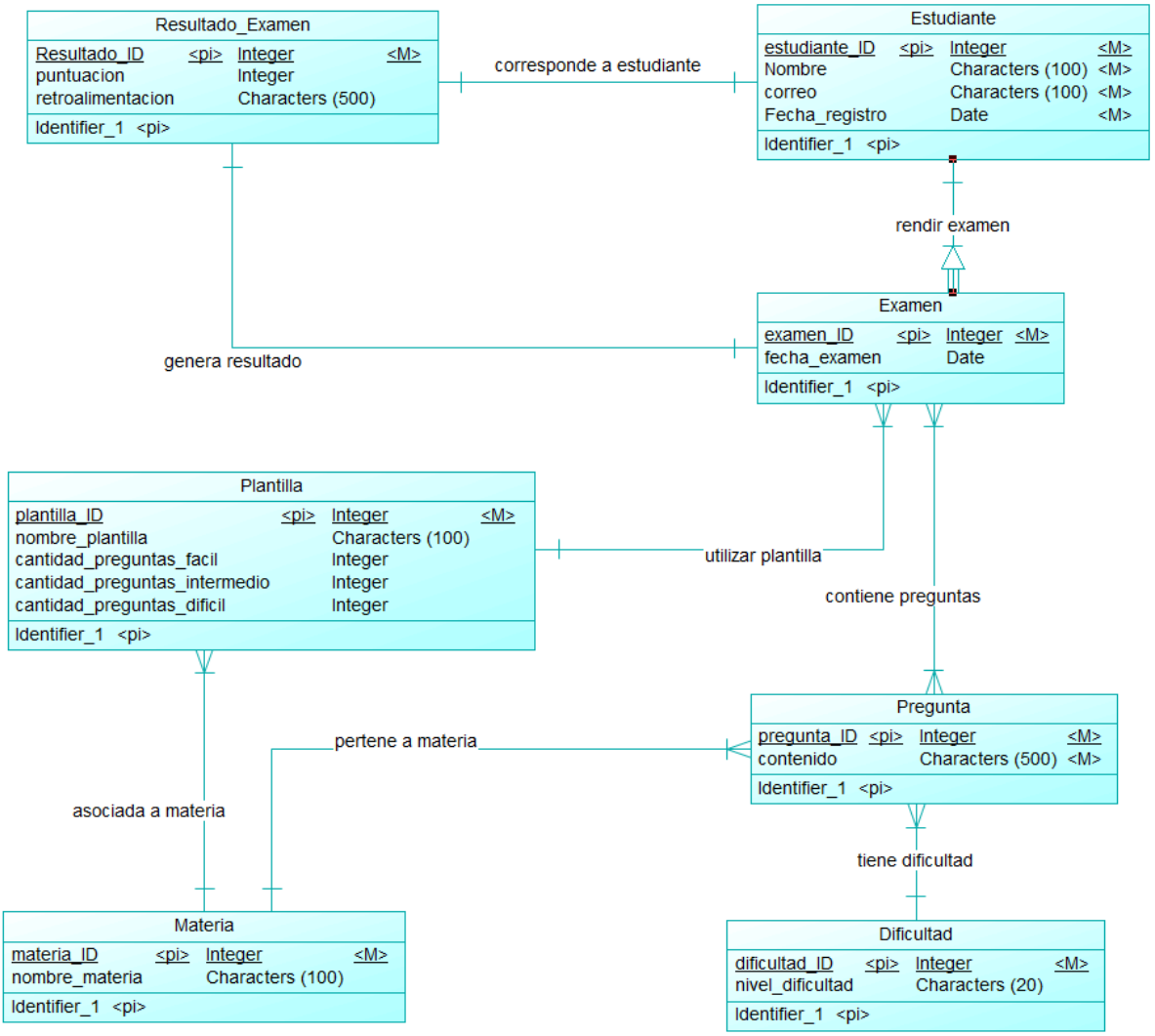
5.2. **Clasificación por dificultad:** Las plantillas deben incluir preguntas de distintos niveles de dificultad, y deben respetarse los porcentajes configurados por el administrador.

5.3. **Feedback inmediato:** El sistema debe calificar automáticamente las evaluaciones y proporcionar retroalimentación inmediata, mostrando respuestas correctas e incorrectas.

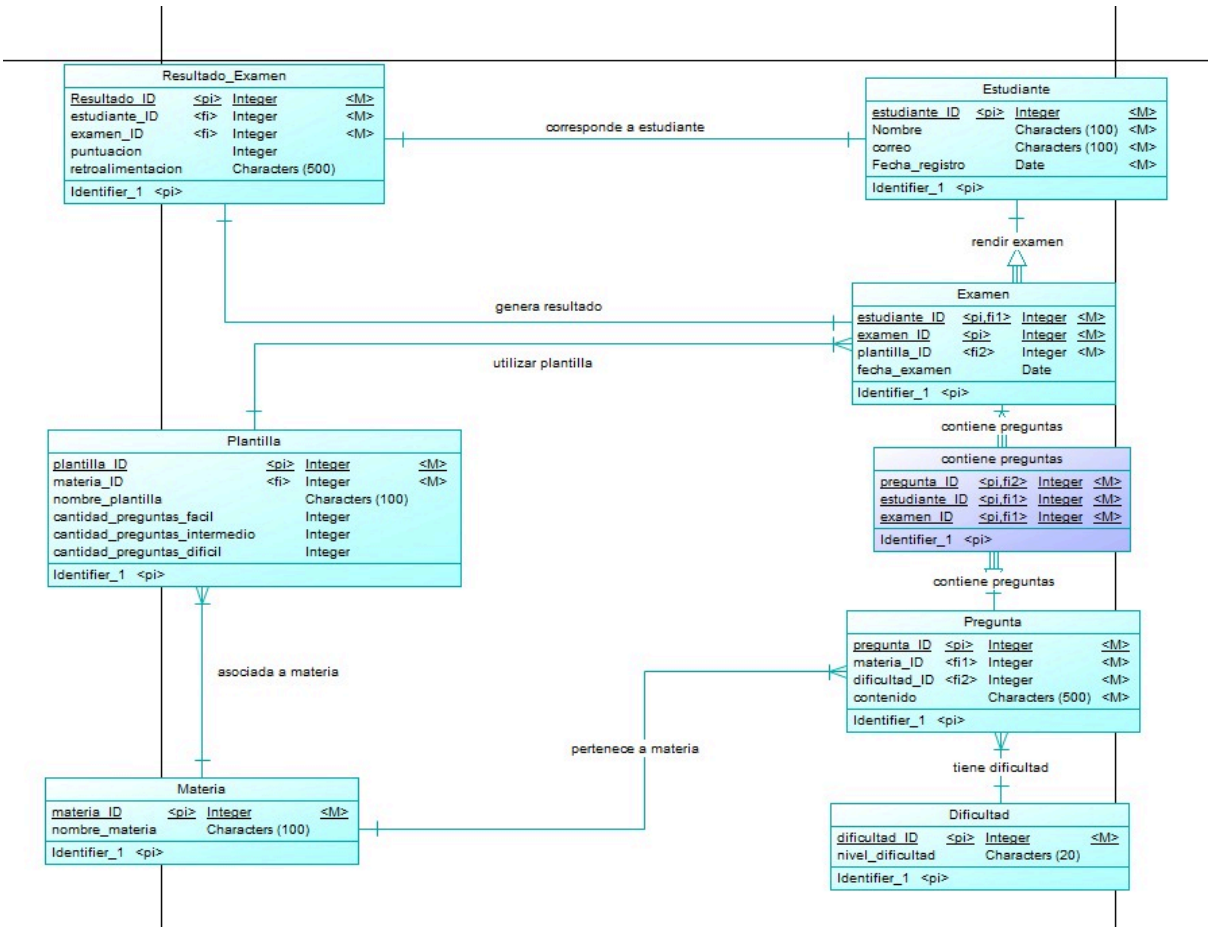
5.4. **Soporte multimoneda:** Todas las transacciones deben registrar la moneda utilizada y permitir conversiones en tiempo real basadas en tasas de cambio actualizadas.

5.5. **Seguridad:** Todas las operaciones deben cumplir con altos estándares de seguridad, incluyendo el cifrado de datos sensibles y el acceso controlado por roles.

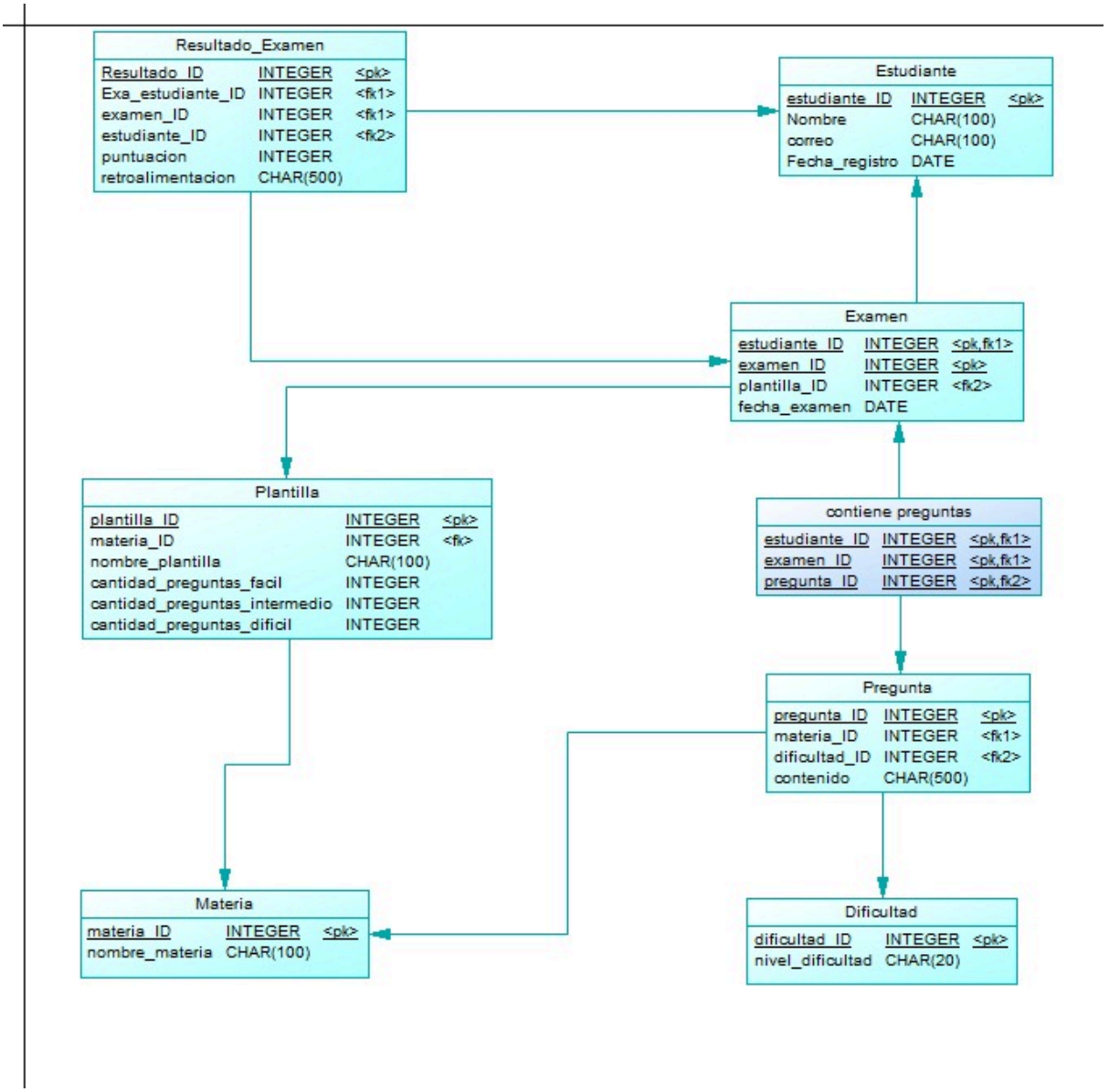
6. Modelo de datos conceptual



7. Modelo de datos lógico



8. Modelo de datos físico



9. Relación de objetos de base de datos

1. **Tabla Materia**
   - **Objetivo:** Almacenar información sobre las materias disponibles en el sistema.
   - **Atributos:**
     - `materia_ID`: Identificador único de la materia.
     - `nombre_materia`: Nombre de la materia.
2. **Tabla Dificultad**
   - **Objetivo:** Almacenar los niveles de dificultad posibles para las preguntas de los exámenes.
   - **Atributos:**
     - `dificultad_ID`: Identificador único del nivel de dificultad.

        - `nivel_dificultad`: Descripción del nivel de dificultad (Fácil, Intermedio, Difícil).
3. **Tabla Estudiante**
    - **Objetivo:** Almacenar información sobre los estudiantes registrados en el sistema.
    - **Atributos:**
        - `estudiante_ID`: Identificador único del estudiante.
        - `nombre`: Nombre completo del estudiante.
        - `correo`: Correo electrónico del estudiante (único).
        - `fecha_registro`: Fecha en la que el estudiante se registró en el sistema.
4. **Tabla Plantilla**
    - **Objetivo:** Definir las plantillas de los exámenes, asociando materias con preguntas de diferentes niveles de dificultad.
    - **Atributos:**
        - `plantilla_ID`: Identificador único de la plantilla.
        - `materia_ID`: Relación con la tabla Materia.
        - `nombre_plantilla`: Nombre de la plantilla.
        - `cantidad_preguntas_facil`: Cantidad de preguntas de nivel fácil.
        - `cantidad_preguntas_intermedio`: Cantidad de preguntas de nivel intermedio.
        - `cantidad_preguntas_dificil`: Cantidad de preguntas de nivel difícil.
    - **Restricciones:** Relación con la tabla Materia (`materia_ID`).
5. **Tabla Pregunta**
    - **Objetivo:** Almacenar las preguntas de los exámenes.
    - **Atributos:**
        - `pregunta_ID`: Identificador único de la pregunta.
        - `materia_ID`: Relación con la tabla Materia.
        - `dificultad_ID`: Relación con la tabla Dificultad.
        - `contenido`: Texto de la pregunta.
    - **Restricciones:** Relación con la tabla Materia (`materia_ID`) y Dificultad (`dificultad_ID`).
6. **Tabla Examen**
    - **Objetivo:** Almacenar la información de cada examen tomado por un estudiante.
    - **Atributos:**
        - `examen_ID`: Identificador único del examen.
        - `estudiante_ID`: Relación con la tabla Estudiante.
        - `plantilla_ID`: Relación con la tabla Plantilla.

        - `fecha_examen`: Fecha en que se realizó el examen.
    - **Restricciones:** Relación con la tabla Estudiante (`estudiante_ID`) y Plantilla (`plantilla_ID`).
7. **Tabla Resultado_Examen**
    - **Objetivo:** Almacenar los resultados de los exámenes, incluyendo la puntuación y retroalimentación.
    - **Atributos:**
        - `resultado_ID`: Identificador único del resultado.
        - `estudiante_ID`: Relación con la tabla Estudiante.
        - `examen_ID`: Relación con la tabla Examen.
        - `puntuacion`: Puntuación obtenida por el estudiante.
        - `retroalimentacion`: Comentarios sobre el desempeño del estudiante.
    - **Restricciones:** Relación con la tabla Estudiante (`estudiante_ID`) y Examen (`examen_ID`).
8. **Tabla Examen_Pregunta**
    - **Objetivo:** Establecer la relación entre un examen y las preguntas que contiene.
    - **Atributos:**
        - `pregunta_ID`: Relación con la tabla Pregunta.
        - `estudiante_ID`: Relación con la tabla Estudiante.
        - `examen_ID`: Relación con la tabla Examen.
    - **Restricciones:** Relación con las tablas Pregunta (`pregunta_ID`), Estudiante (`estudiante_ID`), y Examen (`examen_ID`).

Índices

- **Índice `idx_pregunta_materia`:** Indexa la columna `materia_ID` de la tabla Pregunta.
- **Índice `idx_pregunta_dificultad`:** Indexa la columna `dificultad_ID` de la tabla Pregunta.
- **Índice `idx_examen_estudiante`:** Indexa la columna `estudiante_ID` de la tabla Examen.
- **Índice `idx_examen_plantilla`:** Indexa la columna `plantilla_ID` de la tabla Examen.
- **Índice `idx_resultado_examen`:** Indexa la columna `examen_ID` de la tabla Resultado_Examen.
- **Índice `idx_examen_pregunta`:** Indexa la columna `examen_ID` de la tabla Examen_Pregunta.

Triggers

1. **Trigger `trg_prevenir_resultado_duplicado`:**
    - **Objetivo:** Evitar que un estudiante registre más de un resultado por examen.
    - **Condición:** Se verifica antes de insertar un nuevo registro en `Resultado_Examen`.
2. **Trigger `trg_validar_puntuacion`:**
    - **Objetivo:** Asegurarse de que la puntuación del examen esté entre 0 y 100.
    - **Condición:** Se verifica antes de insertar o actualizar un registro en `Resultado_Examen`.
3. **Trigger `trg_validar_plantilla_preguntas`:**
    - **Objetivo:** Validar que la suma de las preguntas de diferentes niveles de dificultad no supere 100.
    - **Condición:** Se verifica antes de insertar o actualizar un registro en `Plantilla`.

Procedimientos y Funciones

1. **Procedimiento `Registrar_Examen_Resultado`:**
    - **Objetivo:** Registrar un nuevo examen y su resultado.
    - **Parámetros:** `examen_ID`, `estudiante_ID`, `plantilla_ID`, `fecha_examen`, `puntuacion`, `retroalimentacion`.
2. **Función `Obtener_Promedio_Estudiante`:**
    - **Objetivo:** Calcular el promedio de puntuación de un estudiante.
    - **Parámetro:** `estudiante_ID`.
3. **Procedimiento `Mostrar_Detalle_Examen`:**
    - **Objetivo:** Mostrar el detalle de los exámenes realizados por un estudiante.
    - **Parámetro:** `estudiante_ID`.

10. Esquema de base de datos: Base de datos en Oracle Database

- **Scripts de Creación del Esquema de Base de Datos**

```
-- Tabla Materia
CREATE TABLE Materia (
   materia_ID INT PRIMARY KEY,
   nombre_materia VARCHAR(50) NOT NULL
);

-- Tabla Dificultad
CREATE TABLE Dificultad (
   dificultad_ID INT PRIMARY KEY,
   nivel_dificultad VARCHAR(20) NOT NULL
);

-- Tabla Estudiante
CREATE TABLE Estudiante (
   estudiante_ID INT PRIMARY KEY,
   nombre VARCHAR(100) NOT NULL,
   correo VARCHAR(100) UNIQUE NOT NULL,
   fecha_registro DATE DEFAULT SYSDATE
);

-- Tabla Plantilla
CREATE TABLE Plantilla (
   plantilla_ID INT PRIMARY KEY,
   materia_ID INT,
   nombre_plantilla VARCHAR(100) NOT NULL,
   cantidad_preguntas_facil INT,
   cantidad_preguntas_intermedio INT,
   cantidad_preguntas_dificil INT,
   FOREIGN KEY (materia_ID) REFERENCES Materia(materia_ID)
);

-- Tabla Pregunta
CREATE TABLE Pregunta (
   pregunta_ID INT PRIMARY KEY,
   materia_ID INT,
   dificultad_ID INT,
   contenido VARCHAR(500) NOT NULL,
   FOREIGN KEY (materia_ID) REFERENCES Materia(materia_ID),
   FOREIGN KEY (dificultad_ID) REFERENCES Dificultad(dificultad_ID)
);
```

```
-- Tabla Examen
CREATE TABLE Examen (
    examen_ID INT PRIMARY KEY,
    estudiante_ID INT,
    plantilla_ID INT,
    fecha_examen DATE DEFAULT SYSDATE,
    FOREIGN KEY (estudiante_ID) REFERENCES Estudiante(estudiante_ID),
    FOREIGN KEY (plantilla_ID) REFERENCES Plantilla(plantilla_ID)
);

-- Tabla Resultado_Examen
CREATE TABLE Resultado_Examen (
    resultado_ID INT PRIMARY KEY,
    estudiante_ID INT,
    examen_ID INT,
    puntuacion INT,
    retroalimentacion VARCHAR(255),
    FOREIGN KEY (estudiante_ID) REFERENCES Estudiante(estudiante_ID),
    FOREIGN KEY (examen_ID) REFERENCES Examen(examen_ID)
);

-- Tabla Examen_Pregunta (Relación entre Examen y Pregunta)
CREATE TABLE Examen_Pregunta (
    pregunta_ID INT,
    estudiante_ID INT,
    examen_ID INT,
    PRIMARY KEY (pregunta_ID, estudiante_ID, examen_ID),
    FOREIGN KEY (pregunta_ID) REFERENCES Pregunta(pregunta_ID),
    FOREIGN KEY (estudiante_ID) REFERENCES Estudiante(estudiante_ID),
    FOREIGN KEY (examen_ID) REFERENCES Examen(examen_ID)
);
```

- **Scripts de Índices para Optimización**

```
-- Índices para claves foráneas y consultas frecuentes
CREATE INDEX idx_pregunta_materia ON Pregunta(materia_ID);
CREATE INDEX idx_pregunta_dificultad ON Pregunta(dificultad_ID);
CREATE INDEX idx_examen_estudiante ON Examen(estudiante_ID);
CREATE INDEX idx_examen_plantilla ON Examen(plantilla_ID);
CREATE INDEX idx_resultado_examen ON Resultado_Examen(examen_ID);
CREATE INDEX idx_examen_pregunta ON Examen_Pregunta(examen_ID);
```

- **Scripts (Triggers)**

```
-- Trigger para asegurar que un estudiante no pueda insertar más de un resultado por examen
```

```
CREATE OR REPLACE TRIGGER trg_prevent_duplicate_result
BEFORE INSERT ON Resultado_Examen
FOR EACH ROW
DECLARE
   v_count INT;
BEGIN
   SELECT COUNT(*) INTO v_count
   FROM Resultado_Examen
   WHERE estudiante_ID = :NEW.estudiante_ID
     AND examen_ID = :NEW.examen_ID;

   IF v_count > 0 THEN
      RAISE_APPLICATION_ERROR(-20001, 'El estudiante ya tiene un resultado registrado
para este examen.');
   END IF;
END;
/

-- Trigger para validar puntuación entre 0 y 100
CREATE OR REPLACE TRIGGER trg_validate_score
BEFORE INSERT OR UPDATE ON Resultado_Examen
FOR EACH ROW
BEGIN
   IF :NEW.puntuacion < 0 OR :NEW.puntuacion > 100 THEN
      RAISE_APPLICATION_ERROR(-20002, 'La puntuación debe estar entre 0 y 100.');
   END IF;
END;
/

-- Trigger para validar la cantidad total de preguntas en la plantilla
CREATE OR REPLACE TRIGGER trg_validate_plantilla_preguntas
BEFORE INSERT OR UPDATE ON Plantilla
FOR EACH ROW
BEGIN
   IF :NEW.cantidad_preguntas_facil + :NEW.cantidad_preguntas_intermedio +
:NEW.cantidad_preguntas_dificil > 100 THEN
      RAISE_APPLICATION_ERROR(-20003, 'La suma de preguntas fáciles, intermedias y
difíciles no puede exceder 100.');
   END IF;
END;
/
```

- **Scripts de Script de Carga de Datos de Prueba**

```
-- Insertar datos en Materia
```

```
INSERT INTO Materia (materia_ID, nombre_materia) VALUES (1, 'Matemáticas');
INSERT INTO Materia (materia_ID, nombre_materia) VALUES (2, 'Física');
INSERT INTO Materia (materia_ID, nombre_materia) VALUES (3, 'Química');

-- Insertar datos en Dificultad
INSERT INTO Dificultad (dificultad_ID, nivel_dificultad) VALUES (1, 'Fácil');
INSERT INTO Dificultad (dificultad_ID, nivel_dificultad) VALUES (2, 'Intermedio');
INSERT INTO Dificultad (dificultad_ID, nivel_dificultad) VALUES (3, 'Difícil');

-- Insertar datos en Estudiante
INSERT INTO Estudiante (estudiante_ID, nombre, correo, fecha_registro)
VALUES (1, 'Juan Pérez', 'juan.perez@mail.com', SYSDATE);
INSERT INTO Estudiante (estudiante_ID, nombre, correo, fecha_registro)
VALUES (2, 'Ana López', 'ana.lopez@mail.com', SYSDATE);

-- Insertar datos en Plantilla
INSERT INTO Plantilla (plantilla_ID, materia_ID, nombre_plantilla,
cantidad_preguntas_facil, cantidad_preguntas_intermedio, cantidad_preguntas_dificil)
VALUES (1, 1, 'Examen Final Matemáticas', 5, 3, 2);
INSERT INTO Plantilla (plantilla_ID, materia_ID, nombre_plantilla,
cantidad_preguntas_facil, cantidad_preguntas_intermedio, cantidad_preguntas_dificil)
VALUES (2, 2, 'Examen Física Básica', 4, 4, 2);

-- Insertar datos en Pregunta
INSERT INTO Pregunta (pregunta_ID, materia_ID, dificultad_ID, contenido)
VALUES (1, 1, 1, '¿Cuánto es 2 + 2?');
INSERT INTO Pregunta (pregunta_ID, materia_ID, dificultad_ID, contenido)
VALUES (2, 1, 2, 'Deriva la función f(x) = 3x^2.');
INSERT INTO Pregunta (pregunta_ID, materia_ID, dificultad_ID, contenido)
VALUES (3, 2, 3, 'Explica la teoría del campo eléctrico.');

-- Insertar datos en Examen
INSERT INTO Examen (examen_ID, estudiante_ID, plantilla_ID, fecha_examen)
VALUES (1, 1, 1, SYSDATE);
INSERT INTO Examen (examen_ID, estudiante_ID, plantilla_ID, fecha_examen)
VALUES (2, 2, 2, SYSDATE);

-- Insertar datos en Resultado_Examen
INSERT INTO Resultado_Examen (resultado_ID, estudiante_ID, examen_ID, puntuacion,
retroalimentacion)
VALUES (1, 1, 1, 85, 'Buen trabajo, sigue practicando.');
INSERT INTO Resultado_Examen (resultado_ID, estudiante_ID, examen_ID, puntuacion,
retroalimentacion)
VALUES (2, 2, 2, 90, 'Excelente desempeño.');
```

```
-- Insertar datos en Examen_Pregunta
INSERT INTO Examen_Pregunta (pregunta_ID, estudiante_ID, examen_ID)
VALUES (1, 1, 1);
INSERT INTO Examen_Pregunta (pregunta_ID, estudiante_ID, examen_ID)
VALUES (2, 2, 2);
```

- **Scripts de Subprogramas Almacenados para Demostración de Funcionalidad**
  a. Procedimiento para Registrar un Examen y su Resultado

```
CREATE OR REPLACE PROCEDURE Registrar_Examen_Resultado (
   p_examen_ID IN INT,
   p_estudiante_ID IN INT,
   p_plantilla_ID IN INT,
   p_fecha_examen IN DATE,
   p_puntuacion IN INT,
   p_retroalimentacion IN VARCHAR
) AS
BEGIN
   -- Insertar en Examen
   INSERT INTO Examen (examen_ID, estudiante_ID, plantilla_ID,
fecha_examen)
   VALUES (p_examen_ID, p_estudiante_ID, p_plantilla_ID,
p_fecha_examen);

   -- Insertar en Resultado_Examen
   INSERT INTO Resultado_Examen (resultado_ID, estudiante_ID,
examen_ID, puntuacion, retroalimentacion)
   VALUES (p_examen_ID, p_estudiante_ID, p_examen_ID, p_puntuacion,
p_retroalimentacion);

   COMMIT;
   DBMS_OUTPUT.PUT_LINE('Examen y resultado registrado con éxito.');
EXCEPTION
   WHEN OTHERS THEN
      ROLLBACK;
      DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
END;
/

-- Ejecución:
BEGIN
```

```
    Registrar_Examen_Resultado(3, 1, 1, SYSDATE, 75, 'Buen desempeño
general.');
END;
/
```

b. Función para Obtener la Puntuación Promedio de un Estudiante

```
CREATE OR REPLACE FUNCTION Obtener_Promedio_Estudiante (
    p_estudiante_ID IN INT
) RETURN NUMBER IS
    v_promedio NUMBER;
BEGIN
    SELECT NVL(AVG(puntuacion), 0)
    INTO v_promedio
    FROM Resultado_Examen
    WHERE estudiante_ID = p_estudiante_ID;

    RETURN v_promedio;
END;
/

--Ejecución:
DECLARE
    v_promedio NUMBER;
BEGIN
    v_promedio := Obtener_Promedio_Estudiante(1);
    DBMS_OUTPUT.PUT_LINE('El promedio del estudiante es: ' ||
v_promedio);
END;
/
```

c. Procedimiento para Mostrar Detalles del Examen de un Estudiante

```
CREATE OR REPLACE PROCEDURE Mostrar_Detalle_Examen (
   p_estudiante_ID IN INT
) AS
BEGIN
   FOR r IN (
      SELECT E.examen_ID, P.nombre_plantilla, R.puntuacion,
R.retroalimentacion, E.fecha_examen
      FROM Examen E
      JOIN Plantilla P ON E.plantilla_ID = P.plantilla_ID
      JOIN Resultado_Examen R ON E.examen_ID = R.examen_ID AND
E.estudiante_ID = R.estudiante_ID
      WHERE E.estudiante_ID = p_estudiante_ID
   ) LOOP
      DBMS_OUTPUT.PUT_LINE('Examen ID: ' || r.examen_ID);
      DBMS_OUTPUT.PUT_LINE('Plantilla: ' || r.nombre_plantilla);
      DBMS_OUTPUT.PUT_LINE('Puntuación: ' || r.puntuacion);
      DBMS_OUTPUT.PUT_LINE('Retroalimentación: ' ||
r.retroalimentacion);
      DBMS_OUTPUT.PUT_LINE('Fecha Examen: ' || r.fecha_examen);
      DBMS_OUTPUT.PUT_LINE('----------------------------');
   END LOOP;
END;
/

--Ejecución:
BEGIN
   Mostrar_Detalle_Examen(1);
END;
/
```