**Universidad ORT Uruguay**

**Facultad de Ingeniería**

**Bernard Wand Polak**

**Base de Datos 2**

**Obligatorio - Entrega 1**

**Rodrigo Demicheli, Nro est. 103036**

**Mauricio Carballo, Nro est. 169199**

**Grupo N6B**

**Docente: Gabriel Fialco**

**Formulario de Antecedentes**

**Curso Base de Datos II**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nro. Estudiante** | 103036 | |  | | --- | | Foto | |
| **Nombre:** | Rodrigo |
| **Apellido:** | Demicheli |
| **Grupo / Turno:** | N6B |
| **¿Trabaja en algo relacionado con la carrera?**  **¿Qué tareas desempeña?** | Si, programación y bases de datos. |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Nro. Estudiante** | 169199 | |  | | --- | | Foto | |
| **Nombre:** | Mauricio |
| **Apellido:** | Carballo |
| **Grupo / Turno:** | N6B |
| **¿Trabaja en algo relacionado con la carrera?**  **¿Qué tareas desempeña?** | Si, programación y bases de datos. |
|  |  |

Tabla de contenido

[1. Análisis del problema 1](#_Toc308035426)

[1.1 Supuestos 1](#_Toc308035427)

[1.2 Modelo Relacional 1](#_Toc308035428)

[1.3 Restricciones 3](#_Toc308035429)

[1.4 Justificación de las restricciones globales 5](#_Toc308035430)

[1.4.1 Salon 5](#_Toc308035431)

[1.4.2 Inscribe 5](#_Toc308035432)

[1.4.3 Aprueba 5](#_Toc308035433)

[1.4.4 Rinde 5](#_Toc308035434)

[2. DDL de la solución 6](#_Toc308035435)

[3. Casos de prueba 8](#_Toc308035436)

# 1. Análisis del problema

## 1.1 Supuestos

**SUP1:** Para rendir un examen, este debe estar disponible.

**SUP2:** La calificación máxima en un examen es 100 puntos.

**SUP3:** La calificación de un examen para aprobarlo debe ser mayor o igual a 70 puntos.

**SUP4:** Todas las fechas y horas son GMT-0.

**SUP5:** En un mismo salón pueden tomarse diferentes exámenes al mismo tiempo.

## 1.2 Modelo Relacional

**Estudiante** (nroEstudiante, nroPasaporte, pais, nombre, apellido, ciudadResidencia)

nroEstudiante PK

(nroPasaporte, pais) AK

**Examen** (nroExamen, descripcion, disponible)

nroExamen PK

**Institucion** (nombre, pais, ciudad, direccion)

nombre PK

**Salon** (nombreInstitucion, nroSalon, nroSillaMaximo, nroSillaMinimo)

(nombreInstitucion, nroSalon) PK

nombreInstitucion FK Institucion (nombre)

**Inscribe** (nroEstudiante, nroExamen)

(nroEstudiante, nroExamen) PK

nroEstudiante FK Estudiante

nroExamen FK Examen

**Aprueba** (nroEstudiante, nroExamen, fecha, calificacion)

(nroEstudiante, nroExamen) PK

nroEstudiante FK Estudiante

nroExamen FK Examen

**Instancia\_Ex** (nroExamen, nombreInstitucion, fecha, hora)

(nroExamen, nombreInstitucion, fecha) PK

nroExamen FK Examen

nombreInstitucion FK Institucion (nombre)

**Rinde** (nroExamen, nroEstudiante, nombreInstitucion, nroSalon, fecha, nroSillaAsignado)

(nroExamen, nroEstudiante, nombreInstitucion, nroSalon, fecha) PK

(nroExamen, nroEstudiante) FK Inscribe

(nroExamen, nombreInstitucion, fecha) FK InstanciaEx

(nombreInstitucion, nroSalon) FK Salon

(nroEstudiante, fecha) AK Rinde

## 1.3 Restricciones

## 1.4 Justificación de las restricciones globales

### 1.4.1 Salon

**Motivo:** al modificar el rango máximo o mínimo de números de silla del salón, no puede quedar ninguna silla asignada a un estudiante para rendir un examen fuera del nuevo rango.

**Tablas involucradas:** Rinde, Salon.

Al rendir un Examen, a un Estudiante se le asigna un numero de silla. Ese número de silla debe estar dentro del rango mínimo y máximo de números de silla del Salon donde se va a rendir el Examen. Si se modificara dicho rango, eso no puede provocar que las sillas asignadas al Salon queden fuera del nuevo rango.

### 1.4.2 Inscribe

**Motivo:** al inscribirse un estudiante a un examen, este debe estar disponible.

**Tablas involucradas:** Inscribe, Examen.

Asumimos que un estudiante puede inscribirse a un examen solo si este está disponible, entonces al insertar o actualizar una tupla en la tabla Inscribe, hay que controlar que el examen al que se está inscribiendo el estudiante esté disponible en la tabla Examen.

### 1.4.3 Aprueba

**Motivo**: La fecha de aprobación de un examen debe ser como mínimo siete días posterior a la fecha del examen.

**Tablas involucradas:** Rinde, Instancia\_ex, Aprueba.

Esta restricción esta especificada explícitamente en la realidad del problema, y para controlar que se cumpla generamos un trigger sobre la tabla Aprueba, que en el insert o update de alguna tupla, chequea que el examen que se está señalando como aprobado, haya sido rendido como mínimo siete días antes de la fecha de aprobación del mismo.

### 1.4.4 Rinde

**Motivo**: El numero de silla debe estar dentro del rango de los números de silla de Salón.

**Tablas involucradas:** Salon, Instancia\_ex, Inscribe, Rinde.

Se está controlando que al insertar una tupla en la tabla Rinde, el campo nro\_silla\_asignado tenga un valor que este dentro del rango mínimo y máximo de números de silla del Salon donde se Rinde el Examen.

**Motivo**: El numero de silla no puede estar ocupado.

**Tablas involucradas:** Rinde.

Al insertar o actualizar una tupla de la tabla Rinde, se controla que el numero de silla ya no esté ocupado por otro Estudiante.

**Motivo**: Se debe controlar que si el estudiante ya aprobó el examen no pueda volver a rendirlo.

**Tablas involucradas:** Rinde, Aprueba.

Esta restricción se especifica en la letra del problema.

# 2. DDL de la solución

DROP TABLE aprueba CASCADE CONSTRAINTS;

DROP TABLE estudiante CASCADE CONSTRAINTS;

DROP TABLE examen CASCADE CONSTRAINTS;

DROP TABLE inscribe CASCADE CONSTRAINTS;

DROP TABLE instancia\_ex CASCADE CONSTRAINTS;

DROP TABLE institucion CASCADE CONSTRAINTS;

DROP TABLE salon CASCADE CONSTRAINTS;

DROP TABLE rinde\_data CASCADE CONSTRAINTS;

DROP VIEW rinde CASCADE CONSTRAINTS;

CREATE TABLE estudiante (

nro\_estudiante NUMBER CHECK(nro\_estudiante >=0),

nro\_pasaporte NUMBER NOT NULL CHECK(nro\_pasaporte >=0),

pais VARCHAR2(30) NOT NULL,

nombre VARCHAR2(50) NOT NULL,

apellido VARCHAR2(50) NOT NULL,

ciudad\_residencia VARCHAR2(50),

CONSTRAINT estudiante\_pk PRIMARY KEY (nro\_estudiante),

CONSTRAINT estudiante\_ak UNIQUE (nro\_pasaporte, pais)

);

CREATE TABLE examen (

nro\_examen NUMBER CHECK(nro\_examen >=0),

descripcion VARCHAR2(200),

disponible CHAR(1) CHECK (disponible IN ('S', 'N')) NOT NULL,

CONSTRAINT examen\_pk PRIMARY KEY (nro\_examen)

);

CREATE TABLE institucion (

nombre VARCHAR2(50),

pais VARCHAR2(30) NOT NULL,

ciudad VARCHAR2(50) NOT NULL,

direccion VARCHAR2 (100) NOT NULL,

CONSTRAINT institucion\_pk PRIMARY KEY (nombre)

);

CREATE TABLE salon (

nombre\_institucion VARCHAR2(50),

nro\_salon NUMBER CHECK(nro\_salon >=0),

nro\_silla\_max NUMBER NOT NULL CHECK (nro\_silla\_max >= 0),

nro\_silla\_min NUMBER NOT NULL CHECK (nro\_silla\_min >= 0),

CONSTRAINT salon\_pk PRIMARY KEY (nombre\_institucion, nro\_salon),

CONSTRAINT salon\_fk FOREIGN KEY (nombre\_institucion) REFERENCES institucion (nombre),

CONSTRAINT check\_nroSillaMax CHECK (nro\_silla\_max >= nro\_silla\_min)

);

CREATE TABLE inscribe (

nro\_estudiante NUMBER CHECK(nro\_estudiante >=0),

nro\_examen NUMBER CHECK(nro\_examen >=0),

CONSTRAINT inscribe\_pk PRIMARY KEY (nro\_estudiante, nro\_examen),

CONSTRAINT inscribe\_fk\_nro\_est FOREIGN KEY (nro\_estudiante) REFERENCES estudiante (nro\_estudiante),

CONSTRAINT inscribe\_fk\_nro\_ex FOREIGN KEY (nro\_examen) REFERENCES examen (nro\_examen)

);

CREATE TABLE aprueba (

nro\_estudiante NUMBER CHECK(nro\_estudiante >=0),

nro\_examen NUMBER CHECK(nro\_examen >=0),

fecha DATE NOT NULL,

calificacion NUMBER (3) NOT NULL CHECK(calificacion BETWEEN 70 and 100),

CONSTRAINT aprueba\_pk PRIMARY KEY (nro\_estudiante, nro\_examen),

CONSTRAINT aprueba\_fk\_nro\_est FOREIGN KEY (nro\_estudiante) REFERENCES examen (nro\_examen),

CONSTRAINT aprueba\_fk\_nro\_ex FOREIGN KEY (nro\_examen) REFERENCES examen (nro\_examen)

);

CREATE TABLE instancia\_ex (

nro\_examen NUMBER CHECK(nro\_examen >=0),

nombre\_institucion VARCHAR2(50),

fecha DATE,

hora TIMESTAMP NOT NULL,

CONSTRAINT instancia\_ex\_pk PRIMARY KEY (nro\_examen, nombre\_institucion, fecha),

CONSTRAINT instancia\_ex\_fk\_nro\_ex FOREIGN KEY (nro\_examen) REFERENCES examen (nro\_examen),

CONSTRAINT instancia\_ex\_fk\_nombre FOREIGN KEY (nombre\_institucion) REFERENCES institucion (nombre)

);

CREATE TABLE rinde\_data (

nro\_examen NUMBER CHECK(nro\_examen >=0),

nro\_estudiante NUMBER check(nro\_estudiante >=0),

nombre\_institucion VARCHAR2(50),

nro\_salon NUMBER check(nro\_salon >=0),

fecha DATE,

nro\_silla\_asignado NUMBER check(nro\_silla\_asignado >=0) NOT NULL,

CONSTRAINT rinde\_pk

PRIMARY KEY (nro\_examen, nro\_estudiante, nombre\_institucion, nro\_salon, fecha),

CONSTRAINT rinde\_fk\_inscribe

FOREIGN KEY (nro\_examen, nro\_estudiante)

REFERENCES inscribe (nro\_examen, nro\_estudiante),

CONSTRAINT rinde\_fk\_instancia\_ex

FOREIGN KEY (nro\_examen, nombre\_institucion, fecha)

REFERENCES instancia\_ex (nro\_examen, nombre\_institucion, fecha),

CONSTRAINT rinde\_fk\_salon

FOREIGN KEY (nombre\_institucion, nro\_salon)

REFERENCES salon (nombre\_institucion, nro\_salon),

CONSTRAINT rinde\_unique UNIQUE (nro\_estudiante, fecha)

);

CREATE VIEW rinde AS SELECT \* FROM rinde\_data;

Para la relación Rinde decidimos usar una tabla mutante porque uno de los triggers que se crearon en esa relación, hacia una consulta a la tabla en el momento de la ejecución del trigger. Para esto seguimos el procedimiento visto en el práctico.

# 3. Casos de prueba