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
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
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

1.1 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

Estudiante		
Restriccion	**Clasificacion**	**Implementacion**
nroEstudiante identifica al estudiante	De clave	PRIMARY KEY
(pais, nroPasaporte) identifica al estudiante	De clave	UNIQUE
pais debe tener valor	De dominio	NOT NULL
nroPasaporte debe tener valor	De dominio	NOT NULL
nroEstudiante debe ser mayor o igual que cero	De dominio	CHECK
nroPasaporte debe ser mayor o igual que cero	De dominio	CHECK

Examen		
Restriccion	**Clasificacion**	**Implementacion**
nroExamen identifica al examen	De clave	PRIMARY KEY
disponible debe tener valor	De dominio	NOT NULL
disponible debe valer 'S' o 'N'	De dominio	CHECK
nroExamen debe ser mayor o igual que cero	De dominio	CHECK

Institucion		
Restriccion	**Clasificacion**	**Implementacion**
nombre identifica la institucion	De clave	PRIMARY KEY
pais debe tener valor	De dominio	NOT NULL
ciudad debe tener valor	De dominio	NOT NULL
direccion debe tener valor	De dominio	NOT NULL

Salon		
Restriccion	**Clasificacion**	**Implementacion**
(nombreInstitucion, nroSalon) identifica el salon	De clave	PRIMARY KEY
nombreInstitucion referencia al atributo nombre de la relacion Institucion	De integridad referencial	REFERENCES
nroSillaMaximo debe tener valor	De dominio	NOT NULL
nroSillaMinimo debe tener valor	De dominio	NOT NULL
nroSalon debe ser mayor o igual que cero	De dominio	CHECK
nroSillaMaximo debe ser mayor o igual que cero	De dominio	CHECK
nroSillaMinimo debe ser mayor o igual que cero	De dominio	CHECK
nroSillaMaximo debe ser mayor o igual que nroSillaMinimo	De dominio	CHECK
No se puede modificar el rango de numeros de silla del salon si esto produce que alguna silla asignada para rendir un examen quede fuera del nuevo rango.	Global	TRIGGER

Inscribe		
Restriccion	**Clasificacion**	**Implementacion**
(nroEstudiante, nroExamen) identifica la inscripcion	De clave	PRIMARY KEY
nroEstudiante referencia a la relacion Estudiante	De integridad referencial	REFERENCES
nroExamen referencia a la relacion Examen	De integridad referencial	REFERENCES
nroEstudiante debe ser mayor o igual que cero	De dominio	CHECK
nroExamen debe ser mayor o igual que cero	De dominio	CHECK
Al inscribirse a un examen, este debe estar disponible	Global	TRIGGER

Aprueba		
Restriccion	**Clasificacion**	**Implementacion**
(nroEstudiante, nroExamen) identifica la aprobacion	De clave	PRIMARY KEY
nroEstudiante referencia a la relacion Estudiante	De integridad referencial	REFERENCES
nroExamen referencia a la relacion Examen	De integridad referencial	REFERENCES
calificacion debe tener valor	De dominio	NOT NULL
calificacion debe estar entre 0 y 100	De dominio	CHECK
fecha debe tener valor	De dominio	NOT NULL
nroEstudiante debe ser mayor o igual que cero	De dominio	CHECK
nroExamen debe ser mayor o igual que cero	De dominio	CHECK
La fecha de aprobacion de un examen debe ser como minimo siete dias posterior a la fecha del examen.	Global	TRIGGER

InstanciaEx		
Restriccion	**Clasificacion**	**Implementacion**
(nroExamen, nombreInstitucion, fecha) identifica la instancia del examen	De clave	PRIMARY KEY
nroExamen referencia a la relacion Examen	De integridad referencial	REFERENCES
nombreInstitucion referencia al atributo nombre de la relacion Institucion	De integridad referencial	REFERENCES
nroExamen debe ser mayor o igual que cero	De dominio	CHECK
hora debe tener valor	De dominio	NOT NULL

Rinde		
Restriccion	**Clasificacion**	**Implementacion**
(nroExamen, nroEstudiante, nombreInstitucion, nroSalon, fecha) identifica la tupla de la relacion	De clave	PRIMARY KEY
(nroEstudiante, nroExamen) referencia a la relacion Inscribe	De integridad referencial	FOREIGN KEY
(nroExamen, nombreInstitucion, fecha) referencia a la relacion InstanciaEx	De integridad referencial	FOREIGN KEY
(nombreInstitucion, nroSalon) referencia a la relacion Salon	De integridad referencial	FOREIGN KEY
nroExamen debe ser mayor o igual que cero	De dominio	CHECK
nroEstudiante debe ser mayor o igual que cero	De dominio	CHECK
nroSalon debe ser mayor o igual que cero	De dominio	CHECK
El mismo numero de estudiante no puede estar en mas de una tupla de rinde con a la misma fecha.	De Clave	UNIQUE
El numero de silla debe estar dentro del rango de los numeros de silla de Salon	Global	TRIGGER
Se debe controlar que si el estudiante ya aprobo el examen no pueda volver a rendirlo	Global	TRIGGER
A cada estudiante la primera vez que se inscribe a un examen, se le otorga un número de silla dentro de un salón de la institución elegida. Luego por razones de control, cada examen debe rendirlo en el mismo salón y silla a excepción de que se solape con otro estudiante, en dicho caso el sistema deberá buscar otro salón, en la misma institución, con el número de silla disponible, si no es posible deberá conseguir en el salón inicial otra silla, de no haber buscará una disponible en otro salón. Si al final no se puede encontrar silla el estudiante no podrá en dicha instancia rendir el examen.	Global	TRIGGER

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 número 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 posteriores 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 número 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: 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.

Motivo: A cada estudiante la primera vez que se inscribe a un examen, se le otorga un número de silla dentro de un salón de la institución elegida. Luego por razones de control, cada examen debe rendirlo en el mismo salón y silla a excepción de que se solape con otro estudiante, en dicho caso el sistema deberá buscar otro salón, en la misma institución, con el número de silla disponible, si no es posible deberá conseguir en el salón inicial otra silla, de no haber buscará una disponible en otro salón. Si al final no se puede encontrar silla el estudiante no podrá en dicha instancia rendir el examen.
Tablas involucradas: Rinde, Salon.

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

3.1 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.

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 2, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO institucion VALUES ('inst1', 'paraguay', 'asuncion', 'chilavert road 1354');
INSERT INTO salon VALUES ('inst1', 1, 30, 1);
INSERT INTO institucion VALUES ('inst2', 'ARGENTINA', 'BS', 'Kichener');
INSERT INTO salon VALUES ('inst2', 1, 5, 1);
INSERT INTO examen VALUES (1, 'test de cooper', 'S');
INSERT INTO inscribe VALUES (1, 1);
INSERT INTO instancia_ex VALUES (1, 'inst1',
       to_date('2011/01/01', 'yyyy/mm/dd'),
       to_date('2011/01/01 16:00:', 'YYYY/MM/DD HH24:MI:');
INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 5);
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?
Actualizo los rangos de número de silla de un Salon de manera que todas las tuplas de Rinde quedan consistentes.	`UPDATE salon SET nro_silla_min = 10, nro_silla_max = 17 WHERE nombre_institucion = 'inst2';`	1 fila actualizada.	Si.
Actualizo el rango mínimo de número de silla de un Salon de manera que alguna tupla de Rinde queda inconsistente.	`UPDATE salon SET nro_silla_min = 10 WHERE nombre_institucion = 'inst1';`	Error -20001 ,'NO SE PUEDE ODIFICAR EL RANGO DE SILLAS DEL SALON PUES YA HAY SILLAS ASIGNADAS EN LA RELACION RINDE'	Si.
Actualizo el rango máximo de número de silla de un Salon de manera que alguna tupla de Rinde queda inconsistente.	`UPDATE salon SET nro_silla_max = 2 WHERE nombre_institucion = 'inst1';`	Error -20001, 'NO SE PUEDE ODIFICAR EL RANGO DE SILLAS DEL SALON PUES YA HAY SILLAS ASIGNADAS EN LA RELACION RINDE'	Si.

3.2 Inscribe

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

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 2, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO examen VALUES (1, 'test de cooper', 'S');
INSERT INTO examen VALUES (2, 'tiro al arco', 'N');
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?
Inscribo un Estudiante en un Examen que está disponible.	`INSERT INTO inscribe VALUES (1, 1);`	1 fila insertada.	Si.
Inscribo un Estudiante en un Examen que no está disponible.	`INSERT INTO inscribe VALUES (1, 2);`	Error -20001, 'El examen debe estar disponible para poder inscribirse a él.'	Si.

3.3 Aprueba

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

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 2, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO institucion VALUES ('inst1', 'paraguay', 'asuncion', 'chilavert road 1354');
INSERT INTO salon VALUES ('inst1', 1, 30, 1);
INSERT INTO examen VALUES (1, 'test de cooper', 'S');
INSERT INTO inscribe VALUES (1, 1);

INSERT INTO instancia_ex VALUES (1, 'inst1',
                                 to_date('2011/01/01', 'yyyy/mm/dd'),
                                 to_date('2011-12-02 16:00:', 'YYYY-MM-DD HH24:MI:'));
INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 5);
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?
Apruebo un Examen 15 días después de la fecha en que se rindió.	`INSERT INTO aprueba VALUES (1, 1, to_date('2011/01/15', 'yyyy/mm/dd'), 95);`	1 fila insertada.	Si.
Apruebo un Examen 5 días después de la fecha en que se rindió.	`INSERT INTO aprueba VALUES (1, 1, to_date('2011/01/05', 'yyyy/mm/dd'), 95);`	Error -20001, 'La fecha de aprobación de un examen debe ser al menos siete días posterior a la fecha del mismo.'	Si.

3.4 Rinde

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

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 1, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (2, 2, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (3, 3, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO institucion VALUES ('inst1', 'paraguay', 'asuncion', 'chilavert road 1354');
INSERT INTO salon VALUES ('inst1', 1, 30, 1);
INSERT INTO examen VALUES (1, 'test de cooper', 'S');

INSERT INTO inscribe VALUES (1, 1);
INSERT INTO inscribe VALUES (2, 1);
INSERT INTO inscribe VALUES (3, 1);

INSERT INTO instancia_ex VALUES (1, 'inst1',
       to_date('2011/01/01', 'yyyy/mm/dd'),
       to_date('2011-12-02 16:00:', 'YYYY-MM-DD HH24:MI:'));
INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?
Inserto una tupla en Rinde donde el numero de silla esta dentro del rango de números de silla del salon.	`INSERT INTO rinde VALUES (1, 2, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`	1 fila insertada.	Si.
Actualizo una tupla en Rinde donde el numero de silla esta dentro del rango de números de silla del salon.	`UPDATE rinde SET nro_silla_asignado = 5 WHERE nro_examen = 1 AND nro_estudiante = 1 AND nombre_institucion = 'inst1' AND nro_salon = 1 AND fecha = to_date('2011/01/01', 'yyyy/mm/dd');`	1 fila actualizada.	Si.
Inserto una tupla en Rinde donde el número de silla esta fuera del rango de números de silla del salon.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1500);`	Error -20001, 'El numero de silla no es válido, debe estar dentro del rango máximo y mínimo del salon.'	Si.
Actualizo una tupla en Rinde donde el número de silla esta fuera del rango de números de silla del salon.	`UPDATE rinde SET nro_silla_asignado = 700 WHERE nro_examen = 1 AND nro_estudiante = 1 AND nombre_institucion = 'inst1' AND nro_salon = 1 AND fecha = to_date('2011/01/01', 'yyyy/mm/dd');`	Error -20001, 'El numero de silla no es válido, debe estar dentro del rango máximo y mínimo del salon.'	Si.

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

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 1, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (2, 2, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (3, 3, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO institucion VALUES ('inst1', 'paraguay', 'asuncion', 'chilavert road 1354');
INSERT INTO salon VALUES ('inst1', 1, 30, 1);
INSERT INTO examen VALUES (1, 'test de cooper', 'S');

INSERT INTO inscribe VALUES (1, 1);
INSERT INTO inscribe VALUES (2, 1);
INSERT INTO inscribe VALUES (3, 1);

INSERT INTO instancia_ex VALUES (1, 'inst1', to_date('2011/01/01', 'yyyy/mm/dd'),
to_date('2011-12-02 16:00:', 'YYYY-MM-DD HH24:MI:'));
INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);

INSERT INTO aprueba VALUES (1, 1, to_date('2011/01/10', 'yyyy/mm/dd'), 80);
INSERT INTO instancia_ex VALUES (1, 'inst1', to_date('2011/02/01', 'yyyy/mm/dd'),
to_date('2011-12-02 18:00:', 'YYYY-MM-DD HH24:MI:'));

INSERT INTO examen VALUES (2, 'test drive MINI cooper XD', 'S');
INSERT INTO instancia_ex VALUES (2, 'inst1', to_date('2011/01/01', 'yyyy/mm/dd'));
INSERT INTO inscribe VALUES (1, 2);
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?
Aprueba un examen que no aprobó.	`INSERT INTO aprueba VALUES (1, 2, to_date('2011/02/10', 'yyyy/mm/dd'), 80);`	1 fila insertada.	Si.
Rinde un examen que ya tiene aprobado.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/02/01', 'yyyy/mm/dd'), 1);`	Error, -20001,'El estudiante ya tiene aprobado el examen '	Si.
Cambio un estudiante a un examen que ya tiene aprobado.	`UPDATE rinde SET nro_examen = 2 WHERE nro_estudiante = 1 AND nombre_institucion = 'inst1' AND nro_salon = 1 AND nro_silla_asignado = 1 AND fecha = to_date('2011/01/01', 'yyyy/mm/dd');`	Error, -20001,'El estudiante ya tiene aprobado el examen '	Si.

Motivo: A cada estudiante la primera vez que se inscribe a un examen, se le otorga un número de silla dentro de un salón de la institución elegida. Luego por razones de control, cada examen debe rendirlo en el mismo salón y silla a excepción de que se solape con otro estudiante, en dicho caso el sistema deberá buscar otro salón, en la misma institución, con el número de silla disponible, si no es posible deberá conseguir en el salón inicial otra silla, de no haber buscará una disponible en otro salón. Si al final no se puede encontrar silla el estudiante no podrá en dicha instancia rendir el examen.

Considerando las tablas inicialmente vacías, inserts previos necesarios para las pruebas:

```
INSERT INTO estudiante VALUES (1, 1, 'paraguay', 'est1', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (2, 2, 'paraguay', 'est2', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (3, 3, 'paraguay', 'est3', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (4, 4, 'paraguay', 'est4', 'apellido', 'asuncion');
INSERT INTO estudiante VALUES (5, 5, 'paraguay', 'est5', 'apellido', 'asuncion');
INSERT INTO institucion VALUES ('inst1', 'paraguay', 'asuncion', 'chilavert road 1354');
INSERT INTO salon VALUES ('inst1', 1, 2, 1);
INSERT INTO salon VALUES ('inst1', 2, 2, 1);
INSERT INTO examen VALUES (1, 'test de cooper', 'S');

INSERT INTO inscribe VALUES (1, 1);
INSERT INTO inscribe VALUES (2, 1);
INSERT INTO inscribe VALUES (3, 1);
INSERT INTO inscribe VALUES (4, 1);
INSERT INTO inscribe VALUES (5, 1);

INSERT INTO instancia_ex VALUES (1, 'inst1', to_date('2011/01/01', 'yyyy/mm/dd'),
to_date('2011/01/01 16:00:', 'yyyy/mm/dd HH24:MI:'));
```

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?

La silla está disponible en el salón en el que se quiere insertar.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`INSERT INTO rinde VALUES (1, 3, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
		3 filas insertadas.	Si.

No había ningún salón que tuviera el número de silla disponible, entonces busco una silla disponible en el salón inicial.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`--Este es el insert problemático`
	`INSERT INTO rinde VALUES (1, 3, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
		2 filas insertadas.
		
		Error, -20001, 'MISMO SALON NUEVA SILLA, DEBE INSERTAR EN EL SALON 1, SILLA 2 '
			Si.

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?

La silla en el salón inicial está ocupada, entonces se busca otro salón que tenga disponible el mismo número de silla.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`--Este es el insert problemático`
	`INSERT INTO rinde VALUES (1, 3, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
		2 filas insertadas.
		
		Error, -20001, 'NUEVO SALON MISMA SILLA, DEBE INSERTAR EN EL SALON 2, SILLA 1 '
			Si.

No se encontró la silla buscada en ningún salón, entonces se busca la primera silla libre en cualquier salón.	`--aumento la cantidad de sillas disponibles`
	`UPDATE SALON SET nro_silla_max = 3 where nro_salon = 1 and nombre_institucion = 'inst1';`
	
	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`INSERT INTO rinde VALUES (1, 3, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 4, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`--Este es el insert problemático.`
	`INSERT INTO rinde VALUES (1, 5, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
		1 fila actualizada.
		
		4 filas insertadas.
		
		Error, -20001 'CUALQUIER SALON, CUALQUIER SILLA, DEBE INSERTAR EN EL SALON 1, SILLA 3'
			Si.

No puede dar el examen, no hay sillas disponibles en ningún salón de la Institución.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	`INSERT INTO rinde VALUES (1, 3, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	`INSERT INTO rinde VALUES (1, 4, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`--Este es el insert problemático`
	`INSERT INTO rinde VALUES (1, 5, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
		4 filas insertadas.
		
		Error -20001, 'NO PUEDES DAR EL EXAMEN'.
			SI.

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?

La silla está disponible en el salón en el que se quiere actualizar.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`--Este es el update exitoso`
	`UPDATE rinde`
	`SET nro_salon= 2`
	`WHERE nro_estudiante = 2;`
		2 filas insertadas.
		
		1 filas actualizadas.
			Si.

No permite el update y en el mensaje informa que se debe usar la silla 2 del salón 1, si se quiere realizar un update exitoso.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`--Este es el update problemático`
	`UPDATE rinde`
	`SET nro_salon= 1`
	`WHERE nro_estudiante = 2;`
		2 filas insertadas.
		
		Error, -20001, 'MISMO SALON NUEVA SILLA, DEBE INSERTAR EN EL SALON 1, SILLA 2 '
			Si.

No permite el update y en el mensaje informa que se debe usar la silla 1 del salón 2, si se quiere realizar un update exitoso.	`INSERT INTO rinde VALUES (1, 1, 'inst1', 1, to_date('2011/01/01', 'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2, 'inst1', 2, to_date('2011/01/01', 'yyyy/mm/dd'), 2);`
	
	`--Este es el update problemático`
	`UPDATE rinde`
	`SET nro_salon= 1`
	`WHERE nro_estudiante = 1;`
		2 filas insertadas.
		
		Error, -20001, 'NUEVO SALON MISMA SILLA, DEBE INSERTAR EN EL SALON 2, SILLA 1 '
			SI.

Casos de prueba	Sentencia SQL	Resultado Esperado	Resultado obtenido ok?

No permite el update y en el mensaje informara que se debe usar la silla 3 del salón 1, si se quiere realizar un update exitoso.	`--aumento la cantidad de sillas disponibles`
	`UPDATE SALON SET nro_silla_max = 3`
	`where nro_salon = 1 and`
	`nombre_institucion = 'inst1';`
	
	`INSERT INTO rinde VALUES (1, 1,`
	`'inst1', 1, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 2,`
	`'inst1', 1, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 2);`
	
	`INSERT INTO rinde VALUES (1, 3,`
	`'inst1', 2, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 1);`
	
	`INSERT INTO rinde VALUES (1, 4,`
	`'inst1', 2, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 2);`
	
	`--Este es el update problemático.`
	`UPDATE rinde`
	`SET nro_salon= 2`
	`WHERE nro_estudiante = 3;`
		1 fila actualizada.
		
		4 filas insertadas.
		
		Error, -20001 'CUALQUIER SALON, CUALQUIER SILLA, DEBE INSERTAR EN EL SALON 1, SILLA 3'
			Si.

No permite el update y en el mensaje informa que para esa instancia de examen no hay mas lugares.	`INSERT INTO rinde VALUES (1, 1,`
	`'inst1', 1, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 1);`
	`INSERT INTO rinde VALUES (1, 2,`
	`'inst1', 1, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 2);`
	`INSERT INTO rinde VALUES (1, 3,`
	`'inst1', 2, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 1);`
	`INSERT INTO rinde VALUES (1, 4,`
	`'inst1', 2, to_date('2011/01/01',`
	`'yyyy/mm/dd'), 2);`
	
	`--Este es el update problemático`
	`UPDATE rinde`
	`SET nro_salon= 2`
	`WHERE nro_estudiante = 3;`
		4 filas insertadas.
		
		Error -20001, 'NO PUEDES DAR EL EXAMEN'.
			SI.