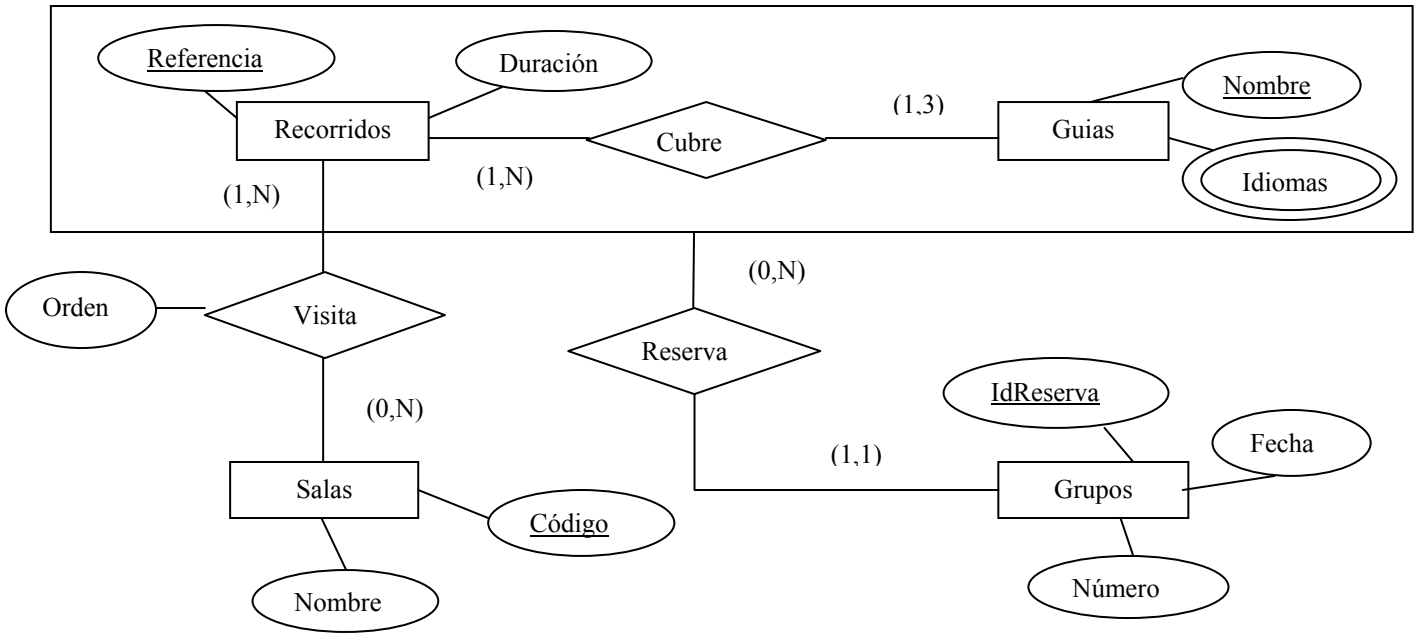
# Examen de Bases de datos
# Grado de Ingeniería en Informática, Febrero, 2015

1) (3 puntos) El museo de la ciudad necesita gestionar las visitas guiadas de grupos y para ello se debe implementar una base de datos relacional que cumpla los siguientes requisitos:
   Los grupos pueden solicitar reservas de los recorridos que ofrece el museo de la ciudad. Cada grupo de visitantes que desee una vista guiada indicará el recorrido, la fecha y el número de personas del grupo. El museo les asignará un identificador de reserva y uno de los guías que pueden cubrir el recorrido solicitado. De cada guía se conoce su nombre y los idiomas que habla. Un recorrido, identificado mediante una referencia y con una duración determinada, consiste en una visita a un subconjunto de salas del museo en un orden predeterminado. Cada sala tiene asignados un código único y un nombre. Cada guía cubre al menos un recorrido y como mucho tres. Todos los recorridos tienen asignado al menos un guía.
   a. (1,5 puntos) Se pide diseñar el esquema conceptual con el modelo entidad-relación especificando las claves primarias y las restricciones bien de participación mínimo-máximo o de cardinalidad.

**Solución:**




   b. (1 punto) Traduce dicho esquema al relacional e incluye las restricciones de integridad referencial.

**Solución:**

*Tipos de entidades:*
Recorridos(Referencia, Duración)
Guías (Nombre)
Idiomas (Nombre, Idioma)
$\prod_{Nombre}(Idiomas) \subseteq \prod_{Nombre}(Guías)$
Salas(Código, Nombre)
Grupos(IdReserva, Fecha,Número,Nombre,Referencia) Se ha incluido aquí la relación Reserva
$\prod_{Nombre, Referencia}(Grupos) \subseteq \prod_{Nombre, Referencia}(Cubre)$ (El tipo de relación Cubre está a continuación)

*Tipos de relaciones:*
Visita(Referencia, Código, Orden)
$\prod_{Referencia}(Visita) \subseteq \prod_{Referencia}(Recorridos)$
$\prod_{Codigo}(Visita) \subseteq \prod_{Codigo}(Salas)$
Cubre(Nombre,Referencia)
$\prod_{Nombre}(Cubre) \subseteq \prod_{Nombre}(Guías)$
$\prod_{Referencia}(Cubre) \subseteq \prod_{Referencia}(Recorridos)$

c. (0,5 puntos) Indica las restricciones no reflejadas en el esquema relacional.

**Solución:**

- Cada guía solo cubre hasta 3 recorridos
- Restricciones de participación total en relaciones con cardinalidad N a N:
  - Cada recorrido contiene al menos una sala
  - Cada recorrido es cubierto al menos por un guía.

2) (2 puntos) Dadas las relaciones:
Cruceros(Referencia, Compañía, Precio)
Puertos(Código, Nombre, País)
Recorridos(Referencia, Código, NúmParada)
Se pide escribir las siguientes consultas con **álgebra relacional**:

a. (0,25 puntos) Mostrar los cruceros que hacen escala en algún puerto de Italia. Esquema: (Referencia)

**Solución:**

$\prod_{Referencia}$ (Recorridos ⋈ ($\sigma_{Pais='Italia'}$(Puertos)))

b. (0,25 puntos) Mostrar los cruceros que hacen escala en todos los puertos de España. Esquema: (Referencia)

**Solución:**

$\prod_{Referencia,\ Código}$ (Recorridos) ÷ $\prod_{Código}$ ($\sigma_{País='España'}$ (Puertos))

Y con **SQL**:

c. (0,5 puntos) Mostrar todas las compañías con más de 3 cruceros con precio superior a 1.000 €. Esquema: (Compañía)

**Solución:**

```
SELECT Compañía
FROM Cruceros
WHERE Precio > 1000
GROUP BY Compañía
HAVING COUNT (*) > 3;
```

d. (0,5 puntos) Mostar los cruceros más baratos que zarpan desde Barcelona. Esquema: (Referencia)

**Solución:**

```
SELECT Referencia
FROM  Cruceros NATURAL JOIN Recorridos NATURAL JOIN Puertos
WHERE NúmParada = 1 AND Nombre = 'Barcelona'
AND Precio = (SELECT MIN(Precio)
              FROM Cruceros NATURAL JOIN Recorridos NATURAL JOIN Puertos
              WHERE NúmParada = 1 AND Nombre = 'Barcelona');
```

e. (0,5 puntos) Mostrar los cruceros que no hacen escala en ningun puerto de Grecia. Esquema: (Referencia)

**Solución:**

```
SELECT Referencia FROM Cruceros C
WHERE NOT EXISTS (SELECT *
                  FROM Recorridos NATURAL JOIN Puertos
                  WHERE Referencia = C.Referencia and País='Grecia');
```

3) (2,5 puntos) Dado el siguiente esquema:
Guías(Nombre, Referencia)
Grupos(IdReserva, Fecha, Nombre, Referencia)
Recuento(Referencia,Visitas)

a. (1,5 puntos) Escribe un procedimiento almacenado que recibe como parámetros de entrada la referencia de un recorrido (que se asume que existe) y una fecha. El procedimiento asignará el primer guía libre por orden alfabético (asumimos que no hay guías con el mismo nombre) y dará de alta al grupo (en la tabla Grupos) con un identificador de reserva que se obtendrá de una secuencia denominada seq_grupos. Se debe imprimir un mensaje indicando el guía asignado y declarar una excepción que se lance para dar un mensaje si no hay guías libres.

**Solución:**

```
CREATE TABLE Guías(Nombre VARCHAR(20), Referencia VARCHAR(20));
CREATE TABLE Grupos(IdReserva VARCHAR(20), Fecha DATE, Nombre VARCHAR(20),
Referencia VARCHAR(20));
CREATE TABLE Recuento(Referencia VARCHAR(20), Visitas NUMBER(4,0));

SET SERVEROUTPUT ON SIZE 100000;

CREATE SEQUENCE seq_grupos;

CREATE OR REPLACE PROCEDURE AsignaGuía (p_numRef IN VARCHAR, p_día DATE) AS

  v_guía VARCHAR(20);
  exc_sin_guía EXCEPTION;

BEGIN

  SELECT MIN(Nombre) INTO v_guía
  FROM  Guías G
  WHERE Referencia = p_numRef
  AND NOT EXISTS
      (SELECT * FROM Grupos WHERE Nombre=G.Nombre AND Fecha=p_día);

  IF v_guía IS NULL THEN
    RAISE exc_sin_guía;
  END IF;

  INSERT INTO Grupos VALUES (seq_grupos.NEXTVAL, v_guía, p_numRef, p_día);
```

```
    DBMS_OUTPUT.PUT_LINE('Se ha asignado el guía ' || v_guía);

    EXCEPTION
    WHEN exc_sin_guía THEN
        DBMS_OUTPUT.PUT_LINE('No hay guías disponibles');
END;


EXECUTE AsignaGuía('Número',TO_DATE('01/01/2015','dd/mm/yyyy'));
```

b. (1 punto) Para evitar mantener todas las reservas de grupos en la base de datos se utiliza la tabla Recuento, que incluye el número total de visitas por cada recorrido en el campo Visitas. Crea un disparador (trigger) que mantenga la tabla Recuento al borrar en la tabla Grupos.

**Solución:**

```
CREATE OR REPLACE TRIGGER tri_ActualizaRecuento
BEFORE DELETE ON Grupos
FOR EACH ROW
DECLARE
    v_cont NUMBER;

BEGIN
    SELECT COUNT(*) into v_cont FROM Recuento WHERE Referencia=:OLD.Referencia;

    IF v_cont=0 THEN
      INSERT INTO Recuento VALUES (:OLD.Referencia, 1);
    ELSE
      UPDATE Recuento SET Visitas = Visitas + 1
      WHERE Referencia = :OLD.Referencia;
    END IF;

END;
```

4) (1 punto) Indica las diferentes formas de finalizar una transacción.

**Solución:**

- COMMIT: Se *comprometen* todas las modificaciones sobre la BD realizadas desde el inicio de la transacción. El compromiso sucede en las siguientes situaciones:
  - Ejecución explícita de la sentencia COMMIT.
  - Ejecución de una instrucción DDL. COMMIT automático antes de su ejecución.
  - Automático al finalizar la sesión del usuario.
  - Automático después de cada instrucción de modificación (con SET AUTOCOMMIT ON en Oracle).

- ROLLBACK: Se *retroceden* todas las modificaciones sobre la BD realizadas desde el inicio de la transacción. El retroceso sucede en las siguientes situaciones:
  - Ejecución explícita de la sentencia ROLLBACK.
  - Ejecución implícita de la sentencia ROLLBACK cuando se produce un error durante la ejecución de un proceso.

5) (1,5 puntos) Dado el esquema R(A,B,C,D) y el conjunto de dependencias funcionales $\{ BD \rightarrow C, AD \rightarrow C, CD \rightarrow B, BC \rightarrow A \}$ se pide:
   a. (1 punto) Determinar si esta en Forma Normal de Boyce-Codd.

**Solución:**

Hay que comprobar que los antecedentes de las dependencias funcionales sean superclave:
$BD^+$ = ABCD (superclave)
$AD^+$ = ABCD (superclave)
$CD^+$ = ABCD (superclave)
$BC^+$ = BCA. BC no es superclave y por tanto no está en Forma Normal de Boyce-Codd.

b. (0,5 puntos) Determinar las claves candidatas.

**Solución:**

- De 1 atributo:
  No hay claves candidatas de un atributo porque ninguna D.F. tiene un antecedente con un solo atributo.
- De 2 atributos:
  Las superclaves que se han visto en el apartado anterior. No puede haber más porque solo hay D.F. con 2 atributos en el antecedente.
- No puede haber claves candidatas con un número mayor de atributos por el mismo motivo.