Proyecto agencia de viajes

Ángel López Manríquez

9 de diciembre de 2019

Análisis de requerimientos

Modelo entidad relacion normalizado

Modelo relacional

Restricciones de integridad

Vistas

Disparadores

Análisis de requerimientos

1. Se les ofrecerá servicio tanto a estudiantes como a personas ajenas al instituto. Los estudiantes tendrán mas beneficios frente al resto. Los viajes serán en grupo.
2. Se recabara información de las personas, tanto para los miembros de la agencia como para los clientes.
    2.1 **DNI** Como identificación les pediremos su documento nacional de identidad o bien el pasaporte para el caso de extranjeros.
    2.2 **correo** Se les enviara información por correo, informándoles de los futuros viajes así como comprobantes de pago.
    2.3 **Numero de celular** Para fomentar la comunicación grupal, se creara un grupo de whatsapp, sera muy útil en lo largo del camino.
    2.4 **Dirección**
    2.5 **Nacionalidad**
    2.6 **Edad**
3. Un estudiante o bien una persona externa de el instituto tiene

50 % del coste total. No se recibirán cantidades inferiores en el primer pago si es que desea pagar por plazos.

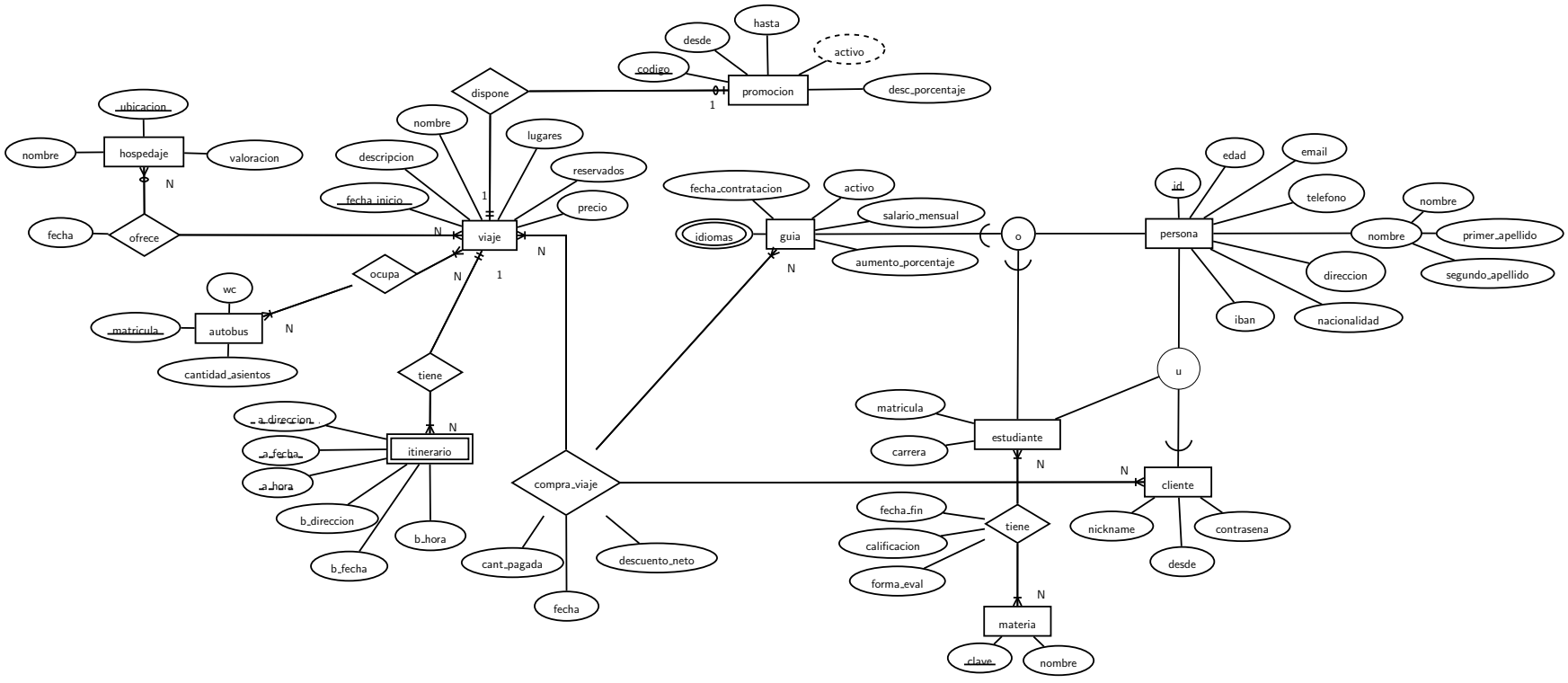
4. Con el fin de motivar el rendimiento escolar, se ofrecerá descuento para todo estudiante, el cual sera directamente proporcional a la nota que posea.
   - 0 % para todo estudiante con nota inferior a 5.
   - 5 % para todo estudiante con nota inferior a 6 y mayor a 5.
   - 10 % para todo estudiante con nota inferior a 7 y mayor a 6.
   - 15 % para todo estudiante con nota inferior a 8 y mayor a 7.
   - 20 % para todo estudiante con nota inferior a 9 y mayor a 8.
   - 25 % para todo estudiante con nota inferior o igual a 10.
5. Como toda agencia de viajes, se contara con un guía.
6. Cualquier persona con una edad mayor a los 18 años puede aspirar a trabajar con nosotros.
7. Un cliente tendrá una cuenta, con su nickname y su

8. El sueldo base mensual para un guía sera de 500 euros mensuales.
9. Con el fin de ayudar a la comunidad estudiantil con recursos escasos, toda persona estudiante que trabaje como agente de viajes dispondrá de un aumento del al menos el 10 % con respecto a una persona ajena al instituto. Si el estudiante habla mas de dos idiomas se agregara un 10 %.
10. Se llevara un registro de las personas que han comprado un viaje, así como si aplicaron algún descuento.
11. Se podrá disponer de promociones si se tiene un código de oferta, así como su nombre de promoción. Las promociones están delimitadas por fechas y ofrecen descuentos.
12. Para cada viaje tendrá determinado costo y la cantidad pagada y al menos un lugar a visitar, claro.
13. Cada viaje sera coordinado por varios guías, cada guia para un

14. No puede haber otro viaje si ya hay uno en curso.
15. Se interesa saber la cantidad de descuento hecha por un cliente en un viaje con un guía en concreto.
16. Un viaje puede o no contar con algún descuento, este tendrá un código asociado que lo hace único, así como las fechas donde el descuento aplicara.
17. Para transportar a los clientes se usara autobús.
18. Cada viaje tendrá al menos un autobús. Cada autobús tendrá asientos indicando su propietario para un viaje en concreto.
19. Cada viaje tendrá su itinerario, indicando la hora, fecha y punto de reunión para la salida, así como su llegada, si se conoce.
20. Interesa saber las materias cursadas por los estudiantes, asi como su nota.

Modelo entidad relacion normalizado

El siguiente model ha sido normalizado hasta la tercera forma normal. Esta compuesto de entidades debiles, fuertes, atributos multivalor, relacion ternaria, herencia y clasificacion.



Modelo relacional

Se presenta la implementacion en *Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production*.

```
CREATE TABLE persona (
    id VARCHAR2(32) NOT NULL,
    email VARCHAR2(64),
    nombre VARCHAR2(32),
    edad NUMBER,
    primer_apellido VARCHAR2(32),
    segundo_apellido VARCHAR2(32),
    direccion VARCHAR2(128),
    iban VARCHAR2(32),
    telefono NUMBER,
    PRIMARY KEY (id)
);
```

```
1  CREATE TABLE guia (
2      desde DATE DEFAULT SYSDATE,
3      salario_mensual NUMBER DEFAULT 500,
4      persona_id VARCHAR2(32),
5      activo CHAR(1) DEFAULT '1',
6      aumento_porcentaje NUMBER DEFAULT 0,
7      CONSTRAINT guia_pk PRIMARY KEY (persona_id),
8      CONSTRAINT guia_fk FOREIGN KEY (persona_id)
9          REFERENCES persona(id) ON DELETE CASCADE
10 );
```

```
1  CREATE TABLE guia_idioma (
2      guia_id VARCHAR(32),
3      idioma VARCHAR(16),
4      CONSTRAINT guia_idioma_pk PRIMARY KEY (guia_id,
       ↪  idioma),
5      CONSTRAINT guia_idioma_fk FOREIGN KEY (guia_id)
6          REFERENCES guia(persona_id) ON DELETE CASCADE
7  );
```

```
1  CREATE TABLE estudiante (
2      persona_id VARCHAR2(32),
3      matricula VARCHAR2(16),
4      carrera VARCHAR2(32),
5      CONSTRAINT estudiante_pk PRIMARY KEY (persona_id),
6      CONSTRAINT estudiante_fk FOREIGN KEY (persona_id)
7          REFERENCES persona(id) ON DELETE CASCADE
8  );
```

```
1  CREATE TABLE materia (
2      clave VARCHAR2(8) NOT NULL,
3      nombre VARCHAR2(64) NOT NULL,
4      CONSTRAINT materia_pk PRIMARY KEY(clave)
5  );
```

```
1  CREATE TABLE estudiante_materia (
2      estudiante_id VARCHAR2(32) NOT NULL,
3      materia_id VARCHAR2(8) NOT NULL,
4      calificacion NUMBER NOT NULL,
5      fecha_fin DATE NOT NULL,
6      forma_eval VARCHAR2(16) NOT NULL,
7      CONSTRAINT estudiante_materia_pk PRIMARY
       ↪  KEY(estudiante_id, materia_id)
8  );
```

```
1  CREATE TABLE cliente (
2      id VARCHAR2(16) NOT NULL,
3      nickname VARCHAR2(32),
4      contrasena VARCHAR2(64),
5      desde DATE,
6      CONSTRAINT cliente_pk PRIMARY KEY (id),
7      CONSTRAINT cliente_fk FOREIGN KEY(id)
8          REFERENCES persona(id) ON DELETE CASCADE
9  );
```

```
1  CREATE TABLE viaje (
2      fecha_inicio DATE NOT NULL,
3      descripcion VARCHAR2(256),
4      nombre VARCHAR2(64),
5      precio NUMBER,
6      reservados NUMBER DEFAULT 0,
7      lugares NUMBER NOT NULL,
8      CONSTRAINT viaje_pk PRIMARY KEY(fecha_inicio)
9  );
```

```
1  CREATE TABLE promocion (
2      codigo VARCHAR(32) NOT NULL,
3      viaje_id DATE NOT NULL,
4      desde DATE DEFAULT SYSDATE,
5      hasta DATE,
6      desc_porcentaje NUMBER,
7      CONSTRAINT promocion_pk PRIMARY KEY (codigo),
8      CONSTRAINT promocion_fk FOREIGN KEY (viaje_id)
9          REFERENCES viaje(fecha_inicio) ON DELETE CASCADE
10 );
```

```
CREATE TABLE compra_viaje (
    viaje_id DATE NOT NULL,
    guia_id VARCHAR2(32) NOT NULL,
    cliente_id VARCHAR2(32) NOT NULL,
    cant_pagada NUMBER,
    fecha DATE DEFAULT SYSDATE,
    descuento_neto NUMBER DEFAULT 0,
    CONSTRAINT compra_viaje_pk
        PRIMARY KEY (viaje_id, guia_id, cliente_id),
    CONSTRAINT cv_viaje_fk FOREIGN KEY (viaje_id)
        REFERENCES viaje(fecha_inicio) ON DELETE CASCADE,
    CONSTRAINT cv_guia_fk FOREIGN KEY (guia_id)
        REFERENCES guia(persona_id) ON DELETE CASCADE,
    CONSTRAINT cv_cliente_fk FOREIGN KEY (cliente_id)
        REFERENCES cliente(id) ON DELETE CASCADE
);
```

```
1  CREATE TABLE itinerario (
2      viaje_id DATE NOT NULL,
3      a_direccion VARCHAR(256) NOT NULL,
4      a_fecha DATE NOT NULL,
5      a_hora VARCHAR2(8) DEFAULT '09:00',
6      b_direccion VARCHAR(256) ,
7      b_fecha DATE ,
8      b_hora VARCHAR2(8),
9      CONSTRAINT itinerario_pk
10         PRIMARY KEY(viaje_id, a_direccion, a_fecha,
           ↪  a_hora),
11     CONSTRAINT itinerario_fk
12         FOREIGN KEY(viaje_id)
13         REFERENCES viaje(fecha_inicio) ON DELETE CASCADE
14 );
```

```
1  CREATE TABLE autobus (
2      matricula VARCHAR2(16) NOT NULL,
3      wc NUMBER,
4      cantidad_asientos NUMBER,
5      CONSTRAINT autobus_pk PRIMARY KEY(matricula)
6  );
```

```
1  CREATE TABLE viaje_autobus (
2      matricula VARCHAR2(16) NOT NULL,
3      viaje_id DATE NOT NULL,
4      CONSTRAINT viaje_autobus_pk PRIMARY KEY(matricula,
       ↪  viaje_id),
5      CONSTRAINT viaje_autobus_viaje_fk FOREIGN
       ↪  KEY(viaje_id)
6          REFERENCES viaje(fecha_inicio) ON DELETE CASCADE,
7      CONSTRAINT viaje_autobus_autobus_fk FOREIGN
       ↪  KEY(matricula)
8          REFERENCES autobus(matricula) ON DELETE CASCADE
9  );
```

```
1  CREATE TABLE hospedaje (
2      ubicacion VARCHAR2(128) NOT NULL,
3      nombre VARCHAR2(128) NOT NULL,
4      valoracion NUMBER,
5      CONSTRAINT hospedaje_pk PRIMARY KEY (ubicacion)
6  );
```

```
1  CREATE TABLE hospedaje_viaje (
2      hospedaje_id VARCHAR2(128) NOT NULL,
3      viaje_id DATE NOT NULL,
4      fecha DATE DEFAULT SYSDATE,
5      CONSTRAINT hospedaje_viaje_hospedaje_fk FOREIGN
       ↪  KEY(hospedaje_id)
6          REFERENCES hospedaje(ubicacion) ON DELETE CASCADE,
7      CONSTRAINT hospedaje_viaje_viaje_fk FOREIGN
       ↪  KEY(viaje_id)
8          REFERENCES viaje(fecha_inicio) ON DELETE CASCADE,
9      CONSTRAINT hospedaje_viaje_pk PRIMARY
       ↪  KEY(hospedaje_id, viaje_id)
10 );
```

Restricciones de integridad

```sql
-- Restriciones de integridad
-- Email de una persona validado por expresion regular
ALTER TABLE persona
    ADD CONSTRAINT chk_persona_email
    CHECK (REGEXP_LIKE
     ↪  (email,'^[A-Za-z]+[A-Za-z0-9.]+@[A-Za-z0-9.-]+\.[A-Za-z]
```

```sql
-- Las fechas de la promocion consistentes, asi como el
↪  rango del porcentaje
ALTER TABLE promocion ADD CONSTRAINT
↪  chk_promocion_fechas_descuento
    CHECK (desde <= hasta AND desc_porcentaje BETWEEN 0
     ↪  AND 1);
```

```sql
-- la calificacion posible esta entre 0 y 10
ALTER TABLE estudiante_materia
    ADD CONSTRAINT chk_es_mat_cal
    CHECK (calificacion BETWEEN 0 AND 10);
```

```sql
-- a lo mas habria dos banos en un autobus
ALTER TABLE autobus
    ADD CONSTRAINT chk_autobus_wc
    CHECK (wc BETWEEN 0 AND 2);
```

```
1  -- No puede haber mas lugares reservados del total de
   ↪  plazas
2  ALTER TABLE viaje
3      ADD CONSTRAINT chk_viaje_reservados
4      CHECK (reservados <= lugares);
```

Vistas

```sql
-- Vistas
-- obtener el fragmento minitermino mixto de los estudiantes con su
-- clave, nombre y nota media
CREATE OR REPLACE VIEW v_estudiante_nota_media AS
    SELECT
        p.id,
        p.nombre, p.primer_apellido, p.segundo_apellido,
        AVG(em.calificacion) AS nota_media
    FROM persona p
        JOIN estudiante_materia em ON p.id = em.estudiante_id
        JOIN materia m  ON em.materia_id = m.clave
    GROUP BY p.id, p.nombre, p.primer_apellido, p.segundo_apellido;
```

```sql
-- visualizar la cantidad invertida por los clientes en el viaje con su nombre
CREATE OR REPLACE VIEW v_cliente_inversion AS
    SELECT
        p.id, c.nickname, SUM(cv.cant_pagada) AS total
    FROM persona p
        JOIN cliente c ON p.id = c.id
        JOIN compra_viaje cv ON c.id = cv.cliente_id
    GROUP BY p.id, c.nickname
    ORDER BY total DESC;
```

```sql
-- clientes que no son estudiantes
CREATE OR REPLACE VIEW v_cliente_no_estudiante AS
    SELECT p.id, p.nombre FROM persona p
        JOIN cliente c ON p.id = c.id
    AND NOT p.id IN (SELECT persona_id FROM estudiante);
```

Disparadores

```
1  -- Definicion de disparadores
2  -- Consistencia en el aumento del guia
3  CREATE OR REPLACE TRIGGER t_guia_aumento_porcentaje
4    BEFORE INSERT OR UPDATE OF activo ON guia
5    FOR EACH ROW
6    DECLARE
7      student_too NUMBER DEFAULT 0;
8      previous_percentage_applied NUMBER DEFAULT 0;
9  BEGIN
10   IF INSERTING THEN
11     SELECT count(*) INTO student_too
12       FROM estudiante
13       WHERE persona_id = :new.persona_id;
14     IF student_too > 0 AND :new.aumento_porcentaje = 0
       ↪  THEN
15       :new.aumento_porcentaje := 0.1;
16     END IF;
17   ELSE
18       IF :new.activo = 0 AND :old.aumento_porcentaje > 0
         ↪  THEN
19         :new.aumento_porcentaje := :old.aumento_porcentaje
                 0.1;
```

```
20        END IF;
21      END IF;
22  END;
23  /
```

```
1  -- al detectar que se tiene al menos 3 idiomas, agregar un
   ↪  aumento de sueldo
2  CREATE OR REPLACE TRIGGER t_idioma_incremento_sueldo
3    BEFORE INSERT ON guia_idioma
4    FOR EACH ROW
5    DECLARE
6        idioma_count NUMBER ;
7  BEGIN
8    SELECT count(*) INTO idioma_count FROM guia g
9      JOIN guia_idioma gi ON g.persona_id = gi.guia_id
10     WHERE persona_id = :new.guia_id;
11   IF idioma_count = 2 THEN
12     UPDATE guia
13       SET aumento_porcentaje = aumento_porcentaje + 0.1
14       WHERE persona_id = :new.guia_id;
15   END IF;
16 END;
17 /
```

```
1  -- Los guias tendran al menos 18 anios de edad
2  CREATE OR REPLACE TRIGGER t_guia_edad
3    BEFORE INSERT ON guia
4    FOR EACH ROW
5    DECLARE
6      v_edad NUMBER := 0;
7  BEGIN
8    SELECT edad INTO v_edad FROM persona WHERE id =
     ↪  :new.persona_id;
9    IF v_edad < 18 THEN
10     RAISE_APPLICATION_ERROR(-20000, 'No tiene la edad
       ↪  suficiente (18).');
11   END IF;
12 END;
13 /
```

```
1  -- Hacer la reserva de la compra con haber pagado al menos
   ↪  la mitad y se
2  -- aplican los descuentos
3  CREATE OR REPLACE TRIGGER t_compra_contador
4    BEFORE INSERT ON compra_viaje
5    FOR EACH ROW
6    DECLARE
7      v_lugares NUMBER;
8      v_precio NUMBER;
9      v_descuento_neto NUMBER := 0;
10     v_nota_media NUMBER := 5;
11     v_student_too NUMBER := 0;
12 BEGIN
13   SELECT lugares, precio
14     INTO v_lugares, v_precio
15     FROM viaje WHERE fecha_inicio = :new.viaje_id;
16   SELECT count(*)
17     INTO v_student_too
18     FROM estudiante e
19     JOIN cliente c ON e.persona_id = c.id
20     WHERE c.id = :new.cliente_id;
21   IF v_student_too = 1 THEN
```

```
22      SELECT FLOOR(nota_media) INTO v_nota_media
23        FROM v_estudiante_nota_media
24        WHERE id = :new.cliente_id;
25      IF v_nota_media >= 6 THEN  -- o es estudiante,
        ↪  aplicamos descuento
26        v_descuento_neto := v_precio * 0.05 * (v_nota_media
          ↪  - 5);
27        :new.descuento_neto := v_descuento_neto;
28      END IF;
29    END IF;
30    IF :new.cant_pagada < (v_precio - v_descuento_neto) *
      ↪  0.5 THEN
31      RAISE_APPLICATION_ERROR(-20000, 'Se debe de pagar al
        ↪  menos la mitad del viaje para reservar.');
32    END IF;
33    UPDATE viaje SET reservados = reservados + 1 WHERE
      ↪  fecha_inicio = :new.viaje_id;
34  END;
35  /
```