**UNIVERSIDAD MAYOR REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA**

**FACULTAD DE CIENCIAS Y TECNOLOGÍA**

**CARRERA DE INGENIERÍA EN CIENCIAS DE LA COMPUTACION**



**Examen Final – Base de datos III**

Universitario: Peñafiel Picha Erick Martin

Carrera: Ingeniería en ciencias de la computación

Docente: Espinoza Rodriguez Edgar Tito

Fecha: 28/06/2024

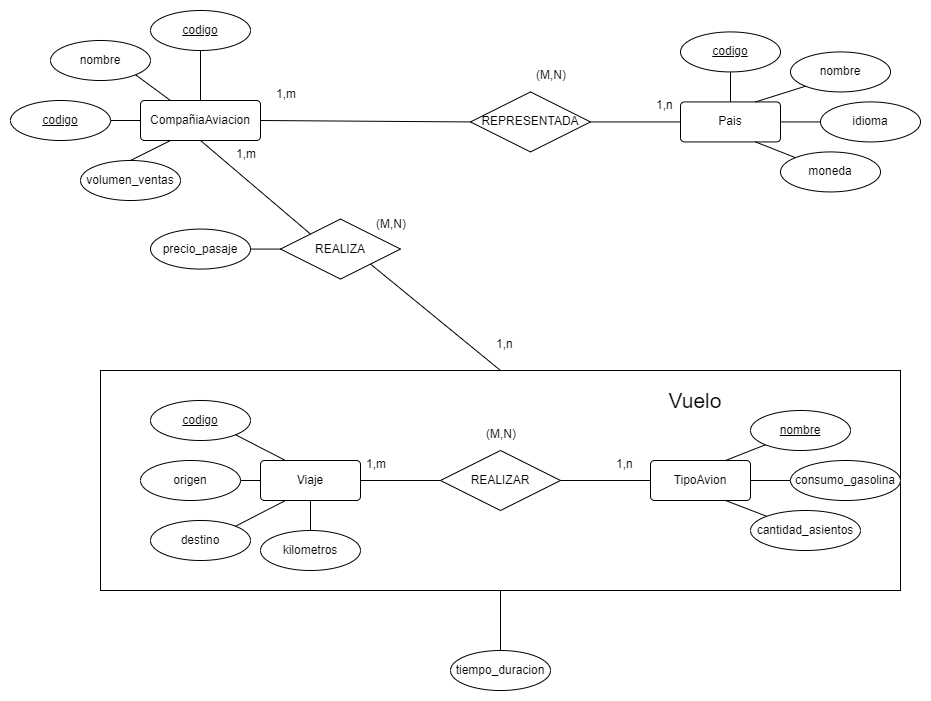
Sucre – Bolivia

**Examen Final**

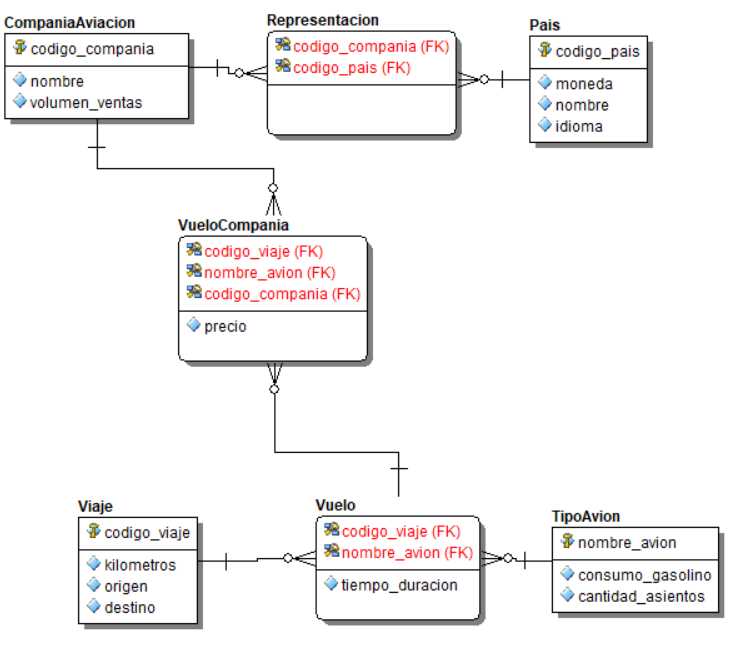
**Ejercicio**

NABOL Bolivia cuenta con información sobre las diferentes compañías de aviación que existen en el mundo. De cada compañía se conoce su nombre (BOA, Ecojet, etc.), su volumen de ventas y un código que la identifica. Cada compañía puede estar representada en diferentes países y en un país pueden estar representadas diversas compañías. De cada país se conoce su código, nombre, idioma y tipo de moneda. También se sabe que por vía aérea se realizan diferentes viajes. Cada viaje posee un código, un lugar de origen, un destino y una cantidad de kilómetros a recorrer. Además, se conoce que existen distintos tipos de aviones. Cada tipo de avión se identifica por un nombre (DC100, IL700, etc.), y posee un consumo de gasolina por kilómetro y una cierta cantidad de asientos. Un mismo viaje puede ser realizado por distintos tipos de aviones y un tipo de avión puede ser utilizado en diversos viajes. Un viaje realizado por un tipo de avión constituye un vuelo y para cada vuelo se conoce su tiempo de duración. Una compañía de aviación realiza muchos vuelos y un mismo vuelo puede ser cubierto por diversas compañías. Para cada vuelo que ofrece una compañía se conoce el precio del pasaje

**Diagrama Entidad Relacion**

****

**Modelo Logico**

****

**Diccionario de Datos**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabla** | **Campo** | **Obligatorio** | **Restricción de Dominio Adicional** | **Calculado** | **Descripción Cálculo** | **Valor por Defecto** |
| Pais | codigo | X |  |  |  |  |
|  | nombre | X |  |  |  |  |
|  | idioma | X |  |  |  |  |
|  | moneda | X |  |  |  |  |
| CompaniaAviacion | codigo | X |  |  |  |  |
|  | nombre | X |  |  |  |  |
|  | volumen\_ventas |  | volumen\_ventas >= 0 |  |  |  |
| Representacion | codigo\_compania | X |  |  |  |  |
|  | codigo\_pais | X |  |  |  |  |
| Viaje | codigo | X |  |  |  |  |
|  | origen | X |  |  |  |  |
|  | destino | X |  |  |  |  |
|  | kilometros |  | kilometros > 0 |  |  |  |
| TipoAvion | nombre | X |  |  |  |  |
|  | consumo\_gasolina |  | consumo\_gasolina > 0 |  |  |  |
|  | cantidad\_asientos |  | cantidad\_asientos > 0 |  |  |  |
| Vuelo | codigo\_viaje | X |  |  |  |  |
|  | nombre\_avion | X |  |  |  |  |
|  | duracion |  | duracion > 0 |  |  |  |
|  | precio |  | precio >= 0 |  |  |  |
| VueloCompania | codigo\_viaje | X |  |  |  |  |
|  | nombre\_avion | X |  |  |  |  |
|  | codigo\_compania | X |  |  |  |  |

**Diseño Fisico**

**Analizar Transacciones**



**Espacio en Disco**



|  |  |  |
| --- | --- | --- |
| **Espacio tabular** | **Bytes** | **MB** |
| Espacio tablas | 4,353,440 | 4.151 |
| Espacio índices | 1,568,500 | 1,568,500 |
| **Subtotal** | 5,921,940 | 5,793 |
| Espacio adicional (20%) | 1,741,376 | 1.159 |
| **TOTAL** | 10,448,256 | 6.952 |

**Codigo SQL**

CREATE TABLE Pais (

codigo VARCHAR2(10) PRIMARY KEY,

nombre VARCHAR2(100) NOT NULL,

idioma VARCHAR2(50) NOT NULL,

moneda VARCHAR2(50) NOT NULL

);

CREATE INDEX idx\_nombre\_pais ON Pais(nombre);

CREATE TABLE CompaniaAviacion (

codigo VARCHAR2(10) PRIMARY KEY,

nombre VARCHAR2(100) NOT NULL,

volumen\_ventas NUMBER CHECK (volumen\_ventas >= 0)

);

CREATE INDEX idx\_nombre\_compania ON CompaniaAviacion(nombre);

CREATE TABLE Representacion (

codigo\_compania VARCHAR2(10),

codigo\_pais VARCHAR2(10),

PRIMARY KEY (codigo\_compania, codigo\_pais),

FOREIGN KEY (codigo\_compania) REFERENCES CompaniaAviacion(codigo)

ON DELETE CASCADE,

FOREIGN KEY (codigo\_pais) REFERENCES Pais(codigo)

ON DELETE CASCADE

);

CREATE INDEX idx\_codigo\_compania ON Representacion(codigo\_compania);

CREATE INDEX idx\_codigo\_pais ON Representacion(codigo\_pais);

CREATE TABLE Viaje (

codigo VARCHAR2(10) PRIMARY KEY,

origen VARCHAR2(100) NOT NULL,

destino VARCHAR2(100) NOT NULL,

kilometros NUMBER CHECK (kilometros > 0)

);

CREATE INDEX idx\_origen ON Viaje(origen);

CREATE INDEX idx\_destino ON Viaje(destino);

CREATE TABLE TipoAvion (

nombre VARCHAR2(50) PRIMARY KEY,

consumo\_gasolina NUMBER CHECK (consumo\_gasolina > 0),

cantidad\_asientos NUMBER CHECK (cantidad\_asientos > 0)

);

CREATE TABLE Vuelo (

codigo\_viaje VARCHAR2(10),

nombre\_avion VARCHAR2(50),

duracion NUMBER CHECK (duracion > 0),

precio NUMBER CHECK (precio >= 0),

PRIMARY KEY (codigo\_viaje, nombre\_avion),

FOREIGN KEY (codigo\_viaje) REFERENCES Viaje(codigo)

ON DELETE CASCADE,

FOREIGN KEY (nombre\_avion) REFERENCES TipoAvion(nombre)

ON DELETE CASCADE

);

CREATE INDEX idx\_duracion ON Vuelo(duracion);

CREATE INDEX idx\_precio ON Vuelo(precio);

CREATE TABLE VueloCompania (

codigo\_viaje VARCHAR2(10),

nombre\_avion VARCHAR2(50),

codigo\_compania VARCHAR2(10),

PRIMARY KEY (codigo\_viaje, nombre\_avion, codigo\_compania),

FOREIGN KEY (codigo\_viaje, nombre\_avion) REFERENCES Vuelo(codigo\_viaje, nombre\_avion)

ON DELETE CASCADE,

FOREIGN KEY (codigo\_compania) REFERENCES CompaniaAviacion(codigo)

ON DELETE CASCADE

);

-- Crear el Paquete Especificación

CREATE OR REPLACE PACKAGE AviacionPkg IS

-- Funciones

FUNCTION obtener\_precio\_vuelo(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2

) RETURN NUMBER;

FUNCTION calcular\_consumo\_gasolina(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2

) RETURN NUMBER;

-- Procedimientos

PROCEDURE agregar\_compania\_aviacion(

p\_codigo IN VARCHAR2,

p\_nombre IN VARCHAR2,

p\_volumen\_ventas IN NUMBER

);

PROCEDURE registrar\_pais(

p\_codigo IN VARCHAR2,

p\_nombre IN VARCHAR2,

p\_idioma IN VARCHAR2,

p\_moneda IN VARCHAR2

);

PROCEDURE registrar\_representacion(

p\_codigo\_compania IN VARCHAR2,

p\_codigo\_pais IN VARCHAR2

);

PROCEDURE registrar\_viaje(

p\_codigo IN VARCHAR2,

p\_origen IN VARCHAR2,

p\_destino IN VARCHAR2,

p\_kilometros IN NUMBER

);

PROCEDURE registrar\_tipo\_avion(

p\_nombre IN VARCHAR2,

p\_consumo\_gasolina IN NUMBER,

p\_cantidad\_asientos IN NUMBER

);

PROCEDURE registrar\_vuelo(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2,

p\_duracion IN NUMBER,

p\_precio IN NUMBER

);

PROCEDURE registrar\_vuelo\_compania(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2,

p\_codigo\_compania IN VARCHAR2

);

END AviacionPkg;

/

-- Crear el Paquete Cuerpo

CREATE OR REPLACE PACKAGE BODY AviacionPkg IS

-- Funciones

FUNCTION obtener\_precio\_vuelo(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2

) RETURN NUMBER IS

v\_precio NUMBER;

BEGIN

SELECT precio INTO v\_precio

FROM Vuelo

WHERE codigo\_viaje = p\_codigo\_viaje

AND nombre\_avion = p\_nombre\_avion;

RETURN v\_precio;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END obtener\_precio\_vuelo;

FUNCTION calcular\_consumo\_gasolina(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2

) RETURN NUMBER IS

v\_kilometros NUMBER;

v\_consumo\_por\_km NUMBER;

v\_consumo\_total NUMBER;

BEGIN

SELECT v.kilometros, t.consumo\_gasolina

INTO v\_kilometros, v\_consumo\_por\_km

FROM Viaje v, TipoAvion t

WHERE v.codigo = p\_codigo\_viaje

AND t.nombre = p\_nombre\_avion;

v\_consumo\_total := v\_kilometros \* v\_consumo\_por\_km;

RETURN v\_consumo\_total;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END calcular\_consumo\_gasolina;

-- Procedimientos

PROCEDURE agregar\_compania\_aviacion(

p\_codigo IN VARCHAR2,

p\_nombre IN VARCHAR2,

p\_volumen\_ventas IN NUMBER

) IS

BEGIN

INSERT INTO CompaniaAviacion (codigo, nombre, volumen\_ventas)

VALUES (p\_codigo, p\_nombre, p\_volumen\_ventas);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20003, 'La compañía con este código ya existe.');

END agregar\_compania\_aviacion;

PROCEDURE registrar\_pais(

p\_codigo IN VARCHAR2,

p\_nombre IN VARCHAR2,

p\_idioma IN VARCHAR2,

p\_moneda IN VARCHAR2

) IS

BEGIN

INSERT INTO Pais (codigo, nombre, idioma, moneda)

VALUES (p\_codigo, p\_nombre, p\_idioma, p\_moneda);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20004, 'El país con este código ya existe.');

END registrar\_pais;

PROCEDURE registrar\_representacion(

p\_codigo\_compania IN VARCHAR2,

p\_codigo\_pais IN VARCHAR2

) IS

BEGIN

INSERT INTO Representacion (codigo\_compania, codigo\_pais)

VALUES (p\_codigo\_compania, p\_codigo\_pais);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20005, 'Esta representación ya existe.');

END registrar\_representacion;

PROCEDURE registrar\_viaje(

p\_codigo IN VARCHAR2,

p\_origen IN VARCHAR2,

p\_destino IN VARCHAR2,

p\_kilometros IN NUMBER

) IS

BEGIN

INSERT INTO Viaje (codigo, origen, destino, kilometros)

VALUES (p\_codigo, p\_origen, p\_destino, p\_kilometros);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20006, 'El viaje con este código ya existe.');

END registrar\_viaje;

PROCEDURE registrar\_tipo\_avion(

p\_nombre IN VARCHAR2,

p\_consumo\_gasolina IN NUMBER,

p\_cantidad\_asientos IN NUMBER

) IS

BEGIN

INSERT INTO TipoAvion (nombre, consumo\_gasolina, cantidad\_asientos)

VALUES (p\_nombre, p\_consumo\_gasolina, p\_cantidad\_asientos);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20007, 'El tipo de avión con este nombre ya existe.');

END registrar\_tipo\_avion;

PROCEDURE registrar\_vuelo(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2,

p\_duracion IN NUMBER,

p\_precio IN NUMBER

) IS

BEGIN

INSERT INTO Vuelo (codigo\_viaje, nombre\_avion, duracion, precio)

VALUES (p\_codigo\_viaje, p\_nombre\_avion, p\_duracion, p\_precio);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20008, 'El vuelo con este código de viaje y nombre de avión ya existe.');

END registrar\_vuelo;

PROCEDURE registrar\_vuelo\_compania(

p\_codigo\_viaje IN VARCHAR2,

p\_nombre\_avion IN VARCHAR2,

p\_codigo\_compania IN VARCHAR2

) IS

BEGIN

INSERT INTO VueloCompania (codigo\_viaje, nombre\_avion, codigo\_compania)

VALUES (p\_codigo\_viaje, p\_nombre\_avion, p\_codigo\_compania);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

RAISE\_APPLICATION\_ERROR(-20009, 'El vuelo por esta compañía ya existe.');

END registrar\_vuelo\_compania;

END AviacionPkg;