

Ejercicio de Evaluación Bases de Datos SQL: Creación de Una Base de Datos

Soheil Moattar M. Berenguer

Complutense University of Madrid



Abstract

El objetivo de este documento consiste en la creación de una base de datos relacional. El proceso consistirá de las diferentes fases que suelen llevarse a cabo en la creación de una BD-SQL como son el diseño entidad-relacion, modelo relacional y consultas posteriores. Para cada paso se han incluido graficos o *code snippets* de *MYSQL WORKBENCH*®.

1. Diseño Conceptual

1.1. Identificación de Entidades

Las diferentes *entidades* en este problema son:

1. Tienda Aplicaciones. Ejemplos: Apple App Store ,Google Play Store, App World, Market Place ,etc ...
2. Empresa desarrolladoras de apps. Ejemplos: Orangsoft, Algeworks, etc ...
3. Empleado. Estos empleados trabajan desarrollando apps en las empresas desarrolladoras de apps mencionadas.
4. Apps. Ejemplos: Angry Birds, Instagram, Tinder, Twitter, etc ...
5. Usuarios. Estos usuarios se descargan las apps de las tiendas de aplicaciones.
6. País. Nos servira tanto para usarlo despues como residencia fiscal de las empresas como para la nacionalidad de los usuarios.
7. Categoría. Esta entidad nos sera util para cruzarla con Apps en una tabla. Ejemplos: Productividad, Juegos, Redes Sociales, etc ...

1.2. Identificación de Entidades y Sus respectivos Atributos

A continuación se mostraran graficamente las entidades y sus atributos.

1.2.1. Tienda Aplicaciones

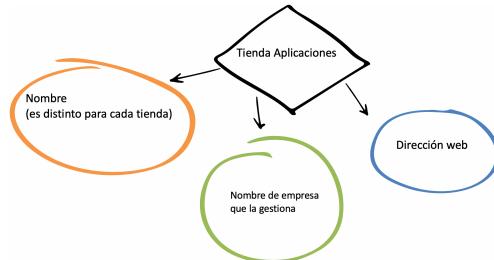


Figure 1: Diagrama tienda de aplicaciones y sus atributos

Comentarios: De la tienda sabemos el nombre, que es distinto para cada tienda, quien la gestiona (Android, Apple, Amazon,) y dirección web.

1.2.2. Empresa Desarrolladora

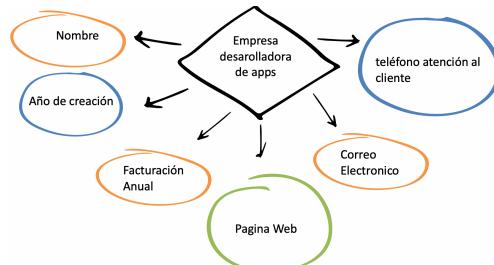


Figure 2: Diagrama empresa desarrolladora y sus atributos

Comentarios: De las empresas de servicios que desarrollan las aplicaciones (apps), conocemos su nombre, país en el que paga sus impuestos, año de creación, correo electrónico , pagina web y teléfonos de atención al cliente. Como puede tener varios teléfonos, el atributo teléfono es multivalorado y crea tabla. Ademas Sabemos que estas empresas no tienen ninguna conexión con las tiendas de aplicaciones.

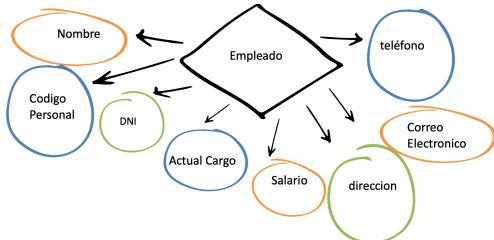


Figure 3: Diagrama personal y sus atributos

1.2.3. Empleado

Comentarios: Del empleado nos interesa el nombre, código personal en la empresa, DNI, cargo actual, salario, dirección (calle, número, código postal), correo electrónico y teléfono. Como la mayoría van a ser personas extranjeras el nombre solo estará compuesto de Nombre y Primer Apellido y los consideraremos como un único atributo.

1.2.4. App

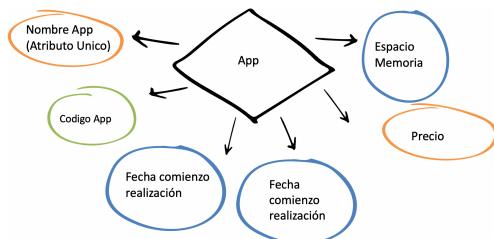


Figure 4: Diagrama app y sus atributos

Comentarios: De las apps conocemos su nombre que es único, el código de aplicación, la fecha en la que se comenzó a realizar y fecha de terminación, precio y espacio de memoria.

1.2.5. Usuario

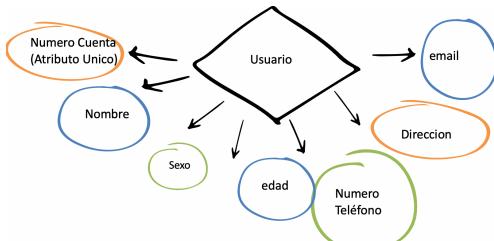


Figure 5: Diagrama usuario y sus atributos

Comentarios: Del usuario conocemos el número de cuenta que es único, nombre, sexo, edad, dirección, email, si se descarga la aplicación en el teléfono conocemos el número de móvil .

1.2.6. País

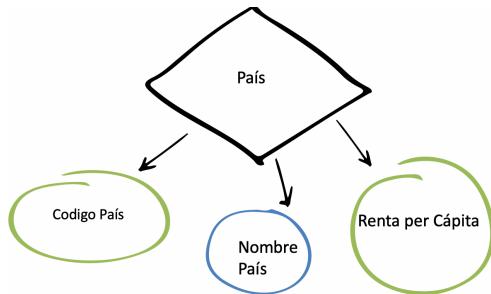


Figure 6: Diagrama país y sus atributos

Comentarios: Del país conocemos el código del país, nombre y renta per cápita.

1.2.7. Categoría

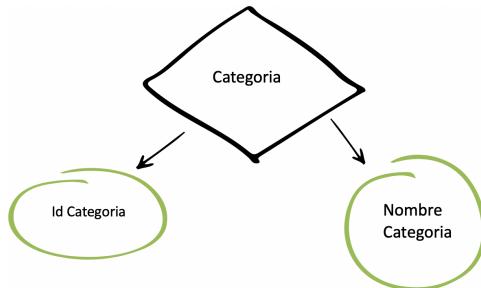


Figure 7: Diagrama categoría y sus atributos

Comentarios: Del país conocemos el código del país, nombre y renta per cápita.

1.3. Identificación de las relaciones

1.3.1. Empresa y País

Cada Empresa solo puede tener una residencia fiscal pero un país puede ser la residencia tributaria de varios países. Entonces la relación país: empresa es de 1:N.

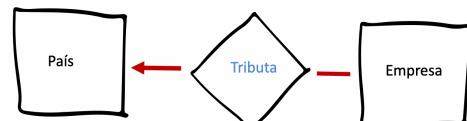


Figure 8: Relación (cardinalidad) entidades país y empresa

1.3.2. Empresa y Empleado

En cada empresa de desarrollo de apps trabajan una serie de empleados. El empleado **no** puede trabajar en varias empresas al mismo tiempo. Entonces en la relación *trabaja*, las entidades empresa y empleado tienen una relación 1:N así que pondremos el código de la empresa en la tabla empleado (Se coloca la clave primaria del lado 1 en la entidad que está en la parte N).

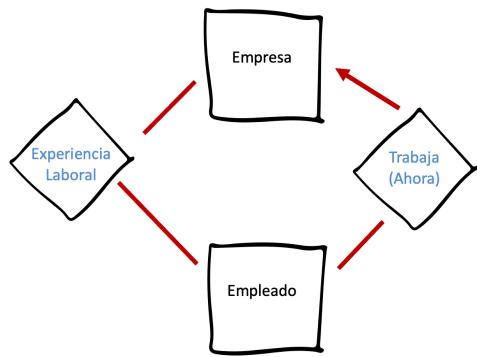


Figure 9: Relación (cardinalidad) entidades empresa y empleado

Por otro lado el empleado puede haber trabajado en varias empresas del sector e incluso puede trabajar en la misma empresa en distintos periodos de tiempo, nos interesa conocer la experiencia profesional del empleado. En este caso la relación sera de N:N y crea tabla.

1.3.3. Empresa y App

Empresa y App tienen una relación 1:N así que pondremos el código de la empresa en la tabla app (Se coloca la clave primaria del lado 1 en la entidad que esta en la parte N)

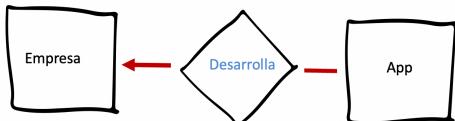


Figure 10: Relación (cardinalidad) entidades empresa y app

1.3.4. Empleado y App

Entre empleado y app hay dos tipos de relaciones. la relación *dirige* es una relación 1:N (ya que cada aplicación es dirigida por un empleado y un empleado puede dirigir varias aplicaciones) así que pondremos el código del empleado que dirige la app en la tabla app (Se coloca la clave primaria del lado 1 en la entidad que esta en la parte N)

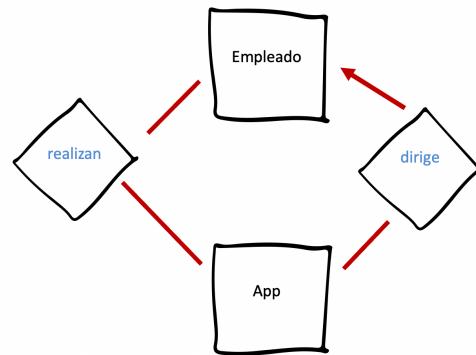


Figure 11: Relación (cardinalidad) entidades empleado y app

En el caso de la relación *realizan* cada aplicación está realizada por un grupo de empleados. La relación entre cada app con cada empleado (N:N) debe quedar registrada, así si en un futuro se quiere subir el sueldo de los realizadores de cierta app podremos tener acceso a los empleados que trabajaron sobre ella

Empleado realiza app, relación N:N se convierte en una tabla(cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (empleado y app), además tendrá los atributos de la relación.)

1.3.5. App y Categoría

Una app puede pertenecer a varias categorías y una categoría puede contener varias apps (N:N).

Esta relación N:N se convierte en una tabla. (cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (app y categoría), además tendrá los atributos de la relación)

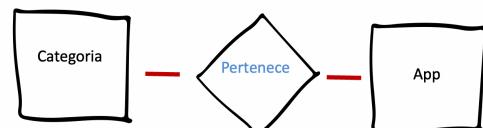


Figure 12: Relación (cardinalidad) entidades tienda aplicaciones y app

1.3.6. Tienda Aplicaciones y App

Las aplicaciones son subidas a las tiendas o plataformas . Una misma aplicación puede ser subida a varias tiendas, por supuesto una tienda tiene muchas aplicaciones (N:N).

App esta disponible en tienda, relación N:N se convierte en una tabla. (cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (app y

tienda), además tendrá los atributos de la relación)

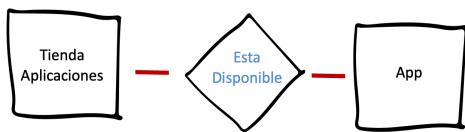


Figure 13: Relación (cardinalidad) entidades tienda aplicaciones y app

1.3.7. App y Usuario

Un usuario puede descargar o no aplicaciones y cada app puede ser descargada por varios usuarios. El usuario no puede descargar dos veces la misma aplicación.

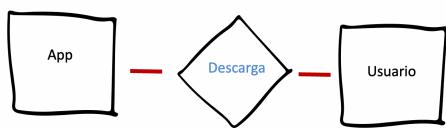


Figure 14: Relación (cardinalidad) entidades tienda aplicaciones y app

Usuario descarga app, relación N:N se convierte en una tabla, cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (usuario y app), además tendrá los atributos de la relación.

1.3.8. Modelo Entidad-Relacion Global

Juntamos los graficos de relaciones anteriores en un mismo grafico.

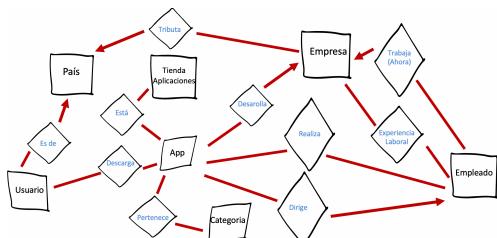


Figure 15: Relación (cardinalidad) entre entidades global

Comentarios: Tal y como se ha indicado en el enunciado sabemos que las empresas desarrolladoras de apps no tienen ninguna conexión con las tiendas de aplicaciones.

2. Diseño Logico Entidades(Paso a Tablas)

Nota: En cada caso el atributo(s) subrayado sera el Primary Key.

2.1. País

pais(CodPais, NomPais, RentaPCap)

País y empresa tienen una relación 1:N así que colocamos la clave primaria del lado 1 (país) en la entidad que esta en la

parte N (empresa)

2.2. Empresa

En este caso una empresa tiene varios números de teléfono de atención al cliente así que será un atributo multivalorado. Un número de teléfono no puede estar compartido por dos empresas, por eso el atributo multivalorado genera tabla.

empresa(
CodEmpresa, Nombre, AnoCreacion,
FacturacionAnual, Paginaweb, email)

2.3. Telefono

Como es una relación 1:N se coloca la clave primaria del lado 1 (empresa) en la entidad que está en la parte N (telefono)

telefono(numtel, cEmpresa)

2.4. Empleado

Como ya comentamos antes como la mayoría van a ser personas extranjeras el nombre solo estará compuesto de Nombre y Primer Apellido y los consideraremos como un único atributo.

empleado(
CodEmpleado, NombreEmpleado, NIF, CargoActual,
Salario, direccion, email, telefono, Paginaweb, email)

2.5. Tienda Aplicaciones

tiendaApp(
NombreTienda, EmpresaGestora, website)

2.6. App

Una app puede ser incluida en varias categorías. Nos interesa tener una tabla que cruce las apps con las categorías así podríamos buscar categorías y que nos aparezcan todas las apps que están en esa categoría. Entonces categoría es un atributo multivalorado y genera tabla.

App(
NombreApp, CodigoApp, FchComienzoRealizacion,
FchFinRealizacion, Categoría, Precio, EspacioMemoria
)

2.7. Categoría

Como es una relación 1:N se coloca la clave primaria del lado 1 (empresa) en la entidad que está en la parte N (telefono)

categoria(idCategoria, NombreCategoria)

2.8. Usuario

usuario(
NumeroCuenta, Nombre, Sexo,
Edad, NumTel, direccion, Email)

3. Diseño Logico Relaciones(Paso a Tablas)

3.1. Esde

Cada usuario solo puede tener una nacionalidad así que tenemos relación 1:N y ponemos la clave primaria del 1 (país) en la tabla del N (usuario).

```
usuario(  
    NumeroCuenta, Nombre, Sexo,  
    Edad, NumTel, direccion, Email, codPais)
```

3.2. Tributa

Es una relación 1:N. Ponemos la clave primaria del 1 (país) en la tabla del N (empresa).

```
empresa(  
    CodEmpresa, Nombre, AnoCreacion,  
    FacturacionAnual, Paginaweb, email, idPais)
```

3.3. Desarolla

Como la relación empresa app es de 1:N incluiremos la clave primaria del 1 en el N

```
App(  
    NombreApp, CodigoApp, FchComienzoRealizacion,  
    FchFinRealizacion, Categoria,  
    Precio, EspacioMemoria, IDempresa)
```

3.4. Trabaja

En la relación *Trabaja*, empresa y empleado tienen una relación 1:N. Por eso incluiremos la clave primaria del 1 en el N

```
empleado(  
    CodEmpleado, NombreEmpleado, NIF, CargoActual,  
    Salario, direccion, email, telefono,  
    Paginaweb, emailidEmpresa)
```

Por otro lado la relación experiencia laboral es de N:N y crea una tabla cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (empleado y empresa), además tendrá los atributos de la relación

```
ExpLabo(  
    CodEmpleado, cEmpresa, Puesto,  
    FechaInicio, FechaFinalizacion)
```

```
ExpLaboral(  
    cEmpresa, cEmpleado, Puesto, FechaInicio, FechaFin)
```

3.5. Realiza

```
haRealizado(  
    nomApp, cEmpleado)
```

3.6. Dirige

Como la relación es de 1:N incluiremos la clave primaria del 1 en el N

```
App(  
    NombreApp, CodigoApp, FchComienzoRealizacion,  
    FchFinRealizacion, Categoria, Precio,  
    EspacioMemoria, IDempresa, IDdirigio)
```

3.7. Esta

Las aplicaciones son subidas a las tiendas o plataformas. Una misma aplicación puede ser subida a varias tiendas, por supuesto una tienda tiene muchas aplicaciones. (N:N)

App está disponible en tienda, relación N:N se convierte en una tabla, Como la relación es de 1:N incluiremos la clave primaria del 1 en el N

```
Estadisponible(  
    NombreApp, NombreTienda)
```

3.8. Pertenece

```
EsCategoria(  
    NombreApp, idcategoria)
```

3.9. Descarga

```
Descarga(  
    NombreApp, idUser, fechaDescarga,  
    numdescargas, tiendaDescarga)
```

4. Implementación en MySQL

4.1. Pasos Iniciales

4.1.1. Definición del Data Base

```
--  
11 -- Definir un data base con el nombre APPStore  
12 -- en la que estarán todas nuestras tablas  
13  
14 • DROP DATABASE IF EXISTS APPSTORE;  
15 • CREATE DATABASE APPSTORE;  
16 • use APPSTORE;  
17
```

Figure 16: Definición de la base de datos que almacenara nuestras tablas

A continuación hacemos *Drop* de las tablas ya existentes para que no tengamos ninguna tabla con el mismo nombre en nuestro data base APPSTORE.

4.1.2. Drop Already Existing Tables

Drop de tablas de entidades:

```
--  
19 # Drop Already Existing tables with the following names:  
20  
21 #Entidades  
22 • DROP TABLE IF EXISTS empresa;  
23 • DROP TABLE IF EXISTS telefono;  
24 • DROP TABLE IF EXISTS empleado;  
25 • DROP TABLE IF EXISTS tienda; #tienda de aplicaciones  
26 • DROP TABLE IF EXISTS app;  
27 • DROP TABLE IF EXISTS usuario;  
28 • DROP TABLE IF EXISTS pais;  
29 • DROP TABLE IF EXISTS categoria;
```

Drop de tablas de relaciones:

```
31 #Relaciones  
32 • DROP TABLE IF EXISTS ExpLabo; #Cardinalidad entre empresa y empleado  
33 • DROP TABLE IF EXISTS estaDisponible; #Cardinalidad entre app y tienda  
34 • DROP TABLE IF EXISTS haRealizado; #Cardinalidad entre empleado y app  
35 • DROP TABLE IF EXISTS Descarga; # #Cardinalidad entre usuario y app  
36 • DROP TABLE IF EXISTS esCategoria; # #Cardinalidad entre app y categoria  
37
```

4.2. Definición de la Estructura

Paso seguido definiremos la estructura de nuestras tablas y sus llaves primarias.

```
41 #Paso a Tablas  
42  
43 • CREATE TABLE pais(  
44 CodPais INT PRIMARY KEY,  
45 NombrePais VARCHAR (30),  
46 RentPCapi numeric(10,2) default 0 # 10 digitos de los cuales 2 son decimales  
47 );  
48  
49  
50 • CREATE TABLE empresa(  
51 CodEmpresa INT PRIMARY KEY,  
52 Nombre VARCHAR(30) NOT NULL,  
53 AñoCreacion CHAR (4),  
54 direccion_ofi_cntral VARCHAR (30), # Ciudad, Calle, Numero  
55 FacturacionAnual numeric(9,2), #Facturacion en Millones de $  
56 PaginaWeb VARCHAR(30) NOT NULL,  
57 Email VARCHAR (30) NOT NULL,  
58 codPais INT,  
59 FOREIGN KEY (codPais) REFERENCES pais(CodPais)  
60 ON DELETE RESTRICT ON UPDATE cascade  
61 );
```

Comentarios: En la tabla *empresa* podríamos haber definido direccion como un atributo compuesto pero para el actual problema no es algo que despues nos vaya a servir mucho.

Como ya habiamos mencionado colocamos la llave primaria de *país* en la tabla *empresa* debido a su relacion 1:N.

Tambien añadimos el termino *On Update Cascade* para que cada vez que se actualize un registro en la tabla país tambien se inserten las respectivas empresas que *tributan* en ese país. Pero en este caso el termino *On Delete Restrict* para que en caso de que se borre un país **no** se borren todas las empresas que tributan en él.

En general Los comandos *ON DELETE ...* y *ON UPDATE ...* son para saber que hacer con los Foreign Key en caso de aplicar cambios.

```
65 #Una empresa puede tener varios numeros de telefono,  
66 #lo que lo convierte en un atributo multivariado  
67 #Los atributos multivariados crean tabla  
68 • CREATE TABLE telefono(  
69 NumTelefono VARCHAR (10),  
70 codempresa INT,  
71 Primary Key (NumTelefono),  
72 #Foreign Key significa que es clave primaria (Primary Key) en otra tabla  
73 Foreign Key (codempresa)  
74 References empresa (CodEmpresa)  
75 ON DELETE CASCADE  
76 ON UPDATE CASCADE  
77 );  
78
```

Comentarios: Una empresa puede tener varios numeros de telefono, pero un numero de telefono **no** puede estar compartido por varias empresas lo que lo convierte en un atributo multivariado. Los atributos multivariados crean tabla. la llave primaria de *empresa* en la tabla *telefono* debido a su relacion 1:N.

```
89 • CREATE TABLE empleado(  
90 CodEmpleado CHAR(10) PRIMARY KEY ,  
91 NIF VARCHAR (9) NOT NULL,  
92 nombreCompleto VARCHAR(30) NOT NULL,  
93 CarreraActual VARCHAR (10),  
94 Salario numeric (9,2), # Salario en $  
95 direccion VARCHAR(100),  
96 telefono VARCHAR (10) NOT NULL,  
97 emailEmpleado VARCHAR (50) NOT NULL,  
98 cod_empresa INT, #Codigo (Primary Key) de la empresa para la que trabaja ahora  
99 FOREIGN KEY (cod_empresa)  
100 REFERENCES empresa(CodEmpresa)  
101 ON DELETE CASCADE  
102 ON UPDATE CASCADE  
103 );  
104 # Si se elimina una empresa, se elimina todo el personal que trabaja  
105 #en el ahora
```

Comentarios: El empleado no puede trabajar en varias empresas al mismo tiempo empresa y empleado tienen una relacion 1:N asi que pondremos el codigo de la empresa en la tabla empleado (Se coloca la clave primaria del lado 1 en la entidad que esta en la parte N).

Ademas utilizando los terminos de *On Update Cascade* y *On Delete Cascade* en caso de actualizacion o eliminacion de una *empresa* se borrarán todos los empleados que *trabajen* en esa empresa.

Por ultimo decir que los atributos Nombre, NIF, email y telefono del empleado son imprescindibles en cualquier empresa para identificar o contactar con un empleado. Es por ello que utilizamos el termino *NOT NULL* para ellos.

```
118  
119 • CREATE TABLE tienda (  
120 NombreTienda VARCHAR (30) PRIMARY KEY,  
121 EmpresaGestora VARCHAR (30) NOT NULL,  
122 Website VARCHAR(30)  
123 );  
124
```

Comentarios: Como el nombre de cada tienda es unico podemos utilizarlo como llave primaria.

```

--+
192 #El nombre de cada app es unico, por eso podemos utilizar este
193 #atributo como primary key
194 • CREATE TABLE app(
195     Nombre VARCHAR (50) PRIMARY KEY,
196    CodigoApp CHAR (4) NOT NULL,
197     StartDevDate DATE ,
198     EndDevDate DATE ,
199     Precio numeric(6,2) not null,
200     EspacioMemoria numeric(6,2) not null,
201     cod_empresa INT,
202     cod_empleado_dirigido CHAR(6),
203
204     FOREIGN KEY (cod_empresa) REFERENCES empresa(CodEmpresa)
205     ON DELETE CASCADE ON UPDATE CASCADE,
206
207     FOREIGN KEY (cod_empleado_dirigido)
208     REFERENCES empleado(CodEmpleado)
209     ON DELETE RESTRICT ON UPDATE CASCADE
210 );

```

Comentarios: El nombre de cada app es unico, por eso podemos utilizar este atributo como primary key (aunque tambien podriamos haber definido un atributo *CodApp* y ponerlo como llave primaria)

Empresa y App tienen una relacion 1:N asi que pondremos el codigo de la empresa en la tabla app (Se coloca la clave primaria del lado 1 en la entidad que esta en la parte N).

Lo mismo pasa con el empleado que dirige la app y la App tienen una relacion 1:N asi que pondremos el codigo del empleado que dirige la app en la tabla app (Se coloca la clave primaria del lado 1 en la entidad que esta en la parte N).

En este caso la *app* es eliminada en caso de eliminacion de la *empresa* porque ya no habra quien de soporte o lance actualizaciones pero no es eliminada en el caso que el empleado que dirijo la app sea eliminado (por que al fin y al cabo es otro empleado mas).

Aun asi los update seran en cascada.

```

176
177
178 • CREATE TABLE categoria (
179     idCategoria INT AUTO_INCREMENT PRIMARY KEY,
180     nombre_categoria VARCHAR (50) NOT NULL
181 );
182

```

Comentarios: La tabla categoria solo tienen dos campos. El campo de *idCategoria* sera la Primary Key e incrementara de forma automatica.

```

186 • CREATE TABLE usuario (
187     CuentaUsuario VARCHAR (10) PRIMARY KEY,
188     nombreUsuario varchar(40) not null,
189     direccion varchar(40), #Hoy en dia la direccion no es muy importante
190     telefono varchar(9), #Podemos tenerlo o no
191     email varchar(30) NOT NULL, # Ya que el telefono podemos no tenerlo o no,
192     #es vital tener el email del usuario
193     sexo char (2) NOT NULL,
194     edad SMALLINT NOT NULL,
195     Codpais INT,
196     FOREIGN KEY (Codpais) REFERENCES pais(CodPais)
197     ON DELETE RESTRICT ON UPDATE CASCADE
198 );

```

Comentarios: Segun se nos ha dicho el numero de telefono podemos tenerlo o no. Por eso es imprescindible tener el correo electronico del usuario para que las empresas le puedan enviar promociones o informacion al usuario en caso de necesidad. La llave primaria de *pais* en la tabla *usuario* debido a su relacion 1:N. En este caso nos interesa poner el termino *On Delete Restrict* porque un usuario nos aporta bastante informacion aparte de su nacionalidad y **no** nos interesaría perderla.

```

268 • CREATE TABLE ExpLobo(
269     cod_empresa INT NOT NULL,
270     cod_empleado CHAR(6) NOT NULL,
271     Puesto VARCHAR (30),
272     FechaInicio DATE NOT NULL,
273     FechaFin DATETIME NOT NULL DEFAULT NOW(), #Si la fecha fin no ha sido
274     #introducida asumimos por defecto que aun a dia de hoy sigue trabajando ahí
275     PRIMARY KEY (cod_empresa,cod_empleado, FechaInicio,FechaFin),
276     FOREIGN KEY (cod_empresa) REFERENCES empresa(CodEmpresa)
277     ON DELETE CASCADE ON UPDATE CASCADE,
278     FOREIGN KEY (cod_empleado) REFERENCES empleado(CodEmpleado)
279     ON DELETE CASCADE ON UPDATE CASCADE
280 );
281 /*
282 Eliminamos la experiencia laboral en caso de eliminacion
283 de la empresa o el empleado
284 */

```

Comentarios: En este caso como un empleado puede haber trabajado en la misma empresa en varios periodos de tiempo diferentes tambien pondremos las fechas de inicio y fin de esa etapa como llaves primarias.

Si la fecha de finalizacion no ha sido indicada asumimos por defecto que aun a dia de hoy el empleado sigue trabajando en esa empresa. Ademas utilizando *On Update Cascade* y *On Delete Cascade* actualizaremos o eliminaremos la experiencia laboral en caso de eliminacion de la empresa o el empleado.

```

251 • CREATE TABLE estaDisponible(
252     nombre_tienda VARCHAR (30),
253     nombre_app VARCHAR (50),
254     PRIMARY KEY (nombre_tienda,nombre_app),
255     FOREIGN KEY (nombre_tienda) REFERENCES tienda(NombreTienda)
256     ON DELETE CASCADE ON UPDATE CASCADE,
257
258     FOREIGN KEY (nombre_app) REFERENCES app(Nombre)
259     ON DELETE CASCADE ON UPDATE CASCADE
260 );
261

```

Comentarios: Las aplicaciones son subidas a las tiendas o plataformas Una misma aplicacion puede ser subida a varias tiendas, por supuesto una tienda tiene muchas aplicaciones. (N:N)

App esta disponible en tienda, relacion N:N se convierte en una tabla, cuya clave primaria es la union de las claves primarias de las entidades que participan en la relacion (app y tienda), ademas tendra los atributos de la relacion.

```

282 • CREATE TABLE harealizado(
283     codigo_empleado CHAR (6),
284     nombre_App VARCHAR (50),
285     PRIMARY KEY (codigo_empleado,nombre_App),
286     FOREIGN KEY (codigo_empleado) REFERENCES empleado (CodEmpleado)
287     ON DELETE CASCADE ON UPDATE CASCADE,
288
289     FOREIGN KEY (nombre_App) REFERENCES app(Nombre)
290     ON DELETE CASCADE ON UPDATE CASCADE
291 );
292

```

Comentarios: Empleado realiza app, relacion N:N se convierte en una tabla, cuya clave primaria es la union de las claves primarias de las entidades que participan en la relacion (tienda y app), ademas tendra los atributos de la relacion

```

302 • Ⓜ Create Table es_categoria(
303     nombr_app VARCHAR (50),
304     id_categ Int,
305     PRIMARY KEY (nombr_app,id_categ),
306     FOREIGN KEY (nombr_app) REFERENCES app(Nombre)
307     ON DELETE CASCADE ON UPDATE CASCADE,
308     FOREIGN KEY (id_categ) REFERENCES categoria(idCategoria)
309     ON DELETE CASCADE ON UPDATE CASCADE
310 );

```

Comentarios: Cada app puede ser incluida en una o varias categorías.

Las entidades que participan - categoria y app - tienen una relación N:N que se convierte en una tabla, cuya clave primaria es la unión de las claves primarias de las dos entidades participantes.

```

CREATE TABLE descarga(
idusuario VARCHAR (10) NOT NULL,
nombre_app VARCHAR (50) NOT NULL,
#Nombre de la tienda de la que el usuario
#se ha descargado la app
377 #Puede ser Null en caso de que no haya descarga
378 nombre_tienda VARCHAR (30),
379 #Si la jamas ha sido descargada la
380 #fecha de descarga sera null
381 fecha_descarga date not null ,
382 # Por defecto asumimos que el usuario se ha descargado la app
383 num_descargas integer default 1,
384
385 constraint check (num_descargas<2),
386 PRIMARY KEY (idusuario,nombre_app),
387 FOREIGN KEY (idusuario) REFERENCES usuario(CuentaUsuario)
388 ON DELETE CASCADE ON UPDATE CASCADE,
389 FOREIGN KEY (nombre_app) REFERENCES app(Nombre)
390 ON DELETE CASCADE ON UPDATE CASCADE
391
392 );

```

Comentarios: Usuario descarga app, relación N:N se convierte en una tabla, cuya clave primaria es la unión de las claves primarias de las entidades que participan en la relación (usuario y app), además tendrá los atributos de la relación.

El numero de descargas siempre sera 0 (no descargado) o 1 (descargado) El usuario solo puede descargarse la app una vez
El trigger check nos ayuda a validar los datos entrantes.

4.3. Introducción de Datos en Tablas

En nuestro caso vamos a introducir los datos de una de estas dos maneras:

1. Utilizando Insert Into {table name}
2. Utilizando LOAD DATA LOCAL INFILE {csv file path}

El sistema operativo utilizado es *Mac OS X®* y por eso al contrario que en los sistemas operativos windows la función "SELECT [GLOBAL — SESSION] VARIABLES [LIKE 'pattern' — WHERE expr]" no puede ser utilizada para indicarnos el nombre del directorio donde hay que poner los archivos csv.

Por esta razón lo mas fácil fue colocar la expresión *LOCAL* para que así no tengamos problemas a la hora de importar los archivos csv.

```

INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (1,'USA','74725.10');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (2,'China','10500.40');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (3,'Canada','43258.17');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (4,'Germany','46208.42');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (5,'Sweden','52274.41');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (6,'Spain','27063.19');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (7,'Russia','10126.72');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (8,'France','39030.36');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (9,'India','1927.708');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (10,'Saudi Arabia','20110.31');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (11,'Brazil','6796.84');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (12,'Australia','51692.84');
INSERT INTO pais(CodPais,NomPais,RentPCapi) VALUES (13,'Japan','40193.25');

```

```

428 #Insertar datos del archivo csv a la tabla empresa
429
430 • Ⓜ LOAD DATA LOCAL INFILE '/Users/soheilmattarmohammadiberenguer/Documents/Complu_Uni/HomeWork/SQL/MI_tarea/Empresas.csv'
431 INTO TABLE empresa
432 CHARACTER SET latin1
433 FIELDS TERMINATED BY ','
434 LINES TERMINATED BY '\n';
435
436 #Insertar datos del archivo csv a la tabla tienda
437
438 • Ⓜ LOAD DATA LOCAL INFILE '/Users/soheilmattarmohammadiberenguer/Documents/Complu_Uni/HomeWork/SQL/MI_tarea/tiendas.csv'
439 INTO TABLE tienda
440 CHARACTER SET latin1
441 FIELDS TERMINATED BY ','
442 LINES TERMINATED BY '\n';
443
444 ...

```

```

468 #Insertar en tabla empleado
469 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
470 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100001', '85732996020',
471 ,'"Jane Doe","Senior Software Engineer",520000.00,
472 ,"(California, Mountain View, St. Shoreline Blv. 34)","+185 4155552671",
473 ,'"Doe@hotmail.com",3);
474
475 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
476 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100002', '938568K',
477 ,'"Marcel Dettmann","VR Engineer",345500.00,"(Berlin, Friedrichschain Strasse 54)",
478 ,"+49 654030418","MarcelDettmann@yahoo.de",6);
479 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
480 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100003', '57326020B',
481 ,'"Pierre Boulez","Backend Developer",187000.00,"(Paris, Vie de Arc 57)",
482 ,"+33 4155592671","PBou75@yahoo.fr",7);
483
484 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
485 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100004', '287632004020',
486 ,'"Jeff Mills","Mobile Developer",180700.00,"(California, Palo Alto, Fifth Road 12)",
487 ,"+185 732996020","JMILLS@gmail.com",2);
488
489 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
490 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100005', '831630054191',
491 ,'"Niki Istrefi","Mobile Developer",215000.00,"(California, San Francisco 128)",
492 ,"+185 654983520","Nistrefi@gmail.com",1);

```

```

495 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
496 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100006', '123831630054191',
497 ,'"Wang Zhang Liu","VR Engineer",98000.00,"(Beijing, Ping Li Un 77)"',
498 ,"+86 727 9863 5566","wzLiu@gmail.com",9);
499
500 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
501 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100007', '789831630054228',
502 ,'"Chen Yang Li","Principal Engineer",150000.00,"(Pekin, Mao Street, 99)"',
503 ,"+86 433 3083 2270","ChenYangLi@yahoo.com",8);
504 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
505 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100008', '554831630059090',
506 ,'"Yang Zhao Hun","Mobile Developer",140000.00,",+86 128 5352 2266",
507 ,"+YangZhao1995@yahoo.com",9);
508 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
509 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100009', 'SW57326020',
510 ,'"Ida Engberg","Data Developer",183000.00,"(Estokholm, Norrmalm 55)", "+45 636555129",
511 ,"+Engberg.Ida@gmail.com",10);
512 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
513 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100010', '46571178Z',
514 ,'"Esteban Garcia",
515 ,'"Data Developer",98000.00,"(Madrid, Calle Cuatro Amigos 6)",
516 ,"+34 636555129","e_garcia@yahoo.es",5);

```

```

520
521 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
522 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100011', '15789733456509',
523 ,'"Richie Hawtin","Senior Software Engineer",NULL,"(London, St. John 88)", "+25 392996011",
524 ,'"Richard_H@yahoo.com",3);
525
526 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
527 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100012', '4782334M , "Cesar Almena",
528 ,'"Mobile Developer",63000.00,"(Madrid, Calle San Bernardo 72)", "+34 883665218",
529 ,'"C_Almena_91@yahoo.es",5);
530
531 • Ⓜ INSERT INTO empleado(CodEmpleado,NIF,nombreCompleto,CargoActual,Salario,
532 direccion,telefono,emailEmpleado,cod_empresa) VALUES ('100013', '85732996020',
533 ,'"Peter Griffin","Backend Developer",187000.00, "", "+185 8225552115",
534 ,'"petergriff@gmail.com",4);

```

```

--+
639 • INSERT INTO Explab
640     VALUES (3,'100010', "Data Developer","2014-07-10","2017-12-01");
641 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
642     VALUES (10,'100010', "Data Developer","2018-01-01");
643
644
645 • INSERT INTO ExpLab(cod_empresa,cod_empleado,Puesto,FechaInicio)
646     VALUES (10,'100011', "Senior Software Engineer","2008-06-01");
647
648 • INSERT INTO ExpLab
649     VALUES (9,'100012', "Mobile Developer","2017-11-01","2018-11-13");
650 • INSERT INTO ExpLab(cod_empresa,cod_empleado,Puesto,FechaInicio)
651     VALUES (5,'100012', "Mobile Developer","2019-01-01");
652
653 • INSERT INTO ExpLab
654     VALUES (3,'100013', "Backend Developer" , "2006-12-01","2021-09-01");
655 • INSERT INTO ExpLab(cod_empresa,cod_empleado,Puesto,FechaInicio)
656     VALUES (2,'100013', "Backend Developer","2021-10-01");
657

666 #Insertar en tabla categoria
667
668 • INSERT INTO categoria VALUES (1,'Medical');
669 • INSERT INTO categoria VALUES (2,'Entertainment');
670 • INSERT INTO categoria VALUES (3,'Educational');
671 • INSERT INTO categoria VALUES (4,'Games');
672 • INSERT INTO categoria VALUES (5,'Utilities');
673 • INSERT INTO categoria VALUES (6,'Traveling');
674 • INSERT INTO categoria VALUES (7,'Sports');
675 • INSERT INTO categoria VALUES (8,'News');
676 • INSERT INTO categoria VALUES (9,'Lifestyle');
677 • INSERT INTO categoria VALUES (10,'Video and Image');
678 • INSERT INTO categoria VALUES (11,'Fashion');
679 • INSERT INTO categoria VALUES (12,'Music');
680 • INSERT INTO categoria VALUES (13,'Social Networking');
681 • INSERT INTO categoria VALUES (14,'Business');
682 • INSERT INTO categoria VALUES (15,'Cooking');

587 #Insertar en tabla Exp_Lab
588 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio,FechaFin)
589     VALUES (3,'100001','Data Developer","2010-08-25","2013-03-30");
590 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio,FechaFin)
591     VALUES (6,'100001','Software Engineer","2013-04-15","2015-08-21");
592
593 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
594     VALUES (3,'100001','Data Developer","2016-01-21");
595
596 • INSERT INTO Explab VALUES (4,'100002', "VR Engineer","2008-02-01","2018-09-21");
597
598 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
599     VALUES (6,'100002', "VR Engineer","2018-10-12");
600
601 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
602     VALUES (7,'100003', "Backend Developer","2018-10-12");
603
604 • INSERT INTO Explab
605     VALUES (1,'100004', "Mobile Developer","2012-09-01","2019-03-18");

613 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
614     VALUES (2,'100004', "Mobile Developer","2020-10-12");
615
616 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
617     VALUES (1,'100005', "Mobile Developer","2013-08-15");
618
619 • INSERT INTO Explab
620     VALUES (9,'100006', "Mobile Developer","2017-10-14","2020-05-07");
621 • INSERT INTO Explab
622     VALUES (8,'100006', "Mobile Developer","2020-06-11","2021-06-30");
623 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
624     VALUES (9,'100006', "Production Engineer","2020-07-01");
625
626 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
627     VALUES (8,'100007', "Principal Engineer","2019-03-04");
628
629 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
630     VALUES (9,'100008', "Mobile Developer","2017-10-15");
631
632 • INSERT INTO Explab
633     VALUES (3,'100009', "Data Developer","2013-09-15","2018-03-08");
634 • INSERT INTO Explab(cod_empresa,cod_empleado,Puesto,FechaInicio)
635     VALUES (10,'100009', "Data Developer","2018-04-01");

721 #Insertar en tabñla es_categoria
722
723 • INSERT INTO es_categoria VALUES ("Birdees",2);#Entertainment
724 • INSERT INTO es_categoria VALUES ("Birdees",13); # Social Networking
725
726 • INSERT INTO es_categoria VALUES ("DentistPro",1);#Medical
727
728 • INSERT INTO es_categoria VALUES ("My Job Angel",5); #Utilities
729 • INSERT INTO es_categoria VALUES ("My Job Angel",13);#Social Networking
730 • INSERT INTO es_categoria VALUES ("My Job Angel",14);#Business
731
732 • INSERT INTO es_categoria VALUES ("Merry Kitchen",3); #Educational
733 • INSERT INTO es_categoria VALUES ("Merry Kitchen",9); #Lifestyle
734 • INSERT INTO es_categoria VALUES ("Merry Kitchen",15); #Cooking
735
736 • INSERT INTO es_categoria VALUES ("Gmail",5); #Utilities
737 • INSERT INTO es_categoria VALUES ("Gmail",13);#Social Networking
738
739 • INSERT INTO es_categoria VALUES ("Google Drive",5); #Utilities
740
741 • INSERT INTO es_categoria VALUES ("Google Maps",6);#Traveling
742 • INSERT INTO es_categoria VALUES ("Google Maps",5); #Utilities

746 • INSERT INTO es_categoria VALUES ("Facebook",2);#Entertainment
747 • INSERT INTO es_categoria VALUES ("Facebook",4);#Games
748
749 • INSERT INTO es_categoria VALUES ("Facebook",9);#Lifestyle
750 • INSERT INTO es_categoria VALUES ("Facebook",13);#Social Networking
751
752 • INSERT INTO es_categoria VALUES ("Instagram",2);#Entertainment
753 • INSERT INTO es_categoria VALUES ("Instagram",9);#Lifestyle
754 • INSERT INTO es_categoria VALUES ("Instagram",10);#Video and Image
755 • INSERT INTO es_categoria VALUES ("Instagram",13);#Social Networking
756
757 • INSERT INTO es_categoria VALUES ("WhatsApp",5);#Utilities
758 • INSERT INTO es_categoria VALUES ("WhatsApp",13);#Social Networking
759
760 • INSERT INTO es_categoria VALUES ("Cabilify",5);#Utilities
761 • INSERT INTO es_categoria VALUES ("Cabilify",6);#Traveling
762
763 • INSERT INTO es_categoria VALUES ("Flow",5);#Utilities
764 • INSERT INTO es_categoria VALUES ("Flow",14);#Business
765
766 • INSERT INTO es_categoria VALUES ("L'équipe, sports en direct",2);#Entertainment
767 • INSERT INTO es_categoria VALUES ("L'équipe, sports en direct",8);#News
768

```

```

```
771 • INSERT INTO es_categoria VALUES ("TikTok",2);#Entertainment
772 • INSERT INTO es_categoria VALUES ("TikTok",4);#Games
773 • INSERT INTO es_categoria VALUES ("TikTok",9);#Lifestyle
774 • INSERT INTO es_categoria VALUES ("TikTok",10);#Video and Image
775 • INSERT INTO es_categoria VALUES ("TikTok",13);#Social Networking
776
777 • INSERT INTO es_categoria VALUES ("Hello",13);#Social Networking
778
779 • INSERT INTO es_categoria VALUES ("QQ",2);#Entertainment
780 • INSERT INTO es_categoria VALUES ("QQ",13);#Social Networking
781
782 • INSERT INTO es_categoria VALUES ("Tencent Music",2);#Entertainment
783 • INSERT INTO es_categoria VALUES ("Tencent Music",12);#Music
784
785 • INSERT INTO es_categoria VALUES ("PubG",2);#Entertainment
786 • INSERT INTO es_categoria VALUES ("PubG",4);#Games
787
788 • INSERT INTO es_categoria VALUES ("Spotify",2);#Entertainment
789 • INSERT INTO es_categoria VALUES ("Spotify",12);#Music
790

697 #Insertar en tabla estaDisponible
698
699
700 • LOAD DATA LOCAL INFILE '/Users/soheilmoattaromohammadiberenguer/Documents/Complu_Uni/HomeWork/SOL/MI_Tarea/EstaDisp.csv'
701 INTO TABLE estaDisponible
702 CHARACTER SET latin1
703 FIELDS TERMINATED BY ','
704 LINES TERMINATED BY '\n';

796 #Insertar en tabla hrealizado
797 #El empleado que dirige la app tambien es incluido en el grupo
798 # de empleados que han dirigido la app aunque podríamos no hacerlo.
799
800 • INSERT INTO hrealizado VALUES ("000001","Flow");
801 • INSERT INTO hrealizado VALUES ("000001","Google Maps");
802
803 • INSERT INTO hrealizado VALUES ("000002","Facebook");
804 • INSERT INTO hrealizado VALUES ("000002","Instagram");
805 • INSERT INTO hrealizado VALUES ("000002","Flow");
806
807 • INSERT INTO hrealizado VALUES ("000003","L'équipe, sports en direct");
808
809 • INSERT INTO hrealizado VALUES ("000004","Birdees");
810 • INSERT INTO hrealizado VALUES ("000004","My Job Angel");
811
812 • INSERT INTO hrealizado VALUES ("000005","Birdees");
813 • INSERT INTO hrealizado VALUES ("000005","My Job Angel");
814 • INSERT INTO hrealizado VALUES ("000005","DentistPro");
815
816 • INSERT INTO hrealizado VALUES ("000006","QQ");
817 • INSERT INTO hrealizado VALUES ("000006","Tencent Music");
818 • INSERT INTO hrealizado VALUES ("000006","PubG");

```
826 • INSERT INTO hrealizado VALUES ("000007","TikTok");
827 • INSERT INTO hrealizado VALUES ("000007","DentistPro");
828
829 • INSERT INTO hrealizado VALUES ("000008","QQ");
830 • INSERT INTO hrealizado VALUES ("000008","Tencent Music");
831 • INSERT INTO hrealizado VALUES ("000008","PubG");
832
833 • INSERT INTO hrealizado VALUES ("000009","Google Maps");
834 • INSERT INTO hrealizado VALUES ("000009","Gmail");
835 • INSERT INTO hrealizado VALUES ("000009","Spotify");
836
837 • INSERT INTO hrealizado VALUES ("000010","Merry Kitchen");
838 • INSERT INTO hrealizado VALUES ("000010","Cabify");
839
840 • INSERT INTO hrealizado VALUES ("000011","Facebook");
841 • INSERT INTO hrealizado VALUES ("000011","Instagram");
842 • INSERT INTO hrealizado VALUES ("000011","Whatsapp");
843
844 • INSERT INTO hrealizado VALUES ("000012","QQ");
845 • INSERT INTO hrealizado VALUES ("000012","Cabify");
846
847 • INSERT INTO hrealizado VALUES ("000013","Gmail");
848 • INSERT INTO hrealizado VALUES ("000013","Google Drive");

858 #insertar en tabla descargas
859 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
860 VALUES ("50000001","Birdes","Appstore","2020-11-14");
861 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
862 VALUES ("50000001","Spotify","Appstore","2020-10-23");
863
864 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
865 VALUES ("50000003","DentistPro","App Store","2021-01-10");
866 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
867 VALUES ("50000013","Gmail","App Store","2022-01-03");
868 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
869 VALUES ("50000013","Flow","App Store","2021-01-03");
870 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
871 VALUES ("50000001","Cabify","App Store","2022-01-06");
872
873 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
874 VALUES ("50000003","TikTok","AppGallery","2020-12-15");
875 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
876 VALUES ("50000003","QQ","AppGallery","2022-01-01");
877
878 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
879 VALUES ("50000004","TikTok","MyApp","2020-12-15");
880 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
881 VALUES ("50000004","Tencent Music","MyApp","2022-01-01");

884 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
885 VALUES ("50000006","Cabify","Google Play","2020-09-21");
886 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
887 VALUES ("50000006","Instagram","Google Play","2020-09-21");
888 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
889 VALUES ("50000006","Facebook","Google Play","2020-09-30");
890 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
891 VALUES ("50000006","WhatsApp","Google Play","2020-10-02");
892 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
893 VALUES ("50000006","TikTok","Google Play","2020-10-02");
894
895 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
896 VALUES ("50000007","Flow","App World","2021-01-03");
897 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
898 VALUES ("50000007","Instagram","Google Drive","2020-09-21");
899
900 • INSERT INTO descarga (idusuario,nombre_app,nombre_tienda,fecha_descarga)
901 VALUES ("50000008","Flow","App World","2021-01-01");
902
903 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
904 VALUES ("50000009","Instagram","App Store","2021-11-11");
905 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
906 VALUES ("50000009","Google Maps","App Store","2021-09-21");

910 • INSERT INTO descarga (idusuario,nombre_app,nombre_tienda,fecha_descarga)
911 VALUES ("50000009","TikTok","App Store","2021-01-10");
912 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
913 VALUES ("50000011","Instagram","Google Play","2021-01-08");
914 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
915 VALUES ("50000011","Birdees","Google Play","2021-01-07");
916 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
917 VALUES ("50000011","TikTok","Google Play","2021-09-21");
918 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
919 VALUES ("50000012","TikTok","App Store","2021-01-21");
920 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
921 VALUES ("50000012","Instagram","Google Play","2021-01-10");
922 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
923 VALUES ("50000012","Facebook","Google Play","2022-01-08");
924 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
925 VALUES ("50000014","Google Maps","App Store","2022-01-21");
926 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
927 VALUES ("50000016","Instagram","App Store","2022-01-17");
928 • INSERT INTO descarga (idusuario,nombre_app,nombre_tienda,fecha_descarga)
929 VALUES ("50000018","Tencent Music","AppGalley","2021-11-17");
930 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
931 VALUES ("50000018","QQ","AppGalley","2022-01-17");
932 • INSERT INTO descarga(idusuario,nombre_app,nombre_tienda,fecha_descarga)
933 VALUES ("50000018","Cabify","Google Play","2020-08-13");

936 ##### Queries, Triggers y Vistas #####
937
938
939
940 -- 1- Apps gratis en orden de nombre descendiente
941 • select precio, Nombre
942 from app
943 where precio=0
944 order by Nombre desc;
945

```

```

949 -- 2- nombre de los empleados por empresa (en la actualidad)
950
951 • SELECT Nombre, nombreCompleto, CargoActual
952   from empresa inner join empleado
953   on empresa.CodEmpresa=empleado.cod_empresa
954   order by Nombre;
955
961 -- 3- Cuantas Apps tiene cada tienda
962 • select NombreTienda, count(nombre_app) as num
963   from tienda inner join estaDisponible on
964     tienda.NombreTienda=estaDisponible.nombre_tienda
965   group by nombreTienda;
966
972 -- 4- tienda que tiene todas las apps
973 • select NombreTienda, count(nombre_app) as num
974   from tienda inner join estaDisponible on
975     tienda.NombreTienda=estaDisponible.nombre_tienda
976   group by nombreTienda
977   having num= (select count(Nombre) from app);
978
979 ## Vemos que ninguna tienda tiene todas las apps
980
988 -- 5- Que usuario ha descargado el que y
989 -- ordene por nombre de usuario
990 • Select NombreUsuario, nombre_app
991   from usuario inner join descarga on
992     usuario.CuentaUsuario=descarga.idusuario
993   order by usuario.NombreUsuario desc;
994
1003 -- 6- Que usuario ha descargado el que y
1004 -- ordene por nombre de usuario y no de edad y pais
1005 -- del usuario
1006 • Select NombreUsuario , edad, nombre_app, NomPais
1007   from usuario inner join descarga on
1008     usuario.CuentaUsuario=descarga.idusuario
1009   inner join pais on usuario.Codpais=pais.CodPais
1010   order by usuario.NombreUsuario desc;
1011
1018 -- 7- Nombre , Edad ,sexo y nacionalidad de usuario que han descargado la app Birdee
1019 • select NombreUsuario ,edad, sexo, NomPais as "Nacionalidad"
1020   from usuario inner join descarga on
1021     usuario.CuentaUsuario=descarga.idusuario
1022   inner join pais on usuario.Codpais=pais.CodPais
1023   where nombre_app= "Birdees"
1024   order by edad;
1025
1032 -- 8- Numero de descargas por pais
1033 • select NomPais as "Nacionalidad" ,
1034   count(nombre_app) as "Numero Descargas"
1035   from pais inner join usuario on pais.CodPais=usuario.Codpais
1036   inner join descarga on usuario.CuentaUsuario=descarga.idusuario
1037   group by pais.NomPais
1038   order by NomPais ;
1039
1046 -- 9- Crear View con numero de descargas
1047
1048 • DROP VIEW IF EXISTS NumeroDescargas;
1049
1050 • create view NumeroDescargas(nombreApp, numDescargas) as
1051   select Nombre, count(nombre_app)
1052     from app inner join descarga on app.Nombre=descarga.nombre_app
1053   group by nombre_app;
1054
1055 • select * from NumeroDescargas;
1056
1057
1062 -- 10- Las apps con mas descargas
1063 • select nombreApp, numDescargas
1064   from NumeroDescargas
1065   where numDescargas = (select max(numDescargas) from NumeroDescargas);
1066
1067
1076 -- 11 - Que empleados han realizado app "google Maps"
1077
1078 • select nombreCompleto as "Nombre Empleado",
1079   salario as "Salario"
1080   from empleado left join harealizado
1081   on empleado.CodEmpleado=harealizado.codigo_empleado
1082   where harealizado.nombre_App="Google Maps"
1083   order by nombreCompleto;
1084
1090
1091 -- 12- Nombre de apps que empiezan por f
1092 • select Nombre
1093   from app
1094   where Nombre like 'f%';
1095
1105 -- 13- Nombre de empleados que fueron contratados
1106 -- por Alphabet entre 2000 y 2011
1107 • SELECT Nombre , empleado.nombreCompleto, FechaInicio
1108   from empresa inner join Explabo on empresa.CodEmpresa=Explabo.cod_empresa
1109   inner join empleado on Explabo.cod_empleado=empleado.CodEmpleado
1110   where Nombre="Alphabet" and year(FechaInicio) between 2000 and 2011
1111   order by FechaInicio;
1112
1118
1119 -- 14- After Update Trigger: Aumentar sueldo de empleado 10000
1120 delimiter ;
1121 • update empleado set salario = salario + 50000 where CodEmpleado = "100002";
1122
1123
1128
1129 -- 15- Cuantas Descargas han tenido las apps realizadas por
1130 -- cada empleado
1131
1132 • select nombreCompleto, nombreApp, numDescargas
1133   from NumeroDescargas inner join harealizado
1134   on NumeroDescargas.nombreApp= harealizado.nombre_App
1135   inner join empleado on harealizado.codigo_empleado=empleado.CodEmpleado
1136   order by numDescargas desc;
1137
1143 -- 16 - Mes del año en el que mas descargas hubo
1144
1145 • DROP VIEW IF EXISTS Mesdescargas;
1146
1147 • CREATE VIEW Mesdescargas(Mes,numeroDescargas) AS SELECT Month(fecha_descarga), count(nombre_app)
1148   AS totaldescargas FROM
1149   descarga GROUP BY Month(fecha_descarga)
1150   ORDER BY COUNT(nombre_app) DESC limit 3;
1151 • select * from Mesdescargas;
1152

```