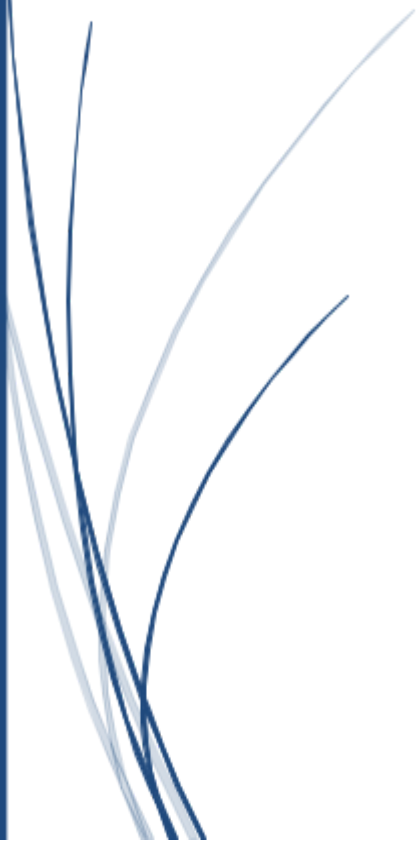
Universitat de les Illes Balears

Sergi Moreno Pérez

28-4-2021

21716 – Base de Dades I. Grup 1



ÍNDEX

Introducció

La cadena de restaurants PizzaCapell volen disposar d'una base de dades per a la gestió de les comandes que fan els clients de la cadena. Aquest document es compon d'un model conceptual i d'un de relacional que ens serviran per a descriure i conformar el que seria una base de dades adequada i eficient pels requeriments que se sol·liciten.

Després d'haver desenvolupat dits models, hi ha una sèrie de comandes en codi SQL per a la creació de la base de dades corresponent.

La conclusió a aquesta primera pràctica es troba a la penúltima fulla del document, just abans dels enllaços empleats per la realització de la pràctica.. En aquest apartat es descriu d'una manera concisa el que he pogut extreure de la pràctica i alguns dels dubtes sorgits a la part inicial del treball.

Capítol I. Model conceptual

➢ Classes

Persona: Client i Repartidor

Feim servir una classe Persona on apareixerà tota aquella informació que compartiran els usuaris que formaran part del sistema especificat, els repartidors i els clients. Així, trobam herència de PERSONA a Client i a Repartidor amb els atributs de nom, adreça i de telèfon. Aquestes dues classes filles també tenen les seves particularitats, no només en temes de funcionalitats, sinó de la informació que se vol emmagatzemar d'elles, on del client volem conèixer el seu email i del repartidor la data del seu contracte. L'herència és {M, XOR}, Mandatory ja que m'he cenyit a l'enunciat i he suposat que només voldríem guardar a la base de dades informació del usuaris prèviament anomenats. En tot cas, seria XOR ja que una persona únicament podria ser un d'ells, no els dos a la vegada.

Zona

La classe Zona ens interessa per distingir de quina zona és un client i poder fer que un dels restaurants es disposi a fer la comanda. De la zona he decidit que s'hagués de guardar el nom de la zona en qüestió.

Restaurant

Serà necessari registrar quin Restaurant és l'encarregat de fer una comanda, que a la vegada designarà a un dels seus repartidors per a què entregui la comanda realitzada. Bastaria en saber el telèfon i l'adreça d'aquest restaurant, per contra el nom del restaurant no l'he volgut incloure perquè, en tractar-se d'un cadena de restaurant, tots els restaurants compartirien el mateix nom. També caldria dir que en un primer moment vaig pensar en un possible atribut que indiqués el nombre de taules existents als restaurants, però vaig acabar per descartar aquesta idea ja que aquest és un sistema basat en l'entrega de comandes a domicili, fent innecessari el manteniment d'aquesta informació.

Comanda: Completa i Incompleta

Per a la comanda faríem servir una classe homònima on trobaríem dades com l'hora en la que ha estat realitzada, el mètode en el que ha estat realitzada, un booleà per saber si ja ha estat pagada o no, un atribut enumerat perquè es pugui conèixer l'estat de la comanda ('en procés', 'entregat') i un nombre de ticket per a identificar de quina comanda es tracta.

L'atribut estat no sabia si finalment incloure-ho, però m'he decantat pel sí perquè he cregut que seria convenient que el repartidor pogués saber l'estat de la comanda que se li ha assignat ja que potser

que la comanda hagi estat pagada en el moment de la sol·licitud, però encara l'haurà d'entregar al client. També queda oberta la possibilitat de que el propi client en pogués comprovar l'estat.

Aquesta classe farà herència en altres dos: Incompleta i Completa. Incompleta serà quan l'entrega de la comanda no hagi pogut estar finalitzada per algun motiu que serà registrat. Per altra banda, Completa només haurà de contenir l'hora en la que s'ha entregat la comanda. Aquesta herència l'he definida com a Optional perquè he pensat que podríem crear un objecte comanda fora que hagi de ser heretat de primeres. Una vegada hagi estat sa comanda entregada o no, ja la podrem heretar en alguna de la seves filles, però només en una d'elles, per això serà herència del tipus XOR.

Producte

Pel que fa a la classe Producte, l'he vista com aquella classe de la que se forma la comanda, és a dir aquells productes que selecciona el client per a compondre la seva comanda. Aquest producte sirà emmagatzemat amb una imatge i una descripció, un atribut enumerat que indica de quin tipus de producte es tracta (pizza, beguda, pasta, entrant, postre), el seu preu i quin és el nombre amb el que apareix a la carta, que és amb el que se li identifica.

Encarrec

Entre la classe Comanda i la classe Producte hi trobam la classe associativa Encarrec que ens servirà per indicar quinaquantitat d'un cert producte ha seleccionat un client per a la seva comanda. Per això, a aquesta classe només he inclòs un atribut per a la quantitat del producte determinat.

Cal destacar que a la classe Comanda no ens fa falta cap atribut amb el preu d'aquesta ja que vindrà determinat per la quantitat del producte a través d'Encarrec i del preu present a Producte. En cas de que s'hagi d'aplicar algun descompte per l'hora d'arribada de l'entrega el podrem saber degut a que tindrem registrat quina ha estat l'hora en la que s'ha realitzat la comanda i en quina hora ha arribat aquesta al domicili del client.

- **Relacions**

R_CLIENT_COMANDA()

Relació associativa que descriu que un client del sistema ha d'haver fet almenys una comanda i en pot fer moltes, mentre que una comanda és d'un sol client.

R_CLIENT_ZONA()

Un client ha de ser d'una zona, però a una zona hi poden viure múltiples clients. Tampoc registrarem cap zona on no hagi algun client vivint.

R_COMANDA_PRODUCTE()

Una comanda està composta, mínim per un producte, i pot ser de molts. D'entre els producte, hi podem trobar algun que no hagi format part de cap comanda pel moment, mentre que d'altres poden haver format part de diverses comandes dels clients.

R_COMANDA_REPARTIDOR()

Una comanda, en el moment que sigui assignada, correspondrà únicament a un repartidor del restaurant. Els repartidors d'un restaurant poden estar registrats fora haver entregat cap comanda i poden haver fet multitud.

R_RESTAURANT_COMANDA()

Bastant similar a la relació anterior, una comanda s'assigna a un sol restaurant i un restaurant pot no haver fet comanda encara i pot cobrir-ne moltes.

R_RESTAURANT_REPARTIDOR()

Els repartidors pertanyeran només a un restaurant, mentre que el restaurant pot no comptar amb cap repartidor o pot tenir-ne molts.

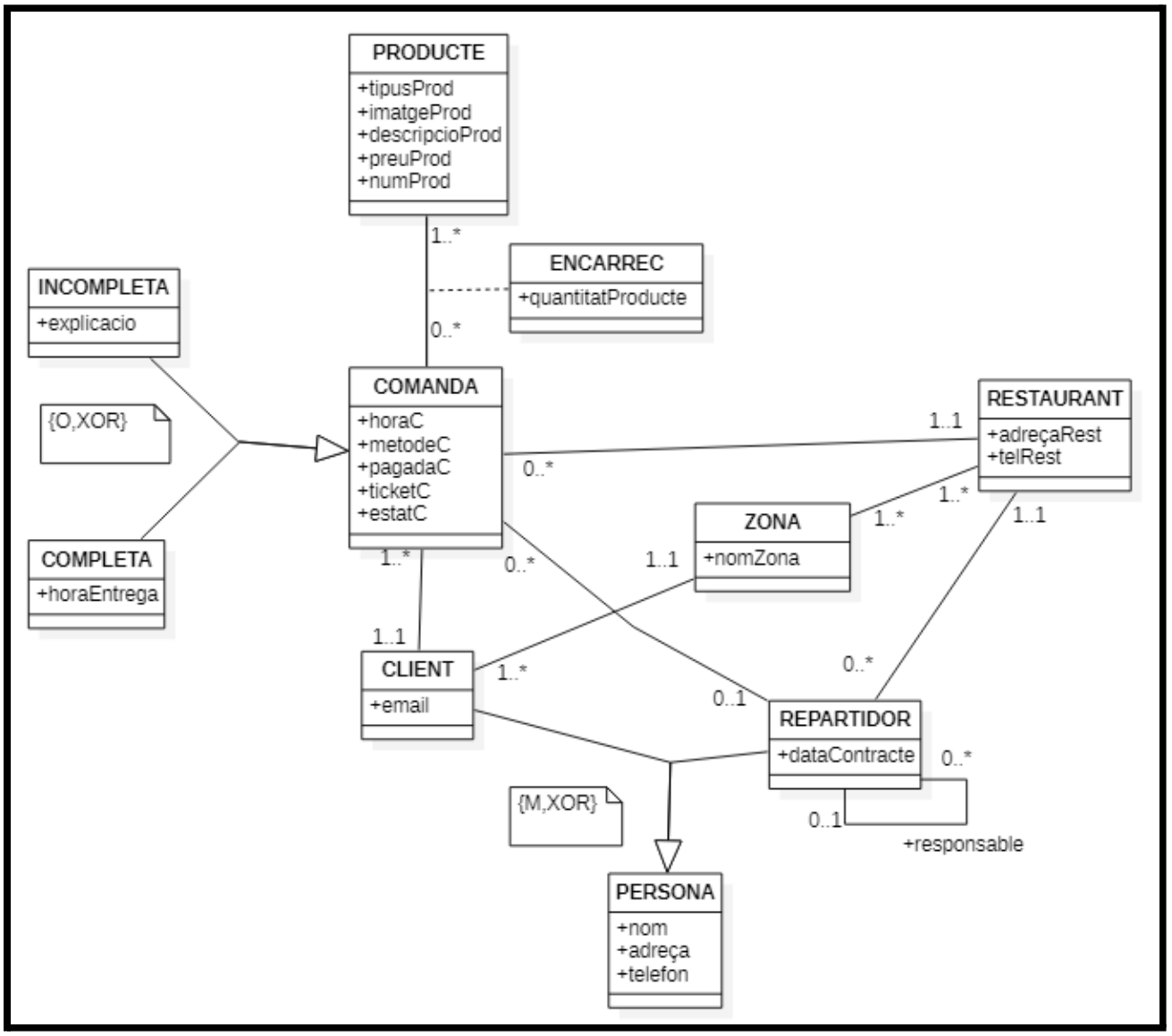
R_RESTAURANT_ZONA()

Una zona podrà ser coberta per diversos restaurants i a la vegada, un restaurant donarà servici a múltiples zones.

R_REPARTIDOR_REPARTIDOR()

Es tracta d'una associació reflexiva on un repartidor pot ser responsable i estar al càrrec de cap o múltiples repartidors i un repartidor pot estar baix el càrrec d'un o cap altre repartidor responsable d'ell.

Imatge del model conceptual de la base de dades



Capítol II. Model relacional

Una vegada hem obtingut el model conceptual, farem servir l'algorisme de traducció per a passar-ho a model relacional. Aquest algorisme consta de 5 passes que seran desenvolupades a continuació.

- **Passes 1 i 2**

La traducció comença amb les passes 1 i 2, on es selecciona una clau principal per cada classe donada del model conceptual i cada entitat d'aquestes esdevé una taula del relacional. Les claus escollides apareixeran subratllades:

Persona(nom, adreça, telefon, idPersona)

Client(email, idClient)

Repartidor(dataContracte, idRep)

Comanda(ticketC, horaC, metodeC, pagadaC, estatC)

Completa(horaEntrega, idCompleta)

Incompleta(explicació, idIncompleta)

Zona(nomZona, idZona)

Restaurant(adreçaRest, telRest, idRest)

Producte(tipusProd, imatgeProd, descripcioProd, preuProd, numProd)

Encarrec(quantitatProducte, idEncarrec)

- **Passes 3 i 4**

Les següents passes són passar de cada relació del conceptual a una taula del model relacional, amb els seus atributs junt amb les claus de les classes que relaciona; i seleccionar una clau principal per a cada una d'aquestes noves relacions.

R_CLIENT_COMANDA(DNIclient, ticketC)

R_CLIENT_ZONA(idClient, idZona)

R_COMANDA_PRODUCTE(ticketC, numProd)

R_COMANDA_REPARTIDOR(ticketC, idRep)

R_RESTAURANT_COMANDA(idRest, ticketC)

R_RESTAURANT_REPARTIDOR(idRest, idRep)

R_RESTAURANT_ZONA(idRest, idZona)

R_REPARTIDOR_REPARTIDOR(idRep, idResponsable)

En aquesta darrera relació, idResponsable és el mateix que idRep.

- **Passa 5**

Finalment, la darrera passa és estudiar la possibilitat de reduir el nombre de taules resultants a través de la fusió d'algunes d'elles. Les candidates són les que tenen la mateixa clau. Tot i això, s'ha de tenir en conte el significat i la conveniència de continuar amb dita fusió.

Les relacions Zona, Restaurant i Producte es mantenen igual:

- Zona(nomZona, idZona)
- Restaurant(adreçaRest, telRest, idRest)
- Producte(tipusProd, imatgeProd, descripcioProd, preuProd, numProd)

A la relació Encarrec substituïm la seva clau primària idEncarrec per les claus primàries de Comanda i de Producte, ticketC i idProd respectivament.

- Encarrec(quantitatProducte, ticketC, idProd)

En el cas de l'herència Persona, en ser del tipus {M,XOR} el més avantatjós és fusionar la classe mare amb les seves classes filles, per separat en ser del tipus XOR; així aconseguim estalviar l'emmagatzematge d'informació redundant.

- PersonaClient(nom, adreça, telefon, email, idClient, idZona)
  idZona: Foreign Key a Zona(idZona)

- PersonaRepartidor(nom, adreça, telefon, dataContracte, idRep, idResponsable, idRest)
  idResponsable: Foreign Key a PersonaRepartidor(idRep)
  idRest: Foreign Key a Restaurant(idRest)

Pel cas de l'altre herència {O,XOR} present a la base de dades, es perd espai en qualsevol cas, no es pot fusionar; mantindrem la classe mare i les seves classes filles.

- Comanda(ticketC, horaC, metodeC, pagadaC, estatC, idClient, idRep, idRest)
  idClient: Foreign Key a PersonaClient(idClient)
  idRep: Foreign Key a PersonaRepartidor(idRep)
  idRest: Foreign Key a Restaurant(idRest)

- Completa(horaEntrega, ticketC)
  ticketC: Foreign Key a Comanda(ticketC)

- Incompleta(explicació, ticketC)
  ticketC: Foreign Key a Comanda(ticketC)

De les taules Restaurant i Zona es manté la taula de la seva relació ja que és tracta d'una relació de molts a molts. En ser així, aquesta taula mantindrà una idRest i idZona com a Primary Keys.

- R_RESTAURANT_ZONA(idRest, idZona)

Capítol III. Codi SQL

Per generar la base de dades desenvolupada es fan servir les següents línies de codi SQL.

He intentat generar les taules, amb els seus atributs, les seves claus principals i foranes; amb una única instrucció. Fins a la creació de la taula *producte* no vaig ser conscient de descriure com a NOT NULL alguns del atributs especificats, en ser necessàriament atributs que s'han de conèixer. Així, una vegada vaig haver creat totes les taules necessàries per a cada relació final, extretes del model relacional, vaig fer servir la instrucció ALTER TABLE per modificar aquells valors que fossin adients de les dues primeres taules generades. Per les claus primàries i foranes, ve implícit que són atributs *not null*.

Pel cas del nombre de caràcters dels atributs VARCHAR o del nombre d'enters per atributs *int*, m'he fet servir de valors estimats en funció del que volia representar cada atribut. Per exemple, en tractar-se de ids m'he fet servir del tipus int amb 4 dígits.

Creació de la base de dades:

```
CREATE DATABASE pizzahut
```

Un cop creada, ens diposam crear les taules per les diferents relacions. L'ordre seguit no és aleatori, ja que, com s'ha descrit anteriorment, n'hi ha algunes que tenen claus foranes que fan referència a altres taules, pel que no seria possible ni adequat voler generar una taula amb una clau forana apuntant a una altra, si aquesta segona encara no s'ha implementat.

Les 3 primeres taules que generam són aquelles que no conten amb cap Foreign Key:

```
CREATE TABLE zona (idZona int(4) PRIMARY KEY, nomZona VARCHAR(20))
```

```
CREATE TABLE restaurant (idRest int(4) PRIMARY KEY, adreçaRest VARCHAR(20), telRest int(9))
```

Per crear la taula *producte* he cercat per internet informació sobre atributs que fossin imatges i he trobat que es defineixen com a VARBINARY(MAX), on MAX és el tamany de la imatge (enllaç al final del document).

```
CREATE TABLE producte (
        numProd int(2) PRIMARY KEY,
        tipusProd ENUM('entrant', 'pizza', 'pasta', 'postre', 'beguda') NOT NULL,
        imatgeProd VARBINARY(100),
        descripcioProd VARCHAR(50),
        preuProd float(5))
```

A partir d'aquest punt, ens trobam amb relacions amb referències. No generarem una taula si no tenim totes les seves claus foranes apuntant a taules ja existents.

```
CREATE TABLE persona_client (
        nom VARCHAR(20) NOT NULL,
        adreça VARCHAR(20) NOT NULL,
        telefon int(9),
        email VARCHAR(25),
        idClient int(4) PRIMARY KEY,
        idZona int(4) NOT NULL,
        FOREIGN KEY (idZona) REFERENCES zona(idZona))

CREATE TABLE persona_repartidor (
        nom VARCHAR(20) NOT NULL,
        adreça VARCHAR(20) NOT NULL,
        telefon int(9),
        dataContracte date NOT NULL,
        idRep int(4) PRIMARY KEY,
        idResp int(4),
        idRest int(4),
        FOREIGN KEY (idResp) REFERENCES persona_repartidor(idRep),
        FOREIGN KEY (idRest) REFERENCES restaurant(idRest))

CREATE TABLE comanda (
        ticketC int(3) PRIMARY KEY,
        horaC time NOT NULL,
        metodeC VARCHAR(20) NOT NULL,
        pagadaC boolean NOT NULL,
        estatC ENUM('en procés', 'entregat') NOT NULL,
        idClient int(4) NOT NULL,
        idRep int(4),
        idRest int(4) NOT NULL,
        FOREIGN KEY (idClient) REFERENCES persona_client(idClient),
        FOREIGN KEY (idRep) REFERENCES persona_repartidor(idRep),
```

```
        FOREIGN KEY (idRest) REFERENCES restaurant(idRest))

CREATE TABLE completa (
        ticketC int(3) PRIMARY KEY,
        horaEntrega time NOT NULL,
        FOREIGN KEY (ticketC) REFERENCES comanda(ticketC))

CREATE TABLE incompleta (
        ticketC int(3) PRIMARY KEY,
        explicacio VARCHAR(50) NOT NULL,
        FOREIGN KEY (ticketC) REFERENCES comanda(ticketC))

CREATE TABLE r_restaurant_zona (idRest int (4), idZona int(4), PRIMARY KEY (idRest, idZona))

CREATE TABLE encarrec (
        quantitatProducte int (3) NOT NULL,
        ticketC int(4),
        idProd int(4),
        PRIMARY KEY (ticketC,idProd))
```

Modificam atributs a NOT NULL a les dos primeres taules que s'han generat a la base de dades. Després de que m'aparegués un error de sintaxis intentant fer un ALTER TABLE _ ALTER COLUMN _ vaig decidir eliminar els atributs en qüestió (ja que no afectaven en res a les relacions amb les altres taules) i vaig tornar a afegir-los, aquest pic amb la restricció de no poder ser atributs amb valor *null*.

- Taula zona

```
ALTER TABLE zona DROP COLUMN nomZona
ALTER TABLE zona ADD COLUMN nomZona VARCHAR(20) NOT NULL
```

- Taula restaurant

```
ALTER TABLE restaurant DROP COLUMN adreçaRest AND telRest
ALTER TABLE restaurant DROP COLUMN adreçaRest AND telRest
ALTER TABLE restaurant ADD COLUMN adreçaRest VARCHAR(20) NOT NULL
ALTER TABLE restaurant ADD COLUMN telRest int(9) NOT NULL
```

Un cop he finalitzat la implementació de la base de dades per a la franquícia PizzaCapell, he vist adequat exportar la base de dades i com només havíem d'entregar un sol document, he afegit íntegrament el contingut de l'arxiu a dintre d'aquest. (*Extensió de les **8 següents pàgines***)

Contingut de l'arxiu pizzahut.sql

-- phpMyAdmin SQL Dump

-- version 5.0.4

-- https://www.phpmyadmin.net/

--

-- Servidor: 127.0.0.1

-- Tiempo de generación: 27-04-2021 a las 18:55:04

-- Versión del servidor: 10.4.17-MariaDB

-- Versión de PHP: 8.0.2

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";

START TRANSACTION;

SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;

/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;

/*!40101 SET NAMES utf8mb4 */;

--

-- Base de datos: `pizzahut`

--

-- --

--

-- Estructura de tabla para la tabla `comanda`

--

CREATE TABLE `comanda` (

```
  `ticketC` int(3) NOT NULL,
  `horaC` time NOT NULL,
  `metodeC` varchar(20) NOT NULL,
  `pagadaC` tinyint(1) NOT NULL,
  `estatC` enum('en procés','entregat') NOT NULL,
  `idClient` int(4) NOT NULL,
  `idRep` int(4) DEFAULT NULL,
  `idRest` int(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `completa`
--

CREATE TABLE `completa` (
  `ticketC` int(3) NOT NULL,
  `horaEntrega` time NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `encarrec`
--

CREATE TABLE `encarrec` (
  `quantitatProducte` int(3) NOT NULL,
  `ticketC` int(4) NOT NULL,
  `idProd` int(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------
```

```
--
-- Estructura de tabla para la tabla `incompleta`
--

CREATE TABLE `incompleta` (
  `ticketC` int(3) NOT NULL,
  `explicacio` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `persona_client`
--

CREATE TABLE `persona_client` (
  `nom` varchar(20) NOT NULL,
  `adreça` varchar(20) NOT NULL,
  `telefon` int(9) DEFAULT NULL,
  `email` varchar(25) DEFAULT NULL,
  `idClient` int(4) NOT NULL,
  `idZona` int(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `persona_repartidor`
--

CREATE TABLE `persona_repartidor` (
  `nom` varchar(20) NOT NULL,
  `adreça` varchar(20) NOT NULL,
```

```
  `telefon` int(9) DEFAULT NULL,
  `dataContracte` date NOT NULL,
  `idRep` int(4) NOT NULL,
  `idResp` int(4) DEFAULT NULL,
  `idRest` int(4) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `producte`
--

CREATE TABLE `producte` (
  `numProd` int(2) NOT NULL,
  `tipusProd` enum('entrant','pizza','pasta','postre','beguda') NOT NULL,
  `imatgeProd` varbinary(100) DEFAULT NULL,
  `descripcioProd` varchar(50) DEFAULT NULL,
  `preuProd` float DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `restaurant`
--

CREATE TABLE `restaurant` (
  `idRest` int(4) NOT NULL,
  `adreçaRest` varchar(20) NOT NULL,
  `telRest` int(9) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------
```

```
--
-- Estructura de tabla para la tabla `r_restaurant_zona`
--

CREATE TABLE `r_restaurant_zona` (
  `idRest` int(4) NOT NULL,
  `idZona` int(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `zona`
--

CREATE TABLE `zona` (
  `idZona` int(4) NOT NULL,
  `nomZona` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Índices para tablas volcadas
--

--
-- Indices de la tabla `comanda`
--
ALTER TABLE `comanda`
  ADD PRIMARY KEY (`ticketC`),
  ADD KEY `idClient` (`idClient`),
  ADD KEY `idRep` (`idRep`),
  ADD KEY `idRest` (`idRest`);
```

```
--
-- Indices de la tabla `completa`
--
ALTER TABLE `completa`
  ADD PRIMARY KEY (`ticketC`);

--
-- Indices de la tabla `encarrec`
--
ALTER TABLE `encarrec`
  ADD PRIMARY KEY (`ticketC`,`idProd`);

--
-- Indices de la tabla `incompleta`
--
ALTER TABLE `incompleta`
  ADD PRIMARY KEY (`ticketC`);

--
-- Indices de la tabla `persona_client`
--
ALTER TABLE `persona_client`
  ADD PRIMARY KEY (`idClient`),
  ADD KEY `idZona` (`idZona`);

--
-- Indices de la tabla `persona_repartidor`
--
ALTER TABLE `persona_repartidor`
  ADD PRIMARY KEY (`idRep`),
  ADD KEY `idResp` (`idResp`),
  ADD KEY `idRest` (`idRest`);

--
```

```
-- Indices de la tabla `producte`
--
ALTER TABLE `producte`
 ADD PRIMARY KEY (`numProd`);


--
-- Indices de la tabla `restaurant`
--
ALTER TABLE `restaurant`
 ADD PRIMARY KEY (`idRest`);


--
-- Indices de la tabla `r_restaurant_zona`
--
ALTER TABLE `r_restaurant_zona`
 ADD PRIMARY KEY (`idRest`,`idZona`);


--
-- Indices de la tabla `zona`
--
ALTER TABLE `zona`
 ADD PRIMARY KEY (`idZona`);


--
-- Restricciones para tablas volcadas
--


--
-- Filtros para la tabla `comanda`
--
ALTER TABLE `comanda`
 ADD CONSTRAINT `comanda_ibfk_1` FOREIGN KEY (`idClient`) REFERENCES `persona_client`
(`idClient`),
 ADD CONSTRAINT `comanda_ibfk_2` FOREIGN KEY (`idRep`) REFERENCES `persona_repartidor`
(`idRep`),
```

```
 ADD CONSTRAINT `comanda_ibfk_3` FOREIGN KEY (`idRest`) REFERENCES `restaurant` (`idRest`);


--
-- Filtros para la tabla `completa`
--
ALTER TABLE `completa`
 ADD CONSTRAINT `completa_ibfk_1` FOREIGN KEY (`ticketC`) REFERENCES `comanda` (`ticketC`);


--
-- Filtros para la tabla `incompleta`
--
ALTER TABLE `incompleta`
 ADD CONSTRAINT `incompleta_ibfk_1` FOREIGN KEY (`ticketC`) REFERENCES `comanda`
(`ticketC`);


--
-- Filtros para la tabla `persona_client`
--
ALTER TABLE `persona_client`
 ADD CONSTRAINT `persona_client_ibfk_1` FOREIGN KEY (`idZona`) REFERENCES `zona` (`idZona`);


--
-- Filtros para la tabla `persona_repartidor`
--
ALTER TABLE `persona_repartidor`
 ADD CONSTRAINT `persona_repartidor_ibfk_1` FOREIGN KEY (`idResp`) REFERENCES
`persona_repartidor` (`idRep`),
 ADD CONSTRAINT `persona_repartidor_ibfk_2` FOREIGN KEY (`idRest`) REFERENCES `restaurant`
(`idRest`);
COMMIT;


/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Final de l'arxiu pizzahut.sql

Conclusions

La realització d'aquesta pràctica m'ha estat més complicada del que vaig pensar en un primer moment. Havia estat practicant amb els exercicis de classe i alguns del voluntaris i, encara que cometia alguns errors, pensava que amb l'experiència que duia em seria suficient per acabar la pràctica fàcilment.

Quan vaig llegir l'enunciat per primer pic vaig sentir que pot ser la pràctica que duia pel moment no fos suficient. Després de llegir-lo atentament vàries vegades i de separar-ho per funcionalitats i anotant la informació que volia registrar l'empresa de cada entitat del seu sistema, vaig poder anar estructurant-me mentalment i vaig començar a dissenyar el model conceptual que em semblava adequat per complir amb el que es demanava. Així va ser com, una vegada superat l'impacte inicial d'un enunciat llarg on es requereixen bastantes coses, l'experiència que hagués pogut extreure dels exercicis realitzats i del que vaig aprendre dels exemples fets a classe, a poc a poc vaig anar completant els diversos capítols, polint detalls i aclarint els dubtes que em sorgissin al procés.

A l'hora de començar el semestre, no tenia cap tipus de coneixement de com funcionava una base de dades ni de com es podia crear. Gràcies a haver-me enfrontat a aquesta pràctica, he pogut acabar de comprendre conceptes que pensava assolits als exercicis de classe, però dels que havia detalls que no acabava d'entendre.

Enllaços

Forum de microsoft amb la informació extreta per declarar l'atribut *imatge* de la taula *producte*:

https://social.technet.microsoft.com/Forums/es-ES/8f949660-9a6a-4d03-ad18-d5302f154ac1/insertar-una-imagen-sql?forum=sqlserveres