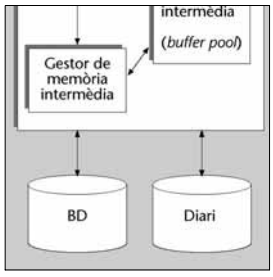
Evolució dels models de bases de dades i arquitectura

funcional d'un SGBD

Ramon Segret i Sala
Jaume Sistac i Planas

P01/11031/00002

Universitat Oberta de Catalunya



Índex

Introducció

Aquest mòdul de l'assignatura *Sistemes de gestió de bases de dades* introdueix la resta de continguts i ho fa en dos apartats.

D'acord amb l'enfocament general de l'assignatura, que és la compleció dels coneixements fonamentals del camp de les bases de dades i dels seus programaris específics –els gestors de bases de dades– que s'han estudiat en altres assignatures, aquest mòdul té en compte les matèries estudiades allà i en representa l'arrodoniment.

Recordeu els coneixements fonamentals sobre bases de dades en les assignatures *Bases de dades I* i *Bases de dades II*.

En el primer apartat d'aquest mòdul, "Evolució dels models de bases de dades", s'exposa esquemàticament el desenvolupament del camp de les bases de dades des dels seus orígens, els fitxers, fins a l'actualitat, els SGBD que incorporen característiques relacionals i orientades a l'objecte, prenent com a fil conductor d'aquesta evolució els diferents models conceptuals que l'han marcat. D'aquesta manera es completen els coneixements d'introducció a les bases de dades i als sistemes de gestió de bases de dades que ja coneixeu d'altres assignatures, i se'ls dóna una dimensió històrica.

Abreugem *sistema de gestió de bases de dades* amb la sigla SGBD.

Vegeu els mòduls "Introducció a les bases de dades" de l'assignatura *Bases de dades I* i "Introducció als sistemes de gestió de bases de dades" de l'assignatura *Bases de dades II*.

El segon apartat, "Arquitectura funcional d'un SGBD", és més curt i fa de pont cap a la resta de l'assignatura, que se centra en el programari que implementa els models que s'han vist. Aquest programari –anomenat SGBD– s'estudia en aquesta assignatura: es passa revista als diferents components, i s'estén en aquells que, o bé encara no s'han vist en les assignatures anteriors, o bé no s'han vist amb prou profunditat. Per a identificar aquests darrers seguirem, també esquemàticament i des del punt de vista de la funcionalitat, el processament d'una sentència SQL a través d'un SGBD relacional.

Objectius

Amb els materials didàctics associats a aquest mòdul, assolireu els objectius següents:

1. Conèixer els diferents models de bases de dades, encara que sigui d'una manera esquemàtica.

2. Comprendre què aporta cada model en l'evolució tècnica del camp de les bases de dades.

3. Adquirir una visió històrica i contextualitzada de l'evolució esmentada.

4. Conèixer el panorama actual de les bases de dades i dels principals SGBD del mercat.

5. Entendre l'estructura funcional d'un SGBD relacional per a poder-hi situar cada mòdul d'aquesta assignatura i veure la relació que tenen entre si.

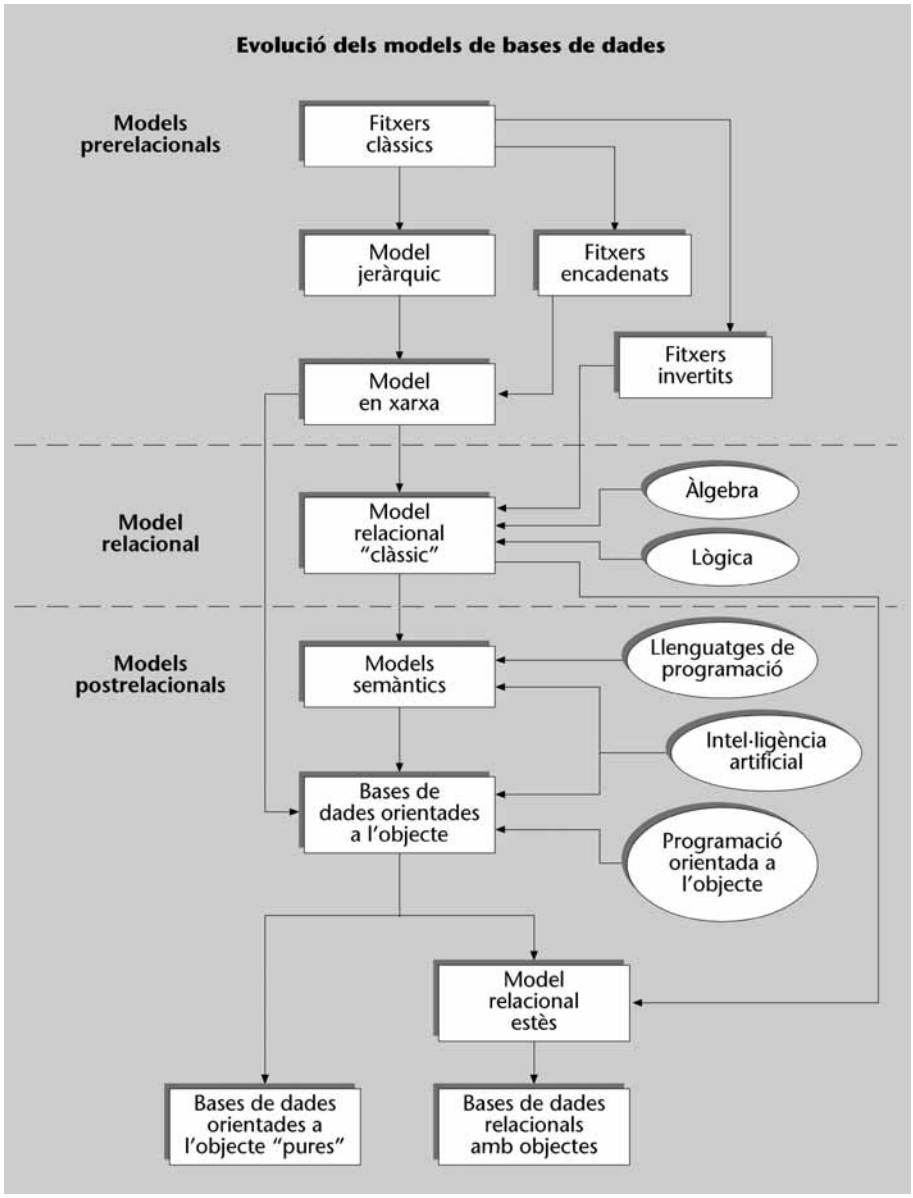
1. Evolució dels models de bases de dades

En aquest apartat exposem la successió dels diferents mecanismes conceptuals que s'han fet servir per a modelitzar l'emmagatzematge de dades en medis no volàtils al llarg dels aproximadament cinquanta anys d'existència de la tecnologia informàtica.

Seguirem la línia principal d'evolució d'aquests mecanismes conceptuals, anomenats **models de bases de dades**, prescindint d'endinsar-nos massa en ramificacions col·laterals, per a poder proporcionar una visió més entenedora d'aquest fet, encara que sigui simplificada.

La figura 1 ens ajudarà a seguir l'explicació d'aquest primer apartat.

Figura 1



1.1. Models prerelacionals

Els primers models abans del model relacional són els fitxers clàssics, el model jeràrquic i els models en xarxa.

1.1.1. Els fitxers clàssics

Els primers mecanismes emprats van ser els fitxers, a vegades qualificats amb l'adjectiu *clàssics* per a distingir-los de desenvolupaments posteriors que van donar lloc a altres fitxers amb més funcionalitats que les disponibles en un principi*.

*** Per exemple, fitxers encadenats i fitxers invertits, dels quals parlem en aquest mateix mòdul.**

Un fitxer és un contenidor que emmagatzema dades, estructurades seguint unes regles relativament senzilles, que podem resumir de la manera següent:

- Un fitxer emmagatzema les dades que corresponen a totes les instàncies d'una determinada entitat.

Instància = ocurrència

- El fitxer és una col·lecció de registres, un per cada instància de l'entitat.
- El registre és una col·lecció de camps, un per cada atribut de l'entitat.

Exemples de fitxers més complexos

Les regles esmentades en aquest apartat no són, en absolut, rígides i trobem fitxers amb estructures força més complexes, com en els casos següents:

- Un fitxer pot emmagatzemar més d'un tipus d'entitat. Per exemple, un fitxer de factures, amb les capçaleres i les línies de detall.
- Un camp es pot repetir dins d'un registre. Per exemple, el fitxer del personal d'una empresa on es guarden, entre altres dades, els noms dels fills dels empleats. Com que un empleat determinat pot tenir de zero a *n* fills, el camp que correspon a l'atribut fill en el registre d'aquest empleat pot o bé no existir, o estar repetit *n* vegades.

Model de fitxers?

En el cas dels fitxers, les regles no són rígides; per això normalment no es parla encara de *model*.

En principi, els **fitxers clàssics** no proporcionen cap programari especialitzat d'accés; per tant, tota la programació necessària per a fer servir les dades que resideixen en un fitxer ha d'estar continguda en cada un dels programes que hi accedeixen.

Aquest és un requeriment fonamental dels fitxers i evidentment introdueix complexitat al sistema. La complexitat no procedeix tant de l'estructura del fitxer, que ja hem vist que és relativament senzilla, com del fet d'haver de conèixer exactament la implementació física de cada fitxer concret. El programa ha de conèixer l'estructura del fitxer per a entendre'l i poder-lo utilitzar.

Concretament, els programes han de poder localitzar registres d'un fitxer i després han de saber interpretar els diferents camps dels registres. El primer dels requeriments està relacionat amb l'organització del fitxer, i el segon, amb l'estructura del registre.

Les organitzacions dels fitxers clàssics

Les organitzacions dels fitxers clàssics són de dos tipus:

- Seqüencial: als fitxers organitzats seqüencialment, els registres es troben consecutius, en el mateix ordre en què han estat afegits al fitxer. Els programes localitzen un registre llegint-los tots, un darrere l'altre, fins que troben el que volen.
- Directa: als fitxers amb organització directa s'estableix una correspondència entre cada registre –normalment entre un camp determinat del registre– i la seva posició al fitxer. Els programes calculen la posició del registre que volen localitzar a partir d'un valor, que és el contingut del camp que s'ha definit per a calcular la correspondència.

Cal destacar que, malgrat la possible existència d'un camp que ens serveix per a localitzar registres, aquest concepte no és encara el de clau primària que trobarem més endavant en altres models. Li manquen, fonamentalment, els elements formals que caracteritzen aquell concepte.

L'estructura del registre

Respecte a l'estructura del registre assenyalarem que, en molts casos, quan es tracta d'estructures senzilles –el registre té sempre el mateix nombre de camps–, llavors als programes que les tracten els cal només conèixer l'ordre i característiques de cada camp dins del registre.

Però si l'estructura és més complexa, com en el cas de camps de longitud variable, de camps repetitius, de camps opcionals, de registres de diferents tipus i altres de similars, llavors calen una sèrie de convencions per a implementar totes aquestes característiques més complexes i, en conseqüència, els programes les han de conèixer per a poder localitzar correctament la dada buscada dins l'estructura del registre. Aquestes convencions poden ser uns separadors que indiquin quan comença un determinat registre o un camp, o un comptador que indiqui quantes ocurrències d'un camp hi ha a continuació, o una anotació de la longitud d'un camp concret.

Com a resum podem dir que els fitxers exigeixen que els programes que els tracten en coneguin totes les característiques d'implementació i que aquestes característiques són de caire físic.

1.1.2. El model jeràrquic

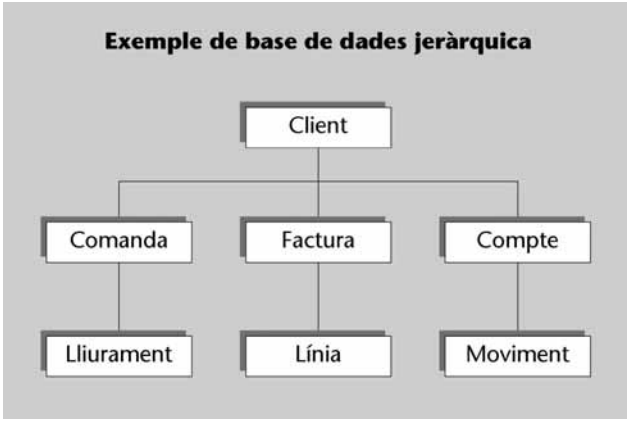
Els programes i aplicacions informàtiques que requerien fitxers senzills els van utilitzar durant molt de temps. Però quan es requerien fitxers complexos, la mateixa complexitat va portar a encapsular el tractament del fitxer en un programari especialitzat, separat dels programes d'aplicació. D'aquesta manera van aparèixer els programaris de gestió de fitxers que, en anar gestionant cada vegada més funcionalitats, serien els nuclis i orígens dels sistemes de gestió de bases de dades (SGBD).

Un d'aquests productes, l'IDS, construït per Charles Bachman a General Electric el 1964, està considerat per molts com el primer SGBD, però és un altre el que realment va ser el primer SGBD comercial amb incidència real al mercat. Es tracta de l'IMS, d'IBM, que entre 1966 i 1969 es va configurar com a SGBD generalitzat, des de les primeres aplicacions en el món de l'aviació, per a les quals va néixer, fins a l'ús massiu en els grans bancs d'arreu del món.

L'IMS, a més de ser un programari específic que incorporava moltes de les funcionalitats que es consideren definitòries d'un SGBD, suportava un model de dades més ric que el dels fitxers. Aquest model estructura les dades en forma de jerarquies i per això s'anomena **model jeràrquic**, del qual l'IMS és pràcticament l'únic producte de mercat que ha existit.

Veurem les característiques del model jeràrquic amb l'ajuda de l'exemple il·lustrat en la figura 2. Es tracta d'una base de dades per a guardar les diferents informacions relacionades amb els clients d'una empresa de distribució.

Figura 2



Les **característiques estructurals més importants de l'IMS**, i per tant del model jeràrquic, són les següents:

- El contenidor que emmagatzema les dades s'anomena *base de dades*. Una base de dades conté les dades que corresponen a totes les instàncies d'un conjunt d'entitats relacionades.

- Les entitats relacionades d'una base de dades s'estructuren entre si de manera que formen una jerarquia o arbre. El node arrel correspon a l'entitat principal i els nodes dependents corresponen a les entitats subordinades.

- La base de dades és una col·lecció d'arbres, un per cada instància de l'entitat principal.

- Cada instància d'arbre conté una instància de l'entitat principal i totes les instàncies de les entitats dependents que corresponen a aquesta instància d'entitat principal. La manera natural de recórrer una instància d'arbre és el preordre.

- Cada node de l'arbre és una col·lecció de camps, un per cada atribut de l'entitat que aquest node emmagatzema.

Aquestes característiques del model jeràrquic són les que el programador ha de conèixer per a poder fer-lo servir. Els detalls tècnics de la implementació física d'aquests arbres s'encapsulen dins l'IMS (que en allò que ateny a aquesta característica ja és un veritable SGBD) i, per tant, no són visibles als usuaris de les bases de dades.

Precisament per a gestionar la implementació física de les bases de dades jeràrquiques i decidir entre diferents opcions quan això sigui possible, apareix una figura nova: l'administrador de bases de dades (ABD). L'ABD coneix a fons l'SGBD i suporta les necessitats de programadors i usuaris de les bases de dades.

Abreugem *administrador de bases de dades* amb la sigla ABD.

De manera similar al cas dels fitxers, també en el model jeràrquic els programes han de poder localitzar arbres concrets d'una base de dades i després els han de saber recórrer per a trobar els camps que necessiten. El primer dels requeriments està relacionat amb l'organització de la base de dades i el segon amb l'estructura de l'arbre.

Les organitzacions i l'estructura d'arbre de les bases de dades jeràrquiques

Les **organitzacions** de les bases de dades jeràrquiques són similars a les que ja hem vist per als fitxers. Aquí interessa localitzar el node arrel de cada arbre i, depenent de la implementació de la base de dades, això es fa seqüencialment, arbre darrere arbre, o bé d'una manera directa establint una correspondència entre un camp d'aquest node arrel i la seva posició en la base de dades.

Respecte a l'**estructura de l'arbre,** l'IMS permet diferents maneres de recórrer els nodes d'un arbre concret, segons les necessitats de cada programa. Per a possibilitar aquesta funcionalitat empra, normalment, apuntadors físics.

Fixem-nos que l'aparició del model jeràrquic va representar un avenç per les raons següent:

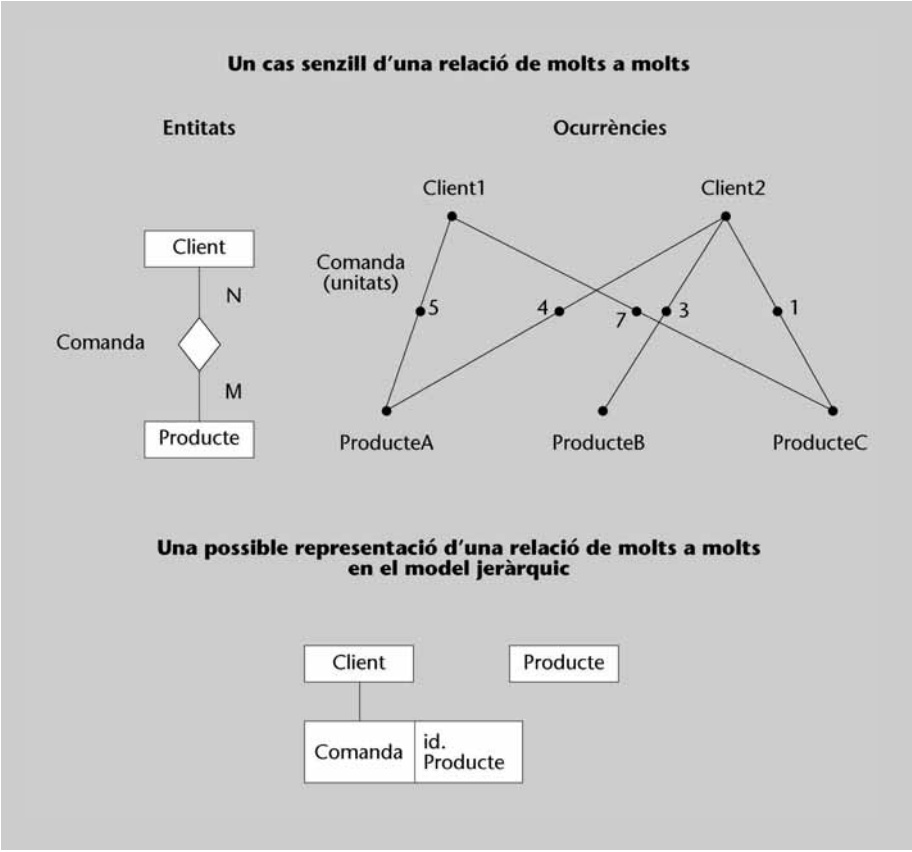
a) El suportava un veritable SGBD, que proporcionava un conjunt de funcionalitats noves per comparació amb els fitxers.

b) Hi ha una separació de nivells: un nivell extern, el del model jeràrquic, per a ús dels programadors, i un nivell intern, el de la implementació, gestionat per l'ABD.

1.1.3. Els models en xarxa

El model jeràrquic no modelitza totes les possibilitats. Fins i tot casos tan corrents com clients i productes, relacionats per les comandes, (vegeu la figura 3) no es poden representar com una sola jerarquia. Un client normalment compra més d'un producte i un producte és comprat per més d'un client. Ens trobem davant d'una relació de molts a molts entre dues entitats i això només es pot representar en el model jeràrquic utilitzant més d'un arbre i introduint una certa redundància entre si, tal com il·lustra la part inferior de la figura 3.

Figura 3

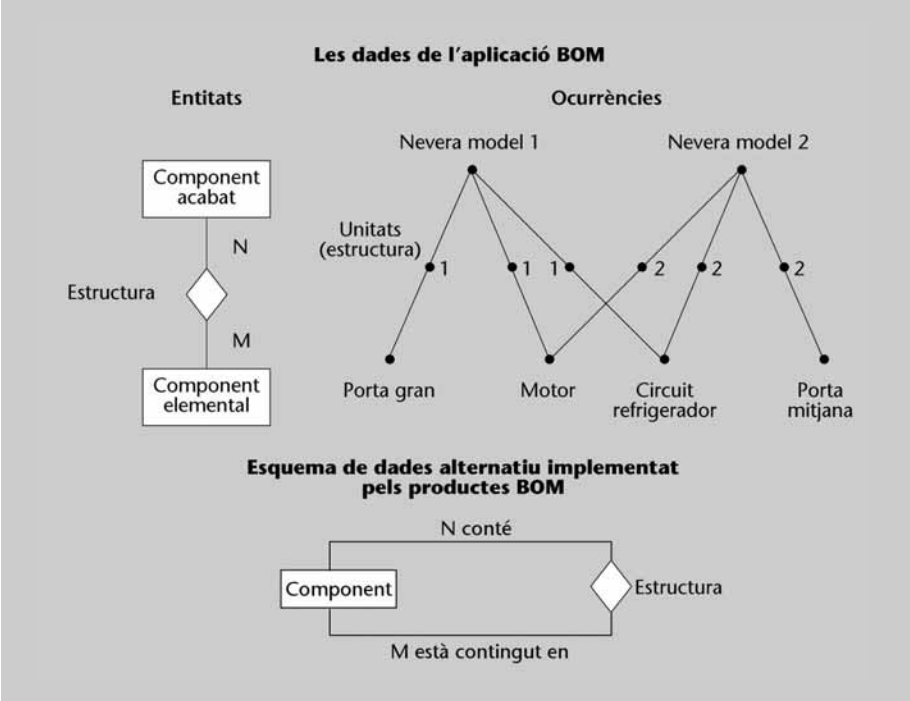


Ja el mateix IMS es va ampliar per a suportar casos com aquest a base de relacionar entre si arbres diferents. Però no va passar de ser un afegit al model jeràrquic que en traïa la senzillesa original.

Contemporàniament a l'aparició del model jeràrquic s'havia estudiat com proporcionar un sistema d'emmagatzematge adequat per a una aplicació molt important per a les empreses de fabricació. Ens referim a la que es va conèixer amb la sigla BOM, de l'anglès *Bill of Materials*. Es tractava de disposar de la informació relativa a tots els materials, des dels components elementals fins als productes acabats, i de com uns participaven en el muntatge dels altres. Una base de dades d'aquesta índole permetia planificar la producció.

El model de dades del BOM tampoc no és una jerarquia, ja que es tracta també d'una relació de molts a molts entre dues entitats, tal com mostra la figura 4.

Figura 4



Un esquema de dades alternatiu per a suportar el BOM és el constituït pels dos elements *Component* i *Estructura*. *Component* aplega tots els components, siguin acabats o elementals, i *Estructura* guarda la informació de com un component entra a formar part d'un altre.

Precisament el producte BOMP, d'IBM, que va sortir al mercat per a mecanitzar l'aplicació BOM, implementava aquest darrer esquema de dades mitjançant dos fitxers encadenats. Un dels fitxers, anomenat *mestre*, contenia les instàncies de *Component*, i l'altre, les d'*Estructura*. I s'anomenaven *encadenats* perquè un conjunt d'apuntadors físics entre registres dels dos fitxers permetia seguir cadenes que saltaven successivament d'un a l'altre.

Generalització

Posteriorment el model de fitxers encadenats es va utilitzar per a altres tipus d'aplicacions. La mateixa IBM va treure un producte nou, el DBOMP, amb aquesta vocació generalista, però va ser un altre, el TOTAL, de Cincom Systems, el que es va imposar al mercat i hi va perdurar força temps.

Encara que els fitxers encadenats resolien l'emmagatzematge de dues entitats relacionades de molts a molts, és a dir, d'una xarxa senzilla, no són aquests productes els que després s'han conegut com els SGBD d'un model de bases de dades nou, el model en xarxa.

El model en xarxa té les seves arrels en els treballs de Bachman i l'IDS, i el seu desenvolupament en el treball d'un comitè, el Data Base Task Group (DBTG) del CODASYL. En successives publicacions, el 1971, 1973 i 1978, la més important de les quals és la del 1973, el DBTG publicà les especificacions d'un model de bases de dades que volia superar les limitacions del model jeràrquic i esdevenir un estàndard.

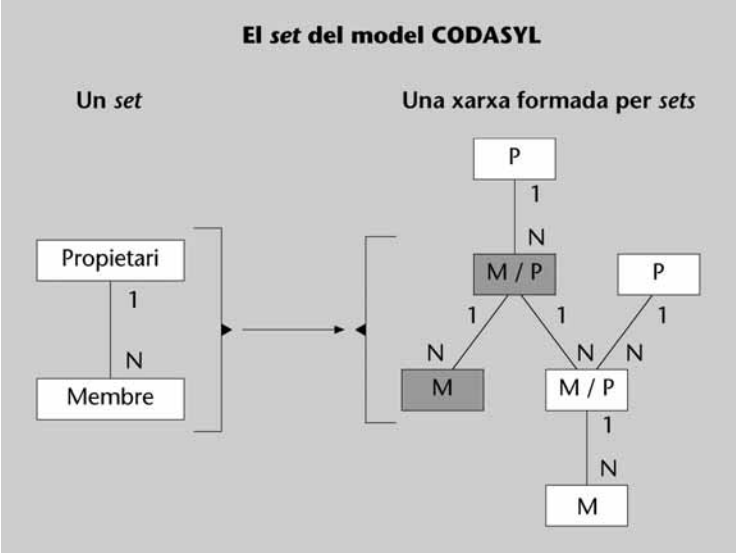
CODASYL

CODASYL és la sigla de Conference on Data Systems and Languages, i és un organisme que agrupa experts amb ànim d'establir estàndards en la indústria informàtica.

El **model en xarxa**, o **model CODASYL**, com també se'l va conèixer, defineix el *set* com a peça fonamental per a construir un disseny de bases de dades. Un *set* és constituït per dues entitats, l'entitat *Propietari** i l'entitat *Membre***, lligades per una relació d'u a molts, tal com es pot veure en la figura 5. El *set* permet a l'usuari del model de desplaçar-se d'una instància de *Propietari* a totes les instàncies dels seus *Membres* i d'una instància de *Membre* a la instància del seu *Propietari*.

*** En anglès, *Owner*.**
**** En anglès, *Member*.**

Figura 5



Amb el *set* com a element bàsic, es poden fer les construccions següents:

- L'entitat *Propietari* d'un *set* pot ser propietari d'altres *sets* i/o membre d'altres *sets*.
- L'entitat *Membre* d'un *set* pot ser propietari d'altres *sets* i/o membre d'altres *sets*.

D'aquesta manera tenim una veritable xarxa d'entitats. La potència de modelització d'aquest model és superior al jeràrquic i al dels fitxers encadenats, però per la mateixa raó la seva utilització és més complexa. El programador, en un model en xarxa, es pot desplaçar pràcticament de qualsevol entitat a qualsevol altra, sempre seguint els camins establerts pels *sets*.

Aquesta llibertat, però, significa que ha de conèixer bé el disseny de la base de dades i decidir com serà el desplaçament més adequat per a cada cas. Aquesta imatge del programador com la persona que ha de trobar el camí per la xarxa va fer néixer el concepte de **navegació per la base de dades**.

Exemples d'SGBD de tipus CODASYL

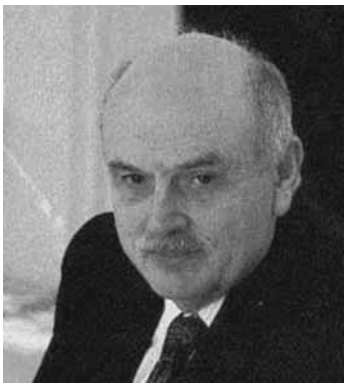
Alguns SGBD de tipus CODASYL són, entre d'altres, IDS-II, de Honeywell; DMS 1100, d'Univac; IDMS, de Cullinane; DBMS, de DEC; UDS, de Siemens.

Tret d'IBM, que va continuar amb l'IMS i el model jeràrquic, pràcticament totes les altres companyies que volien tenir part en el mercat emergent d'SGBD van adoptar el model en xarxa i van treure els seus SGBD que seguien més o menys l'estàndard CODASYL.

1.2. El model relacional "clàssic" i la dècada dels setanta

Als anys setanta s'estableixen les bases del model relacional i s'avança en el camp de la indexació.

1.2.1. El model relacional



Edward F. Codd

El 1970, Edward F. Codd, que treballava llavors al departament de recerca d'IBM, publica un article on s'exposen les idees que originarien un nou model de bases de dades: el **model relacional**. Contràriament als models anteriors, que no disposaven d'una definició formal, aquest nou model es presenta completament formalitzat, tant en l'estructura de les dades com en les operacions definides. La influència matemàtica, tant algebraica com lògica, hi és palesa.

Esmentem només aquí els **trets fonamentals del model relacional**:

- Els elements estructurals són les relacions, les tuples i els atributs, popularitzats de seguida com a **taules**, **files** i **camps**, que són la seva concreció física en les implementacions dels SGBD relacionals.
- Sobre aquests elements es va definir un conjunt d'operacions procedimentals, que es va anomenar **àlgebra relacional**, les principals de les quals són la unió, la intersecció, la diferència, el producte cartesià, la selecció, la projecció i la combinació*.
- Posteriorment es va definir un llenguatge declaratiu, que es va acabar anomenant **SQL**, amb les quatre sentències de manipulació de dades: SELECT, INSERT, UPDATE i DELETE.

Vegeu els mòduls "El model relacional i l'àlgebra relacional" i "El llenguatge SQL", de l'assignatura *Bases de dades I*.

*** *Combinació* és el nom català de l'operació *join*.**

Ens interessa, tanmateix, explicitar un conjunt d'elements estructurals, fruit de la definició formal del model, que si bé en una primera impressió podrien semblar menors, marquen clarament diferències amb els models anteriors. Encara que relacions, tuples i atributs són similars, en primera instància, a fitxers, registres i camps, hi ha diferències significatives:

a) Cada relació té un atribut o conjunt d'atributs, anomenat **clau primària**, amb un valor diferent per a cada tupla i que serveix per a identificar-la. Llavors, en una relació no hi pot haver dues tuples idèntiques.

b) Els atributs no poden prendre més d'un valor en una tupla.

c) L'ordre de les tuples dins de la relació i l'ordre dels atributs dins de la tupla són irrellevants.

D'aquestes característiques, ja podem entreveure un tret fonamental del model relacional. Es refereix a la forma d'identificar els diferents elements estructurals del model. Aquesta identificació es fa per **nom** i per **valor**. Una relació

té un nom, una tupla dins d'una relació s'identifica pel valor de la seva clau principal, un atribut dins d'una tupla s'identifica pel nom de l'atribut.

Resumint, podem dir que el model relacional identifica els seus elements per apuntadors simbòlics: noms i valors. No fa servir cap element físic per a aquest fi, sigui un apuntador físic o un ordre establert.

De manera conseqüent amb aquesta forma d'identificació, la interrelació de dues relacions es duu a terme amb la **clau forana**, que és un conjunt d'atributs en una de les relacions, de tal manera que els valors que prenen per a una tupla determinada coincideixen amb el valor de la clau principal d'alguna tupla en l'altra relació. Aquesta igualtat de valors permet passar d'una relació a una altra.

Amb tot això arribem a un punt molt important del model relacional, que és la separació clara que hi ha entre el nivell extern, o model pròpiament dit, i el nivell intern, o implementació. Aquesta separació és molt més gran que en qualsevol dels models precedents. El model relacional és potent i pensat per a l'usuari, mentre que la implementació, que no està definida en el model, es deixa als constructors dels SGBD relacionals.

Per aquesta gran distància entre els llenguatges dels nivells extern i intern apareix un component típic dels SGBD d'aquest model, l'**optimitzador**, la feina del qual és trobar el millor pla d'execució per a les sentències SQL d'una consulta d'usuari.

Llenguatges i nivells

El nivell extern s'expressa en un llenguatge declaratiu, l'SQL, i l'intern en un de procedimental, l'àlgebra relacional. L'optimitzador realitza la gran simplificació de passar d'un univers de pràcticament infinites sentències SQL a un de limitat a les set operacions bàsiques de l'àlgebra relacional.

La definició formal del model relacional és a la vegada prou potent i senzilla per a poder augmentar el nombre de possibles usuaris de la base de dades. Conceptes com taules, files i camps, i l'ús de l'SQL ja estaven no solament a l'abast dels usuaris informàtics –programadors i ABD– sinó que per primera vegada es preveia que estarien també a l'abast d'usuaris no informàtics, que podrien accedir més fàcilment a les dades emmagatzemades en les bases de dades.

Malgrat els evidents avantatges del model relacional, va transcórrer pràcticament una dècada abans que no sortissin al mercat els primers SGBD relacionals i encara va passar molt més temps abans que no s'hi imposessin.

Els pioners de l'SGBD relacionals

Les idees de Codd van propiciar dins IBM un projecte d'investigació per a construir un prototip d'SGBD relacional, que s'anomenà System/R. Pràcticament en paral·lel, a la Universitat de Califòrnia, a Berkeley, M. Stonebraker, un altre dels pioners dels prototips d'SGBD relacionals, dirigia un projecte similar, el projecte Ingres. De les experiències d'aquests dos projectes se n'aprofitaran els SGBD comercials que aniran sortint.

Si deixem el món de la recerca i ens fixem en el de les empreses que comercialitzen SGBD, veurem que durant aquesta dècada dels setanta, aquestes continuen venent productes basats en els models anteriors. IBM continua comercialitzant IMS –model jeràrquic– i la resta, SGBD basats en les especificacions CODASYL –model en xarxa. Uns tercers en discòrdia, Datacom i sobretot Adabas, treuen al mercat uns SGBD construïts segons un model alternatiu, que esmentarem més endavant.

No és fins al 1981-1982 que apareixen al mercat els primers productes relacionals: Oracle, un primer fruit del System/R, encara que comercialitzat per una al-

tra empresa que porta el nom del producte i que es crea en aquest moment, i Ingres. Uns dos anys més tard, IBM treu el DB2, producte basat en el treball del seu departament de recerca.

1.2.2. La indexació

L'accés per valor, propi del model relacional, necessita una implementació adequada per a poder transformar un valor en la localització física on resideix l'element de dades associat a aquest valor. Per a acomplir això s'empren tècniques d'aleatorització* i, sobretot, índexs.

*** En anglès, *hash*.**

Els **índexs** són estructures del tipus d'arbres balancejats, o arbres B.**

**** En anglès, *B trees*.**

Concretament, els arbres B^+ tenen unes característiques òptimes que fa que siguin utilitzats en la majoria dels SGBD relacionals.

Vegeu els arbres B^+ i el seu ús en els SGBD relacionals en el mòdul "Implementació de mètodes d'accés" de l'assignatura *Bases de dades II*.

Tanmateix, els índexs no sorgeixen amb la implementació del model relacional, són anteriors a aquesta. L'IMS n'usava per a localitzar una instància concreta d'una jerarquia, i fins i tot es troben en alguns sistemes de fitxers clàssics. D'altra banda, havien estat profusament emprats en un model diferent de bases de dades, el de fitxers invertits, en el qual es basaven productes com els esmentats Datacom i Adabas. En essència, es tracta de fitxers on tots o alguns camps estan indexats, cosa que facilita l'accés per valor d'un atribut o d'una combinació d'atributs.

Però els índexs existents abans del començament de la dècada dels setanta són índexs que es desorganitzaven ràpidament amb les altes i baixes d'entrades, per la seva estructura rígida. En conseqüència, el seu temps de resposta també degenerava apreciablement. El 1972/1973, coincidint amb el començament del projecte System/R, apareixen els índexs balancejats (B-trees, B^+-trees), que amb la seva estructura més flexible absorbeixen molt millor els canvis i garanteixen uns temps de resposta millors i més constants.

Així, doncs, els índexs balancejats van ajudar a desenvolupar l'accés per valor dels SGBD relacionals i d'aquesta manera van contribuir a fer-los realment operatius comercialment, cosa que probablement no hagués estat possible amb els índexs de la generació anterior.

1.3. Models postrelacionals i la dècada dels vuitanta

Abans que el model relacional es materialitzi en els primers SGBD relacionals del mercat, al món universitari s'investiga en uns models de dades que capturin més significat de l'entorn real que es vol modelitzar, això és, uns models

L'etapa de publicació dels treballs sobre els models semàntics més importants s'estén de 1974 a 1985.

que siguin semànticament més rics. Es considera que el model relacional és relativament pobre, quant a la semàntica que pot descriure, amb tots els avantatges que això representa, com per exemple, el fet de ser ràpidament comprès, però també amb els desavantatges de no modelitzar naturalment característiques més complexes.

1.3.1. Models semàntics

Fruit d'aquestes investigacions comencen a aparèixer uns models, anomenats **models semàntics**, que incorporen conceptes desenvolupats prèviament en altres àrees.

1) Del camp dels **llenguatges de programació** incorporen el concepte de **domini**. De fet, en el model relacional es parla de domini i els SGBD relacionals n'implementen un nombre reduït. Són els dominis més senzills i necessaris: enters, decimals, caràcters, booleans, etc. En podríem dir *dominis sintàctics*. Però en determinats llenguatges de programació ja s'havien implantat uns dominis més potents, els dominis semàntics, definits pel programador.

El domini semàntic es potencia quan s'hi afegeix la possibilitat de definir les operacions permeses sobre aquest domini. Tenim llavors el concepte de **tipus de dada**. Aquests conceptes són interessants perquè restringeixen el marge d'error dels usuaris del sistema en prohibir-los operacions incorrectes i sense sentit.

Tipus abstractes de dades

El concepte de tipus de dada encara evolucionà més amb la separació de la definició i la implementació del tipus i amb l'encapsulament d'aquest, cosa que donà origen al concepte de tipus abstracte de dades o TAD.

Avantatges dels dominis semàntics

L'exemple següent il·lustra els conceptes d'operacions i tipus de dades en l'àmbit dels dominis semàntics. En el model relacional clàssic podem tenir definits en una taula els atributs següents: número de telèfon i prefix, com dos enters. Atès que pertanyen al mateix domini sintàctic, `enter`, podem fer comparacions entre ells o fins i tot sumar-los, operacions que evidentment no tenen cap sentit. Un sistema que disposi de la potència del concepte de *tipus de dada* no permetria aquestes operacions, ja que en definir el tipus `número de telèfon`, per exemple, hauríem limitat el conjunt d'operacions vàlides a la de concatenació amb l'atribut `prefix`. Evidentment, tindríem un SGBD semànticament més potent i menys vulnerable a usos incorrectes dels usuaris.

2) Un altre camp del qual els models semàntics es van nodrir va ser el de la **intel·ligència artificial**, concretament de la branca de representació del coneixement. En aquesta disciplina, una de les formes de representar el coneixement són les xarxes semàntiques, on trobem conceptes com:

Podeu repassar aquests conceptes en el mòdul "Introducció al disseny de bases de dades" de l'assignatura *Bases de dades I*.

a) Instanciació/classificació (`instance of`**)**. Es refereix a la interrelació que hi ha entre una entitat i les seves instàncies o ocurrències. Exemple: l'entitat *vaixell* i uns vaixells concrets.

- Instanciació és aquesta relació en un sentit: l'entitat *vaixell* s'instancia en les ocurrències Queen Mary i Titanic.
- Classificació és en sentit contrari: els dos vaixells esmentats es classifiquen com a instàncies de l'entitat *vaixell*.

- `Instance of` és l'expressió anglesa que s'ha imposat: el Queen Mary és una `instance of` l'entitat *vaixell*.

b) Generalització/especialització (`is a`**)**. És una categoria d'interrelacions entre entitats. Exemple: l'entitat *vaixell* és una generalització de les entitats *mercant*, *creuer*, *petrolier* i *cuirassat*. I cadascuna d'aquestes és una especialització de l'entitat *vaixell*.

c) Agregació/descomposició (`part of`**)**. És una altra categoria d'interrelacions entre entitats, quan una entitat és l'agregació de les altres. Exemple: l'entitat *vaixell* és una agregació d'entitats dels seus components: motors, hèlixs, cobertes, etc.

d) Associació o ***grupping*** **(**`member of`**)**. És encara una altra categoria del mateix, que cobreix interrelacions menys estructurals, com per exemple, l'entitat *comboi* és una associació de vaixells que naveguen junts.

Per contra, el model relacional clàssic només preveu un mateix tipus d'interrelació entre taules. Tots els tipus anteriors es representen de la mateixa manera, i es perd capacitat semàntica de modelització.

En aquell moment, aquests models buscaven enriquir la semàntica més enllà de la proporcionada pel model relacional mitjançant conceptes com el d'entitat, diferents categories d'interrelacions i el de domini semàntic per als atributs. Van restar com a treballs de recerca i no es van implementar en SGBD nous. De totes maneres, moltes de les seves idees es van fer servir comercialment en metodologies de disseny de bases de dades, en aquelles fases prèvies a la implementació del disseny en un SGBD concret.

Exemples de models semàntics

Alguns models semàntics són Abrial (1974), Entitat-relació (Chen, 1976), RM/T (Codd, 1979), NIAM (1982), ACM/PCM (1982).

Aquests dissenys reflectien tota una sèrie de consideracions semàntiques que, en no poder-les garantir l'SGBD, s'havien d'incloure en la programació de les aplicacions, però que, no obstant això, es considerava interessant que s'explicitessin i es recollissin en el procés de disseny i construcció de bases de dades i aplicacions, ja que milloraven la qualitat global del sistema desenvolupat.

El model ER com a model semàntic

Nosaltres mateixos, en el mòdul "Introducció al disseny de bases de dades" de l'assignatura *Bases de dades I*, fem servir un model semàntic, el model entitat-relació (ER), derivat dels treballs de Chen.

1.3.2. Bases de dades orientades a l'objecte

La programació orientada a l'objecte (OO) tenia ja uns anys quan, al final de la dècada dels vuitanta, apareixen les primeres bases de dades orientades a l'objecte. Comencen essent, fonamentalment, extensions dels llenguatges OO per a proporcionar la capacitat de persistència a les dades manipulades pels programes. Per aquesta raó, aquestes bases de dades estan fortament influïdes pels conceptes propis dels llenguatges OO, principalment pel d'objecte, cosa que és evident.

Abreugem l'expressió *orientat a l'objecte* amb la sigla OO.

Podem veure el concepte d'**objecte** com l'assumpció de la representació de la complexitat en el llarg camí de modelitzar la realitat de les dades. En el subapartat anterior hem vist aquesta evolució des dels dominis més senzills fins als tipus (abstractes) de dades. L'objecte incorpora i modelitza encara

més potència semàntica en permetre que els seus atributs siguin elements semànticament tan rics com TAD, altres objectes i fins i tot diverses menes de col·leccions d'objectes.

Una característica important i distintiva dels objectes és el seu identificador: identificador d'objecte o, abreujadament, OID.

La sigla OID prové de l'expressió anglesa *Object identifier*.

L'**identificador d'objecte** és un identificador únic per a cada objecte, independent dels valors dels seus atributs, que no canvia al llarg de la vida de l'objecte i que no és reaprofitable per a un segon objecte, encara que el primer deixi d'existir.

Valors iguals amb OID diferents

Es pot donar el cas de valors iguals amb OID diferent. Per exemple, dues peces iguals poden tenir tots els seus atributs iguals però no deixen de ser dos objectes diferents. Una cosa és la igualtat –el cas d'aquestes dues peces iguals– i l'altra la identitat –una peça només és idèntica a si mateixa.Valors iguals amb OID diferents

Com podem observar, és un concepte diferent del de *clau primària* del model relacional, ja que aquesta depèn d'algun valor present en la tupla i, en canvi, l'OID és independent de qualsevol valor.

Els primers SGBD orientats a l'objecte

Alguns dels primers sistemes de bases de dades orientades a l'objecte van ser: GemStone, Orion, O_2, Iris i ObjectStore.

Malgrat que aquí encara no es parli de *model de bases de dades orientat a l'objecte* perquè, de fet, aleshores no existia un model així, sinó només uns productes que donaven persistència als llenguatges OO, sí que, com a mínim, podem pensar que aquests productes proporcionaven un model conceptual que hereta els conceptes del model semàntic vist en el subapartat anterior. En efecte hi trobem:

a) Classe d'objecte i objecte, que podem veure com l'entitat i instància del model semàntic amb la potència afegida d'incorporar més possibilitats a l'hora de definir el tipus de dada dels atributs.

b) Diverses possibilitats d'interrelacionar classes d'objectes, com ens trobàvem també en el model semàntic. Una d'aquestes possibilitats, com és natural en l'orientació a l'objecte, és l'herència, que també podem considerar relacionada amb el concepte de generalització-especialització vist allà.

1.4. Models postrelacionals i la dècada dels noranta

Al tombant de la dècada dels vuitanta a la dels noranta apareixen dos manifestos, signats per prestigiosos informàtics del camp de les bases de dades, que pretenen d'indicar cap a on han d'evolucionar els sistemes de bases de dades.

1.4.1. Els manifestos

El **primer manifest**, titulat *Manifest dels sistemes de bases de dades orientades a l'objecte*, de 1989, relaciona un seguit de característiques que, segons els autors, haurien de tenir els sistemes de bases de dades per a poder suportar tot tipus d'aplicacions. Aquestes característiques són clarament orientades a l'objecte com, per exemple, el suport d'objectes de qualsevol complexitat, d'identificador d'objecte, de classes d'objectes, d'encapsulació, de jerarquia de classes i d'herència.

Com es pot veure, és una aposta clara per trobar un model de bases de dades orientat a l'objecte. A més d'aquestes característiques, el manifest també n'esmenta unes altres, que són les pròpies d'un SGBD, com la persistència de les dades, el control de la concurrència i la recuperació enfront d'errors.

Així, doncs, la idea de fons és dotar els llenguatges OO de persistència i convertir aquest entorn en un veritable SGBD a base d'afegir-hi totes les funcionalitats proporcionades per aquests gestors de dades.

El **segon manifest**, titulat *Manifest dels sistemes de bases de dades de tercera generació*, de 1990, dóna la volta al dilema i propugna ampliar o estendre els SGBD relacionals afegint-los les característiques dels models semàntics, cosa que els proporciona unes funcionalitats similars a les de les bases de dades orientades a l'objecte.

Els partidaris del primer manifest no estan d'acord amb les tesis del segon. Opinen que només les bases de dades OO anomenades per alguns *pures* poden suportar adequadament aplicacions que requereixen un model de dades complex com, per exemple, CASE, CAD/CAM, CIM i similars.

Aplicacions típiques amb models complexos

Algunes aplicacions amb models de dades complexos són les que esmentem a continuació:

- CASE: Computer Aided Software Engineering
- CAD/CAM: Computer Aided Design/Computer Aided Manufacturing
- CIM: Computer Integrated Manufacturing

Els que defensen el segon manifest argumenten que és raonable conservar tot l'esforç de desenvolupament acumulat actualment en els SGBD relacionals, que han provat durant força temps la seva validesa i solidesa al mercat. Evidentment és l'opció que recolzen i segueixen els grans fabricants d'aquest tipus de programari, com és el cas d'IBM, Oracle, Informix i Microsoft, entre altres.

La dicotomia dels dos manifestos es materialitza en els SGBD que es troben presents al mercat, com es pot veure a continuació.

1.4.2. Bases de dades orientades a l'objecte "pures"

Els SGBDOO d'aquesta dècada

Els SGBDOO d'aquesta dècada continuen essent els que van aparèixer abans: GemStone, O_2, Objectivity, ObjectStore, Poet.

En el camp de les bases de dades orientades a l'objecte, que qualificarem de *pures* per a distingir-les millor dels sistemes que veurem en el subapartat següent, a més dels diferents SGBD orientats a l'objecte presents al mercat, esmentarem l'esforç de modelització i estandardització portat a terme. Cal tenir en compte que un dels retrets que es feia a aquests productes és que, a diferència dels relacionals, no tenien cap teoria elaborada en què basar-se.

El grup ODMG

ODMG es va constituir com un grup vinculat a OMG (Object Management Group), organització oberta creada el 1989 per a estandarditzar programari OO. Un dels seus estàndards més coneguts és CORBA.

L'esforç d'estandardització el va portar a terme, i encara el porta, un grup anomenat Object Database Management Group (ODMG). Integrat majoritàriament per persones que treballen en constructors d'SGBD orientats a l'objecte,

aquest grup ha creat un estàndard de persistència d'objectes que actualment es troba en la seva tercera versió (ODMG 3.0) i defineix elements com:

- Un **model d'objectes** (*Object Model*), basat en CORBA.
- Un **llenguatge de definició** dels objectes persistents (*Object Definition Language*).
- Un **llenguatge de manipulació** d'aquests objectes (*Object Manipulation Language*).
- Un **llenguatge d'interrogació** per valor dels atributs, que busca una potència i senzillesa similars a la de l'SQL (*Object Query Language* o *OQL*).
- Unes **interfícies** per a usar els elements anteriors des de diferents llenguatges OO: Smalltalk, C++ i Java.

L'estàndard CORBA

La *Common Object Request Broker Architecture*, l'arquitectura de sistemes d'objectes distribuïts definida per l'OMG, s'ha vist en el mòdul "Introducció al programari distribuït" de l'assignatura *Enginyeria del programari I*.

1.4.3. El model relacional estès i els SGBD relacionals amb objectes

Paral·lelament als esforços d'estandardització dels sistemes de bases de dades orientats a l'objecte, els SGBD relacionals, durant aquests últims anys, han continuat augmentant la potència de la seva interfície principal, el llenguatge SQL: hi han incorporat capacitat de modelitzar la realitat, sobretot en la línia dels models semàntic i orientat a l'objecte.

Relacional amb objectes

Fem servir aquest neologisme per traduir l'anglès *object-relational*, que s'usa àmpliament per a designar aquest model. En la literatura tècnica anglesa trobareu també la sigla ORDBMS (*Object-relational Database Management System*), referida als SGBD que el suporten.

Aquest enriquiment del llenguatge SQL s'inscriu en un procés que comença amb la seva aparició i que encara no ha acabat.

L'evolució del procés d'estandardització de l'SQL

L'SQL va ser estandarditzat per primera vegada el 1986 per ANSI. Aquesta versió constava només del nucli més bàsic del llenguatge i el 1989 ANSI i ISO la van ampliar amb la integritat referencial. Aquest primer estàndard –el de 1986 i el de 1989– es coneix com SQL1. L'SQL dels diferents SGBD relacionals comercials s'acostava a aquest estàndard, però sense complir-lo en la seva totalitat. D'una banda, deixaven d'implementar alguns elements mentre que n'implementaven alguns altres de propis fora de l'estàndard.

Perquè l'estàndard no quedés desfasat respecte a l'ampliació constant que experimentava l'SQL de les cases comercials, se'n va definir un de nou anomenat SQL2, o SQL92 ja que es publicà l'any 1992. Juntament amb millores en l'ortogonalitat del llenguatge i ampliacions en l'operativa, com els *outer-joins* i més possibilitats dels cursors, trobem ja característiques que podem considerar com una primera extensió del llenguatge: el concepte de *domini*, les restriccions (*constraints*) i nous tipus de dades (`DATE` i `TIME`, per a expressar el temps).

Michael Stonebraker

Un dels promotors del manifest de bases de dades de tercera generació és Michael Stonebraker, que des del sistema Ingres, passant pel Postgres i Illustra, fins a Informix 9, ha treballat en el camp dels SGBD i els ha enriquit amb noves funcionalitats en la línia dels actuals sistemes relacionals amb objectes.

A partir de 1992 comença realment la cursa de l'extensibilitat de l'SQL i ho fa seguint la filosofia del manifest de les bases de dades de tercera generació. Cal tenir present també que els SGBD comercials més importants ja començaven a incorporar llavors diverses funcionalitats que estenien clarament el model relacional. Era el cas de l'Universal Server d'Oracle, de l'Universal Server d'Informix o del DB2 Universal Database d'IBM. Aquests productes se'ls bateja

amb el nom de **sistemes de gestió de bases de dades relacionals esteses** (o **sistemes de gestió de bases de dades relacionals amb objectes**).

La nova estandardització, basada en el conjunt de les diferents propostes i coneguda com SQL3, va sortir en fases successives. La primera, el 1995, va estandarditzar la interfície SQL/CLI. La segona, el 1996, ho va fer amb l'SQL/PSM i els disparadors (*triggers*). La tercera i última va acabar sortint el 1999 i, per aquest motiu, l'SQL3 es coneix també com a SQL:1999. Popularment, se'l coneix també com SQL OO, ja que incorpora característiques d'aquest paradigma. És l'estàndard que tenim actualment per a les bases de dades esteses o relacionals amb objectes.

CLI prové de l'expressió anglesa *command level interface*.

PSM prové de l'expressió anglesa *persistent stored modules*.

Versions d'SGBD relacionals amb objectes

Els diferents SGBD basats en SQL es poden considerar relacionals amb objectes a partir de les versions següents: Oracle, versió 8; Informix, versió 9; DB2, versió 6.

S'hi defineixen els aspectes següents:

a) Nous tipus de dades aptes per a emmagatzemar gran quantitat d'informació. Són els *large objects* (LOB), que s'empraran en multimèdia –text, imatge, vídeo, àudio–, en els sistemes geogràfics i en moltes altres aplicacions.

b) Més funcionalitat, com els disparadors (*triggers*), ja a l'any 1996, i les consultes recursives, que es fan amb la clàusula `WITH`.

c) Característiques típicament orientades a l'objecte, com:

- Els tipus de dades complexos proporcionats pel sistema (`row`, `array`, `ref`) o definits per l'usuari (*udtypes* o *distinct types*).
- Les funcions definides per l'usuari (*user defined functions*).
- El suport de veritables objectes amb la possibilitat de crear taules d'un tipus concret i taules que hereten d'altres taules (`CREATE TABLE OF TYPE` i `CREATE TABLE UNDER`).

SGBD amb estructura de components

Els SGBD comercials que segueixen de més a prop l'SQL3 –els SGBD relacionals amb objectes– han acabat adoptant una estructura de components. Hi ha un nucli important, que és un motor de base de dades potent i de propòsit general –això és, per a qualsevol de les aplicacions més usuals del mercat i que, per això, les cases comercials l'han anomenat Universal Server o Universal Database, emfasitzant la seva aplicació universal.

En aquest nucli es poden connectar diferents components especialitzats per a un determinat camp d'aplicacions més noves: aplicacions geogràfiques, multimèdia, de tractament de textos, etc. Aquests components especialitzats estenen la funcionalitat de l'SGBD del nucli i es poden comercialitzar junt amb el nucli o bé com a productes a part. Cada casa constructora els ha batejat amb un nom diferent. Així, IBM els anomena *extenders*, Informix, *datablades*, i Oracle, *cartridges*.

L'esforç d'estendre l'SQL continua. Totes aquelles característiques poc madures per a pertànyer a l'estàndard SQL:1999 són estudiades per a un pròxim pa-

quet, que ja s'ha batejat com a SQL4. En paral·lel, trobem ja algun SGDB comercial, com Oracle, on podem definir taules d'objectes, cadascun amb el seu OID, i operacions específiques per taula (els mètodes de la classe d'objectes). Com es pot veure, es tracta d'un suport força complet del model OO incrustat dins un SGBD originalment relacional.

1.5. Present i futur dels models de bases de dades

Si ens atenim al nombre de sistemes instal·lats actualment al món, el vencedor per franca majoria és encara el model relacional clàssic. Però, seguidament, n'haurem de considerar el successor natural, el model relacional estès, tant pel percentatge no menyspreable d'instal·lacions que ostenta com pel fet que molts dels sistemes relacionals clàssics passaran naturalment als estesos, ja que aquest pas pot ser simplement un canvi de versió del mateix producte comercial.

El model relacional estès sembla, doncs, el destinat a ser el vencedor en la pugna entre les diferents opcions.

Els sistemes de bases de dades orientades a l'objecte *pures* tenen una presència molt minoritària al mercat i molts analistes els veuen només com a sistemes *nínxol* que subsistiran gràcies a la seva gran adequació a una sèrie d'aplicacions que, en canvi, són poc apropiades per a les bases de dades relacionals. Tanmateix, es pot pensar que la seva sort augmenti paral·lelament a l'auge de l'enginyeria de programari OO que es viu en l'actualitat. Les arquitectures d'objectes distribuïts, que s'analitzen i dissenyen amb UML i es programen amb Java, poden fer més atractives les bases de dades OO, ja que pertanyen al mateix model i no hi ha cap mena de discrepància en tot el procés de desenvolupament de l'aplicació.

UML

L'Unified Modeling Language permet modelitzar un sistema OO. La seva exposició ocupa dos mòduls de l'assignatura *Enginyeria de programari I*.

Encara que els SGBD orientats a l'objecte i els SGBD relacionals segueixin dos models diferents, no hi ha dubte que es van acostant quant al suport que realment proporcionen. Ja hem vist com els relacionals s'estenen cap als OO. També els OO s'han estès cap als relacionals amb els treballs de l'ODMG, especialment amb l'intent de disposar d'un llenguatge d'interrogació per valor (l'OQL).

OQL

La manca d'aquesta característica era una realitat en els primers sistemes de bases de dades OO i en limitava l'ús a aplicacions molt específiques.

Hi ha analistes que pensen que els dos models, encara que no convergiran en un d'únic –de fet s'ha mirat d'anar per aquest camí, però sembla que sense èxit–, permetran una explotació creuada: aplicacions SQL podran accedir a bases de dades OO i aplicacions OQL accediran a bases de dades relacionals. No hem d'oblidar tampoc els interessos econòmics de les empreses fabricants dels SGBD que, juntament amb el mercat, tenen una gran influència en el rumb que acabarà prenent l'evolució d'aquests productes informàtics.

Finalment, és interessant d'esmentar les conclusions de l'anomenat Asilomar Report, resultat d'una reunió de treball, el 1998, de setze destacats investigadors, en l'àrea de les bases de dades, de la universitat, empresa i govern dels EUA. Recomanen dirigir la investigació futura cap a:

a) SGBD que s'autoafinin*. Cal evitar el nombre elevat de paràmetres o opcions dels sistemes actuals.

*** En anglès, *selftuning*.**

b) Possibilitar la comunicació entre una gran quantitat d'SGBD presents en una xarxa, com Internet.

c) Repensar l'arquitectura tradicional dels SGBD en vista dels avenços en maquinari: memòries principals de gran capacitat, paral·lelisme en la unitat central de procés, una fiabilitat més gran dels components –cosa que pot permetre l'operació 7×24–, discos RAID, arquitectures distribuïdes.

d) Disposar d'una bona arquitectura d'aplicacions pensada per als sistemes relacionals amb objectes.

e) Suportar l'accés a dades poc estructurades, com també és el cas de les que trobem a Internet. Per això caldrà desenvolupar estàndards i sistemes de metadades.

1.6. Altres tipus de bases de dades

A més dels models de bases de dades que hem descrit, trobareu molts altres noms i conceptes que fan referència a altres tipus de bases de dades, esmentats en la literatura científica i comercial sobre el tema. Això és així perquè al llarg de la vida dels SGBD ha aparegut tot un repertori de noves tecnologies informàtiques, tecnologies que els sistemes de bases de dades han incorporat o bé que han estat ampliats per a suportar-les quan ha convingut de fer-ho.

Així, tenim bases de dades:

- dissenyades per a un sol usuari –les bases de dades personals– o per a ser compartides per diferents usuaris –les multiusuari;
- que suporten i aprofiten l'arquitectura específica d'un ordinador, com les que aprofiten el paral·lelisme d'unitats de procés, de memòries o de discos, per exemple;
- que s'han adaptat als diferents models de computació distribuïda, des de les purament centralitzades, passant per les primeres bases de dades client/servidor, després per les bases de dades distribuïdes –més generals, que s'adapten a aplicacions distribuïdes en més de dos nivells i que suporten rèpliques de dades–, fins a arribar a les mòbils d'avui en dia;
- per a suportar adequadament tipus específics de dades, per exemple, el temps en les bases de dades temporals, dades topogràfiques en les bases de dades espacials o àudio i vídeo en les bases de dades multimèdia;

- dissenyades no solament per a emmagatzemar dades, sinó per a tenir un comportament més actiu i reaccionar davant de determinats fets, com són les bases de dades actives, proveïdes de regles i disparadors;

- per a suportar aplicacions específiques, com les bases de dades documentals per a cerques bibliogràfiques o els magatzems de dades actuals (*datawarehouse, datamart*) dissenyats per a aplicacions d'anàlisi de dades (OLAP).

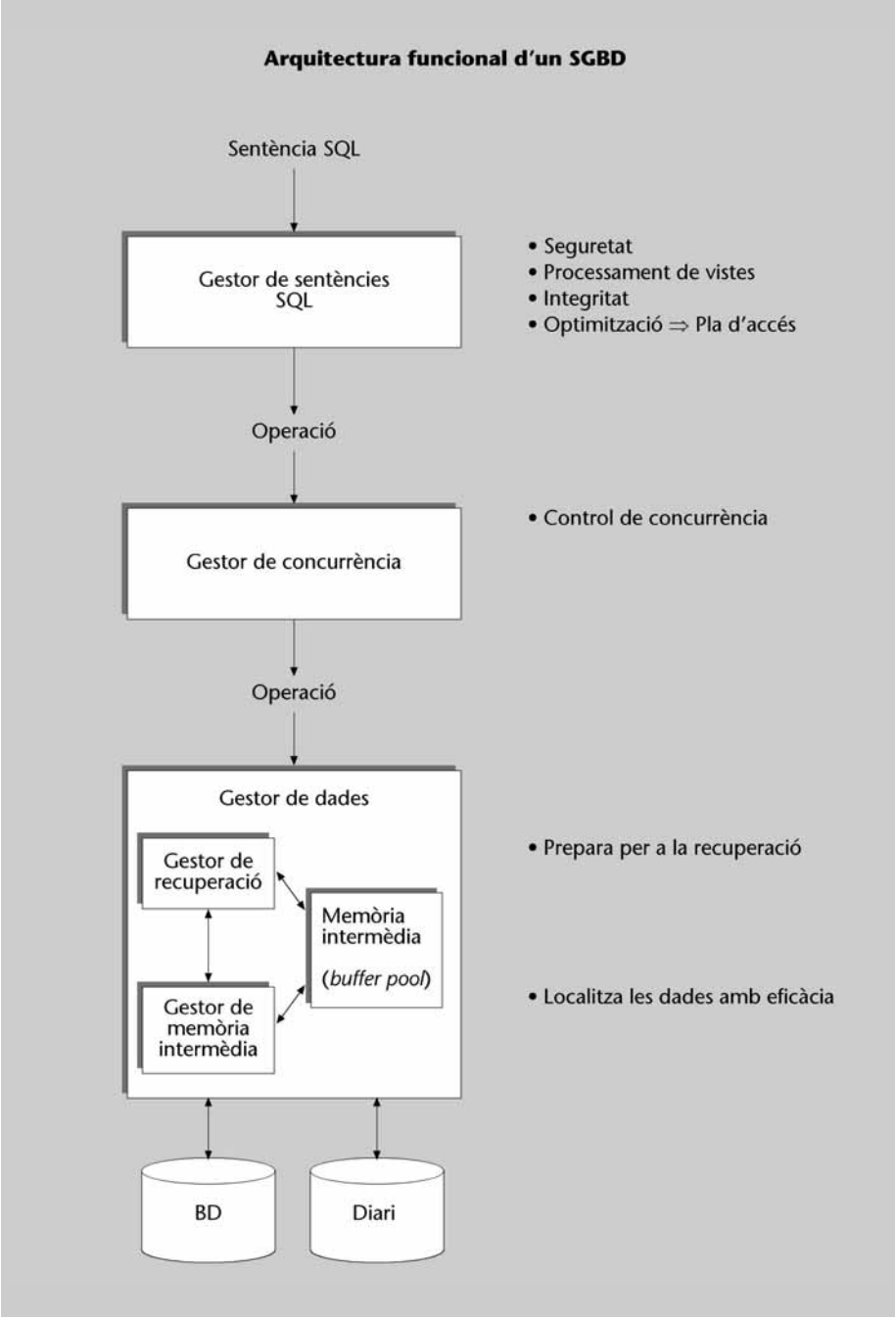
Tota aquesta diversitat funcional de bases de dades, apareguda per a donar resposta a l'evolució de les necessitats i de la tecnologia, no ha d'emmascarar, però, la línia principal de l'evolució dels models de bases de dades, que és el que hem desenvolupat en aquest apartat i que s'il·lustra a la taula següent amb la menció de les fites més importants.

Fets destacats en l'evolució de les bases de dades	
1964	IDS, fet per Charles Bachman a General Electric, es pot considerar el primer SGBD. Font d'idees de molts desenvolupaments posteriors.
1969	Primera versió d'IMS, a IBM.
1970	E.F. Codd, d'IBM, publica "A Relational Model of Data for Large Shared Data Banks", origen del model relacional.
1971	Primer DBTG Report de CODASYL (el més important es publicarà el 1973). A partir de llavors els grans fabricants, tret d'IBM, comercialitzen SGBD segons el model CODASYL: IDS (Honeywell), DMS 1100 (UNIVAC), IDMS (Cullinane, després Cullinet).
1973	Projecte System/R, a IBM, San José, i en paral·lel projecte Ingres, a la Universitat de Berkeley (Califòrnia), amb M. Stonebraker.
1974	Primera publicació de l'SQL, anomenat primerament SEQUEL.
1976	Model ER, de Chen.
1977	ANSI/SPARC Report: Arquitectura de tres nivells.
1981/82	Apareixen els primers SGBD relacionals, amb llenguatges d'interrogació: Oracle, amb SQL; Ingres, amb QUEL.
1983	Primera versió de DB2, d'IBM.
1986 i 89	Estàndard SQL1.
1987	Apareixen els primers SGBDOO: GemStone.
1989	Manifest del sistema de bases de dades orientades a l'objecte.
1990	Manifest del sistema de bases de dades de tercera generació.
1992	Estàndard SQL92 (SQL2).
1993	Versió 1 de l'estàndard ODMG per a bases de dades OO.
1995	Comencen a aparèixer SGBDR amb característiques OO.
1999	Estàndard SQL:1999 (SQL3). Hi havia hagut lliuraments previs el 1995 i 1996. També, versió 3 de l'estàndard ODMG.

2. Arquitectura funcional d'un SGBD

Com ja s'ha dit a la introducció, en aquest apartat seguirem el processament d'una sentència SQL a través d'un SGBD relacional per a descobrir i justificar les diferents funcionalitats que ha de proporcionar un programari d'aquesta mena. Aquestes funcionalitats s'agrupen en uns blocs que anomenarem *gestors*. El conjunt dels gestors dibuixen una arquitectura de l'SGBD, que trobareu representada en la figura 6.

Figura 6



Repetim, però, que aquesta arquitectura és funcional i no ha de coincidir necessàriament amb l'arquitectura real dels components que constitueixen un SGBD concret, que evidentment té en compte moltes altres consideracions. Aquesta simplificació funcional ens serveix aquí per a introduir els temes que s'aniran veient en els altres mòduls.

2.1. Processament d'una sentència SQL

Abans de processar una sentència SQL cal assegurar-se que l'usuari que n'ha demanat l'execució estigui autoritzat a poder realitzar el conjunt d'operacions que implica la sentència esmentada. Per a decidir-ho, s'empren un conjunt de conceptes i un conjunt de tècniques anomenades **tècniques de seguretat**.

La seguretat proporcionada pels SGBD es veu en el mòdul "El component de processament de consultes i peticions SQL", d'aquesta assignatura.

Si la sentència SQL no supera els filtres de la seguretat és rebutjada pel sistema i la seva execució ja no segueix endavant. En cas contrari, en cas que sigui autoritzada a executar-se, fa una transformació per a eliminar les referències a vistes. Com sabem, una sentència SQL es pot referir a operacions sobre vistes i sobre taules; doncs bé, en el pas de **processament de vistes**, les referències a vistes són substituïdes per les referències equivalents a taules. La sentència queda més homogènia, només amb referències a taules.

El processament de vistes s'explica en el mòdul "El component de processament de consultes i peticions SQL" d'aquesta assignatura.

Reduïda la sentència SQL a una sèrie d'operacions sobre taules, cal mirar llavors que no incompleixi cap dels **requeriments d'integritat** expressats en la definició de la base de dades. Ens referim a tot allò que s'ha definit de les claus primàries i foranes, com també les diferents restriccions que poden haver estat definides sobre columnes i sobre taules.

La integritat ja s'ha estudiat en els mòduls "El model relacional i l'àlgebra relacional" i "El llenguatge SQL", de l'assignatura *Bases de dades I*, i ampliada en el mòdul "Components lògics d'una base de dades" de l'assignatura *Bases de dades II*. Per aquesta raó, en aquesta assignatura no s'esmentarà res més d'aquest tema.

Un cop superats tots aquests controls, la sentència s'ha d'executar. Però el llenguatge de la sentència SQL és declaratiu, per això s'ha de transformar en un conjunt d'accions expressades en un llenguatge procedimental. El primer pas és traduir la sentència SQL a operacions de l'àlgebra relacional i posteriorment descompondre cadascuna d'aquestes operacions de l'àlgebra relacional en operacions encara més elementals sobre els elements físics de la base de dades.

Cadascuna d'aquestes dues transformacions no és determinista. El conjunt i l'ordre d'execució de les operacions de l'àlgebra relacional que calen per a resoldre una sentència SQL normalment no seran únics. De la mateixa manera, les operacions físiques –per exemple, la lectura seqüencial d'un fitxer, l'accés a una fila mitjançant un índex– que cal executar per a satisfer una operació de l'àlgebra relacional determinada i l'ordre en què s'han d'executar també presenten un ventall de possibilitats. El **procés d'optimització** s'encarrega precisament d'escollir, en cadascuna de les transformacions, la millor combinació possible d'operacions procedimentals que satisfà l'execució de la sentència SQL. El procés d'optimització acaba reflectint aquest resultat en un procediment anomenat **pla d'accés**.

L'optimització ja s'ha tractat en el mòdul "Altres temes de bases de dades" de l'assignatura *Bases de dades II*, però es repassa i s'amplia en el mòdul "El component de processament de consultes i peticions SQL" d'aquesta assignatura.

En l'arquitectura funcional dels SGBD, els quatre passos de control de la seguretat, processament de vistes, control de la integritat i optimització, que hem vist que ha sofert la sentència SQL, s'engloben com a funcions del **gestor de sentències SQL**, que representa la part del davant* o part de l'SGBD més en contacte amb l'usuari.

*** En anglès, *front-end*.**

Un cop la sentència SQL ha estat transformada en el pla d'accés, cada una de les operacions elementals que el componen segueix altres tractaments a través de la resta de l'SGBD.

En primer lloc, es comprova per a cada operació elemental que l'acció que intenta fer no entri en conflicte amb l'acció d'una altra operació que també s'estigui executant. Per exemple, cal evitar que dues operacions elementals d'actualització fetes en benefici de dues sentències SQL diferents actualitzin alhora una mateixa dada. Ens cal, doncs, un **gestor de concurrència**, que controli l'execució concurrent de múltiples operacions elementals.

El control de concurrència s'ha estudiat en el mòdul "Transaccions en les bases de dades. Control de concurrència i recuperació" de l'assignatura *Bases de dades II*. De totes maneres, en trobareu una petita ampliació en el mòdul "El component de processament de consultes i peticions SQL" d'aquesta assignatura.

Finalment, si no hi ha cap problema de concurrència, cada operació elemental s'executa contra la base de dades. La part de l'SGBD que s'encarrega d'això és el **gestor de dades**, que és la del darrere*, ja que és la més allunyada de l'usuari.

*** En anglès, *back-end*.**

Abans de fer la lectura o escriptura d'una dada, que és el que normalment demana de fer l'operació elemental, cal prendre'n nota, ja que en cas de fallades posteriors del sistema, es tindrà un diari d'operacions i amb aquest la possibilitat de recuperar-les. Les funcions del gestor de dades que s'encarreguen d'això constitueixen el **gestor de recuperació**.

Un mecanisme molt important dels SGBD per a augmentar la velocitat de procés és la **memòria intermèdia***. En aquesta memòria es guarden les dades a les quals l'SGBD ha accedit recentment, ja que, a causa del tipus de processament que fan aquests productes, la probabilitat de tornar-hi a accedir és alta. D'aquesta manera, s'estalvien moltes operacions físiques d'entrada/sortida, molt més costoses en temps que els accessos a la memòria principal, on resideix la memòria intermèdia. El **gestor de memòria intermèdia** resol en la memòria intermèdia totes aquelles operacions elementals que pot i només va físicament a la base de dades quan és necessari.

*** En anglès, *buffer pool*.**

La recuperació s'ha vist en el mòdul "Transaccions en les bases de dades. Control de concurrència i recuperació" de l'assignatura *Bases de dades II*. En el mòdul "El component de gestió de les dades" d'aquesta assignatura s'explica amb més profunditat alhora que s'estudia també la gestió de la memòria intermèdia.

Pel que fa a la base de dades, cal que aquesta tingui una organització adequada perquè la cerca de les dades es faci d'una manera òptima. Tota l'última part de l'assignatura tracta d'aquesta qüestió, que és el que entenem per disseny de la base de dades. Com es veurà, aquest disseny té diferents etapes, cosa que fa que parlem d'un disseny conceptual, d'un disseny lògic i, finalment, d'un disseny físic.

Trobareu tres mòduls dedicats al disseny de bases de dades. En el mòdul "Disseny conceptual i lògic de bases de dades" es repassen conceptes del mòdul "Introducció al disseny de bases de dades" de l'assignatura *Bases de dades I*, mentre que els altres dos mòduls, "Reconsideració dels models conceptual i lògic" i "Disseny físic de bases de dades" tracten matèries que encara no s'han vist.

Resum

En aquest mòdul hem passat revista a l'evolució del món de les bases de dades i dels seus gestors per mitjà dels diferents models que han aparegut al llarg del temps i que s'han imposat, sigui al mercat o a la universitat. Aquesta evolució ens ha d'ajudar a comprendre millor el camp de les bases de dades i les principals tendències de l'actualitat.

Posteriorment, ens hem servit del processament esquemàtic d'una sentència SQL per a descobrir quines són les funcions més importants que ha de proporcionar un SGBD. Aquest coneixement ens ha d'ajudar a contextualitzar els diferents mòduls d'aquesta assignatura, ja que aprofundeixen precisament en aquestes funcions.

Activitats

1. Us aconsellem d'anar a les webs de les principals empreses fabricants d'SGBD actuals. Busqueu la informació relativa a aquests productes –penseu que alguna d'aquestes cases comercialitzen molts altres productes a més d'SGBD– i observeu quines són les característiques més sobresortints dels gestors de bases de dades que trobeu en aquestes webs.

Com a guia aquí teniu una relació d'adreces web d'unes quantes d'aquestes empreses:

a) Empreses que comercialitzen actualment SGBD relacionals amb objectes:
- www.ibm.com
- www.informix.com
- www.microsoft.com
- www.oracle.com
- www.sybase.com

b) Empreses que comercialitzen SGBD orientats a l'objecte:
- www.gemstone.com
- www.objectdesign.com
- www.poet.com

Exercicis d'autoavaluació

1. Enumereu els diferents models pensats per a emmagatzemar dades en suports no volàtils que han aparegut al llarg de la història de la informàtica.

2. Indiqueu algunes característiques importants i definitòries de cada un dels models enumerats a l'exercici anterior. Entenem per *característiques definitòries* allò que han aportat com a innovació i millora en el camp de la gestió de les bases de dades.

Solucionari

Exercicis d'autoavaluació

1. Una possible relació cronològica dels diferents models apareguts al món de les bases de dades és la següent:

- Fitxers clàssics (encara que pròpiament no són un model de bases de dades, sí que són una primera estructura d'emmagatzematge de dades)
- Model jeràrquic
- Model en xarxa
- Model relacional
- Model de fitxers invertits (pràcticament contemporani dels principis del model relacional)
- Models semàntics
- Model orientat a l'objecte
- Model relacional estès

2. Les característiques més importants de cada un d'aquests models són les següents:

- Fitxers clàssics: són el primer sistema d'emmagatzemar dades en mitjans no volàtils. Van definir el fitxer com un conjunt de registres i el registre com un conjunt de camps.
- Model jeràrquic: pràcticament va ser el primer model de bases de dades, suportat per un programari específic, l'SGBD, que incorporava la majoria de funcionalitats presents avui dia en tots els gestors de bases de dades. Una característica fonamental és que allibera els programes d'haver de tractar tots els detalls de l'emmagatzematge de les dades en el nivell físic.
- Model en xarxa: primer intent reeixit de definir un model de bases de dades a partir del consens dels experts i crear un estàndard, que va ser adoptat per molts SGBD. Suport d'un model de dades estructuralment ric, com és la xarxa.
- Model relacional: model basat en una clara i sòlida teoria, fonamentada en l'àlgebra i la lògica. Ocultació pràcticament total dels detalls d'implementació física i obtenció d'un model i llenguatge d'alt nivell acostat a l'usuari final. Cerques de dades emprant índexs de bon rendiment.
- Model de fitxers invertits: tècniques de cerques de dades amb combinacions de més d'un índex.
- Models semàntics: possibilitat de modelització de tota una sèrie de característiques semàntiques addicionals, que dóna com a resultat un model semànticament més ric i, per tant, més proper al món real.
- Model orientat a l'objecte: suport del model orientat a l'objecte en el món de les bases de dades. Augmenta la capacitat de modelització en poder representar més naturalment els objectes del món real.
- Model relacional estès: proporciona *grosso modo* les prestacions dels models relacionals i orientat a l'objecte junts. S'estructura amb un nucli central relacional potent al qual es poden afegir diferents extensions clarament orientades a l'objecte per a suportar i gestionar tipus de dades complexos.

Glossari

fitxer invertit
Fitxer amb nombrosos camps indexats. En el límit, tots els camps poden estar indexats (fitxer totalment invertit). Permeten de combinar índexs en una cerca i tenen molt bon rendiment per a aquest tipus de peticions.

fitxers encadenats
Conjunt de fitxers conceptualment relacionats pel fet de contenir informacions complementàries. Els programes poden reconstruir la informació que busquen passant d'un fitxer a un altre a través d'adreces o apuntadors.

model CODASYL
Vegeu *model en xarxa*.

model de bases de dades
Mecanisme conceptual per a representar l'emmagatzematge de les dades en mitjans no volàtils.

model en xarxa
Model de bases de dades que estructura les dades en xarxes. Va ser estandarditzat pel comitè CODASYL, per això també se'l coneix amb aquest nom. Aquest model va ser implementat per nombrosos SGBD comercials.

model jeràrquic
Model de bases de dades que estructura les dades en jerarquies o arbres. És el model suportat principalment pel producte IMS de la companyia IBM.

model orientat a l'objecte
Model de dades, i posteriorment de bases de dades, que modelitza la realitat com un conjunt d'objectes de qualsevol complexitat.

models postrelacionals
Models de bases de dades concebuts com una millora o extensió del model relacional, principalment en la línia de proporcionar més potència semàntica al model.

models prerelacionals
Models de bases de dades anteriors a l'aparició, acceptació i difusió del model relacional.

model relacional
Model de bases de dades que estructura les dades en relacions, tuples i atributs i segueix les indicacions de la proposta seminal d'E.F. Codd.

model relacional estès
Model que combina l'enfocament del model relacional amb característiques similars a les del model orientat a l'objecte.

models semàntics
Models de bases de dades amb capacitat de modelitzar una sèrie de característiques semàntiques no presents en els models relacional i anteriors.

Bibliografia

Bernstein, P. i altres (1998). "The Asilomar Report on Database Research". *SIGMOD Record* (vol. 27, núm. 4, desembre, pàg. 74).

Cattell, R.G.G.; Barry, D.K. (2000). *The Object Data Standard: ODMG 3.0*. San Francisco: Morgan Kaufmann Publishers.

Codd, E.F. (1970, juny). "A Relational Model of Data for Large Shared Data Banks". *Communications of the ACM* (vol. 13, núm. 6, pàg. 377).

Date, C.J. (1995). *An Introduction to Database Systems* (6a. ed.). Reading: Addison-Wesley.

DeWitt, D. i altres (1991, abril). "El manifiesto del sistema de base de datos orientado al objeto". *Novatica* (vol. 17, núm. 91, pàg. 7).
Article publicat a *Novatica*, traducció al castellà de l'original anglès.

Melton, J.; Simon, A. (2001). *SQL:1999 – Understanding Relational Language Components*. San Francisco: Morgan Kaufmann Publishers.

Stonebraker, M. i altres (1991, abril). "Manifiesto del sistema de bases de datos de tercera generación". *Novatica* (vol. 17, núm. 91, pàg. 19).
Article publicat a *Novatica*, traducció al castellà de l'original anglès.