Les dades: conceptes introductoris

Rafael Camps Paré

PID_00171642

UOC
Universitat Oberta de Catalunya

Índex

Introducció

Les dades que s'utilitzen en els sistemes d'informació (SI) s'acostumen a emmagatzemar en **bases de dades (BD)**. Per a poder parlar i raonar amb certa propietat sobre les BD, ens convindrà tenir clar què són les **dades** i la **informació**, abstraccions que els informàtics representem físicament sobre dispositius d'emmagatzematge extern no volàtil. Amb aquest objectiu, haurem d'adquirir algunes nocions teòriques fonamentals i disposar d'eines formals en què basar-nos.

En aquest mòdul didàctic estudiarem els elements bàsics del món de les representacions informàtiques, i la seva correspondència amb el món real i amb el món de les abstraccions. Introduirem els termes més habituals i els conceptes fonamentals sobre dades i informació, que farem servir en la resta de l'assignatura per a estudiar les BD.

Objectius

En els materials didàctics d'aquest mòdul l'estudiant trobarà les eines indispensables per a assolir els objectius següents:

1. Saber situar els termes bàsics més habituals en el camp de les dades i la informació (atribut, clau, entitat, fitxer, base de dades, suport, etc.) en el marc teòric construït en l'assignatura.

2. Saber distingir clarament el món de les representacions sobre suports físics informàtics del món de les concepcions o abstraccions.

3. Poder enumerar els tipus bàsics d'accés a les dades i veure els sistemes d'organització com a mitjans per a fer-los eficients.

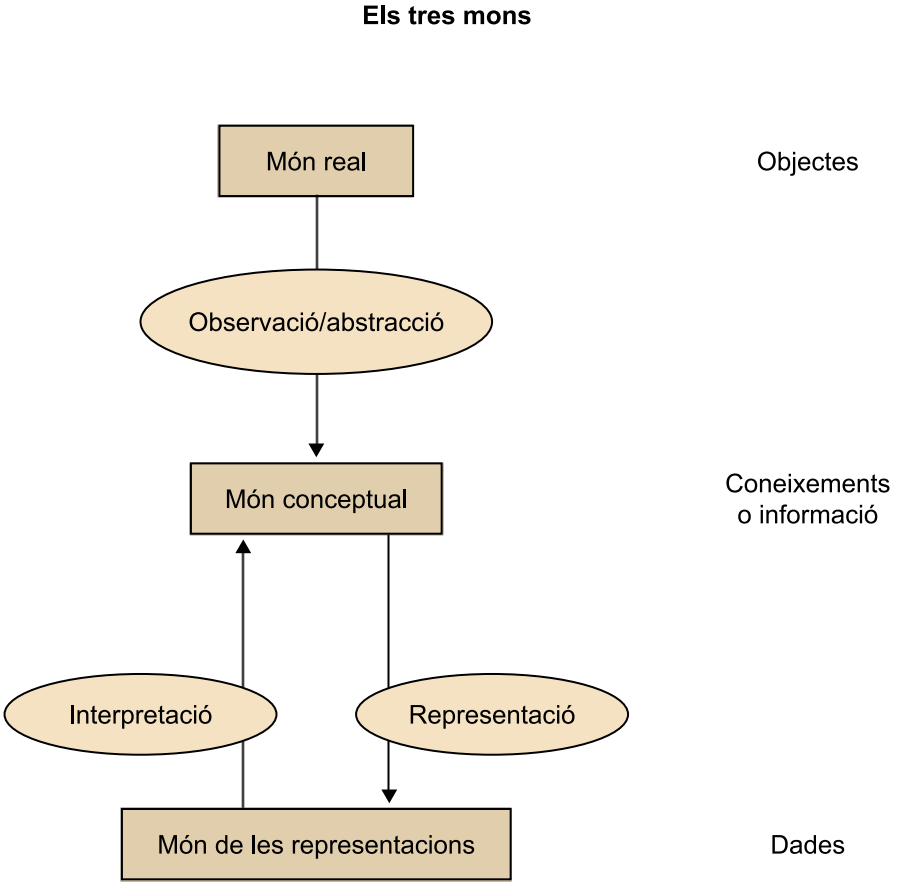
4. Entendre que les representacions informàtiques es poden estudiar des d'un nivell, o punt de vista, purament lògic, allunyat de la realització física (implementació), o bé des d'un nivell físic. En aquesta assignatura adoptarem bàsicament un punt de vista lògic.

5. Ser capaç de descriure i avaluar les característiques bàsiques dels suports de les memòries persistents per a l'emmagatzematge de dades.

1. Els tres mons: el real, el conceptuali el de les representacions

Per a tenir un marc on situar els termes i els conceptes que explicarem en l'assignatura, distingirem tres àmbits diferents:

- El món real amb els objectes del nostre interès.
- El món de les conceptualitzacions lògiques.
- El món de les representacions informàtiques.



1.1. La realitat: els objectes

Per tal d'analitzar o construir un sistema d'informació[1] (SI) determinat, ens cal conèixer el món real al qual aquest SI ha de fer referència o modelitzar; així, el nostre món real podrà ser un hospital, una empresa distribuïdora de productes alimentaris, la matriculació dels estudiants d'una universitat, etc.

(1)Un SI recull, emmagatzema i distribueix informació sobre l'estat d'un domini.

Exemples d'objectes concrets

El malalt Joan Garcia, el llit 34 de la segona planta, el magatzem de Sòria, el camió B-3452-AG, l'alumna Maria Pi, l'assignatura *Química I*, la malaltia meningitis, la devolució d'una

comanda concreta, un determinat accident de trànsit, són alguns exemples d'objectes que pertanyen al món real.

El **món real**, la part de la realitat que ens interessa, és el que percebem amb els nostres sentits i és compost per objectes concrets, físics o no.

1.2. Les concepcions: la informació

Observant el món real, els humans som capaços de deduir-ne coneixements, informació. L'observació dels objectes del món real ens porta a fer-ne l'anàlisi i la síntesi; després, n'obtenim abstraccions, en fem classificacions (podem saber que dos objectes són de la mateixa classe malgrat que siguin diferents), en deduïm propietats i interrelacions, etc.

De l'observació n'obtenim informació

L'observació del camp de la matriculació en una universitat ens permetre conèixer diferents classes o tipus d'objectes, com ara l'*estudiant* o l'*assignatura*. Deduïm que tot estudiant tindrà les propietats (són abstraccions) *data de naixement*, *DNI*, *nom*, etc. i així obtenim informacions com les següents: l'estudiant de nom Joan Garcia té el DNI 34.567.854 i el seu any de naixement és el 1979.

El conjunt dels coneixements obtinguts observant un món real, l'anomenem **món conceptual** o **món de les concepcions**. En l'esfera de les concepcions construïm un model abstracte, conceptual, del món real, i això ens ajuda a raonar i a expressar-nos.

El **procés d'observació/abstracció** és bàsicament un procés per a modelitzar l'estructura, les propietats i el funcionament de la realitat.

De fet, hi ha diferències entre *coneixement* i *informació*. La **informació** és un coneixement transmissible, és a dir, que es pot representar. Els únics coneixements que ens interessaran aquí són, doncs, les informacions.

Un mateix món real pot ser vist, concebut, modelitzat, de maneres diferents per diferents observadors (fins i tot per un mateix observador) segons el seu entorn o marc de referència. Per exemple, no veu de la mateixa manera l'àmbit de la gestió d'un centre universitari un professor que un administratiu de secretaria. Tenen marcs de referència diferents. No estan interessats en els mateixos conceptes. El professor, a diferència de l'administratiu, no necessitarà conèixer l'import de la matrícula, no voldrà distingir les abstraccions *estudiant amb beca* i *estudiant sense beca*. Els professors estaran interessats en la qualificació numèrica, mentre que el servei administratiu potser només tindrà en compte la forma textual de la qualificació.

Veiem, doncs, que en el pas del món real al de les concepcions hi ha pluralisme. L'observació i l'anàlisi d'una mateixa part d'una organització o empresa poden portar a concepcions diferents, totes igualment vàlides i que poden haver de coexistir.

1.3. Les representacions: les dades

El món de les concepcions o dels coneixements és un món mental. Però per a poder treballar amb aquests coneixements i poder comunicar-los, necessitem projectar els pensaments a l'exterior representant-los físicament d'alguna manera. Aquest és el **món de les representacions**.

Representació de coneixements

Podem representar coneixements escrivint a mà sobre un paper, gravant bytes en un disc magnètic segons un format i una codificació determinats, etc.

Nosaltres aquí ens ocuparem de les representacions informàtiques, i parlarem de dades, fitxers, bases de dades, registres, camps, *bytes*, discs, etc.

Donem el nom de **dades** a les representacions físiques dels coneixements que tenim dels objectes del món real. El pas dels coneixements a les dades, el pas d'una concepció a una representació informàtica, no és automàtic. És un procés humà, un procés de disseny.

Òbviament, en aquest cas, com en el cas del pas del món real al de les concepcions, també hi ha pluralisme. Un mateix conjunt de coneixements es pot representar de moltes maneres, per exemple: en forma de base de dades relacional o com a fitxers tradicionals, amb vectors o sense, amb longitud fixa o variable, amb codificació ASCII o EBCDIC, etc. Una visió o concepció del món real d'un hospital, d'una universitat o d'una distribuïdora de productes podrà ser representada de moltes maneres sobre suports físics infor-màtics.

Sense cap mena de dubte, les feines més importants de l'analista/dissenyador de SI o d'aplicacions informàtiques són les següents:

Dissenys diferents

Es poden fer molts dissenys diferents de representació informàtica corresponents a un únic model conceptual d'una realitat. Tots poden representar la mateixa realitat, però tindran una eficiència diferent segons la utilització que se'n faci.

1) Analitzar els objectes del món real, fer-ne abstraccions i obtenir-ne una concepció lògica, un model conceptual.

2) Dissenyar una representació informàtica concreta que es pugui tractar eficientment.

El fet de saber observar la realitat, fer-ne les abstraccions lògiques més escaients, l'habilitat per a l'anàlisi i la síntesi, esdevenen les qualitats fonamentals que ha de tenir el desenvolupador de SI. I aquestes qualitats s'han d'educar i conrear.

Evolució del disseny d'aplicacions

El pas d'un món conceptual a un món de representacions informàtiques s'ha fet més senzill a mesura que la tecnologia informàtica avançava i se'n simplificava la utilització. Els anys seixanta i setanta el desenvolupador d'aplicacions es veia obligat a tenir en compte una multitud de detalls físics de la representació informàtica. Actualment, la simplificació del procés de disseny de la representació fa que el procés d'observació/abstracció esdevingui la tasca principal del desenvolupador de SI.

1.4. La interpretació

Acabem de veure el camí que ens porta de la realitat als coneixements, i d'aquests a les dades o representacions. Però ens farà falta poder interpretar la representació. El procés invers al de representació, l'anomenem **interpretació**.

Per a reflexionar

Com es pot obtenir coneixements o informació, d'una representació?

Si veiem una dada, una representació extreta d'una base de dades relativa a la matriculació d'estudiants, que consta de la sèrie de símbols: 1 9 9 9, no en podrem obtenir cap informació si no sabem si representa l'any de matriculació, l'any de naixement, l'import de la matrícula, el número de la matrícula, etc., i en qualsevol cas no sabrem de quin estudiant concret (de quin objecte del món real) es tracta. Veiem, doncs, que per a poder interpretar les dades s'ha de saber, a més, a qui i a què (a quins conceptes) fan referència.

Hem dit que una informació és un coneixement que es pot representar, però ara, mirant el camí invers, podrem dir que la **informació** és el significat que donem a les dades.

2. El món conceptual: entitats i atributs

Com ja hem vist, el món conceptual és el món de les abstraccions lògiques, el domini de la informació. Aquest camp és el fonamental per a concebre (analitzar i dissenyar) el SI.

Vegeu també

Per a ampliar la informació sobre el món conceptual, consulteu el subapartat 1.2 d'aquest mòdul didàctic.

2.1. La informació: expressió lingüística

Quan parlem d'informació, ens movem en l'àmbit de les concepcions. Tota informació es refereix a un **objecte** i ens en descriu una **propietat**. Per exemple, una informació sobre un estudiant (l'objecte) podria ser la propietat "va néixer el 1979".

En termes lingüístics, una **informació** (un coneixement elemental) es pot expressar amb un **subjecte** (l'estudiant concret) i un **predicat** ("va néixer el 1979"). El predicat és format pel verb i el complement.

Amb connectors lògics (o, i, no) podem expressar coneixements més complexos. Per exemple, "aquest estudiant concret s'anomena Joan Garcia i va néixer el 1979".

2.2. Entitats, atributs i valors

Des d'un punt de vista informàtic fem servir uns termes diferents dels utilitzats en lingüística.

Anomenem **entitats** els objectes que conceptualitzem com a distingibles els uns dels altres (és a dir, que són identificables) i dels quals ens interessen algunes propietats. El terme *entitat* es correspon amb el terme **subjecte** del camp de la lingüística. És la conceptualització de l'objecte al qual fa referència la informació.

El **predicat** és la propietat descrita, i les seves dues parts, verb i complement, les anomenem **atribut** (*any de naixement*) i **valor** (*1979*), respectivament.

Figura 2

Els components d'una informació elemental		
Subjecte	**Predicat**	
Aquest estudiant	va néixer l'any	1979
▼ Entitat	▼ Atribut	▼ Valor

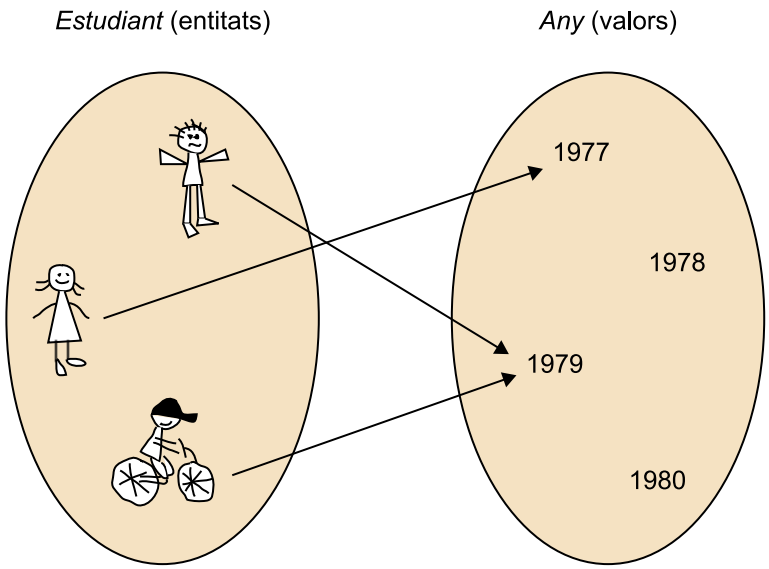
Tota informació es caracteritza pels tres elements següents: entitat, atribut i valor. Si només coneixem l'atribut (*any de naixement*) i el valor (*1979*), no tenim informació, ja que no sabem a quina entitat (*estudiant*) fa referència. Si no coneixem l'atribut, no sabrem a què fa referència el valor (el número 1979és l'any de naixement? O potser es tracta del número de matrícula o de l'alçada en mil·lí-metres?).

Per a aclarir i precisar el significat d'aquests tres termes, *entitat*, *atribut* i *valor*, utilitzarem conceptes elementals de la teoria de conjunts.

Situats en aquest marc de la teoria de conjunts podem veure l'atribut *any de naixement* com una correspondència entre els estudiants i els anys del calendari. Cada estudiant té un sol any de naixement i diferents estudiants poden tenir el mateix any de naixement. És a dir, la correspondència entre els estudiants i els anys pot ser vista com una **aplicació** (en el sentit de les matemàtiques) del conjunt dels estudiants sobre el conjunt dels anys.

Figura 3

L'atribut *any de naixement*



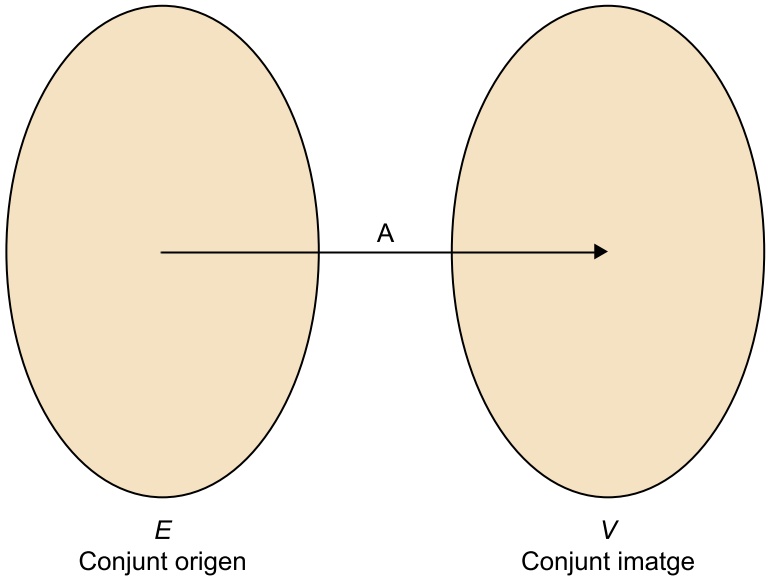
Si *E* és un conjunt d'entitats individuals (conceptualitzacions dels objectes del món real) i *V* és un conjunt de valors, podem definir l'**atribut *A*** com l'aplicació de *E* sobre *V*. Si expressem l'aplicació en termes d'una funció, direm que *V* 5 *A*(*E*).

{ Entitat } →{ Valor }

Atribut

Figura 4

L'atribut *A* com a aplicació de *E* sobre *V*



Per a un mateix conjunt origen podem definir diferents aplicacions sobre diversos conjunts imatge. Dit d'una altra manera, una entitat pot tenir més d'un atri-but.

Exemple d'entitat multiatribut

Suposem que el que cal saber dels estudiants és el número de matrícula, el número de DNI, el nom i l'any de naixement.

Les entitats tindran quatre atributs i un valor per a cada atribut. Representem ara, en la figura 5 de la pàgina següent, els atributs com a aplicacions.

2.3. El temps

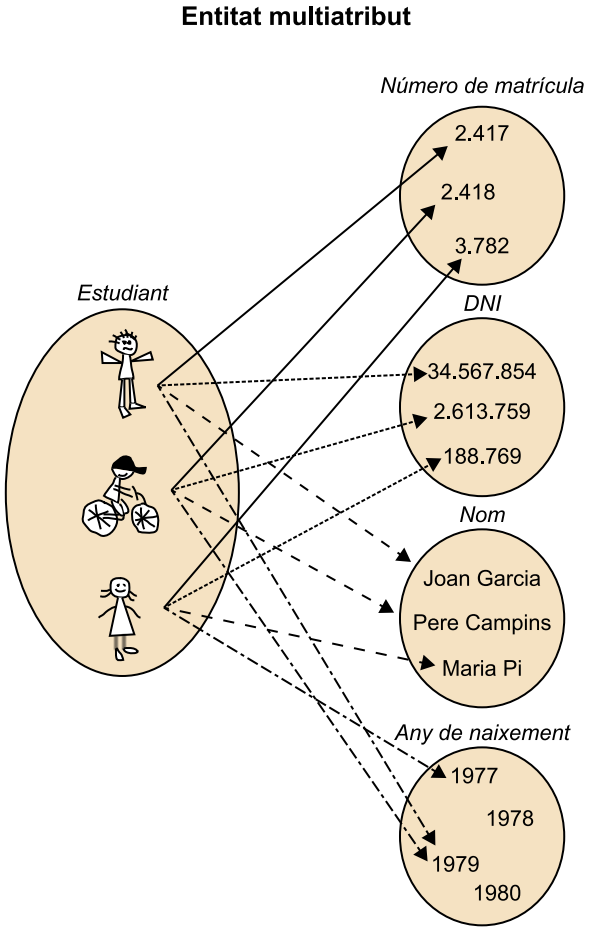
Realment la informació no és independent del temps. El sou d'un empleat, l'alçada d'un estudiant, el nombre de fills, etc., varien en el temps. En un SI ens pot interessar mantenir el valor actual dels atributs, però potser també

hi volem incloure valors anteriors. Així, doncs, el valor *3* de l'atribut *nombre de fills* d'en Joan Garcia no constitueix una informació prou completa, si no sabem a quin moment correspon.

Fins i tot els atributs estables, com per exemple el DNI d'un estudiant, poden canviar en el món real. Però encara que no sigui així, en un SI tot atribut pot canviar de valor en el temps. Per exemple, hem introduït un DNI erroni, i ho detectem i el canviem al cap d'uns mesos. Com que durant aquests mesos hem pogut comunicar el DNI erroni al món exterior, ens convindria tenir registrat en el SI els dos números de DNI i la data del canvi.

En general, per a tenir ben caracteritzada una informació no n'hi ha prou amb els tres elements, **entitat**, **atribut**, **valor**, sinó que ens farà falta el **temps**.

Figura 5



I potser no en tindrem prou amb només un temps, sinó que ens caldran diversos temps: el moment en què el canvi es va produir al món real, el moment en què es va introduir al SI, etc.

Tant les tècniques de modelització conceptual que es fan servir en l'àmbit professional, com les bases de dades i els fitxers actuals, no donen gaires facilitats per a considerar el temps com un element caracteritzador de la informació. En els anys vinents això canviarà, però mentrestant, la responsabilitat d'incloure el temps als SI correspon al dissenyador. Per exemple, es podria dissenyar un fitxer que contingués les dades actuals, sense cap atribut que fes referència al temps, i un fitxer històric en què cada enregistrament d'informació anés acompanyat d'una data i una hora.

2.4. Dominis i valors nuls

El conjunt de tots els valors vàlids, o legals, que pot arribar a tenir un atribut, rep el nom de **domini de l'atribut**.

Pot passar que el valor d'un atribut determinat d'alguna entitat individual sigui desconegut o no existeixi. Llavors direm que el **domini d'aquest atribut accepta el valor nul**.

Exemple

El nombre enter 981 o la sèrie de símbols A-321.6, per exemple, no formen part del domini de l'atribut *any de naixement* dels estudiants del nostre món real.

D'un determinat estudiant, en podem desconèixer el nom o l'any de naixement. O pot ser que algun estudiant no tingui DNI. En aquests casos, en definir el domini de l'atribut haurem de dir si hi acceptem el valor nul o no.

Nota

No s'ha de confondre el valor nul amb el zero o amb els espais en blanc. Per exemple, el color d'un import desconegut no és zero.

2.5. Identificadors i claus

Recordem que en la teoria de conjunts s'anomena **aplicació injectiva** aquella aplicació en la qual a cada element del conjunt imatge li correspon un element del conjunt origen com a màxim. Així, l'atribut *any de naixement* no és una aplicació injectiva, perquè hi poden haver diferents estudiants que hagin nascut el mateix any.

Però l'atribut *número de matrícula* sí que és una aplicació injectiva, ja que al nostre món real, l'àmbit de la matriculació d'estudiants, no s'accepta que dos estudiants tinguin el mateix número de matrícula, perquè precisament s'utilitza per a poder distingir uns estudiants dels altres, és a dir, per a identificar-los.

Els atributs que concebem com a aplicacions injectives s'anomenen **identificadors**.

Els atributs són identificadors o no, segons els objectes que ens interessa modelar. Si ens referim a les persones, llavors el *DNI* d'una persona és un identificador. Però si el món real que considerem és relatiu a assegurances d'accidents,

els objectes del nostre interès seran els accidents de trànsit, i aleshores l'atribut *DNI* (el DNI del conductor) no serà un atribut identificador, ja que dos accidents podrien ser del mateix conductor.

Una entitat pot tenir més d'un identificador o bé no tenir-ne cap. Així els estudiants poden quedar identificats tant pel *número de matrícula* com pel *DNI*. Però pot passar que l'entitat no tingui cap atribut identificador. Així, per exemple, si considerem els objectes *ciutat* amb els atributs *nom ciutat, nombre d'habitants, país* i *superfície d'arbrat*, ens trobem que el *nom ciutat* no identifica una ciutat, ja que hi poden haver ciutats amb el mateix nom a diferents països. Llavors, per a identificar les ciutats haurem d'utilitzar conjuntament la parella d'atributs *país* i *nom ciutat*.

Conjunts d'atributs

Com ja hem vist anteriorment, les entitats corresponen a objectes que podem identificar o distingir. Per a distingir els estudiants, podem fer servir l'atribut *número de matrícula*, ja que és un atribut identificador. Però en el cas dels accidents de trànsit, el *DNI* del conductor no ens identifica l'accident. Com que no hi ha un atribut identificador, podríem identificar els accidents amb la parella *DNI del conductor* i *data i hora*, o potser el conjunt d'atributs *país*, *nom ciutat*, *carrer*, *número*, *data i hora*, o qualsevol altre conjunt d'atributs que ens diferenciés els accidents.

Vegeu també

Vegeu el concepte d'entitat al subapartat 2.2 d'aquest mòdul didàctic.

Tot atribut o conjunt d'atributs que permet identificar les entitats individuals rep el nom de **clau**.

Nom de les ciutats

El nom de les ciutats no és suficient per a identificar-les, perquè un sol nom pot referenciar diverses ciutats; per exemple, hi ha una ciutat anomenada *Barcelona* aVeneçuela.

En el cas dels estudiants, tant l'atribut *número de matrícula* com el *DNI* són claus i cadascun és un atribut identificador. En el cas de les ciutats, la parella d'atributs *país* i *nom ciutat* constitueix una clau, però cap dels dos no és identificador.

Nota

Tot atribut identificador és una clau, però no tota clau és un atribut identificador.

Altres significats del terme *clau*

En el camp dels fitxers i les bases de dades (BD), el terme *clau* s'utilitza també amb altres significats. Per exemple, s'acostuma a anomenar *clau* l'atribut o conjunt d'atributs que es fa servir per a efectuar una cerca en un fitxer.

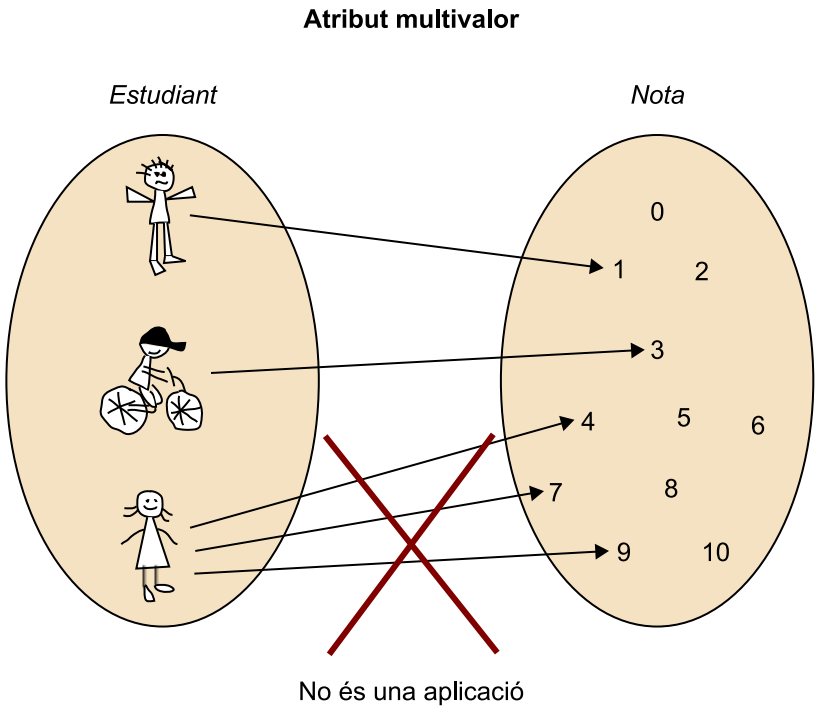
Podem cercar dintre d'un fitxer els estudiants que tenen l'atribut *nota* igual a 8,5, és a dir, fent servir com a "clau" de cerca la nota. Però, òbviament, la nota no és una clau en el sentit que aquí donem a aquest terme.

2.6. Atributs multivalor

Com que l'atribut és una aplicació entre conjunts, a cada entitat li pot correspondre com a màxim un sol valor. En conseqüència un atribut no podrà ser multivalor (o multivaluat).

Per exemple, no serà possible un atribut *nota* tal que cada estudiant pugui tenir més d'una nota, ja que llavors no seria una aplicació sinó una correspondència.

Figura 6



Aquesta restricció és pròpia del model relacional i ha estat seguida al peu de la lletra per la majoria dels sistemes de gestió de BD del mercat. Com que aquí ens mourem dintre l'àmbit del model relacional, no acceptarem els atributs multivalor. A la pràctica s'utilitzen sovint, especialment en els fitxers clàssics.

2.7. L'entitat: instància i tipus

Fins aquí hem fet servir el terme *entitat* per a anomenar la conceptualització d'un objecte del món real, una instància: un estudiant concret, un accident concret, etc. Però també el farem servir per a anomenar l'entitat genèrica, el tipus, l'abstracció *estudiant* o *accident* (no un estudiant o un accident concret, ni el conjunt dels estudiants o dels accidents). Totes les entitats *estudiants* són elements del conjunt d'estudiants. Tots els alumnes són individus o instàncies del mateix tipus, són instàncies del tipus d'entitat *estudiant*.

Així, doncs, el terme *entitat* tindrà dues[2] accepcions:

1) L'entitat com a individu, o **instància**.
2) L'entitat com a classe, o **tipus**.

Totes les instàncies d'un mateix tipus, totes les entitats individuals d'una mateixa entitat genèrica, tenen els mateixos atributs.

[(2)] Alguns autors també inclouen una tercera accepció i consideren una entitat com el conjunt d'instàncies d'una entitat tipus.

Tots els estudiants tenen número de matrícula, DNI, nom i data de naixement, i és per això que els considerem de la mateixa entitat tipus *estudiant*.

Habitualment només concretarem quina de les dues accepcions del terme *entitat* fem servir –instància o tipus– quan no quedi prou clar pel context.

3. El món de les representacions

Ara veurem els principals conceptes i termes que s'utilitzen en el camp de les representacions informàtiques, el món de les dades.

3.1. La representació tabular

La informació pertany al domini conceptual o mental. Però per a transmetre-la i processar-la necessitem representar-la físicament. La representació informàtica d'una informació elemental s'anomena **dada**. El món de les representacions serà el món de les dades i per a descriure'l parlarem de fitxers, registres, camps, BD, suports, etc.

La figura 5 és, en realitat, una representació gràfica, no informatitzada, de la informació dels estudiants. S'ha fet utilitzant aquest paper com a suport. Però, com hem pogut observar, amb tantes fletxes i conjunts, no resulta gaire còmoda per a ser processada o transmesa, especialment en un cas real, en què hi haurien desenes d'atributs i milers d'estudiants. Resulta molt més senzilla una **representació tabular** amb una fila per a cada entitat individual i una columna per a cada atribut.

Vegeu també

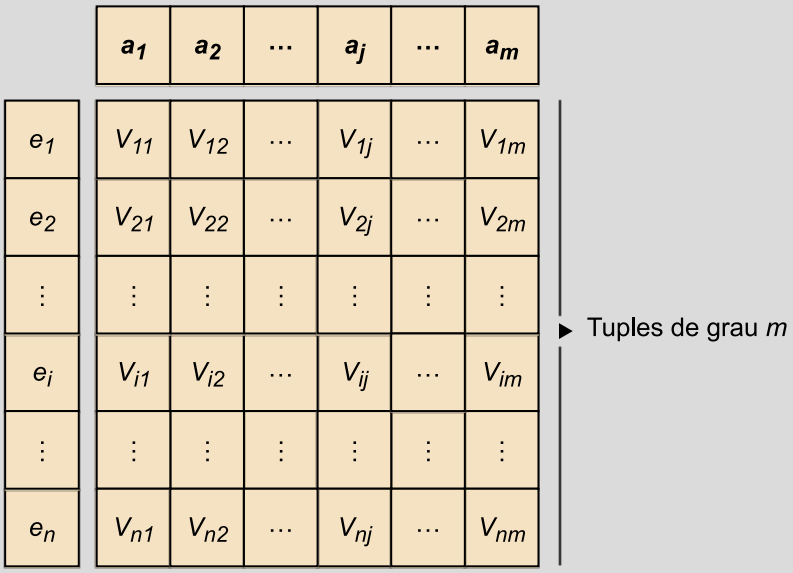
Podeu veure la figura 5 en el subapartat 2.2 d'aquest mòdul didàctic.

Representació tabular de la informació de la figura 5

Estudiant	*Número de matrícula*	*DNI*	*Any de naixement*	*Nom*
	2.417	34.567.854	1979	Joan Garcia
	3.782	188.769	1977	Maria Pi
	2.418	2.613.759	1979	Pere Campins

La taula anterior és una representació tabular, formalment molt similar a la representació típica en fitxers informàtics. És com un fitxer de dades d'estudiants que té un registre per a cada estudiant (en aquests moments només en té tres) amb quatre camps per a cada registre.

Una **representació tabular** d'un conjunt de *n* entitats *ei* on cadascuna de les quals té *m* atributs *aj* és, de fet, un conjunt de *n* tuples de grau *m* formades pels valors *vij*.



L'esquema (format o capçalera) d'aquesta taula es podria escriure de la ma-nera següent: E(a_1, a_2, ..., a_j, ..., a_m). Podríem considerar-ho una representació de l'entitat tipus E, és a dir, el tipus de les entitats instància e_i, *i* 5 1 a *n*. Totes les *ei* tenen la mateixa estructura; en altres termes, tenen els mateixos atributs a_j, on *j* 5 1 a *m*.

3.2. Fitxers, registres i camps

Tradicionalment les dades han estat emmagatzemades en fitxers sobre suports magnètics. El terme *fitxer* és emprat en l'àmbit dels sistemes operatius (SO) en un sentit molt més genèric que aquí. Evidentment, en aquesta assignatura no tractarem de fitxers de programes, però tampoc no parlarem de fitxers de text lliure, fitxers de gràfics, etc. Tractarem només de fitxers de dades estructurades en registres i de bases de dades, que és el que normalment s'utilitza en els SI.

Un **fitxer de dades** és una representació informàtica equivalent a la representació tabular:

a) La representació d'una entitat, l'equivalent a una fila de la taula, rep el nom de **registre**.

b) La representació del valor d'un atribut d'una entitat s'anomena **camp**.

El conjunt de camps constitueix el registre, i el conjunt de registres constitueix el fitxer.

Podem considerar que en el món dels fitxers tradicionals de dades l'equivalent dels atributs són les capçaleres dels camps.

Les dades (les informacions elementals) de cadascun dels nostres estudiants estaran emmagatzemades en una estructura de quatre camps, un per a cada atribut. Cada camp contindrà un valor, una dada. El conjunt de les dades d'un estudiant forma el registre –la "fitxa"– de l'estudiant, i el conjunt dels registres dels estudiants formen el fitxer d'estudiants.

Figura 8

Fitxer d'estudiants				
número de matrícula	*DNI*	*any de naixement*	*nom*	▶ Nom o capçalera dels camps
2.417	34.567.854	1979	Joan Garcia	
3.782	188.769	1977	Maria Pi	▶ Registres
2.418	2.613.759	1979	Pere Campins	

***Camp*: un terme polivalent**

El terme *camp* s'utilitza, en la pràctica, en diversos sentits semblants, cosa que pot portar a confusió. És freqüent utilitzar-lo en el sentit de la representació d'un valor, però sovint s'utilitza amb el significat del continent, és a dir, el lloc on s'emmagatzema el valor, i també és freqüent fer-lo servir per a denominar la capçalera. Així, es parla del domini d'un camp, de camps identificadors, de claus formades per un camp identificador o diversos camps no identificadors, de camps multivalor, etc.

3.3. Bases de dades

Considerem ara un món conceptual format per diferents entitats tipus. La seva representació informàtica podria fer-se mitjançant un conjunt de fitxers.

De moment, en aquest mòdul introductori donarem el nom de **base de dades** (BD) a un conjunt de fitxers de dades interrelacionats.

Suposem que els tipus d'objectes del nostre interès són *estudiants*, *assignatures* i *professors*, i que els atributs de les tres entitats són els següents:

a) Estudiant: *número de matrícula, DNI de l'estudiant, any de naixement, nom de l'estudiant*.

b) Assignatura: *codi, nom de l'assignatura, crèdits*.

c) Professor: *DNI del professor, nom del professor, despatx*.

Podrem representar aquestes entitats mitjançant tres fitxers, un per a cada entitat, amb els camps corresponents als atributs. Però hi falta la informació que permet interrelacionar les entitats entre si. Suposem que aquestes interrelacions són les següents:

Activitat

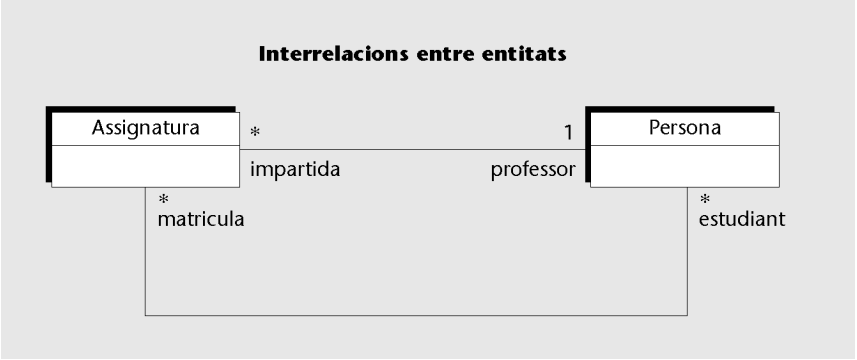
Quines són les interrelacions entre estudiants, assignatures i professors?

1) Tot estudiant pot cursar més d'una assignatura i, evidentment, tota assignatura pot ser cursada per molts estudiants.

2) Tota assignatura és donada per un sol professor, però cada professor pot donar diverses assignatures.

3) Suposem també que ens interessa la nota que l'estudiant té de cada assignatura. Es tracta d'un atribut, *nota*, que no és pròpiament de l'estudiant (ja que en té una per assignatura) ni de l'assignatura (ja que en té tantes com estudiants la cursen). És com si fos un atribut de la interrelació entre assignatura i estudiant.

Figura 9



Fixem-nos ara en els problemes que planteja la representació informàtica d'aquestes interrelacions:

a) La **interrelació entre assignatures i professors** es podria representar afegint als registres de les assignatures un camp *DNI del professor* amb el valor del DNI del professor que la dóna. Així, una assignatura tindria un sol professor i un mateix professor podria aparèixer en diferents assignatures.

b) La **interrelació entre assignatures i estudiants** és més complexa i es podria representar mitjançant camps complexos de tipus vector; així s'imitarien els atributs multivalor, aquí prohibits. Però podríem optar per tenir un altre fitxer (una nova entitat tipus específica per a descriure aquesta interrelació), el qual tindria els camps següents, tots monovalor: *codi, número de matrícula, nota*. Aquest nou fitxer tindria un registre per a cada parella realment existent d'estudiant-assignatura.

Hem representat la informació del nostre món real amb quatre fitxers de dades. Si haguéssim d'escriure un programa per a mostrar una llista de notes acompa-nyades del nom de l'estudiant, el nom de l'assignatura i el nom del

professor, hauríem de fer que llegís i interrelacionés tots quatre fitxers. Els programes que creen o actualitzen aquests fitxers no poden ser gaire senzills, ja que han de mantenir la coherència del conjunt.

Per exemple, en suprimir un professor del fitxer de professors s'ha d'eliminar també de les assignatures que donava, o en incloure la nota d'un estudiant no s'ha de posar un codi d'assignatura que no existeixi al fitxer d'assignatures, etc.

Veiem, doncs, que els **conjunts de fitxers interrelacionats** ens plantegen certes dificultats. Els programaris tradicionals de gestió de fitxers, els *File Management Systems*, no s'ocupen de les possibles interrelacions entre fitxers, i les deixen en mans dels usuaris informàtics. Al final dels anys setanta van començar a sortir al mercat programaris especialitzats en aquests conjunts complexos de dades sota el nom de *Database Management Systems* o *Sistemes de gestió de BD* (SGBD). Els SGBD són bastant més sofisticats que els sistemes de gestió de fitxers, i el seu objectiu és facilitar l'ús de les BD, el disseny, la programació, el manteniment, la utilització simultània per molts usuaris, etc.

3.4. L'enregistrament físic i els suports

La **memòria interna** (RAM) dels ordinadors és volàtil. Així, les dades que hi emmagatzema un programa desapareixen quan aquest acaba la seva execució.

Per a emmagatzemar les dades de manera persistent fan falta memòries externes –perifèrics d'emmagatzemament– que siguin suports físics permanents.

Potser els nostres néts o besnéts no arribaran a recordar com s'emmagatzemaven i es gestionaven les dades permanents sense els ordinadors, però per a nosaltres encara són habituals les representacions físiques sobre paper o cartolines, on les dades estan escrites amb un format determinat, amb un cert llenguatge, un tipus de lletra, tinta, etc.

Exemples

- Un llibre de registre de moviments bancaris, que té una ratlla, un registre, per a cada moviment.
- A la secretaria d'una escola hi ha un arxivador amb calaixos plens de fitxes, on s'hi enregistra la informació dels estudiants, una fitxa per estudiant. És el fitxer d'estudiants.

Sobre els suports informàtics, els programes hi escriuen registres de dades. Les dades d'un registre són gravades pel programa en un cert **format** i amb una **codificació**; per exemple, el camp *nom* podria ser de longitud variable amb un prefix que n'indiqués la longitud, i la seva codificació podria ser ASCII; el *DNI* podria estar en binari pur i ocuparia tres octets, etc.

Generalment, el dissenyador dels fitxers o la BD per a un SI concret pot decidir detalls sobre l'enregistrament de les dades. És part de l'anomenat **disseny físic**.

3.5. Organització

Les fitxes de cartró dels estudiants de la secretaria no informatitzada potser estan organitzades o col·locades per ordre alfabètic segons el nom. Per a facilitar-hi l'accés potser hi ha pestanyes separadores per les dues primeres lletres del nom. Per a poder cercar una fitxa sabent només el número de matrícula,

sense haver de mirar seqüencialment totes les fitxes dels estudiants anteriors (estan per ordre alfabètic), es podria disposar d'una llista ordenada pel número de matrícula que ens donés el nom de l'estudiant. Aquesta llista actuaria, doncs, com un índex que ens ajudaria a fer més ràpides les cerques.

Els programaris de fitxers i els de BD ens donen unes **possibilitats d'organització** semblants a les del món no informàtic i unes altres de molt més sofisticades. Són semblants a les que l'estudiant coneix com a estructures de dades en memòria interna. Són les seqüències, llistes encadenades, vectors, índexs en forma d'arbre equilibrat, *hashing*, etc. Però es tindran en compte les característiques pròpies dels suports persistents.

Activitat

Quina serà l'estructura, l'organització, que donarem a les dades en un suport informàtic? Com les col·locarem?

El dissenyador d'un SI, quan fa el disseny físic dels fitxers o de la BD, ha de prendre decisions respecte a quins sistemes d'organització s'utilitzaran.

3.6. Accés a les dades

Una cosa és com estan organitzades les dades (la col·locació) i una altra és com s'hi accedeix (l'obtenció). Totes les organitzacions accepten diverses maneres d'accedir a les dades i és el propi usuari, o potser el programari, qui escull com ho fa.

Hi ha dues formes bàsiques d'accés a les dades: l'**accés seqüencial** i l'**accés directe**. La diferència essencial és que l'accés seqüencial a un registre pressuposa l'accés previ a tots els registres anteriors, mentre que l'accés directe no. L'accés seqüencial és un accés "al següent"; en canvi, l'accés directe és un accés "al desitjat".

Una altra dicotomia usual en les formes d'accés és l'**accés per valor** i l'**accés per posició**. L'accés per valor ens porta al registre en funció del valor d'algun dels seus atributs, sense tenir en compte la posició que ocupa el registre. L'accés per posició, en canvi, ens porta a un lloc –una posició– on hi ha un registre de dades, sense tenir en compte el contingut.

Combinant les dues classificacions anteriors, tenim les quatre formes d'accés més habituals:

La posició *p*

Quan aquí parlem d'una posició *p*, no ens referim a una posició byte (el byte número *p* dins el fitxer), sinó a una posició registre. Cada registre ocupa una posició i a cada posició hi pot haver un registre. Recordeu que aquí només tractem de fitxers de dades estructurades en registres.

1) Accés seqüencial per posició (SP): després d'haver accedit a un registre que ocupa una posició, es demana accedir al registre que ocupa la posició següent. Aquest tipus d'accés era el natural en el cas de dades emmagatzemades en cintes magnètiques, però també és molt utilitzat en suports típics d'accés directe com els discs.

Per exemple, per a construir un quadre resum del fitxer d'estudiants es podria usar l'accés SP, ja que s'han de llegir tots els estudiants sense importar-ne l'ordre lògic.

2) Accés directe per posició (DP): es demana accedir al registre que ocupa la posició *p*.

Per exemple, utilitzaríem accessos directes per posició en el cas que volguéssim programar una cerca dicotòmica o una cerca *hashing*.

3) Accés seqüencial per valor (SV): després d'haver accedit a un registre es demana accedir al registre següent, respecte a l'ordre d'un atribut (camp) determinat.

Per exemple, en un accés SV per *número de matrícula*, un cop obtingut l'estudiant que té el 2.418 de número de matrícula (vegeu la figura 7), s'obtindrà l'estudiant 3.782. Seria aquest el tipus d'accés que faríem servir en un programa que subministrés una llista d'estudiants ordenada per *número de matrícula*, malgrat que el fitxer fos una seqüència ordenada per *nom*, però que estigués equipat d'un índex per *número de matrícula* (precisament aquest era el cas de la secretaria no informatitzada).

4) Accés directe per valor (DV): es demana accedir al registre que té, per a un atribut (camp) determinat, un valor donat.

Per exemple, vull accedir a les dades de l'alumne Joan Garcia (el registre on el camp *nom* val Joan Garcia).

Així, doncs, podem fer un quadre resum de les diferents formes d'accés:

Figura 10

Quatre formes d'accés		
	Per posició	**Per valor**
Seqüencial	SP	SV
Directe	DP	DV

3.7. Nivell lògic i nivell físic

L'enregistrament de les dades, l'organització i els accessos es poden veure des d'un punt de vista més o menys allunyat de la realització física.

Al món de les representacions informàtiques, s'acostuma a distingir dos punts de vista o nivells: el **nivell físic**, quan és necessari considerar la realització física, i el **nivell lògic**, quan no cal conèixer-la.

Anys seixanta i setanta

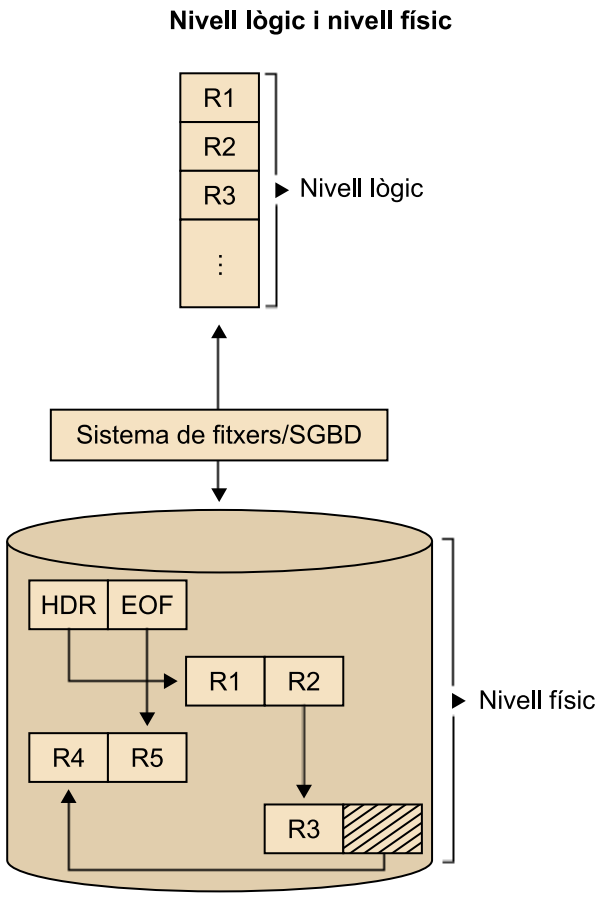
Els programaris actuals especialitzats en fitxers i BD ens donen la separació de nivells desitjada (si bé no del tot completa). Però als anys seixanta i setanta aquesta separació quasi no existia. Els programadors d'aplicacions havien d'incloure als seus programes consideracions relatives a índexs, controls de paritat, mesura de la pista del disc, etc.

Els programadors treballen a nivells diferents en funció de les seves necessitats:

1) Nivell lògic: per exemple, el programador d'aplicacions que treballa amb un llenguatge d'alt nivell com el C, el C++, el Java, etc., pot veure o imaginar que els fitxers són formats simplement per registres, l'un darrere l'altre, i que contenen camps amb lletres i números. No veu, ni li cal conèixer, la realització física que potser constarà d'encadenaments de registres físics (cadascun amb diversos registres lògics), marques separadores entre camps, compressió de dades, índexs, etc. El programador d'aplicacions treballarà a nivell lògic.

2) Nivell físic: entrem al nivell físic quan hem de considerar la realització física. Un programador de programari bàsic (SGBD, SO, etc.), un dissenyador físic d'una BD, un tècnic de sistemes que administra una BD, etc., han d'entrar al nivell físic.

Figura 11



Per a fer la programació senzilla i independent de les realitzacions, interessa que els programes no hagin de gestionar l'organització i els accessos al nivell físic, només al nivell lògic.

En aquesta assignatura farem referència quasi exclusivament al nivell lògic.

4. La memòria persistent

Abans d'acabar aquest primer mòdul farem una petita incursió en un tema de nivell molt físic: les memòries externes amb suports permanents.

4.1. Justificació de la utilització de la memòria persistent

La necessitat d'emmagatzemar les dades ens obliga a utilitzar memòries no volàtils amb suports permanents, com ara els discs magnètics, els discos òptics, memòries *flash*, cintes, etc. Però la **no-volatilitat** no és l'única raó que en justifica la utilització, ja que hi ha també la seva **gran capacitat** i el **preu baix** per *byte*.

El **principal inconvenient** d'aquests perifèrics d'emmagatzematge persistent és el **temps d'accés**, que és significativament més lent que el de la memòria interna.

4.2. Esquema de l'E/S

L'estudiant ja coneix el funcionament de la comunicació física entre els perifèrics d'emmagatzematge i la memòria interna (no persistent). Sabem que la unitat de transferència entre la memòria externa (o persistent) i la interna és el bloc. El **bloc** és allò que es llegeix o s'escriu de cop en una sola operació física d'E/S (entrada/sortida).

Per exemple, en el cas dels discs, el bloc mínim serà un sector, però s'acostuma a llegir de cop tota una sèrie de sectors.

A vegades es dóna el nom de **registre físic** al bloc, i el de **registre lògic** al que aquí anomenem simplement registre[3]. Com que la mida d'un registre sol ser molt més petita que la d'un bloc, s'agrupen els registres (lògics) en blocs (registres físics). En el món de les BD es fa servir sovint el terme **pàgina** com a sinònim de *bloc*.

(3)Per exemple, les dades d'un estudiant.

Sabem que l'**entrada** (o la **sortida**) **dels blocs** es fa cap als (o des dels) *buffers* en la memòria interna. Actualment és habitual que, en un sistema informàtic que serveix a un SI multiusuari, s'executin molts processos simultàniament. Cada procés pot treballar amb més d'un fitxer de dades, és a dir, li pot convenir tenir uns quants blocs als *buffers*. La mida dels blocs està molt condicionada per l'espai disponible en la memòria interna per a tot aquest gran conjunt de *buffers*.

El sistema de fitxers del SO i, eventualment, l'SGBD s'encarreguen de les operacions d'E/S de blocs i de gestionar l'espai dedicat als *buffers*. Però els programes d'usuari, els que escriu el programador d'aplicacions, no entren en aquest nivell físic, es queden en un nivell lògic. Així, les operacions que fan són lectures/escriptures de registres lògics[4]. El programari s'encarrega de passar registres entre els programes d'usuari i els *buffers*. El programa d'usuari dema-na/envia un registre i el programari (SO/SGBD) li serveix/accepta des de/a els *buffers* de blocs.

(4)Llegir o escriure les dades d'un estudiant.

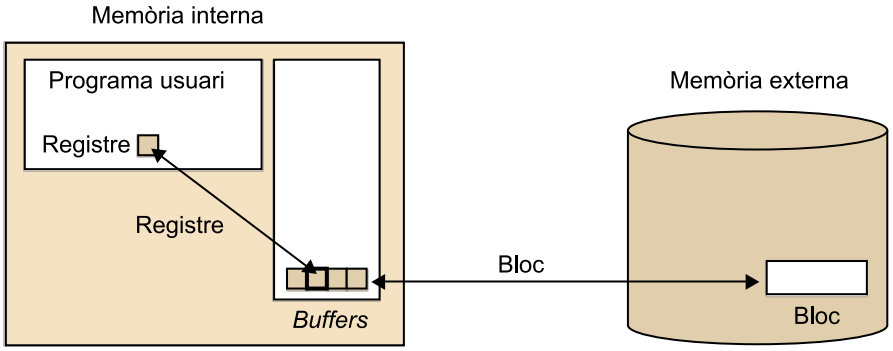
De la mateixa manera que la unitat de transferència entre la memòria persistent i els *buffers* és el **bloc**, la unitat de transferència entre aquestes i el programa d'usuari és el **registre**.

Operacions lògiques i físiques

Si en un bloc caben 100 registres, i s'està treballant seqüencialment, cada 100 lectures o escriptures efectuades pel programa s'executarà una lectura o una escriptura d'un bloc. Serà el programari l'encarregat de fer, mitjançant els *buffers*, l'adaptació entre les operacions lògiques i les físiques.

Figura 12

Esquema bàsic de l'E/S



4.3. Temps d'accés

Les memòries persistents solen tenir parts mòbils. Això fa que el seu temps d'accés sigui molt més gran que el de la memòria interna. Aquest és el seu inconvenient principal i la causa per la qual les estructures de dades per a la memòria persistent tenen particularitats diferents de les que s'utilitzen per a les memòries internes.

El temps necessari per a completar una operació física de lectura o escriptura d'un bloc a una memòria persistent mòbil consta de dues parts (temps d'accés 1 temps de transferència):

a) El **temps d'accés** és el temps necessari perquè el mecanisme es col·loqui a l'inici del bloc que s'ha de llegir o escriure.

b) El **temps de transferència** és el temps necessari per a llegir o escriure el bloc.

Per a aclarir aquests conceptes, a continuació els explicarem amb més dete-niment per al cas dels discs magnètics:

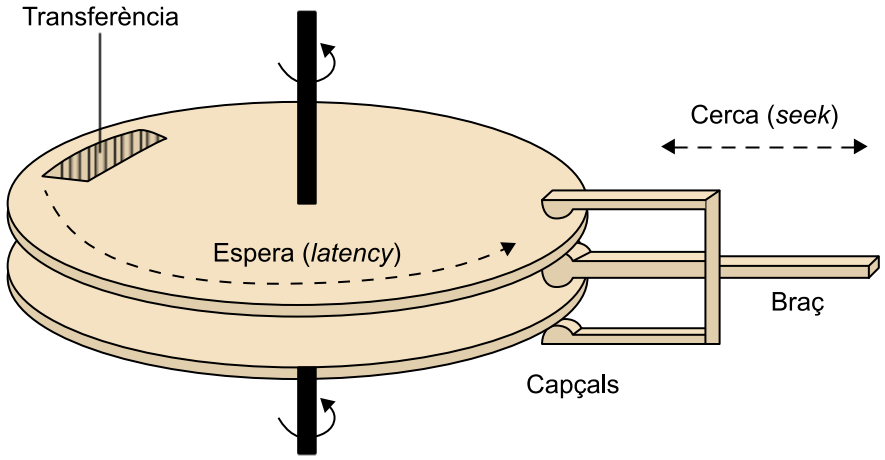
1) El **temps d'accés en el cas dels discs magnètics** consta de dues parts: temps de cerca (*seek*) + temps d'espera (*latency* o *rotational delay*).

a) En el **temps de cerca**, el braç portador dels capçals es col·loca al cilindre seleccionat.

b) Després, en el **temps d'espera**, s'espera que la rotació del disc (que no s'atura mai) faci passar per davant del capçal el sector on s'inicia l'operació. Aquest temps depèn, doncs, de la velocitat de rotació. Així, si el disc gira a 7.200 rpm, el temps d'espera màxim serà de 8,3 ms (és a dir, 7.200/60) i el mitjà, de 4,2 ms (temps d'espera mitjà = temps d'espera màxim/2).

Figura 13

Components del temps d'accés als discs



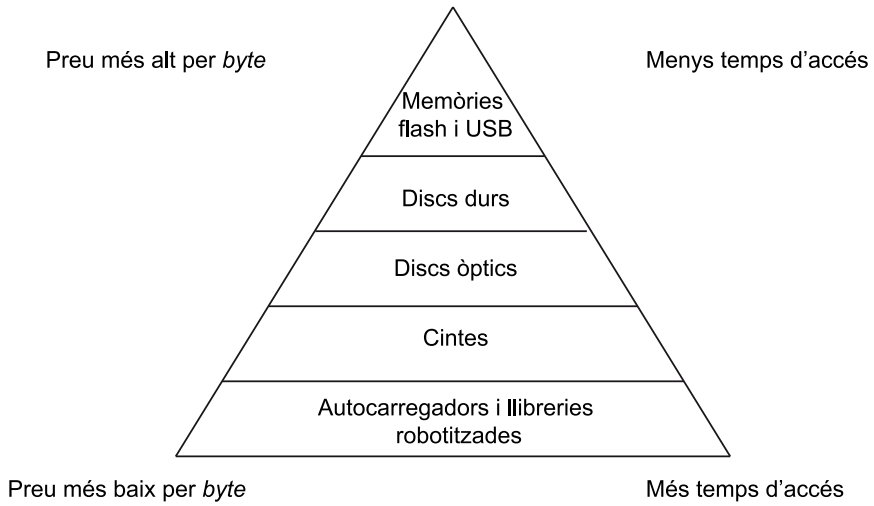
2) El **temps de transferència** serà el temps que es trigui a llegir o escriure tot el conjunt de sectors que intervenen en l'operació, és a dir, el temps que trigui a passar el bloc per davant del capçal. Fixem-nos que aquest temps depèn de la mida del bloc i de la velocitat de rotació.

Els dispositius de disc acostumen a anar equipats amb un *buffer* local per a poder fer la transferència a la memòria interna asíncronament i a gran velocitat, aprofitant l'amplada de banda del canal.

4.4. Característiques bàsiques dels suports

Acabem de recordar una característica bàsica dels suports: el temps d'accés. De la resta de característiques que ens poden interessar en farem un breu recordatori.

En la figura següent, representem un triangle amb els tipus de perifèrics més utilitzats actualment per a emmagatzemar dades. Les memòries que apareixen en nivells més alts corresponen a aquelles que permeten millor temps d'accés i en conseqüència tenen un preu més elevat. Hi ha un altre mecanisme d'emmagatzemament força utilitzat durant els últims anys que no s'acostuma a incloure en la jerarquia de memòries externes: l'emmagatzemament en xarxa. Aquest tipus d'emmagatzemament permet emmagatzemar dades remotament utilitzant una xarxa de computadors.



És interessant fer notar que per norma general com més amunt de la piràmide estigui un dispositiu, amb més freqüència canvien les dades que s'hi emmagatzemen. Així, doncs, les dades emmagatzemades en memòries Flash, USB i discos durs acostumen a ser actualitzades molt freqüentment, mentre que les dades emmagatzemades en cintes o biblioteques robotitzades poden no canviar mai.

Les principals **característiques dels suports** que cal tenir en compte són les següents: capacitat, temps d'accés, velocitat de transferència, preu per megabyte, fiabilitat (hi ha diversitat de tipus de mesures, com per exemple errors/hora, temps mitjà entre dues fallides, etc.), vida útil, utilitat, transportabilitat (si és extraïble/intercanviable) i compartició (si és d'ús exclusiu o bé el poden utilitzar simultàniament diversos processos).

Resum

En aquest primer mòdul hem fet una introducció als conceptes bàsics que fonamenten la resta de l'assignatura.

Hem explicat que els coneixements que obtenim observant els objectes del món real són abstraccions que anomenem **informació**. Una informació expressa el **valor d'un atribut** (propietat) per a una **entitat** determinada (objecte). Hem formalitzat alguns d'aquests conceptes utilitzant la teoria de conjunts.

A continuació s'ha distingit entre **entitat genèrica** (o tipus) i **entitat instància**. Les entitats instància s'hauran de diferenciar les unes de les altres mitjançant un atribut (identificador) o un conjunt d'atributs, que anomenem **clau**.

La representació informàtica d'una informació rep el nom de **dada**. Les dades de cada objecte s'agrupen en **registres** i els registres s'estructuren en **fitxers** o **BD** (conjunts de fitxers interrelacionats). Aquests fitxers o BD són emmagatzemats en **memòries externes** permanents, el temps d'accés de les quals és molt més alt que el de les memòries internes, que són volàtils.

Exercicis d'autoavaluació

1. Quins són els tres elements que determinen una informació?

2. Indiqueu què podria correspondre en el món de la nostra secretaria no informatitzada als conceptes següents:

- Entitat instància.
- Entitat tipus.
- Base de dades.
- Suport permanent.

3. Els valors de les dades per si sols són suficients per a ser interpretats i obtenir-ne informació?

Solucionari

Exercicis d'autoavaluació

1. Entitat, atribut i valor (hi podríem afegir el temps).

2.

- Entitat instància: fitxa d'un estudiant.
- Entitat tipus: tipus (format) de la fitxa dels estudiants.
- Base de dades: conjunt de fitxers, llibretes, papers, etc., que contenen la informació relativa als estudiants, les assignatures, els professors i les seves interrelacions.
- Suport permanent: cartró o paper.

3. El valor "1988", per exemple, per si sol no és suficient per a saber si es tracta de la data de naixement o de la data de matrícula o d'un import d'un pagament, etc. Si sabem que l'atribut s'anomena *DAT4*, encara no sabrem gran cosa. Hem d'esbrinar a quin atribut pertany el valor i, a més, quina semàntica té l'atribut.

Glossari

atribut *m* Propietat d'una entitat.

base de dades *f* Conjunt de fitxers interrelacionats.

camp *m* Representació del valor d'un atribut.

clau *f* Atribut o conjunt d'atributs que permet identificar els objectes (distingir-los els uns dels altres).

dada *f* Nom que rep la informació en el món de les representacions informàtiques.

entitat *f* Conceptualització d'un objecte del món real. El concepte del qual una entitat és instància s'anomena també.

fitxer *m* Conjunt de registres relatius a un mateix tipus d'entitat.

identificador *m* Un atribut és identificador si és clau (monoatribut).

memòria persistent *f* Memòria auxiliar externa amb suport permanent que s'utilitza per a mantenir emmagatzemades les dades permanentment.

organització *f* Fa referència a la manera com es col·loquen –s'estructuren– les dades per a facilitar-ne la utilització posterior.

registre *m* Conjunt de dades relatives a un objecte.

SI *m* Vegeu **sistema d'informació**.

sistema d'informació Sistema que recull, emmagatzema i distribueix informació sobre l'estat d'un domini.sigla

Bibliografia

Bibliografia bàsica

Falkenberg, E. D. (1996). "A Framework of Information System Concepts. The FRISCO Report". *IFIP WG 8.1 Task Group FRISCO*. Des de l'any 1998 també està disponible al web. És conegut com informe FRISCO. Molt interessant per a aquells que vulgueu aprofundir en marcs conceptuals del tipus dels "tres mons" que hem emprat aquí.

Silberschatz, A.; Korth, H. F.; Sudarshan, S. (2006). *Fundamentos de diseño de bases de datos* (5a. ed.). Madrid: McGraw-Hill.

Bibliografia complementària

Per a ampliar els vostres coneixements sobre les memòries persistents, els documents tècnics i comercials dels fabricants o distribuïdors poden ser una bona font d'informació. Una via d'accés a aquests documents pot ser Internet.