

Introducció a les bases de dades. Conceptes bàsics.

Components lògics d'una BD:

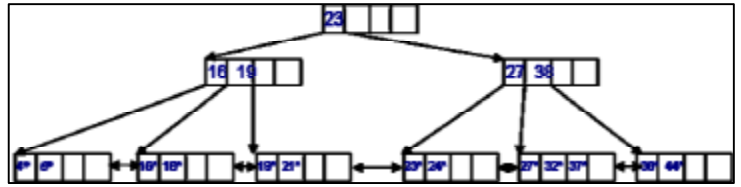
- Components lògics de dades.
 - Taules
 - Vistes
Una vista serà actualitzable sempre que faci referència a una única taula o vista actualitzable, en la que s'inclouin totes les columnes clau i totes les columnes que incorporin la restricció NOT NULL
WITH CHECK OPTION garanteix que les insercions que es facin mitjançant la vista només siguin de files que efectivament hagin de pertànyer a la vista.
- Components lògics de control.
 - Procediments emmagatzemats.
 - Disparadors
 - Privilegis
- El catàleg.

Components d'emmagatzematge d'una BD:

- El nivell lògic.
- El nivell físic
 - La pàgina
 - El extensió
 - El fitxer
- El nivell virtual
L'espai virtual es allò que independitza el nivell lògic de la BD del físic. Enllaça les dades de forma que, des del nivell lògic sembla que siguin contigües.
 - **Espai de taules:** Es correspon a una taula de la BD.
 - **Espai fragmentat :** Es correspon a la divisió d'una o diverses taules, (espai d'agrupació fragmentat), de forma que cada divisió pot guardar-se en fitxers diferents, però des del nivell lògic les dades son tractades com una unitat.
 - **Espai d'agrupació:** Diverses taules comparteixen un mateix espai, seguint un criteri d'agrupació.
 - **Espai d'objectes grans:** Tal com imatges. Es poden guardar en fitxers diferents.
 - **Espai d'índexs:** Els índexs definits sobre la BD es guarden en espais independents.
- Altres.
 - El catàleg
 - Taules temporals
 - El dietari.

Tipus d'índex :

- Índex/Arbres B+
Les dades es col·loquen en una estructura de tipus arbre, ordenat de forma que, per trobar una dada, es necessiten pocs accessos a disc. Els arbres B+ poden ser agrupats o no.



Capacitat màxima d'un arbre B+:

$(2d + 1)^{h-1} * 2d$. On d es l'ordre de l'arbre. I h es l'alçada. L'ordre, (d), correspon a la meitat de la capacitat dels nodes fulla de l'arbre. L'arbre d'exemple correspon a un arbre d'ordre $d = 2$ i alçada $h = 3$.

Calcular el nombre d'accessos per trobar qualsevol valor d'un arbre B+:

El nombre d'accessos coincidirà amb l'alçada, (h), de l'arbre. Per calcular l'alçada hem de saber l'ordre, (d):

$$2d * (\text{tamany_dada} + \text{rowId}) + 2 * \text{apuntador_node} = \text{tamany_pagina}$$

$$\text{Sabent } d \text{ tenim que: } (2d + 1)^{h-1} * 2d = \text{capacitat_arbre} \Rightarrow h = \log_{2d+1}(\text{capacitat_arbre} / 2d)$$

- Agrupat
Un índex agrupat es aquell on les dades estan físicament ordenades per el valor que proporciona l'índex.
- Dispersió, (HashTable)
Faciliten l'accés directe per valor, però no el seqüencial. Les entrades es col·loquen a les pàgines seguint una funció de dispersió. Els índexs de dispersió son mes òptims que els arbres B+ pel que fa als accessos directes per valor. Mentre que per accessos seqüencials per valor son mes òptims els arbres B+.

Factor de càrrega : $C = M / (N * L)$. On:

C: Factor de càrrega.

M: Les entrades que contindrà l'índex.

N: El nombre de pàgines primàries de l'índex.

L: El nombre d'entrades que hi caben a cada pàgina.

$$L = (\text{Mida_de_la_pàgina} - \text{mida_apuntador_pàgina_excedents}) / \text{mida_entrada_índex}$$

$$\text{mida_entrada_índex} = \text{tamany_dades} + \text{tamany_rowid}$$

Un índex de dispersió amb 50000 entrades, 30 pàgines primàries i 10 dades per pàgina, tindrà un factor de càrrega de $C = 50.000 / (30 * 10) = 166.6$.

Com mes baix sigui el factor de càrrega menys excedents hi haurà i menys E/S seran necessàries.

Transaccions:

Propietats ACID per a transaccions:

- Atomicitat.
El conjunt d'operacions de la transacció ha de ser considerat com una unitat

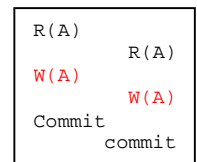
- Consistència.
Una transacció que parteix d'un estat consistent a la BD, ha d'acabar al mateix estat.
- Aïllament.
El comportament d'una transacció no ha de ser afectat per la execució concurrent d'altres transaccions.
- Definitivitat.
Els resultats d'una transacció confirmada han de ser permanents, independentment que es produeixin fallades o desastres.

Interferències entre transaccions:

- Actualització perduda.

Aquesta situació es produeix quan es perd el canvi que ha efectuat una operació d'escriptura.

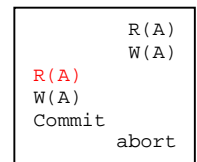
Per evitar aquesta interferència es requereix un nivell d'aïllament mínim de *Read uncommitted*.



- Lectura no confirmada.

Aquesta situació es produeix quan una transacció llegeix una dada que ha estat modificada anteriorment per una altra transacció que després avorta.

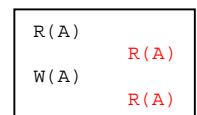
Per evitar aquesta interferència es requereix un nivell d'aïllament mínim de *Read commited*.



- Lectura no repetible.

Aquesta situació es produeix quan una transacció llegeix la mateixa dada en dos ocasions diferents i n'obté valors diferents, a causa d'una modificació feta per una altra transacció.

Per evitar aquesta interferència es requereix un nivell d'aïllament mínim de *Repeatable read*.



- Anàlisi inconsistent, (fantasmes).

Aquesta situació es produeix quan una transacció llegeix unes dades a la vegada que una altra modifica part d'aquestes dades. La primera transacció pot obtenir una visió de les dades incorrecta, que mai no s'hauria produït si les dues transaccions s'haguessin executat l'una darrera l'altra.

Per evitar aquesta interferència es requereix un nivell d'aïllament mínim de *Serializable*.

Nivells d'aïllament.

- Read uncommitted

No efectua cap reserva de lectura.

- Read committed

A cada lectura es tanca la reserva de lectura.

- Repeatable read

Les reserves de lectura es mantenen fins que la transacció no necessita tornar-los a llegir. En aquell moment, els tanca.

Molts SGBD no distingeixen entre *Repeatable Read* i *Serializable*, davant la impossibilitat de saber de forma eficient si una transacció necessitarà o no tornar a llegir una dada.

- Serializable

Es demanen reserves per totes les operacions, i es mantenen fins al final de la transacció.

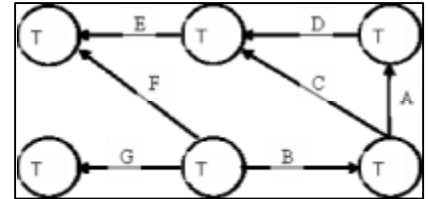
Abraçada mortal.

Es produeix quan dues transaccions requereixen dels mateixos recursos, i cada una en posseeix una part d'aquests recursos. Generalment els SGBD solucionen aquestes situacions mitjançant grafs d'espera. Cancel·len una de les transaccions i així l'altra pot continuar.

Transaccions. Grafs.

- Graf de precedències.

Es útil per trobar interferències. Si el graf presenta cicles, significa que existeixen interferències, i per tant l'horari que representa no serà seriàble. Existeixen tants horaris equivalents com recorreguts vàlids presenti el graf.



- Graf d'espera.

Es útil per localitzar abraçades mortals. Si el graf presenta cicles, significa que existeixen abraçades mortals.

BDs distribuïdes

Distribució de les dades.

- BDs centralitzades.
Totes les dades es troben a la mateixa màquina. Propietària i fortament acoblada.
- BDs distribuïdes, (client/servidor).
Les dades es troben repartides a diferents màquines connectades en xarxa. Federada i lliurement acoblada. Diferents SGBDs i SOs.
 - Regles de les BD distribuïdes
 1. Màxima autonomia local.
 2. Igualtat entre servidors.
 3. Operació continuada.
 4. Transparència de localització.
 5. Transparència de fragmentació.
 6. Transparència de reproducció.
 7. Processament distribuït de consultes.
 8. Gestió distribuïda de transaccions.
 9. Independència del maquinari.
 10. Independència del sistema operatiu.
 11. Independència dels protocols de xarxa.
 12. Independència del SGBD
 - Atomicitat de les transaccions, (protocol de confirmació en dues fases).
Es el mecanisme que segueixen els servidors implicats en una transacció per a garantir que, si una transacció es dona per confirmada en un dels servidors, també es confirmi a la resta de servidors implicats.
 - Reproducció
Consisteix en l'emmagatzemament de còpies, (reproduccions), de taules o de fragments de taules en diversos servidors, de manera que les actualitzacions que s'hi duguin a terme a la còpia d'un servidor es propaguin a totes les altres.

- **Reproducció síncrona.**
Les actualitzacions que s'efectuïn en un servidor es propaguen de forma immediata a les reproduccions de la resta de servidors.
- **Reproducció asíncrona.**
Els canvis es propaguen periòdicament, a petició d'un administrador, segons un horari, o d'altres circumstàncies.
- Caus persistents.
Consisteix a emmagatzemar a un servidor una còpia d'una part de les dades emmagatzemades en un altre servidor. La part de les dades que es guarda es la que sembla que te mes possibilitats de tornar a ser accedida. Si un client consulta dades que es troben al cau persistent, no serà necessari demanar-les al servidor que te les dades complertes.
- Fragmentació de les dades.
 - **Fragmentació vertical.**
Consisteix a repartir l'emmagatzemament de les columnes d'una taula entre diversos servidors, col·locant a cada servidor aquelles columnes a les que accediran mes freqüentment. Per poder mantenir la integritat entre els diferents fragments es necessari que a cada fragment es guardin les claus primàries.
 - **Fragmentació horitzontal.**
Consisteix a repartir l'emmagatzemament de les files d'una taula entre diversos servidors, col·locant a cada servidor aquelles files a les que accediran mes freqüentment.

Distribució del processament.

Model abcde:

- a: Entrada/Sortida bàsica.
S'encarrega de controlar l'ús dels dispositius d'E/S emprats per l'usuari, (pantalla, ratolí, teclat, ...).
- b: Gestió de la interfície amb l'usuari.
S'encarrega de presentar i captar les dades de l'aplicació.
- c: Part lògica de la aplicació.
Efectua les actualitzacions i consultes que siguin necessàries atenent a les peticions dels usuaris.
- d: Gestió de l'accés a les dades.
Ofereix serveis d'alt nivell per a consultes i actualitzacions.
- e: Accés bàsic a les dades.
S'encarrega de llegir i escriure les dades a la memòria externa.

Distribució del processament:

- a/bcde : La part client s'encarrega només del control dels dispositius de l'usuari. La resta de l'aplicació es controlada per el servidor. (caixa tonta).
- ab/cde: Al client es controlen únicament els aspectes d'interacció amb l'usuari. La resta de l'aplicació es controlada per el servidor. (Monitors transaccionals. Crides remotes, (RPC). Procediments emmagatzemats a BD).

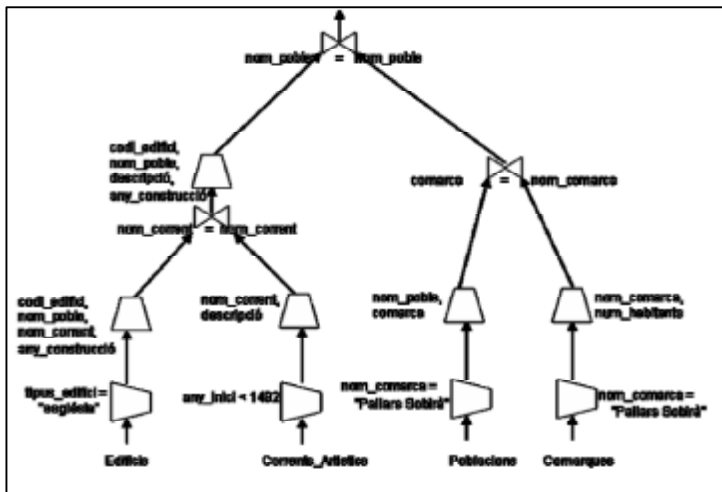
- abc/de: Al client es controlen els aspectes d'interacció amb l'usuari i la lògica de l'aplicació. Al servidor es controlen les peticions del client per a fer consultes i actualitzacions d'alt nivell. (Així es com funcionen la majoria dels SGBD comercials).
- abcd/e: El client controla tots els aspectes de l'aplicació excepte l'accés físic a les dades.

Optimització de consultes:

- Optimització sintàctica.

Es el procés que determina un ordre d'execució raonablement òptim de les operacions que inclou la consulta.

S'associa un arbre sintàctic a la consulta. Sobre aquest arbre s'aplica un algoritme de reestructuració que dona lloc a un "pla d'accés" de la consulta optimitzada. L'algoritme consta de 5 passos:



1. Separació d'operacions de restriccions per obtenir operacions de restricció d'un predicat.
2. Baixar les operacions de restricció tan avall com sigui possible.
3. Agrupar les operacions de restricció que hagin quedat juntes.
4. Baixar les operacions de projecció d'àlgebra relacional tan avall com sigui possible dins l'arbre sintàctic.
5. Agrupar les operacions de projecció d'àlgebra relacional que afectin a la mateixa relació en una única operació de projecció.

- Optimització física.

Consisteix en avaluar el cost de la execució de l'arbre sintàctic òptim associat a una consulta. Aquest cost serà la suma dels costos de totes les operacions que figuren a l'arbre.

- Optimització semàntica.

Consisteix a simplificar les consultes formulades per els usuaris mitjançant l'ús de la informació que proporcionen les regles d'integritat que s'hagin definit sobre la BD i les lleis de la lògica.

Java:

Llibreria *java.SQL* de java

- Comandes de Connection
 - commit i rollback
 - createStatement, (stmt.executeUpdate precisarà de paràmetres).
 - prepareStatement, (stmt.executeUpdate anirà sense paràmetres).
 - prepareCall, (executa un procediment emmagatzemat).
 - setAutoCommit

- Comandes de Statement i CallableStatement
 - executeUpdate
 - close
 - executeQuery, (retorna un ResultSet)
- ResultSet
 - getString, getInteger, ...
 - next

SQL:

- Tècniques de SQL programat:
 - **SQL hostatjat:** Es caracteritza principalment per permetre introduir sentències SQL dins d'una aplicació desenvolupada amb un llenguatge de programació determinat. Per fer això es necessita de:
 - Un precompilador. Que separa les sentències SQL de la resta de l'aplicació.
 - Un conjunt de "variables pont" entre l'aplicació i les sentències SQL.
 - **SQL/CLI :** Es caracteritza per permetre que una aplicació desenvolupada en un llenguatge de programació pugui incorporar sentències SQL mitjançant la crida a procediments disponibles a llibreries. Aquests procediments seran els encarregats d'establir la connexió amb el SGBD mitjançant SQL dinàmic. Un exemple: ODBC de micros\$ft
 - **JDBC:** Es una API estandard de java per treballar amb SQL
- Tractament d'errors.
 - Raise Exception -746 à Codi d'error per us de l'aplicació.
- Gestió del catàleg.
 - GRANT SELECT/UPDATE/DELETE... à Atorga permisos a usuaris.
 - CREATE ROLE à Crea un "rol" que després pot ser atorgat a usuaris.