1. **Organizace dat**
   - Jedná se o prostředek k rychlejšímu vyhledání požadovaného záznamu
   - hlediska: rychlost přístupu k datům, velikost obsazeného prostoru
   - druhy: Seznamová organizací, Invertová organizace a B-strom(velké databáze, používají indexy).

2. **Seznamová organizace dat**
   - Na záznam, který obsahuje nejnižší hodnotu nám ukazuje hlavička.
   - každý záznam má své číslo(svou adresu)
   - sousední věty jsou spojeny ukazovátkem, který se vyskytuje přímo v souboru jako další sloupec = jednocestný seznam.
   - U dvoucestného seznamu nám přibude i další "hlavička", která ukazuje na poslední záznam, takže ji spíše nazýváme patičkou.

   INSERT = záznam přidám na libovolné místo takže např. na konec souboru a včlením ho do seznamu. Někdy se nevčleňuje hned na správné místo, ale pouze na začátek či konec seznamu a k setřídění dochází až při nižším zatížení aplikace (v noci).
   UPDATE = změní se hodnota, seznam se příslušně přeorganizuje podle nové hodnoty.
   DELETE = záznam se označí a vyřadí ze seznamu, smaže se až při nižším zatížení aplikace (v noci).
   Rychlost versus datový prostor = nižší vyhledávací rychlost, méně náročný na paměť.

3. **invertovaný seznam(Invertová organizace dat)**
   - Pro každý klíč tohoto souboru se vytváří tzv. invertovaný soubor (= indexová tabulka)
   - Tento typ invertovaného souboru má jednu nevýhodu a to nestejnou délku věty = to není vhodné pro rychlý výběr. Toto lze vyřešit přesunem seznamu výskytů do oficiální oblasti zvané tabulka výskytů.
   - Invertovaný soubor se nemusí pořizovat pouze pro jeden sloupec, může být vytvořeno několik invertovaných souborů pro několik sloupců
   Úplný invertovaný soubor = pro všechny sloupce tabulky jsou vytvořeny invertované soubory
   SELECT
      Sekvenční = projdu pomocí invertovaného souboru vytvořeného pro primární klíč.
      Náhodný = nemusí se ani použít invertovaný soubor.
   INSERT = nový záznam uložíme na libovolné místo a upravíme invertovaný soubor (aktualizace invertovaného souboru a zatřídění do tabulky výskytů).
   UPDATE = změna hodnoty a tím pádem i změna příslušného invertovaného souboru.
   DELETE = označení záznamu a vyřazení jen z invertovaného souboru. Tím pádem je záznam nedostupný. Z vlastní tabulky se smaže např. v noci. Vhodný pro tabulky s několikanásobným výskytem stejných hodnot.
   Rychlost versus datový prostor = poměrně vysoká vyhledávací rychlost, náročnější na paměť.

4. **Spojení relací, k čemu je?**

   Spojeni relací, značení R*S:
   Použití: spojení tabulek s různym počtem sloupců
   3 druhy:
      1) theta spojení – spojí to co vyhovuje podmínce
      2) přirozené spojení – spojí se ty řádky, které mají stejné hodnoty požadovaných atributů
      3) kompozice – typ přirozeného spojení kde „vyhodíme“ všechny atributy přes které je spojení provedeno

5. **Určete projekci R[A1, A3] (bylo v písemce)**

   Vyberou se dané sloupce a odstraníme duplicitu (duplicitu záznamů):
   (záznam – přes všechny sloupce, nesmí být duplicitní záznamy Aa, Bb, Db)

6. **dekompozice R(A, B, C, D, E), B->C, E (bylo v písemce)**
   R1 (A,B,D)
   R2 (B,C,E)

   Dekompozice: X-> YZ => X->Y a X->Z

7. **Co je ROLL-BACK, jak a na co se používá?**
   Roll-Back, undo – zpětné použití žurnálu, vychází se ze současného stavu DB, využívá se žurnálu – zpětným odvíjením se dostaneme až do konzistentního stavu. Má význam pouze pro nedokončené transakce.

8. **Jaká pravidla je třeba dodržovat při paralelním zpracování transakcí, když zamykají a odemykají objekty? (bylo v písemce)**
   – objekt smí být zamknut pouze jednou transakcí a pokud jiné transakce chce číst hodnotu x, tak je pozastavena
   – musí být proveden transakcí, kterou byl zamknut
   - pokud transakce objekt odemkne nesmí už **žádný** zamknout (to tam většině chybělo)

9. **Paralelní zpracování transakcí**
   SŘBD obsahuje 3 moduly pro paralelní zpracování:
   modul řízení transakcí (RT) – na který se transakce obracejí s žádostí o provedení operace
   plánovač – zabezpečuje synchronizaci požadavků více transakcí (z modulu RT), požadavky zpracovává do tzv. plánů
   modul řízení dat (RD) – v databázi vykonává čtení, zápis podle požadavků plánovače

   U paralelního běhu se hlídá uspořádanost – aby výsledek dopadl stejně jako při sériovém uspořádání. Chyba nastane, když transakce nastanou navzájem – transakce se musí řídit.

10. **Dvoufázový protokol**
    1) objekt může být uzamčen v každém okamžiku jen pro jednu transakci
    2) jakmile transakce objekt odemkne, nesmí ho znovu zamknout

11. **Vysvětlete pojem entita.**
    - objekt reálného světa, jednoznačně identifikovatelný a schopný nezávislé existence
    ERA model, E-R-A model, ER model a E-R model. - E = Entita (množina dat)

12. **přidání prvku do seznamové organizace; načrtnout obrázek**
    INSERT = záznam přidám na libovolné místo takže např. na konec souboru a včlením ho do seznammu. Někdy se nevčleňuje hned na správné místo, ale pouze na začátek či konec seznamu a k setřídění dochází až při nižším zatížení aplikace (v noci).

13. **selekce**
Výběr řádků, které splňují podmínku

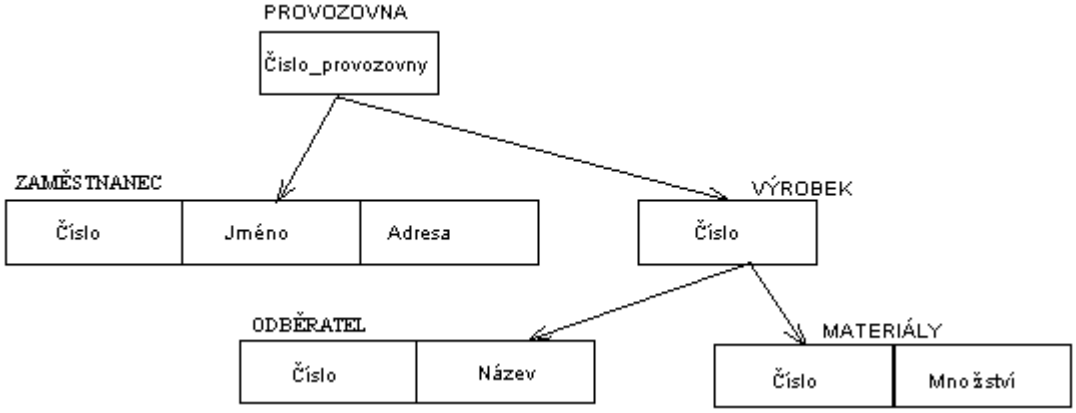
14. **princip síťového modelu (co je to?)**
- jiná varianta pohledu na data
- typ záznamu a spojka
Typ záznamu – popíšeme položky kterých se záznam skládá
Spojka – ukazatel na záznam logicky související
U typu záznamu určujeme klíč
Spojka definuje spojení mezi dvěma typy záznamu

Př: Student ----- dostane ----- známka
Typ záznamu spojka typ záznamu

Spojka – s informací
Bez informace

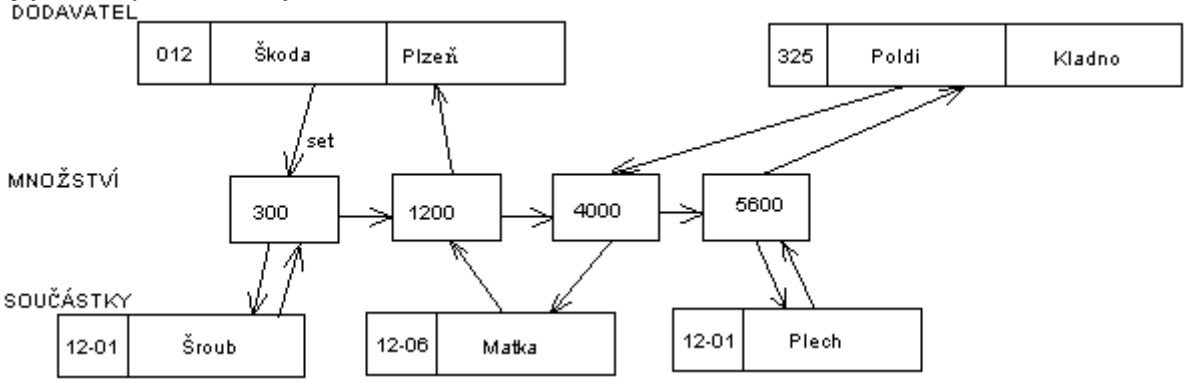
15. **integritní omezení**
Vymezují hodnoty objektů databáze tak, aby mohly mít v reálném světě smysl
Entitní integrita
požadavek na jednoznačnou identifikaci řádky v tabulce
definicí primárního klíče (jedinečný, vyplněn)
Doménová integrita
omezuje hodnoty položek (intervalem, výčtem)
v normě SQL je příkaz CREATE DOMAIN
Referenční integrita
popisuje, jaké vztahy platí mezi tabulkam
obvykle se zavádí cizí klíč

16. **hierarchický model dat**
Data jsou organizována do stromové struktury. Každý záznam představuje uzel ve stromové struktuře, vzájemný vztah mezi záznamy je typu rodič/potomek



Použití hierarchického modelu je vhodné tam, kde i zájmová realita má hierarchickou strukturu. Nalezení dat v hierarchické databázi vyžaduje navigaci přes záznamy směrem dolů (potomek), nahoru (rodič) a do strany (další potomek). Mezi nevýhody hierachického modelu patří:
- v některých případech nepřirozená organizace dat (zejména obtížné znázornění vztahu M:N, který se řeší např. pomocí virtuálních záznamů),
- složité operace vkládání a rušení záznamů.

17. **Síťový model dat**
Síťový model dat je v podstatě zobecněním hierarchického modelu dat, který doplňuje o mnohonásobné vztahy. Tyto vztahy jsou označovány jako C-množiny neboli Sets (dále budeme používat pojem set, pro který neexistuje ekvivalentní český výraz). Tyto sety propojují záznamy různého či stejného typu, přičemž spojení může být realizováno na jeden nebo více záznamů.



Přístup k propojeným záznamům je přímý bez dalšího vyhledávání, k dispozici jsou tyto operace:
nalezení záznamu podle klíče,
posun na prvního potomka v dílčím setu,
posun stranou na dalšího potomka v setu,
posun nahoru z potomka na jeho rodiče v jiném setu.
Nevýhodou síťové databáze je zejména nepružnost a obtížná změna její struktury.

18. **Co je výskyt setu u síťového modelu?**
V databázi je vztah reprezentován řadou **výskytů setu**. Výskyt setu obsahuje právě jeden výskyt záznamu vlastníka a právě ty výskyty záznamů člena setu, které jsou s vlastníkem výskytu setu v příslušném vztahu. Výskyt setu může obsahovat pouze výskyt záznamu vlastníka (prázdný výskyt setu) a množiny členů dvou výskytů téhož typu setu jsou disjunktní.
Výskyty setů se realizují zřetězeným seznamem: z vlastníka na prvního člena, členové setu mezi sebou, z posledního člena zpět na vlastníka. Pak procházení celé množiny je velmi rychlé, bez zbytečných přenosů záznamů mezi diskem a pamětí. Pokud jsou členy setu navíc umisťovány „blízko sebe“, může být počet přenosů ještě menší, pokud je více členů množiny umístěno v jednom fyzickém bloku.
Pro zařazení záznamů do setů platí:
• výskyt setu obsahuje právě jeden výskyt záznamu vlastníka a právě ty výskyty záznamů členů setu, které jsou s vlastníkem výskytu setu v příslušném vztahu,
• výskyt setu může obsahovat pouze výskyt záznamu vlastníka (prázdný výskyt setu),
• množiny členů dvou výskytů téhož typu setu jsou disjunktní

19. **Přímý zápis do báze dat**
- tvorba žurnálu s bezprostředním zapsáním změn
- do žurnálu zapisuje staré hodnoty
- po obnově se provádí roll-back

20. **funkční nezávislost atributů**
Máme relaci R a v ní dvě množiny atributů A a B, pak existuje funkční závislost B na A, jestliže jedné A-hodnotě se přiřadí nejvýše jedna B-hodnota.
Nezávislost – opak :-)

21. **Práva**
příkaz GRANT - Přiřazení práv
ALL RIGHTS – všechna práva
<privilegia> – jen konkrétní privilegia
ALL NOT <privilegia> – vše kromě vyjmenovaných
ON <tabulka>
TO <uživatel>
[WITH GRANT OPTION] – práva může předávat dál

příkaz REVOKE - Odebrání práv

Privilegia: READ, INSERT, UPDATE, DELETE, DROP

Pokud odebereme práva uživateli s GRANT OPTION, odeberou se i všem, komu je předal dál – řešeno pomocí časových značek v tabulce práv.

22. **relační schéma R*S pro R(A1, A2, A3, č_osoby) a S(B1, B2, č_osoby) (bylo v písemce)**
(R[č_Osoby]*S)[A1,A2,A3,B1,B2, č_osoby]

23. **definice journálu (bylo v písemce)**
zaznamenávají se tam : identifikátor transakce
identifikátor objektu s kterým se pracuje
identifikátor uživatele, který transakci spustil
nová hodnota(někdy i stará)
čas spuštění transakce

24. **Dekompozice – příklad, relace s 5 atributy R(A,B,C,D,E) B,C klic A.E zavisi na B navrhnete vhodnou dekompozici na 2 relace, která bude bezeztratová (bylo v písemce)**
RES : R1 (B,C,D) klice B,C
R2 : (B,A,E) klice B

25. **byly dány 3 transakce s položkou V1 (V1 počet součástek)**
a. operace V1:= V1+50
b. operace V1:= V1-20
c. tisk vsech záznamů s žímto atributem V1
jak postupovat pri techto transakcich (TZN. Pri zapisu, cteni, zmene, jeslti zamykat nebo ne)

a) lock(V1), V1 <- V1+50, unlock(x)
b) zamykat
c) netřeba zamykat read(V1)

26. **Vyjmenovat položky v tabulce u příkazu GRANT (tzn. Vyjmenovat položky tabulky SYSAUTH**
READ, INSERT, UPDATE, DELETE, DROP

27. **napsat v SQL …. (bylo v písemce-my měli create table – dvě tabulky a propojit)**

```
SELECT Jmeno FROM  Student a, Známka b
WHERE Studijní_obor = 'počítačové sítě'
AND a.č_stud = b.č_stud AND
   Předměd = 'DB1' AND Známka=1;

UPDATE čtenář
SET ADRESA = nová adresa
WHERE č_čtenáře = '123';

INSERT INTO ČTENÁŘ (č_čt, jméno)
VALUES ('123','Novák Jan')

CREATE TABLE Statistika
                (č_čt char (4),
                Počet smallint,
   PRIMARY KEY (č_čt),
   FOREIGN KEY (počet),
   REGERENCES cizi_tabulka (jméno primárního klíče v cízí tabulce);
```

28. **SQL**
DDL
– CREATE TABLE, ALTER TABLE, DROP TABLE
- CREATE VIEW, ALTER…
- CREATE INDE, DROP INDEX
- SET TRANSACTION
- GRANT, REBOKE

DML
- SELECT, INSERT, UPDATE, DELETE

29. **Co je to usporadatelnost a jak ji lze dosáhnout**
U paralelního běhu se hlídá uspořádanost – aby výsledek dopadl stejně jako při sériovém uspořádání. Chyba nastane, když transakce nastanou navzájem – transakce se musí řídit.

SŘBD zabezpečuje – definovani databaze, vkladani, vyber a ochranu dat, komunikace mezi systemem a uzivatelem