Otázka 16 – A7B36DBS

Zadání

Principy databázových systémů. Systém řízení bází dat, databáze, datový slovník, víceuživatelský přístup, kategorie DB uživatelů, architektura DB stroje. (A7B36DBS)

Slovníček pojmů

- **administrátor** správce DB systému
- **banka dat** organizační forma systému zpracování dat zahrnující bázi dat a systém řízení báze dat
- **báze dat** množina souborů a jejich popisu, které jsou navzájem v určitém logickém vztahu a jsou spravované SŘBD
- **databáze (data)** je logicky uspořádaná (integrovaná) kolekce navzájem souvisejících dat**,** je sebevysvětlující, protože data jsou uchovávána společně s popisy, známými jako metadata (také schéma databáze)
- **systém řízení báze dat** (SŘBD, angl. database management system – DBMS) je obecný softwarový systém, který řídí sdílený přístup k databázi, a poskytuje mechanizmy pomáhající zajistit bezpečnost a integritu uložených dat
- **databázové tabulky** tabulka – sloupec (název sloupce) – řádek – hodnota
- **primární klíč** identifikátor řádku, jednoznačně identifikuje daný řádek, má zároveň minimální délku, může to být 1 atribut nebo může být složený z více atributů
- **cizí klíč** primární klíč použitý v další tabulce pro určení vazeb mezi tabulkami, cizí klíč jako zprostředkování vazby do jiné tabulky se bude vždy skládat ze stejných sloupců jako primární klíč
- **integrita dat** pod integritou nebo konzistencí rozumíme fakt, že data věrně zobrazují reálný stav, který popisují, nejsou ve sporu, nic nechybí. Základním předpokladem udržení dat v konzistentním stavu je kvalitně navržená datová základna.
- **indexy** pomocné dodatečné informace, které slouží zejména pro zrychlení zpracování našich požadavků
- **SQL** (structured query language) příkazy jazyka SQL jsou členěny do několika skupin: DDL, DML, správa tabulek, indexů a jiných databázových objektů, nastavování parametrů DBMS, správa přístupových práv uživatelů. Mezi jednotlivými DBMS jsou mírné odlišnosti. Jazyk SQL byl navržen jako tzv. neprocedurální jazyk, v příkazech popisuje, „co“ chce získat, a ne „jak“ (postup, proceduru) to chceme získat. Liší se tak od většiny rogramovacích jazyků, které jsou procedurální a ve kterých vždy popisujeme přesný postup toho, co se má provést.
  Nejpoužívanější příkazy:

SELECT
INSERT
UPDATE
DELETE
CREATE TABLE
DROP TABLE

- **transakce** jedním ze základních problémů pro udržení konzistence dat v databázi je situace, když během zpracování příkazu dojde k nestandardnímu ukončení práce s databází. Jeho příčinou může být například fyzická chyba zařízení počítače nebo výpadek elektrického proudu. Nastane-li tato chyba během příkazu, který aktualizuje hodnoty v databázi, stávají se data nekonzistentní – zvýšení platu se provedlo pouze u části zaměstnanců, a ne u všech, jak bylo zamýšleno; byl vymazán zaměstnanec z tabulky zaměstnanců, ale již se nestihlo vymazat všechny jeho podřízené záznamy (výplaty, řešené úkoly …), klasika převody peněz v bance apod. Z těchto důvodů byl vytvořen koncept transakčního zpracování. Transakce je nedělitelný logický celek, který bude buď proveden celý, nebo se neprovede žádná jeho část. Transakce se skládá z SQL příkazů, které jsou postupně vykonávány. Od začátku transakce až do jejího ukončení příkazem COMMIT jsou všechny změny provedené v datech uschovány, aby bylo možné v libovolném okamžiku provést návrat do stavu, který byl na začátku transakce. Dojde-li před ukončením transakce k chybě (např. vlivem výpadku proudu), jsou při opětovném spuštění databáze všechna data uvedena do stavu, v jakém se nacházela na začátku transakce. Nemůže se tedy stát, aby se databáze dostala do nekonzistentního stavu tím, že by se provedl pouze příkaz vymazání zaměstnance z tabulky zaměstnanců a již se nestihlo provést vymazání všech ostatních, na něj se odkazujících záznamů. Procesu, který provede návrat ke stavu na počátku transakce, se říká „odrolování“ (z anglického rollback). Odrolování je možné vyvolat kdykoliv před ukončením transakce pomocí příkazu ROLLBACK. Zadáním příkazu COMMIT je aktuální transakce ukončena a všechny dosud provedené změny jsou natrvalo uloženy do databáze. V tomto okamžiku začíná nová transakce, která bude ukončena dalším zadáním příkazu COMMIT. Po zadání příkazu COMMIT již není možný návrat do stavu, ve kterém byla data na začátku transakce.
- **žurnál** speciální soubor, do kterého se ukládají změny (v podstatě transakce) provedené v tabulkách, obecně obsahuje identifikátor transakce, identifikátor objektu, kdo transakci provedl, čas provedení, nová hodnota objektu, stará hodnota objektu (v některých případech)
- **rozšíření jazyka SQL** procedury Jazyk SQL se neustále vyvíjí a tvůrci databázových systémů se snaží neustále reagovat na požadavky uživatelů a tvůrců databázových aplikací. Mezi tato rozšíření patří možnost definice procedur jako bloku SQL příkazu nebo celé procedury při vzniku určité události. Z hlediska jazyka SQL se jedná o porušení jeho základního principu. Jazyk SQL je navržen jako neprocedurální jazyk, kterým popisujeme „co chceme“ a ne, „jak to chceme udělat“. Existují situace, kdy je vhodné použít neprocedurální jazyk, a naopak situace, pro které se více hodí jazyk procedurální. Proto byly procedury přidány do databázových systémů a můžeme je zejména při vytváření databázových aplikací používat současně s příkazy jazyka SQL
- **trigger** je SQL příkaz nebo procedura, která je spouštěna při vzniku určité události (proto název trigger – anglicky „spoušť“). Touto událostí může být přidání řádku do tabulky, změna hodnot ve sloupci nebo vymazání řádku z tabulky. Pomocí triggerů můžeme řešit zajištění integrity dat v případě, že databázový systém neumožňuje její zajištění automaticky nebo potřebuje provést dodatečné akce. Definice triggerů je zatím pouze v připravovaném návrhu standardu SQL3. Proto se můžeme setkat s velkými odlišnostmi jejich syntaxe mezi různými databázovými systémy
- **přístupová práva** pracuje-li s databází více uživatelů, není žádoucí, aby všichni mohli provádět v databázi změny, případně, aby měli přístup ke všem informacím uloženým v

databázi. Přístupová práva se týkají vždy jednoho databázového objektu (tabulky,pohledy apod.) a konkrétního uživatele. Rozlišujeme dvě skupiny přístupových práv:
    - právo na čtení
    - právo na zápis (aktualizaci)

  automaticky jsou všechna práva přidělena uživateli, který objekt (např. tabulku) vytvořil. Ten má zároveň přidělovat tato práva dalším uživatelům, včetně práva na další přidělování vlastněných práv. Práva se přidělují pomocí příkazu GRANT a odejmout je můžeme pomocí příkazu REVOKE
- **slovník dat** (z anglického „data dictionary“) popisuje strukturu všech objektů uložených v databázi.

Principy databázových systémů

základní přístupy ke zpracování dat

- **Souborově orientovaný přístup**

Historicky nejstarší způsob. Program, který zpracovává data má svá vlastní data. Dnes se tento způsob používá v aplikacích, které nejsou databázové (např. výpočty). Při souborově orientovaném přístupu se používají různé organizace dat. Každý takový program musí mít v sobě přístupový mechanismus k datům, to je velká nevýhoda, která obnáší hodně práce. Další problémy:

- **Izolace dat** soubor obsahuje základní strukturu soubor – záznam – položka, ale na úrovni dat nelze podchytit vazbu mezi soubory.
- **Duplicita dat** více aplikací pracuje se stejnými daty, ale protože aplikace vyžadují každá svoje vlastní data, musí se data duplikovat.
- **Závislost:** data – programy

- **Databázový přístup**

Odstraňuje nevýhody souborově orientovaného přístupu. Definice dat je provedena mimo aplikační programy. Data mohou být uložena nezávisle na aplikačním programu. V aplikačních programech není zabudován mechanismus přístupu k datům

Vlastnosti databázových systémů:

- sdílení dat (víceuživatelský přístup)
- unifikované rozhraní a jazyk(y) definice dat a manipulace s daty
- znovu využitelnost dat
- bezespornost dat
- snížení objemu dat (odstranění redundance)

Systém řízení báze dat

- Definování a redefinování dat v databázi (data definition)

organizace datových souborů (vytváření a změny datových struktur)

- Vytváření obsahu databáze

aktualizace datových souborů (vkládání dat, změny, aktualizace dat)

- Výběr a výstup (data display)

prezentování, zobrazování, prohlížení) dat z databáze

- Tvorba formulářů

obrazovek, pohledů a výstupních sestav

- Kontrola integrity dat (data integrity)

poskytuje metodu nebo metody pro definování a zajištění správnosti dat

- Kontrola přístupových práv

určuje, kdo a jak může přistupovat k datům
- Programovací jazyk pro vytváření vlastních aplikací

Aby mohl být nějaký programový systém označený za SŘBD, musí být jednak schopen efektivně pracovat s velkým množstvím dat, ale také musí být schopný řídit (vkládat, modifikovat, mazat) a definovat strukturu těchto perzistentních dat (čímž se liší od prostého souborového systému).

V současnosti používané databázové systémy mají i mnoho dalších charakteristických vlastností:
- podporu pro definici datových modelů (například relační, logický, objektový)
- správa klíčů: vlastní (interně implementované) indexování, dodržování unikátnosti hodnot ve sloupcích, nad kterými je definován unikátní nebo primární klíč; implementace fulltextového vyhledávání pro fulltextové klíče; implementace cizích klíčů
- využití některého jazyka vyšší úrovně pro manipulaci a definici dat (např. SQL, QBE, datalog, Common English Query) a vyřešení komunikačního kanálu mezi uživatelem či skriptem a SŘBD v tomto jazyku,
- autentizaci uživatelů a jejich autorizaci k operacím nad daty (u každého uživatele může být definováno, jaký typ příkazů je oprávněn spouštět)
- správu transakcí, atomicitu jednotlivých příkazů
- robustnost a zotavitelnost po chybách bez ztráty dat
- uložené procedury
- triggery
- integritu dat; například nepovolením vložení duplicitního řádku s unikátním klíčem nebo řádku s hodnotami NULL u sloupců, které NULL být nesmějí
- kanály pro hlášení zpráv po úspěšně vykonaných dotazech, chybových hlášek, varování
- pokročilé funkce jako např. Common Table Expressions, zpožděné zápisy, a jiné
- profilování, statistické informace o běhu dotazů, procesů, přístupu uživatelů atd.
- nezávislost dat změna struktury v datech nevyvolá změny v aplikačních programech, uživatelské programy vůbec neví, kde jsou data uložena

Následující seznam obsahuje příklady některých systémů řízení báze dat.
- Oracle
- DB2
- Sybase Adaptive Server Enterprise
- FileMaker
- Firebird
- Ingres
- Informix
- Microsoft Access
- Microsoft SQL Server
- Microsoft Visual FoxPro
- MySQL
- PostgreSQL
- Progress
- SQLite
- Teradata
- CSQL
- OpenLink Virtuoso

Databáze

Databáze je logicky uspořádaná (integrovaná) kolekce navzájem souvisejících dat. Je sebevysvětlující, protože data jsou uchovávána společně s popisy, známými jako metadata (také schéma databáze). Data jsou ukládána tak, aby na nich bylo možné provádět strojové dotazy – získat pro nějaké parametry vyhovující podmnožinu záznamů.
Někdy se slovem „databáze“ myslí obecně celý databázový systém.

Hlavním smyslem databáze je schraňovat datové záznamy a informace za účelem:

- sdílení dat více uživateli,
- zajištění unifikovaného rozhraní a jazyku definice dat a manipulace s daty,
- znovuvyužitelnosti dat
- bezespornosti dat a
- snížení objemu dat (odstranění redundance).

- Požadavky kladené na databázi
  - Neredundantnost - eliminace zbytečných duplicit
  - Vícenásobná využitelnost - ke stejným datům může přistupovat více uživatelů (podle oprávnění)
  - Integrita dat - databáze obsahuje prostředek, který zamezí tomu, aby uložená data byla ve sporu (nekonzistentní)
  - Nezávislost dat - změna struktury v datech nevyvolá změny v aplikačních programech, uživatelské programy vůbec neví, kde jsou data uložena
  - Datový model databáze by měla umožnit implementovat libovolný datový model

Datový slovník

Jsou zde uloženy informace o existujících tabulkách a jejich sloupcích, popis omezení pro data v tabulkách, popis definovaných indexů a další charakteristiky dat používané pro zrychlení zpracování příkazů. Jednou z podmínek relačních databázových systému je, že i tyto informace jsou uloženy v relačních tabulkách a jsou dosažitelné pomocí jazyka SQL. V každém databázovém systému se proto můžete setkat s různým počtem tzv. systémových tabulek. Z našeho hlediska se jedná o normální tabulky a můžeme zjistit jejich obsah pomocí příkazu SELECT. Systémové tabulky sice můžeme číst, ale bývá zakázáno v nich údaje měnit. Názvy systémových tabulek a jejich popis jsou součástí dokumentace ke každému databázovému prostředí

Víceuživatelský přístup

Představme si, že dva nebo více uživatelů najednou mění stejný prvek ve stejné databázi. Konflikt nastává v momentě, kdy se jeden uživatel pokusí uložit změny po té, co jiný uživatel své změny uložil dříve. Riziko konfliktních situací narůstá s časem potřebným na provedení transakce.

Pojem transakce v databázovém kontextu znamená posloupnost operací nad prvky databáze, která realizuje jednu ucelenou operaci z pohledu uživatele. V geografických informačních systémech (GIS) se setkáváme hlavně s termínem dlouhá transakce. Mnoho GIS editací si vyžádá více než pár minut a některé úpravy zaberou hodiny, dny či dokonce měsíce než jsou dokončeny.

Existují dvě základní možnosti jak přistoupit k víceuživatelské editaci. Podstatou prvního z nich, pesimistického přístupu, je vyloučení možnosti, že konflikt nastane. Objekty databáze editované

jedním uživatelem jsou pro ostatní uživatele nepřístupné, databázový systém provede zamčení celé databáze, jedné tabulky, jednoho řádku nebo jedné buňky. Tento způsob zamykání je dobře realizovatelný u klasických (lexikálních) databází, v případě prostorových je situace obtížnější. Ochránit před dvojí editací je potřeba celé prvky, které je zapotřebí zamknout. Ty mohou ale zasahovat i do jiných výřezů než je ten aktuálně editovaný. Kvůli jednomu objektu se zbytečně znepřístupní příliš velká oblast databáze. Z tohoto důvodu se pesimistický přístup v prostorových databázích příliš neuplatňuje.

Optimistický přístup vychází z předpokladu, že ke konfliktu nedojde, případně jen v omezeném množství. Nedochází k zamykání databáze a jejích částí, uživatelé editují stejnou databázi ve stejný okamžik. Po dokončení transakce a uložení změn následuje kontrola konfliktů. V případě konfliktu je na editorovi (uživatel, který uložil změny) nebo na správci databáze rozhodnout, která ze tří verzí se do databáze uloží (viz tabulka 1 – „Řešení konfliktu - tři možnosti uložení “).

Tabulka 1. Řešení konfliktu - tři možnosti uložení

Verze	Popis	Z pohledu editora
edited version	verze uživatele, který způsobil konflikt	moje verze
conflict version	verze uživatele, který uložil dříve	cizí verze
pre-edited version	verze existující před začátkem editace uživatele, který způsobil konflikt	původní verze

V případě dlouhé transakce jsou všechny změny provedené uživatelem uchovávány v takzvaných Add a Delete tabulkách, souhrnně označovaných jako Delta tabulky (delta tables). Prvky nově přidané znamenají záznam do tabulky Add (dodatky), naopak prvky smazané se zaznamenají do tabulky Delete (výmazy). Úprava existujícího prvku (změna geometrie, polohy...) vyžaduje zápis do obou tabulek najednou (starý tvar je smazán a nový přidán). Ukázka delta tabulek je na obrázku 2 – „Dlouhá transakce - delta tabulky“.

A víceuživatelský prostředí je jednou v které mohou ostatní uživatelé připojit a proveďte změny ke stejné databázi, se kterou pracujete.V důsledku toho několik uživatelů může pracovat s objekty stejné databáze ve stejné době.Proto víceuživatelský prostředí zavádí možnost databáze, které jsou ovlivněny změny provedené jinými uživateli v průběhu provádění změn a naopak.

klíč problém se při práci s databázemi v víceuživatelský prostředí je přístupová oprávnění.Oprávnění pro databázi určují rozsah prací lze provést pomocí databáze.Například měnit objekty v databázi, musíte mít oprávnění pro databázi odpovídající zápis.Další informace o oprávněních v databázi naleznete v dokumentaci k databázi.Další informace viz Oprávnění a databáze Visual nástroje (databázové nástroje Visual).

Při ukládání změn provedených v tabulkách, Návrhář ověří, že databáze nebyla změněna od posledního uložení změn.Pokud jiný uživatel provedl změny, budete upozorněni, byl modifikován databáze.Můžete potřebovat odsouhlasit tyto změny.Další informace naleznete v tématu Sloučení změn provedených ve více uživatelů (databázové nástroje Visual).

V prostředí je důležité mít na paměti, aby se zabránilo konfliktních změn.Další informace naleznete v tématu Otázky vývoj databáze (databázové nástroje Visual).

Jedním ze způsobů, jak se vyhnout problémům je v kopii databáze, jako je například databáze zkouška, při provádění změn, pak můžete vytvořit změnový skript, který je možné spustit, proveďte tyto změny v původní databázi po vyřešení konfliktů v režimu offline.Další informace viz Vývoj, test a výrobní databáze (databázové nástroje Visual).

Kategorie DB uživatelů

Typičtí uživatelé
Databázový administrátor

- Instalace DBMS
  - často vyžaduje hlubší zásahy do konfigurace OS – nastavení semaforů, velikost sdílené paměti, parametry FS,….
- údržba verzí, „patchování“
- konfigurace DBMS, konfigurace klient-server
- vytvoření a údržba databáze
- zálohování a obnova databáze při pádu
- monitoring a plánování růstu databáze
- audit
- Když databáze nefunguje, je to jeho starost
- Má na starost replikaci dat na sekundární stroje a load-balancing

Administrátor - správce dat

- Obvykle ne jednotlivec, ale odborný útvar (osoby zodpovědné za autorizovaný přístup do databáze). Může obsahovat i projektanty IS, kteří v průběhu života IS modifikují aplikaci v souladu s měnícími se skutečnostmi.
- Vytváří strukturu databáze/tabulek
- Ručí za integritu dat
- Když jsou chyby v datech, je to jejich starost

Příležitostný uživatel

- Vyžadují data z databáze v různých, předem nepředvídatelných souvislostech
- Obvykle ovládají silnější dotazovací jazyk (např. SQL)
- Mění své požadavky v závislosti na svých okamžitých potřebách, tj. aplikují interaktivní či ad hoc dotazy

Aplikační programátor

- Vytváří aplikace pro použití naivními uživateli
- Zná strukturu databáze a píše nad ní dotazy
- Využívá pouze DML, nevytváří strukturu databáze/tabulek

Naivní uživatel

- Využívá aplikační rozhraní vytvořené programátory
- Nemá žádné znalosti o databázích (pro svou práci je nepotřebuje, všechno si 'nakliká' v aplikaci)

Architektura DB stroje

Architektury databázových systému se obecně dělí na

- centralizované (kde se databáze předpokládá fyzicky na jednom počítači)
- distribuované

případně na

- jednouživatelské
- víceuživatelské.

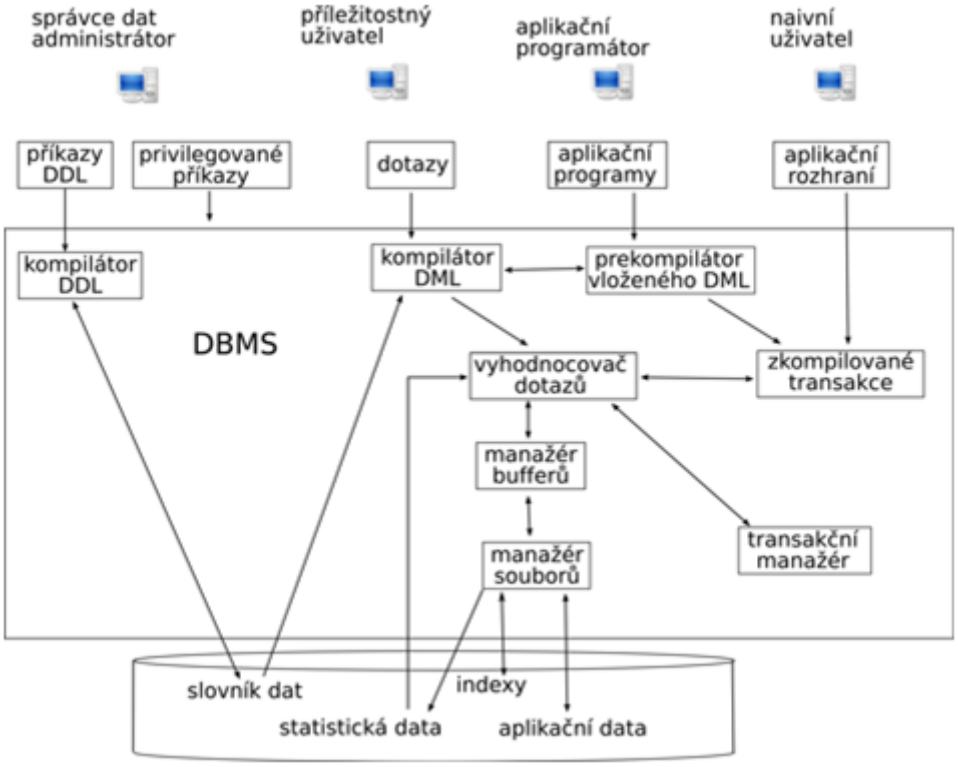
Distribuované databázové systémy
Distribuovaný systém řízení báze dat je vlastně speciálním případem obecného distribuovaného výpočetního systému. Jeho implementace zahrnuje fyzické rozložení dat (včetně možných replikací databáze) na více počítačů – uzlu, přičemž jejich popis je integrován v globálním databázovém schématu. Data v uzlech mohou být zpracovávána lokálními SRBD, komunikace je organizována v sítovém provozu pomocí speciálního softwaru, který umí zacházet s distribuovanými daty. Fyzicky se řeší rozložení do uzlu, svázaných komunikačními kanály, a jeho transparence (neviditelnost – navenek se má tvářit jako jednolitý systém). Každý uzel v síti je sám o sobe databázový systém a z každého uzlu lze zpřístupnit data kdekoliv v síti.
Dále se dělí na dva typy:
• Federativní databáze – neexistuje globální schéma ani centrální řídící autorita, řízení je také distribuované.
• Heterogenní databázové systémy – jednotlivé autonomní SRBD existují (vznikly nezávisle na sobe) a jsou integrovány, aby spolu mohly komunikovat. Výhodou oproti centralizovaným systémům je vyšší efektivita (data mohou být uložena blízko místa nejčastějšího používání), zvýšená dostupnost, výkonnost a rozšiřitelnost; nevýhodou zůstává problém složitosti implementace, distribuce řízení a nižší bezpečnost takových řešení.

Víceuživatelské databázové systémy

Víceuživatelské jsou takové systémy, které umožňují vícenásobný uživatelský přístup k datům ve stejném okamžiku. V důsledku možného současného přístupu více uživatelů je nutné systém zabezpečit tak, aby i nadále zajišťoval integritu a konzistenci uložených dat. Existují obecně dva možné přístupy:
• Uzamykání – Dříve často používaná metoda založená na uzamykání aktualizovaných záznamu, v případe masivního využití aktualizačních příkazů u ní ale muže docházet k značným prodlevám.
• Multiversion Concurency Control – Modernější vynález. Jeho princip spočívá v tom, že při požadavku o aktualizaci záznamu v tabulce je vytvořena kopie záznamu, která není pro ostatní uživatele až do provedeného commitu viditelná.

Architektura databázových strojů

Základní architektura databázového stroje [DBS] je pro všechny databáze podobná. Základní architektura se řídí přibližně tímto diagramem.



Jinými slovy, DBS musí zpracovávat příkazy od 4 základních skupin uživatelů.

1. Správce dat neboli databázový administrátor [DBA] - je jediný, kdo je schopen přistupovat přímo k datům v databázi a měnit přímo její strukturu. K této práci používá příkazy Data Definition Language neboli DDL, tyto Database management system [DBS] zpracovává pomocí kompilátoru DDL, který je kompiluje přímo do datového slovníku. Dále má DBA také přístup k privilegovaným příkazům, které slouží například ke spouštění, zastavování nebo restartování DBS.
2. Příležitostný uživatel - nemá přístup ani k DDL příkazům, ani k privilegovaným příkazům, ale připojuje se přímo k databázi a své dotazy posílá přímo v Data Manipulation Language [DML] (většinou se jedná o jazyk na bázi SQL). Je schopen číst a měnit data v databázi, ale není schopen měnit její strukturu (má přístup k příkazům jako SELECT, INSERT, UPDATE, ale ne CREATE TABLE, ALTER TABLE, atd.). O tyto dotazy se stará kompilátor DML, který je kompiluje do vnitřní formy databázového stroje.
3. Aplikační programátor - tvoří aplikační programy, které mají stejné možnosti, jako příležitostný uživatel, ale obalují logiku dotazů do uživatelsky přívětivějšího kabátku.
4. Naivní uživatel - přistupuje k datům pouze přes aplikační rozhranní. O struktuře databáze nemá většinou absolutně žádné ponětí, pouze zadá dotaz do aplikace a dostane odpověď.

Zpracování dotazu:

1. DDL příkazy jsou pouze zkompilovány a rovnou předány databázi.
2. DML dotazy, ať už od uživatele, nebo přes aplikační rozhranní jsou zpracovány následovně:
    a. Uživatel se přihlásí k databázi, jeho přihlašovací údaje jsou zkontrolovány proti tabulce uživatelů. Může být i kontrolováno, zda se přihlašuje ze správného počítače (podle IP adresy).
    b. Je provedena kontrola, zda přihlášený uživatel má právo provádět tento dotaz.
    c. Dotaz přijde do DML kompilátoru, který jej přeloží na low-level instrukce.

- U aplikací se často používá prekompilátor DML, který má časté dotazy dopředu zkompilovány, čímž urychluje hledání výsledku.

d. Zkompilovaný dotaz se následně pošle do vyhodnocovače dotazů, kde DBS zjistí, která data má vrátit jako odpověď. Toto je řízeno transakčním manažerem.
e. Následně jde dotaz přes manažer bufferů a souborů, které zařídí správné vyhledání v indexech a navrácení správných dat jako odpověď na dotaz.
f. Tato odpověď jde zpátky do vyhodnocovače dotazů, který jej vrátí uživateli.

Příkazy DDL	Příkazy pro definici struktury tabulek/databáze
Privilegované příkazy	Přidělování uživatelských práv, nastavování parametrů SŘBD, apod.
Kompilátor DDL	Zpracovává definici schématu a ukládá jí do slovníku dat.
Kompilátor DML	Kompiluje do programů nižší úrovně přístupu k databázi. Tento kód je připojen ke zbytku aplikačního programu či předdefinované uživatelské transakci.
Prekompilátor vloženého DML	Zpracovává příkazy DML vyskytující se v hostitelském jazyku.
Vyhodnocovač dotazů	Je zřízen pro dotazy v dotazovacím jazyku. Interpretuje nebo kompiluje dotaz, případně zařizuje jeho optimalizace (za použití např. statistických dat) a komunikuje přímo přes manažer bufferů (a souborů) s databází.
Zkompilované transakce	Zkompilované a uložené transakce.
Manažer souborů	Prostředník mezi operačním systémem a SŘBD v souvislosti s přenosem dat mezi diskem a vyrovnávacími paměťmi (buffery) ve vnitřní paměti počítače.
Manažer bufferů	Prostředník mezi vyhodnocovačem dotazů a manažerem souborů. Stará se o správné načítání dat ze souborů do bufferů a předávání již načtených dat vyhodnocovači dotazů.
Transakční manažer	Zpracování dotazů a dalších uživatelských transakcí.
Slovník dat	Množina metadat, ve kterých jsou organizovány veškeré definice týkající se logického a fyzického schématu databáze. Z hlediska SŘBD se jedná o skupinu tabulek (a pohledů), do kterých nelze zapisovat. Datový slovník mimo jiné obsahuje: přesnou definici datových prvků, uživatelská jména, role a privilegia, schémata, integritní omezení, uložené procedury a uložené procedury (stored procedures) a spouštěče (triggers), obecnou strukturu databáze, přidělení místa (space allocation). Datový slovník umožňuje zachovat konzistenci dat v různých tabulkách
Statistická data	Statistická údaje o rozložení dat v tabulce.
Indexy	Datová struktura, která umožňuje rychlejší prohledávání tabulek. (viz níže)
Aplikační data	Data uložená v databázi.