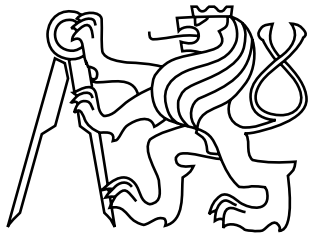
DBS - Fyzický pohled na data

Michal Valenta

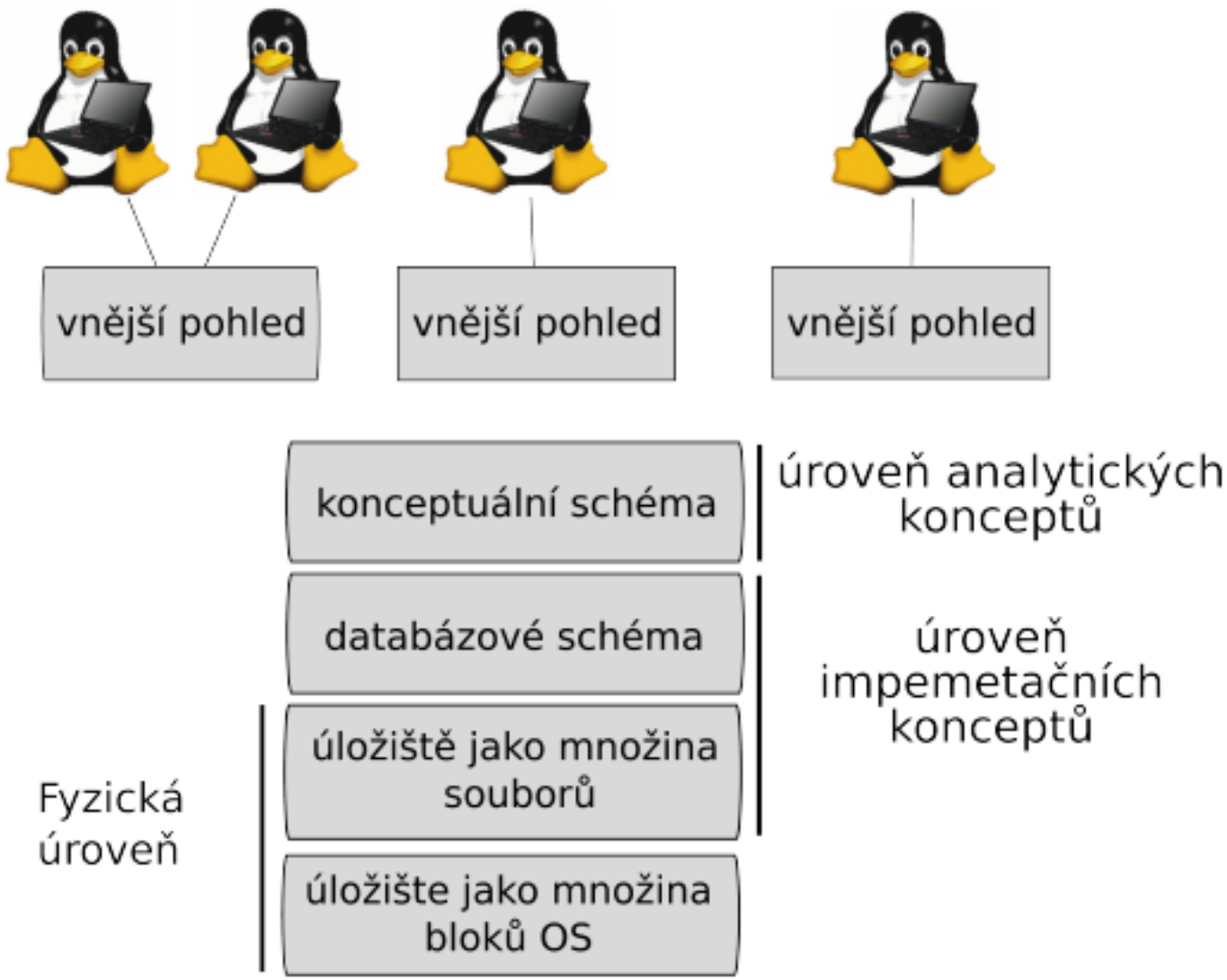
Katedra softwarového inženýrství FIT
České vysoké učení technické v Praze

BI-DBS, ZS 2014/15

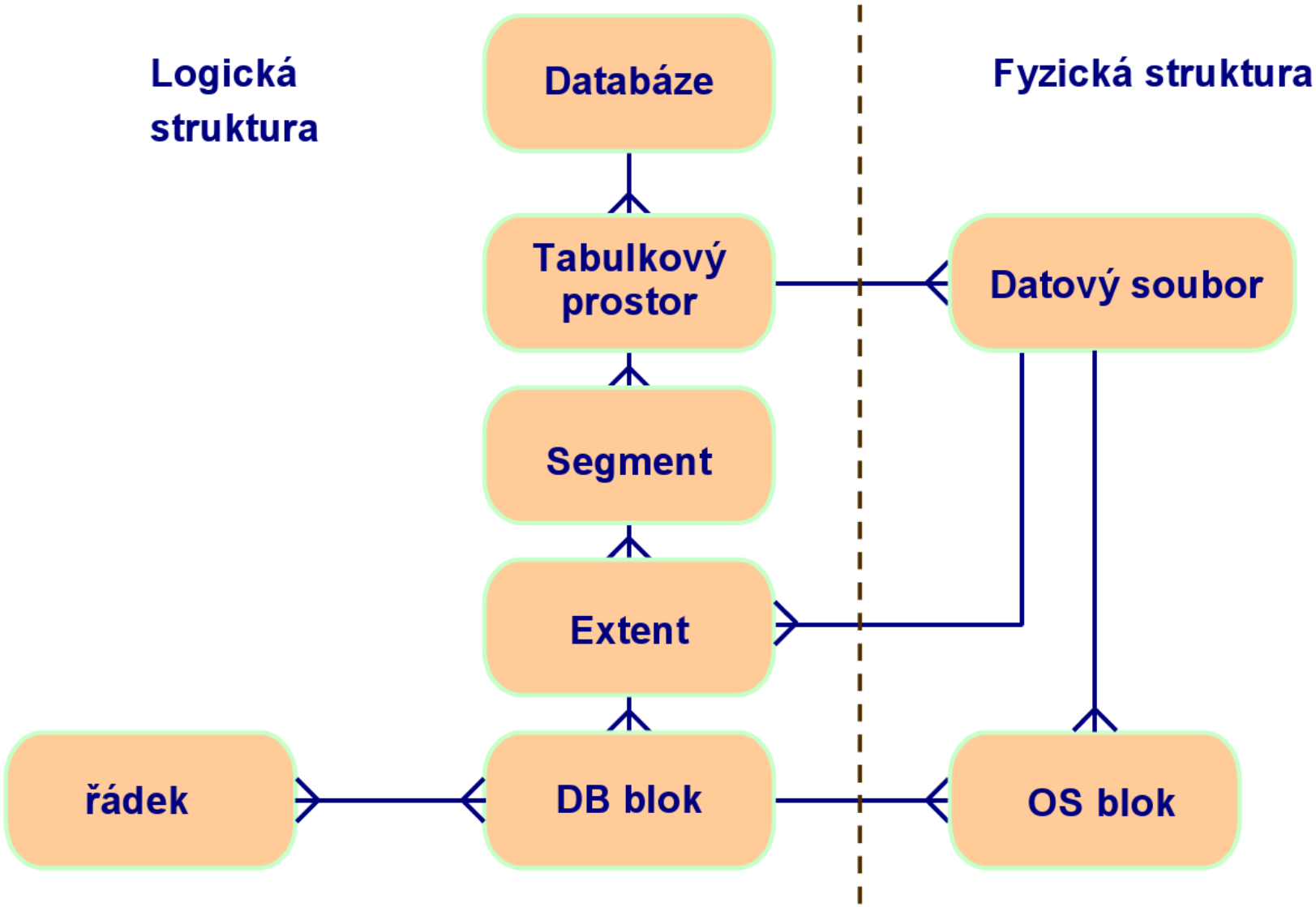
https://edux.fit.cvut.cz/courses/BI-DBS/



Různé úrovně pohledu na data

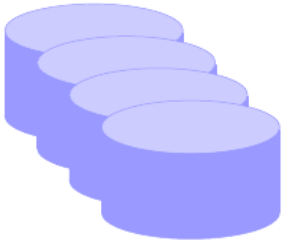


Oracle – fyzická a logická struktura databáze

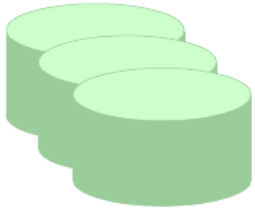


Oracle – fyzická struktura databáze

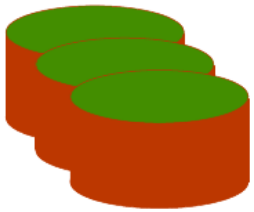
Řídicí soubor a jeho zrcadlení



Datové soubory



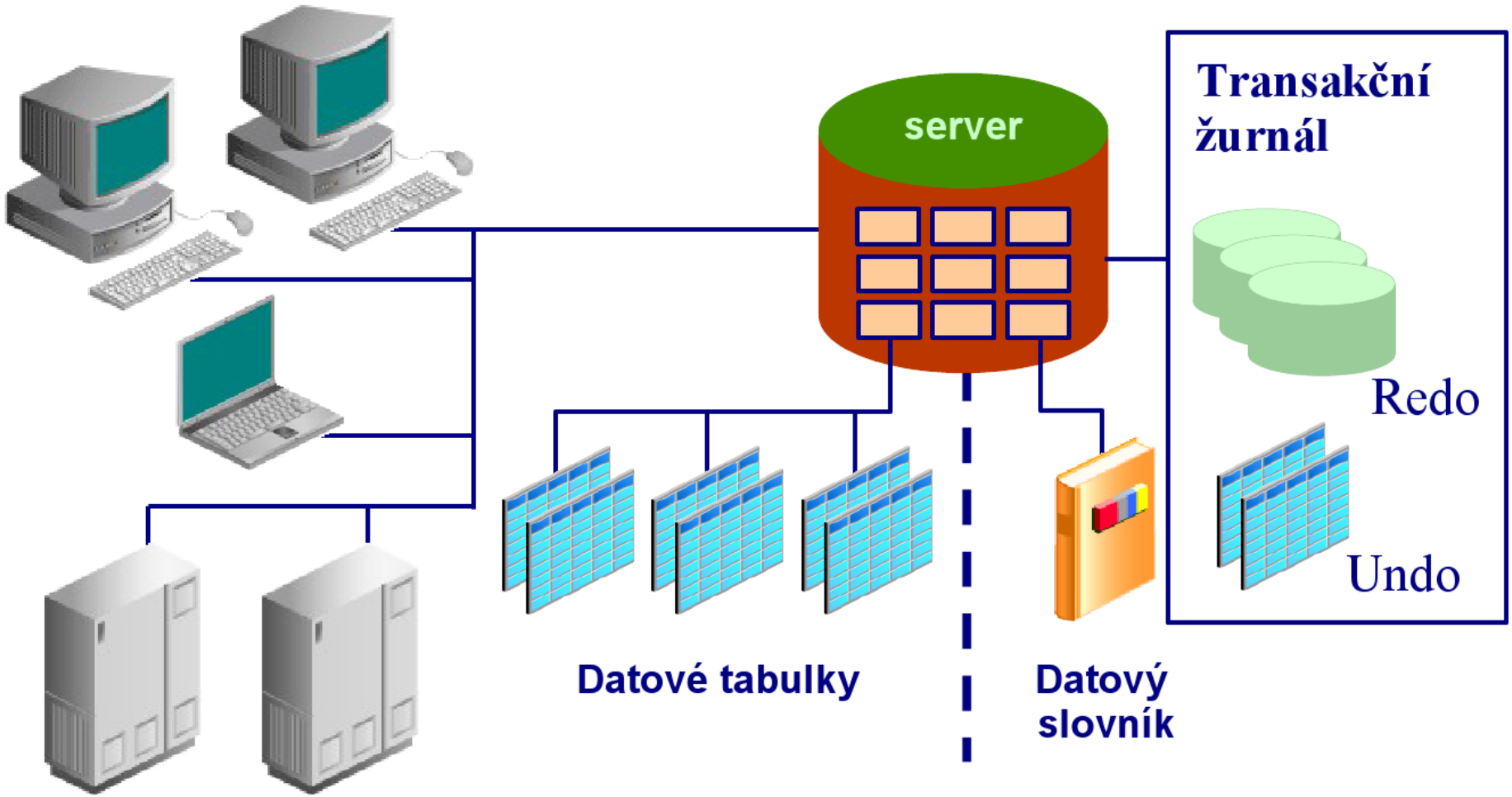
Online redo log soubory a jejich zrcadlení



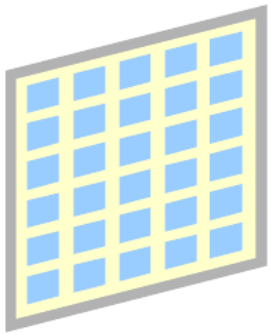
Archivní kopie redo log souborů

Parametry

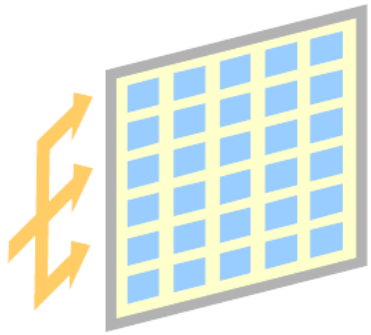
Dvou a vícevrstvá architektura



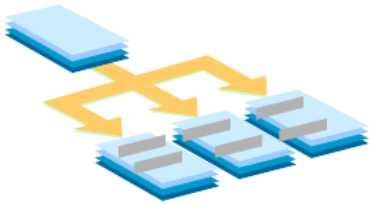
Oracle – fyzická organizace relační tabulky



HEAP tabulka



HEAP tabulka s indexy



Tabulka s indexovou organizací

tabulka ve shluku (cluster)

Adresa řádku (Oracle, ale podobně i jinde)

```
select rowid, t.* from Titul t
ROWID                          TITUL_ID  NAZEV              ROK_VYROBY
------------------------------ --------- ------------------ ----------
AAAsaTAAHAAA656AAA             1         Název_titulu _1    01.01.2005
AAAsaTAAHAAA656AAB             2         Název_titulu _2    01.01.2005
AAAsaTAAHAAA656AAC             3         Název_titulu _3    01.01.2005
. . .
```

Asociativní výběr – „prohrabání hromady“

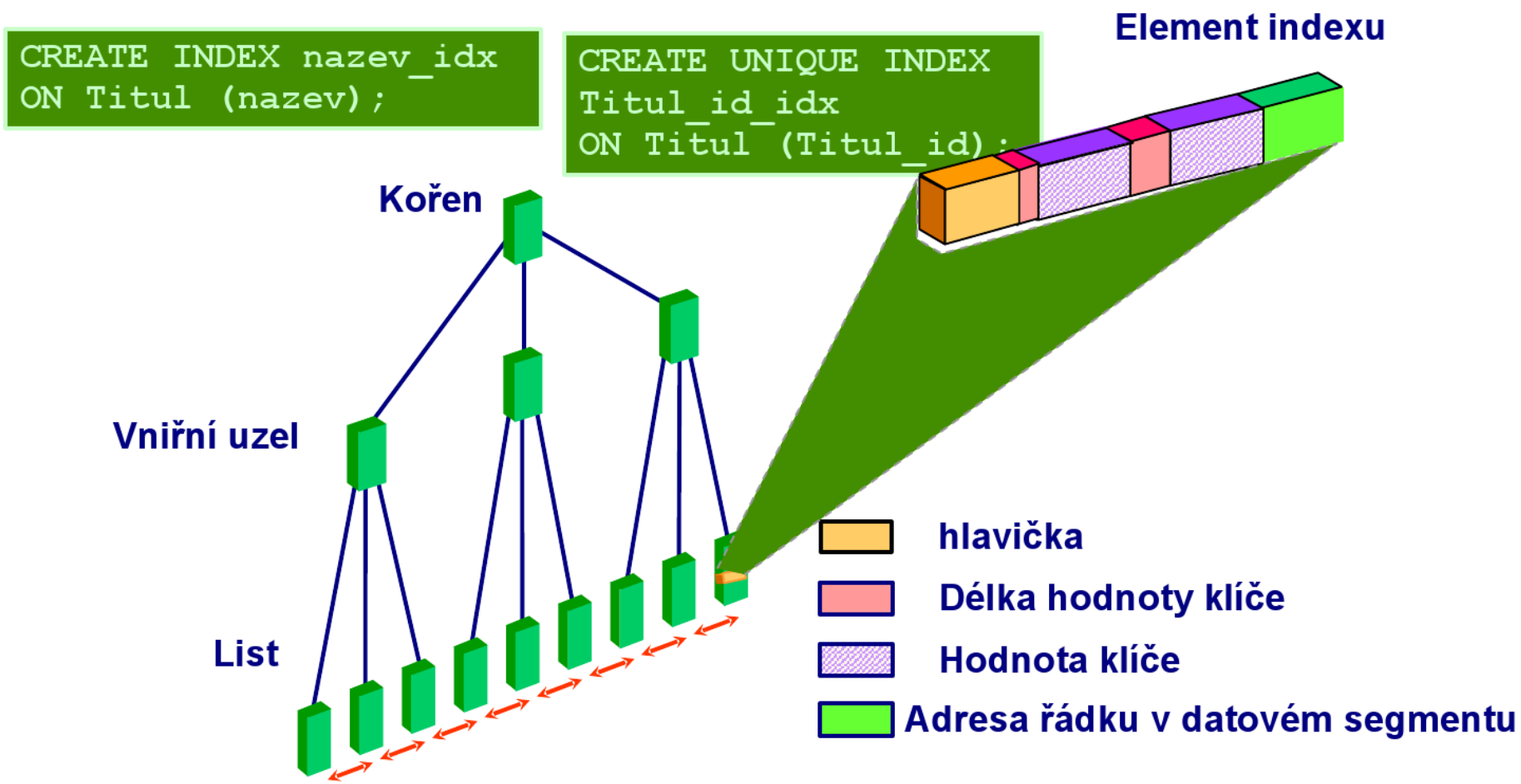
```
select * from Titul t where NAZEV='Název_titulu _2'
```

Adresní výběr (nadrelační rys):

```
select * from Titul t where ROWID ='AAAsaTAAHAAA656AAB'
```

Index typu B*-Tree

CREATE INDEX nazev_idx on Titul (nazev);
CREATE UINIQUE INDEX titul_id_idx on Titul (titul_id);



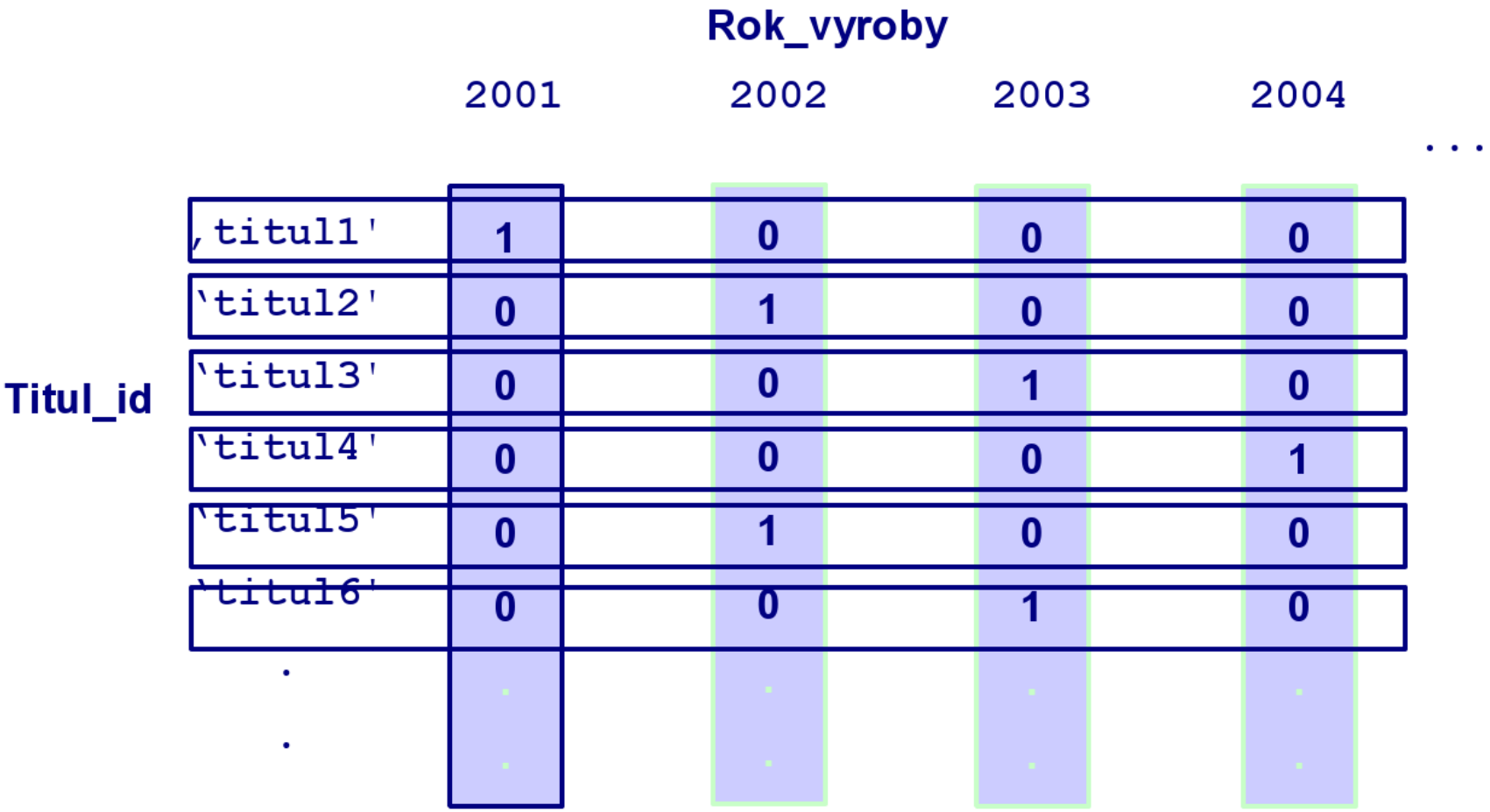
Index typu B-Tree - charakteristika

B - strom (řádu m) je m-ární strom, splňující následující omezení

- Kořen má nejméne 2 potomky, pokud není listem
- každý uzel kromě kořene a listu má nejméně [m/2] a nejvýše m potomků
- každý uzel ma nejméně $\lceil m/2 \rceil - 1$ a nejvíce $m - 1$ datových záznamů (většinou jen klíčů)
- všechny cesty ve stromě jsou stejně dlouhé
- data v nelistovém ulzu jsou organizována
  $p_0(k_1, p_1), (k_2, p_2), ..., (k_n, p_n), u$
  - kde $p_0, p_1, ..., p_n$ jsou ukazatele na potomky
  - $k_0, k_1, ..., k_n$ jsou klíče
  - u je nevyužitý prostor
  - záznamy(k_i, p_i) jsou uspořádany vzestupně podle klíčů
  - $\lceil m/2 \rceil - 1 <= n <= m - 1$
- odpovídá-li ukazateli p_i, kde $i \in < 1, n >$ podstrom $U(p_i)$ platí:
  - (i) pro každé $k \in U(p_{i-1})$ je $k <= k_i$
  - (ii) pro každé $k \in U(p_i)$ je $k > k_i$
- listy obsahují úplnou množinu klíčů a mohou se lišit strukturou.

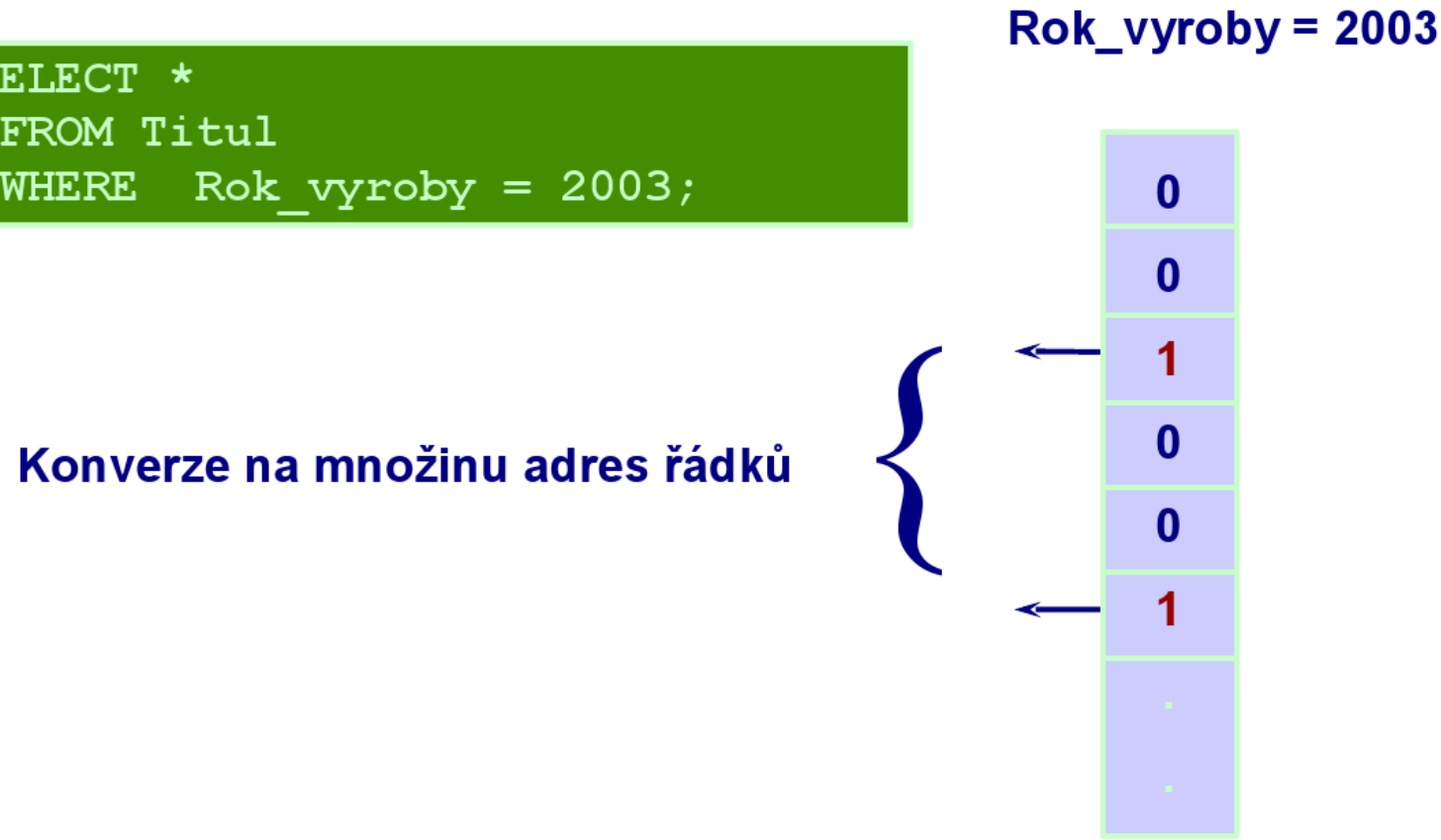
Bitmapové indexy

```
CREATE BITMAP INDEX rok_id_bix
ON Titul (rok_vyroby);
```



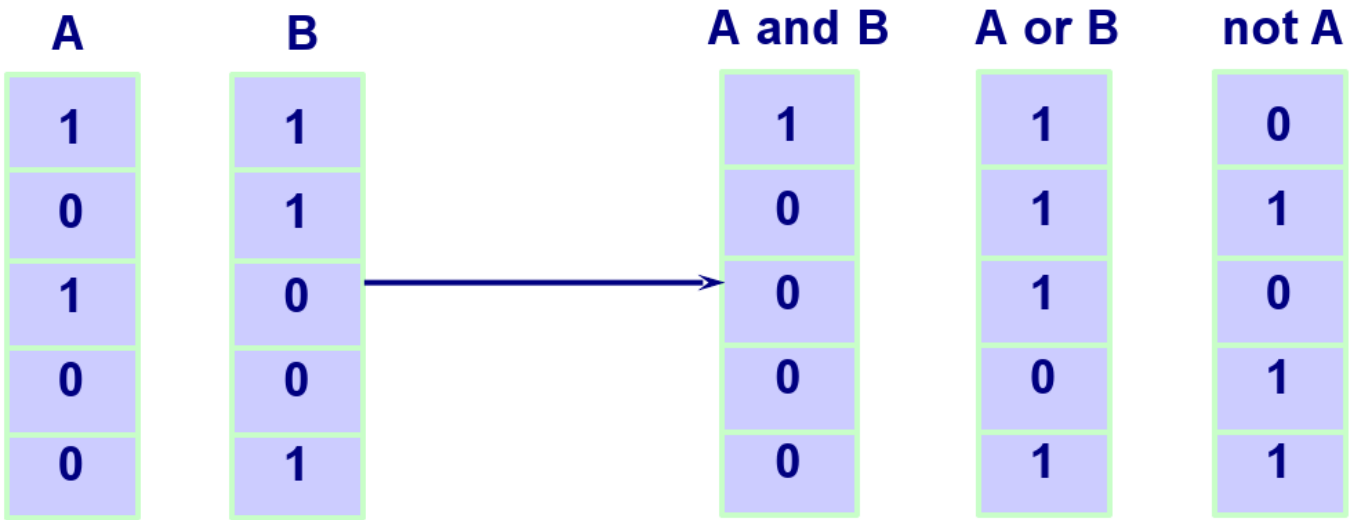
Použití bitmapového indexu při vyhodnocení dotazu

```
SELECT *
 FROM Titul
 WHERE  Rok_vyroby = 2003;
```



Použití bitmapového indexu při vyhodnocení dotazu

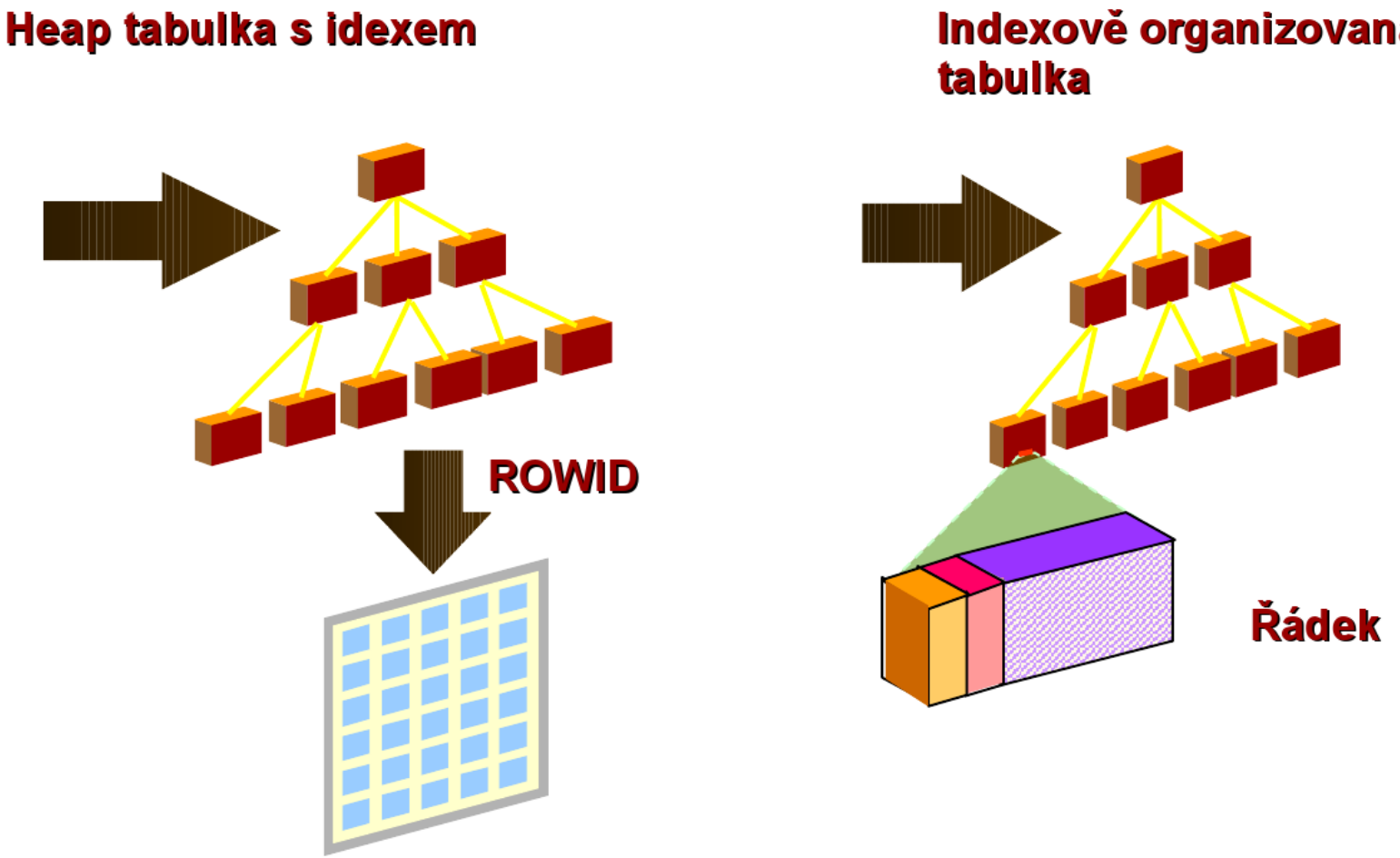
- Výběrová podmínka s operátorem IN
- Výberová podmínka s operátorem AND/OR



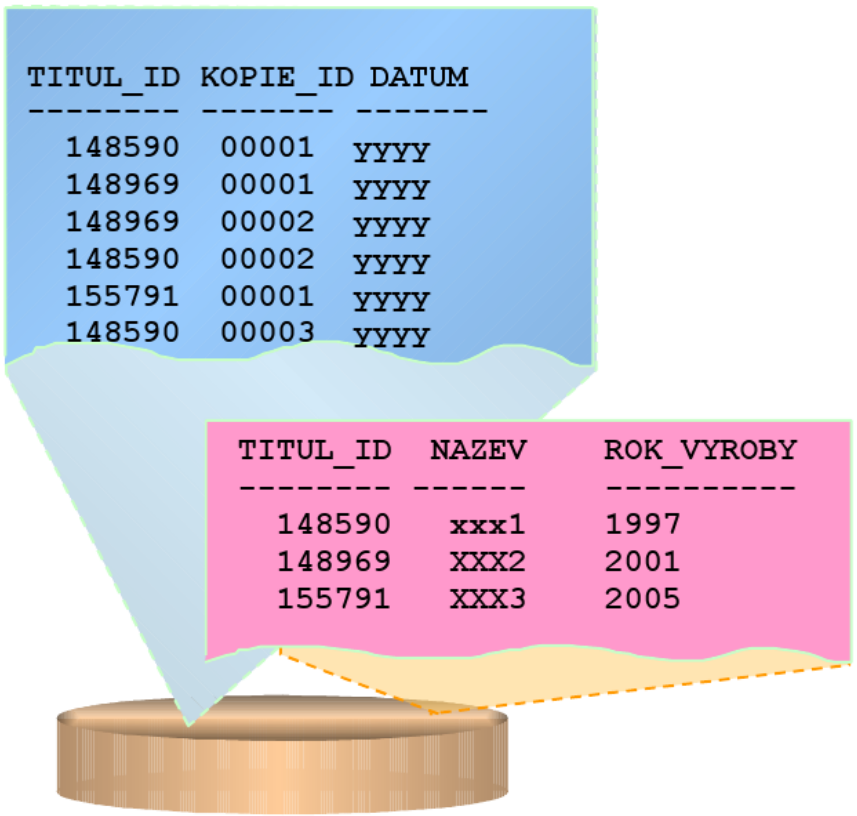
Porovnání indexů B-strom a Bitmap

B-strom	Bitová mapa
Sloupce s vysokou kardinalitou	Sloupce s nízkou kardinalitou
DML operace relativně drahé	DML operace velmi drahé
Vhodné pro OLTP	Vhodné pro ad hoc dotazy v datových skladech DSS

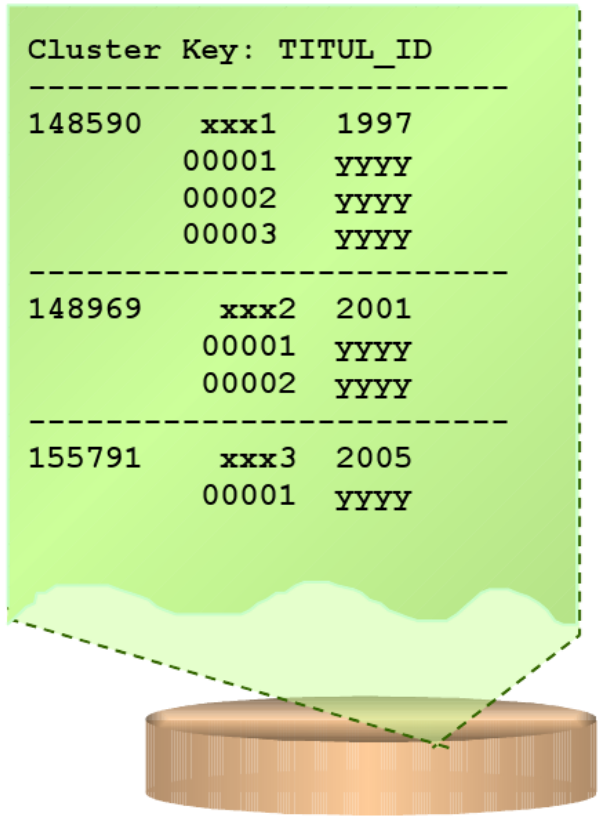
Indexově organizovaná tabulka (Index-organized table)



Shluk (Cluster)



Samostatné tabulky s vazbou přes cizí klíč



Tabulky ve shluku

Přístup k SQL z aplikace

- fáze zpracování dotazu
  - parse
  - bind
  - execute
  - fetch
  - + struktura pro návrat chybového kódu
- standardizované API a drivery (ODBC, JDBC, ...)
- specifické knihovny (PHP, ...)
- většinou API kopíruje fáze zpracování SQL dotazu

... a přístupy, které odstiňují používání databáze:

- hibernate, tapestry, ...
- Ruby on Rails, JAXB, ...

Další techniky zrychlení přístupu k datům

Připomenutí pojmů:

- OLTP versus DSS systémy
- optimalizace dotazu, prováděcí plán, cena dotazu
- fáze zpracování dotazu (příkazu)

Používané techniky:

- zavedení redundance, materializované pohledy
- využití vyrovnávacích pamětí pro data, ale také pro parsované SQL příkazy a kurzory

Optimalizace aplikace a databáze:

- návrh struktury (normalizace vs. denormalizace)
- speciální struktury pro uložení dat
- optimalizace dotazů (zjištění dotazů, které je třeba optimalizovat)
- systémové zdroje (paměť - velikost a struktura)
- konfigurace serveru (disky, parametry databáze, zálohování)