Porovnání webových technologií ASP, ASP.NET a PHP

Lukáš Kouřil

Bakalářská práce
2006

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2005/2006

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Lukáš KOUŘIL**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Porovnání skriptovacích jazyků PHP, ASP/ASP.NET**

Zásady pro vypracování:

Navrhněte sadu benchmarků pro měření výkonnosti systémů PHP a ASP/ASP.NET na serverech Linux a Windows. Benchmarky musí zahrnovat co nejvíce faktorů, které ovlivňují výkonnost celé webové aplikace – tzn. včetně použití různých databázových serverů. U jazyka PHP otestujte i použití PHP kompilátorů jako php Accelerator, Zend nebo Turck MMCache.

Na základě výsledků vašich testů navrhněte a popište správné techniky pro programování v daném prostředí pro dosažení maximálního možného výkonu. Tyto optimalizační techniky demonstrujte na příkladech.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

L. Welling a L. Thomson: PHP a MySQL – Rozvoj webových aplikací
Chris Payne: Naučte se ASP.NET za 21 dní
http://msdn.microsoft.com
http://www.php.net

Vedoucí bakalářské práce: **Ing. Tomáš Dulík**
Ústav aplikované informatiky

Datum zadání bakalářské práce: **14. února 2006**

Termín odevzdání bakalářské práce: **16. června 2006**

Ve Zlíně dne 14. února 2006

prof. Ing. Vladimír Vašek, CSc.
pověřený děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Cílem této práce je najít nejvýkonnější řešení v oblasti webových technologií včetně spolupráce s databázemi, což je v době moderního Internetu stěžejní problém při vytváření webových aplikací. Teoretická část seznamuje s dostupnými webovými technologiemi, se způsobem testování výkonnosti, prostředky potřebnými pro testování a s algoritmy užitými k měření výkonnosti. Praktická část podrobně probírá technologie, na něž je tato práce zaměřená, a obeznamuje se zjištěnými výsledky.

Klíčová slova: ASP, ASP.NET, PHP, SQL Server 2005 Express, MySQL, PostgreSQL, FirebirdSQL, Oracle, webové technologie, server, databáze, výkonnost

ABSTRACT

The aim of this work is find high performance solution in the sphere of web technologies, inclusive of cooperation with databases. In the era of modern Internet, this is a pivotal problem of web applications development. The theoretical part presents available web technologies, ways to performance testing, resources, that are necessary to testing and algorithms for performance measuring. The practical part goes separately through technologies that are aimed by this work and apprises of ascertained results.

Keywords: ASP, ASP.NET, PHP, SQL Server 2005 Express, MySQL, PostgreSQL, FirebirdSQL, Oracle, web technologies, server, databases, performance

PODĚKOVÁNÍ

Rád bych poděkoval mému vedoucímu práce, panu Ing. Tomáši Dulíkovi, za jeho neocenitelnou podporu, připomínky a rady, bez kterých by tato práce nevznikla.

Motto

„

Habemus duas aures, sed unum os,

ut plura audiamus quam loquamur.

“

Latinské přísloví

OBSAH

ÚVOD **10**
I TEORETICKÁ ČÁST **11**
1 PŘEHLED WEBOVÝCH TECHNOLOGIÍ **12**
1.1 ASP 12
1.2 PHP 12
1.3 ASP.NET 12
1.4 JSP 13
1.5 SSI 13
1.6 CGI 13
1.7 ISAPI 13
2 ZPŮSOBY PROVÁDĚNÍ TESTŮ **14**
2.1 HARDWARE 14
2.2 SOFTWARE NUTNÝ K BĚHU SKRIPTŮ 14
2.2.1 Operační systémy 14
2.2.2 Serverový software 14
2.2.3 Databázový software 14
2.2.4 Skriptovací prostředí 15
2.2.5 Ostatní software 15
3 TESTOVACÍ SKRIPTY **16**
3.1 MĚŘENÍ „HRUBÉHO“ VÝKONU 16
3.1.1 Eratostenovo síto 16
3.2 URČENÍ VHODNÉHO TYPU ALGORITMU 16
3.3 RYCHLOST ZPRACOVÁNÍ ŘETĚZCŮ 16
4 DATABÁZE **17**
4.1 PROPOJENÍ S DATABÁZÍ 17
4.1.1 Výběr z dat dotazem bez omezujících podmínek 17
4.1.2 Výběr z dat dotazem s jednoduchou omezující podmínkou 18
4.1.3 Výběr dat pomocí dotazu s vícenásobnými omezujícími podmínkami 18
4.1.4 Výběr dat pomocí dotazu s vloženým poddotazem 18
4.1.5 Dotaz s napojením dvou různých tabulek 18
4.2 UKÁZKY ČÁSTÍ TABULEK 19
II PRAKTICKÁ ČÁST **21**
5 ASP **22**

5.1 PODROBNÝ POHLED NA ASP 22
5.2 PROSTŘEDÍ PRO PROVÁDĚNÍ TESTŮ 23
5.3 TESTOVACÍ SKRIPTY 23
5.3.1 Práce s řetězci 23
5.3.2 Druh algoritmu 23
5.3.3 Hrubý výkon 24
5.3.4 Práce s databázemi 25
5.4 VÝSLEDKY TESTŮ 27
5.4.1 Skripty pro určení výkonu ASP 27
5.4.2 ASP a databáze Microsoft SQL Server 2005 Express 29
5.4.3 ASP a databáze FirebirdSQL 31
5.4.4 ASP a databáze Oracle 33
5.4.5 Srovnání výkonu databází u ASP 35
6 PHP 36
6.1 PODROBNÝ POHLED NA PHP 36
6.2 KOMPILÁTORY PHP 36
6.3 PROSTŘEDÍ PRO PROVÁDĚNÍ TESTŮ 37
6.4 TESTOVACÍ SKRIPTY 37
6.4.1 Práce s řetězci 37
6.4.2 Druh algoritmu 38
6.4.3 Hrubý výkon 39
6.4.4 Práce s databázemi 40
6.5 VÝSLEDKY TESTŮ NA OPERAČNÍM SYSTÉMU MANDRIVA LINUX 2006 42
6.5.1 Skripty pro určení výkonu PHP 42
6.5.2 PHP a databáze MySQL 45
6.5.3 PHP a databáze PostgreSQL 49
6.5.4 PHP a databáze FirebirdSQL 53
6.5.5 PHP a databáze Oracle 57
6.5.6 Srovnání výkonu databází u PHP pod operačním systémem Linux 61
6.6 VÝSLEDKY TESTŮ PHP NA OPERAČNÍM SYSTÉMU MICROSOFT WINDOWS XP PROFESSIONAL SP2 62
6.6.1 Skripty pro určení výkonu PHP 62
6.6.2 PHP a databáze MySQL 65
6.6.3 PHP a databáze PostgreSQL 69
6.6.4 PHP a databáze FirebirdSQL 73
6.6.5 PHP a databáze Oracle 77
6.6.6 Srovnání výkonu databází u PHP pod operačním systémem Windows XP 81
6.7 SROVNÁNÍ VÝSLEDKŮ PHP 82
6.7.1 Výkon PHP 82
6.7.2 PHP a databáze MySQL 83
6.7.3 PHP a databáze PostgreSQL 84
6.7.4 PHP a databáze FirebirdSQL 85
6.7.5 PHP a databáze Oracle 86

6.7.6 Celkové srovnání databází a PHP 87
7 ASP.NET 88
7.1 TECHNOLOGIE .NET FRAMEWORK 88
7.2 PODROBNÝ POHLED NA ASP.NET 88
7.3 PROSTŘEDÍ PRO PROVÁDĚNÍ TESTŮ 89
7.4 TESTOVACÍ SKRIPTY 90
7.4.1 Práce s řetězci 90
7.4.2 Druh algoritmu 91
7.4.3 Hrubý výkon 92
7.4.4 Práce s databázemi 94
7.5 VÝSLEDKY TESTŮ 96
7.5.1 Skripty pro určení výkonu ASP.NET 96
7.5.2 ASP.NET a databáze Microsoft SQL Server 2005 Express 98
7.5.3 ASP.NET a databáze FirebirdSQL 100
7.5.4 ASP.NET a databáze Oracle 102
7.5.5 Srovnání výkonu databází u ASP.NET 104
7.6 MONO - ALTERNATIVNÍ IMPLEMENTACE .NET FRAMEWORKU 104
7.6.1 Výsledky ASP.NET pod Monem 105
8 VYHODNOCENÍ 109
8.1 VYHODNOCENÍ PHP 109
8.1.1 Zpracování skriptů 109
8.1.2 Databáze 109
8.1.3 Řešení PHP 110
8.2 CELKOVÉ VYHODNOCENÍ 110
8.2.1 Nejvýkonnější technologie pro vývoj webových aplikací 111
8.2.2 Databáze 111
8.2.3 Nejvýkonnější databázový server 112
8.2.4 Nejvýkonnější dostupné efektivní řešení 112
8.3 PROČ ZVOLIT ASP.NET 112
8.3.1 Přehled free webhostingů pro ASP.NET 113
8.4 PROČ ZVOLIT PHP 113
8.4.1 Přehled free webhostingů pro PHP 113
9 TECHNIKY OPTIMALIZACE 115
9.1 OPTIMALIZACE ASP.NET 115
9.1.1 Šířka pásma 115
9.1.2 Přístup k databázím 115
9.1.3 Vhodné používání Cache 116
9.1.4 „Správné“ programování 116
9.2 OPTIMALIZACE PHP 116
9.2.1 Předgenerovaný obsah 116
9.2.2 Konfigurace PHP 116
9.2.3 Otevírání souborů 117
9.2.4 Práce se sessions 117

ZÁVĚR **118**
SEZNAM POUŽITÉ LITERATURY **119**
SEZNAM OBRÁZKŮ **120**
SEZNAM TABULEK **122**
SEZNAM VÝPISŮ ZDOJOVÝCH KÓDŮ **124**
SEZNAM PŘÍLOH **125**

ÚVOD

Výkonnost technologií pro tvorbu webových aplikací je stejně důležitá jako motor v automobilu. Stejně, jako obchodní zástupce na cestě ke klientovi potřebuje rychlý vůz, aby dorazil včas a nenechal zákazníka čekat, tak i webové aplikace potřebují dostatečně výkonné technologie, aby běželi svižně a jejich potenciální zákazníci, byť i s pomalejším připojením k Internetu, nemuseli dlouho čekat a neklikli mezitím vedle ke konkurenci.

Tato práce si klade za cíl provést výkonnostní testy technologií ASP, ASP.NET a PHP, včetně propojení s nejrozšířenějšími databázemi a provést jejich srovnání. Tím bude možné, snad jednoznačně, určit, jaký „motor“ má pohánět výkonnou webovou aplikaci.

Podobných testů jako je tento, je možné nalézt na Internetu spousty. Rovněž existují i komplexnější testy jako např. test[7], dále reálné testy s webovými systémy, měření reálné rychlosti generování stránek při zatěžování z více PC, viz. test[8], a pod. Některé z nich jsou uvedeny v příloze P1.

Výkonnost samotných skriptovacích technologií se dá předpokládat už z jejich koncepce. Nelépe by mělo dopadnout ASP.NET, které využívá kompilace, a tím i rychlého zpracování. Na dalším místě by pak měly skončit technologie ASP a PHP, které jsou interpretované. Zda-li se tyto předpoklady ověří, ukážou až výsledky měření.

Velká část testování je věnována měření výkonnosti databázových serverů. Jejich výkonnost se nedá, jako v případě skriptovacích technologií, předvídat. Podle dostupných testů, viz. příloha P1, se výsledky databází rozcházejí a je těžké se rozhodnout, ke kterému výsledku se přiklonit. Po podrobnějším pohledu je tento rozptyl výsledků zapříčiněn i z důvodů chyb při měření, jako je tomu např. u testu[1].

Testovací skripty v této práci se řadí mezi umělé, tzn. jsou vytvořeny pouze za účelem měření a nemají praktické využití.

K měření výkonnosti bude přistupováno s co největší objektivitou a s testy, které by mohly co nejvíce vypovědět o výkonu dané technologie a databáze. Navíc, na rozdíl od podobných testů, bude testování provedeno i na rozdílných operačních systémech a webových serverech.

I. TEORETICKÁ ČÁST

1 PŘEHLED WEBOVÝCH TECHNOLOGIÍ

Na Internetu je možné se setkat s dynamickými stránkami vytvořenými v mnoha skriptovacích technologiích. Od webů s dynamickými prvky psanými v SSI, CGI, ISAPI, které patří mezi nejstarší technologie, až po ty nejnovější. Jsou to ASP a PHP, dvě navzájem konkurující si technologie, ASP.NET, náhradník starého interpretovaného ASP, ze kterého se stalo robustní kompilované moderní prostředí pro vývoj webu, a JSP – Java Server Pages.

1.1 ASP

ASP (*Active Server Pages*) společnosti Microsoft je již poměrně stará skriptovací technologie, která v současné době není dále vyvíjena a která skončila u verze 3.0. Jako jazyk ASP je možné zvolit JavaScript nebo VBScript, které však nejsou prováděny na straně klienta, ale na serveru.

1.2 PHP

PHP (*Hypertext Preprocesor*) je v podstatě alternativa k ASP, která je Open Source. Stále je rozšiřována a zatím poslední nejnovější verze nese číslo 5. Jde přímo o skriptovací jazyk, založený na jazyku C/C++, Javě a Perlu, který je interpretovaný serverem. Podporuje pluginy, které umožňují např. on-line vytváření obrázků v různých formátech, generování PDF, práci se ZIP archivy, zpracování kreditních karet a finančních transakcí pomocí Verisign Payment Services a mnoho dalšího. Umožňuje psát objektově orientovaný kód.

1.3 ASP.NET

ASP.NET je prostředí pro vývoj webových aplikací, které využívá .NET Frameworku. Nejnovější verze má číslo 2.0. Umožňuje psát aplikace a stránky v libovolném jazyku, který umožňuje kompilaci do mezikódu. Patří sem např. Visual Basic.Net, Visual C++, C#, J#, Cobol a další. Webové aplikace jsou kompilované, takže dosahují vysokých výkonů.

Další technologie jsou uvedeny pouze pro zajímavost a nebudou zahrnuty do testování.

1.4 JSP

JSP (*Java Server Pages*), obdoba ASP a ASP.NET, produkt Sun Microsystems. Je založena na technologii JAVA a rovněž tento jazyk i využívá k psaní kódu. Ke svému běhu potřebuje tzv. Java Virtual Machines, které mají obdobnou funkci jako CLR u ASP.NET (viz. kapitola 7.2). Do testů není tato technologie zahrnuta, neboť přesahuje rámec této práce. Java je podle testu[7] cca 2x pomalejší než technologie .NET, a je tak možné ke zjištění výkonu přepočítat výsledky ASP.NET.

1.5 SSI

SSI (*Server Side Includes*) je nejjednodušší způsob dynamického generování internetových stránek. Zapisuje se do HTML dokumentu jako poznámky ve tvaru *<!-- příkaz parametr="hodnota" -->*.

1.6 CGI

CGI (*Common Gateway Interface*) se objevilo přibližně ve stejné době jako SSI, které zajišťuje komunikaci externích aplikací s webovými servery. CGI umožňuje spouštět libovolné programy a předávat jim parametry. Nevýhodou jsou problémy se zabezpečením aplikací.

1.7 ISAPI

ISAPI (*Internet Server Application Programming Interface*) je v podstatě alternativou k CGI. Pokud uživatel volá program CGI, vždy se spouští znova (server vždy spouští samostatnou instanci programu). Jestliže tedy poslalo požadavek mnoho uživatelů, mohou se brzy vyčerpat zdroje serveru. ISAPI tento problém řeší tím, že při prvním požadavku na spuštění aplikace, ji nahraje do paměti, ve které zůstane a stále reaguje na požadavky uživatelů. Technologie ISAPI umožňuje kromě jiného také vytváření filtrů, což jsou vlastně knihovny DLL. Tyto filtry tvoří rozhraní mezi uživatelem a serverem.

2 ZPŮSOBY PROVÁDĚNÍ TESTŮ

V rámci co největší objektivity jsou testy provedeny pouze na jediném počítači s různými operačními systémy a potřebným softwarem. Díky tomu, je při zjištění skutečné výkonnosti umožněno dostatečně kvalitní porovnání jednotlivých technologií, aniž by byly výsledky ovlivněny různou hardwarovou konfigurací.

2.1 Hardware

- procesor AMD Athlon 1.4GHz, jádro Thunderbird, FSB266
- základní deska ECS K7S5AL, SIS735, FSB266
- paměť 512MB DDR400, Transcend JM CL2.5
- harddisk Seagate 7200.7, 80GB, UATA100

2.2 Software nutný k běhu skriptů

2.2.1 Operační systémy

- Microsoft Windows XP Professional SP2
- Mandriva Linux 2006

2.2.2 Serverový software

- Microsoft IIS 5.0 *(Internet Information Services)*
- Apache 2.0.54
- Mono 1.1.15

2.2.3 Databázový software

- Microsoft SQL Server 2005 Express
- MySQL 5.1.17

- PostgreSQL 8.1.2
- FirebirdSQL 1.5.3
- Oracle 10.2 Express

2.2.4 Skriptovací prostředí

- ASP 3.0
- ASP.NET, .NET Framework 2.0
- PHP 5.1.2

2.2.5 Ostatní software

- Zend Optimizer 3.0.0

3 TESTOVACÍ SKRIPTY

Skripty jsou rozčleněny do různých oblastí, podle způsobu, jakým zatěžují server a skriptovací technologie. Tyto oblasti jsou uvedeny v následujících kapitolách.

3.1 Měření „hrubého“ výkonu

Tyto skripty slouží k zjištění rychlosti zpracování matematických algoritmů jednotlivými skriptovacími technologiemi. Pro měření jsou použity algoritmy pro výpočet Ludolfova čísla π a výpočet prvočísel pomocí Eratostenova síta.

3.1.1 Eratostenovo síto

Eratostenovo síto je slavný algoritmus, který zjišťuje počet prvočísel mezi číslem 2 a uživatelem zadaným stropem. V našem případě je tato hodnota 100000. Algoritmus je založen na postupném vypouštění násobků jednotlivých čísel ze seznamu od dvou do stanovené horní hranice. Po projití celé posloupnosti zůstane množina prvočísel menších než stanovená horní hranice.

3.2 Určení vhodného typu algoritmu

Porovnání rychlosti zpracování iterační a rekurzivní funkce. Rekurzivní funkce jsou mnohem kratší a elegantnější než iterační, naproti tomu jsou pomalejší a vyžadují více paměti, protože volají samy sebe a tím dochází k otevření mnoha instancí samotných funkcí.

3.3 Rychlost zpracování řetězců

Rychlost zpracování řetězců je u webových aplikací velmi důležitá. Testovací skript provádí obrácení řetězce o délce 100000 znaků a zjišťuje rychlost samotné reverze. U ASP.NET je porovnáno i zpracování řetězců pomocí komponenty StringBuilder, která má mnohonásobně urychlit práci s řetězci, u PHP je porovnána práce s řetězci pomocí lokálních a globálních proměnných.

4 DATABÁZE

Databáze dnes tvoří nedílnou součást internetových aplikací. Pří hledání nejvýkonnějšího řešení pro provoz webu je tudíž nezbytné zahrnout do testování i spolupráci skriptovacích technologií s databázemi. ASP a ASP.NET nejčastěji spolupracují s databázovými servery Microsoftu, a to SQL Serverem 2000, resp. MSDE 2000, nebo novějším SQL Serverem 2005. Testování je prováděno na Express verzi SQL Serveru 2005. PHP využívá v nejvíce případech databázových serverů MySQL a PostgreSQL, jenž oba budou zahrnuty do testování. Pro úplnost jsou do testů ASP, ASP.NET i PHP zahrnuty také databáze FirebirdSQL a Oracle.

4.1 Propojení s databází

Možnosti přístupu k databázím je nejdůležitější vlastnost skriptovacích technologií. Testy provádí dotazování na databázový server pomocí skriptů s různými dotazy. K testování jsou využity dvě tabulky, jejichž sloupce jsou různého typu, obsahující celkem 19615 záznamů, ze vzorové databáze AdventureWorksDB společnosti Microsoft.

Dotazy [1], [2], [3] a [4] vracejí všechny záznamy z tabulky 1. Rozdíl mezi těmito dotazy je pouze v odlišné realizaci.

Dotaz [5] je odlišný, jeho výsledkem je stále číslo 1, neboť dochází k zjišťování počtu stejných prvků u sloupce, jehož hodnoty jsou unikátní čísla. Je to docíleno vícenásobnými omezujícími podmínkami.

Dotaz [6] navrací všechny řádky z tabulky 2. V omezující podmínce jsou nejprve z tabulky 1 vybrány sloupce *id* u každého řádku, jejichž hodnota je větší než 0. Tím je zajištěno, že budou vybrány všechny řádky tabulky 1. Poté jsou vybrány z tabulky 2 všechny řádky, jejichž hodnoty ve sloupci *id* jsou stejné jako *id* zjištěné v podmínce.

Dotaz [7] spojí dvě tabulky pomocí sloupců id a navrátí takto upravené všechny záznamy.

4.1.1 Výběr z dat dotazem bez omezujících podmínek

Porovnání rychlosti zpracování navzájem ekvivalentních, ale jinak sestavených dotazů. Jsou to dotazy:

```
1 SELECT * FROM tabulka
2 SELECT (všechny_sloupce_tabulky) FROM tabulka
```

4.1.2 Výběr z dat dotazem s jednoduchou omezující podmínkou

Opět porovnání rychlosti zpracování navzájem ekvivalentních dotazů, které jsou ale odlišně sestaveny.

```
3 SELECT * FROM tabulka WHERE (id>0)
4 SELECT * FROM tabulka WHERE id BETWEEN 19615
```

4.1.3 Výběr dat pomocí dotazu s vícenásobnými omezujícími podmínkami

```
5 SELECT count(id) FROM tabulka WHERE (id>0) GROUP BY id HAVING (id>0)
  ORDER BY id
```

4.1.4 Výběr dat pomocí dotazu s vloženým poddotazem

```
6 SELECT (všechny_sloupce_tabulky2) FROM tabulka2 WHERE (id IN (SELECT
  id FROM tabulka1 WHERE (id > 0)))
```

4.1.5 Dotaz s napojením dvou různých tabulek

```
7 SELECT * FROM tabulka2 INNER JOIN tabulka1 ON tabulka2.id = tabul-
  ka1.id
```

4.2 Ukázky částí tabulek

id (i...	address1 (varchar)	address2 (varchar)	city (varchar)
4	9539 Glenside Dr	NULL	Bothell
5	1226 Shoe St.	NULL	Bothell
6	1399 Firestone Drive	NULL	Bothell
7	5672 Hale Dr.	NULL	Bothell
8	6387 Scenic Avenue	NULL	Bothell
9	8713 Yosemite Ct.	NULL	Bothell
10	250 Race Court	NULL	Bothell
11	1318 Lasalle Street	NULL	Bothell
12	5415 San Gabriel Dr.	NULL	Bothell
13	9265 La Paz	NULL	Bothell
14	8157 W. Book	NULL	Bothell
15	4912 La Vuelta	NULL	Bothell
16	40 Ellis St.	NULL	Bothell
17	6696 Anchor Drive	NULL	Bothell
18	1873 Lion Circle	NULL	Bothell
19	3148 Rose Street	NULL	Bothell
20	6872 Thornwood Dr.	NULL	Bothell
21	5747 Shirley Drive	NULL	Bothell

Obr. 1 – Tabulka "address1", část 1.

provinceid...	code (var...	guid (text)	date (date)
79	98011	e5946c78-4bcc-477f-9fa1-cc09de16a880	1999-03-07
79	98011	fbaff937-4a97-4af0-81fd-b849900e9bb0	1999-01-20
79	98011	febf8191-9804-44c8-877a-33fde94f0075	1999-03-17
79	98011	0175a174-6c34-4d41-b3c1-4419cd6a0446	2000-01-12
79	98011	3715e813-4dca-49e0-8f1c-31857d21f269	1999-01-18
79	98011	268af621-76d7-4c78-9441-144fd139821a	2002-07-01
79	98011	0b6b739d-8eb6-4378-8d55-fe196af34c04	1999-01-03
79	98011	981b3303-aca2-49c7-9a96-fb670785b269	2003-04-01
79	98011	1c2c9cfe-ab9f-4f96-8e1f-d9666b6f7f22	2003-02-06
79	98011	e0ba2f52-c907-4553-a0db-67fc67d28ae4	2004-01-15
79	98011	a1c658ae-c553-4a9d-a081-a550d39b64df	2000-01-05
79	98011	f397e64a-a9d8-4e57-9e7c-b10928acadd6	2003-12-20
79	98011	0312b65f-cb60-4396-9ec7-a78b2eac1a1b	2002-12-11
79	98011	ce9b3b47-9267-4727-bcd2-687c47482c06	2003-12-10
79	98011	963854f7-e3cb-46a1-a3db-1b05f71d6473	2004-01-01
79	98011	6b7acb0f-cdbf-44fd-ba14-eb08a56c1582	2004-05-04
79	98011	4b1f1ed4-97a4-43fd-bb1e-9e05817718e8	1999-03-09
79	98011	d83299d7-a0f4-4055-9bd5-5908e245d757	1999-03-15

Obr. 2 – Tabulka "address1", část 2.

uid (int8)	id (int8)	typeid (int8)	rowguid (text)	date (date)
1	832	3	314f2574-1f75-457f-9bd1-74d1ce53daa5	2001-08-01
2	297	5	c33e7580-8404-43f7-a783-21b927020b98	2002-08-01
2	833	3	b4435c21-4d7f-40e5-abea-4966428597d5	2002-08-01
3	559	3	991e29fa-e7c2-487a-bdd8-33ca4a36f61b	2001-09-01
4	560	3	bf40660e-40b6-495b-99d0-753ce987b1d1	2002-07-01
4	11380	5	7c6297be-ea24-433f-a038-50e082d341b8	2002-07-01
5	988	3	e968e6ac-47ae-4c04-8207-443b842de930	2002-09-01
6	989	3	3db4d2ff-611f-416c-a6e4-41292b13e04b	2003-09-01
7	990	3	33e5eb41-f032-42e1-ab6f-3386454b4543	2002-07-01
8	908	3	f74bacdc-44a5-4bdf-b17d-5960dc2523e3	2003-08-01
9	909	3	69f116d0-dba3-47ce-b7da-6d135279ab7b	2002-11-01
10	445	3	40e68361-485d-45f4-b18c-282d13f4552e	2002-09-01
11	446	3	26b1ebc5-570f-4d4b-baf2-c8c855b94151	2001-08-01
12	447	3	e25d9b43-edec-4e35-aa93-b12417ef7a61	2002-08-01
13	675	3	fed38668-ade8-4d01-bf5e-a12f0084a8e0	2002-08-01
14	715	3	8ad02a7c-2e91-4d21-a0a3-849c8468c1ec	2003-09-01
15	405	3	b60ff526-efdf-4406-bdc6-fbeac58643c4	2003-09-01
16	635	3	bff85ebd-ccd0-4daa-afbb-b415edf44265	2002-09-01

Obr. 3 – Tabulka “address2“

II. PRAKTICKÁ ČÁST

5 ASP

Active Server Pages jsou dnes velmi rozšířeným a výkonným nástrojem na tvorbu dynamických internetových stránek. Dá se pomocí nich lehce přistupovat k databázím, zapisovat do souborů na serveru apod.

Tuto technologii využívá například oblíbený český portál ATLAS, *<http://www.atlas.cz>*, který měsíčně provádí přes 1,5 miliónů požadavků nebo elektronický obchod firmy Dell, který týdně vyřizuje až 400 000 požadavků.

5.1 Podrobný pohled na ASP

Technologie ASP (Active Server Pages – aktivní serverové stránky) je filtr ISAPI vyvinutý společností Microsoft. Pokud uživatel požaduje soubor s příponou .asp, filtr ISAPI ASP interpretuje kód ASP a webovému serveru odešle jen čistý kód HTML (může však obsahovat i skripty vykonávané na straně uživatele, jako např. JavaScript), který jej následně odešle prohlížeči. Koncovému uživateli se pak stránka jeví jako stránka napsaná jen v HTML. ASP je otevřené prostředí, které je kompatibilní s libovolným ActiveX skriptovacím jazykem a jehož je i skriptovací jazyk základem. Žádny jazyk ASP však neexistuje. Lze používat skoro jakýkoliv skriptovací jazyk, nejvíce používané jsou však VBScript (Visual Basic Script) a JScript (což je implementace JavaScriptu od Microsoftu), které jsou podporovány jako výchozí. K psaní je však možné využít i jazyků jako jsou Perl, REXX, TCL, Python a další, které jsou implementovány v podobě pluginů. ASP také podporuje volání COM a DCOM komponent a tím umožňuje spouštět již zkompilované části kódu a využívat dříve napsané části aplikace. Je tak možné například přímo z ASP provádět operace v kancelářských aplikacích Microsoft Word a Excel, pomocí COM komponenty ADO komunikovat přes ODBC nebo OLE DB s ODBC databázemi. ASP je rovněž integrováno s Microsoft Transaction Serverem a umožňuje spouštění skriptů v transakcích.

5.2 Prostředí pro provádění testů

Skriptovací technologie ASP je doménou hlavně webových serverů společnosti Microsoft, běžících na operačních systémech téže společnosti. Jde o webové servery, resp. služby IIS (Internet Information Services) verze 5.0, nověji pak 6.0. Testování je proto provedeno na operačním systému Microsoft Windows XP Professional SP2 v kombinaci s IIS 5.0 a s databázovým serverem Microsoft SQL Server 2005 Express, FirebirdSQL 1.5.3 a Oracle 10.2 Express.

5.3 Testovací skripty

Dále je uveden kompletní přepis testovacích algoritmů v úpravě pro ASP. Připojení k databázím se provádí pomocí ADODB, takže není nutné provádět úpravu skriptů pro různé databáze.

5.3.1 Práce s řetězci

Prog. 1 – ASP: Práce s řetězci

```
s = ""
for i = 1 to 100000
   s = s & "a"
next
```

5.3.2 Druh algoritmu

Prog. 2 – ASP: Iterace

```
str = ""
s = ""
for i = 1 to 100
   str = str & "s"
next
function iterate (str)
   for i = (len(str)) to 1 step -1
      s = s & (mid(str,i,1))
   next
```

Pokračování

Prog. 2 – Pokračování

```
end function
for x = 1 to 1000
    iterate(str)
next
```

Prog. 3 – ASP: Rekurze

```
str = ""
s = ""
for i = 1 to 100
str = str & "a"
next
function recursion(str)
    if (len(str))>1 then
       recursion(mid(str,2))
    end if
    s = s & (mid(str,1,1))
 end function
 for x = 1 to 1000
    recursion(str)
 next
```

5.3.3 Hrubý výkon

Prog. 4 – ASP: Výpočet Ludolfova čísla

```
 pi_const = 3.141592
 pi = 0
 i = 1
 for x = 1 to 1000
    do
       pi = pi + (4 / i – 4 / (i + 2))
       i = i + 4
    loop while (pi <= pi_const)
 next
```

Prog. 5 – ASP: Eratostenovo síto

```
intval = 100000
dim cisla(100001)
limit = 2
   do while(limit*limit < intval)
   limit = limit + 1
   loop
   for i = 2 to limit
      if (cisla(i) = false) then
         for k = i + i to intval step +i
            cisla(k) = true
         next
      end if
   next
celkem = 0
for i = 2 to intval
   if (cisla(i) = false) then
      celkem = celkem + 1
   end if
next
```

5.3.4 Práce s databázemi

Prog. 6 – ASP: Práce s databází Microsoft SQL Server 2005 Express

```
function test1()
   set rs = conn.execute("SELECT * FROM dbo.address")
   set rs = nothing
end function


function test2()
   set rs = conn.execute("SELECT 'id','address1','address2',
         'city','provinceid','code','guid','date' FROM dbo.address ")
   set rs = nothing
end function


function test3()
```

Pokračování

Prog. 6 - Pokračování

```
    set rs = conn.execute("SELECT * FROM dbo.address WHERE (id > 0)")
    set rs = nothing
end function

function test4()
    set rs = conn.execute("SELECT * FROM dbo.address WHERE
          (id BETWEEN 1 AND 19615)")
    set rs = nothing
 end function


 function test5()
    set rs = conn.execute("SELECT count(id) FROM dbo.address
          WHERE (id > 0) GROUP BY id HAVING (id > 0) ORDER BY id")
    set rs = nothing
 end function


 function test6()
    set rs = conn.execute("SELECT 'uid','id','typeid','rowguid',
          'date' FROM dbo.address2 WHERE ('id' IN (SELECT 'id' FROM
           dbo.address WHERE (provinceid > 0)))")
    set rs = nothing
 end function


 function test7()
    set rs = conn.execute("SELECT * FROM dbo.address2 INNER JOIN
          dbo.address ON dbo.address2.id = dbo.address.id")
    set rs = nothing
 end function


 set conn = server.createobject("ADODB.Connection")
 conn.open("mssql")

 for x = 1 to 100
    test1()
 next
```

Prog. 6 - Pokračování

```
for x = 1 to 100
   test2()
 next
for x = 1 to 100
   test4()
 next
 for x = 1 to 100
   test5()
 next
 for x = 1 to 100
   test6()
 next
 for x = 1 to 100
   test7()
 next

 conn.close
 set conn = nothing
```

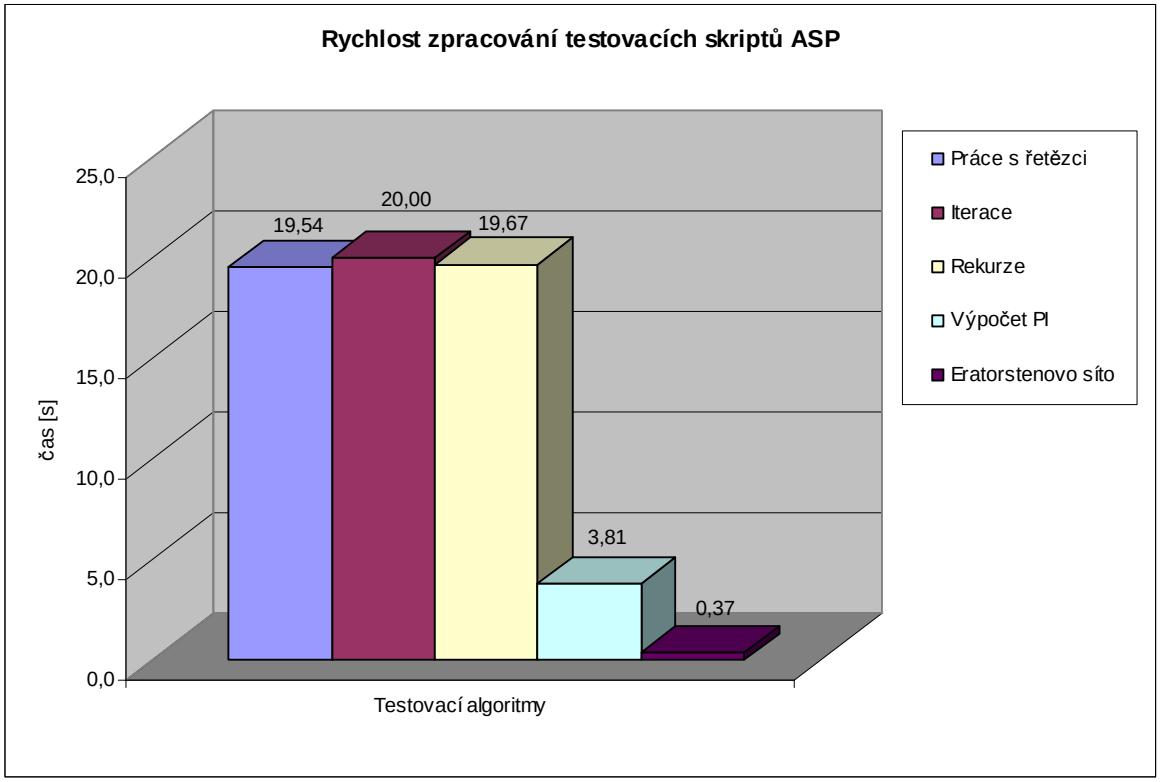
5.4 Výsledky testů

5.4.1 Skripty pro určení výkonu ASP

Tab. 1 – Naměřené hodnoty pro skripty ASP

Měření / čas [s]	Práce s řetězci	Druh algoritmu		Hrubý výkon	
		Iterace	Rekurze	Výpočet π	Eratoste-novo síto
1	19,78125	19,73438	20,71875	3,79688	0,35938
2	19,09375	20,57813	20,51563	3,82813	0,34375
3	19,03125	20,34375	20,92188	3,79687	0,42188
4	18,87500	21,25000	18,93750	3,84375	0,37500
5	20,09375	19,76563	19,12500	3,81250	0,40625

Měření / čas [s]	Práce s řetězci	Druh algoritmu		Hrubý výkon	
		Iterace	Rekurze	Výpočet π	Eratoste-novo síto
6	19,29688	19,70313	19,65625	3,78125	0,42165
7	19,32813	19,64063	19,06250	3,79712	0,35832
8	20,01563	19,89063	19,32813	3,81344	0,37028
9	19,95313	19,48438	19,15625	3,82865	0,34771
10	19,84375	20,37500	19,78125	3,78126	0,32521
Min	18,87500	19,48438	18,93750	3,78125	0,32521
Max	20,09375	21,25000	20,92188	3,84375	0,42188
Odchylka	0,43098	0,52129	0,70377	0,0199	0,03171
Průměr	19,53125	20,07657	19,72031	3,80798	0,37294
Vážený průměr	19,54297	20,00391	19,66797	3,80686	0,37279

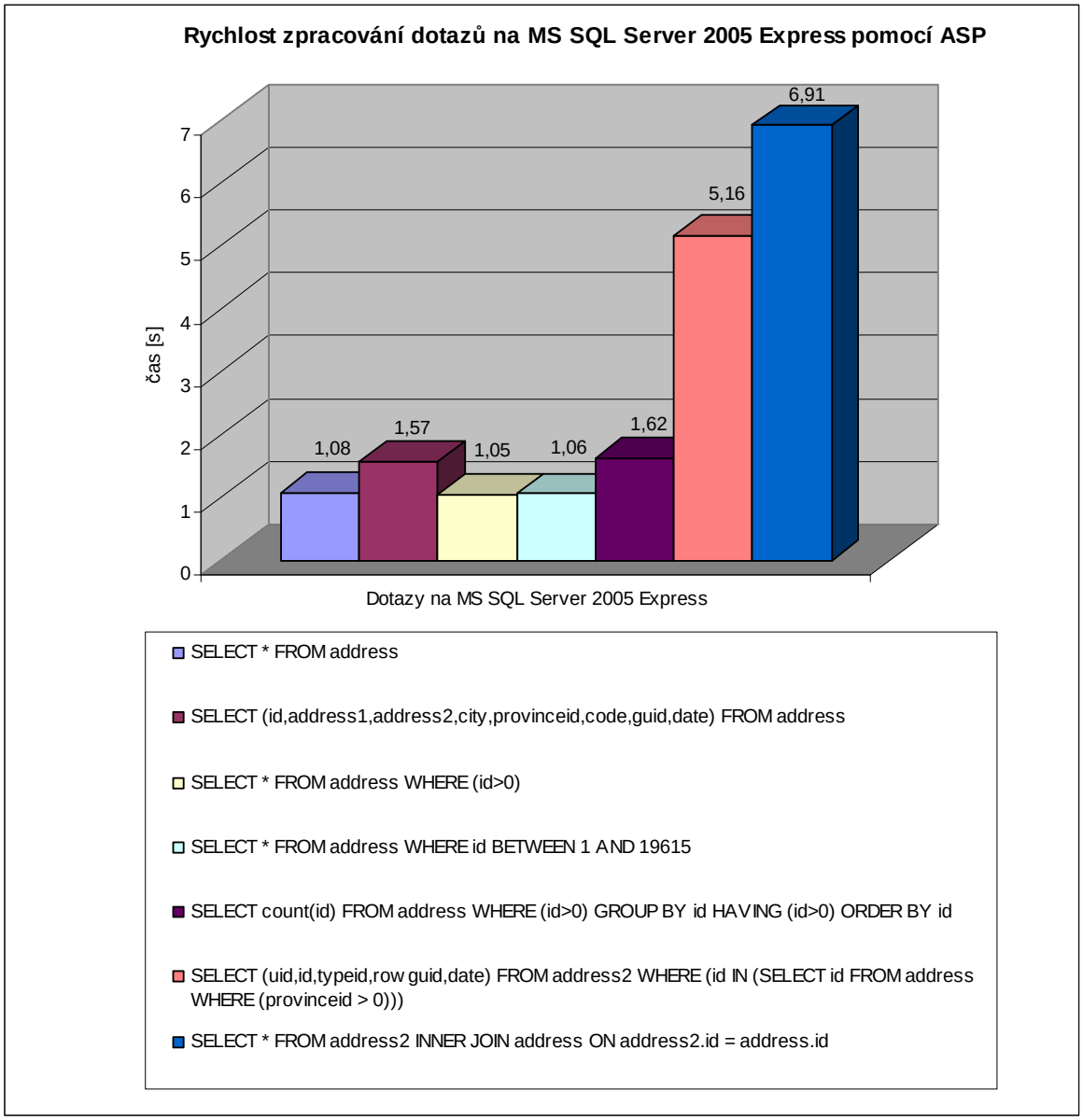


Obr. 4 – Testovací skripty ASP

5.4.2 ASP a databáze Microsoft SQL Server 2005 Express

Tab. 2 – Naměřené hodnoty pro práci ASP s databází Microsoft SQL Server 2005 Express

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,56388	0,54688	0,31250	0,37500	0,42188	2,04688	5,96875
2	1,00000	1,51563	1,14063	1,15625	1,81250	5,23438	7,06250
3	1,03125	1,64063	0,95313	1,18750	1,51563	5,46875	6,90625
4	1,07813	1,59375	1,31250	1,10938	1,71875	5,12500	6,87500
5	1,26563	1,50000	1,12500	1,18750	1,53125	5,34375	6,81250
6	1,17188	1,54688	0,84375	0,78125	1,71875	5,32813	7,07813
7	1,03125	1,64063	1,01563	1,18750	1,62500	4,93750	6,85938
8	1,14063	1,56250	1,17188	1,01563	1,56250	4,87500	6,87500
9	0,95313	1,54688	1,06250	0,89063	1,57813	5,14063	6,85938
10	1,25000	1,84375	1,10938	1,15625	1,71875	5,31250	7,03125
Min	0,54688	0,54688	0,31250	0,37500	0,42188	2,04688	5,96878
Max	1,26563	1,84375	1,31250	1,18750	1,81250	5,46875	7,07813
Odchylka	0,19390	0,32911	0,26034	0,24834	0,37766	0,96088	0,30161
Průměr	1,04688	1,49375	1,00469	1,00469	1,52031	4,88125	6,83281
Vážený průměr	1,082033	1,56836	1,06055	1,06055	1,62109	5,16211	6,91016

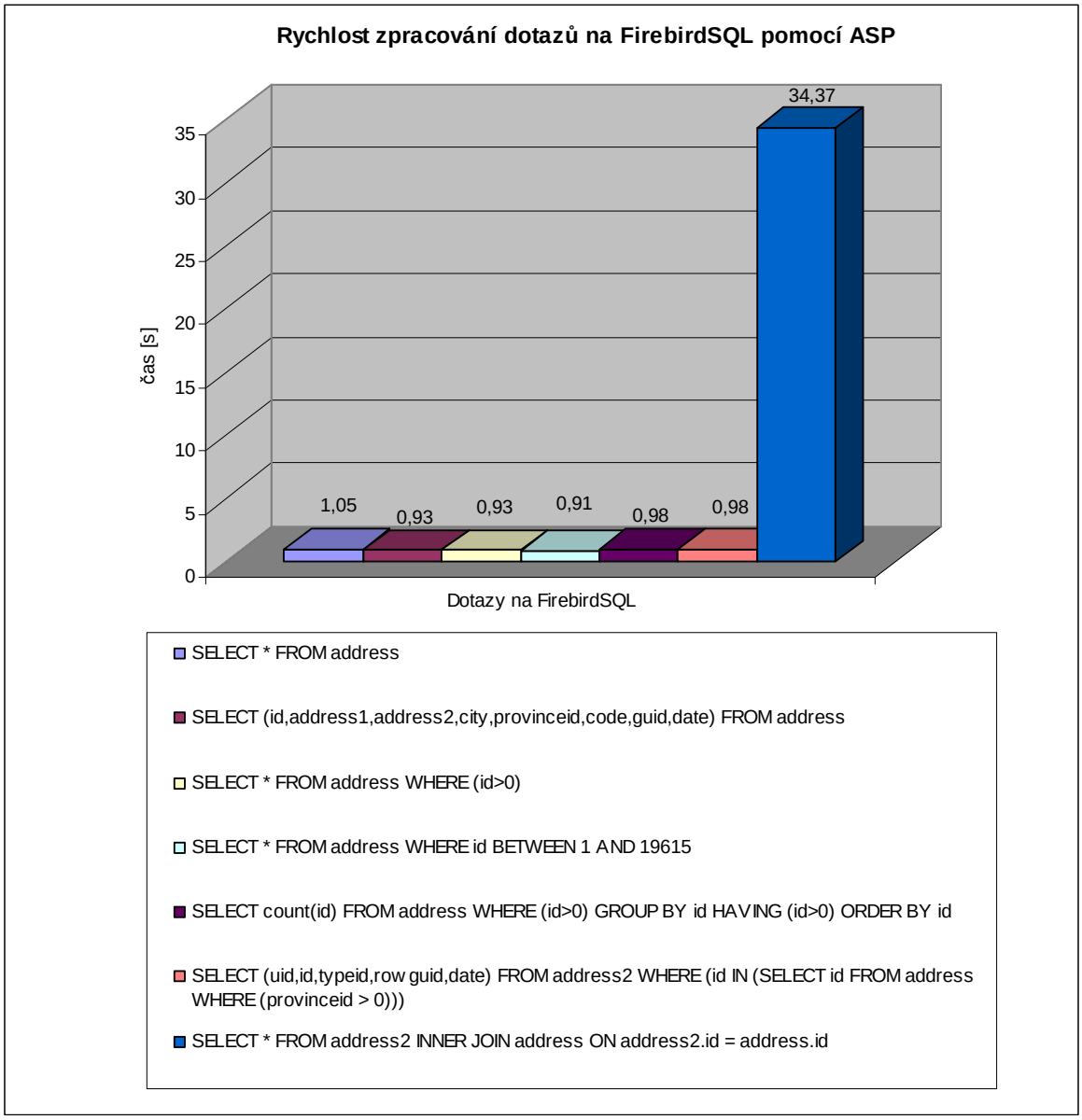


Obr. 5 – ASP a databáze Microsoft SQL Server 2005 Express

5.4.3 ASP a databáze FirebirdSQL

Tab. 3 - Naměřené hodnoty pro práci ASP s databází FirebirdSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,92188	0,92188	0,98438	0,96875	0,98438	1,03125	34,21875
2	0,95313	0,96875	1,15625	1,06250	1,10938	1,01563	35,98435
3	0,96875	0,90625	0,92188	0,93750	1	0,98438	34,01563
4	1,09375	1	0,96875	0,89063	1,12500	0,90625	34,28125
5	1,07813	0,93750	0,93750	0,90625	0,92188	0,96875	34,81250
6	1,62500	0,89063	0,90625	0,89063	0,96875	0,95313	34,21875
7	1,10938	0,92188	0,89063	0,87500	1	0,95313	34,1875
8	1,09375	0,93750	0,89063	0,90625	0,93750	0,98438	34,1875
9	1,03125	0,93750	0,90625	0,9218	0,92188	0,96875	34,15625
10	1,06250	0,90625	0,90625	0,89063	0,93750	0,96875	34,87500
Min	0,92188	0,89063	0,89063	0,87500	0,92188	0,90625	34,01563
Max	1,62500	1	1,15625	1,06250	1,12500	1,03125	35,98435
Odchylka	0,18776	0,03050	0,07597	0,05266	0,06925	0,03281	0,56528
Průměr	1,09375	0,93281	0,94688	0,92500	0,99063	0,97344	34,49375
Vážený průměr	1,04883	0,92969	0,92773	0,91406	0,98242	0,97461	34,36719

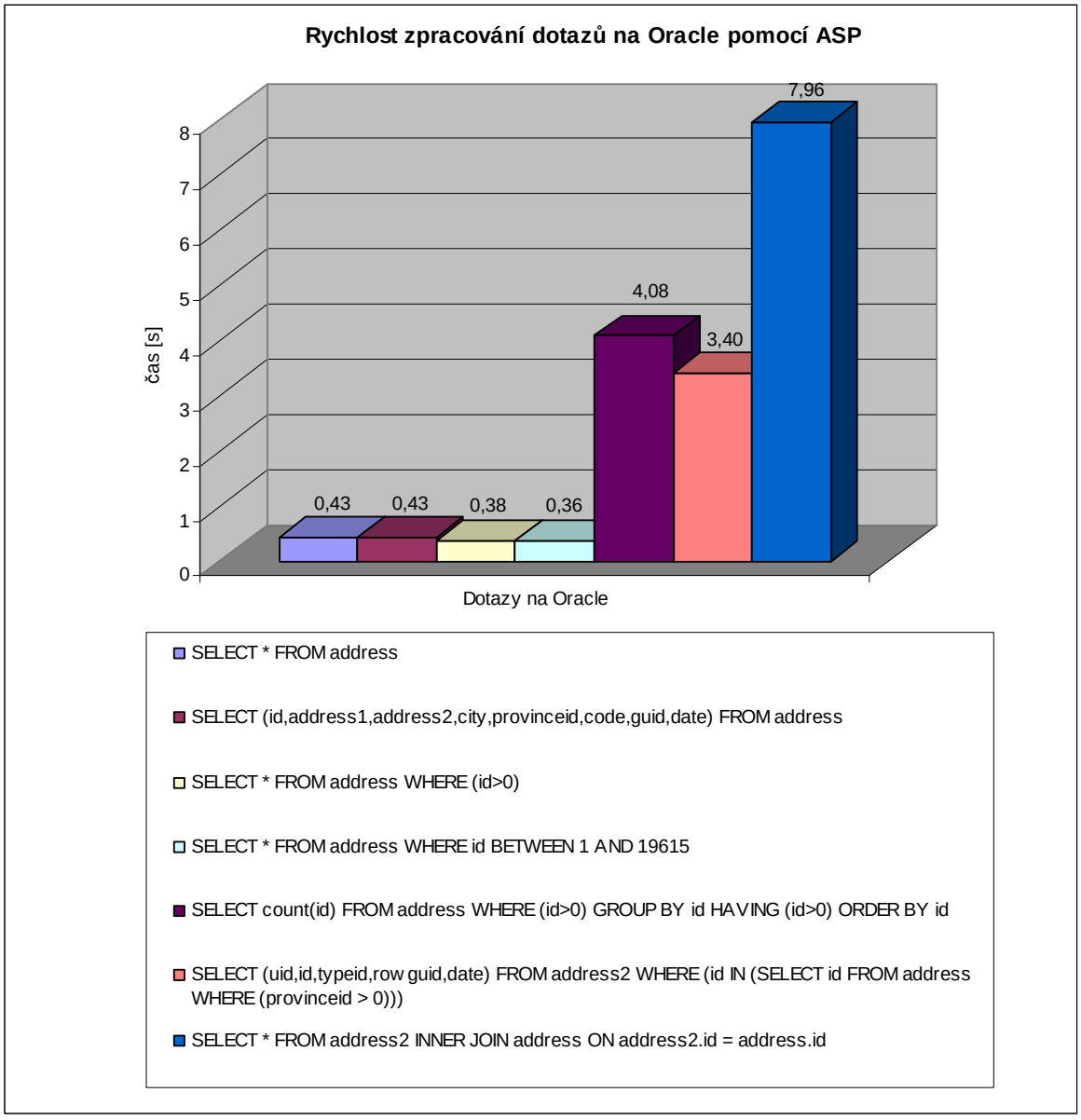


Obr. 6 - ASP a databáze FirebirdSQL

5.4.4 ASP a databáze Oracle

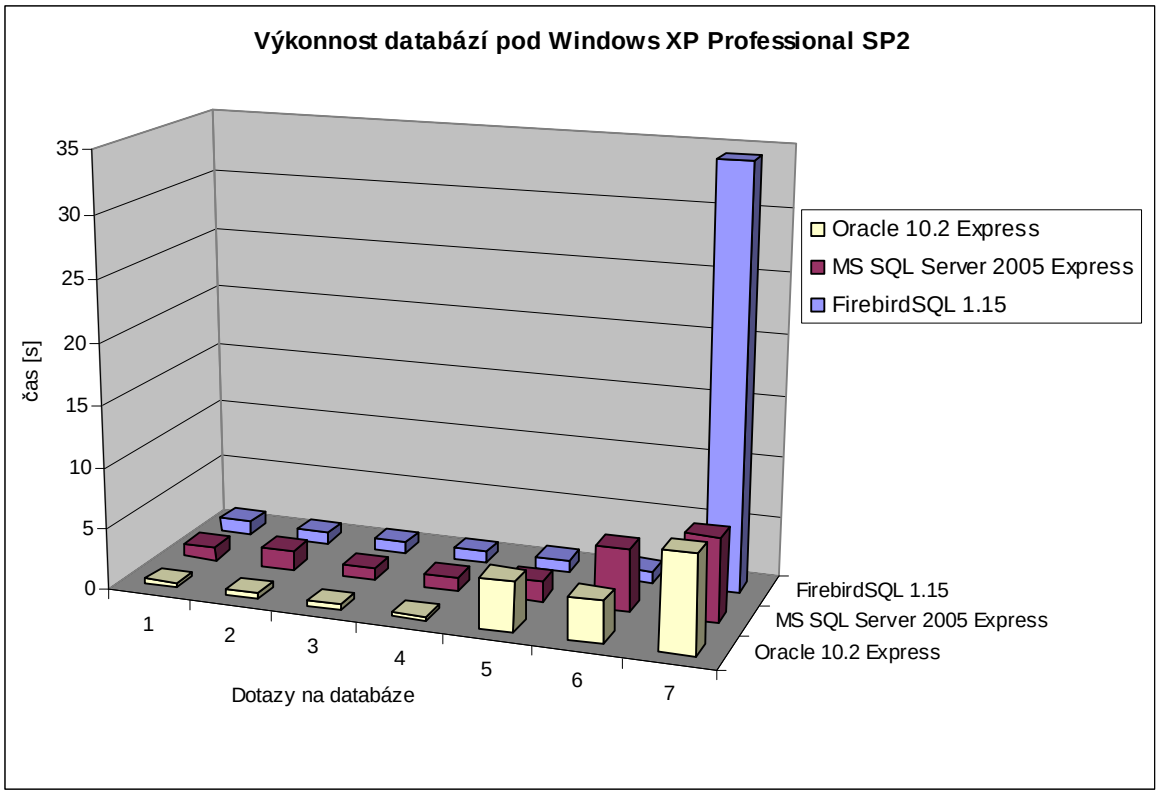
Tab. 4 - Naměřené hodnoty pro práci ASP s databází Oracle

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,76563	0,35938	0,35938	0,34375	4,10938	3,42188	8,10938
2	0,35938	0,46875	0,39063	0,32813	4,15625	3,34375	7,93750
3	0,40625	0,45313	0,39063	0,35938	4,03125	3,42188	7,98438
4	0,39063	0,46875	0,43750	0,34375	4,03125	3,35938	7,81250
5	0,43750	0,45313	0,39063	0,37500	4,39063	3,76563	7,90625
6	0,45313	0,39063	0,35938	0,39063	4,07813	3,39063	7,85938
7	0,45313	0,43750	0,34375	0,34375	4,06250	3,39064	8,07813
8	0,43750	0,42188	0,35938	0,37500	4,10938	3,40625	7,87500
9	0,45313	0,40625	0,34375	0,35938	4,03125	3,43750	8,15625
10	0,40625	0,43750	0,45313	0,37500	4,03125	3,35938	7,89063
Min	0,35938	0,35938	0,34375	0,32813	4,03125	3,34375	7,81250
Max	0,76563	0,46875	0,45313	0,39063	4,39063	3,76563	8,15625
Odchylka	0,10730	0,03369	0,03580	0,01849	0,10416	0,11572	0,11076
Průměr	0,45625	0,42969	0,38281	0,35938	4,10313	3,42969	7,96094
Vážený průměr	0,42969	0,43359	0,37891	0,35938	4,07617	3,39844	7,95508



Obr. 7 - ASP a databáze Oracle

5.4.5 Srovnání výkonu databází u ASP



Obr. 8 - Výkonnost přístupu k databázím pomocí ASP

6 PHP

PHP (což je akronym pro „PHP: Hypertext Preprocesor“) je ve světě oblíbený serverový skriptovací jazyk, který rozšiřuje možnosti HTML stránek. PHP kód je interpretován webovým serverem a na výstupu generuje HTML, ale i jiné formáty jako např. dokumenty PDF, obrazové formáty (JPEG, PNG, …) a další. PHP rovněž spolupracuje s databázemi, a to zejména s MySQL nebo PostgreSQL.

PHP bylo navrženo jediným člověkem, Rasmusem Lerdorfem, v roce 1994 a od té doby je používáno na více než sedmi milionech webových domén na světě, jejichž počet neustále roste.

PHP je Open Source produkt, takže kromě běhového prostředí jsou přístupné i zdrojové soubory, které je možné libovolně upravovat.

6.1 Podrobný pohled na PHP

PHP je velmi vyhledávanou skriptovací technologií. Nejčastěji je nasazeno na serverech s operačními systémy Unix a Linux. Tyto operační systémy jsou poměrně nenáročné na hardware serveru a i jeden takovýto server dokáže obsloužit miliony požadavků za den. Navíc je PHP, stejně jako spousta operačních systémů Linux zdarma, takže ceny web-hostingových služeb jsou u této skriptovací technologie velmi nízké, příp. jsou poskytovány i zdarma. Nic samozřejmě nebrání používat PHP na webových serverech pod operačním systémem Windows. PHP nativně podporuje přístup ke spoustě databází, kde kromě nejpoužívanějšího MySQL, je možné se připojit k PostgreSQL, mSQL, Oracle, Hyperwave, Informix, InterBase, Sybase a dalším. Samozřejmě je možné také použít ODBC (Open Database Connectivity Standard) k přístupu k databázím, k nimž zprostředkuje přístup ODBC ovladač.

Při svém vzniku si PHP vypůjčilo vlastnosti a syntaxi z několika různých programovacích jazyků jako jsou jazyk C, C++, Java, Perl apod. Umožňuje také vytváření objektů a tím i objektové programování webového obsahu.

6.2 Kompilátory PHP

Kromě ASP.NET, které je kompilované, a které již takto bylo od základu vytvořeno, jsou ASP i PHP interpretované skriptovací technologie. To znamená, že takovýto skript je pro-

váděn serverem postupně, řádek po řádku, přímo za běhu. Interpretované skripty se však nejenom nesnadněji odlaďují, ale hlavně jsou mnohokrát pomalejší než kompilované. Proto se již před časem začaly objevovat první, a většinou komerční, kompilátory pro jednu z nejoblíbenějších skriptovacích technologií, pro PHP. Jsou to např. dřívější Zend Encoder, z něhož se stal v současnosti komplexní nástroj pod názvem Zend SafeGuard, Turck MMCache, PHP Accelerator a další.

Zní to sice jako nesmysl, kompilovat interpretované skripty a provádět je, ale jde to, a na Internetu je možné nalézt webová sídla využívající kompilovaného PHP. Proto i tato práce nemůže opomenout na porovnání výkonu zkompilovaných skriptů PHP. Pro tyto potřeby je použit starší Zend Encoder 3.6. Aby takto upravené skripty byly serverem proveditelné, je potřeba rozšířit PHP o běhové prostředí, které je v tomto případě Zend Optimizer 3.0.0.

6.3 Prostředí pro provádění testů

Provádění skriptů PHP je možné na více webových serverech pod různými operačními systémy. Nejčastěji je však PHP provozováno na webovém serveru Apache, pod operačními systémy Linux, méně častěji pak i na operačním systému Windows. V testech jsou zahrnuty obě systémové platformy, Mandriva Linux 2006 (Kernel optimalizován pro procesory 686) a Microsoft Windows XP Professional SP2, s webovým serverem Apache 2.0.54. Použité databáze jsou MySQL 5.1.17, PostgreSQL 8.1.2, FirebirdSQL 1.5.3 a Oracle 10.2 Express. Porovnán bude také výkon kompilovaného PHP pomocí Zend Encoderu 3.6 a běhového prostředí Zend Optimizer 3.0.0. Verze PHP pro všechny testy je 5.1.2.

6.4 Testovací skripty

Dále je uveden kompletní přepis testovacích algoritmů v úpravě pro PHP.

6.4.1 Práce s řetězci

Prog. 7 – PHP: Pomocí lokální proměnné

```
$s = "";
for ($i = 1 ; $i <= 100000 ; $i++)
{
   $s =  $s."a";
}
```

Prog. 8 – PHP: Pomocí globální proměnné

```
$GLOBALS['s'] = "";
for ($i = 1 ; $i <= 100000 ; $i++)
{
   $GLOBALS['s'] = $GLOBALS['s']."a";
}
```

6.4.2 Druh algoritmu

Prog. 9 – PHP: Iterace

```
function iterate($str)
{
   for ($i = 1 ; $i <= strlen($str) ; $i++)
   {
      $s = $s.substr($str,-$i,1);
   }
   return;
}
for ($i = 1 ; $i <= 100 ; $i++) $str .= "a";
for ($x = 1 ; $x <= 1000 ; $x++) iterate($str);
```

Prog. 10 – PHP: Rekurze

```
function recursion($str)
{
   if (strlen($str) > 0) recursion(substr($str,1));
    $s = $s.substr($str,0,1);
    return;
}
for ($i = 1 ; $i <= 100 ; $i++) $str .= "a";
for ($x = 1;$x <= 1000  ;$x++) recursion($str);
```

6.4.3 Hrubý výkon

Prog. 11 – PHP: Výpočet Ludolfova čísla

```
$pi_const = 3.141592;
$pi = 0;
$i = 1;
do
{
   $pi = $pi + (4/$i - 4/($i+2));
   $i = $i + 4;
}while ($pi<=$pi_const);
```

Prog. 12 – PHP: Eratostenovo síto

```
function sito($intval)
{
   $cisla = array();
   $limit = 2;
   while($limit*$limit < $intval)$limit++;
   for ($i = 2 ; $i <= $limit ; $i++)
   {
      if ($cisla[$i] == false)
      {
         for($k = $i+$i ; $k <= $intval ; $k+=$i)$cisla[$k] = true;
      }
   }
   $pocet = 0;
   for ($i = 2 ; $i <= $intval ; $i++)
   {
      if ($cisla[$i] == false)$pocet++;
   }
}
sito(100000);
```

6.4.4 Práce s databázemi

Skripty pro připojení a spuštění dotazů na databázích MySQL, PostgreSQL, FirebirdSQL i Oracle jsou téměř totožné. Budou proto uveřejněny pouze skripty pro práci s databází MySQL.

U ostatních databází jsou odlišné pouze názvy některých funkcí, pro připojení k databázi a dotazování. Tyto změny jsou většinou pouze v prefixu funkcí, např. *mysql_pconnect* pro MySQL a *pg_pconnect* pro PostgreSQL, apod. Další informace naleznete v [7] PHP Manuálu.

Prog. 13 – PHP: Práce s databází MySQL

```
function db_connect()
{
   $result = mysql_pconnect('localhost', 'root', '');
   if (!$result) return false;
   if (!mysql_select_db('test')) return false;
   return $result;
}
function test1()
{
   $query = "SELECT * FROM address";
   $result = mysql_query($query);
}
function test2()
{
   $query = "SELECT (id,address1,address2,city,provinceid,
          code,guid,date) FROM address";
   $result = mysql_query($query);
}
function test3()
{
   $query = "SELECT * FROM address WHERE (id>0)";
   $result = mysql_query($query);
}
```

Pokračování

Prog. 13 - Pokračování

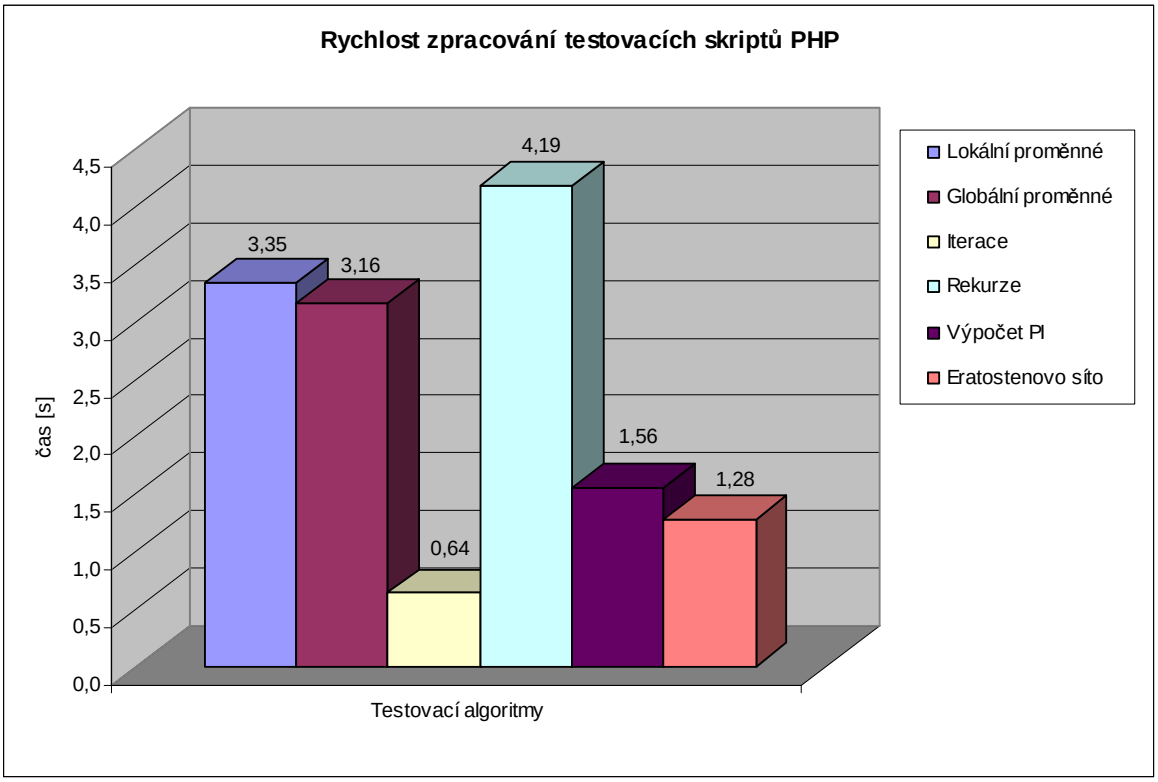
```
 function test4()
 {
    $query = "SELECT * FROM address WHERE id BETWEEN 1 AND 19615";
    $result = mysql_query($query);
 }
 function test5()
 {
    $query = "SELECT count(id) FROM address WHERE (id>0)
            GROUP BY id HAVING (id>0) ORDER BY id";
    $result = mysql_query($query);
  }
 function test6()
 {
    $query = "SELECT (uid,id,typeid,rowguid,date) FROM address2
            WHERE (id IN (SELECT id FROM address WHERE
           (provinceid > 0)))";
    $result = mysql_query($query);
 }
function test7()
{
   $query = "SELECT * FROM address2 INNER JOIN address ON
           address2.id = address.id";
   $result = mysql_query($query);
}

db_connect();


for($x = 1 ; $x <= 100 ; $x++) test1();
for($x = 1 ; $x <= 100 ; $x++) test2();
for($x = 1 ; $x <= 100 ; $x++) test3();
for($x = 1 ; $x <= 100 ; $x++) test4();
for($x = 1 ; $x <= 100 ; $x++) test5();
for($x = 1 ; $x <= 100 ; $x++) test6();
for($x = 1 ; $x <= 100 ; $x++) test7();
```

6.5 Výsledky testů na operačním systému Mandriva Linux 2006

6.5.1 Skripty pro určení výkonu PHP

Tab. 5 – Naměřené hodnoty pro skripty PHP

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
1	3,89469	3,08342	0,35855	3,45151	1,46145	1,43019
2	3,11710	3,13885	0,80685	6,15657	1,36459	1,18718
3	3,30062	3,22721	0,61471	6,82988	1,39151	1,50861
4	3,50985	3,14611	0,64984	4,07504	1,52693	1,16753
5	3,34712	3,26055	0,77701	4,10070	1,73417	1,22088
6	3,34288	3,12766	0,49640	3,68657	1,52811	1,21564
7	3,21556	3,08955	0,52345	3,58946	1,59667	1,22133
8	3,36828	3,25504	0,68443	4,40886	1,82526	1,48861
9	3,33788	3,21963	0,66713	3,63982	1,50439	1,05026
10	3,34673	3,06122	0,72783	3,83908	1,69909	1,31259
Min	3,11710	3,06122	0,35855	3,45151	1,36459	1,05026
Max	3,89469	3,26055	0,80685	6,82988	1,82526	1,50861
Odchylka	0,19739	0,07034	0,13050	1,10174	0,14239	0,14301
Průměr	3,37807	3,16092	0,63062	4,37775	1,56322	1,28028
Vážený průměr	3,34612	3,16093	0,64260	4,18701	1,55529	1,28049

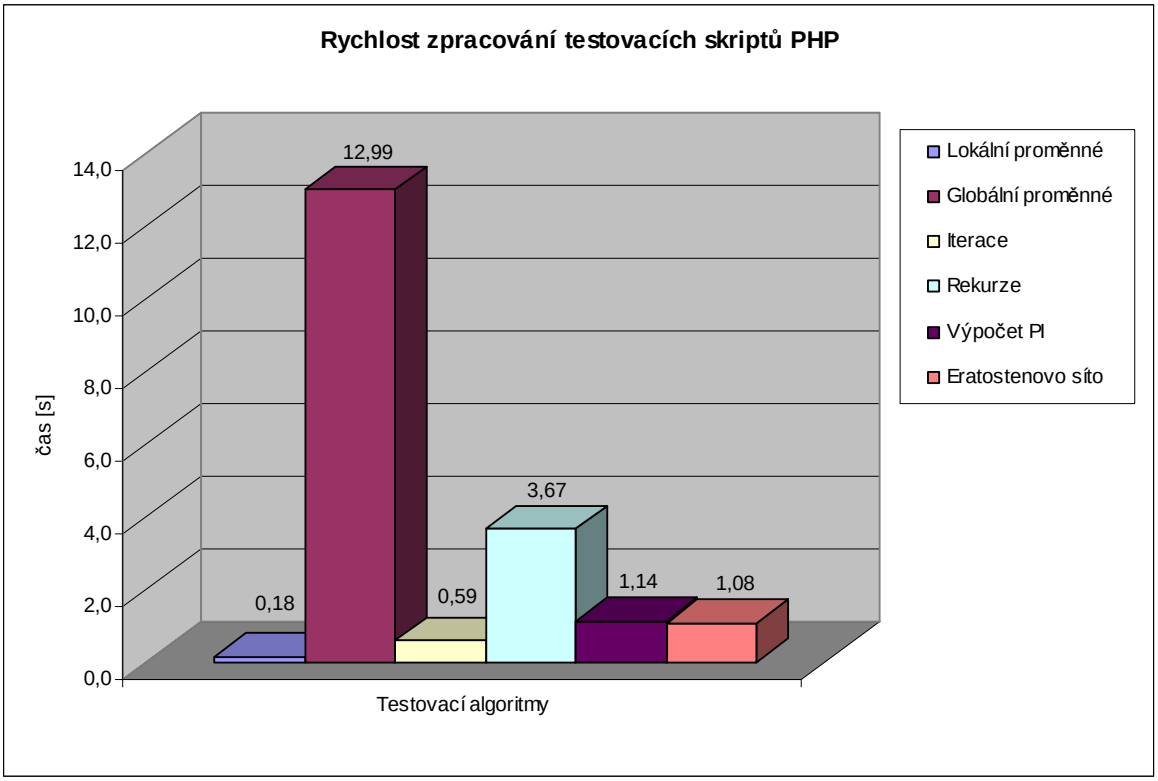


Obr. 9 – Testovací skripty PHP

Tab. 6 – Naměřené hodnoty pro kompilované skripty PHP

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
1	0,13265	11,29229	0,31462	3,56352	0,97488	1,41550
2	0,10177	12,05609	0,37645	3,49217	1,15529	0,90147
3	0,16093	11,48465	0,98738	4,34783	1,58319	1,39479
4	0,28346	12,86645	0,57550	3,72569	1,09748	0,99860
5	0,25922	14,45892	0,85764	3,56307	1,07840	0,92677
6	0,06687	13,06083	0,60337	4,16660	1,37425	1,18454
7	0,45445	13,64365	0,73946	3,44457	0,97266	0,89325
8	0,15599	13,71438	0,75051	3,73749	1,32609	0,87767

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
9	0,13305	13,83672	0,46375	3,32234	1,08334	1,20483
10	0,16954	13,26795	0,38069	3,69944	1,03735	1,11474
Min	0,06687	11,29229	0,31462	3,32234	0,97266	0,87767
Max	0,45445	14,45692	0,98738	4,34783	1,58319	1,41550
Odchylka	0,10739	0,99699	0,21377	0,30455	0,18782	0,19388
Průměr	0,19179	12,96819	0,60494	3,70627	1,16829	1,09122
Vážený průměr	0,17458	12,99134	0,59342	3,67407	1,14089	1,07737

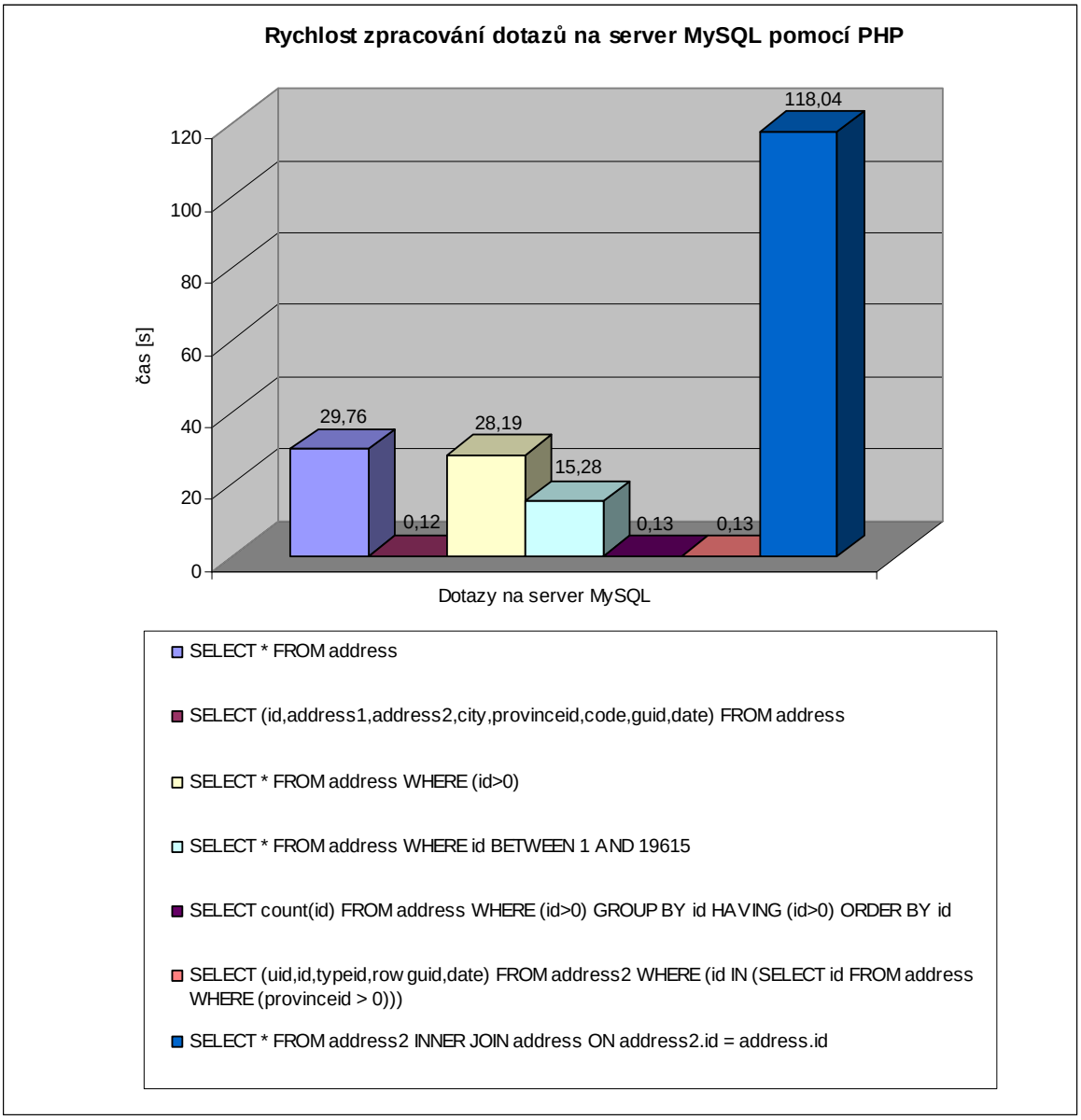


Obr. 10 – Testovací kompilované skripty PHP

6.5.2 PHP a databáze MySQL

Tab. 7 - Naměřené hodnoty pro práci PHP s databází MySQL

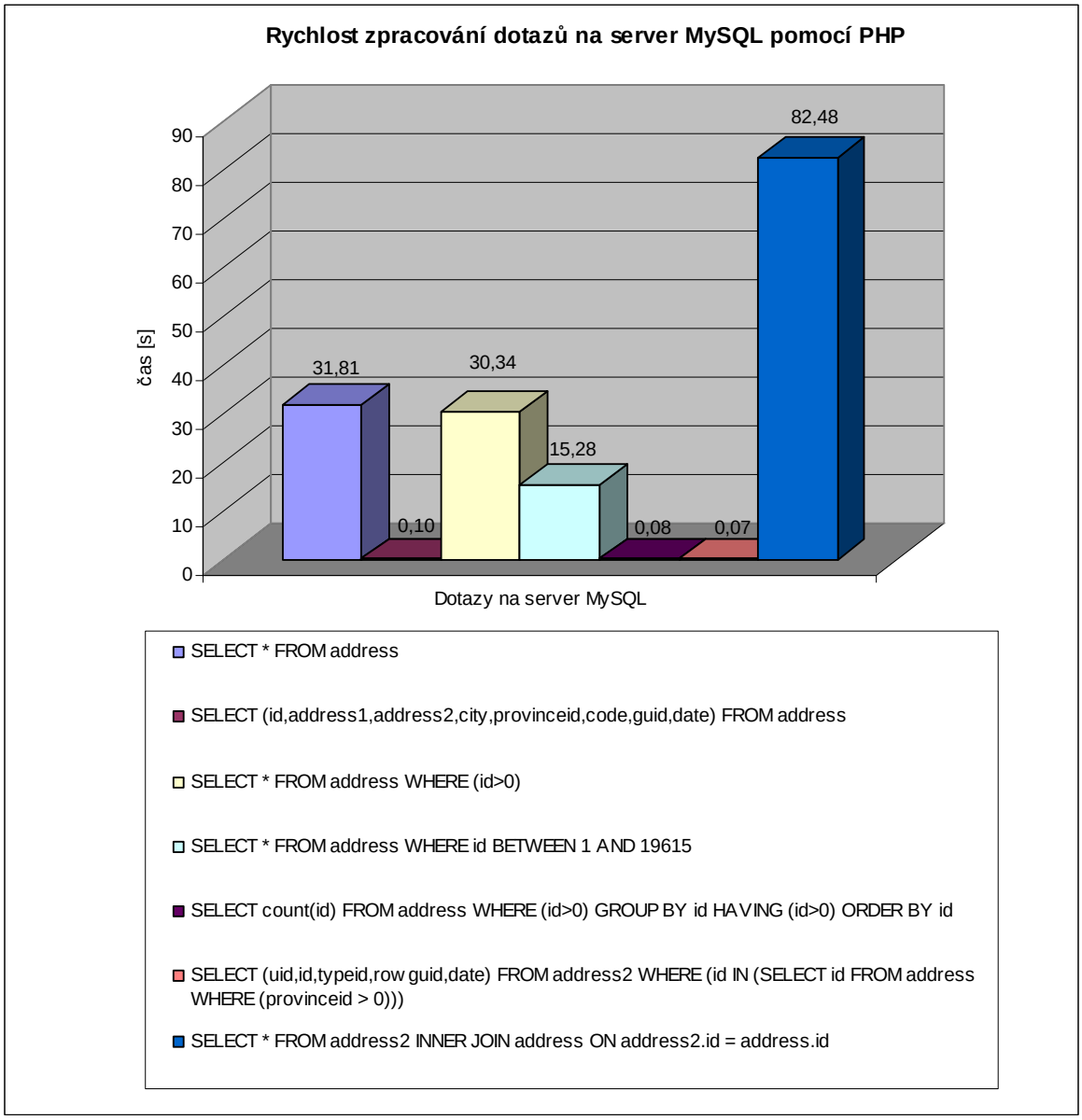
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	46,02991	0,16351	29,07549	15,64776	0,14726	0,04122	87,05673
2	34,72783	0,10201	34,88988	17,68639	0,11505	0,13189	106,06183
3	23,76274	0,13016	23,94348	13,42178	0,13959	0,12742	143,63578
4	39,57595	0,11371	39,36379	21,09221	0,12249	0,14262	101,56220
5	37,02385	0,10262	37,10794	19,17698	0,10828	0,12489	96,80917
6	24,89621	0,14001	23,80063	12,34444	0,13093	0,13658	151,14926
7	24,53431	0,12278	23,58738	13,11673	0,16119	0,12238	134,14737
8	26,08517	0,11221	25,61615	15,02251	0,13291	0,13040	133,15814
9	26,13412	0,10877	26,06945	14,64831	0,12542	0,15989	139,51429
10	25,08821	0,11798	25,04809	13,54254	0,14019	0,15802	89,46236
Min	23,76274	0,10201	23,58738	12,34444	0,10828	0,04122	87,05673
Max	46,02991	0,16351	39,36379	21,09221	0,16119	0,15989	151,14926
Odchylka	7,50594	0,018005	5,70246	2,72558	0,01481	0,03130	23,11372
Průměr	30,78583	0,12138	28,85013	15,56997	0,13233	0,12753	118,25571
Vážený průměr	29,75821	0,11853	28,19376	15,28288	0,13173	0,13428	118,04389



Obr. 11 - PHP a databáze MySQL

Tab. 8 - Naměřené hodnoty pro práci kompilovaného PHP s databází MySQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	26,39871	0,21374	25,65077	14,07591	0,21130	0,21696	109,60322
2	26,87896	0,11709	25,55028	15,00884	0,16125	0,13472	143,67934
3	33,71856	0,09129	33,05195	16,76507	0,03222	0,03165	75,88809
4	32,31182	0,08793	28,21489	13,98315	0,03258	0,03131	76,40468
5	35,29348	0,09239	31,72904	16,50979	0,10696	0,10704	79,55181
6	32,39905	0,09729	28,88791	13,74537	0,09223	0,03176	74,75880
7	32,29735	0,09005	32,35178	16,65873	0,11280	0,10900	90,24730
8	32,36921	0,09304	32,02638	15,71707	0,03341	0,03272	76,27731
9	32,33212	0,09104	32,08061	16,56399	0,10347	0,11357	76,70721
10	32,16324	0,09800	31,79451	13,72552	0,03258	0,03254	75,17965
Min	26,39871	0,08793	25,55028	13,72252	0,03222	0,03131	74,75880
Max	35,29348	0,21374	33,05195	16,76507	0,21130	0,21696	143,67934
Odchylka	2,65604	0,03638	2,69166	1,24237	0,05821	0,05984	21,28100
Průměr	31,61625	0,10719	30,13381	15,27534	0,09188	0,08413	87,82974
Vážený průměr	31,80879	0,09627	30,34198	15,28289	0,08441	0,07413	82,48241

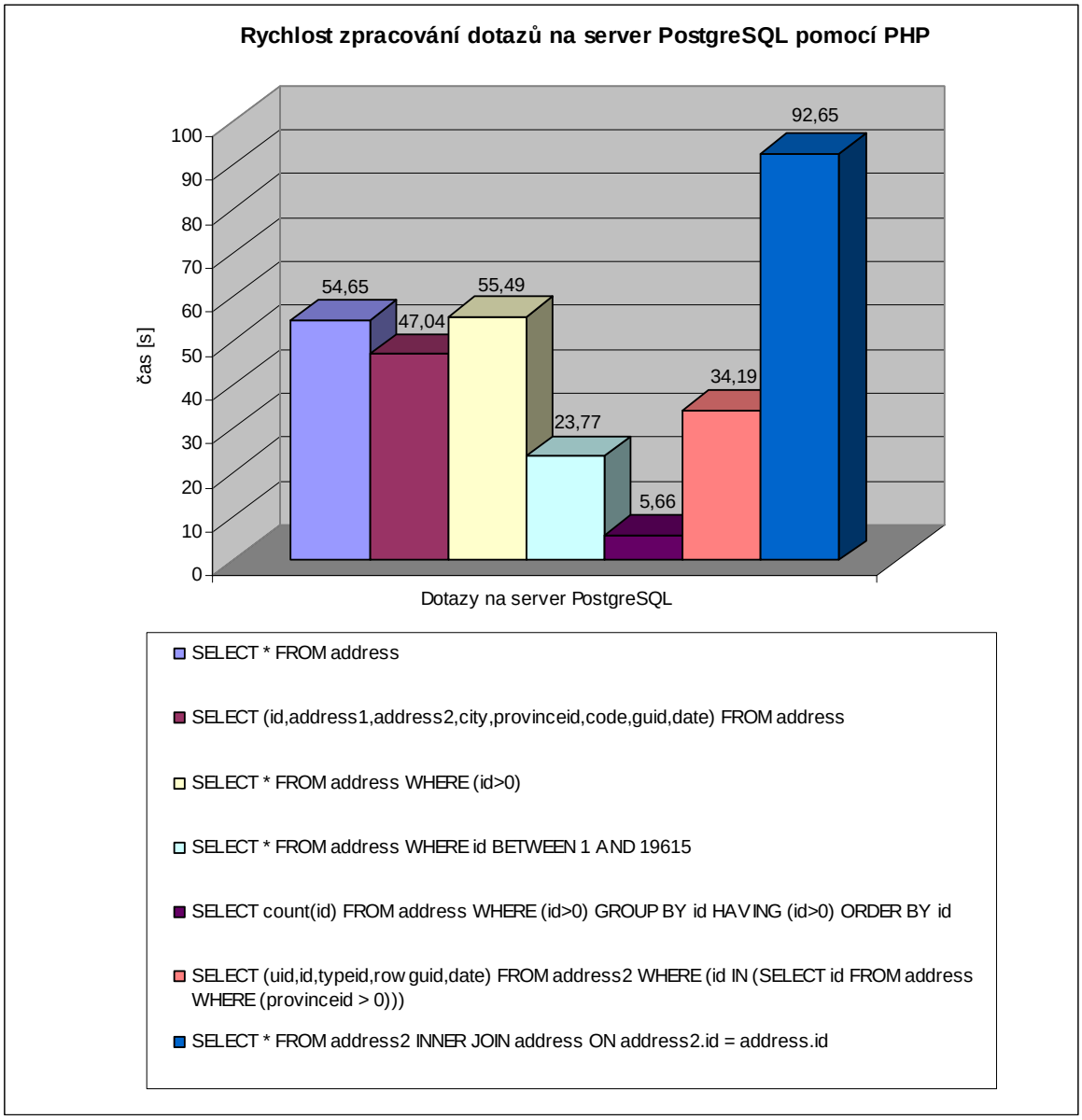


Obr. 12 - Kompilované PHP a databáze MySQL

6.5.3 PHP a databáze PostgreSQL

Tab. 9 - Naměřené hodnoty pro práci PHP s databází PostgreSQL

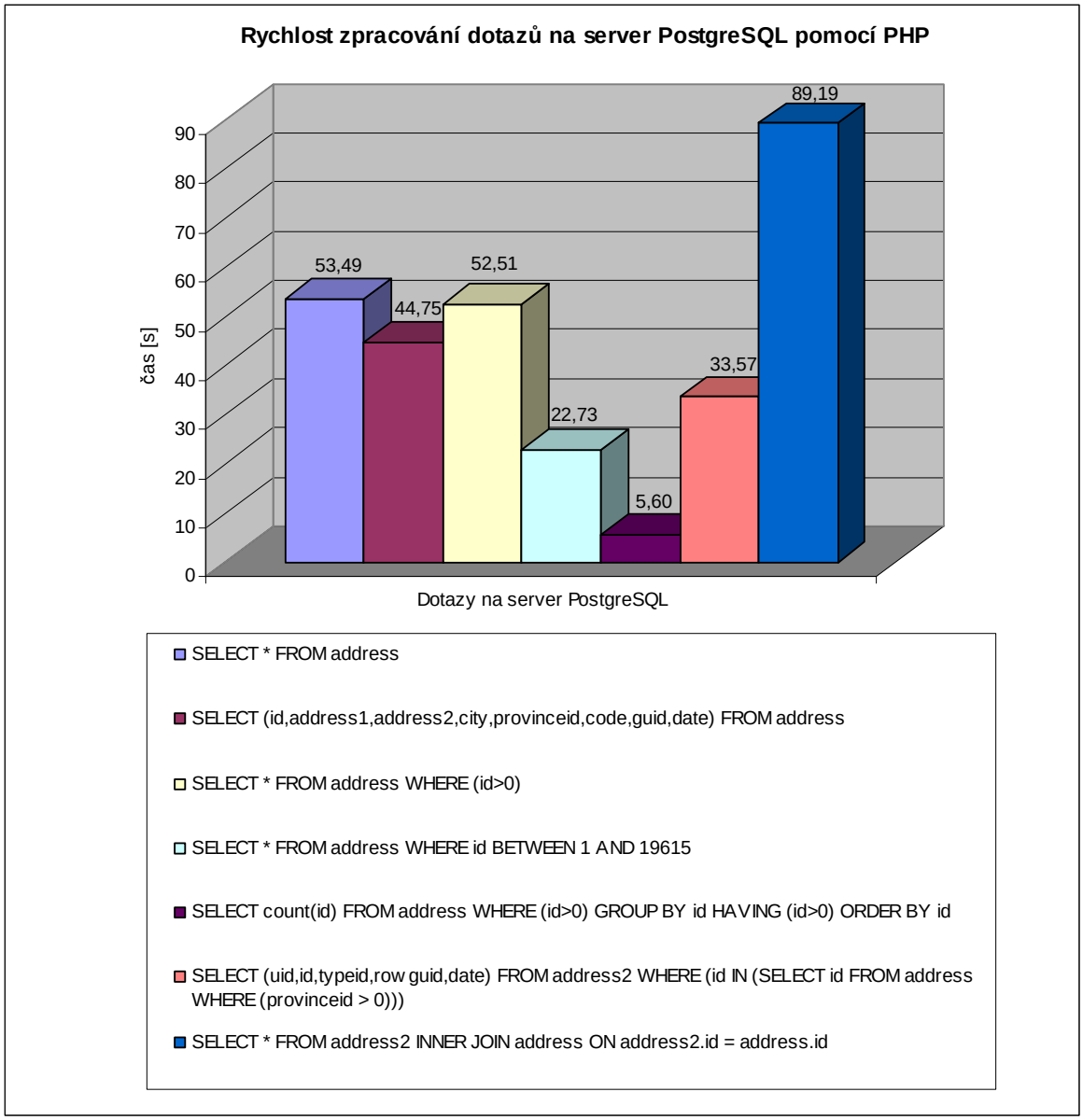
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	65,90323	56,20967	66,57859	28,79680	7,02730	43,98637	115,85197
2	55,34478	47,17249	54,32821	23,47961	5,76695	34,82283	96,19275
3	55,51380	46,09581	54,44384	21,72444	5,33811	33,90326	92,36620
4	55,27778	48,42106	55,70214	24,39814	5,90251	35,90663	95,19987
5	55,94397	48,26591	56,56261	23,48096	5,51812	31,97961	92,92815
6	55,69607	47,43062	55,68270	24,66663	5,81315	34,44760	92,27909
7	52,09114	49,13556	61,60501	27,07452	6,06892	34,89976	91,19098
8	53,69107	43,84307	51,24904	21,88259	5,31243	33,11292	89,86487
9	54,64738	45,91434	54,27813	23,48322	5,50680	34,47211	91,14789
10	50,99756	43,91920	51,30526	21,41415	5,33238	31,38440	88,46375
Min	50,99756	43,84307	51,24904	21,41415	5,31243	31,38440	88,46375
Max	65,90323	56,20947	66,57859	28,79680	7,02730	43,98637	115,85197
Odchylka	3,80731	3,32511	4,43081	2,24063	0,49076	3,30218	7,41497
Průměr	55,41068	47,64077	56,17355	24,04011	5,75867	34,89155	94,54855
Vážený průměr	54,65075	47,04437	55,48849	23,77376	5,65587	34,19309	92,64623



Obr. 13 - PHP a databáze PostgreSQL

Tab. 10 – Naměřené hodnoty pro práci kompilovaného PHP s databází PostgreSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	54,46879	36,45700	45,25272	18,54746	4,39434	26,65708	73,88272
2	53,41343	45,26863	53,62479	23,17787	5,69441	34,91566	90,20229
3	53,54256	44,95059	53,59852	23,15684	5,65942	34,07051	90,74476
4	53,83063	45,19965	53,46478	23,14795	5,58025	33,87919	88,05630
5	52,88896	44,45771	53,09051	22,82037	5,64817	34,11093	90,73937
6	52,92319	44,77824	45,64796	20,49700	5,25336	29,42413	82,54019
7	53,29049	44,96722	52,99899	22,85416	5,62249	33,88973	89,82180
8	53,67492	44,7929	53,81935	23,21941	5,64531	34,63487	91,05781
9	53,45612	44,15721	53,99870	23,05211	5,68456	34,44895	90,98845
10	53,78911	44,70104	53,85664	23,13513	5,78559	34,11227	90,40106
Min	52,88896	34,45700	45,25272	18,54746	4,39434	26,65708	73,88272
Max	54,46879	45,26863	53,99870	23,21941	5,78559	34,91566	91,05781
Odchylka	0,43767	2,52461	3,25749	1,49015	0,39071	2,58150	5,25470
Průměr	53,52782	43,97298	51,93530	22,36083	5,49679	33,01433	87,84347
Vážený průměr	53,49006	44,75052	52,51269	22,73018	5,59850	33,57132	89,18677

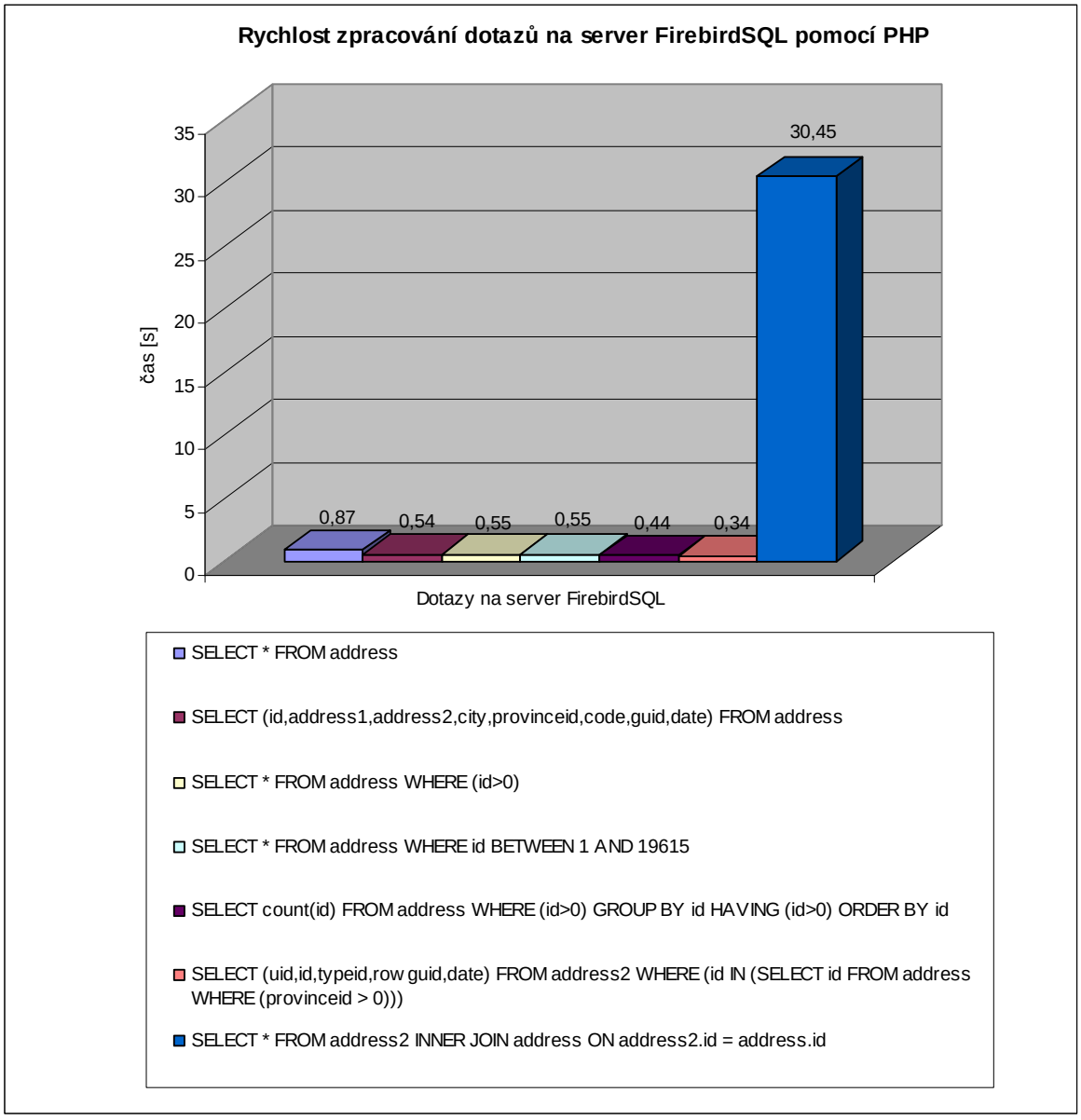


Obr. 14 - Kompilované PHP a databáze PostgreSQL

6.5.4 PHP a databáze FirebirdSQL

Tab. 11 - Naměřené hodnoty pro práci PHP s databází FirebirdSQL

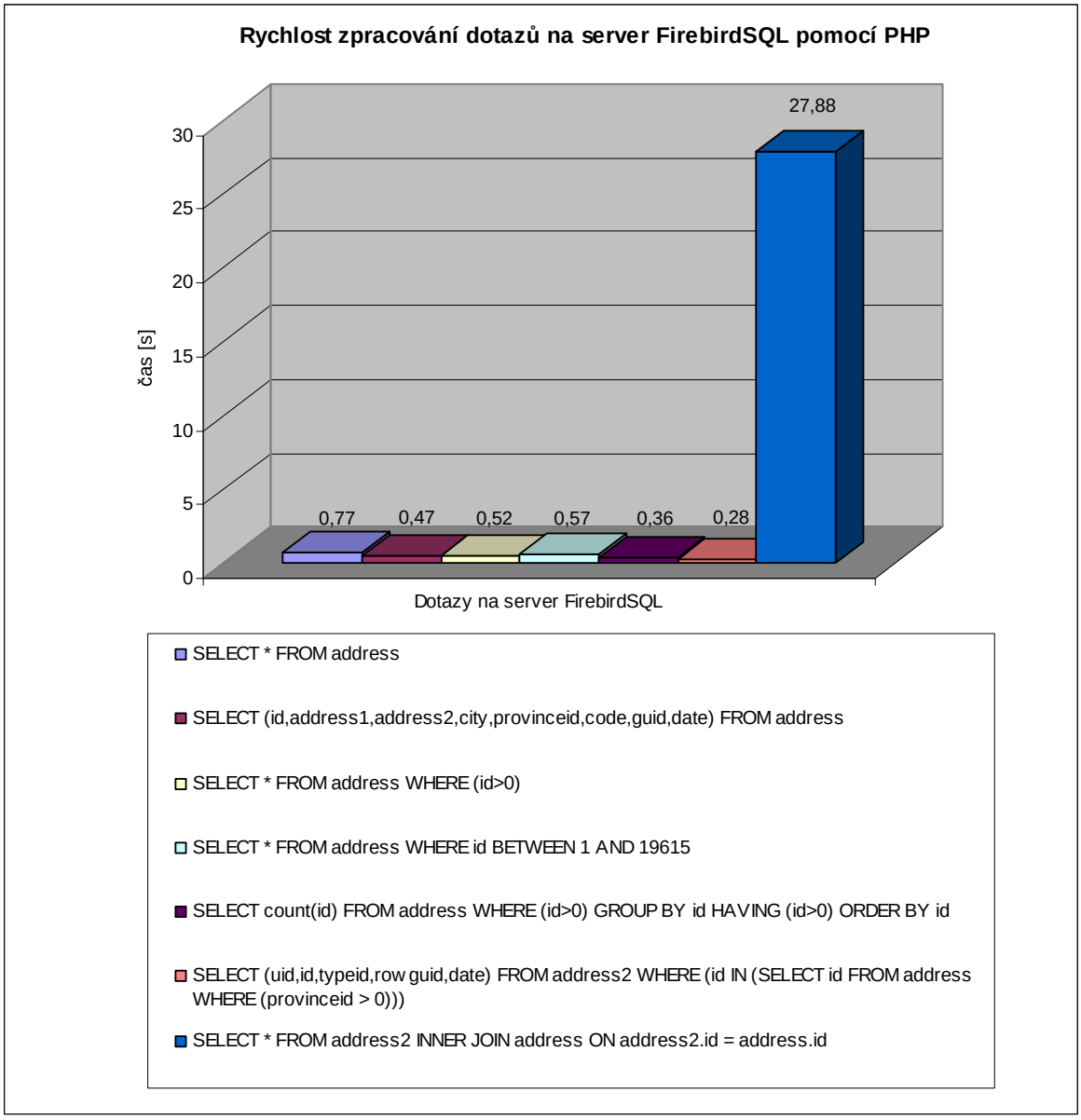
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1,00157	0,65578	0,59554	0,56081	0,47441	0,39481	33,74401
2	0,89440	0,51224	0,55788	0,53409	0,42015	0,34489	29,67841
3	0,92449	0,52390	0,54551	0,53044	0,43486	0,34906	30,40089
4	0,85041	0,53881	0,54110	0,55871	0,43898	0,32489	30,47100
5	0,84020	0,53115	0,55710	0,55189	0,43794	0,31489	30,12354
6	0,86785	0,54781	0,54118	0,55111	0,43105	0,32748	31,00089
7	0,85490	0,56879	0,54778	0,54965	0,44489	0,33401	30,04849
8	0,85323	0,51081	0,55088	0,54349	0,43049	0,33004	30,41220
9	0,83191	0,53788	0,55129	0,55048	0,43489	0,34464	30,11896
10	0,86880	0,53800	0,53745	0,54711	0,43490	0,34690	30,98776
Min	0,83191	0,51081	0,53745	0,53044	0,42015	0,31489	29,67841
Max	1,00157	0,65578	0,59554	0,56081	0,47441	0,39481	33,74401
Odchylka	0,04831	0,03980	0,01569	0,00914	0,01351	0,02078	1,08637
Průměr	0,87877	0,54652	0,55257	0,54778	0,43826	0,34116	30,69861
Vážený průměr	0,86928	0,53732	0,54909	0,54832	0,43600	0,33774	30,44547



Obr. 15 - PHP a databáze FirebirdSQL

Tab. 12 – Naměřené hodnoty pro práci kompilovaného PHP s databází FirebirdSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,81232	0,51109	0,53049	0,55405	0,39556	0,32048	30,07909
2	0,75409	0,49805	0,50420	0,54008	0,38105	0,28405	28,48089
3	0,79019	0,48480	0,51149	0,58085	0,35012	0,27189	28,18909
4	0,78100	0,48132	0,50988	0,55800	0,35049	0,27101	26,07000
5	0,78519	0,48648	0,52871	0,55308	0,36988	0,26105	27,13128
6	0,74569	0,47106	0,52084	0,57708	0,36132	0,27912	27,11338
7	0,76778	0,47322	0,51891	0,55481	0,35779	0,28519	28,80080
8	0,75131	0,48979	0,50591	0,55662	0,35119	0,27445	28,44649
9	0,75492	0,49008	0,51447	0,58611	0,35785	0,25485	27,10088
10	0,74105	0,48653	0,51905	0,58191	0,30908	0,27192	27,78090
Min	0,74105	0,47106	0,50420	0,54008	0,30908	0,25485	26,07000
Max	0,81232	0,51109	0,53049	0,58611	0,39556	0,32048	30,07909
Odchylka	0,02188	0,01096	0,00844	0,01493	0,02157	0,01688	1,07178
Průměr	0,76835	0,48724	0,51639	0,56426	0,35843	0,27740	27,91928
Vážený průměr	0,76627	0,48628	0,51616	0,56455	0,35996	0,27483	27,88046

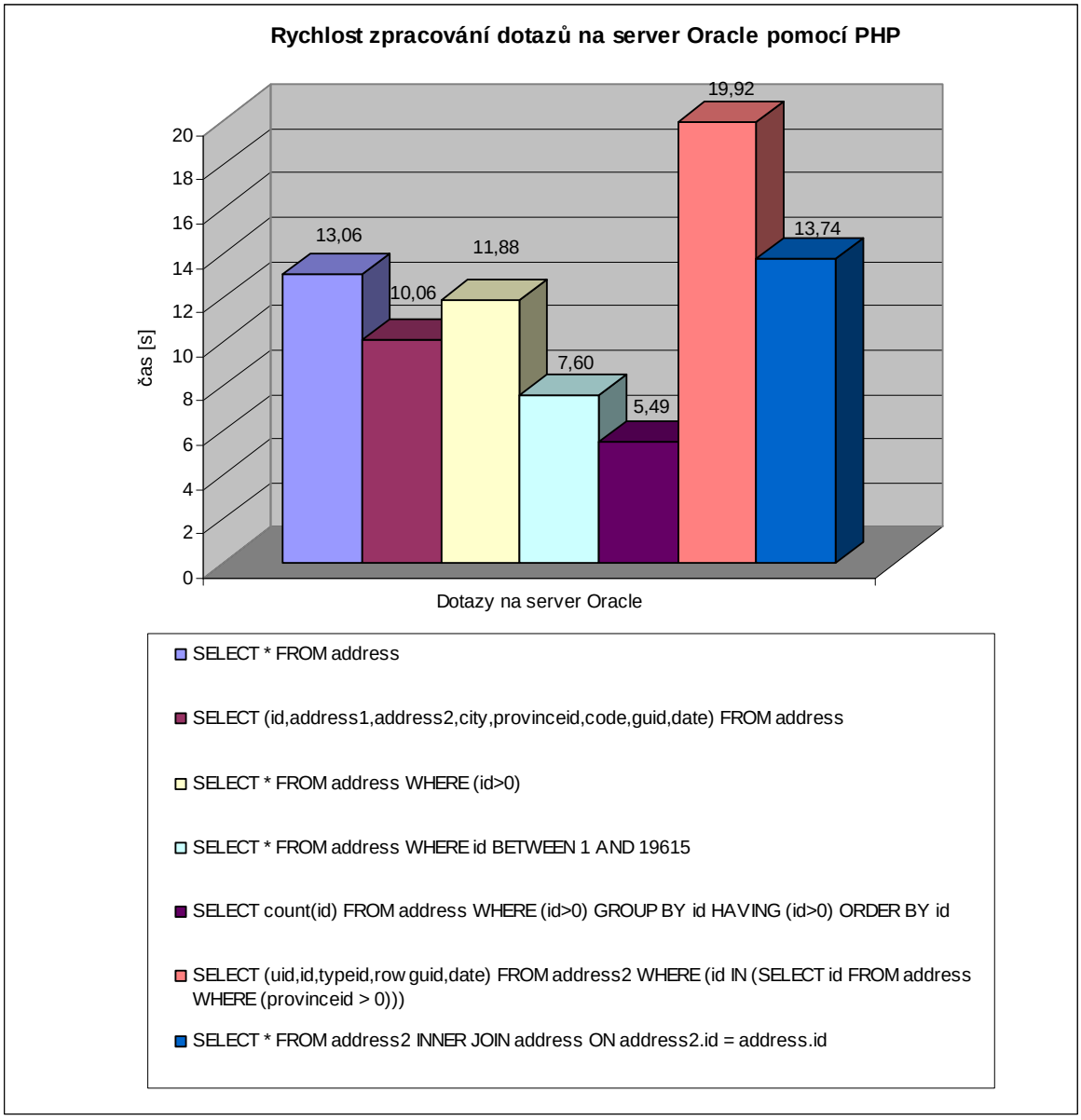


Obr. 16 - Kompilované PHP a databáze FirebirdSQL

6.5.5 PHP a databáze Oracle

Tab. 13 – Naměřené hodnoty pro práci PHP s databází Oracle

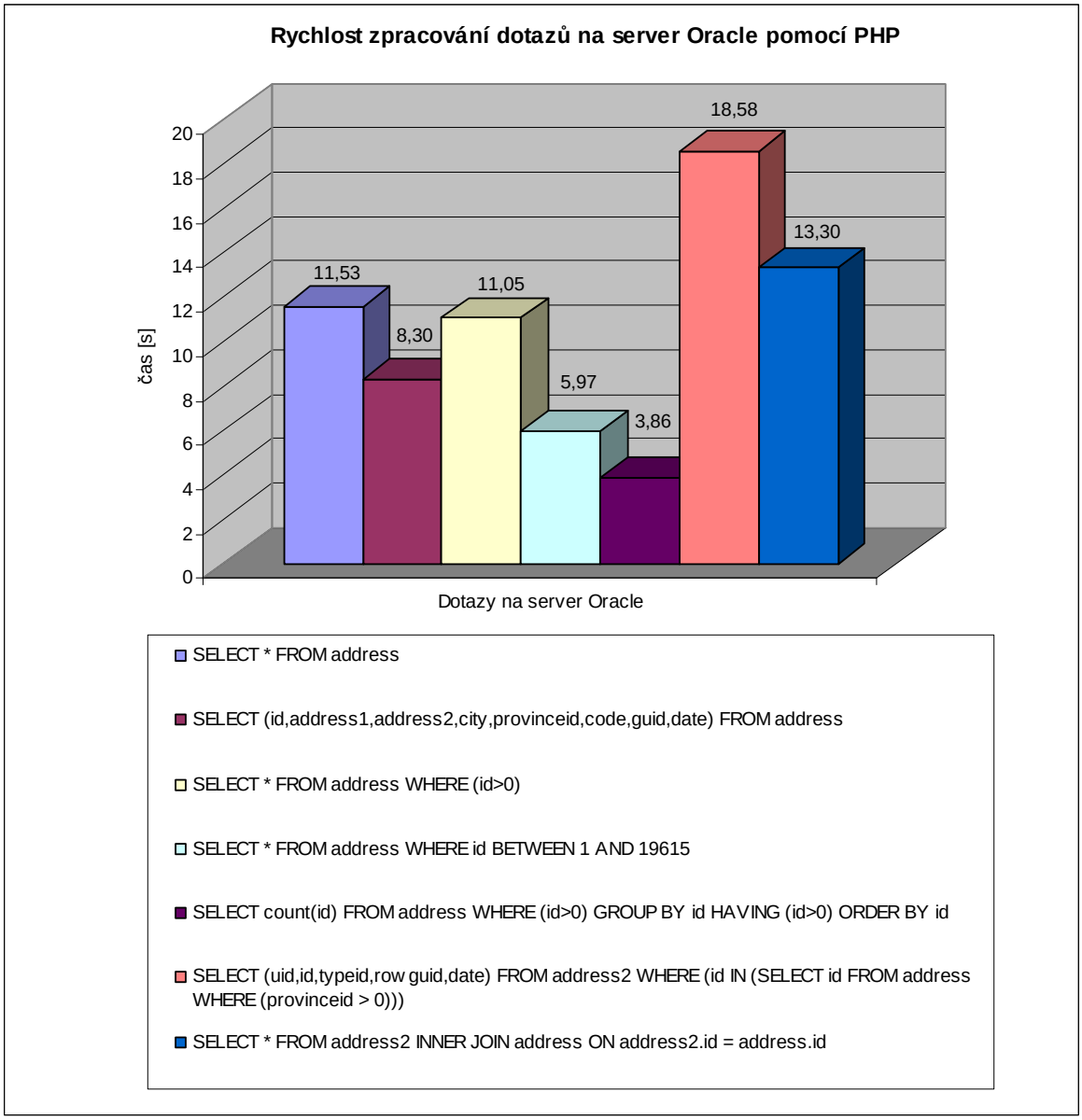
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	13,08489	11,77738	12,10849	7,74446	5,00159	20,81788	13,81789
2	12,87849	9,70046	12,78978	6,10123	5,47118	18,44418	12,90709
3	12,80481	10,40480	12,04049	8,54569	5,44167	23,41988	13,11315
4	13,11662	9,66878	10,49409	8,13878	5,78945	20,00127	15,71790
5	14,01085	9,74002	12,99888	7,58977	4,41790	19,49879	15,14566
6	12,08489	8,64712	11,04890	7,15490	5,11896	19,66124	13,62942
7	12,66775	9,11116	11,14011	8,51411	5,41782	19,71789	13,66119
8	12,48409	10,78419	11,48410	6,14481	5,33212	20,11377	12,17105
9	13,40895	10,98016	11,66549	7,71220	6,31312	20,89145	12,07891
10	13,99888	10,09880	12,78979	7,80489	6,99880	18,66516	15,44988
Min	12,08489	8,64712	10,49409	6,10123	4,41790	18,44418	12,07891
Max	14,01085	11,77738	12,99888	8,54569	6,99880	23,41988	15,71790
Odchylka	0,58696	0,87820	0,79632	0,81410	0,67719	1,32923	1,23025
Průměr	13,05402	10,09128	11,85601	7,54508	5,53026	20,12315	13,76921
Vážený průměr	13,05556	10,06104	11,88339	7,60049	5,48574	19,92093	13,73692



Obr. 17 - PHP a databáze Oracle

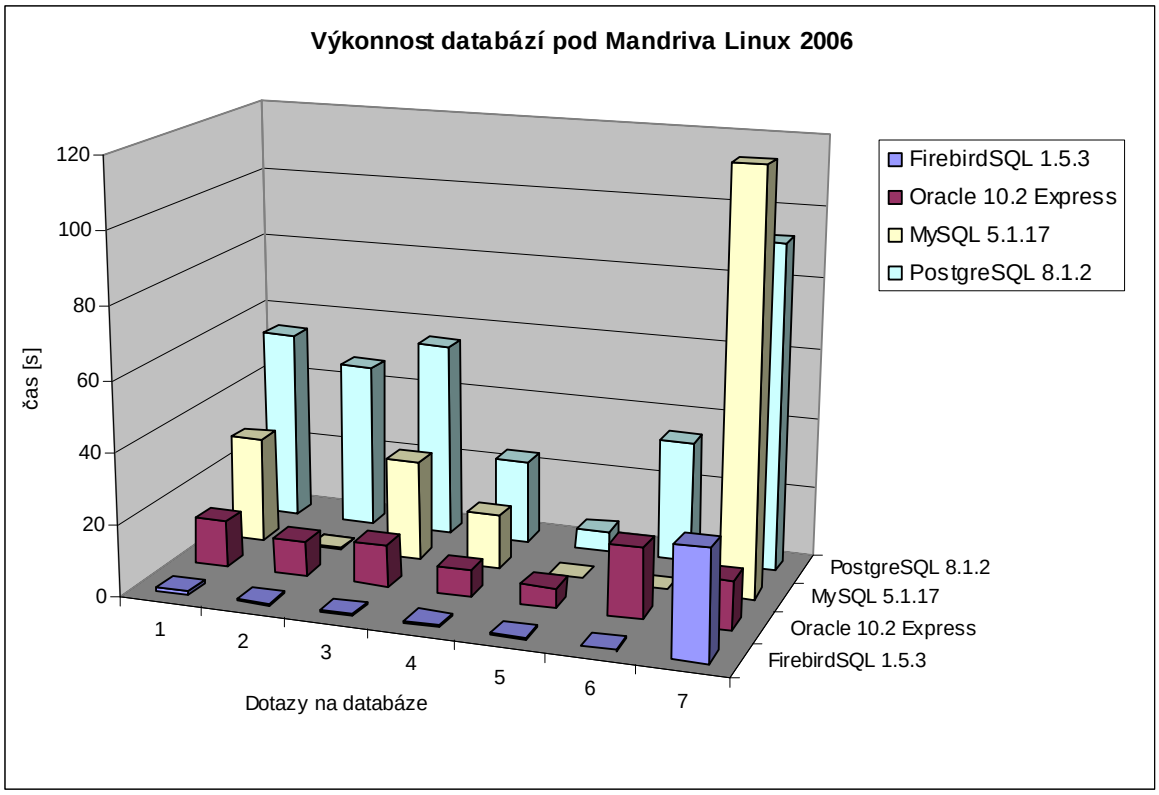
Tab. 14 – Naměřené hodnoty pro práci kompilovaného PHP s databází Oracle

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	12,07084	10,98749	10,64478	5,04895	4,04895	18,12159	12,87810
2	11,45018	10,87811	9,04878	6,70089	3,17891	15,14815	15,08799
3	11,18747	8,11595	11,78919	5,41342	3,97899	19,49805	16,76449
4	11,97905	7,65489	11,16645	5,16549	3,31565	18,87896	11,94490
5	9,44066	7,81898	12,10490	6,01898	4,44319	17,04089	12,48049
6	10,62880	8,10126	12,00789	6,11748	6,48918	18,84199	12,00988
7	12,09871	6,97898	10,41198	8,04895	4,12338	18,54185	13,07489
8	12,78915	6,95119	10,51949	7,41048	3,15942	19,71091	13,77898
9	10,00789	7,17897	10,80485	5,11316	3,11988	18,00985	13,64890
10	13,11138	9,64114	11,04489	5,78495	4,66449	19,98402	13,44984
Min	9,44066	6,95119	9,04878	5,04895	3,11988	15,14815	11,94490
Max	13,11138	10,98749	12,10490	8,04895	6,48918	19,98402	16,76449
Odchylka	1,11773	1,45049	0,86119	0,96970	0,97203	1,36105	1,39733
Průměr	11,47641	8,43070	10,95432	6,08228	4,05220	18,37763	13,51185
Vážený průměr	11,52651	8,29604	11,04869	5,96561	3,86412	18,58051	13,30113

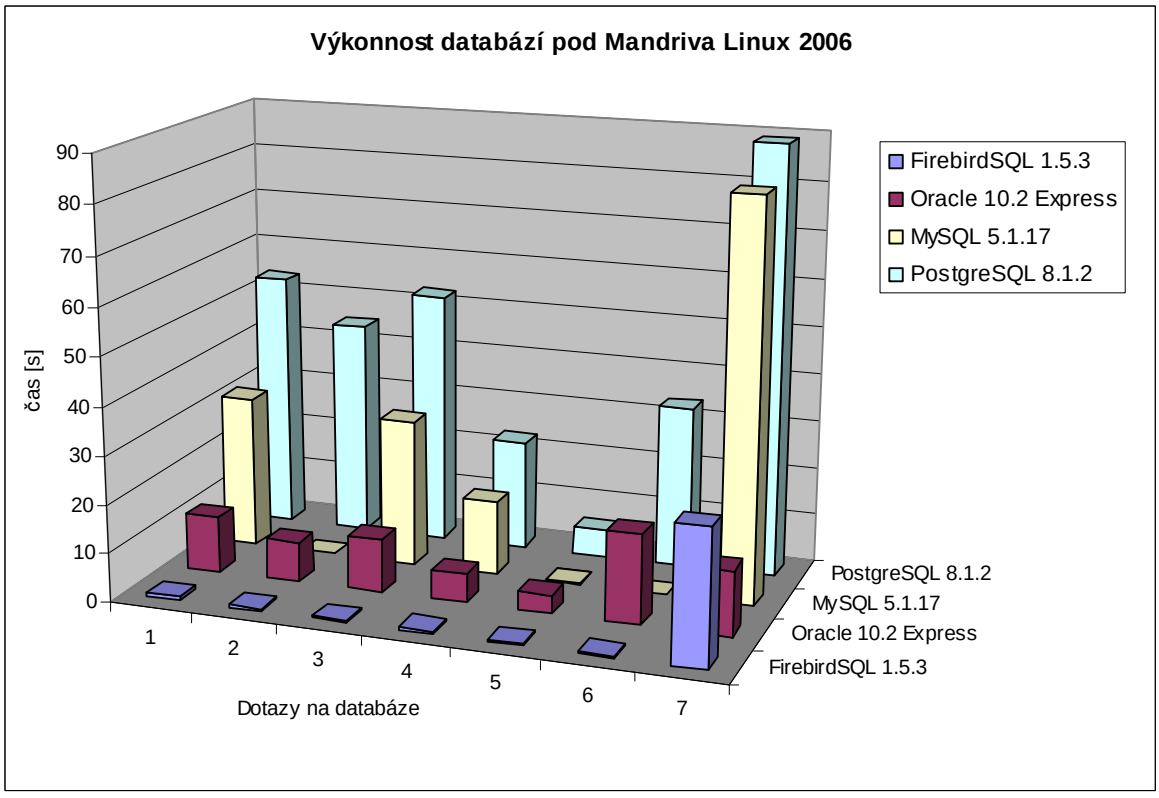


Obr. 18 - Kompilované PHP a databáze Oracle

6.5.6 Srovnání výkonu databází u PHP pod operačním systémem Linux



Obr. 19 - Výkonnost přístupu k databázím pomocí PHP pod Linuxem



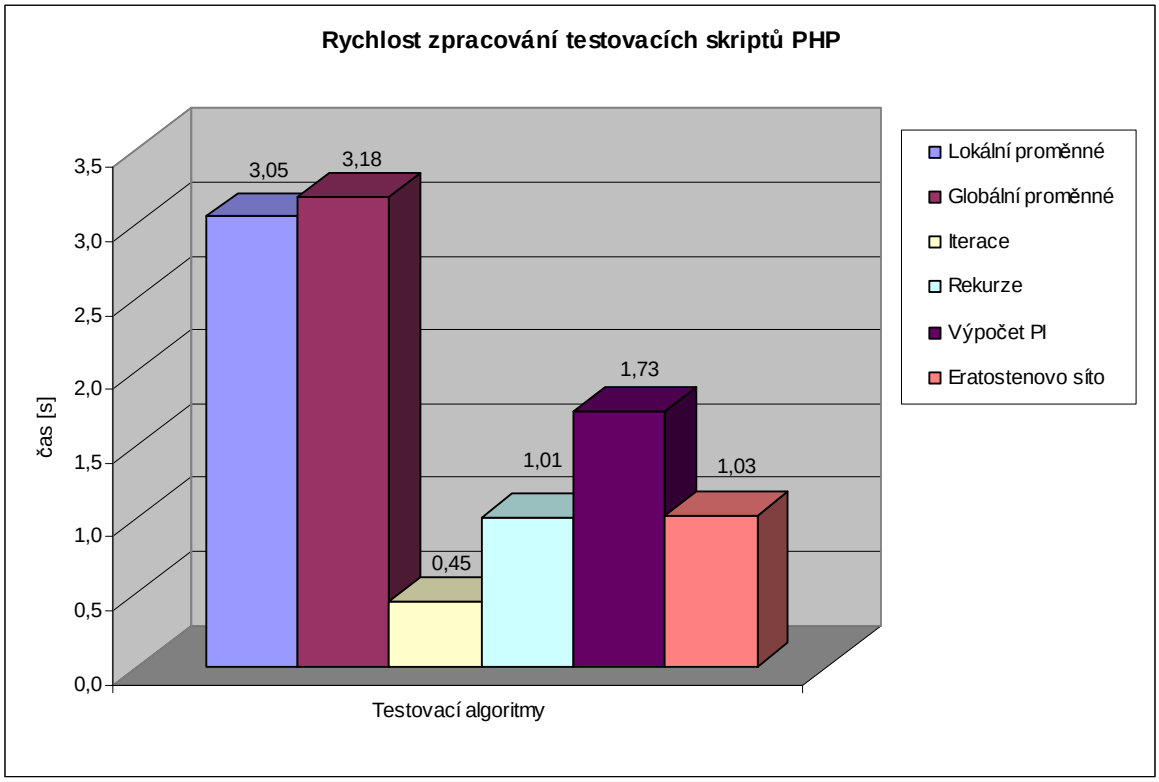
Obr. 20 - Výkonnost přístupu k databázím pomocí kompilovaného PHP pod Linuxem

6.6 Výsledky testů PHP na operačním systému Microsoft Windows XP Professional SP2

6.6.1 Skripty pro určení výkonu PHP

Tab. 15 – Naměřené hodnoty pro skripty PHP

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratos-tenovo síto
1	3,05951	3,02999	0,44863	0,99091	1,72608	0,97096
2	2,87383	3,28260	0,42515	1,04160	1,69899	0,99357
3	2,95941	3,15072	0,47379	0,99591	1,77955	1,07131
4	3,00244	3,13285	0,40808	0,97793	1,77109	1,04299
5	2,99683	3,22296	0,42847	0,97805	1,71275	1,04925
6	3,40842	3,57056	0,48339	1,02554	1,69171	1,00265
7	2,93326	3,07502	0,43689	0,95225	1,71181	0,97383
8	3,11557	3,08073	0,47703	1,05211	1,76197	1,06397
9	3,35415	3,45382	0,40992	1,01393	1,77124	1,06130
10	2,98415	3,00389	0,48155	1,10844	1,70048	1,02544
Min	2,87383	3,00389	0,40808	0,95225	1,69171	0,97096
Max	3,40842	3,57056	0,48339	1,10844	1,77955	1,07131
Odchylka	0,16856	0,17698	0,02820	0,04305	0,03278	0,03594
Průměr	3,06876	3,20031	0,44729	1,01367	1,73257	1,02553
Vážený průměr	3,05067	3,17859	0,44768	1,00950	1,73180	1,02663

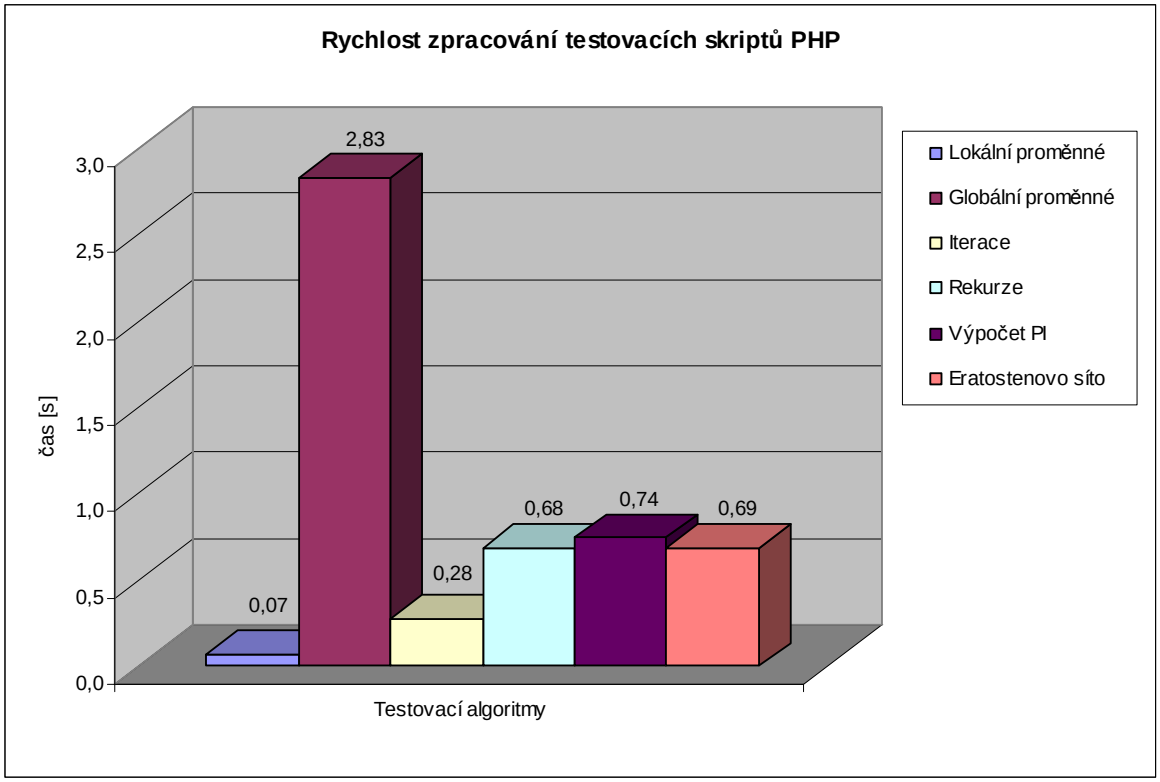


Obr. 21 – Testovací skripty PHP

Tab. 16 – Naměřené hodnoty pro kompilované skripty PHP

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
1	0,07025	3,09420	0,26950	0,71457	0,77328	0,70493
2	0,06672	2,78232	0,28371	0,68592	0,70138	0,71293
3	0,06510	2,72191	0,27807	0,67247	0,71249	0,65649
4	0,22043	2,75826	0,28135	0,67841	0,71305	0,68877
5	0,06476	2,74752	0,24609	0,72210	0,77334	0,66896
6	0,06905	2,78656	0,26819	0,68281	0,70519	0,78819
7	0,08830	3,05013	0,27271	0,67696	0,78486	0,66962
8	0,06903	2,73114	0,27277	0,67819	0,80467	0,66542

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Lokální proměnná	Globální proměnná	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
9	0,05858	2,98848	0,28380	0,66493	0,71185	0,72271
10	0,07004	2,80094	0,30657	0,66129	0,78025	0,66624
Min	0,05858	2,72191	0,24609	0,66129	0,70138	0,65649
Max	0,22043	3,09420	0,30657	0,72210	0,80467	0,78819
Odchylka	0,04597	0,13386	0,01451	0,01876	0,03827	0,03797
Průměr	0,08423	2,84614	0,27628	0,68376	0,74604	0,69443
Vážený průměr	0,07041	2,83067	0,27626	0,68178	0,74429	0,68745

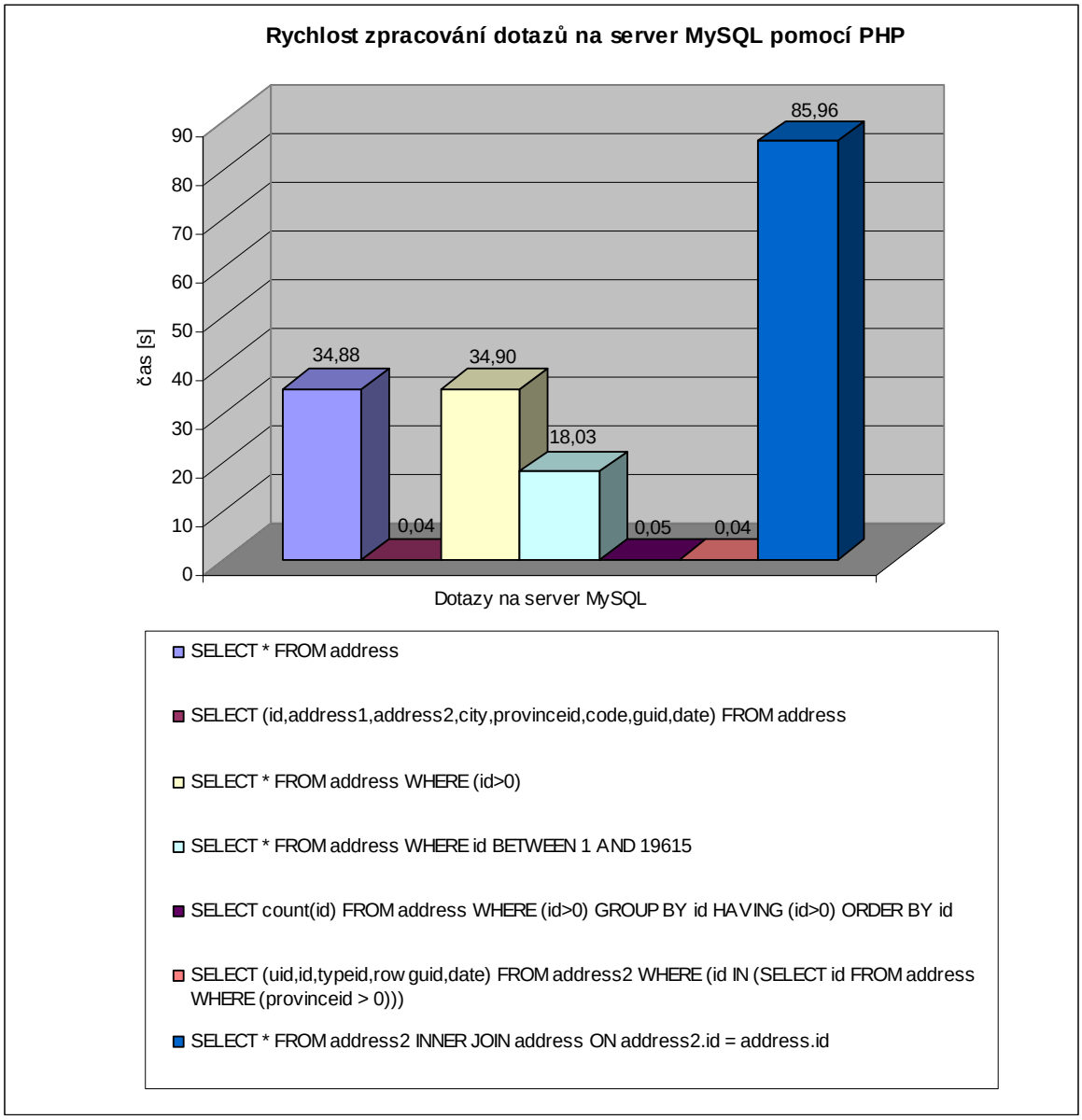


Obr. 22 – Testovací kompilované skripty PHP

6.6.2 PHP a databáze MySQL

Tab. 17 - Naměřené hodnoty pro práci PHP s databází MySQL

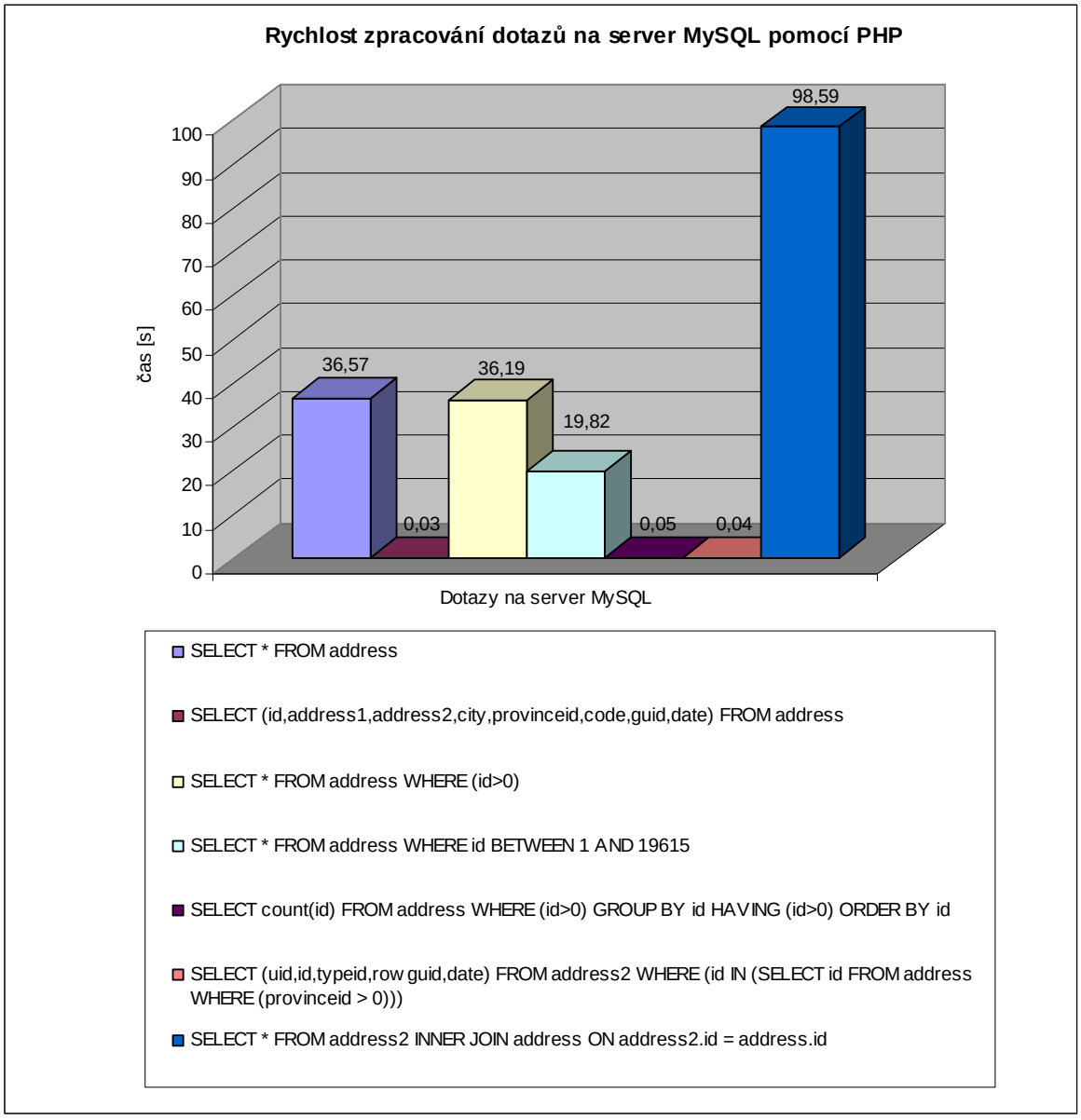
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	37,46879	0,05498	36,60931	19,26745	0,05093	0,04061	89,04533
2	34,55049	0,03460	34,57819	17,86554	0,04455	0,03742	86,24311
3	34,93142	0,03554	34,72372	17,96768	0,04711	0,03593	86,28075
4	34,80169	0,03522	34,83771	17,87693	0,04471	0,03696	85,51336
5	34,91168	0,03753	34,92815	18,02485	0,04441	0,04093	85,62675
6	34,91747	0,03516	34,85697	18,26691	0,05005	0,03545	86,11714
7	34,74431	0,03496	34,93108	18,00283	0,04528	0,03563	85,69375
8	34,98734	0,03493	35,01703	18,02083	0,04570	0,03745	85,62005
9	34,89049	0,03618	35,12913	18,05548	0,04655	0,03499	85,66298
10	34,82032	0,34955	34,79893	17,99317	0,04526	0,03664	86,41353
Min	34,55049	0,03460	34,57819	17,86554	0,04441	0,03499	85,51336
Max	37,46879	0,34955	36,60931	19,26745	0,05093	0,04093	89304533
Odchylka	0,79753	0,09374	0,54235	0,39193	0,00218	0,00195	0,99162
Průměr	35,10240	0,06887	35,04102	18,13417	0,04646	0,03720	89,22168
Vážený průměr	34,87559	0,03806	34,90284	18,02609	0,04615	0,03701	85,95726



Obr. 23 – PHP a databáze MySQL

Tab. 18 – Naměřené hodnoty pro práci kompilovaného PHP s databází MySQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	38,71183	0,04852	38,26702	20,20585	0,06821	0,04205	107,05414
2	35,77089	0,02257	35,45552	19,12696	0,05376	0,03064	97,32329
3	37,09246	0,03389	35,48832	19,10693	0,06381	0,04220	97,40599
4	35,31987	0,03454	35,56408	20,37863	0,05744	0,04099	98,70752
5	38,69546	0,02571	37,22240	20,62804	0,05165	0,03853	99,38526
6	37,52792	0,03050	37,32550	20,10625	0,05302	0,03171	97,18799
7	36,25395	0,03582	36,53108	18,69020	0,04136	0,03650	98,16278
8	35,91294	0,03526	35,88124	20,53211	0,04148	0,03645	99,47454
9	35,55043	0,03534	35,81124	19,72291	0,04951	0,04706	99,44657
10	35,78459	0,03028	35,67749	19,34268	0,04170	0,03339	98,83407
Min	35,31987	0,02257	35,45552	18,69020	0,04136	0,03064	97,18799
Max	38,71183	0,04852	38,26702	20,63804	0,06821	0,04706	107,05415
Odchylka	1,20832	0,00661	0,92387	0,64880	0,00878	0,00495	2,72031
Průměr	36,66203	0,03324	36,32239	19,78506	0,05219	0,03795	99,29841
Vážený průměr	36,57358	0,03267	36,18767	19,81529	0,05155	0,03773	98,59275

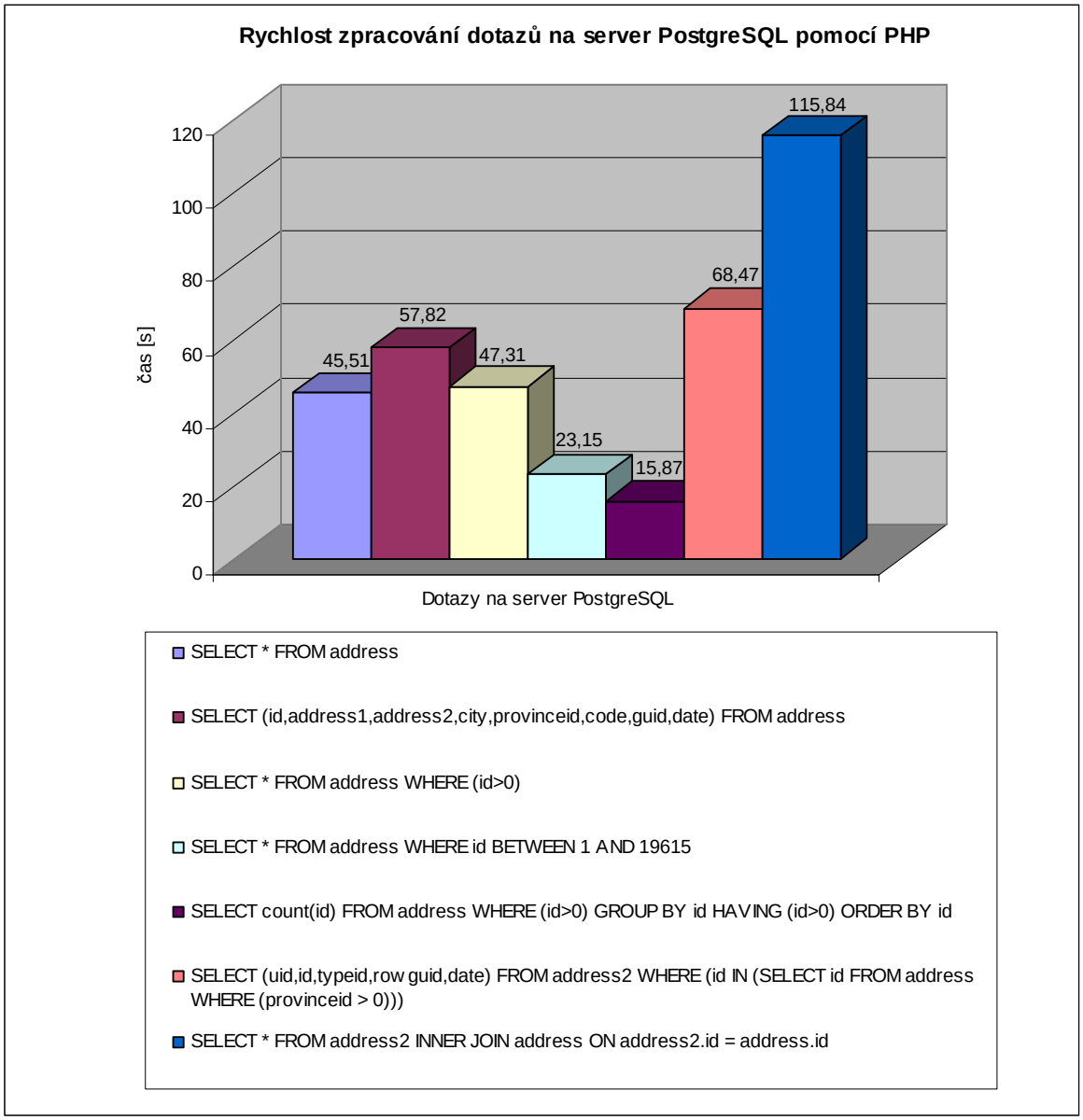


Obr. 24 – Kompilované PHP a databáze MySQL

6.6.3 PHP a databáze PostgreSQL

Tab. 19 – Naměřené hodnoty pro práci PHP s databází PostgreSQL

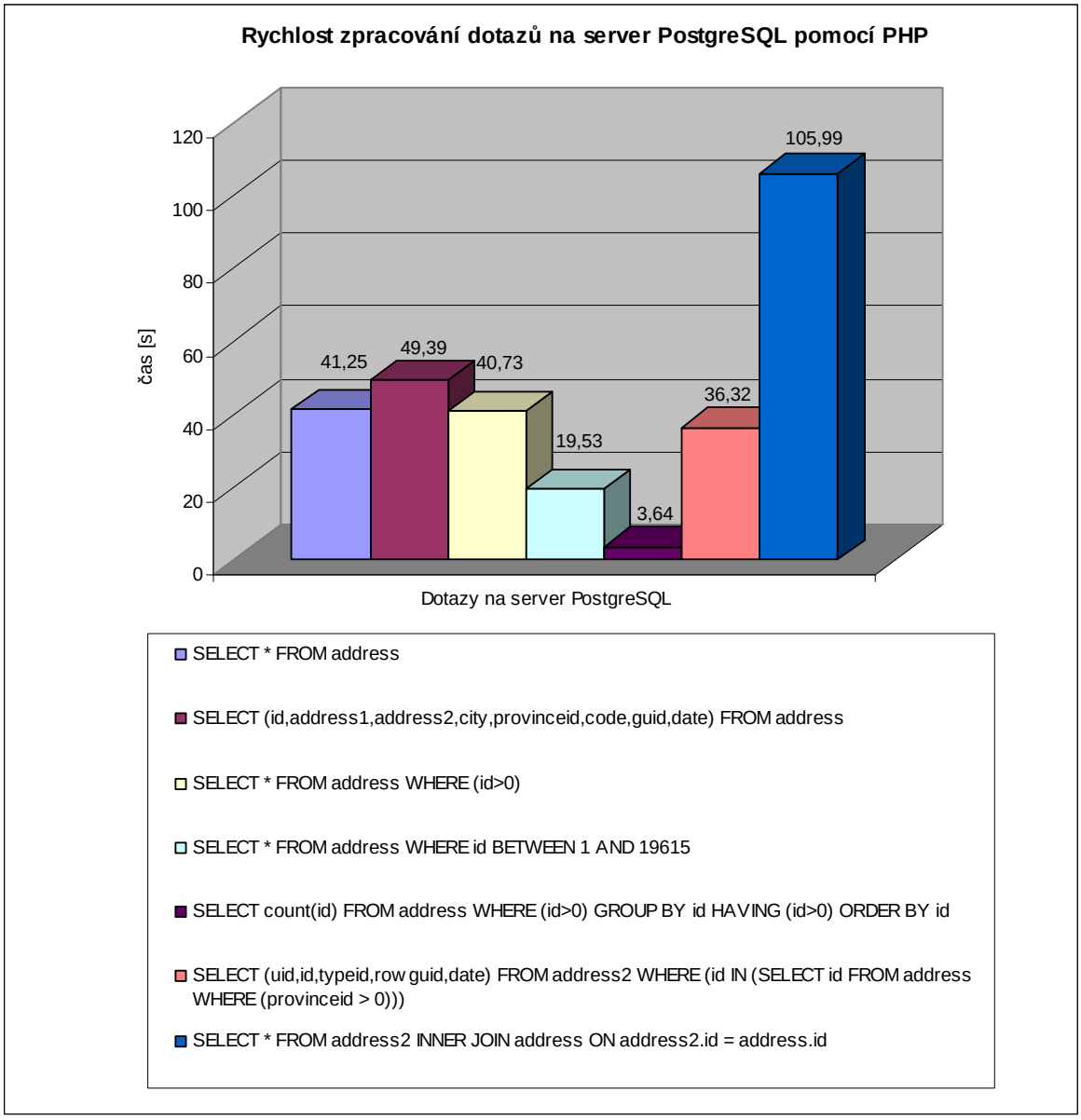
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	52,02919	65,36609	53,91056	26,66982	15,80626	69,04113	116,97834
2	46,17564	56,65459	46,10976	22,61003	15,78514	68,39790	117,05344
3	45,40865	56,82895	46,26501	22,35792	15,82609	68,10274	115,30859
4	44,87306	56,75539	46,01996	22,37246	15,80336	67,58605	115,55977
5	44,68163	56,25207	45,76528	22,62687	15,84487	67,46536	114,35688
6	46,29517	57,06633	47,95734	22,72046	16,32628	69,93022	118,91879
7	46,33624	78,08865	61,82697	27,45268	16,26307	73,84935	117,68093
8	45,29631	56,74138	46,36431	22,97098	15,95285	68,83767	114,80072
9	45,03919	56,89286	46,08757	22,54379	15,66053	67,71835	114,85250
10	44,60892	55,69774	45,63589	22,64642	15,67278	68,15622	114,48763
Min	44,60892	55,69774	45,62589	22,35792	15,66053	67,46536	114,35688
Max	52,02919	78,08865	61,82697	27,45268	16,32628	73,84935	118,91879
Odchylka	2,07938	6,69230	5,00010	1,79811	0,21559	1,79450	1,47735
Průměr	46,07440	59,63441	48,59427	23,49714	15,89412	68,90850	115,99976
Vážený průměr	45,51324	57,81971	47,30997	23,14510	15,86930	68,47129	115,84024



Obr. 25 – PHP a databáze PostgreSQL

Tab. 20 – Naměřené hodnoty pro práci kompilovaného PHP s databází PostgreSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	43,38672	50,25397	46,80736	22,84475	5,74475	39,13188	112,19302
2	44,02697	52,88893	44,05938	20,75792	3,60810	41,77657	141,28817
3	44,18741	52,31960	41,37218	20,01503	3,85699	36,29248	106,25210
4	41,55484	50,78884	41,09689	19,96077	3,72614	35,16225	96,90773
5	40,30000	48,44300	40,23651	19,40137	3,79414	34,72551	99,78701
6	40,83396	48,71461	40,17293	19,15894	3,76040	36,66060	106,99156
7	39,69848	47,81428	39,43113	18,93917	3,41193	36,01815	110,08310
8	40,04786	48,61731	39,66880	19,01502	3,52478	36,24038	105,97291
9	39,79206	48,18523	39,71678	19,01384	3,40032	36,34602	109,78610
10	40,02646	47,60422	39,47859	18,91117	3,42198	34,26736	94,81489
Min	39,69848	47,60422	39,43113	18,91117	3,40032	34,26736	94,81489
Max	44,18741	52,88893	46,80736	22,84475	5,74475	41,77657	141,28817
Odchylka	1,71364	1,79729	2,28645	1,16599	0,65969	2,11711	12,26406
Průměr	41,38548	49,56300	41,20406	19,80180	3,82495	36,66212	108,40766
Vážený průměr	41,24611	49,39211	40,72526	19,53276	3,63806	36,32216	105,99669

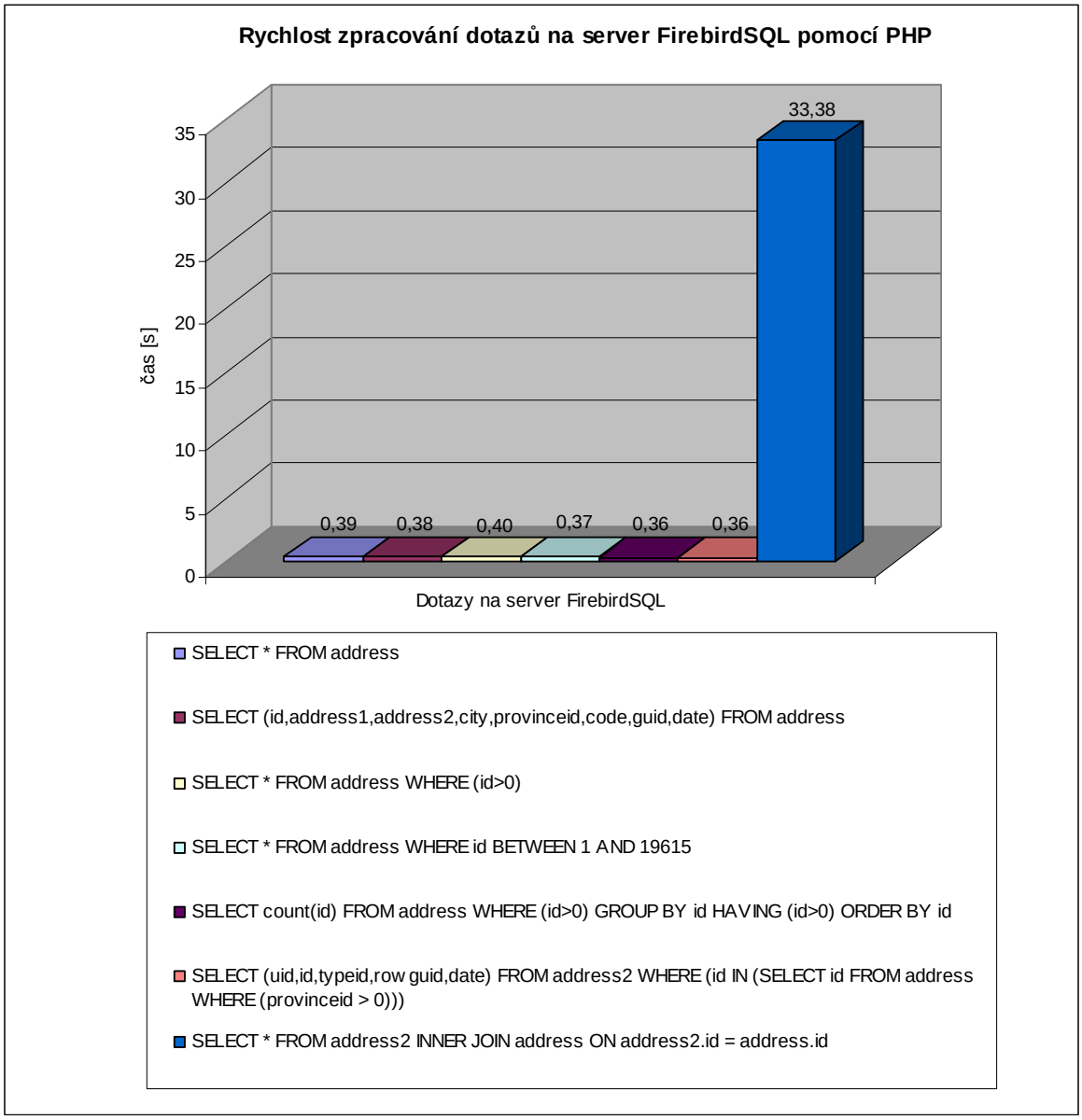


Obr. 26 – Kompilované PHP a databáze PostgreSQL

6.6.4 PHP a databáze FirebirdSQL

Tab. 21 - Naměřené hodnoty pro práci PHP s databází FirebirdSQL

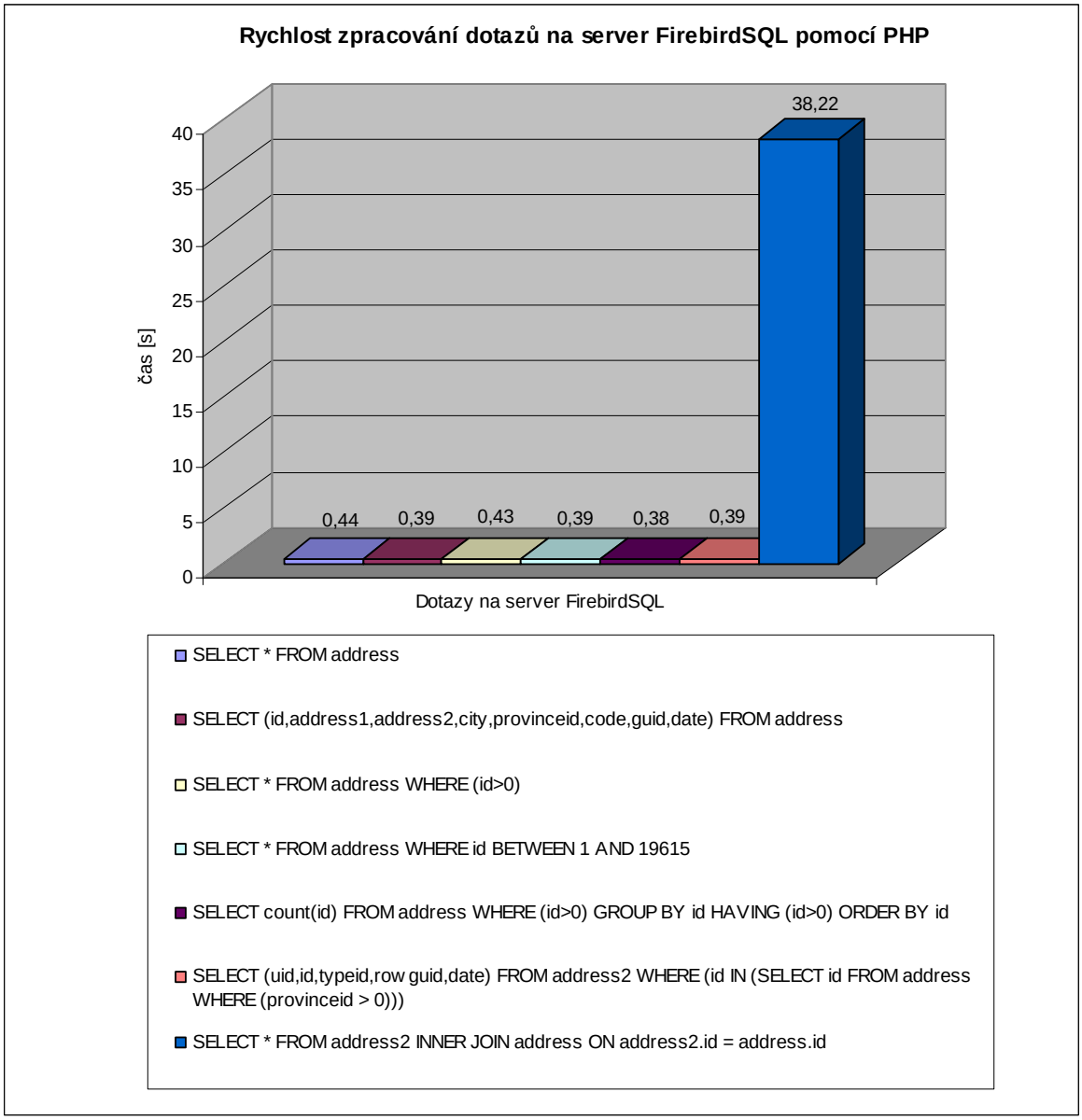
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,61483	0,34232	0,40855	0,44105	0,39388	0,37916	34,89680
2	0,38411	0,37387	0,33643	0,33906	0,34654	0,34331	33,90998
3	0,40202	0,33477	0,39533	0,34400	0,34391	0,35416	33,25439
4	0,33602	0,40536	0,39107	0,36126	0,39686	0,42142	33,08119
5	0,37562	0,41778	0,38357	0,41031	0,36199	0,35541	33,28626
6	0,40614	0,37209	0,43970	0,37199	0,36980	0,35790	33,37650
7	0,40717	0,37951	0,39240	0,38259	0,34111	0,36204	32,93839
8	0,37737	0,38607	0,39494	0,35978	0,34721	0,35976	33,64620
9	0,42457	0,36300	0,40442	0,36182	0,36823	0,36856	33,16396
10	0,37666	0,40096	0,45264	0,39018	0,34543	0,36773	33,30394
Min	0,33602	0,33477	0,33643	0,33906	0,34111	0,34331	32,93839
Max	0,61483	0,41778	0,45264	0,44105	0,39686	0,42142	34,89680
Odchylka	0,07194	0,02518	0,02989	0,02956	0,01956	0,02033	0,53869
Průměr	0,41045	0,37757	0,39991	0,37620	0,36150	0,36695	33,48576
Vážený průměr	0,39421	0,37790	0,40125	0,37274	0,35962	0,36309	33,37780



Obr. 27 – PHP a databáze FirebirdSQL

Tab. 22 – Naměřené hodnoty pro práci kompilovaného PHP s databází FirebirdSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1,03375	0,33812	0,44475	0,42623	0,39199	0,41741	55,60782
2	0,45051	0,62376	0,70729	0,56211	0,51560	0,52412	42,67795
3	0,45229	0,48045	0,52934	0,49661	0,45398	0,47060	49,65075
4	0,45616	0,37195	0,47195	0,37554	0,36523	0,37903	46,84611
5	0,44864	0,37955	0,37853	0,35601	0,34203	0,35596	32,90089
6	0,42922	0,38185	0,40534	0,35758	0,34853	0,36102	34,05442
7	0,41002	0,36884	0,37203	0,38369	0,36476	0,35232	33,59309
8	0,42490	0,38058	0,38297	0,34286	0,36170	0,36393	33,07994
9	0,44911	0,36006	0,40510	0,34991	0,36316	0,37322	32,81475
10	0,44164	0,38980	0,37100	0,36601	0,35043	0,36084	32,94590
Min	0,41002	0,33812	0,37203	0,34286	0,34203	0,35232	32,81475
Max	1,03375	0,62376	0,70729	0,56211	0,51560	0,52412	55,60782
Odchylka	0,17851	0,08024	0,09806	0,06914	0,05296	0,05499	8,14528
Průměr	0,49992	0,40750	0,44883	0,40165	0,38574	0,39585	39,41736
Vážený průměr	0,44443	0,38914	0,42612	0,38895	0,37497	0,38525	38,21863

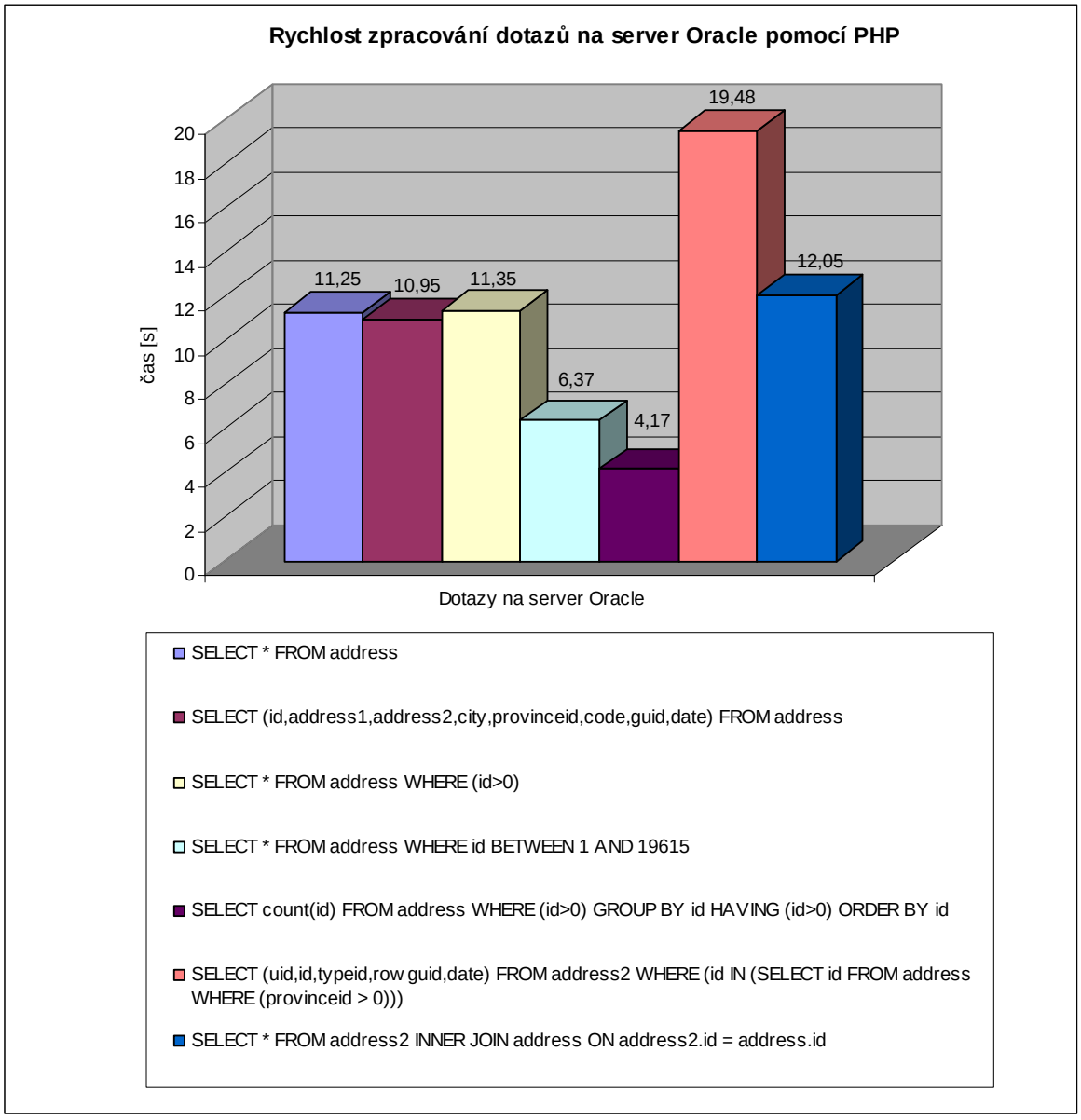


Obr. 28 – Kompilované PHP a databáze FirebirdSQL

6.6.5 PHP a databáze Oracle

Tab. 23 – Naměřené hodnoty pro práci PHP s databází Oracle

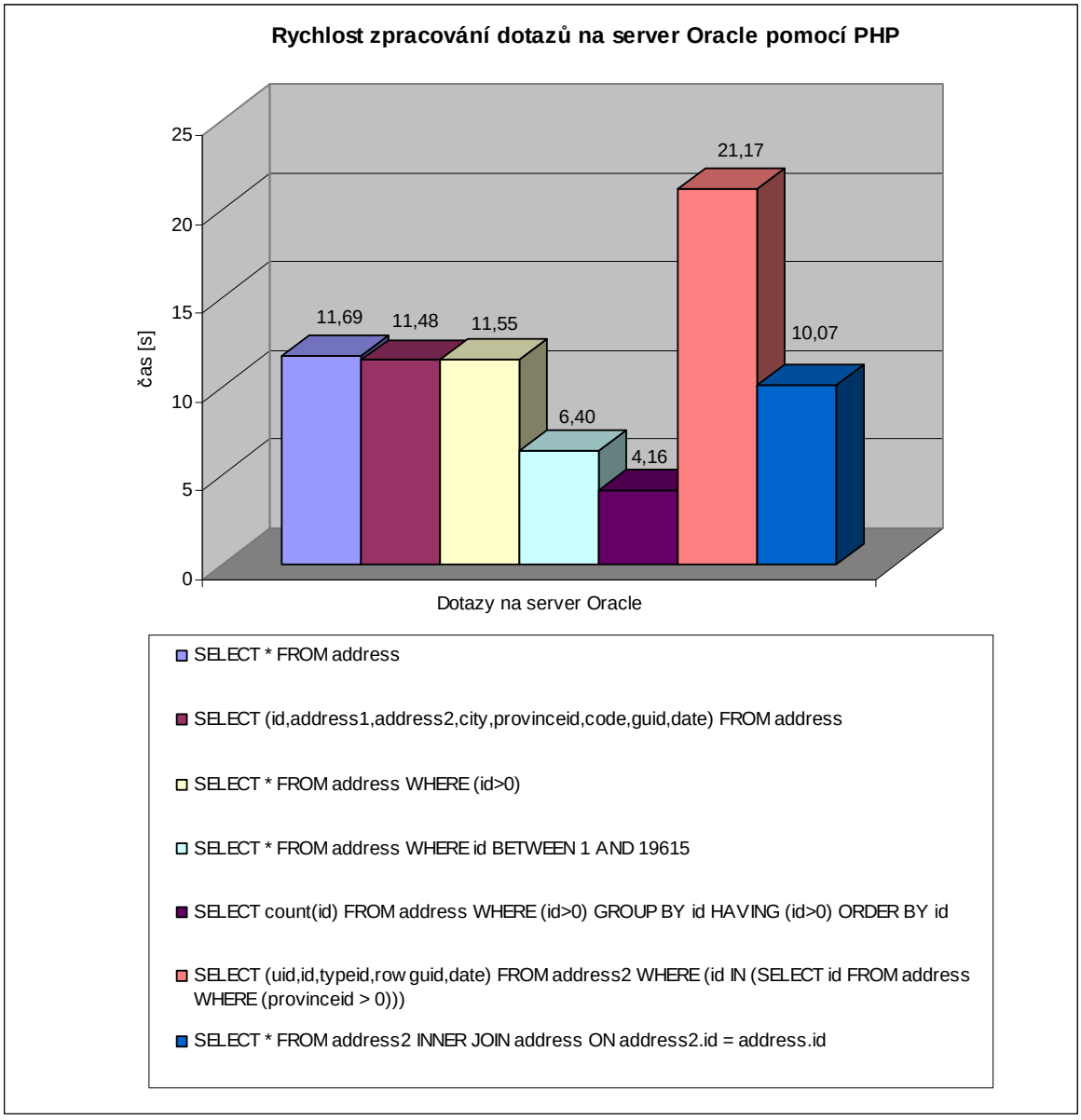
	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	11,01216	10,73078	10,71301	6,10101	4,11147	17,25962	10,36966
2	11,01555	10,79055	10,85117	6,12072	4,22475	20,97865	12,55905
3	10,95366	10,73451	13,74003	10,20039	5,78371	27,02820	23,93151
4	13,44517	13,17008	13,48923	7,36223	4,16862	21,14301	13,92659
5	11,25260	10,50446	10,55110	5,83369	4,10106	17,07845	10,59362
6	10,80482	10,60761	10,67469	6,11383	4,12431	20,62083	18,21730
7	13,16973	12,91455	13,07022	7,17484	4,19691	20,77517	10,09292
8	10,96646	10,60697	10,66771	6,00361	4,17205	17,16410	10,13837
9	10,74508	10,55695	10,63758	6,04079	4,18087	20,65508	10,47633
10	10,85317	10,67726	10,70679	6,02490	4,18653	17,23290	10,12994
Min	10,74508	10,50446	10,55110	5,83369	4,10106	17,07845	10,09292
Max	13,44517	13,17008	13,74003	10,20039	5,78371	27,02820	23,93151
Odchylka	0,95390	0,96170	1,26992	1,26809	0,48766	2,91259	4,37844
Průměr	11,42184	11,12937	11,51015	6,69760	4,32503	19,99460	13,04353
Vážený průměr	11,25352	10,95240	11,35130	6,36774	4,17069	19,47992	12,05136



Obr. 29 – PHP a databáze Oracle

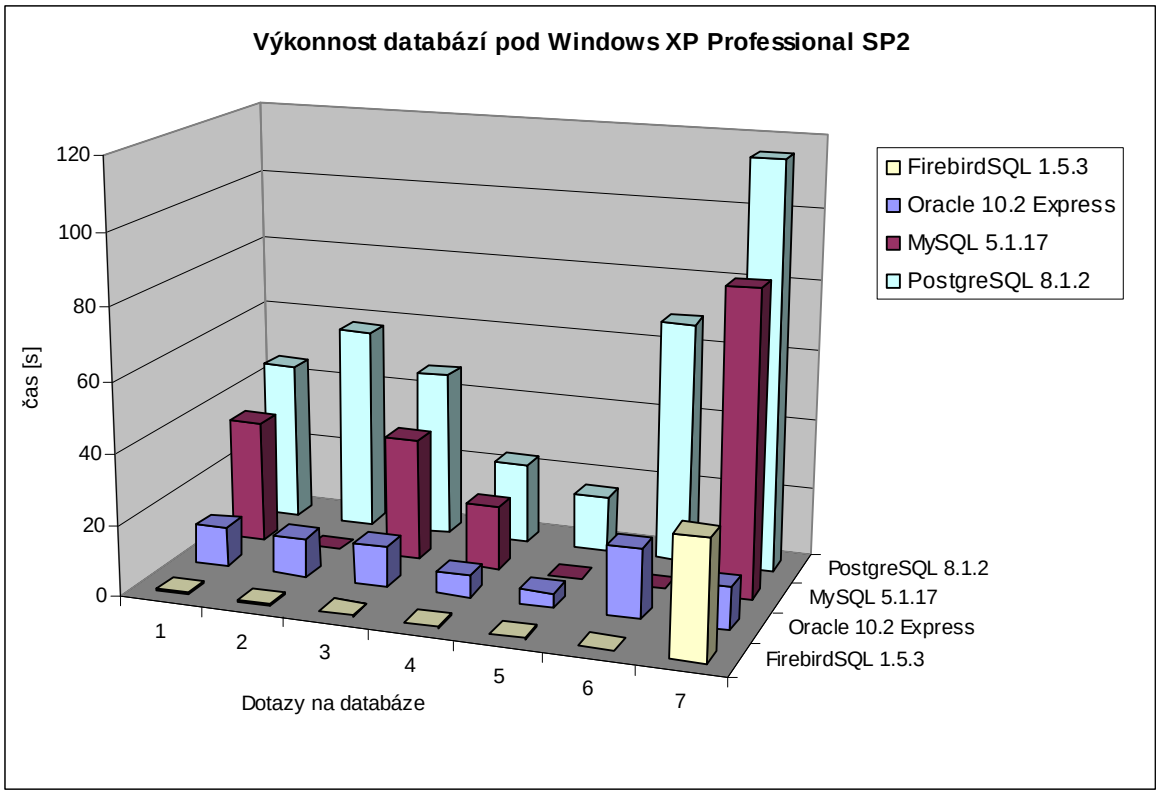
Tab. 24 – Naměřené hodnoty pro práci kompilovaného PHP s databází Oracle

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	13,86491	13,57773	13,70020	7,34389	4,13613	21,22997	10,04842
2	13,78339	13,28695	13,63049	7,38755	4,15379	21,25900	10,27523
3	10,70726	10,73891	10,91275	6,11921	4,23413	21,35076	9,99881
4	13,95927	13,26269	13,45658	7,41346	4,18764	21,20854	10,32668
5	10,96268	11,17209	10,83264	6,00700	4,11101	20,90618	10,04973
6	11,09060	10,78220	10,72905	6,07687	4,16254	21,11923	10,21538
7	10,90424	10,71262	10,80917	6,02260	4,21244	17,54931	9,97399
8	11,01670	10,76707	10,93099	6,09666	4,16121	21,18246	9,90011
9	10,93560	10,93304	10,84922	6,04067	4,13467	21,19366	9,94496
10	10,96985	10,91092	10,97576	6,08837	4,14790	21,21765	10,04986
Min	10,70726	10,71262	10,72905	6,00700	4,11101	17,54931	9,90011
Max	13,95927	13,57773	13,70020	7,41346	4,23413	21,35076	10,32668
Odchylka	1,34562	1,16250	1,25532	0,60466	0,03555	1,09622	0,13730
Průměr	11,81945	11,61442	11,68269	6,45963	4,16415	20,82167	10,07832
Vážený průměr	11,69100	11,48173	11,54970	6,39698	4,16204	21,16458	10,06955

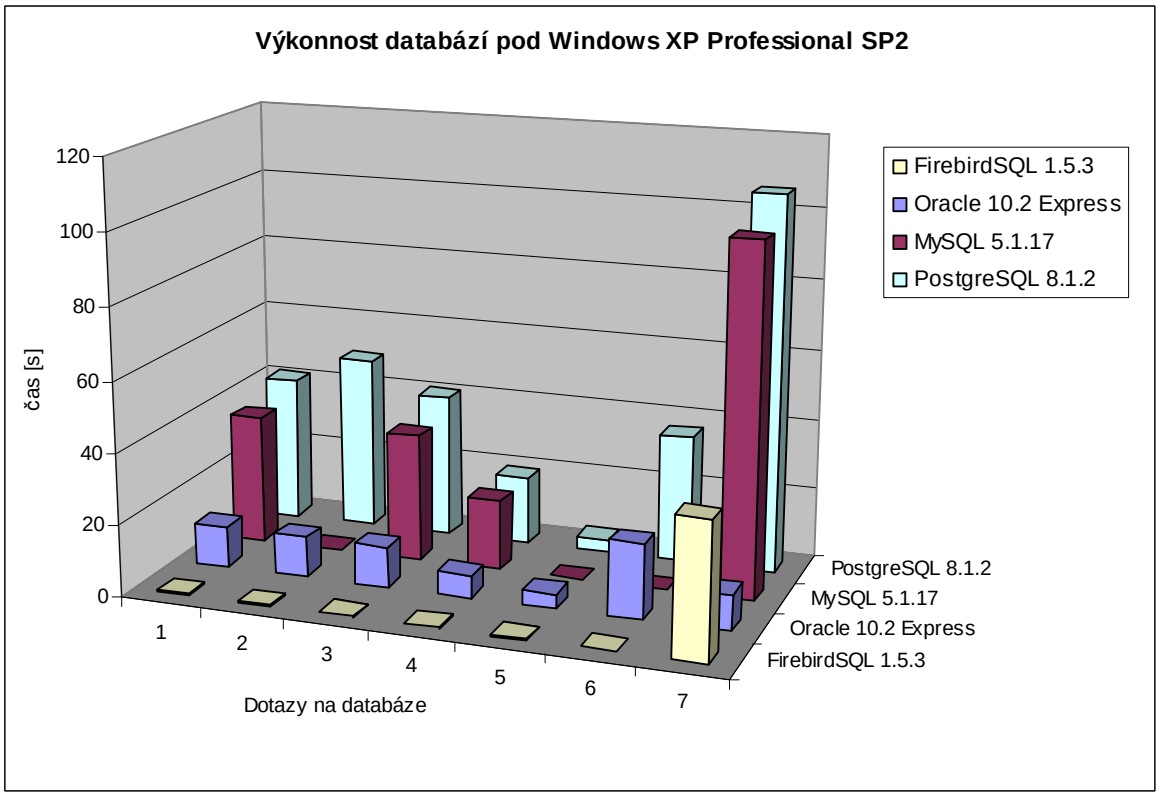


Obr. 30 – Kompilované PHP a databáze Oracle

6.6.6 Srovnání výkonu databází u PHP pod operačním systémem Windows XP



Obr. 31 - Výkonnost přístupu k databázím pomocí PHP pod Windows



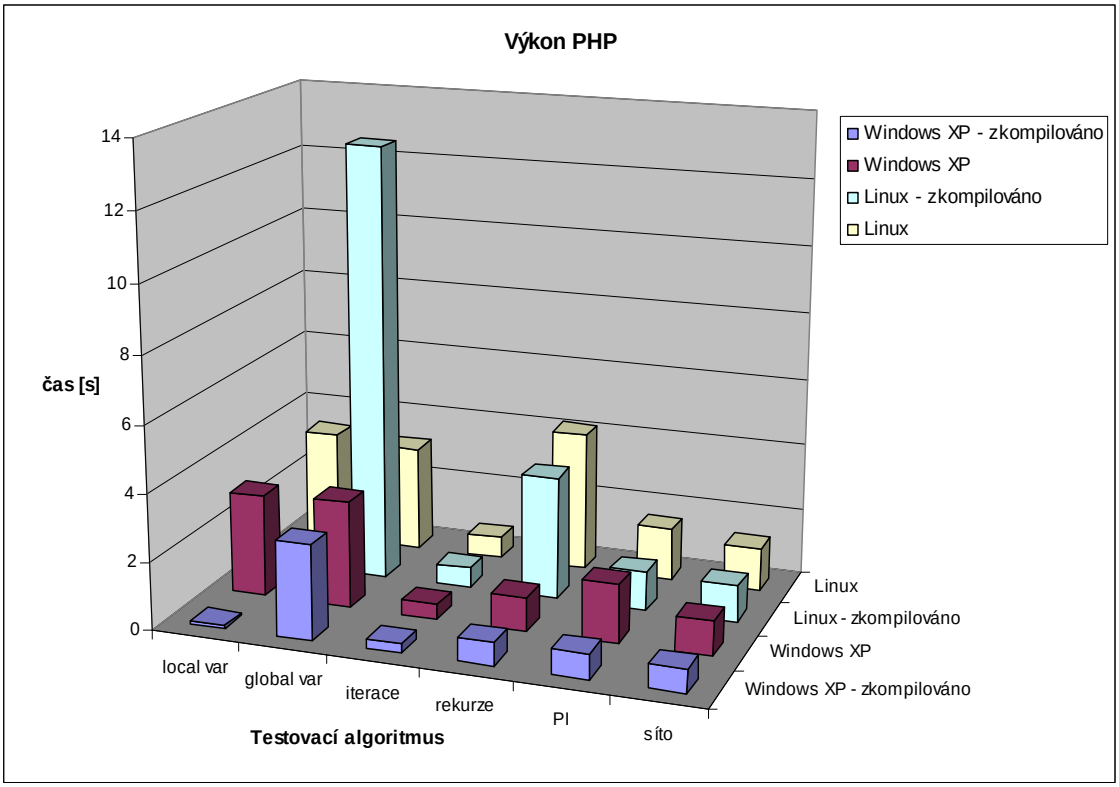
Obr. 32 - Výkonnost přístupu k databázím pomocí kompilovaného PHP pod Windows

6.7 Srovnání výsledků PHP

6.7.1 Výkon PHP

Tab. 25 – Porovnání výsledků skriptů měřících výkon PHP

	Bez kompilace		Kompilováno	
Testovací skripty / čas [s]	**Linux**	**Windows**	**Linux**	**Windows**
Lokální proměnná	3,34612	3,05067	0,17458	0,07041
Globální proměnná	3,16093	3,17859	12,99134	2,83067
Iterace	0,64260	0,44768	0,59342	0,27626
Rekurze	4,18701	1,00950	3,67407	0,68178
Výpočet π	1,55529	1,73180	1,14089	0,74429
Eratostenovo síto	1,28049	1,02663	1,07737	0,68745

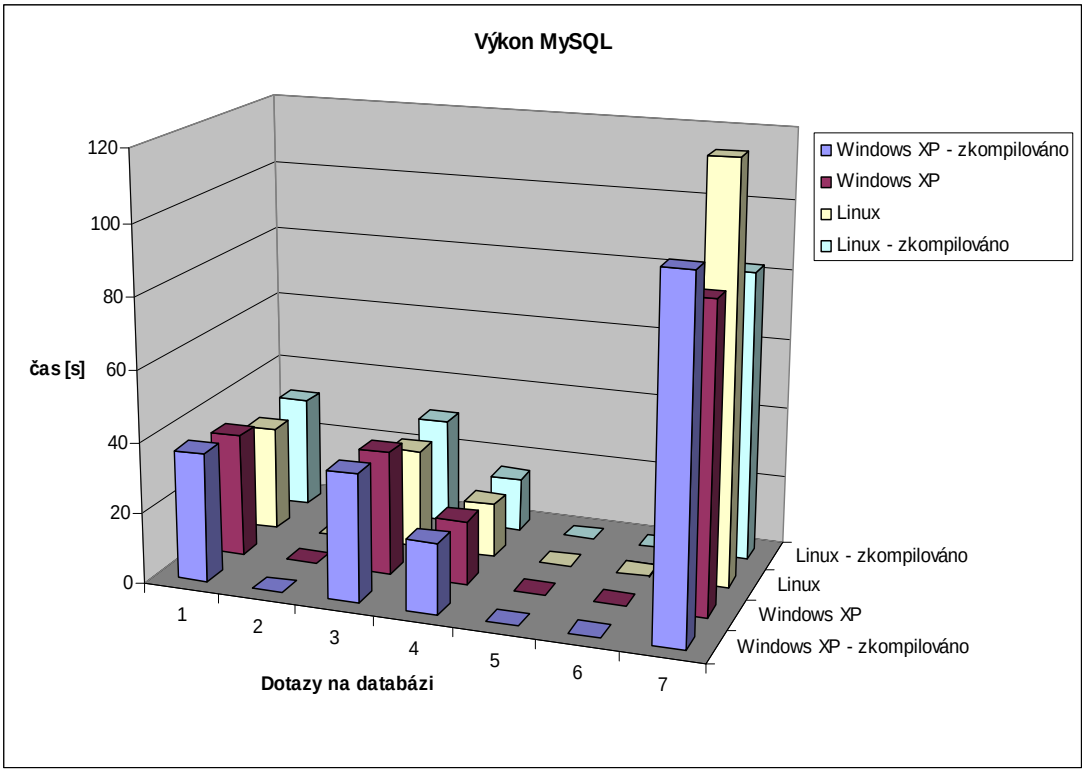


Obr. 33 – Výsledky měření výkonu PHP

6.7.2 PHP a databáze MySQL

Tab. 26 – Porovnání výkonu spolupráce PHP a MySQL

	Bez kompilace		Kompilováno	
Dotaz / čas [s]	**Linux**	**Windows**	**Linux**	**Windows**
1	29,75821	34,87559	31,80879	36,57358
2	0,11853	0,038063	0,096274	0,03267
3	28,19376	34,90284	30,34198	36,18767
4	15,28288	18,02609	15,28286	19,81529
5	0,13173	0,046151	0,084409	0,05155
6	0,13428	0,037011	0,074125	0,03773
7	118,04389	85,95726	82,48241	98,59275

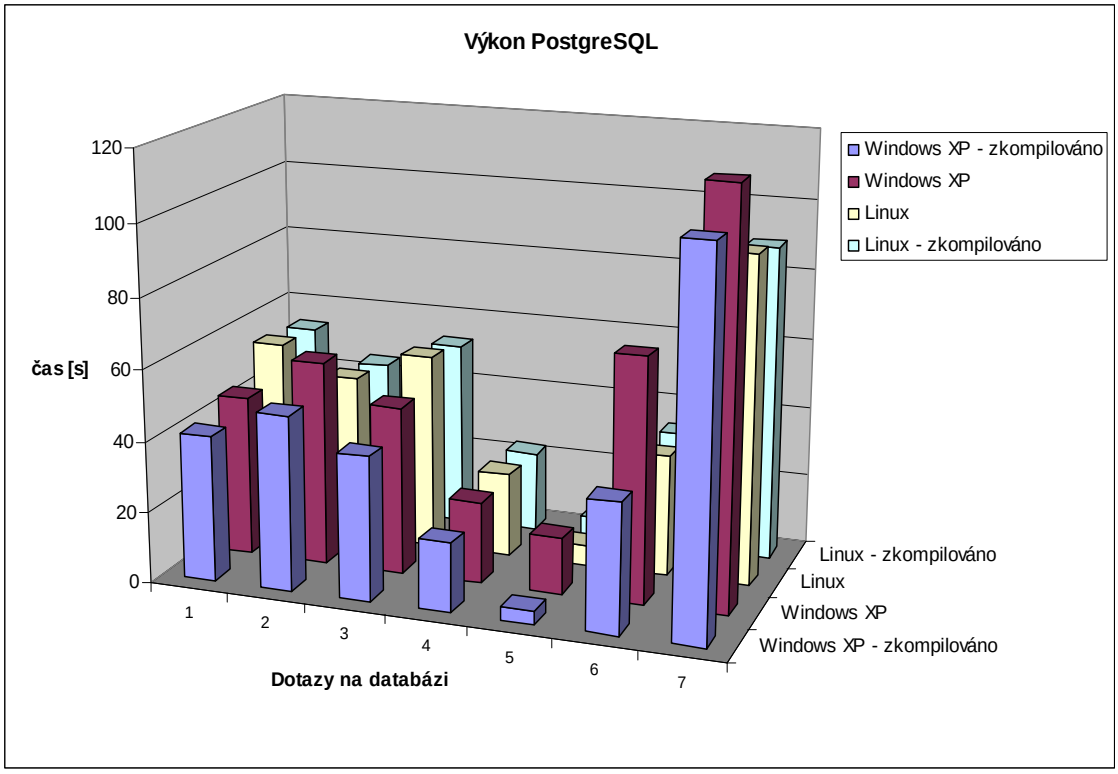


Obr. 34 – Výkon databáze MySQL

6.7.3 PHP a databáze PostgreSQL

Tab. 27 – Porovnání výkonu spolupráce PHP a PostgreSQL

	Bez kompilace		Kompilováno	
Dotaz / čas [s]	**Linux**	**Windows**	**Linux**	**Windows**
1	54,65075	45,51324	53,49006	41,24611
2	47,04437	57,81971	44,75052	49,39211
3	55,48849	47,30997	52,51269	40,72526
4	23,77376	23,14510	22,73018	19,53276
5	5,65587	15,86930	5,59850	3,63806
6	34,19309	68,47129	33,57132	36,32216
7	92,64623	115,84020	89,18677	105,99669

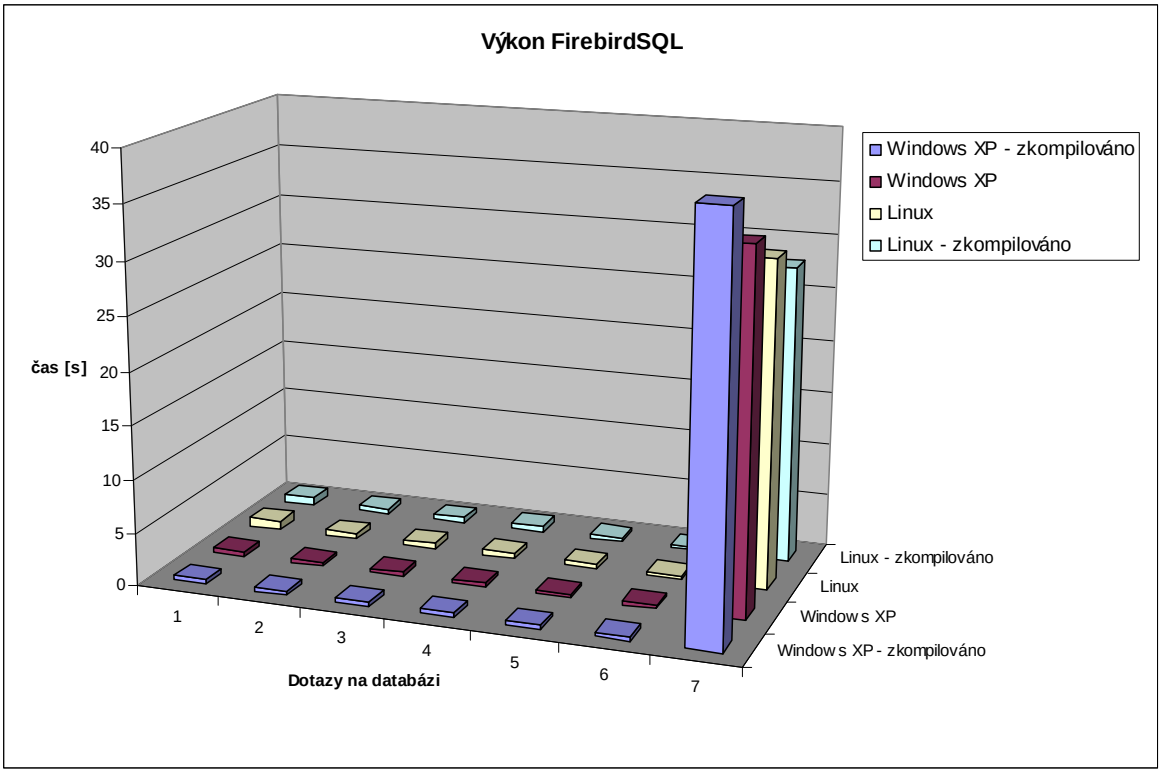


Obr. 35 – Výkon databáze PostgreSQL

6.7.4 PHP a databáze FirebirdSQL

Tab. 28 – Porovnání výkonu spolupráce PHP a FirebirdSQL

	Bez kompilace		Kompilováno	
Dotaz / čas [s]	**Linux**	**Windows**	**Linux**	**Windows**
1	0,86928	0,39421	0,76627	0,44443
2	0,53732	0,37790	0,48628	0,38914
3	0,54909	0,40125	0,51616	0,42612
4	0,54832	0,37274	0,56455	0,38895
5	0,43600	0,35962	0,35996	0,37497
6	0,33774	0,36309	0,27483	0,38525
7	30,44547	33,37780	27,88046	38,21863

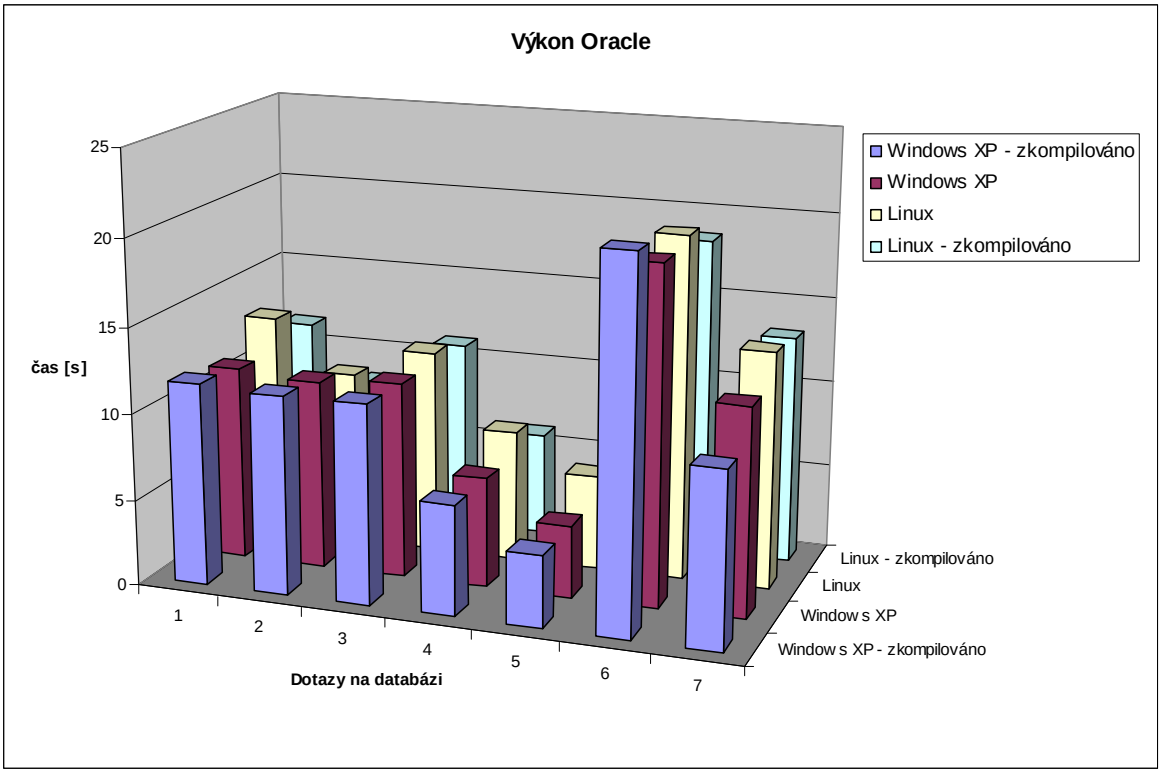


Obr. 36 – Výkon databáze FirebirdSQL

6.7.5 PHP a databáze Oracle

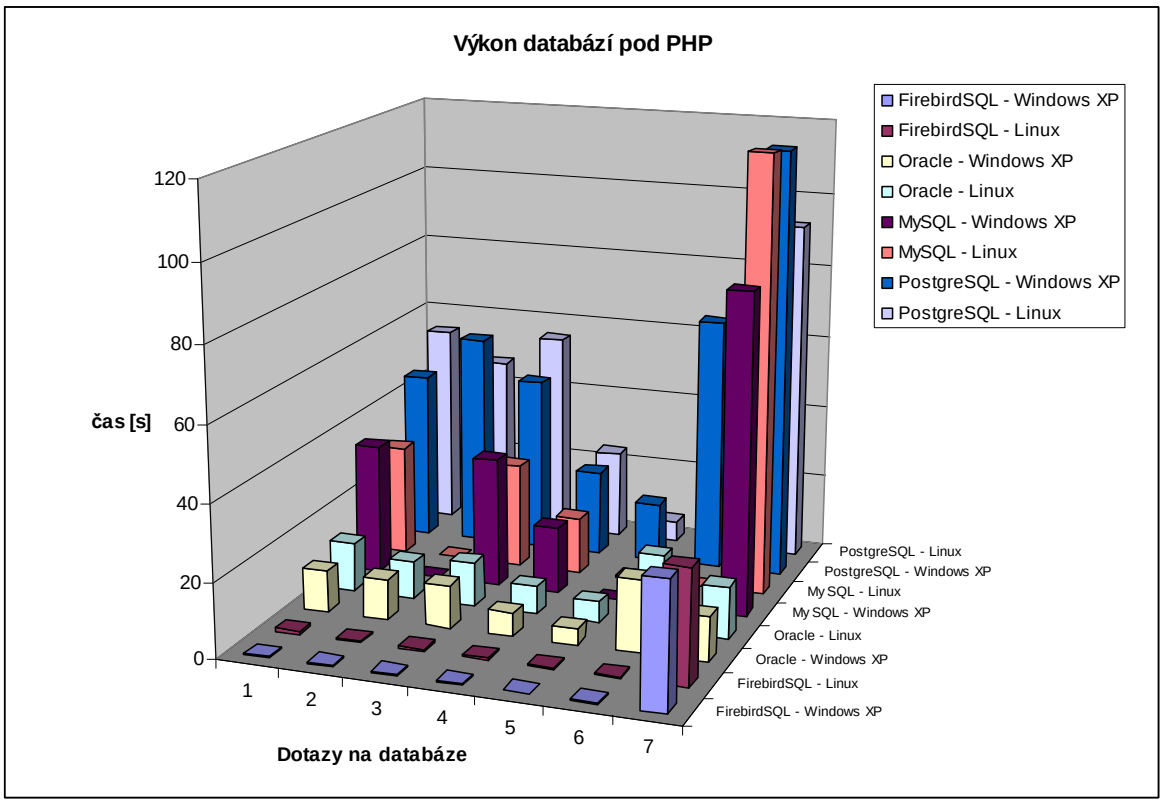
Tab. 29 – Porovnání výkonu spolupráce PHP a Oracle

	Bez kompilace		Kompilováno	
Dotaz / čas [s]	**Linux**	**Windows**	**Linux**	**Windows**
1	13,05556	11,25352	11,52651	11,69100
2	10,06104	10,95240	8,29604	11,48173
3	11,88339	11,35130	11,04869	11,54970
4	7,60049	6,36774	5,96561	6,39698
5	5,48574	4,17069	3,86412	4,16204
6	19,92093	19,47992	18,58051	21,16458
7	13,73692	12,05136	13,30113	10,06955

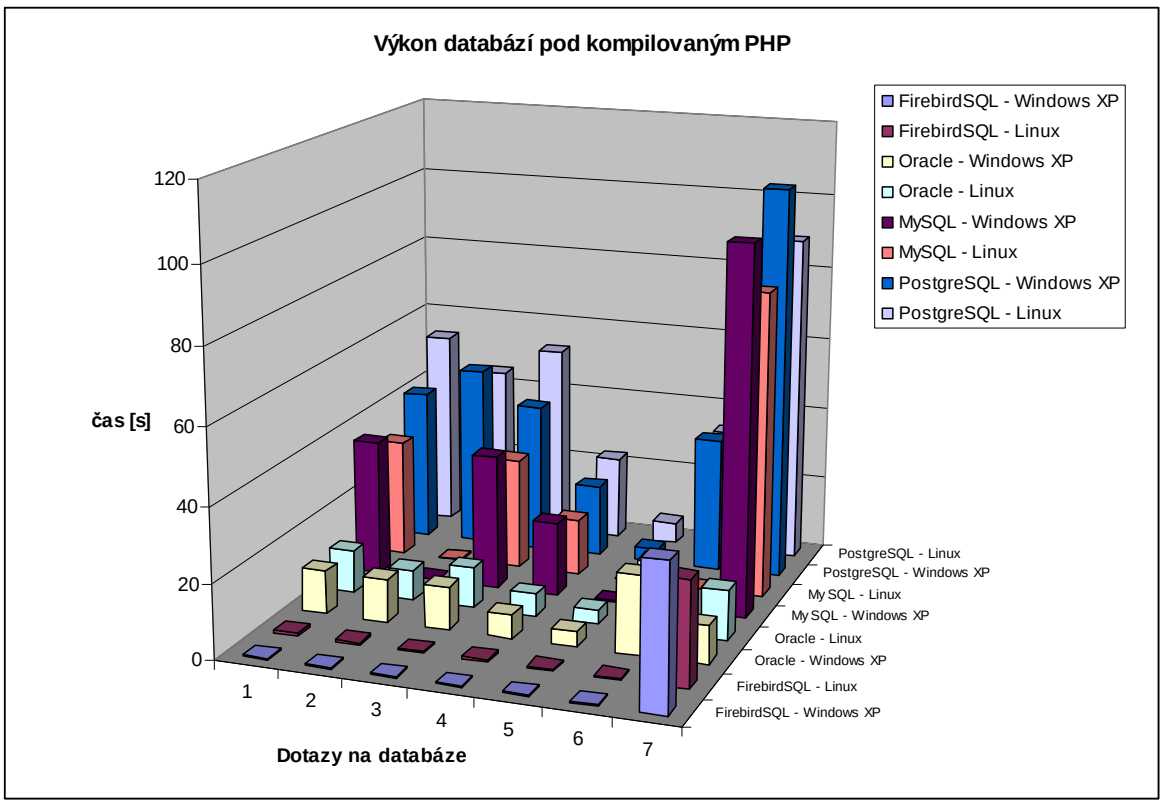


Obr. 37 – Výkon databáze Oracle

6.7.6 Celkové srovnání databází a PHP



Obr. 38 – Výkon databází pod PHP



Obr. 39 – Výkon databází pod kompilovaným PHP

7 ASP.NET

ASP.NET je nadstavba .NET Frameworku společnosti Microsoft pro tvorbu webových aplikací a služeb. Je nástupcem technologie ASP a přímým konkurentem JSP. Nelze však již hovořit o skriptovací technologii, a to právě díky tomu, že ASP.NET přímo vychází z .NET Frameworku a je plně kompilované. Přestože jsou ASP.NET i .NET Framework komerční produkty, jsou zdarma včetně některých vývojových nástrojů jako je Microsoft Visual Web Developer Express, Microsoft Visual Basic.NET a dalších.

7.1 Technologie .NET Framework

Technologie .NET Framework je rozsáhlý objektový model společný jak pro vývoj webových aplikací, tak zejména pro vývoj klasických „WinForm“ aplikací. Je možné se přes něj spojit s databázemi, pracovat se soubory, s cookies, se sessions, s grafikou atd.

Jednou z nejdůležitějších částí .NET Frameworku je CLR (Common Language Runtime) – běhové prostředí. Normální programovací jazyk, ve kterém jsou ASP.NET stránky napsány, je totiž při kompilaci převeden do takzvaného řízeného kódu (MSIL, Microsoft Intermediate Language), což není ani strojový jazyk ani zdrojový kód. Jde o mezijazyk, který vykoná právě CLR.

V současné době je nejnovější .NET Framework verze 2. V tomto případě nejsou ale verze kompatibilní s předcházejícími, jsou to odlišné produkty. Je tak možné mít na serveru nainstalovaný .NET Framework verze 2 i starší 1.1, příp. 1.0 a přepínat se mezi nimi.

7.2 Podrobný pohled na ASP.NET

Díky tomu, že je ASP.NET založeno na CLR, který je sdílen všemi aplikacemi postavenými na .NET Frameworku, je možné realizovat projekty ASP.NET v jakémkoliv jazyce podporujícími kompilaci do mezikódu, jakými jsou nejčastěji Visual Basic.NET, C#, ale i Jscript.NET, Visual C++, mutace Perlu, Pythonu a další. Na rozdíl od skriptovacích jazyků, kde jsou stránky při každém přístupu stále znova a znova interpretovány, jsou stránky ASP.NET mnohem rychlejší, neboť při prvním přístupu dojde ke kompilace a poté se již stále (do doby, než programátor provede změny ve zdrojovém kódu) využívá rychlá zkompilovaná verze v podobě DLL souborů.

ASP.NET přichází se způsobem vývoje webových aplikací podobným jako vývoj aplikací stolních. Jde o webové ovládací prvky (Controls), které jsou ekvivalentem ovládacích prvku ve Windows. Při tvorbě webových stránek je tak možné používat tlačítka (Button), nápisy (Label), textová pole (TextBox), a spoustu dalších. Těmto ovládacím prvkům lze přiřazovat vlastnosti, zachytávat události (OnClick, OnLoad, …) apod. Rovněž je možné vytváření vlastních ovládacích prvků. Všechny ovládací prvky jsou ve webovém prohlížeči klienta zobrazeny jako HTML kód. Toto zobrazování je však velmi inteligentní a při veškeré činnosti bere zřetel na internetový prohlížeč, který klient používá. Např. při používání proměnných se není potřeba starat o jejich přenos. ASP.NET se samostatně rozhodne jakým způsobem toho docílit. Rozhodne se, jestli použít cookies, nebo pokud je má uživatel vypnuté, jestli použít přenos v URL, apod.

Mezi dalšími vlastnostmi je použití šablon, čímž se výrazně redukuje duplicitní kód, velký výběr ovládacích prvků a knihoven tříd, schopnost cachovat celou stránku nebo pouze její část, generování validního HTML 4.0 / XHTML 1.0 / XHTML 1.1 kódu a JavaScriptu a spousta dalších. Nevýhodou je drahý webhosting z důvodu vysoké ceny serverových operačních systémů společnosti Microsoft a nákladů na hardware serverů, který musí být dostatečně výkonný.

7.3 Prostředí pro provádění testů

V běžném provozu na Internetu, je možné se setkat s ASP.NET pouze na serverech s operačními systémy Microsoft a webovými servery IIS. ASP.NET však lze provozovat třeba i pod webovým serverem Apache a operačním systémem Linux, ale to je spíš kuriozita a dosud jsem se s tím nesetkal a ani bych podobné „experimenty“ nedoporučoval. Mezi takovéto případy patří i Mono, které má být alternativní implementace Microsoft .NET Frameworku. Testy ASP.NET jsou provedeny pod operačním systémem Microsoft Windows XP Professional SP2 na webovém serveru IIS 5.0 a databázovém serveru Microsoft SQL Server 2005 Express, FirebirdSQL 1.5.3 a Oracle 10.2 Express. Pro zajímavost je Monu věnována kapitola 7.6.

7.4 Testovací skripty

Dále je uveden kompletní přepis testovacích algoritmů v úpravě pro ASP.NET. Připojení k různým databázím je téměř totožné, liší se pouze názvem služby databázového serveru a je proto uveden pouze skript s napojením na Microsoft SQL Server 2005 Express.

7.4.1 Práce s řetězci

Prog. 14 – ASP.NET: Pomocí klasické metody práce s řetězci

```
Dim s As String = ""
Dim st, et As DateTime
Dim i As Integer

st = DateTime.Now
For i = 1 To 100000
   s = s&"a"
Next
et = DateTime.Now
Dim t as TimeSpan = et – st
Label1.Text = "Zpracoval jsem " & s.Length & " řetězců."
Label2.Text = "Celková doba" & t.TotalSeconds & " s."
```

Prog. 15 – ASP.NET: Pomocí komponenty StringBuilder

```
Dim s As String = ""
Dim i As Integer
Dim st, et As DateTime
Dim sb As StringBuilder

sb = New StringBuilder
st = DateTime.Now
For i = 1 To 100000
   sb.Append("a")
Next
et = DateTime.Now
s = sb.ToString
Dim t as TimeSpan = et – st
Label1.Text = "Zpracoval jsem " & s.Length & " řetězců."
Label2.Text = "Celková doba" & t.TotalSeconds & " s."
```

7.4.2 Druh algoritmu

Prog. 16 – ASP.NET: Iterace

```
Dim str As String = ""
Dim st, et As DateTime
Dim i,x As Integer

For i = 1 To 100
   str &= "a"
Next
st = DateTime.Now
for x=1 to 1000
    iterate(str)
 next
et = DateTime.Now
Dim t As TimeSpan = et – st
Label1.Text = "Reverze řetězce o délce " & str. Length & " znaků. "
Label2.Text = "Celková doba: " & t.TotalSeconds & " s. "

Sub iterate(ByVal str)
   Dim i As Integer
   Dim s As String = ""
   For i = (str.Length) To 1 Step -1
      s = s & Mid(str, i, 1)
   Next
End Sub
```

Prog. 17 – ASP.NET: Rekurze

```
Dim str As String = ""
 Dim st, et As DateTime
 Dim i,x As Integer

 For i = 1 To 100
     str &= "a"
 Next
```

Pokračování

Prog. 17 – Pokračování

```
st = DateTime.Now
 for x=1 to 1000
    recursion(str)
 next
 et = DateTime.Now
 Dim t As TimeSpan = et – st
 Label1.Text = "Reverze řetězce o délce " & str. Length & " znaků. "
 Label2.Text = "Celková doba: " & t.TotalSeconds & " s. "

 Sub recursion(ByVal str)
    Dim s As String = ""
    If (str.ToString.Length > 1) Then
       recursion(Mid(str, 2))
    End If
    s = s & Mid(str, 1, 1)
 End Sub
```

7.4.3 Hrubý výkon

Prog. 18 – ASP.NET: Výpočet Ludolfova čísla

```
Dim st, et As DateTime
Dim pi As Double = 0
Dim pi_const As Double = 3.141592
Dim i As Integer = 1

st = DateTime.Now
Do
    pi = pi + (4 / i - 4 / (i + 2))
    i = i + 4
Loop While (pi <= pi_const)
et = DateTime.Now
Dim t As TimeSpan = et - st
pi = Math.Round(pi, 7)
```

Pokračování

Prog. 18 – Pokračování

```
Dim part() As String = Split(pi.ToString, ",")
Label1.Text = "PI = " & pi_const
Label2.Text = "Výpočet PI na " & part(1).Length & " desetinných
míst."
Label3.Text = "Celková doba: " & t.TotalSeconds & " s.
```

Prog. 19 – ASP.NET: Eratostenovo síto

```
Dim st, et As DateTime
st = DateTime.Now
sito(100000)
et = DateTime.Now
Dim t As TimeSpan = et - st
Label3.Text = "Celková doba: " & t.TotalSeconds & " s."

Sub sito(ByVal intval)
   Dim limit As Integer = 2
   Dim celkem As Integer = 0
   Dim i, k As Integer
   Dim cisla As BitArray
   cisla = New BitArray(intval + 1, True)
   While (limit * limit < intval)
      limit = limit + 1
   End While
   For i = 2 To limit
      If (cisla(i)) Then
         For k = i + i To intval Step +i
            cisla(k) = False
         Next
      End If
   Next
   For i = 2 To intval
      If (cisla(i)) Then
         celkem = celkem + 1
      End If
   Next
End Sub
```

7.4.4 Práce s databázemi

Prog. 20 – ASP.NET: Práce s databází Microsoft SQL Server 2005 Express

```
Dim st, et As DateTime
Dim t As TimeSpan
Dim x As Integer
Dim strConn As String = "Data Source=.\sqlexpress;Initial
                        Catalog=test;Integrated Security=True"
Dim prip As SqlConnection = New SqlConnection(strConn)
Dim test1 As SqlCommand = New SqlCommand("SELECT * FROM
                          [address]", prip)
Dim test2 As SqlCommand = New SqlCommand("SELECT [id],[address1],
                          [address2],[city],[provinceid],[code],
                          [guid],[date] FROM [address]", prip)
Dim test3 As SqlCommand = New SqlCommand("SELECT * FROM [address]
                          WHERE (id>0)", prip)
Dim test4 As SqlCommand = New SqlCommand("SELECT * FROM [address]
                          WHERE (id BETWEEN 1 AND 19615)", prip)
Dim test5 As SqlCommand = New SqlCommand("SELECT count(id) FROM
                          [address] WHERE (id>0) GROUP BY id
                          HAVING (id>0) ORDER BY [id]", prip)
Dim test6 As SqlCommand = New SqlCommand("SELECT [uid],[id],
                          [typeid],[rowguid],[date] FROM [address2]
                          WHERE ([id] IN (SELECT [id] FROM [address]
                          WHERE ([provinceid] > 0)))", prip)
Dim test7 As SqlCommand = New SqlCommand("SELECT * FROM [address2]
                          INNER JOIN [address] ON [address2].[id]
                          = [address].[id]", prip)

prip.Open()

st = DateTime.Now
for x=1 to 100
   test1.ExecuteNonQuery()
next
et = DateTime.Now
t = et - st
Label1.Text = t.TotalSeconds
```

Pokračování

Prog. 20 – Pokračování

```
st = DateTime.Now
for x=1 to 100
    test2.ExecuteNonQuery()
 next
 et = DateTime.Now
 t = et - st
 Label2.Text = t.TotalSeconds


 st = DateTime.Now
 for x=1 to 100
   test3.ExecuteNonQuery()
 next
 et = DateTime.Now
 t = et - st
 Label3.Text = t.TotalSeconds

 st = DateTime.Now
 for x=1 to 100
    test4.ExecuteNonQuery()
 next
 et = DateTime.Now
 t = et - st
 Label4.Text = t.TotalSeconds


 st = DateTime.Now
 for x=1 to 100
    test5.ExecuteNonQuery()
 next
 et = DateTime.Now
 t = et - st
 Label5.Text = t.TotalSeconds


 st = DateTime.Now
 for x=1 to 100
    test6.ExecuteNonQuery()
 next
```

Prog. 20 – Pokračování

```
et = DateTime.Now
t = et - st
Label6.Text = t.TotalSeconds


st = DateTime.Now
for x=1 to 100
   test7.ExecuteNonQuery()
next
et = DateTime.Now
t = et - st
Label7.Text = t.TotalSeconds

prip.Close()
```

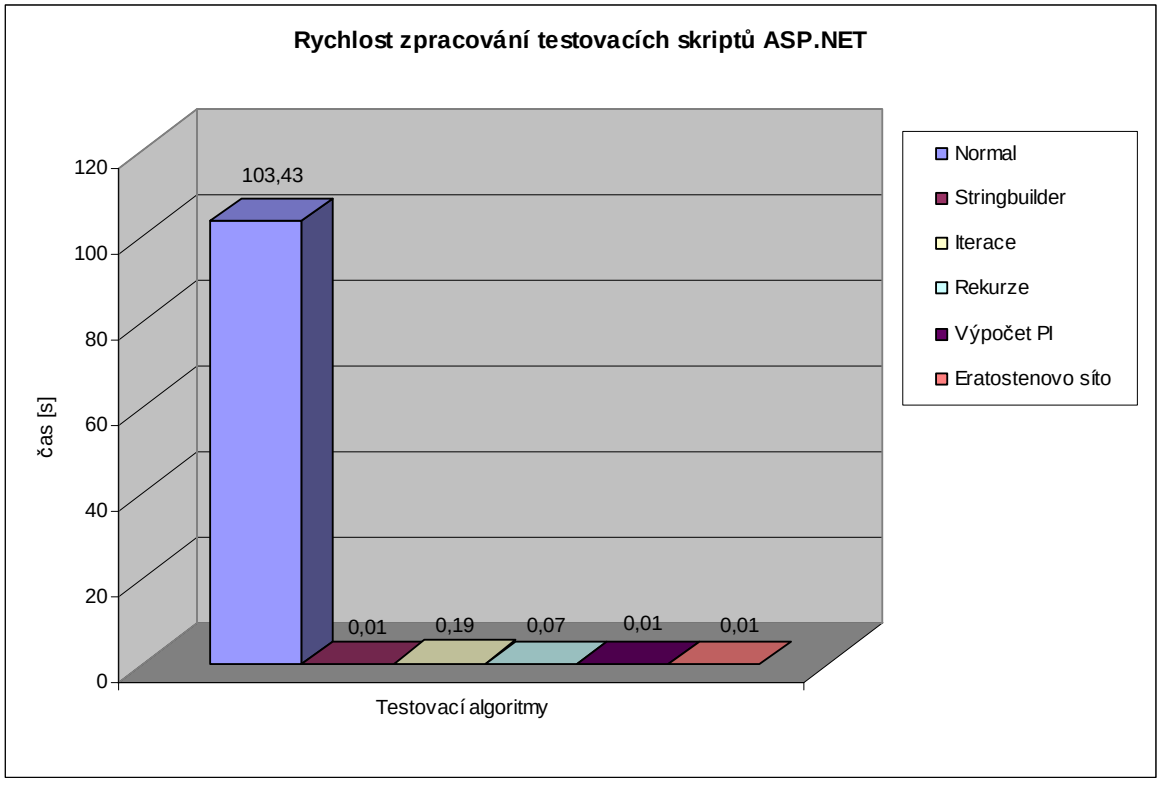
7.5 Výsledky testů

7.5.1 Skripty pro určení výkonu ASP.NET

Tab. 30 – Naměřené hodnoty pro skripty ASP.NET

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Klasická metoda	StringBuilder	Iterace	Rekurze	Výpočet π	Eratostenovo síto
1	104,89063	0	0,35938	0,07813	0,01562	0,17188
2	103,26563	0,01563	0,18750	0,06250	0,03125	0,01563
3	103,21875	0	0,20313	0,06250	0	0,01502
4	103,21875	0,01563	0,17188	0,07812	0	0
5	103,09375	0	0,17525	0,07858	0	0
6	105,48438	0	0,18523	0,07250	0,01563	0,01256
7	103,18750	0	0,19468	0,07985	0,03125	0

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Klasická metoda	StringBuil-der	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
8	103,04688	0	0,18498	0,06540	0	0,01500
9	102,95313	0,01563	0,17187	0,06985	0,01563	0
10	103,51536	0,01563	0,17698	0,07895	0	0,01560
Min	102,95313	0	0,17187	0,06250	0	0
Max	105,48438	0,01563	0,35938	0,07985	0,03125	0,17188
Odchylka	0,82314	0,00766	0,05362	0,00673	0,01220	0,04960
Průměr	103,58748	0,00625	0,20109	0,07264	0,01094	0,02457
Vážený průměr	103,42966	0,00586	0,18495	0,07300	0,00977	0,00923

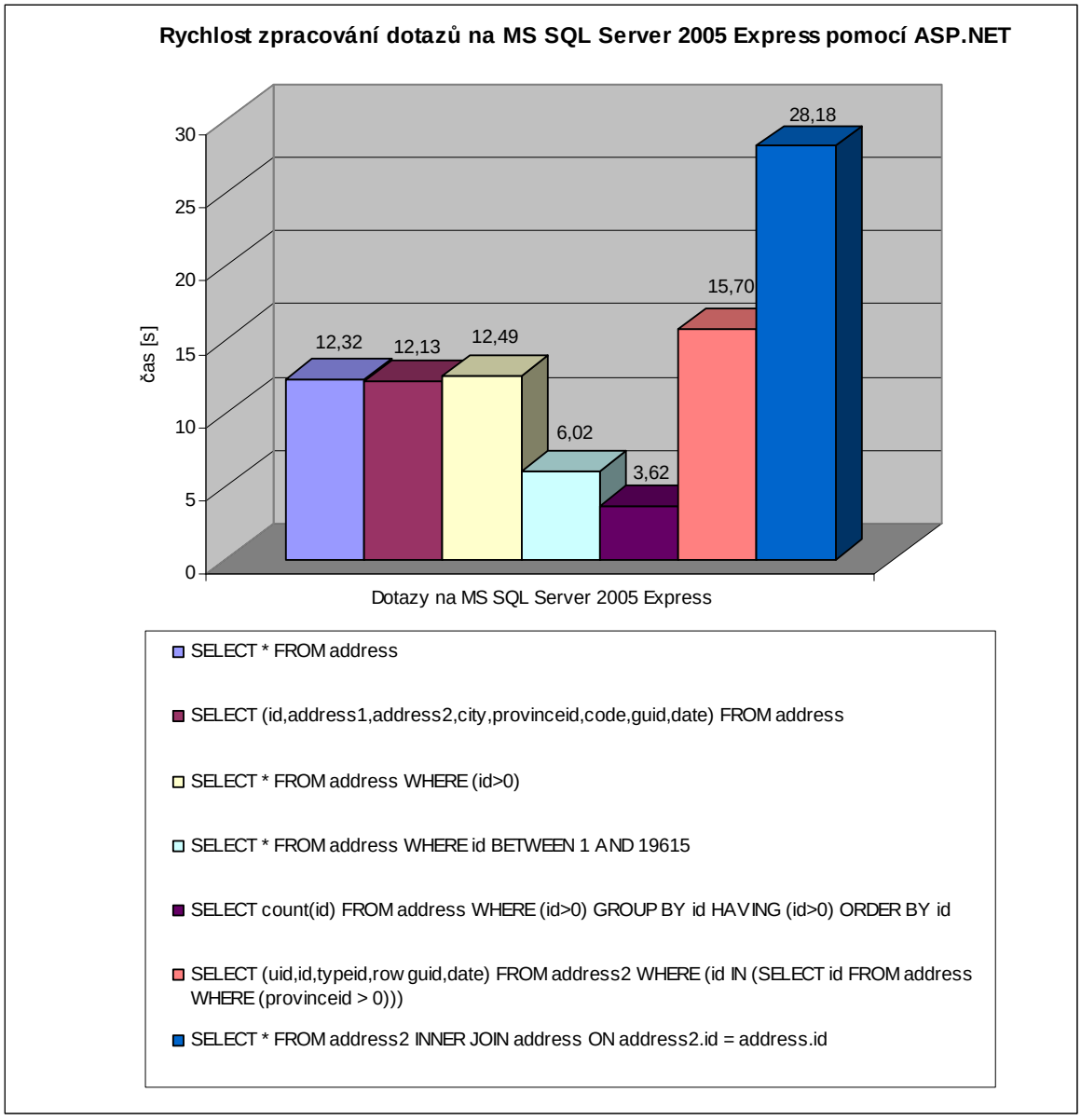


Obr. 40 – Testovací skripty ASP.NET

7.5.2 ASP.NET a databáze Microsoft SQL Server 2005 Express

Tab. 31 – Naměřené hodnoty pro práci ASP.NET s databází Microsoft SQL Server 2005 Express

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	11,81250	11,75000	12,25000	5,89063	3,56250	15,50000	27,70313
2	11,92188	11,84375	12,23438	5,93750	3,60938	15,64063	27,89063
3	12,35938	11,71875	12,20313	5,89063	3,59375	15,65625	27,82813
4	12,20313	12,25000	12,50000	6,03125	3,59375	15,64063	28,01563
5	12,25000	11,96875	12,40625	6,01563	3,67188	15,60938	27,87500
6	12,64063	12,03125	12,42188	5,98438	3,59375	15,70313	27,96875
7	12,51563	12,57813	12,73438	6,17188	6,62500	16,01563	28,54688
8	12,34375	12,95313	12,60938	6,04688	3,68750	15,68750	28,60938
9	12,31250	12,28125	12,75000	6,12500	3,68750	15,84375	28,71875
10	12,93750	12,29688	12,78125	6,12500	3,59375	15,84375	28,92188
Min	11,81250	11,71875	12,20313	5,89063	3,56250	15,50000	27,70313
Max	12,93750	12,95313	12,78125	6,17188	3,68750	16,01563	28,92188
Odchylka	0,30955	0,36890	0,21057	0,09354	0,04239	0,14011	0,41851
Průměr	12,32969	12,16719	12,48906	6,02188	3,62188	15,71409	28,20781
Vážený průměr	12,31836	12,12500	12,48828	6,01953	3,62109	15,70313	28,18164

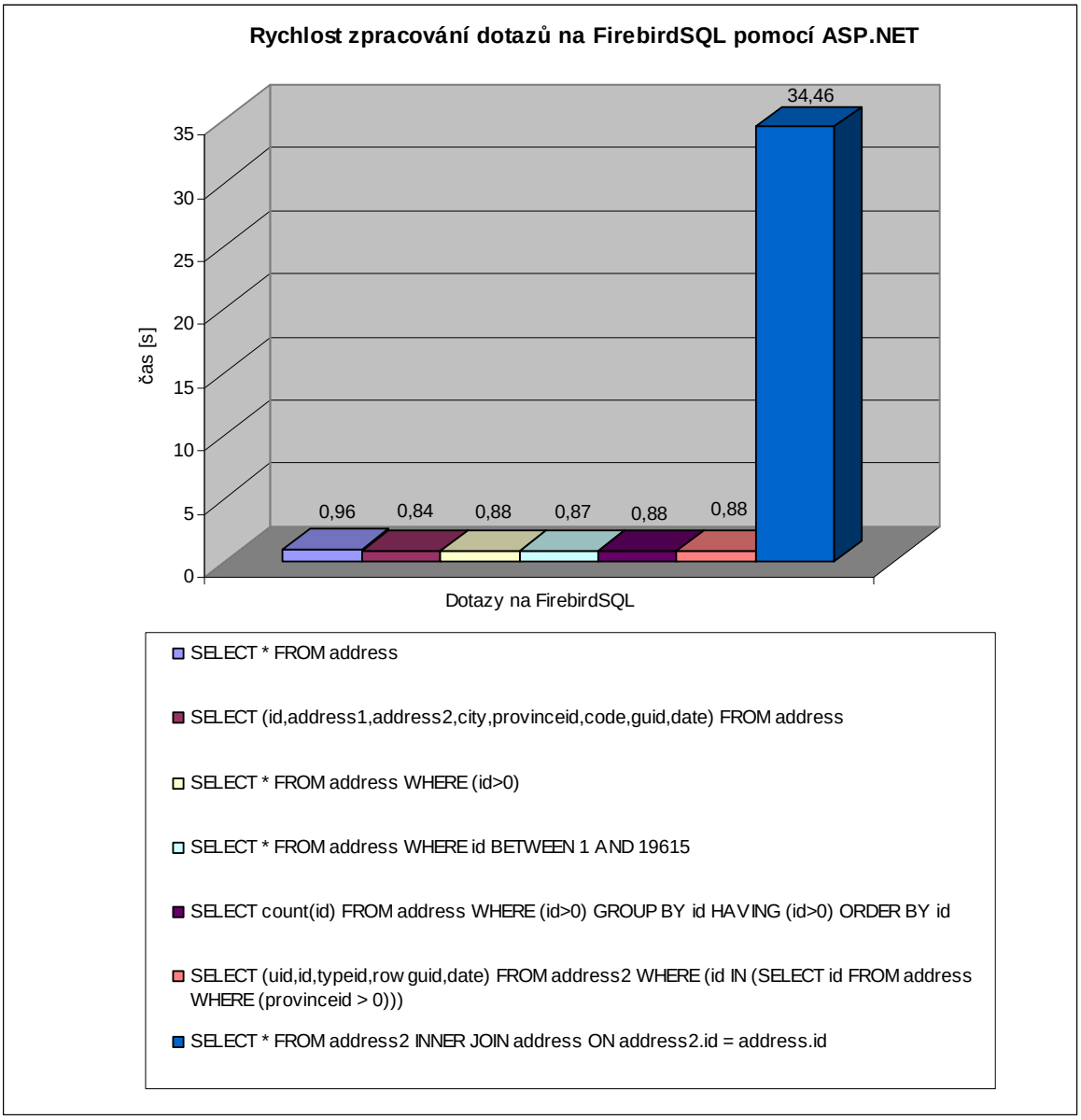


Obr. 41 – ASP.NET a databáze Microsoft SQL Server 2005 Express

7.5.3 ASP.NET a databáze FirebirdSQL

Tab. 32 – Naměřené hodnoty pro práci ASP.NET s databází FirebirdSQL

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,87500	0,76563	0,90625	1	0,90625	0,84375	34,71875
2	0,84375	0,85938	1,46875	0,87500	0,95313	0,90625	34
3	0,95313	0,85938	0,87500	0,87500	0,87500	0,89063	34,73438
4	0,96875	0,93750	0,85938	0,89063	0,84375	0,89063	34,67188
5	1,03125	0,84375	0,89063	0,82813	0,85938	0,89063	34,29688
6	0,90625	0,84375	0,85938	0,85938	0,89063	0,89063	34,28125
7	1,01563	0,81250	0,89063	0,87500	0,89063	0,89063	35,12500
8	0,98438	0,85938	0,84375	0,87500	0,87500	0,85938	34,45313
9	0,95313	0,84375	0,85938	0,87500	0,84375	0,85938	34,35938
10	1,03125	0,82813	0,85938	0,84375	0,85938	0,89063	34,18750
Min	0,84375	0,76563	0,84375	0,82813	0,84375	0,84375	34
Max	1,03125	0,93750	1,46875	1	0,95313	0,90625	35,12500
Odchylka	0,06124	0,04101	0,18009	0,04367	0,03129	0,01875	0,31285
Průměr	0,95625	0,84531	0,93125	0,87969	0,87969	0,88125	34,48281
Vážený průměr	0,96094	0,84375	0,87500	0,87109	0,87500	0,88281	34,46289

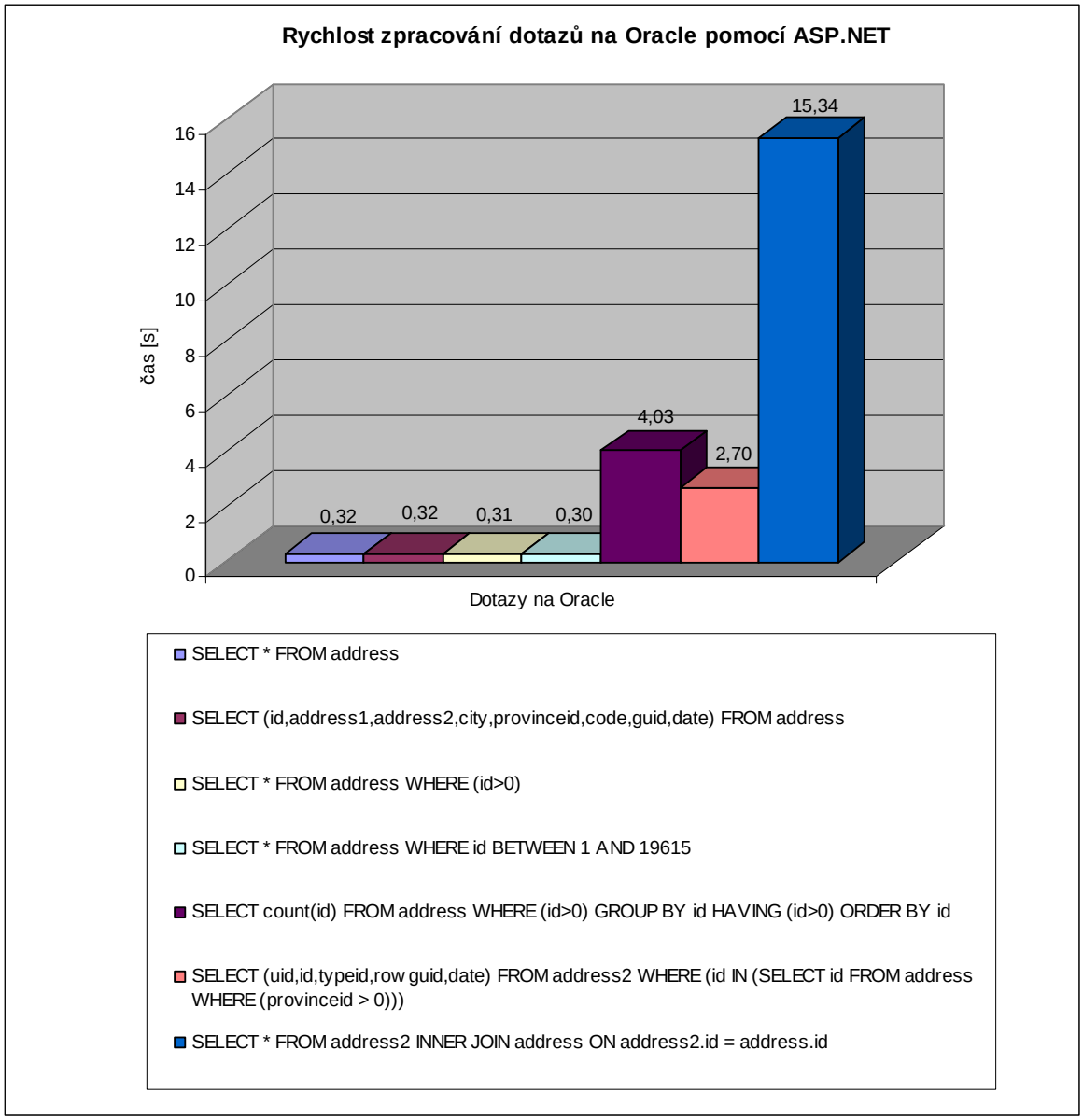


Obr. 42 – ASP.NET a databáze FirebirdSQL

7.5.4 ASP.NET a databáze Oracle

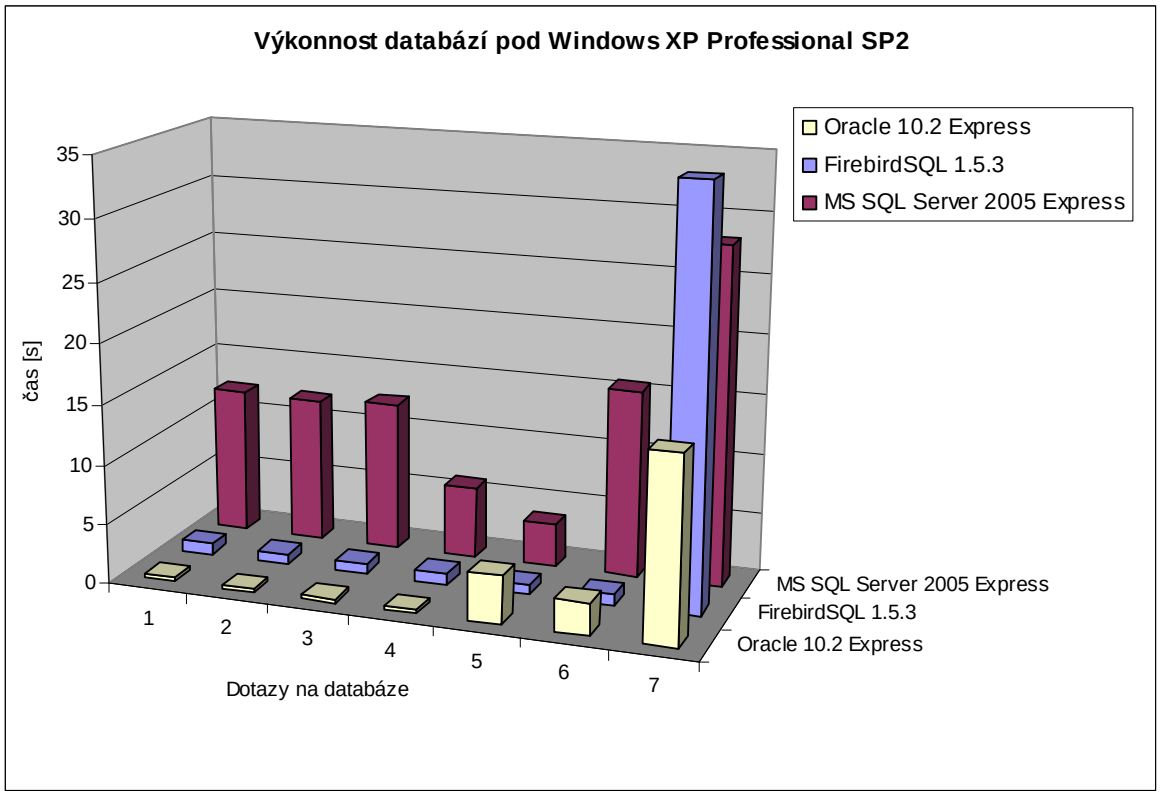
Tab. 33 – Naměřené hodnoty pro práci ASP.NET s databází Oracle

	Dotazy na databázi						
Měření / čas [s]	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0,35938	0,28125	0,26563	0,29688	4,28125	2,70313	17,42188
2	0,28125	0,29688	0,34375	0,37500	4,21875	2,73438	9,28125
3	0,28125	0,32813	0,29688	0,34375	4,10938	2,67188	10,15625
4	0,32813	0,42188	0,34375	0,34375	3,98438	2,71875	12,43750
5	0,31250	0,39063	0,36563	0,29688	4,01563	2,73438	14,84375
6	0,39063	0,32813	0,29688	0,26563	3,96875	2,64063	16,75000
7	0,28125	0,26563	0,23438	0,26563	3,95313	2,68750	17,12500
8	0,35938	0,31250	0,25000	0,29688	4	2,62500	16,98438
9	0,34375	0,32813	0,29688	0,26563	3,98438	2,71875	17,50000
10	0,25000	0,29688	0,34375	0,25000	3,95313	2,68750	17
Min	0,25000	0,26563	0,23438	0,25000	3,95313	2,62500	9,28125
Max	0,39063	0,42188	0,36563	0,37500	4,28125	2,73438	17,50000
Odchylka	0,04262	0,04571	0,04234	0,03940	0,11093	0,03566	3,00605
Průměr	0,31875	0,32500	0,30375	0,30000	4,04688	2,69219	14,95000
Vážený průměr	0,31836	0,32031	0,30469	0,29688	4,02930	2,69531	15,33984



Obr. 43 – ASP.NET a databáze Oracle

7.5.5 Srovnání výkonu databází u ASP.NET



Obr. 44 - Výkonnost přístupu k databázím pomocí ASP.NET

7.6 Mono - alternativní implementace .NET Frameworku

Mono je open source projekt, který umožňuje vývoj a spouštění .NET aplikací pod operačními systémy Linux, Solaris, Mac OS X, Windows a Unix. Mono je v současně době stále ve fázi vývoje a jeho nejnovější verze je 1.1.15.

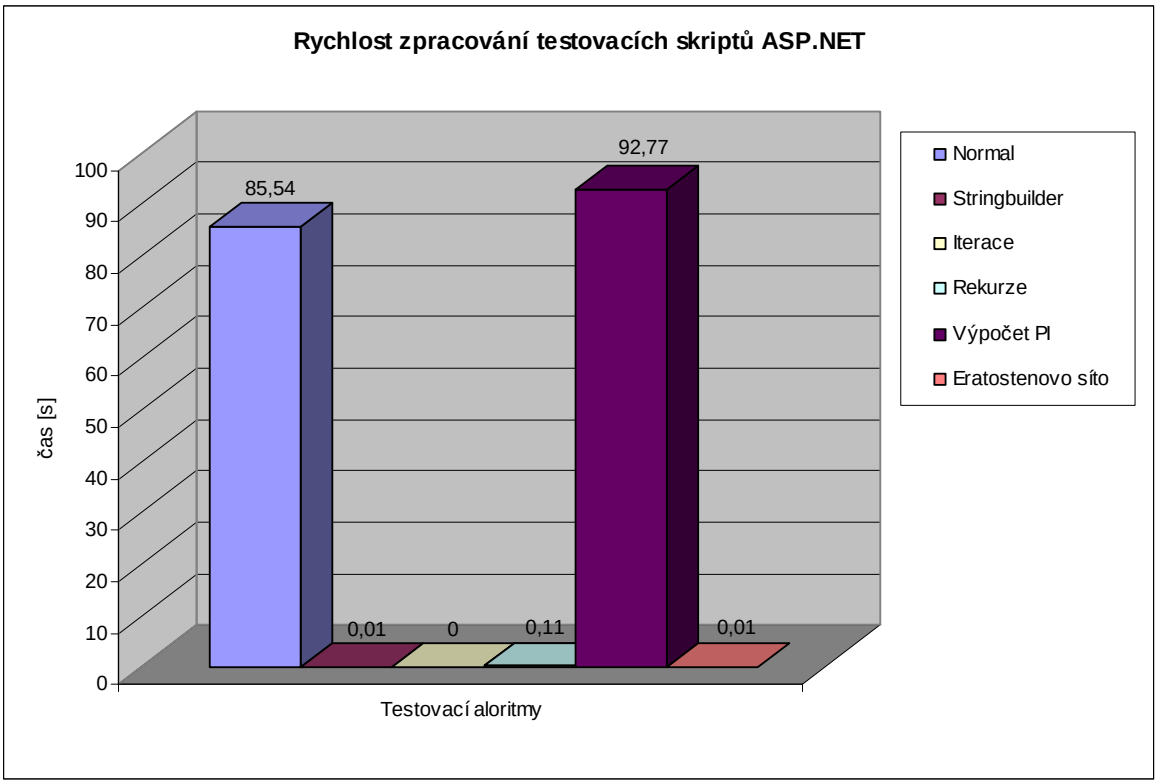
Tento projekt se na první pohled zdá jako velmi přínosný, zejména řešením stěžejního problému, kterým jsou drahé servery Microsoft, spouštěním ASP.NET na serverech Linux a Unix. Bohužel však implementace není vůbec dotažená a dokonce bych řekl, že je až nepoužitelná. Na rozdíl od Microsoft .NET Frameworku je možné používat pouze jeden programovací jazyk, a to C#, dále tu nejsou obsaženy pokročilé funkce, které z Frameworku dělají to, čím vlastně je, tzn. není zde obsažen adaptivní rendering apod. Rovněž připojení k databázím je velmi obtížně realizovatelné, ne-li nemožné (přes všechnu snahu a dostupné návody se mi to nepodařilo).

Zkrátka, Mono ve verzi 1.1.15 je pro nasazení v internetovém provozu nepoužitelné a jde v podstatě pouze o studii v podobě on-line kompilátoru C#. Více informací v odkazu [14].

7.6.1 Výsledky ASP.NET pod Monem

Tab. 34 – Naměřené hodnoty pro ASP.NET pod Monem na Windows XP SP2

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Klasická metoda	**StringBuil-der**	**Iterace**	**Rekurze**	**Výpočet π**	**Eratoste-novo síto**
1	88,09321	0,01686	0	0,11456	93,87579	0,01612
2	85,93816	0,01578	0	0,12545	92,57815	0
3	85,26675	0	0	0,09499	92,82815	0,01601
4	85,31215	0	0	0,09401	92,62501	0
5	84,68712	0,01671	0	0,11450	92,46962	0
6	85,21900	0,01579	0	0,10912	93,46901	0
7	84,79748	0,01678	0	0,12501	92,95319	0,01545
8	85,21979	0,01512	0	0,12545	92,68845	0
9	84,96848	0,01607	0	0,12512	92,45304	0,01588
10	87,60908	0,01644	0	0,09481	92,51559	0
Min	84,68712	0	0	0,09401	92,45304	0
Max	88,09321	0,01686	0	0,12545	93,78579	0,01612
Odchylka	1,12264	0,00650	0	0,01277	0,44845	0,00777
Průměr	85,71112	0,01296	0	0,11230	92,84560	0,00635
Vážený průměr	85,54136	0,01409	0	0,11294	92,76589	0,00592

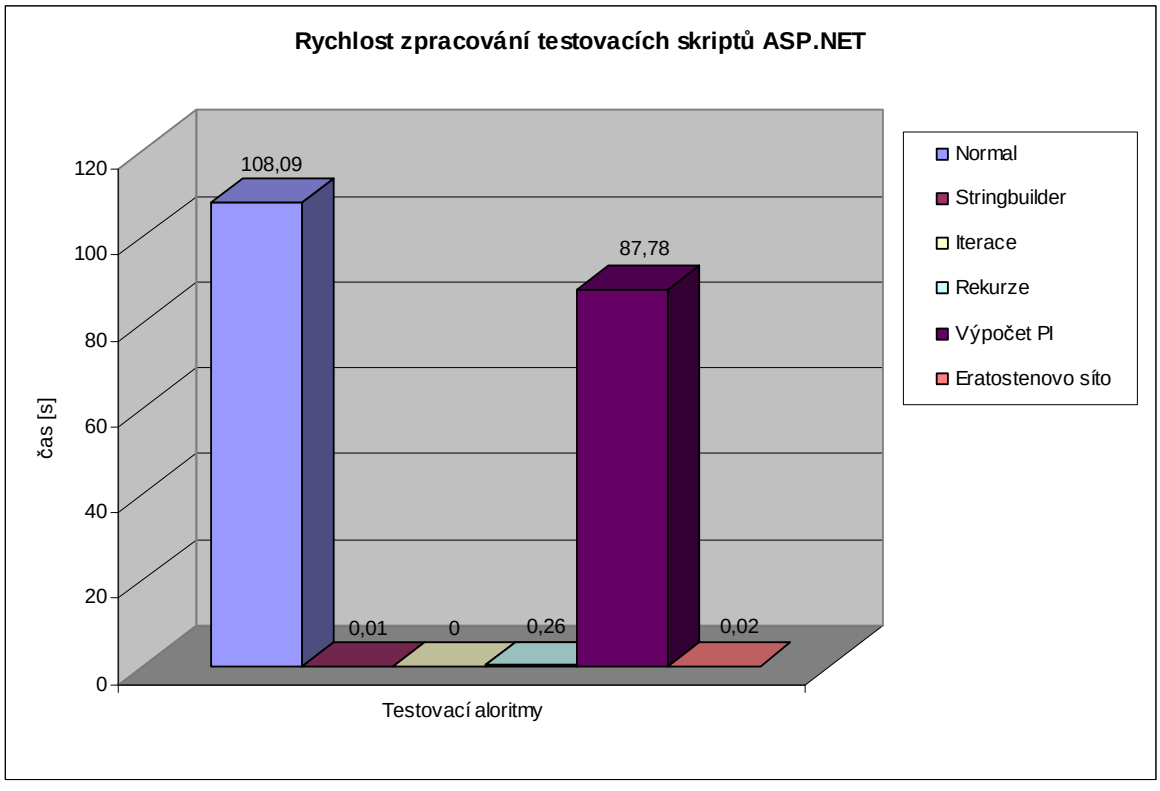


Obr. 45 – Testovací skripty ASP.NET pod Monem na Windows XP SP2

Tab. 35 – Naměřené hodnoty pro ASP.NET pod Monem na Mandriva Linux 2006

Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Klasická metoda	StringBuil-der	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
1	107,22294	0,00831	0,00190	0,60187	80,24052	0,02134
2	102,97134	0,03569	0,00005	0,22810	77,08764	0,01770
3	114,38137	0,02210	0,00005	0,17074	88,69856	0,01969
4	102,40636	0,01061	0,00005	0,36617	88,62929	0,01603
5	109,75532	0,00725	0,00005	0,20555	89,08503	0,18106
6	106,06077	0,00754	0,01008	0,28210	89,57657	0,01634
7	112,16025	0,01379	0,00005	0,31689	89,70183	0,01814

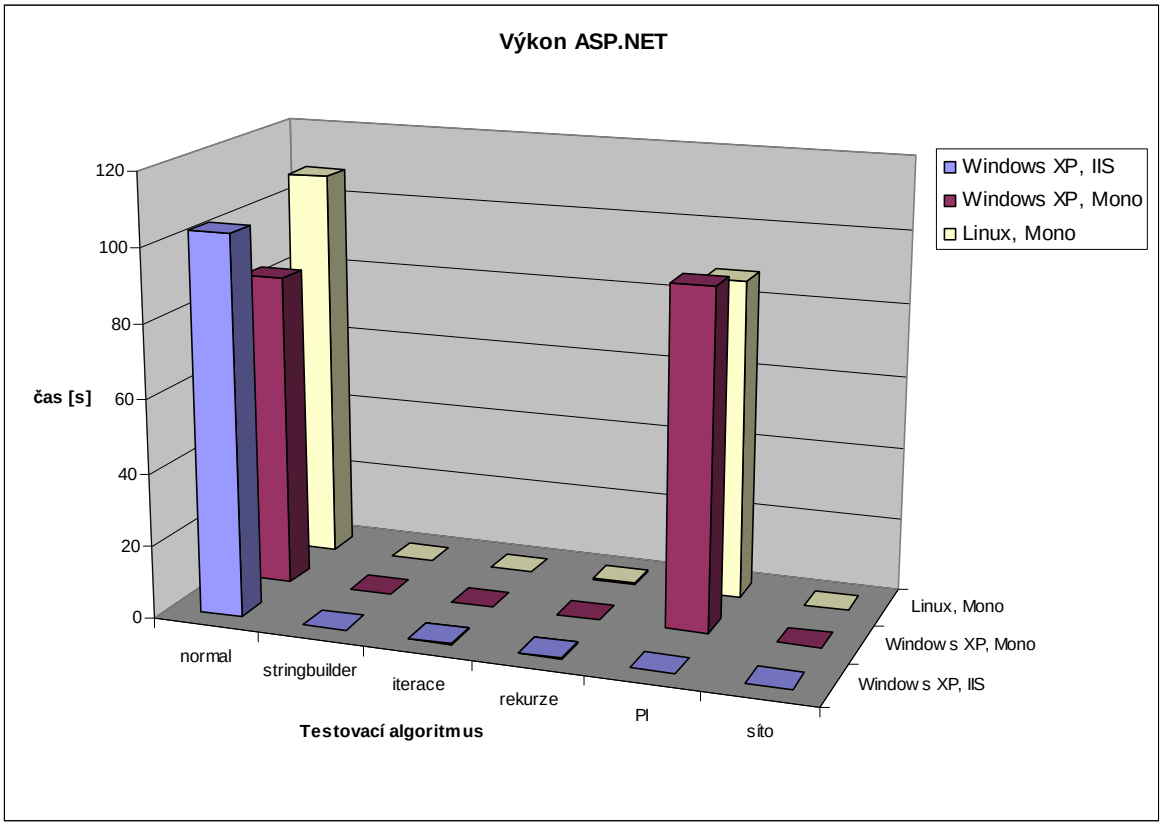
Měření / čas [s]	Práce s řetězci		Druh algoritmu		Hrubý výkon	
	Klasická metoda	StringBuil-der	Iterace	Rekurze	Výpočet π	Eratoste-novo síto
8	109,18827	0,00828	0,01011	0,21975	88,84121	0,00668
9	109,79421	0,02770	0,00005	0,27158	89,10940	0,01847
10	107,55772	0,00793	0,00005	0,16647	88,09877	0,01820
Min	102,40636	0,00725	0,00005	0,16647	77,08764	0,00668
Max	114,38137	0,03569	0,01011	0,60187	89,70183	0,18106
Odchylka	3,55884	0,00957	0,00396	0,12209	4,20394	0,04937
Průměr	108,14985	0,01492	0,00224	0,28292	86,90688	0,03336
Vážený průměr	108,08885	0,01328	0,00153	0,28761	87,78492	0,01824



Obr. 46 - Testovací skripty ASP.NET pod Monem na Mandriva Linux 2006

Tab. 36 – Porovnání skriptů měřících výkon ASP.NET na Monu

Testovací skripty / čas [s]	IIS 5.0	Mono - Windows	Mono - Linux
Klasická metoda	103,42966	85,54136	108,08885
StringBuilder	0,00586	0,01409	0,01328
Iterace	0,18495	0	0,00153
Rekurze	0,07300	0,11294	0,25761
Výpočet π	0,00977	92,76589	87,78492
Eratostenovo síto	0,00923	0,00592	0,01824



Obr. 47 – Výkon ASP.NET na Monu

8 VYHODNOCENÍ

Nejprve bude provedeno vyhodnocení PHP, neboť tato skriptovací technologie byla testována na více systémových platformách v normální i kompilované podobě. Po nalezení nejvýkonnějšího řešení PHP, bude toto zahrnuto do porovnání s ostatními technologiemi. Při vyhodnocování bude kromě výsledků měření brán zřetel i na možnosti, které nabízejí poskytovatelé webhostingů a další faktory ovlivňující volbu, jakou technologii použít, byť nemusí přímo souviset s výkonem.

8.1 Vyhodnocení PHP

8.1.1 Zpracování skriptů

V této oblasti testování výkonu se jeví jako nejvýkonnější kompilované PHP běžící na webovém serveru Apache pod operačním systémem Microsoft Windows XP Professional SP2 a hned za ním klasické, interpretované PHP na stejné konfiguraci. I když provozování PHP pod operačním systémem Windows není tak neobvyklé, jako ASP.NET pod Linuxem, variantu PHP s Windows poskytovatelé webhostingu prakticky nenabízejí. Budeme muset tedy od obou, nejvýkonnějších řešení PHP, opustit.

Interpretované i kompilované PHP pod Linuxem vycházejí téměř stejně. Práce s řetězci pomocí globální proměnné u interpretovaného PHP je však o 76% rychlejší než u kompilovaného. Naproti tomu práce s řetězci pomocí lokální proměnné je o 5,2% rychlejší u kompilovaného PHP. Jasnou volbou je tedy interpretované PHP, a to nejen díky rychlosti, ale také kvůli malé podpoře kompilovaného PHP u webhostingových služeb.

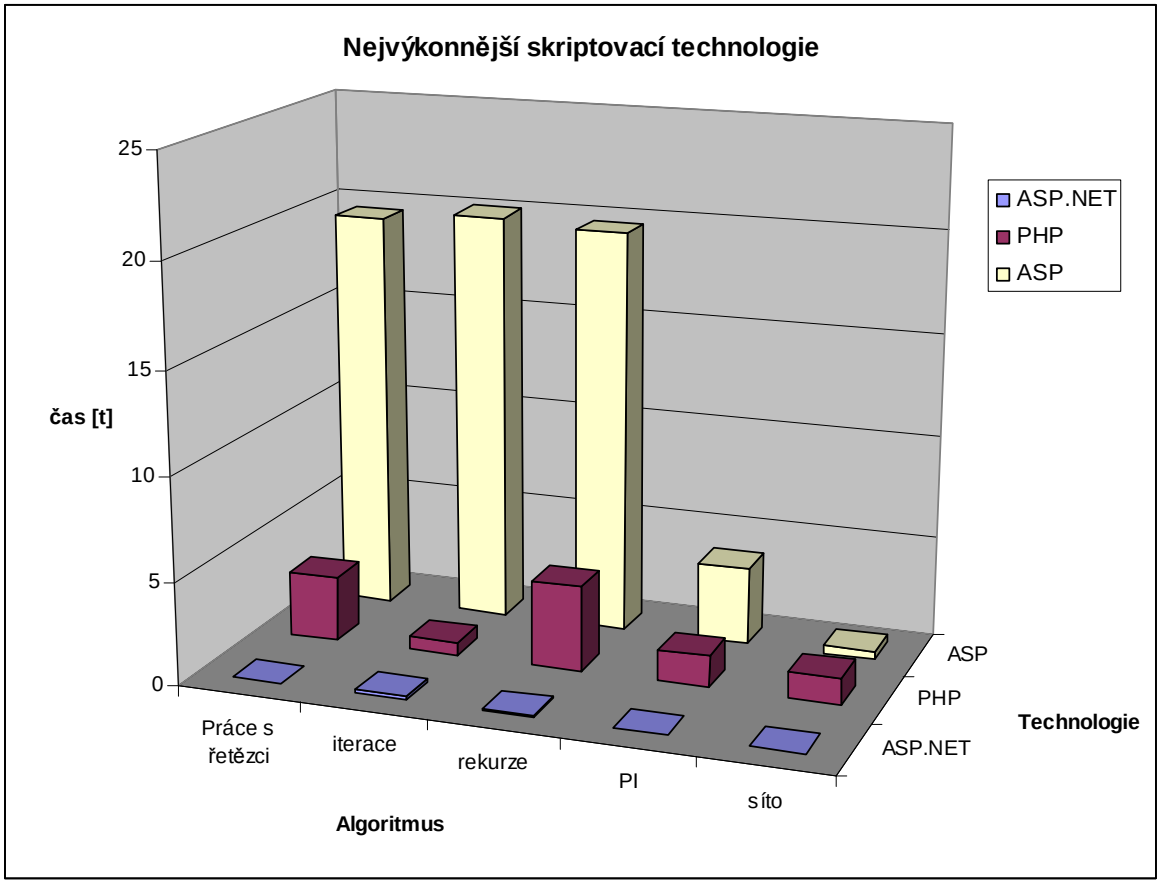
8.1.2 Databáze

Výkon práce s databázemi vychází nejlépe pro FirebirdSQL a Oracle, které jsou velmi rychlé. Naneštěstí se v nabídkách webhostingových služeb téměř nenabízejí a tedy takový, který by tyto databáze nabízel, je nutné zdlouhavě hledat, což není zrovna nejzábavnější. S dostupností je na tom mnohem lépe MySQL, které ani výkonově není nejhorší, takže se jeví jako rozumná volba. PostgreSQL je na tom nejhůře, dosupnost také není nejlepší a tak není téměř žádný důvod jej preferovat.

8.1.3 Řešení PHP

PHP pod operačním systémem Linux s databází MySQL.

8.2 Celkové vyhodnocení



Obr. 48 – Celkové vyhodnocení

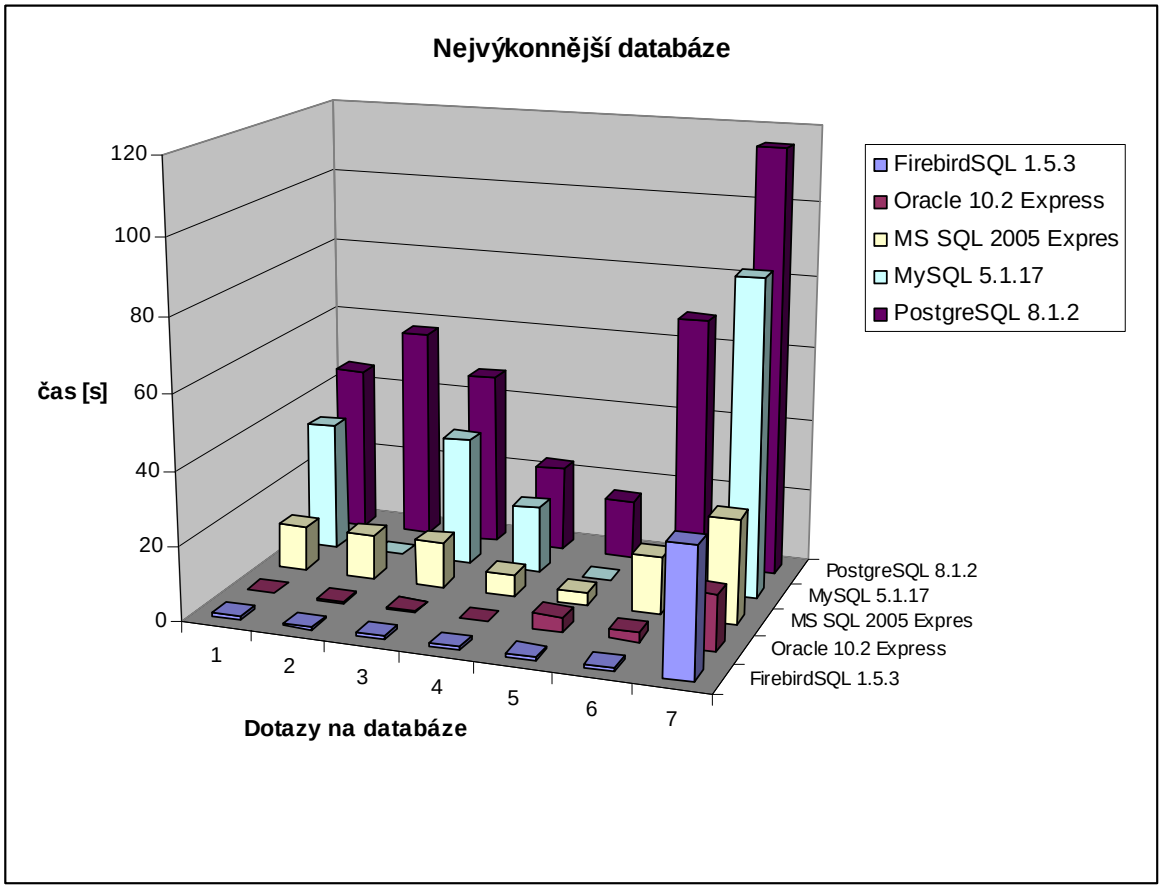
Podle výsledků měření je výrazně nejpomalejší skriptovací technologie ASP. Je to už poměrně stará technologie, která je konkurentem PHP. Webhostingové služby, nabízející podporu ASP, podporují i velmi rychlé ASP.NET a pokud tedy bude někdo chtít využívat technologie Microsoftu, není důvod, proč se zaměřovat na ASP.

Mezi nejvýkonnějšími řešeními jsou dvě zcela různé technologie. PHP a ASP.NET. Je zcela zřejmé, že v případě hodnocení pouze podle výsledků měření – podle výkonu, je jasným vítězem ASP.NET.

8.2.1 Nejvýkonnější technologie pro vývoj webových aplikací

ASP.NET postavený na technologii Microsoft .NET Framework 2.0

8.2.2 Databáze



Obr. 49 – Celkové vyhodnocení databází

Výsledky rychlostí stejných databází u různých skriptovacích technologií byly téměř stejné, což je naprosto v pořádku, neboť výkonnost databází až tak nezávisí na použité skriptovací technologii. Díky této skutečnosti je možné porovnat výkon různých databází dohromady.

Nejvýkonnější je databáze FirebirdSQL a Oracle, kde zejména Oracle je velmi robustním databázovým enginem. Bohužel je však dostupnost těchto databází jak u poskytovatelů webhostingových služeb s ASP, ASP.NET tak i s PHP velmi malá, což je velká škoda.

Velmi dobré výsledky má i Microsoft SQL Server, který je rovněž velmi dobře dostupný a v kombinaci s ASP a ASP.NET se používá téměř vždy.

MySQL je velmi oblíbený databázový server. Svým výkonem poněkud zaostává za svými konkurenty, avšak jeho téměř 100% dostupnost z něj dělá v jedinečné kombinaci s PHP kvalitní zázemí pro webové aplikace postavené právě na PHP.

PostgreSQL nevyšlo z databázových testů nejlépe. Ve všech dotazech je nejpomalejší, v některých případech (dotaz [6]) to je až o 99.5%. Rovněž není příliš dostupné a vždy existuje lepší alternativa než je PostgreSQL.

Jasným vítězem mezi databázemi je FirebirdSQL, avšak na Internetu se téměř vždy využívá kombinací ASP.NET + MS SQL a PHP + MySQL, přestože se obě skriptovací technologie dokáží připojit i k jiným databázím, je nutné stanovit vítěze, který má praktický význam a tím ale FirebirdSQL není.

8.2.3 Nejvýkonnější databázový server

FirebirdSQL

8.2.4 Nejvýkonnější dostupné efektivní řešení

ASP.NET s databází Microsoft SQL Server 2005 Express

8.3 Proč zvolit ASP.NET

ASP.NET je moderní, elegantní technologie, která programátorům velmi usnadňuje práci. Vývoj ASP.NET stránek je hodně podobný psaní stolních aplikací. Programátor se může soustředit více na obsah, než na způsob jakým docílit svého záměru. Velkým kladem je, že je ASP.NET založeno na .NET Frameworku. Díky tomu, je možné pří vytváření webových aplikací využívat stejných prostředků a komponent, jako při vytváření WinForm aplikací. Kompilace umožňuje nejen zvýšení rychlosti, ale také odhalení všech chyb už při kompilaci samotné. Dále ASP.NET dokáže generovat validní stránky, což je velká výhoda. Nelze opomenout ani výbornou aplikaci pro vývoj ASP.NET stránek, a to Microsoft Visual Web Developer Express, která je k dispozici zdarma, a mimo jiné obsahuje i vlastnost IntelliSense, která samostatně doplňuje psaný kód, dává na výběr vlastnosti atributů podle použité funkce, apod.

Asi jedinou nevýhodou je vysoká cena webhostingových služeb. Je to z důvodu požadavků na velký výkon hardware serverů a ceny serverových operačních systémů Microsoftu. Cena

webhostingů s ASP.NET je tak jednou i dvakrát vyšší než u webostingů s PHP. Samozřejmě, je možné nalézt na Internetu i free webhostingy, je jich ale málo a poskytují velmi malé množství místa na discích.

8.3.1 Přehled free webhostingů pro ASP.NET

- <http://www.aspweb.cz>
- <http://www.asp2.cz>
- <http://www.qsh.sk>
- <http://www.aspx.sk>

8.4 Proč zvolit PHP

To, co je u ASP.NET velká nevýhoda, je u PHP velká výhoda. A myslím, že je to i jediná velká výhoda PHP. Jde o cenu webhostingů, kdy se cena placených služeb pohybuje dokonce u 20 až 30 Kč. za měsíc. Spousta poskytovatelů nabízí webhosting ve variantě PHP dokonce zdarma, a to s webovým místem až 1GB, příp. i více. Jinak přestože existuje spousta editorů pro psaní PHP, ať už např. robustní Zend Studio, nebo třeba obyčejné editory podobné Poznámkovému bloku, PHP bude stále interpretovaným skriptovacím jazykem, ve kterém, co si nenapíšete ručně, znak od znaku, to nemáte. Je tak nutné řešit otázky jakým způsobem se připojit k databázi, jakým způsobem přenášet proměnné, je nutné ručně vytvářet a rušit objekty, apod. Přesto je PHP velmi oblíbené a je to jedna z nejrozšířenějších, i když už také poměrně starých, technologií.

8.4.1 Přehled free webhostingů pro PHP

- <http://www.webzdarma.cz>
- <http://www.ic.cz>
- <http://www.php5.cz>
- <http://www.php5.sk>

- <http://www.pipni.cz>
- <http://www.forpsi.com>
- <http://www.fbi.cz>
- <http://www.xnet.cz>

a spousta dalších

9 TECHNIKY OPTIMALIZACE

Ať už se rozhodnete pro ASP.NET nebo PHP, je vždy dobré, ještě před napsáním prvního skriptu webové aplikace, prostudovat možnosti optimalizace webové technologie. Je tak možné dosáhnout částečně vyššího výkonu doslova zadarmo, tj. jen za cenu chvíle strávené čtením pár řádek, které můžete nalézt níže. I když jsou tyto rady psané zvlášť pro ASP.NET i pro PHP, mnoho doporučení je platných pro obě technologie. Je tak výhodné si přečíst celou tuto kapitolu.

9.1 Optimalizace ASP.NET

9.1.1 Šířka pásma

V současné době je sice vysokorychlostní připojení k Internetu již téměř standardem, nicméně při psaní aplikace je vždycky lepší myslet i na klienty s pomalejším připojením, neboť také tito mohou být „zákazníky“ společnosti, pro kterou bude aplikace napsána.

Toho je možné docílit např. minimalizováním používání, resp. vypnutím, stavových proměnných (ViewStates), validací na klientské straně, HTTP 1.1 kompresí. Je doporučeno používat co nejméně a co nejmenších cookies. Je vhodné si odpovědět na otázku, proč neustále načítat statické stránky, když mohou být kešované, a tím i mnohem rychleji přístupné?

Podobných doporučení jsou spousty, stačí se jen zamyslet nebo se podívat na Internetu.

9.1.2 Přístup k databázím

Při přístupu k databázi se dotazujte pouze na potřebná data. Je zbytečné používat *select * from*, když potřebujete pouze data z jednoho sloupce. V některých případech není nutné, aby dotaz vracel výsledky. Tady je možné s výhodou využít metodu *ExecuteNonQuery*. Velké soubory, jako obrázky a jiné (typ BLOB) nepatří do databáze, ale na disk. V databázi může být umístěn pouze odkaz na takovýto soubor. Dále je potřeba se důsledně věnovat otevírání a zavírání připojení k databázi (Connection pooling; *Connection.Open, Connection.Close*). Na závěr musím zdůraznit používání indexů a jejich správnou volbu.

9.1.3 Vhodné používání Cache

Cache je dobrý sluha, ale zlý pán. Pokud se dobře použije, umožní velké urychlení aplikace. Naopak, při špatném použití nebude k užitku a webová aplikace se rozroste o dalších několik, ne-li desítek, řádků a kód se znepřehlední.

Je zcela nevhodné kešovat data závislá na uživateli, data, která se často mění, zdroje alokující mnoho systémových prostředků a data, který musí být synchronizována přes servery.

Do keše patří zpracované data, nikoliv zdrojová a je důležité správně zvolit expiraci dat.

9.1.4 „Správné“ programování

I při samotném programování je možné částečně urychlit nebo zpomalit aplikaci. Pro plynulý běh aplikace a pro její urychlení je dobré např. neřídit běh programu pomocí výjimek, nezapomínat na uvolňování zdrojů, místo *On Error Goto* používat raději *Try/Catch/Finally*, sčítat řetězce zásadně přes StringBuilder, využívat vestavěné funkce (např. pro porovnání řetězců využít *str.Compare(str,str2,false)* a nikoliv *if str == str2*), vyhnout se vícedimensionálním polím, apod.

9.2 Optimalizace PHP

9.2.1 Předgenerovaný obsah

Při použití předgenerovaného obsahu je možné se zbavit neustálého opakovaného interpretování celé stránky, které je v mnoha případech zbytečné. PHP je tedy možné použít jen pro vygenerování statické stránky a následně zobrazovat tuto statickou stránku. Statický kód je generován pravidelně jednou za nějaký časový interval. Záleží na aktuálnosti dat, která jsou zpracovávána na stránkách a na velikosti webu. U jedné stránky, která je aktualizována velmi často může být generování prováděno např. jednou za minutu, u rozsáhlých firemních webů to může být třeba i jednou za hodinu.

9.2.2 Konfigurace PHP

Samotné nastavení PHP v konfiguračním souboru *php.ini* může mít vliv na výkon této skriptovací technologie. Jaká nastavení však použít?

Jsou to:

- zakázání *register_globals* (i když se to některým nebude zrovna líbit)
- zakázání *magic_quotes*
- vypnutí *expose_php, register_argc_argv* pro skripty volané z webu
- nejlépe nezapínání *always_populate_raw_post_data*.

9.2.3 Otevírání souborů

Většina skriptů provádí souborové IO operace. Jsou to nejen čtení a zápis do souboru, ale i *include*, *require* a další. Tyto operace je možné urychlit používáním plné cesty k souboru (tedy např. *"include "/cesta/k/souboru.php";* nebo *include "./soubor.php").*

9.2.4 Práce se sessions

Optimalizace sessions se skládá z:

- nepoužívání *session.auto_start*
- zakázání *session.use_trans.sid*
- nastavení *session.cache_limiter* na *private_no_expire*
- použitím vlastního adresáře pro každého uživatele (virtualhost)
- „ruční" uzavírání sessions.

ZÁVĚR

Při psaní této práce jsem se snažil co nejobjektivněji posoudit klady i zápory jednotlivých technologií, co nejlépe sestavit testovací algoritmy a prověřit možnosti databází. To se myslím podařilo a přesto není možné říct: „používejte pouze ASP.NET“, nebo „používejte pouze PHP“. Proto jsem také do celkového vyhodnocení zahrnul i další faktory, a to zejména faktor finančních nákladů na provoz webových aplikací pod různými technologiemi a faktor dostupnosti u poskytovatelů webhostingů.

Důležitost faktoru finančních nákladů je různá a závisí na cílové skupině zákazníků, tedy těch, co chtějí mít svou vlastní webovou aplikaci. Pro velké společnosti je tento faktor zanedbatelný, či dokonce, nehraje žádnou roli, naopak pro malé společnosti, organizace nebo jednotlivce, má větší význam než výkon jednotlivých skriptovacích technologií. Je tedy zřejmé, že malá firma okamžitě sáhne po PHP, kdežto větší společnost bez rozmýšlení zvolí nejvýkonnější ASP.NET i za cenu vyšších nákladů.

Faktor dostupnosti se projevil v plné výši u databázových serverů, jejichž nabídka je u poskytovatelů webových služeb velmi omezená. Samozřejmě je možné nalézt i výjimky, je ale nutné dlouze hledat, a to ve velké míře na zahraničních serverech. Zde se však opět začíná objevovat finanční faktor.

Na samotný závěr bych rád uvedl mé velké přání, které se týká webových aplikací a skriptovacích technologií: přál bych si, aby se do budoucna zlepšila dostupnost technologií a ceny webhostingů se pro ASP.NET i PHP dostaly na stejnou úroveň, a to nejen z důvodu, že až potom by výsledky zjištěné touto prací nabraly ten pravý význam a přestaly být snižovány omezujícími faktory.

SEZNAM POUŽITÉ LITERATURY

[1] WELLING, L., THOMSON, L. *PHP a MySQL Rozvoj webových aplikací*. 3. vydání. Praha: Softpress. Vydáno 2005. ISBN 80-8649-783-6.

[2] PAYNE, Ch. *Naučte se ASP.NET za 21dní*. Praha: ComputerPress. Vydáno 2002. ISBN 80-7226-605-5.

[3] BLÁHA, M. *Optimalizace výkonu .NET aplikací* [online]. 2006, [cit. 2006-04-29]. <http://blog.vyvojar.cz/michal>

[4] KRYL, M. *PHP optimalizace* [online]. 2006, [cit. 2006-04-29].

< http://kryl.info/clanek/120-php-optimalizace>

[5] *PC svět: Úvod do ASP.NET* [online]. 2006, [cit. 2006-04-29].

<http://www.pcsvet.cz/art/article.php?id=3632>, ISSN 1213-6042

[6] *WebTip: ASP pro začátečníky* [online]. 2006, [cit. 2006-04-29].

<http://www.webtip.cz/art/wt_tech_asp/serial_asp_1.html>

[7] *PHP Manual* [online]. 2006, [cit. 2006-04-29].

<http://www.php.net/docs.php>

[8] *Wikipedie: ASP.NET* [online]. 2006, [cit. 2006-04-29].

<http://cs.wikipedia.org/wiki/ASP.NET>

[9] *Microsoft MSDN Library* [online]. 2006, [cit. 2006-04-29].

<http://msdn.microsoft.com>

SEZNAM OBRÁZKŮ

Obr. 1 – Tabulka "address1", část 1 19
Obr. 2 – Tabulka "address1", část 2 19
Obr. 3 – Tabulka "address2" 20
Obr. 4 – Testovací skripty ASP 28
Obr. 5 – ASP a databáze Microsoft SQL Server 2005 Express 30
Obr. 6 - ASP a databáze FirebirdSQL 32
Obr. 7 - ASP a databáze Oracle 34
Obr. 8 - Výkonnost přístupu k databázím pomocí ASP 35
Obr. 9 – Testovací skripty PHP 43
Obr. 10 – Testovací kompilované skripty PHP 44
Obr. 11 - PHP a databáze MySQL 46
Obr. 12 - Kompilované PHP a databáze MySQL 48
Obr. 13 - PHP a databáze PostgreSQL 50
Obr. 14 - Kompilované PHP a databáze PostgreSQL 52
Obr. 15 - PHP a databáze FirebirdSQL 54
Obr. 16 - Kompilované PHP a databáze FirebirdSQL 56
Obr. 17 - PHP a databáze Oracle 58
Obr. 18 - Kompilované PHP a databáze Oracle 60
Obr. 19 - Výkonnost přístupu k databázím pomocí PHP pod Linuxem 61
Obr. 20 - Výkonnost přístupu k databázím pomocí kompilovaného PHP pod Linuxem 61
Obr. 21 – Testovací skripty PHP 63
Obr. 22 – Testovací kompilované skripty PHP 64
Obr. 23 – PHP a databáze MySQL 66
Obr. 24 – Kompilované PHP a databáze MySQL 68
Obr. 25 – PHP a databáze PostgreSQL 70
Obr. 26 – Kompilované PHP a databáze PostgreSQL 72
Obr. 27 – PHP a databáze FirebirdSQL 74
Obr. 28 – Kompilované PHP a databáze FirebirdSQL 76
Obr. 29 – PHP a databáze Oracle 78
Obr. 30 – Kompilované PHP a databáze Oracle 80
Obr. 31 - Výkonnost přístupu k databázím pomocí PHP pod Windows 81

Obr. 32 - Výkonnost přístupu k databázím pomocí kompilovaného PHP pod Windows 81
Obr. 33 – Výsledky měření výkonu PHP 82
Obr. 34 – Výkon databáze MySQL 83
Obr. 35 – Výkon databáze PostgreSQL 84
Obr. 36 – Výkon databáze FirebirdSQL 85
Obr. 37 – Výkon databáze Oracle 86
Obr. 38 – Výkon databází pod PHP 87
Obr. 39 – Výkon databází pod kompilovaným PHP 87
Obr. 40 – Testovací skripty ASP.NET 97
Obr. 41 – ASP.NET a databáze Microsoft SQL Server 2005 Express 99
Obr. 42 – ASP.NET a databáze FirebirdSQL 101
Obr. 43 – ASP.NET a databáze Oracle 103
Obr. 44 - Výkonnost přístupu k databázím pomocí ASP.NET 104
Obr. 45 – Testovací skripty ASP.NET pod Monem na Windows XP SP2 106
Obr. 46 - Testovací skripty ASP.NET pod Monem na Mandriva Linux 2006 107
Obr. 47 – Výkon ASP.NET na Monu 108
Obr. 48 – Celkové vyhodnocení 110
Obr. 49 – Celkové vyhodnocení databází 111

SEZNAM TABULEK

Tab. 1 – Naměřené hodnoty pro skripty ASP 27
Tab. 2 – Naměřené hodnoty pro práci ASP s databází Microsoft SQL Server 2005 Express 29
Tab. 3 - Naměřené hodnoty pro práci ASP s databází FirebirdSQL 31
Tab. 4 - Naměřené hodnoty pro práci ASP s databází Oracle 33
Tab. 5 – Naměřené hodnoty pro skripty PHP 42
Tab. 6 – Naměřené hodnoty pro kompilované skripty PHP 43
Tab. 7 - Naměřené hodnoty pro práci PHP s databází MySQL 45
Tab. 8 - Naměřené hodnoty pro práci kompilovaného PHP s databází MySQL 47
Tab. 9 - Naměřené hodnoty pro práci PHP s databází PostgreSQL 49
Tab. 10 – Naměřené hodnoty pro práci kompilovaného PHP s databází PostgreSQL 51
Tab. 11 - Naměřené hodnoty pro práci PHP s databází FirebirdSQL 53
Tab. 12 – Naměřené hodnoty pro práci kompilovaného PHP s databází FirebirdSQL 55
Tab. 13 – Naměřené hodnoty pro práci PHP s databází Oracle 57
Tab. 14 – Naměřené hodnoty pro práci kompilovaného PHP s databází Oracle 59
Tab. 15 – Naměřené hodnoty pro skripty PHP 62
Tab. 16 – Naměřené hodnoty pro kompilované skripty PHP 63
Tab. 17 - Naměřené hodnoty pro práci PHP s databází MySQL 65
Tab. 18 – Naměřené hodnoty pro práci kompilovaného PHP s databází MySQL 67
Tab. 19 – Naměřené hodnoty pro práci PHP s databází PostgreSQL 69
Tab. 20 – Naměřené hodnoty pro práci kompilovaného PHP s databází PostgreSQL 71
Tab. 21 - Naměřené hodnoty pro práci PHP s databází FirebirdSQL 73
Tab. 22 – Naměřené hodnoty pro práci kompilovaného PHP s databází FirebirdSQL 75
Tab. 23 – Naměřené hodnoty pro práci PHP s databází Oracle 77
Tab. 24 – Naměřené hodnoty pro práci kompilovaného PHP s databází Oracle 79
Tab. 25 – Porovnání výsledků skriptů měřících výkon PHP 82
Tab. 26 – Porovnání výkonu spolupráce PHP a MySQL 83
Tab. 27 – Porovnání výkonu spolupráce PHP a PostgreSQL 84
Tab. 28 – Porovnání výkonu spolupráce PHP a FirebirdSQL 85
Tab. 29 – Porovnání výkonu spolupráce PHP a Oracle 86
Tab. 30 – Naměřené hodnoty pro skripty ASP.NET 96

Tab. 31 – Naměřené hodnoty pro práci ASP.NET s databází Microsoft SQL Server 2005 Express 98
Tab. 32 – Naměřené hodnoty pro práci ASP.NET s databází FirebirdSQL 100
Tab. 33 – Naměřené hodnoty pro práci ASP.NET s databází Oracle 102
Tab. 34 – Naměřené hodnoty pro ASP.NET pod Monem na Windows XP SP2 105
Tab. 35 – Naměřené hodnoty pro ASP.NET pod Monem na Mandriva Linux 2006 106
Tab. 36 – Porovnání skriptů měřících výkon ASP.NET na Monu 108

SEZNAM VÝPISŮ ZDOJOVÝCH KÓDŮ

Prog. 1 – ASP: Práce s řetězci 23
Prog. 2 – ASP: Iterace 23
Prog. 3 – ASP: Rekurze 24
Prog. 4 – ASP: Výpočet Ludolfova čísla 24
Prog. 5 – ASP: Eratostenovo síto 25
Prog. 6 – ASP: Práce s databází Microsoft SQL Server 2005 Express 25
Prog. 7 – PHP: Pomocí lokální proměnné 37
Prog. 8 – PHP: Pomocí globální proměnné 38
Prog. 9 – PHP: Iterace 38
Prog. 10 – PHP: Rekurze 38
Prog. 11 – PHP: Výpočet Ludolfova čísla 39
Prog. 12 – PHP: Eratostenovo síto 39
Prog. 13 – PHP: Práce s databází MySQL 40
Prog. 14 – ASP.NET: Pomocí klasické metody práce s řetězci 90
Prog. 15 – ASP.NET: Pomocí komponenty StringBuilder 90
Prog. 16 – ASP.NET: Iterace 91
Prog. 17 – ASP.NET: Rekurze 91
Prog. 18 – ASP.NET: Výpočet Ludolfova čísla 92
Prog. 19 – ASP.NET: Eratostenovo síto 93
Prog. 20 – ASP.NET: Práce s databází Microsoft SQL Server 2005 Express 94

SEZNAM PŘÍLOH

PŘÍLOHA P1: Odkazy na dostupné podobné testy a související zdroje

PŘÍLOHA P2: Odkazy na bližší informace o použitém software

PŘÍLOHA P 1: ODKAZY NA DOSTUPNÉ PODOBNÉ TESTY A SOUVISEJÍCÍ ZDROJE

1. <http://www.root.cz/clanky/mysql-vs-postgresql-vs-firebird-ii/>
2. < http://www.newsforge.com/article.pl?sid=04/12/27/1243207>
3. <http://benchw.sourceforge.net/benchw_results_open3.html>
4. <http://cbbrowne.com/info/rdbmssql.html>
5. <http://www.zive.cz/h/Programovani/Ar.asp?ARI=104162>
6. <http://www.osnews.com/story.php?news_id=7800>
7. < http://www.process64.com/articles/xmlmark1/ >
8. <http://www.microsoft.com/technet/archive/itsolutions/intranet/downloads/webstres.mspx?mfr=true>

PŘÍLOHA P 2: ODKAZY NA BLIŽŠÍ INFORMACE O POUŽITÉM SOFTWARE

1. *Microsoft Windows XP Professional SP2*
   <http://www.microsoft.com/cze/windows/xp/pro/>
2. *Mandriva Linux 2006,* <http://wwwnew.mandriva.com/>
3. *Technologie ASP, ASP.NET,* < http://www.asp.net>
4. *Technologie PHP,* <http://www.php.net>
5. *Web server Apache,* <http://www.apache.org>
6. *Web server IIS,* <http://www.microsoft.com/WindowsServer2003/iis/>
7. *Databázový server Microsoft SQL Server 2005 Express*
   <http://msdn.microsoft.com/vstudio/express/sql/>
8. *Databázový server MySQL,* <http://www.mysql.com>
9. *Databázový server PostgreSQL,* <http://www.postgresql.org>
10. *Databázový server FirebirdSQL,* < http://www.firebirdsql.org/>
11. *Databázový server Oracle,* < http://www.oracle.com>
12. *.NET Framework 2.0*, <http://msdn.microsoft.com/netframework/>
13. *Zend Encoder, Zend Optimizer*, <http://www.zend.com>
14. *Mono Project*, < http://www.mono-project.com/>