به نام خدا

تحقیق آزمایشگاه دیتابیس

موضوع اول: نحوه encrypt کردن پسورد در دیتابیس

موضوع دوم: انواع لاگ در در فایل ها در sql

موضوع سوم: لاگ فایل های گرفته شده را چگونه در داخل یک پایگاه داده دیگر انتقال و ذخیره شود.

اعضای گروه:

سبحان کاظمی

علیرضا اسلامی خواه

موضوع اول :

هنگامی که کاربر سعی می کند وارد سیستم شود، می توانیم جدول را با SELECT * FROM users WHERE email = {email} AND password = {password} جستجو کنیم. اگر پرس و جو نتیجه ای را برگرداند، می توانیم کاربر را احراز هویت کنیم.
حتی اگر راه حل فوق کار کند، امن نیست و مستعد حملات بسیاری است. مستقیم ترین حمله ای که ممکن است آشکار نباشد، حمله داخلی است. در این حمله افراد یا کارمندانی که به پایگاه داده شما دسترسی دارند (از جمله خودتان) می توانند به راحتی رمز عبور کاربر را ببینند و اعتبار خود را دریافت کنند.
نقض داده نیز یکی دیگر از مواردی است که چرا رمز عبور متن ساده وحشتناک است. ممکن است شخصی که قصد ندارید پایگاه داده شما را به دست آورد، و این اتفاق همیشه می افتد، حتی برای یک شرکت بزرگ. با یک رمز عبور متنی ساده، مهاجم می تواند با جستجو در جدول کاربران شما، تمام اطلاعات کاربری شما را دریافت کند.
اولین قدمی که می خواهید بردارید این است که رمز عبور کاربری خود را با یک تابع هش قبل از ذخیره آن در پایگاه داده هش کنید. بر خلاف رمزگذاری، عملکرد هش تنها می تواند یک راه باشد و نتیجه هش کردن یک رشته خاص همیشه یکسان خواهد بود. این باعث می شود که عملکرد هش یک فرآیند بسیار مناسب در ذخیره سازی رمز عبور باشد.

یکی از محبوب ترین و ایمن ترین توابع هش SHA256 است. اگر بخواهیم رمز عبور ضعیف را با SHA256 هش کنیم، نتیجه این خواهد بود:

```
9b5705878182ccecf493b6c5ef3d2c723082141d0af33432c997b52dcc9f3e71
```

حمله جدول رنگین کمان یک حمله با جدولی از هش از پیش محاسبه شده رمزهای عبور رایج است. با حمله جدول رنگین کمانی، مهاجم می‌تواند اعتبار کاربرانی را که پسوردهای ضعیفی در پایگاه داده شما دارند، دریافت کند. برای مبارزه با این، معمولاً هنگام هش رمز عبور از نمک استفاده می شود. salt یک مقدار تصادفی تولید شده است که می توانید قبل از هش کردن رمز عبور آن را ترکیب کنید. به عنوان مثال، اگر jvFJ4 را به عنوان یک نمک تولید کنیم و آن را با رمز عبور ضعیف ترکیب کنیم و آن را هش کنیم

(sha256("jvFJ4weakpassword")) تولید می کند:

```
b104c5bf49e2e4937ac2419e94864f7209014a96cae582302f6e5f891e426e22
```

هنگام ورود کاربر چه کاری باید انجام دهیم؟

کاربر ایمیل و رمز عبور ساده خود را ارسال می کند

سیستم باید جدول کاربران را از طریق ایمیل پرس و جو کند، به عنوان مثال، SELECT * FROM users WHERE

'email='codecurated@codecurated.com برای دریافت رمز عبور هش شده و salt
محاسبه SHA-256 (salt + گذرواژه ورودی)
نتیجه را با رمز عبور هش شده مقایسه کنید. اگر یکسان است، به این معنی است که کاربر رمز عبور صحیح را وارد کرده است و سیستم می تواند کاربر را احراز هویت کند.
این جایی است که Bcrypt وارد می شود. Bcrypt یک تابع هش رمز عبور بر اساس Blowfish است که در آن می توانید هزینه اجرای آن را تعیین کنید. این ویژگی، به ویژه، برای هش رمز عبور عالی است، زیرا در آینده عملکرد هش را زمانی که یک ماشین سریع‌تر ظاهر می‌شود، اثبات می‌کند.
می‌توانیم ببینیم که زمان در مقایسه با هزینه سهموی شد. هزینه بالاتر به معنای امنیت بهتر اما تجربه کاربری بدتر است. فقط تصور کنید که اگر هزینه را 20 تنظیم کنید، کاربر باید 53 ثانیه صبر کند.

موضوع دوم:

گزارش تراکنش های SQL Server بطور منطقی عمل می کند که گویی گزارش تراکنش رشته ای از رکوردهای گزارش است. هر رکورد لاگ با یک شماره توالی log (LSN) شناسایی می شود. هر رکورد log جدید با یک LSN که بالاتر از LSN رکورد قبل از آن است در انتهای منطقی گزارش نوشته می شود. رکوردهای log در یک توالی سریال ذخیره می شوند زیرا

ایجاد می شوند به این ترتیب که اگر LSN2 بزرگتر از LSN1 باشد، تغییر توصیف شده توسط رکورد log مورد اشاره LSN2 پس از تغییر توصیف شده توسط رکورد log LSN1 رخ می دهد. هر رکورد لاگ شامل شناسه تراکنشی است که به آن تعلق دارد. برای هر تراکنش، تمام رکوردهای گزارش مربوط به تراکنش به صورت جداگانه در یک زنجیره با استفاده از نشانگرهای رو به عقب که بازگشت تراکنش را سرعت می بخشد، مرتبط می شوند.

مراحل بازیابی یک عملیات به نوع رکورد log بستگی دارد:
عملیات منطقی ثبت شد
برای رول کردن عملیات منطقی به جلو، عملیات دوباره انجام می شود.
برای برگرداندن عملیات منطقی، عملیات منطقی معکوس انجام می شود.
تصویر قبل و بعد ثبت شده است
برای رول کردن عملیات به جلو، تصویر after اعمال می شود.
برای برگرداندن عملیات، تصویر قبل اعمال می شود.

موتور پایگاه داده سرور SQL هر فایل لاگ فیزیکی را به صورت داخلی به چندین فایل لاگ مجازی (VLF) تقسیم می کند. فایل‌های لاگ مجازی اندازه ثابتی ندارند و تعداد ثابتی از فایل‌های لاگ مجازی برای یک فایل لاگ فیزیکی وجود ندارد. موتور پایگاه داده هنگام ایجاد یا گسترش فایل‌های گزارش، اندازه فایل‌های گزارش مجازی را به صورت پویا انتخاب می‌کند. موتور پایگاه داده سعی می کند چند فایل مجازی را حفظ کند.

اندازه فایل های مجازی پس از توسعه یک فایل گزارش، مجموع اندازه گزارش موجود و افزایش اندازه فایل جدید است. اندازه یا تعداد فایل‌های گزارش مجازی را نمی‌توان توسط مدیران پیکربندی یا تنظیم کرد.
اگر فایل‌های لاگ با افزایش‌های کوچک به اندازه بزرگی رشد کنند، فایل‌های لاگ مجازی زیادی خواهند داشت. این می تواند راه اندازی پایگاه داده را کند کند و عملیات پشتیبان گیری و بازیابی را ثبت کند. برعکس، اگر فایل‌های گزارش روی اندازه بزرگ با چند یا فقط یک افزایش تنظیم شوند، فایل‌های لاگ مجازی بسیار بزرگ کمی خواهند داشت. برای اطلاعات بیشتر در مورد تخمین صحیح اندازه مورد نیاز و تنظیم رشد خودکار گزارش تراکنش، به بخش توصیه‌ها در مدیریت اندازه فایل گزارش تراکنش مراجعه کنید.

چه اتفاقی می‌افتد وقتی VLF بیش از حد دارید؟
در طی مراحل اولیه فرآیند بازیابی پایگاه داده، SQL Server تمام VLF های موجود در تمام فایل های گزارش تراکنش ها را کشف می کند و لیستی از این VLF ها را می سازد. این فرآیند بسته به تعداد VLF های موجود در پایگاه داده خاص می تواند زمان بسیار زیادی را به طول بینجامد. هر چه تعداد VLF ها بیشتر باشد، روند طولانی تر می شود. یک پایگاه داده می تواند با تعداد زیادی VLF در نهایت رشد خودکار گزارش تراکنش یا رشد دستی در افزایش های کوچک مواجه شود. هنگامی که

تعداد VLF ها به محدوده چند صد هزار می رسد، ممکن است با برخی یا بیشتر علائم زیر مواجه شوید:
بازیابی یک یا چند پایگاه داده در طول راه اندازی SQL Server به زمان بسیار زیادی نیاز دارد.
بازیابی یک پایگاه داده زمان بسیار زیادی طول می کشد تا کامل شود.
تلاش برای پیوست کردن پایگاه داده زمان بسیار زیادی طول می کشد تا تکمیل شود.
هنگامی که می خواهید انعکاس پایگاه داده را تنظیم کنید، با پیام های خطای 1413، 1443 و 1479 مواجه می شوید که نشان دهنده مهلت زمانی است.
هنگام تلاش برای بازیابی پایگاه داده با خطاهای مرتبط با حافظه مانند 701 مواجه می شوید.

موضوع سوم :

همه پایگاه های داده SQL Server حداقل یک فایل پایگاه داده اصلی و یک فایل گزارش تراکنش دارند. فایل لاگ تراکنش هر تغییر داده و تراکنش DML را که در پایگاه داده اجرا شده است، ثبت می کند. نوشتن در فایل گزارش تراکنش در مقایسه با فایل های پایگاه داده که معمولاً I/O تصادفی هستند، ماهیت ترتیبی دارد. به این ترتیب، قرار دادن فایل log بر روی دیسک فیزیکی جداگانه از پایگاه داده به دیسک اجازه می دهد تا به صورت متوالی کار کند و عملکرد مطلوبی داشته باشد. برای انتقال به این پیکربندی، لازم است پایگاه داده را جدا کرده و ضمیمه کنید.

این مراحل را می توان با دستورات T-SQL یا استودیوی مدیریت سرور SQL (SSMS) انجام داد. ما نمونه ای از هر تکنیک را در این نکته توضیح خواهیم داد

موارد اولیه زیر باید قبل از انتقال فایل گزارش تراکنش به مکان جدید بررسی شوند:

مکان فعلی، اندازه و غیره فایل های پایگاه داده را ثبت کنید
مکان فعلی، اندازه و غیره فایل گزارش تراکنش که قرار است جابجا شود را ثبت کنید
به مکان، اندازه و غیره مقصد آینده فایل گزارش تراکنش توجه کنید
زمانی که هیچ کاربری به برنامه متصل نیست، برای جابجایی پایگاه داده زمان‌بندی زمان‌بندی کنید
اعتبار سنجی پایگاه داده در هیچ طرح تکراری، در یک زمان‌بندی عکس فوری یا عضوی از یک آینه نیست.
اگر چنین است، بر این اساس برنامه ریزی کنید و اسکریپت هایی را برای رسیدگی به این تنظیمات بسازید
مطمئن شوید که عضوی از نقش ثابت db_owner هستید.

به طور خلاصه، سه مرحله اصلی در انتقال یک فایل گزارش به مکان جدید عبارتند از:

دیتابیس را جدا کنید
فایل گزارش را به مکان جدید منتقل کنید
با ارجاع به مکان جدید فایل گزارش تراکنش، پایگاه داده را پیوست کنید
برای اهداف نمایشی، از پایگاه داده AdventureWorks استفاده خواهیم کرد که در حال حاضر بر روی درایو D:\ سرور SQL من نصب شده است. این پایگاه داده ممکن است بر روی درایو دیگری روی دستگاه شما نصب شده باشد، اما مفهوم اصلی نمایش این است که فایل گزارش تراکنش AdventureWorks را به یک درایو فیزیکی دیگر یعنی C:\ منتقل کنید.