Tisdag 29 november – Lagra data, Olika driftmiljöer

Före lunch mestadels teoretisk genomgång. Efter lunch mestadels egna övningar.

Passa på att ställa eventuella frågor om inlämningsuppgiften!

Vid kl 14.30 samling för summering av dagen och möjlighet att ställa frågor.

Normalisera mera, eller mindre?

Lite repetition av vad vi gör när vi normaliserar:

- Skapar en tabell per uppsättning data som hör ihop
- Placerar data som upprepas i separata tabeller
- Delar upp sammansatta egenskaper på minsta möjliga beståndsdelar (atomära värden)

Denormalisering, argument för och emot.

Bättre prestanda vid select, men sämre vid insert och update?

Risk att få en inkonsistent databas (med motsägelser).

Beräknade kolumner, exempelvis sätta flaggor eller uppdatera räknare när andra kolumner ändrats.

Trigger = en procedur som anropas vid en händelse (insert, update, delete) på ett objekt i databasen.

Användbart för att uppdatera redundanta data.

Materialiserad vy, en sparad ögonblicksbild.

Eller vy plus klustrat index.

Fysisk lagring

- RAID[1] för bättre läs- och skrivprestanda, och felsäkerhet.
- Striping, med data "strimlat" över flera diskar.
- Spegling, ökar läsprestanda, sänker skrivprestanda.
- Kontrollsummor.
- Paritetsbitar.

Molnlösningar

- Virtuella maskiner.
- Elastiska lösningar.
- Någon annan som sköter säkerhetskopiering mm.
- Dock mindre flexibelt.
- Dyrare? Beror på hur man menar.

- Skalbarhet, vertikalt och horisontellt (bättre hårdvara vs mera hårdvara).
- Kluster av servrar. Kräver att man koordinerar och replikerar.

Partitionera data

- Fördela olika *tabeller* mellan olika servrar.
- Fördela olika *datamängder* mellan olika servrar.
- Kan dock komplicera join och union. Eller – ibland – effektivisera join-operationer.

---

1 Redundant Array of Inexpensive Disks, eller Redundant Array of Independent Disks.

Var finns databasen, rent fysiskt?

select * from sys.database_files;

- Primär datafil (.mdf)
- Sekundär datafil (.ndf)
- Transaktionslogg (.ldf)

Logiska filnamn (“name”) vs fysiska filnamn (“filename”).
Ganska mycket överkurs: Användardefinierade filgrupper, filnamn, filstorlekar, mm.

DBCC = Database Console Command.
Till exempel “dbcc checkdb;”.

Eller för att visa antal “pages” allokerade i datafiler: “dbcc showfilestats;”.

Behov av att krympa data- eller loggfiler?
Går att göra med “dbcc shrinkfile”.
Till exempel efter större ändringar i schema, eller större operationer insert/delete.

Går även att frigöra utrymme genom att trunkera (avkorta) loggfil.
En fysisk loggfil består av flera virtuella loggfiler (VLF).

Säkerhetskopiera och återställa

“Every day is International Backup Awareness Day”

- Fullständig säkerhetskopiering
- Inkrementell
  - Differentiell inkrementell
  - Kumulativ inkrementell

Varför säkerhetskopiera?

- Fel på hårdvara
- Fel i hanteringen
- Attack utifrån
- Ett verktyg vid flytt från gammal till ny server.

I den allra enklaste varianten:
backup database Test to disk = '[infoga filnamn här]';
respektive
restore database Test from disk = '[infoga filnamn här]';

Skapa schemalagda jobb med sp_add_job, sp_add_jobstep, sp_add_schedule, sp_attach_schedule, sp_add_jobserver.

Tips från coachen: Flytta .bak-filerna till annan server än den där databasen är lagrad.
Den dag servern kraschar vill du inte förlora också säkerhetskopiorna.

Övningar

Prova att skapa en trigger som reagerar på händelser i en tabell och uppdaterar värden i en annan tabell.

Undersök olika kommandon, funktioner och lagrade procedurer för att visa information om databasen.

Testa att skapa säkerhetskopia av din databas.

Testa att radera din databas och återställa den från säkerhetskopia!
Se till att du som reservlösning också har sparat SQL-skript för att återskapa samma tabellstruktur.

Bonusuppgift: Vad kan hända om man återställer en databas till en annan installation av MS SQL Server?

Fortsätt arbeta med inlämningsuppgiften.
Fundera kanske särskilt på när normalisering är bra, jämfört med när redundanta data är bra.