

Kort om backup

Backup

- Kort oppsummert:
 - Backup er enkelt
 - Backup er viktig
 - Backup kan bli dyrt
 - Lite fokus på restore

Faktorer som påvirker backup:

- Backup media (Tape, disk, Optisk)
- Hjemmelaget / kjøpt
- Rutiner / Backup plan
- Mengde data
- Båndbredde
- Applikasjoner med egne dataformater

Begreper rundt backup

- Viktige begreper innen backup er:
 - corpus: Datamengden som skal bli tatt backup av
 - Endringsrate (change rate): Hvor mye av corpus har endret seg mellom hver gang
 - Backup vindu: Det tidsrommet vi kan ta backup
 - Snapshot: Vi tar en lokal kopi og tar backup av den, slik at corpus kan endre seg underveis
 - Lokal / Remote: Hvor backup skal være plassert
 - Medium: Hvor dataene lagres

Partiell, Inkrementell og full backup

- Vi skiller mellom tre strategier for backup:
 - Partiell
 - Inkrementell
 - Full
- Noen ganger flyter begrepene litt sammen
- Ved store datamengder, må man kombinere alle tre

Partiell backup

- Corpus er så stor at vi må dele den opp i deler
 - F.eks 7 deler og tar backup av en del hver dag
- Dersom corpus endrer seg må vi enten kombinere med inkrementell eller begynne med å ta lokal snapshot
- Partiell backup er også en løsning dersom båndbredden er for lav for full backup

Inkrementell backup

- Man tar bare backup av endringene siden sist
- Forutsetter en full backup tidligere
- Fungerer bra så lenge endringsraten er lavere enn (båndbredde + backup vindu)
- Full backup må uansett tas regelmessig

Nyttige Linux verktøy

- Det finnes mange gode løsninger for backup
- De er ofte basert på tre viktige bestanddeler:
 - tar + gz (lag en gzip av en hel mappe)
 - rsync (synkroniser to mapper, gjerne over SSH)
 - scp (bruker SSH til å flytte filer)
 - cp -al (tar kopi, men lager bare ny inode. Bruker ikke mer plass før den originale filen endrer seg. -a beholder eierskap og rettigheter)

MySQL - Log-filer

Error-log

- Inneholder feilsituasjoner og om mysqld serveren har blitt startet/stoppet
- Var egen logfil, men mange automatiske installasjoner lar den peke rett mot
`/var/log/syslog`

Query log

- Inneholder en lang log over ALLE spørringer, gruppert på forbindelser til databasen
- Inneholder også tidspunkt for forbindelsens start og stop, men ikke hvor lang tid hver enkelt spørring tok
- Er ikke skrudd på til vanlig

```
090207 7:22:22      8 Query      SELECT Ifnull(MAX(aID),0) FROM account
8 Query      select balance from account where aID = 3
8 Query      select balance from account where aID = 10
8 Query      update account SET balance = 559 where aID = 3
8 Query      SELECT Ifnull(MAX(tID),0) FROM transactions
8 Query      insert into transactions(type,tID,aIDfrom, aIDto, amount) values
( 'transfer',10, 3, 10, 96 )
090207 7:22:24      8 Quit
```

Binær-log

- En viktig bestanddel av MySQL sin mulighet til å ta backup / restore
- Kun operasjoner som *forandrer* data blir lagret og først *etter* at de er fullført
- Filen er på et binært format for raskest mulig skriving og lesing
`/var/log/mysql/mysql-bin. xxxxxxx`
- Filen `mysql-bin.index` holder orden
- Bruk kommandoen `mysqlbinlog` til å lese den

Binær-log eksempel

Blå = header linje, Sort = transaksjon

```
# at 18760#090207 7:22:20 server id 1 end_log_pos 18916
Query thread_id=8 exec_time=0 error_code=0
SET TIMESTAMP=1233987740;
insert into transactions(type,tID,aIDfrom, aIDto, amount) values
( 'transfer',9, 6, 8, 5637 );
# at 18916#090207 7:22:22 server id 1 end_log_pos 19025
Query thread_id=8 exec_time=0 error_code=0SET
TIMESTAMP=1233987742; update account SET balance = 559 where aID = 3;
# at 19025#090207 7:22:22 server id 1 end_log_pos 19137
Query thread_id=8 exec_time=0 error_code=0
SET TIMESTAMP=1233987742; update account SET balance = 10096 where aID =
10;
```

Slow (langsom) query log

- Inneholder alle queries som varer lengre enn det som er spesifisert i

`long_query_time = X`

```
# Time: 090207  9:28:33# User@Host: bankuser[bankuser] @  
legolas.iu.hio.no [128.39.89.21]# Query_time: 3  Lock_time: 0  
Rows_sent: 1  Rows_examined: 102194
```

```
use bank;SELECT Ifnull(MAX(tID),0) FROM transactions;
```

```
# Time: 090207  9:29:45
```

```
# User@Host: bankuser[bankuser] @ legolas.iu.hio.no  
[128.39.89.21]# Query_time: 2  Lock_time: 0  Rows_sent: 1  
Rows_examined: 102533
```

```
use bank;SELECT Ifnull(MAX(tID),0) FROM transactions;
```

```
# User@Host: bankuser[bankuser] @ legolas.iu.hio.no  
[128.39.89.21]# Query_time: 7  Lock_time: 0  Rows_sent: 1  
Rows_examined: 102428
```

```
SELECT Ifnull(MAX(tID),0) FROM transactions;
```

Logfil vedlikehold

- Linux vil automatisk rotere log-filene for oss hver natt, maks syv netter (er det en bra ting?)
- Se i filen `/etc/logrotate.d/mysql-server`
- Binær-logfiler roteres, men ikke på samme ukentlige sirkel
- Man kan fremprovosere en ny binær-log fil med kommandoen
`mysqladmin flush-logs`

MySQL Backup

Backup

- Når man lager en backup-strategi, er det viktigst å spørre seg om hvordan man vil kunne gjenopprette data
- Backup i MySQL består av det å dumpe data og å ta vare på logfiler
 - Bruker potensielt mye ressurser
- Hvor viktig er dataintegritet i forhold til ytelse?

Situasjoner hvor ekstra backup er nødvendig

- Før man skal oppgradere / gjøre store endringer
- Etter feil som er blitt gjort av sysadmin
- Etter ondsinnede kommandoer har blitt kjørt, f.eks `delete * from posts;`
- Etter tap av maskinvare eller planlagt migrering til ny server

mysqldump

- Vanligste verktøyet til å ta backup av en database og alt dets innhold
- Skriver ut alt som SQL-kode med tanke på at du en gang kanskje vil bruke den til å gjenopprette databasen
 - Backup skrives til skjerm, må omdirigeres til fil
- Kan ta backup av alle “interne” databasene slik som mysql (dvs. brukere og tilgang)

Selektiv backup av databaser

- Kun EN database

```
mysqldump [opsjoner] database [tabeller]
```

- Kun noen databaser

```
mysqldump [opsjoner] --databases db1 db2 db3
```

- Alle databaser

```
mysqldump [opsjoner] --all-databases
```

Opsjoner til mysqldump I

- `--add-drop-table`
Legg ved kommandoer som sletter tabellene først
- `--add-locks`
Legg ved låser rundt hver tabell-gjennoppretting
- `--create-options`
Legg ved alle MySQL opsjoner ved opprettelse av tabeller
- `--disable-keys`
Ikke optimaliser indeks etter hver tuppel er lagt til
- `--extended-insert`
Optimalisering av insert operasjonene

Opsjoner til mysqldump II

- `--lock-tables`
Lås alle tabellene før dump
- `--quick`
Ikke last alt inn i minne før det skrives ut
- `--set-charset`
Legg ved kommando for å sette samme språk
- `--opt`
Denne vil skru på alle opsjonene som vi har gått igjennom hittil

Opsjoner til mysqldump III

- `--master-data=2`
Inkluder siste posisjon (og nummer på fil) i den binære logfilen
- `--flush-logs`
Bytt til ny binær logfil rett etter at lås er tatt, slik at alle endringer etter backup havner i den nye binlog filen

Det finnes mange flere opsjoner til mysqldump, men disse er de viktigste. Se man mysqldump.

mysqldump eksempel

- Typisk backup av alt:

```
mysqldump --opt --master-data=2 --flush-logs  
--all-databases > backup.sql
```

- backup.sql inneholder nå alt man trenger for å gjenopprette databasen

- Man kan da kjøre restore:

```
cat backup.sql | mysql
```

Vanligvis velger man et bedre navn for filen enn backup.sql, f.eks backup_all_04_03_2008.sql

Resultat fra mysqldump i backup.sql

- Takket være --master-data=2

```
-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin.000011',  
MASTER_LOG_POS=98;
```

- Takket være --add-drop-table

```
DROP TABLE IF EXISTS `account`;  
CREATE TABLE `account` (  
  `name` varchar(100) default NULL,  
  `aID` int(11) NOT NULL auto_increment,  
  `balance` int(11) default NULL,  
  PRIMARY KEY (`aID`)  
) ENGINE=MyISAM AUTO_INCREMENT=3976 DEFAULT CHARSET=latin1;
```

- Så kommer data

```
LOCK TABLES `account` WRITE;  
/*!40000 ALTER TABLE `account` DISABLE KEYS */;  
INSERT INTO `account` VALUES ('Olivia',1,7996),('Ethan',  
2,32394),.... forts..
```

Er restore fra mysqlump nok?

- Siden restore fra en slik backup fil fjerner alt, vil du miste all data siden backup ble tatt
- Man kan bruke bin-log til å gjenopprette alle hendelsene frem til nå
 - ─ Ofte vil man gjøre dette manuelt og under oppsyn, f.eks luke ut feilen som forårsaket tapet

Tilbake til Binære logfiler

- Roterer etter 999999 filer på våre maskiner
- Vi må vite hvilken logfil som begynner *etter* vi tok siste mysqldump
- Kan lese logfilene med `mysqlbinlog` som skriver ut SQL kode
- Vi kan gjenopprette alle hendelsene i den binære logfilen med kommandoen:
`mysqlbinlog mysql-bin.xxxxxxx | mysql`

Restore fra mysqlbinlog

- mysqlbinlog alene kommer ikke til å kjøre kommandoene, så man kan trygt bruke det til å sjekke hva som har skjedd

```
mysqlbinlog mysql-bin.* | less
```

- Dersom man vil kjøre restore må man sende utskriften videre til mysql serveren:

```
mysqlbinlog mysql-bin.* | mysql
```

mysqlbinlog

- Vi har mulighet til å gjøre mysqlbinlog litt mer sofistikert:
 - Stopp før dato: `--stop-datetime="dato"`
 - Start fra dato: `--start-datetime="dato"`
eks: `--start-datetime="2005-12-25 11:25:56"`
 - Kun for databasen db: `--database=db`
 - Start fra posisjon n: `--start-position=n`
 - Stopp før posisjon n: `--stop-position=n`

Forstå utskrift fra mysqlbinlog

Posisjon i logfilen

Dato

Posisjon til neste hendelse

```
# at 18916#090207 7:22:22 server id 1 end_log_pos 19025 Query
thread_id=8 exec_time=0 error_code=0
SET TIMESTAMP=1233987742;update account SET balance = 559 where aID
= 3;
```

Dato forblir original

Selve kommandoene som vil bli kjørt

Flere mysql-bin filer

- Det er viktig at man prøver å unngå å kjøre én og én binær logfil slik:

```
mysqlbinlog mysql-bin.0001 | mysql  
mysqlbinlog mysql-bin.0002 | mysql
```

- Temporære operasjoner som spenner begge filene kan forsvinne mellom tilkoblingene til databasen

- Kjør dem heller i en og samme omgang:

```
mysqlbinlog mysql-bin.000[1,2] | mysql  
# eller  
mysqlbinlog mysql-bin.000? | mysql  
# eller  
mysqlbinlog mysql-bin.* | mysql
```

mysqlbinlog eksempler

- Kjør alle operasjoner men stopp FØR posisjon 475
`mysqlbinlog --stop-position=475 mysql-bin.*`
- Kjør alle operasjoner FRA OG MED posisjon 475
`mysqlbinlog --start-position=475 mysql-bin.*`
- Man kan ikke bruke vilkårlige posisjoner, sjekk alltid logfilen manuelt for å finne de rette posisjonene
- Datoer kan derimot velges vilkårlig

Advarsel: restore fra dump havner også i bin-log

- Når man kjører en restore fra mysqldump, vil alle de nye endringene nå være skrevet i bin-log også
- Hvis du altså skulle kjørt alle resterende bin-log filene, ville slutten av siste bin-log fil satt det hele tilbake til da du tok restore
- Dette kan forhindres på flere måter:
 - Bestem stop-posisjonen med `mysqlbinlog`
 - Legg til linjen “`SET sql_log_bin=0`” øverst i `backup.sql`
 - Flush loggene før restore med kommandoen
`mysqladmin flush-logs`

Strategi

- Kjør dump (full backup) med gjevne mellomrom (f.eks ukentlig, hver natt eller ved minst aktivitet)
- Roter binær-logfilene med kortere intervaller (inkrementell backup) etter hver dump
- Hold orden på hvilke binære logfiler som kommer etter hvilken dump
- Total gjenoppretting vil da bestå av to deler:
 1. Restore fra dump
 2. Restore fra påfølgende binær-logfiler

Fjerne uønskede kommandoer fra bin-log

- Det kan være problematisk å fjerne de kommandoene som forårsaket skade med mysqlbinlog kommandoen
- Følgende strategi kan fungere bedre:
 1. Kjør mysqlbinlog fra de aktuelle logfilene men lagre resultatet til en ny fil:
`mysqlbinlog mysql-bin.* > binlog.sql`
 2. Editer binlog.sql og fjern linjer
`jed binlog.sql`
 3. Kjør binlog.sql på databasen
`cat binlog.sql | mysql`

Andre potensielle utfordringer

- Binlog filene roteres automatisk derom de blir for store
 - Kan justeres i `/etc/mysql/my.cnf`
- Logrotate roterer logfiler og binlog automatisk
 - Det er greit så lenge man har kontroll og oversikt
- Dersom backup og binlog lagres på samme maskin er du ikke beskyttet mot tap av disk

Kjøre backup fra annen maskin

- Det er mulig å “hente” data ved å kjøre mysqldump fra en annen maskin:

```
mysqldump --user=root --host=IP --password=passw ... > fil.sql
```

- Brukeren trenger en del rettigheter
- Man kan også hente bin-log filer på samme måten:

```
mysqlbinlog --user=root --host=IP --password=passw --read-from-remote-server --to-last-log mysql-bin.000014 > binlog.sql
```

 - Legg merke til ingen sti til bin-log filen
 - Man må ha en ‘server-id tall’ linje i my.cnf og restarte serveren