Prosjekt 2 – Store datamengder, høst 2015, utgave 1.
Innleveringsfrist: antagelig 30.09.15 kl. 23.50 på Fronter.
***** Dette settet blir utvidet etter hvert, del A og B publiseres herved. *****

Veiledning: onsdag 23, fredag 25, tirsdag 29, onsdag 30. september.

Hensikt:
Hensikten med prosjektet er

A. Å få trening i skriving av enkle lagrede prosedyrer/funksjoner og triggere.
B. Å videre kunnskap om og trening i jobbing med distribuering og replikasjon.
C. …….

Det er opp til dere hvor mye dere legger i å løse dem – dette vil forhåpentligvis vise seg i bedømmelsen av disse.
NB! Det er realistisk at dere må bruke det aktuelle systemets manual på nettet for å finne svar.
Noen endringer i formuleringer underveis må påregnes.

Regn med å kunne forklare løsningene dine for foreleser/gruppelærer.

Del A: Triggere og lagrede prosedyrer.

Nivå 1.

1) Lag et lite system av triggere som legger inn en standardverdi som ut fra en verdi i en annen tabell.
   Eksempel: Vi har en varetabell og en ordrelinjetabell
   Vare (varenr, varenavn, veil_pris, …), Ordrelinje (ordrenr, varenr, aktuell_pris, ….).
   Når det lages en ny ordrelinje, er det lurt å legge inn veil_pris som aktuell_pris, selv om denne kan endres etterpå.
   Lag selv et annet eksempel, men med den samme problemstillingen.

2) Lag et lite system som sørger for å holde en unormalisert struktur konsistent.
   Eksempel: Vi har tabellene Post (postnr, poststed) og Ansatt (ansnr, etternavn, fornavn, postnr, poststed). Dette er jo dårlig normalisering, men kan forsvares hvis man legger til triggere som holder poststedet konsistent i Ansatt-tabellen. Det skal ikke være mulig å endre postnummer i post-tabellen, men navnet på poststedet blir jo endret av og til.
   NB! Tenk gjennom alle triggere som trengs i begge tabeller for å garantere konsistens.
   Selv om mySQL ikke har mulighet for at en trigger ruller tilbake transaksjonen, finnes det måter å få det til på likevel.
   Lag selv et annet eksempel, men med den samme problemstillingen.

3) Lag en tabell med data (**gjør det enkelt, lag f.eks. et par-tre millioner rader** med tilfeldige tall ved hjelp av en lagret prosedyre). Lag deretter en funksjon som skriver ut medianverdien, dvs. det midterste tallet i sortert rekkefølge. Hvis det er et likt antall tall, skal snittet av de to midterste skrives ut.

Nivå 2.

4) Lag to eller flere tabeller som trigger hverandre i sirkel. Kjør disse, og vis hvorledes aktuelt databasesystem reagerer.

5) Lag et litt større, selvvalgt system hvor du bruker en god del triggere. Beskriv først systemet og behovet for triggere.

Del B: Replikasjon.

Nivå 1.

1) Sett opp en replikasjon med en tjener og minst 2 slaver. Dere skal beskrive hva dere gjorde, og vise hvorledes det fungerer i praksis.
   Innlevering: det lureste er at dere filmer hvordan dere satt opp tjeneren og slavene, og viser hvorledes en replikasjon foregår i praksis. Eventuelt kan faglærer tilkalles for at dere gjør en demo.
   For mySQL, se: https://dev.mysql.com/doc/refman/5.7/en/replication.html

2) Store systemer som Google, Facebook, Instagram, Dropboks m.m. må opplagt jobbe med distribusjon og replikasjon.
   Finn helst noe mindre «opplagt» enn de store, og finn stoff om dette. Alternativt kan dere se på replikering i det små, f.eks. hvordan kalenderreplikering.
   Innlevering: En **selvkomponert** beskrivelse av hvorledes et større firma jobber med distribusjon og/eller replikasjon – helst i form av en PowerPoint/Impress. Tett med referanser!
   Hvis vi får tid, får dere noen minutter hver til å presentere funnene deres.

Nivå 2.

3) Gjør videre eksperimenter med replikasjon (evt. distribusjon).

4) Sammenlign distribusjon og replikasjon i to ulike RDMBSer ut fra litteratur dere finner. Dette skal dokumenteres (referanser) og systematiseres, gjerne med oversikter. De som klarer å gjøre noe av denne sammenligningen i praksis får et ekstra +.
   Det samme med evt. presentasjon, se under 2)

Del C: ……