BOKMÅL

Universitetet i Sørøst-Norge

EKSAMENSFORSIDE

Hjemmeeksamen

Emnekode: **DAT1000/DAT1000N**	Emnenavn: Database 1	
Emneansvarlig: Bjørn Kristoffersen	Campus: Bø/Bakkenteigen/Nett	Fakultet: Handelshøyskolen
Utlev. dato og tidspunkt i WISEflow: 11.08.2021 kl. 09:00	Innlev. dato og tidspunkt i WISEflow: 11.08.2021 kl. 13:00	
Antall oppgaver: 3	Antall vedlegg: Ingen	Ant. sider inkl. forside og vedlegg: 4

Hjelpemidler og samarbeid:

Tillatte hjelpemidler: Alle skriftlige hjelpemidler er tillatt.

	Ja	Nei
Er det individuell eksamen?	☒	☐
Er det tillatt å samarbeide med andre personer?	☐	☒

Beskrivelse av individuell eksamen og samarbeid finner du på min.usn.no/eksamen

Kriterier for besvarelsen:

Skrifttype: Ingen spesielle krav.	Skriftstørrelse: Ingen spesielle krav.	Linjeavstand: Ingen spesielle krav.
Antall ord (min/maks): Ingen spesielle krav.	Maks antall sider ekskl. forside og vedlegg: Ingen spesielle krav.	

Kildehenvisning:
Bruk en av de vanlige referansestilene, f.eks. APA, beskrevet her:
https://bibliotek.usn.no/oppgaveskriving/litteraturliste-og-kildehenvisninger/

Vekting av oppgavene ved sensur:

- Oppgave 1: 40 %
- Oppgave 2: 40 %
- Oppgave 3: 20 %

Du bør løse oppgave 1 før oppgave 2, fordi SQL-spørsmålene i oppgave 2 bygger på datamodellen du skal utvikle i oppgave 1. Vektingen gir også en god pekepinn til hvor mye tid du bør sette av til hver oppgave. Legg ved dine egne forutsetninger / tolkinger hvis oppgaveteksten er uklar eller tvetydig. Prøv å svare på så mange spørsmål som mulig. Lykke til!

KANDIDATEN MÅ SELV KONTROLLERE AT OPPGAVESETTET ER FULLSTENDIG

Oppgave 1 (40 %)

I denne oppgaven skal du tegne en logisk datamodell (ER-diagram). Diagrammet skal vise entiteter og attributter med datatyper, forhold med kardinaliteter samt primærnøkler og fremmednøkler. Du skal tegne identifiserende forhold med heltrukne linjer og ikke-identifiserende forhold med stiplede linjer. Du kan tegne diagrammene med MySQL Workbench, eller du kan tegne for hånd (bruk da notasjonen i læreboken) og deretter ta et bilde av modellen, som du limer inn i besvarelsen.

Datamodellen skal lages på en slik måte at SQL-spørringene beskrevet i oppgave 2 lar seg løse.

Beskrivelse av systemet

Du skal lage en database for en jobbsøkerportal på nettet, der bedrifter kan legge ut stillingsannonser og privatpersoner kan søke på ledige stillinger.

Om personer som skal bruke løsningen skal man lagre et fornavn, et etternavn, en adresse, et telefon-nummer og en epostadresse. Det skal være et norsk system, så adressen vil bestå av en gateadresse, et postnummer og et poststed. Ved registrering velger man dessuten et passord som blir kryptert og lagret i databasen. Man kan også legge inn fødelsdatoen sin.

For å registrere en bedrift må først en av de ansatte registrere seg som bruker. Vedkommende vil fungere som kontaktperson for bedriften. En bedrift har alltid et navn, en adresse og et unikt organisasjonsnummer. Man kan også legge inn nettadressen til bedriftens hjemmeside.

En bedrift kan legge ut annonser for ledige stillinger. Det må da legges inn en stillingstittel, en beskrivelse og en søknadsfrist. Stillingen får automatisk tildelt et unikt stillingsnummer. Det kan legges til et arbeidssted hvis dette ikke er det samme som bedriftens adresse. Det kan også legges til en egen kontaktperson for stillingen, hvis ikke blir bedriftens kontaktperson brukt.

Stillinger skal ha en status med en av følgende lovlige verdier: «ny», «aktiv», «avsluttet». Stillingen står i utgangspunktet som «ny» når den blir laget. Bedriften kan gjøre den «aktiv» og synlig for alle når man vil åpne for søking, før den blir satt til «avsluttet» når en person er tilsatt i stillingen, eller man velger å avslutte prosessen av andre grunner.

En stilling skal være i minst én stillingskategori, som «IT», «Omsorg», «Industri», «Skipsfart», osv. Det vil finnes kategorier som ikke er i bruk.

Personer kan velge å søke på en gitt stilling. En slik søknad har en fritekst der man skriver selve brevet. Søkeren kan også lenke søknaden til CV, anbefalingsbrev og andre dokumenter som man har lastet opp. Et gitt dokument kan brukes i flere søknader (søkeren trenger ikke å laste de opp på nytt). Søkeren legger inn en kort beskrivelse for hvert dokument. Dokumentene lagres utenfor databasen, kun filnavnet blir lagret i databasen.

En søknad har også en status, med de samme tillatte verdier som nevnt over for stilling. En søknad på en stilling kan være påbegynt, og er da «ny» inntil personen sender den inn. Først da blir den «aktiv», før den blir «avsluttet» når søknadsfristen utløper. En jobbsøker kan ha flere søknader inne samtidig på forskjellige stillinger, og få listet alle stillinger hun eller han har søkt på.

Det skal være mulig å bekrefte at andre personer har ulike kompetanser, f.eks. «programmering», «SQL» eller «sikkerhet». Listen med mulige kompetansebeskrivelser skal lagres i databasen. En kompetanse vil beskrives med noen få ord, som oftest bare ett. Dato for bekreftelse av en kompetanse skal lagres.

Oppgave 2 (40 %)

I denne oppgaven skal du skrive SQL-spørringer mot databasen du laget datamodell for i oppgave 1. Hvis du har laget modellen med MySQL Workbench, så bør du vurdere å lage en database med «forward engineer», slik at du kan teste ut SQL-spørringene. Men hvis du ikke er trygg på hvordan man gjør dette, så er det kanskje bedre bare å skrive SQL-spørringer uten å teste. Pass også på at du ikke bruker for lang tid på å teste en enkelt spørring, kanskje det bare er en liten detalj som er feil.

Hvis du oppdager at datamodellen du laget i oppgave 1 ikke inneholder nok opplysninger, eller er uegnet for å svare på en av SQL-oppgavene under, og du ikke har tid til å rette opp datamodellen, så kan du på et slikt spørsmål beskrive et tillegg eller en endring i datamodellen, f.eks. ved å skrive: «Jeg antar i dette spørsmålet at databasen inneholder følgende tabeller…». For full uttelling må slike tillegg gi en hensiktsmessig og normalisert database.

Tips: MySQL-dokumentasjonen har forklaring med eksempler til ulike funksjoner. På denne samlesiden er alle funksjonene ordnet etter datatype: https://dev.mysql.com/doc/refman/8.0/en/functions.html

2-a (5 %)
Skriv en SQL-spørring som viser alle personer født mellom 1970 og 1985 på postnummer 3200 og 3800.

2-b (5 %)
Skriv en SQL-spørring som viser aktive stillinger som inneholder «SQL» som del av beskrivelsen. Sorter resultatet med hensyn på søknadsfrist (de nyeste først) og deretter på tittel (hvis det er flere ledige stillinger med frist samme dag).

2-c (5 %)
Skriv en SQL-spørring som viser alle kompetansebekreftelser så langt i år. Ta med navn og alder på personen som har fått bekreftet en kompetanse, dato og selve kompetansen i resultatet.

2-d (5 %)
Skriv en SQL-spørring som viser antall (aktive) søknader for hver ledige stilling som nå er aktiv. Ta med stillingsnummer og tittel i resultatet.

2-e (5 %)
Bruk SQL for å utsette søknadsfristen med en uke for alle ledige stillinger (med status «aktiv») ved en gitt bedrift. Velg bedrift selv.

2-f (5 %)
Skriv en SQL-oppgave som tester evne til å lage *views* som *kobler data fra flere tabeller*, og løs deretter denne oppgaven. Prøv å lage en oppgave som er nyttig. Legg vekt på at du formulerer spørsmålet så presist som mulig. Krevende SQL-oppgaver vil gi mer uttelling enn enkle.

2-g (5 %)
Skriv en SQL-spørring som viser alle stillingskategorier som ikke er i bruk.

2-h (5 %)
Skriv en SQL-spørring som registrerer en ny ledig stilling. Velg eksempeldata selv. Gjør rede for eventuelle forutsetninger som gjelder fremmednøkler.

Oppgave 3 (20 %)

3-a (10 %)

Kløtsj AS trenger et enkelt system for å holde orden på serviceavtaler. For å få til dette så blir det foreslått å lage følgende tabell:

- Avtale(Dato, KlSlett, RegNr, Merke, Modell, Beskrivelse, EierTlf, EierNavn)

Eksempelrad:

- ('2021-14-08', '08:30', 'LY12345', 'Toyota', 'Auris', 'Bytte frontlykter', '12345678', 'Ole Mo')

Denne avtalen er satt opp kl. 08:30 på dato 14. august 2021. Avtalen gjelder bytting av frontlykter på en Toyota Auris med registreringsnummer LY12345. Eieren av bilen heter Ole Mo og har telefonnummer 12345678.

Forklar kort med bruk av konkrete eksempler hva som er uheldig med den nye tabellen. Skriv deretter ned funksjonelle avhengigheter, bestem kandidatnøkkel og utfør normalisering til BCNF. Få fram hvert normaliseringssteg. Du skal ikke legge til flere kolonner. Vis primærnøkler med understreking og merk fremmednøkler med en stjerne i resultatet.

3-b (5 %)

Ta utgangspunkt i databasen du laget til oppgave 1 og diskuter valg av primærnøkler, fremmednøkler og datatyper. Gjør rede for alternative løsninger og begrunn de valgene du tok.

3-c (5 %)

Det finnes ulike måter å begrense hva som skal være lovlige verdier i en databasetabell. Blant annet kan man bruke CHECK-regler og fremmednøkler. Ta igjen utgangspunkt i databasen fra oppgave 1 og vis konkrete eksempler på begrensninger som kan løses med både CHECK-regler og fremmednøkler, og hva som (eventuelt) kun kan løses med CHECK-regler eller kun med fremmednøkler. Klarer du også å finne eksempler som ikke lar seg håndtere med hverken CHECK-regler eller fremmednøkler?