

Eksamensoppgave i Databasesystemer

Forfattere: 233094 & 233099

Gruppe: 40

Høgskolen i Østfold – 15.03.2024

Innholdsfortegnelse

Introduksjon:	2
Kravspesifikasjon:.....	3
Normalisering:	5
ER modellering:	6
Dummy data	7
Opprettelse og fylling av tabeller - med skjermbilder.....	8
SQL spørringer.....	21
Forklaring av tabeller:	28
Konklusjon:	30
Referanseliste	31

Introduksjon:

I denne oppgaven skal vi utforme en database for treningsplanlegging og registrering. Målet er å utvikle et robust system som kan brukes av trenere og utøvere til å strukturere, overvåke og analysere treningsøkter, i tillegg til at man skal ha oversikt over fremgang over tid.

Kunden ønsker et effektivt system der trenere kan lage individuelle treningsplaner for utøvere, med mulighet for å inkludere ulike typer økter som kondisjonsøkter, styrkeøkter eller konkurranser. Det er viktig at trenere kan gjenbruke tidligere øvelser ved opprettelse av nye økter, og at utøvere har tilgang til forklaringer på hvordan hver øvelse skal gjennomføres.

Basert på kundens behov ønsker vi å utvikle en database som gir utøvere mulighet til å se og registrere treningsøkter, inkludert detaljer som distanse, vekt, pulssoner og kommentarer. Denne informasjonen vil være tilgjengelig for både utøvere og trenere, slik at de kan evaluere fremgang over tid.

Vil vi følge en strukturert tilnærming som inkluderer kravspesifikasjon, normalisering av databasen, design av en Entitet-Relasjon (ER) modell, implementering av databasen i MySQL og til slutt fylling av databasen med dummy data.

Videre skal vi utforme SQL-spørringer og views for å demonstrere funksjonaliteten til systemet. Dette kan eksempelvis være progresjon, treningsprogrammer eller treningsøkter. Dette vil gi brukerne et verktøy for å analysere og forbedre treningsregimet og prestasjonene sine over tid.

Kravspesifikasjon:

Brukere – Brukeradministrasjon

- Databasesystemet skal ha brukere med unik ID i tillegg til relevant informasjon som «fornavn», «etternavn», «alder» og kjønn.
- Brukerne skal differensieres som enten "utøver" eller "trener".

Økt – Overblikk over kommende/tidligere økter

- Tabellen «økt» skal si hvordan type økt en utøver skal gjøre, og da per økt.
- Hver økt får sin egen ID, med «økttype», «dato» i tillegg til «brukerID» som sier noe om hvilke bruker som skal gjøre hvilke økt til hvilken dato.
- En økt kan være en kondisjonsøkt, en styrkeøkt eller en konkurranse.
- Det skal være mulig å skrive kommentarer til hver økt (som en dagbok).

Øvelser – Oversikt over øvelser

- Tabellen «øvelser» skal inneholde alle øvelsene en utøver kan gjøre, eller som en trener kan tildele.
- Hver øvelse får tildelt en unik ID, og har et navn og en beskrivelse.

Øvelselogg – Logg av treningsøkt (treningsøvelser)

- Tabellen «Øvelselogg» skal være logger over alle øvelser en utøver gjør i løpet av en økt.
- Hver logg får tildelt en unik ID, og inkluderer brukerens ID, øvelsens ID, dato og puls-sone.
- Avhengig av typen øvelse, kan det også legges til informasjon om repetisjoner og vekt, eller distanse.

Programmer – Oversikt over opprettede programmer

- Tabellen «Programmer» skal vise til opprettede programmer.
- Hvert program får tildelt en unik ID og et beskrivende navn, som kan referere til ukedager eller annet.
- Tabellen viser også oppvarming og avslutning for hvert program.

Program øvelser – Øvelser tilhørende programmer

- Tabellen "Program_øvelser" er en oppslagstabell som knytter øvelser til programmer.
- Tabellen skal inneholde en «program_id» som skal matche «program_id» i tabellen «programmer». I tillegg skal det også være «ØvelseID» som sier noe om hvilke øvelser som skal gjøres.

Ukeoversikt – Oversikt over hvilken plan utøvere følger

- Tabellen "Ukeoversikt" skal inneholde informasjon om hvilken ukeplan en spesifikk utøver følger, navnet/id-en på treneren som tildelte/opprettet denne planen, i tillegg til start og sluttdato for programmet.
- Hver ukeoversikt skal ha sin egen unike ID.

Ukeplan – Henvisning til program

- Tabellen «ukeplan» skal være en oppslagstabell som kan henvises til for å vise hvilke program som skal brukes.
- Den skal ha en primærnøkkel «PlanID» og en foreign-key «ProgramID» som refererer til programmer.

Ukeplan_ukedager – Henvisning til hvilke plan som skal brukes til ulike datoer

- Tabellen «ukeplan_ukedager» skal være en oppslagstabell for å se hvilke plan man skal bruke i henhold til datoen man skal trene.
- Tabellen skal ha en PlanID som referer til «ukeplan»
- Tabellen skal ha en «ukedag» som viser til datoen hvor planene skal brukes.

Normalisering:

1NF – Første Normalform

For å få modellen vår til 1NF startet vi med å sørge for at alle verdiene våre var atomære, altså at vi ikke har lister i koloner eller at all data i en kolonne er av samme datatype. Eksempelvis kunne vi ikke ha fornavn og etternavn i samme kolonne, men måtte separere disse. Deretter undersøkte vi hvilken data som blir redundant og dermed tar opp unødvendig lagringsplass i databasen. Vi evaluerte så hvordan man kan opprette separate tabeller for å lagre informasjonen mer effektivt.

2NF – Andre Normalform

Da vi jobbet med å få database modellen vår til andre normalform, måtte vi fjerne eventuelle partielle avhengigheter. En partiell avhengighet oppstår når en kolonne er avhengig av en del av primærnøkkelen, men ikke hele nøkkelen. Dette fører ofte til et rotete forhold mellom kolonnene innad i tabellen i tillegg til at det skaper en unødvendig rotete struktur. Ved å sørge for at vi ikke hadde noen partielle avhengigheter skaper vi igjen en enda bedre databasestruktur.

3NF – Tredje Normalform

For å få modellen fra andre til tredje normalform måtte vi fjerne alle transitive avhengigheter. Transitive avhengigheter oppstår når to kolonner i en tabell er gjensidig avhengige av hverandre, og i tillegg avhenger av primærnøkkelen. Når en tabell inneholder transitive avhengigheter, inneholder den mest sannsynlig også redundante data, noe som tar unødvendig lagringsplass i databasen. Ved å kvitte oss med disse transitive avhengigheter skaper vi igjen en enda bedre og ryddigere databasestruktur.

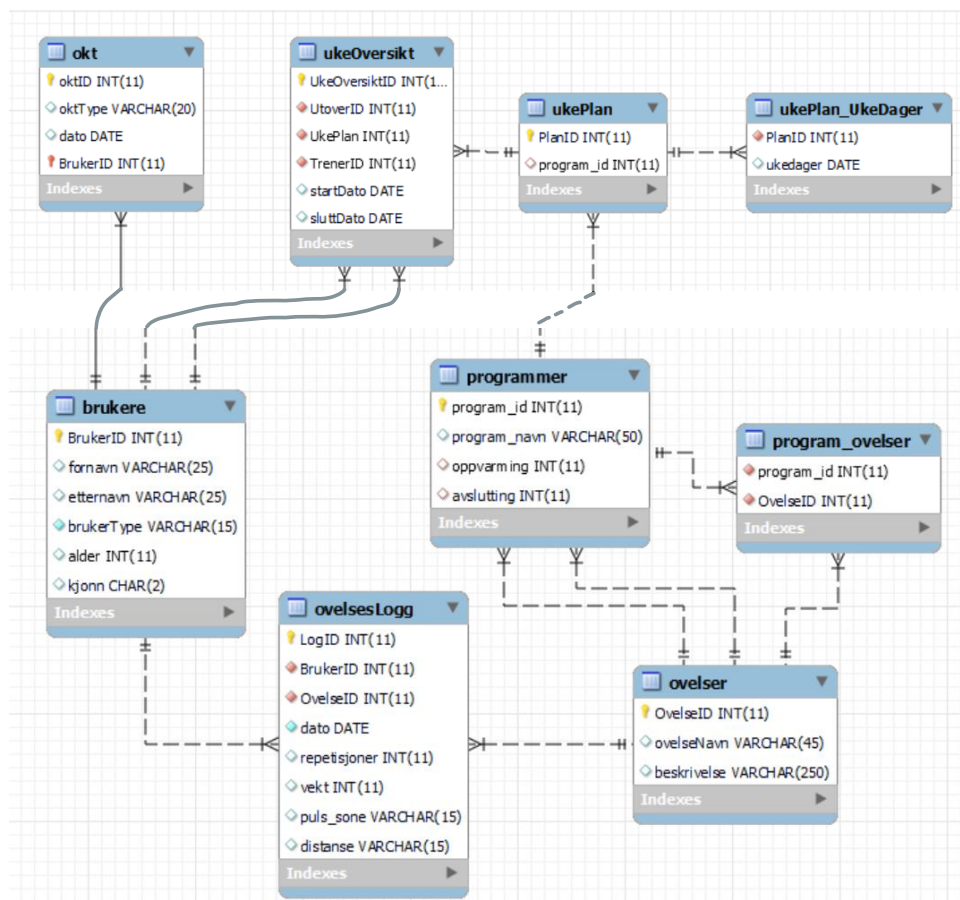
BCNF – Boyce-Codd Normalform

BCNF, eller Boyce-Codd normalform, er en normaliseringsform som bygger på 3NF. Denne normalformen er enda mer restriktiv enn tredje normalform og sikter mot å oppnå en enda bedre database struktur. For å få modellen vår til BCNF måtte vi sørge for at alle minimale determinanter var kandidatnøkler. Dette gjorde vi ved å sikre at alle våre funksjonelle avhengigheter var en triviell funksjonell avhengighet. En triviell funksjonell avhengighet er når det er åpenbart eller trivielt at en funksjonell sammenheng eksisterer mellom to eller flere variabler innad en tabell. For eksempel,

mellom alderen til en person og antall år de har levd er en triviell funksjonell avhengighet fordi alderen og antall år levd refererer til det samme.

Ved å oppnå BCNF på databasen vår kunne vi sikkert si at alle funksjonelle avhengigheter i databasen var godt definerte og at vi ikke hadde noe redundant data. Dette gir oss en veldig god databasestruktur som er lett å vedlikeholde over tid.

ER modellering:



Dummy data

Alle våre tabeller er fylt med dummy-data hvor noen er fylt manuelt, og noen er fylt av ChatGPT, med instruksjoner fra oss. Da ChatGPT ble brukt ble den brukt til å både skrive INSERT INTO scripts, og andre ganger bare informasjonen som vi etterspurte i spørringen. Ett eksempel på dette ligger under, hvor resten av spørringene fra ChatGPT med resultater ligger vedlagt etter litteraturlisten.

Dummy data er fylt inn for vise poenget med databasen og hva de forskjellige tabellene gjør, og er derfor med på å skape perspektiv slik at forståelsen av systemet blir lettere.

Under spurte vi ChatGPT om å lage 50 tilfeldig profiler til «brukere»-tabellen vår med parameterne som vi hadde bestemt på forhånd (brukerID, fornavn, etternavn, brukertype, alder, kjønn), med første linje som eksempel.

På grunnlaget av at det ble 50 brukere, fikk vi ChatGPT til å lage INSERT spørringene for oss. Ettersom det også ble såpas mange linjer, har vi valgt for oversiktens skyld å bare inkludere de 3 første her. Resterende brukere finner du under ved «Opprettelse og fylling av tabeller - med skjermbilder». Strukturen er lik hvor alle har en økende og unik ID, samt ulike/tilfeldige fornavn, etternavn, brukertyper, alder og kjønn. Alder verdiene spurte vi spesifikt om at skulle være mellom 18-60 og kjønn skulle tilfeldig bli delt ut "D" (Dame) eller "M" (Mann).

INSERT INTO brukere (brukerID, fornavn, etternavn, brukerType, alder, kjønn) **VALUES**

(1, 'Nancy', 'Gerlach', 'Kunde', 26, 'D'),

(2, 'Marcella', 'Parisian', 'Trener', 26, 'D'),

(3, 'Guillermo', 'Homenick', 'Kunde', 38, 'D'),

Opprettelse og fylling av tabeller - med skjermbilder

Brukere:

```
CREATE TABLE brukere (  
  BrukerID int NOT NULL,  
  fornavn VARCHAR(25),  
  etternavn VARCHAR(25),  
  brukerType VARCHAR(15) NOT NULL,  
  alder INT,  
  kjonn CHAR(2),  
  
  CONSTRAINT pk_brukere PRIMARY KEY (BrukerID)  
);
```

```
INSERT INTO brukere (brukerID, fornavn, etternavn, brukerType, alder, kjonn) VALUES  
(1, 'Nancy', 'Gerlach', 'Kunde', 26, 'D'),  
(2, 'Marcella', 'Parisian', 'Trener', 26, 'D'),  
(3, 'Guillermo', 'Homenick', 'Kunde', 38, 'D'),  
(4, 'Jared', 'West', 'Kunde', 29, 'M'),  
(5, 'Janie', 'Hilpert', 'Kunde', 60, 'D'),  
(6, 'Francis', 'Champlin', 'Trener', 24, 'M'),  
(7, 'Ramon', 'Osinski', 'Trener', 27, 'M'),  
(8, 'Stewart', 'Howell', 'Kunde', 21, 'M'),  
(9, 'Sue', 'Gutkowski', 'Trener', 41, 'D'),  
(10, 'Oscar', 'Kub', 'Trener', 37, 'M'),  
(11, 'Frankie', 'Kilback', 'Trener', 56, 'D'),  
(12, 'Jared', 'Walter', 'Trener', 34, 'D'),  
(13, 'Kathleen', 'Kirlin', 'Kunde', 18, 'D'),  
(14, 'Mike', 'Stoltenberg', 'Trener', 33, 'D'),  
(15, 'Joanne', 'Padberg', 'Kunde', 42, 'M'),  
(16, 'Jimmy', 'Weissnat', 'Trener', 31, 'D'),  
(17, 'Sharon', 'Koss', 'Kunde', 59, 'D'),  
(18, 'Norman', 'Rosenbaum', 'Kunde', 56, 'D'),  
(19, 'Clark', 'Lesch', 'Trener', 35, 'D'),  
(20, 'Gerald', 'Hamill', 'Kunde', 40, 'D'),  
  
(21, 'Martin', 'Olsen', 'Trener', 27, 'M')  
  
(22, 'Robert', 'Mccullough', 'Kunde', 45, 'M'),  
(23, 'Dorothy', 'Matthews', 'Kunde', 32, 'D'),  
(24, 'Alan', 'Simpson', 'Trener', 39, 'M'),  
(25, 'Denise', 'Mcdonald', 'Kunde', 50, 'D'),  
(26, 'Martin', 'Ferguson', 'Trener', 36, 'M'),  
(27, 'Alice', 'Campbell', 'Kunde', 48, 'D'),  
(28, 'Paul', 'Bowman', 'Trener', 29, 'M'),  
(29, 'Lori', 'Huff', 'Kunde', 55, 'D'),  
(30, 'Terry', 'Floyd', 'Kunde', 43, 'M'),  
(31, 'Teresa', 'Olson', 'Kunde', 34, 'D'),  
(32, 'Ralph', 'Gonzalez', 'Trener', 47, 'M'),  
(33, 'Stephanie', 'Rice', 'Kunde', 30, 'D'),  
(34, 'Lawrence', 'Parker', 'Kunde', 41, 'M'),  
(35, 'Kathryn', 'Armstrong', 'Kunde', 37, 'D'),
```


(36, 'Louis', 'Black', 'Kunde', 52, 'M'),
 (37, 'Melissa', 'Dean', 'Trener', 44, 'D'),
 (38, 'Billy', 'Bates', 'Kunde', 26, 'M'),
 (39, 'Christina', 'Reid', 'Kunde', 49, 'D'),
 (40, 'Philip', 'Chavez', 'Trener', 33, 'M'),
 (41, 'Mary', 'Wagner', 'Kunde', 31, 'D'),
 (42, 'Geraldine', 'Harris', 'Kunde', 58, 'D'),
 (43, 'Ronald', 'Rivera', 'Kunde', 46, 'M'),
 (44, 'Melinda', 'Peters', 'Trener', 35, 'D'),
 (45, 'Bruce', 'Mcguire', 'Kunde', 39, 'M'),
 (46, 'Wanda', 'Hansen', 'Kunde', 53, 'D'),
 (47, 'Harry', 'Hunt', 'Trener', 29, 'M'),
 (48, 'Gloria', 'Fields', 'Kunde', 42, 'D'),
 (49, 'Roger', 'Bryant', 'Trener', 37, 'M'),
 (50, 'Evelyn', 'Duncan', 'Kunde', 44, 'D');

	BrukerID	fornavn	etternavn	brukerType	alder	kjonn
►	1	Nancy	Gerlach	Utover	26	D
	2	Marcella	Parisian	Trener	26	D
	3	Guillermo	Homenick	Utover	38	D
	4	Jared	West	Utover	29	M
	5	Janie	Hilpert	Utover	60	D
	6	Francis	Champlin	Trener	24	M
	7	Ramon	Osinski	Trener	27	M
	8	Stewart	Howell	Utover	21	M
	9	Sue	Gutkowski	Trener	41	D
	10	Oscar	Kub	Trener	37	M
	11	Frankie	Kilback	Trener	56	D

brukere 1 ×

Økt:

```
CREATE TABLE okt (  
    oktID INT,  
    oktType varchar(20),  
    brukerID INT,  
    dato DATE,  
    notater VARCHAR(200),  
  
    CONSTRAINT pk_okt PRIMARY KEY (oktID),  
    CONSTRAINT fk Bruker FOREIGN KEY (brukerID) REFERENCES brukere(BrukerID),  
);
```

```
INSERT INTO okt (oktID, oktType, brukerID, dato) VALUES
```

```
(1, 'Styrkeøkt', 1, '2024-03-10'),  
(2, 'Kondisjonsøkt', 1, '2024-03-11'),  
(3, 'Kondisjonsøkt', 1, '2024-03-12'),  
(4, 'Styrkeøkt', 1, '2024-03-13'),  
(5, 'Kondisjonsøkt', 1, '2024-03-14'),  
(6, 'Kondisjonsøkt', 1, '2024-03-15'),  
(7, 'Konkurranse', 1, '2024-03-16'),  
(8, 'Kondisjonsøkt', 2, '2024-03-10'),  
(9, 'Styrkeøkt', 2, '2024-03-11'),  
(10, 'Konkurranse', 2, '2024-03-12'),  
(11, 'Kondisjonsøkt', 2, '2024-03-13'),  
(12, 'Styrkeøkt', 2, '2024-03-14'),  
(13, 'Konkurranse', 2, '2024-03-15'),  
(14, 'Styrkeøkt', 2, '2024-03-16'),  
(15, 'Styrkeøkt', 3, '2024-03-10'),  
(16, 'Kondisjonsøkt', 3, '2024-03-11'),  
(17, 'Konkurranse', 3, '2024-03-12'),  
(18, 'Styrkeøkt', 3, '2024-03-13'),  
(19, 'Kondisjonsøkt', 3, '2024-03-14'),  
(20, 'Konkurranse', 3, '2024-03-15'),  
(21, 'Konkurranse', 3, '2024-03-16');
```

```
UPDATE okt SET kommentar = 'Styrkeøkt gikk bra, økte vekten på benkpress og følte meg sterk.'
```

```
WHERE oktID = 1;
```

```
UPDATE okt SET kommentar = 'Hadde en intensiv kondisjonsøkt, klarte å forbedre tidene mine.'
```

```
WHERE oktID = 2;
```

```
UPDATE okt SET kommentar = 'Enda en god kondisjonsøkt, holdt en jevn intensitet gjennom hele økten.' WHERE oktID = 3;
```

```
UPDATE okt SET kommentar = 'Sterk styrkeøkt, økte vektene og forbedret formen min.' WHERE oktID = 4;
```

```
UPDATE okt SET kommentar = 'God kondisjonsøkt, fokuserte på utholdenhetstrening.' WHERE oktID = 5;
```

```
UPDATE okt SET kommentar = 'Fullførte en annen vellykket kondisjonsøkt, følte meg energisk.'
```

```
WHERE oktID = 6;
```

```
UPDATE okt SET kommentar = 'Konkurranse gikk bra, presterte bra og følte meg tilfreds.' WHERE oktID = 7;
```

```
UPDATE okt SET kommentar = 'Solid kondisjonsøkt, jobbet med å forbedre min utholdenhet.'
```

```
WHERE oktID = 8;
```

```
UPDATE okt SET kommentar = 'Styrkeøkt var utfordrende, økte vekter og fokuserte på form.' WHERE oktID = 9;
```

```
UPDATE okt SET kommentar = 'Konkurransen var tøff, men jeg klarte å presse meg gjennom.'
```

```
WHERE oktID = 10;
```

UPDATE okt **SET** kommentar = 'Kondisjonsøkten var intens, klarte å slå mine personlige rekorder.'
WHERE oktID = 11;
UPDATE okt **SET** kommentar = 'Sterk styrkeøkt, forbedret styrken og teknikken min.' **WHERE** oktID = 12;
UPDATE okt **SET** kommentar = 'Konkurransen var krevende, men jeg ga mitt beste.' **WHERE** oktID = 13;
UPDATE okt **SET** kommentar = 'Styrkeøkten var utfordrende, men jeg følte meg sterk og fokusert.' **WHERE** oktID = 14;
UPDATE okt **SET** kommentar = 'Gjennomførte en god styrkeøkt, økte vektene gradvis.' **WHERE** oktID = 15;
UPDATE okt **SET** kommentar = 'Kondisjonsøkten var bra, klarte å opprettholde en god intensitet.' **WHERE** oktID = 16;
UPDATE okt **SET** kommentar = 'Konkurransen gikk bra, presterte godt under press.' **WHERE** oktID = 17;
UPDATE okt **SET** kommentar = 'Styrkeøkten var intens, fokuserte på kjerneøvelser.' **WHERE** oktID = 18;
UPDATE okt **SET** kommentar = 'Jobbet med utholdenheten i dagens kondisjonsøkt.' **WHERE** oktID = 19;
UPDATE okt **SET** kommentar = 'Konkurransen var tøff, men jeg kjempet meg gjennom.' **WHERE** oktID = 20;
UPDATE okt **SET** kommentar = 'Konkurranse gikk bra, følte meg godt forberedt og fokusert.' **WHERE** oktID = 21;

Result Grid					
Filter Rows:					
Edit: Export/Import:					
	oktID	oktType	brukerID	dato	notater
▶	1	Styrkeøkt	1	2024-03-11	Styrkeøkt gikk bra, økte vekten på benkpress o...
	2	Kondisjonsøkt	1	2024-03-12	Hadde en intensiv kondisjonsøkt, klarte å forbe...
	3	Kondisjonsøkt	1	2024-03-13	Enda en god kondisjonsøkt, holdt en jevn intens...
	4	Styrkeøkt	1	2024-03-14	Sterk styrkeøkt, økte vektene og forbedret for...
	5	Kondisjonsøkt	1	2024-03-15	God kondisjonsøkt, fokuserte på utholdenhetstr...
	6	Kondisjonsøkt	1	2024-03-16	Fullførte en annen vellykket kondisjonsøkt, følte...
	7	Konkurranse	1	2024-03-17	Konkurranse gikk bra, presterte bra og følte me...
	8	Kondisjonsøkt	3	2024-03-11	Solid kondisjonsøkt, jobbet med å forbedre min ...
	9	Styrkeøkt	3	2024-03-12	Styrkeøkt var utfordrende, økte vekter og foku...
	10	Konkurranse	3	2024-03-13	Konkurransen var tøff, men jeg klarte å presse ...

Øvelser:

```
CREATE table ovelser (  
    OvelseID VARCHAR(50) NOT NULL,  
    beskrivelse VARCHAR(250),  
  
    CONSTRAINT pk_ovelser PRIMARY KEY (OvelseID)  
);  
  
INSERT INTO ovelser VALUES(1, "Jogging", "Løper i en moderat hastighet over en lengre periode");  
INSERT INTO ovelser VALUES(2, "Arm-og-beinsvinger", "Arm- og beinsvinger er en oppvarmingsøvelse der du svinger armene og beina i en kontrollert bevegelse");  
INSERT INTO ovelser VALUES(3, "Nedkjøling", "Nedkjøling og stretching for området som er blitt trent");  
INSERT INTO ovelser VALUES(4, "Løping", "Løping er en form for kardiovaskulær trening der du beveger deg fremover ved å løpe i et jevnt tempo");  
INSERT INTO ovelser VALUES(5, "Yoga", "Yoga er en praksis som kombinerer fysiske øvelser, pusteteknikker og meditasjon for å fremme fysisk og mental helse");  
INSERT INTO ovelser VALUES(6, "Svømming", "Svømming er en aerobisk treningsform der du beveger deg gjennom vann ved å bruke arm- og beinbevegelser");  
INSERT INTO ovelser VALUES(7, "Spasertur", "En spasertur er en enkel og lavintensiv form for fysisk aktivitet der du går i ditt eget tempo");  
INSERT INTO ovelser VALUES(8, "Sykling", "Sykling er en form for kardiovaskulær trening der du beveger deg ved å trække på pedaler og sykle enten innendørs på en stasjonær sykkel eller utendørs på en sykkelvei eller terreng");  
INSERT INTO ovelser VALUES(9, "Benkpress", "Benkpress er en øvelse der du ligger på en flat benk og løfter en vektstang opp og ned ved å bøye og strekke armene");  
INSERT INTO ovelser VALUES(10, "Roing-med-stang-eller-manualer", "Roing med stang eller manualer innebærer å trekke en vektstang eller manualer mot kroppen din mens du står eller sitter");  
INSERT INTO ovelser VALUES(11, "Skulderpress", "Skulderpress er en styrkeøvelse der du løfter en vektstang eller manualer over hodet ved å strekke armene oppover");  
INSERT INTO ovelser VALUES(12, "Bicepscurls", "Bicepscurls er en øvelse der du løfter vektstang eller manualer ved å bøye albue og kontrahere bicepsmuskelen");  
INSERT INTO ovelser VALUES(13, "Triceps-dips", "Triceps dips er en øvelse der du senker og løfter kroppen ved å bøye og strekke albue, mens du støtter deg på en benk eller stol.");  
INSERT INTO ovelser VALUES(14, "Knebøy", "Knebøy er en styrkeøvelse der du senker kroppen nedover ved å bøye knærne og hoftene, og deretter strekker deg opp igjen til stående posisjon.");  
INSERT INTO ovelser VALUES(15, "Utfall", "Utfall er en øvelse der du tar et stort skritt fremover med ett ben, senker kroppen nedover ved å bøye begge knærne, og deretter presser deg tilbake til stående posisjon.");  
INSERT INTO ovelser VALUES(16, "Rumensk-markløft", "Rumensk markløft er en øvelse der du holder en vektstang foran deg med strake armer, bøyer deg fremover i hoftene mens du holder bena strake, og deretter strekker deg opp igjen til oppreist posisjon ved å bruke baken og baksiden av lårene.");  
INSERT INTO ovelser VALUES(17, "Leg-curls", "Leg curls er en øvelse der du ligger på en benk med ansiktet nedover, og bøyer beina ved knærne mens du løfter vekten mot baken ved å trekke hælene mot setet.");  
INSERT INTO ovelser VALUES(18, "Leg-press", "Leg press er en øvelse der du sitter i en maskin med vektstangen hvilende på skuldrene, og deretter presser vekten opp ved å strekke ut bena mens du sitter.");  
INSERT INTO ovelser VALUES(19, "Valgfri-aktivitet", "Valgfri aktivitet som ikke er for krevende, f.eks. en rolig sykkeltur eller en tur i naturen");  
INSERT INTO ovelser VALUES(20, "Full-hvile/lett-aktivitet", "Full hviledag eller veldig lett aktivitet som rolig stretching eller en kort spasertur");  
INSERT INTO ovelser VALUES(21, "Lett-aktivitet", "Lett aktivitet som yoga, svømming eller en spasertur.");
```

INSERT INTO ovelser **VALUES**(22, "Sykling/Svømming", "Velg selv om du vil sykle eller ta en svømmetur");

OvelseID	ovelseNavn	beskrivelse
1	Jogging	Løper i en moderat hastighet over en lengre periode
2	Arm-og-beinsvinger	Arm- og beinsvinger er en oppvarmingsøvelse der du svinger armene og beina i en kontrollert...
3	Nedkjøling	Nedkjøling og stretching for området som er blitt trent
4	Løping	Løping er en form for kardiovaskulær trening der du beveger deg fremover ved å løpe i et je...
5	Yoga	Yoga er en praksis som kombinerer fysiske øvelser, pusteteknikker og meditasjon for å frem...
6	Svømming	Svømming er en aerobisk treningsform der du beveger deg gjennom vann ved å bruke arm- o...
7	Spasertur	En spasertur er en enkel og lavintensiv form for fysisk aktivitet der du går i ditt eget tempo
8	Sykling	Sykling er en form for kardiovaskulær trening der du beveger deg ved å trække på pedaler o...
9	Benkpress	Benkpress er en øvelse der du ligger på en flat benk og løfter en vektstang opp og ned ved ...
10	Roing-med-stang-eller-manualer	Roing med stang eller manualer innebærer å trekke en vektstang eller manualer mot kroppen...
11	Skulderpress	Skulderpress er en styrkeøvelse der du løfter en vektstang eller manualer over hodet ved å ...

øvelser 1 x

Øvelseslogg:

```
CREATE TABLE ovelsesLogg (  
  LogID INT NOT NULL,  
  BrukerID INT NOT NULL,  
  OvelseID INT NOT NULL,  
  dato DATE NOT NULL,  
  sett INT,  
  repetisjoner INT,  
  vekt INT,  
  puls_sone VARCHAR(15),  
  distanse VARCHAR(15),  
  
  CONSTRAINT pk_oLogg PRIMARY KEY (LogID, dato),  
  CONSTRAINT fk_bruker FOREIGN KEY (BrukerID) REFERENCES brukere(BrukerID),  
  CONSTRAINT fk_ovelse FOREIGN KEY (OvelseID) REFERENCES ovelser(OvelseID)  
);
```

```
INSERT INTO ovelsesLogg VALUES  
(4, 1, 9, '2024-03-11', 10, 35, '100-120', '2km'),  
(5, 1, 10, '2024-03-11', 8, 20, '100-120', NULL),  
(6, 1, 10, '2024-03-11', 10, 20, '100-120', NULL),  
(7, 1, 10, '2024-03-11', 10, 20, '100-120', NULL),  
(8, 1, 11, '2024-03-11', 8, 20, '100-120', NULL),  
(9, 1, 11, '2024-03-11', 8, 20, '100-120', NULL),  
(10, 1, 11, '2024-03-11', 8, 20, '100-120', NULL),  
(11, 1, 12, '2024-03-11', 12, 6, '100-120', NULL),  
(12, 1, 12, '2024-03-11', 12, 6, '100-120', NULL),  
(13, 1, 12, '2024-03-11', 10, 8, '100-120', NULL),  
(14, 1, 13, '2024-03-11', 10, 15, '100-120', NULL),  
(15, 1, 13, '2024-03-11', 10, 15, '100-120', NULL),  
(16, 1, 13, '2024-03-11', 10, 15, '100-120', NULL),  
(17, 1, 3, '2024-03-11', NULL, NULL, '100-120', NULL),  
(18, 3, 1, '2024-03-11', NULL, NULL, '120-140', '2km'),  
(19, 3, 9, '2024-03-11', 8, 25, '100-120', NULL),  
(20, 3, 9, '2024-03-11', 8, 35, '100-120', NULL),  
(21, 3, 9, '2024-03-11', 10, 40, '100-120', NULL),  
(22, 3, 10, '2024-03-11', 8, 25, '100-120', NULL),  
(23, 3, 10, '2024-03-11', 10, 25, '100-120', NULL),  
(24, 3, 10, '2024-03-11', 10, 25, '100-120', NULL),  
(25, 3, 11, '2024-03-11', 8, 25, '100-120', NULL),  
(26, 3, 11, '2024-03-11', 8, 25, '100-120', NULL),  
(27, 3, 11, '2024-03-11', 8, 25, '100-120', NULL),  
(28, 3, 12, '2024-03-11', 12, 7, '100-120', NULL),  
(29, 3, 12, '2024-03-11', 12, 7, '100-120', NULL),  
(30, 3, 12, '2024-03-11', 10, 10, '100-120', NULL),  
(31, 3, 13, '2024-03-11', 10, 17, '100-120', NULL),  
(32, 3, 13, '2024-03-11', 10, 17, '100-120', NULL),  
(33, 3, 13, '2024-03-11', 10, 17, '100-120', NULL),  
(34, 1, 1, '2024-03-25', NULL, NULL, '120-140', '2km'),  
(35, 1, 9, '2024-03-25', 8, 25, '100-120', NULL),  
(36, 1, 9, '2024-03-25', 8, 35, '100-120', NULL),  
(37, 1, 9, '2024-03-25', 10, 40, '100-120', NULL),  
(38, 1, 10, '2024-03-25', 8, 25, '100-120', NULL),  
(39, 1, 10, '2024-03-25', 10, 25, '100-120', NULL),  
(40, 1, 10, '2024-03-25', 10, 25, '100-120', NULL),
```



```
(41, 1, 11, '2024-03-25', 8, 25, '100-120', NULL),
(42, 1, 11, '2024-03-25', 8, 25, '100-120', NULL),
(43, 1, 11, '2024-03-25', 8, 25, '100-120', NULL),
(44, 1, 12, '2024-03-25', 12, 7, '100-120', NULL),
(45, 1, 12, '2024-03-25', 12, 7, '100-120', NULL),
(46, 1, 12, '2024-03-25', 10, 8, '100-120', NULL),
(47, 1, 13, '2024-03-25', 10, 17, '100-120', NULL),
(48, 1, 13, '2024-03-25', 10, 17, '100-120', NULL),
(49, 1, 13, '2024-03-25', 10, 17, '100-120', NULL),
(50, 1, 3, '2024-03-25', NULL, NULL, '100-120', NULL);
```

Result Grid								
Filter Rows:								
Edit: Export/Import:								
	LogID	BrukerID	OvelseID	dato	repetisjoner	vekt	puls_sone	distanse
▶	1	1	1	2024-03-11	NULL	NULL	120-140	2km
	2	1	9	2024-03-11	8	20	100-120	NULL
	3	1	9	2024-03-11	8	30	100-120	NULL
	4	1	9	2024-03-11	10	35	100-120	NULL
	5	1	10	2024-03-11	8	20	100-120	NULL
	6	1	10	2024-03-11	10	20	100-120	NULL
	7	1	10	2024-03-11	10	20	100-120	NULL
	8	1	11	2024-03-11	8	20	100-120	NULL
	9	1	11	2024-03-11	8	20	100-120	NULL
	10	1	11	2024-03-11	8	20	100-120	NULL
	11	1	12	2024-03-11	12	6	100-120	NULL

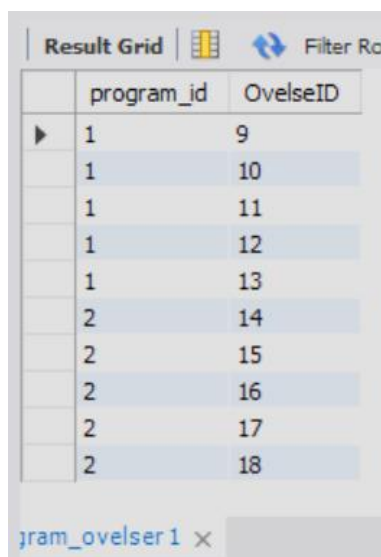
ovelsesLogg 1 x

Program øvelser:

```
CREATE TABLE program_ovelser (  
  program_id INT NOT NULL,  
  OvelseID INT NOT NULL,  
  
  CONSTRAINT fk_programmer FOREIGN KEY (program_id) REFERENCES  
programmer(program_id),  
  CONSTRAINT fk_ovelser FOREIGN KEY (OvelseID) REFERENCES ovelser(OvelseID)  
);
```

```
INSERT INTO program_ovelser VALUES
```

```
(1, 9),  
(1, 10),  
(1, 11),  
(1, 12),  
(1, 13),  
(2, 14),  
(2, 15),  
(2, 16),  
(2, 17),  
(2, 18);
```



The screenshot shows a 'Result Grid' window with a table containing two columns: 'program_id' and 'OvelseID'. The table has 10 rows of data. The first five rows have 'program_id' 1 and 'OvelseID' values 9, 10, 11, 12, and 13. The next five rows have 'program_id' 2 and 'OvelseID' values 14, 15, 16, 17, and 18. The window title is 'program_ovelser1 x'.

	program_id	OvelseID
▶	1	9
	1	10
	1	11
	1	12
	1	13
	2	14
	2	15
	2	16
	2	17
	2	18

Programmer:

```
CREATE TABLE programmer (  
  program_id INT,  
  program_navn VARCHAR(20),  
  oppvarming INT,  
  avslutning INT,
```

```
  CONSTRAINT pk_programmer PRIMARY KEY (program_id),  
  CONSTRAINT fk_ovelser1 (oppvarming) REFERENCES (ovelser.OvelseID),  
  CONSTRAINT fk_ovelser2 (avslutning) REFERENCES (ovelser.OvelseID)  
);
```

```
INSERT INTO programmer (program_id, program_navn, oppvarming, avslutting) VALUES  
(1, 'Mandag', 1, 2),  
(2, 'Tirsdag', 1, 3),  
(3, 'Onsdag', null, null),  
(4, 'Torsdag', 1, 3),  
(5, 'Fredag', null, null),  
(6, 'Lørdag', null, null),  
(7, 'Søndag', null, null);
```

	program_id	program_navn	oppvarming	avslutting
▶	1	Mandag	1	2
	2	Tirsdag	1	3
	3	Onsdag	NULL	NULL
	4	Torsdag	1	3
	5	Fredag	NULL	NULL
	6	Lørdag	NULL	NULL
	7	Søndag	NULL	NULL
★	NULL	NULL	NULL	NULL

Ukeoversikt:

```
CREATE table ukeOversikt (  
  UkeOversiktID INT NOT NULL,  
  UtoverID int NOT NULL,  
  UkePlan INT NOT NULL,  
  TrenerID int NOT NULL,  
  startDato date,  
  sluttDato date,
```

```
  CONSTRAINT pk_uke PRIMARY KEY (UkeOversiktID),  
  CONSTRAINT fk_brukere FOREIGN KEY (UtoverID) REFERENCES brukere(BrukerID),  
  CONSTRAINT fk_brukere1 FOREIGN KEY (TrenerID) REFERENCES brukere(BrukerID),  
  CONSTRAINT fk_uke FOREIGN KEY (UkePlan) REFERENCES ukePlan(PlanID)  
);
```

Result Grid						
Filter Rows:						
	UkeOversiktID	UtoverID	PlanID	TrenerID	startDato	sluttDato
▶	1	1	1	2	2024-03-11	2024-03-17
	2	3	1	6	2024-03-11	2024-03-17
	3	4	2	7	2024-03-11	2024-03-17
	4	5	2	9	2024-03-11	2024-03-17
	5	8	2	6	2024-03-18	2024-03-24
	6	13	1	10	2024-03-18	2024-03-24
	7	15	1	11	2024-03-18	2024-03-24
	8	17	2	12	2024-03-18	2024-03-24
	9	18	3	14	2024-03-18	2024-03-24
	10	20	3	16	2024-03-18	2024-03-24
	11	21	4	19	2024-03-11	2024-03-17

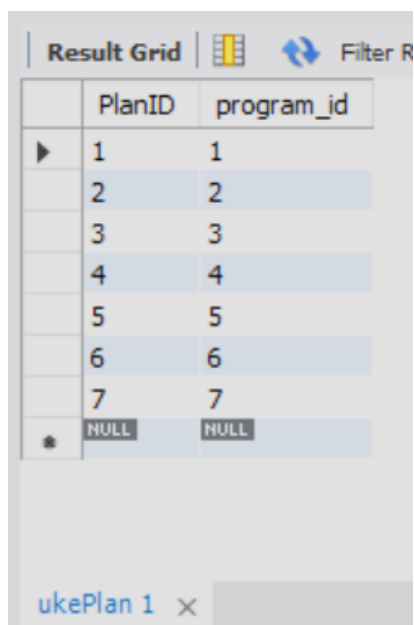
ukeOversikt 1 ×

Ukeplan:

```
CREATE TABLE ukePlan (  
    PlanID INT NOT NULL,  
    program_id INT NOT NULL,  
  
    CONSTRAINT pk_uPlan PRIMARY KEY (PlanID),  
    CONSTRAINT fk_program FOREIGN KEY (program_id) REFERENCES programmer(program_id)  
);
```

```
INSERT INTO ukePlan (PlanID, program_id) VALUES
```

```
(1, 1),  
(2, 2),  
(3, 3),  
(4, 4),  
(5, 5),  
(6, 6),  
(7, 7);
```



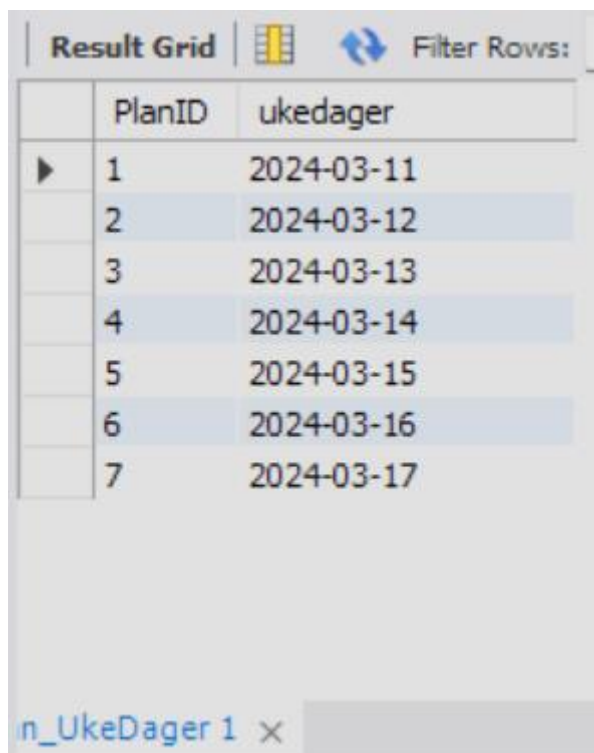
	PlanID	program_id
▶	1	1
	2	2
	3	3
	4	4
	5	5
	6	6
	7	7
★	NULL	NULL

ukePlan 1 ×

Ukeplan-ukedager:

```
CREATE TABLE ukePlan_UkeDager (  
    PlanID INT NOT NULL,  
    ukedager DATE,  
  
    CONSTRAINT pk_ukePlan FOREIGN KEY (PlanID) REFERENCES ukePlan(PlanID)  
);
```

```
INSERT INTO ukePlan_UkeDager (PlanID, ukedager) VALUES  
(1, '2024-03-11'),  
(2, '2024-03-12'),  
(3, '2024-03-13'),  
(4, '2024-03-14'),  
(5, '2024-03-15'),  
(6, '2024-03-16'),  
(7, '2024-03-17');
```



	PlanID	ukedager
▶	1	2024-03-11
	2	2024-03-12
	3	2024-03-13
	4	2024-03-14
	5	2024-03-15
	6	2024-03-16
	7	2024-03-17

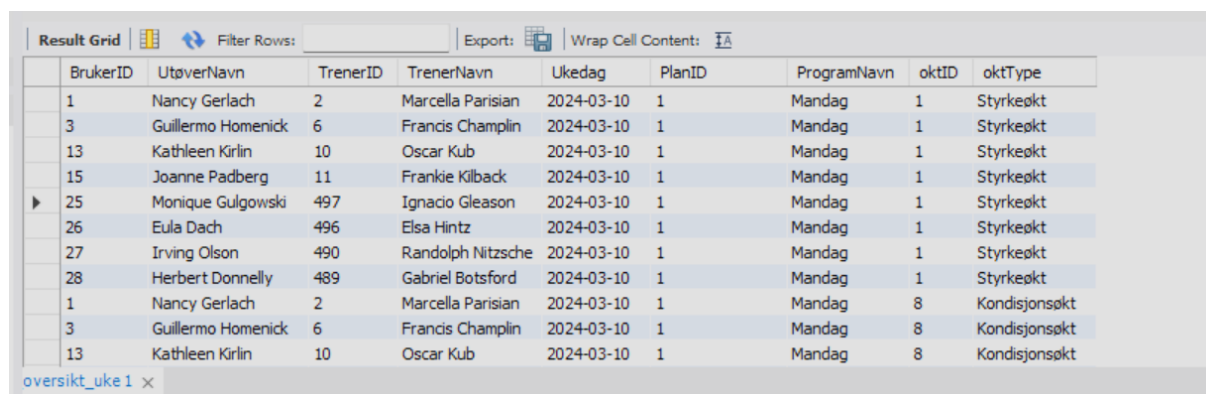
n_UkeDager 1 ×

SQL spørringer

Views:

Bruker oversikt uke:

```
CREATE VIEW bruker_oversikt_uke AS
SELECT
    ukeOversikt.UtoverID AS BrukerID,
    CONCAT(brukere.fornavn, ' ', brukere.etternavn) AS UtøverNavn,
    ukeOversikt.TrenerID AS TrenerID,
    CONCAT(trener.fornavn, ' ', trener.etternavn) AS TrenerNavn,
    ukePlan_UkeDager.ukedager AS Ukedag,
    ukeOversikt.UkePlan AS PlanID,
    programmer.program_navn AS ProgramNavn,
    okt.oktID,
    okt.oktType
FROM
    ukeOversikt
JOIN
    brukere ON ukeOversikt.UtoverID = brukere.BrukerID
JOIN
    brukere AS trener ON ukeOversikt.TrenerID = trener.BrukerID
JOIN
    ukePlan ON ukeOversikt.UkePlan = ukePlan.PlanID
JOIN
    ukePlan_UkeDager ON ukePlan.PlanID = ukePlan_UkeDager.PlanID
JOIN
    programmer ON ukePlan.program_id = programmer.program_id
JOIN
    okt ON ukePlan_UkeDager.ukedager = okt.dato;
```



	BrukerID	UtøverNavn	TrenerID	TrenerNavn	Ukedag	PlanID	ProgramNavn	oktID	oktType
	1	Nancy Gerlach	2	Marcella Parisian	2024-03-10	1	Mandag	1	Styrkeøkt
	3	Guillermo Homenick	6	Francis Champlin	2024-03-10	1	Mandag	1	Styrkeøkt
	13	Kathleen Kirlin	10	Oscar Kub	2024-03-10	1	Mandag	1	Styrkeøkt
	15	Joanne Padberg	11	Frankie Kilback	2024-03-10	1	Mandag	1	Styrkeøkt
▶	25	Monique Gulowski	497	Ignacio Gleason	2024-03-10	1	Mandag	1	Styrkeøkt
	26	Eula Dach	496	Elsa Hintz	2024-03-10	1	Mandag	1	Styrkeøkt
	27	Irving Olson	490	Randolph Nitzsche	2024-03-10	1	Mandag	1	Styrkeøkt
	28	Herbert Donnelly	489	Gabriel Botsford	2024-03-10	1	Mandag	1	Styrkeøkt
	1	Nancy Gerlach	2	Marcella Parisian	2024-03-10	1	Mandag	8	Kondisjonsøkt
	3	Guillermo Homenick	6	Francis Champlin	2024-03-10	1	Mandag	8	Kondisjonsøkt
	13	Kathleen Kirlin	10	Oscar Kub	2024-03-10	1	Mandag	8	Kondisjonsøkt

Bruker_oversikt_uke er et view laget for å illustrere en oversiktsplan for alle brukere. Oversikten inkluderer alle brukere som har fått tildelt en plan. Man kan derfor se hvilke bruker som har hvilke trener, hvilke plan den personen skal følge til hvilken dag, og også få vite hvordan økt det er. View-et er laget for å være allsidig. En eksempel-spørringen mot view-et kan for eksempel være å filtrere på BrukerID. Da vil man få opp relevant informasjon til utøveren i fokus.

I SELECT-setningen velges forskjellige kolonner og fornavn og etternavn slås sammen ved hjelp av CONCAT-funksjonen for å kombinere fornavn og etternavn til både utøveren og treneren.

Det utføres en rekke JOIN-operasjoner for å koble tabellene sammen basert på deres relasjoner. For eksempel, brukes JOIN til å kombinere ukeOversikt-tabellen med brukere-tabellen basert på BrukerID. JOIN brukes også til å kombinere ukePlan_UkeDager-tabellen med okt-tabellen basert på datoen for økten.

Bruker_progresjon:

```
CREATE VIEW bruker_progresjon AS
SELECT
    ovelsesLogg.BrukerID,
    CONCAT(brukere.fornavn, ' ', brukere.etternavn) AS UtøverNavn,
    ovelsesLogg.OvelseID,
    ovelser.ovelseNavn,
    MIN(ovelsesLogg.dato) AS forsteDato,
    MAX(ovelsesLogg.dato) AS sisteDato,
    MIN(ovelsesLogg.vekt) AS minVekt,
    MAX(ovelsesLogg.vekt) AS maxVekt,
    MIN(ovelsesLogg.repetisjoner) AS minRepetisjoner,
    MAX(ovelsesLogg.repetisjoner) AS maxRepetisjoner
FROM
    ovelsesLogg
JOIN
    brukere ON ovelsesLogg.BrukerID = brukere.BrukerID
JOIN
    ovelser ON ovelsesLogg.OvelseID = ovelser.OvelseID
WHERE
    ovelsesLogg.OvelseID NOT IN (1, 3)
GROUP BY
    ovelsesLogg.BrukerID, ovelsesLogg.OvelseID;
```

Result Grid										
Filter Rows:										
Export:										
Wrap Cell Content: 1A										
	BrukerID	UtøverNavn	OvelseID	ovelseNavn	forsteDato	sisteDato	minVekt	maxVekt	minRepetisjoner	maxRepetisjoner
▶	1	Nancy Gerlach	9	Benkpress	2024-03-11	2024-03-25	20	40	8	10
	1	Nancy Gerlach	10	Roing-med-stang-eller-manualer	2024-03-11	2024-03-25	20	25	8	10
	1	Nancy Gerlach	11	Skulderpress	2024-03-11	2024-03-25	20	25	8	8
	1	Nancy Gerlach	12	Bicepscurls	2024-03-11	2024-03-25	6	8	10	12
	1	Nancy Gerlach	13	Triceps-dips	2024-03-11	2024-03-25	15	17	10	10
	3	Guillermo Homenick	9	Benkpress	2024-03-11	2024-03-11	25	40	8	10
	3	Guillermo Homenick	10	Roing-med-stang-eller-manualer	2024-03-11	2024-03-11	25	25	8	10
	3	Guillermo Homenick	11	Skulderpress	2024-03-11	2024-03-11	25	25	8	8
	3	Guillermo Homenick	12	Bicepscurls	2024-03-11	2024-03-11	7	10	10	12
	3	Guillermo Homenick	13	Triceps-dips	2024-03-11	2024-03-11	17	17	10	10

Bruker_progresjon er et view som gir en detaljert oversikt over brukerens fremgang i treningsprogrammet. Visningen er utformet for å vise informasjon om hver brukers fremdrift i deres treningsplan og dermed de ulike øvelsene. Man har full oversikt over

alle brukere, hvilke øvelse det gjelder, første- og sistedato og selvfølgelig vekten og repetisjonene med minimums og maksimumsverdiene en utøver har klart.

I SELECT-setningen er ulike kolonner valgt for å gi en omfattende oversikt over brukerens progresjon. Fornavn og etternavn kombineres ved hjelp av CONCAT-funksjonen for å gi fullstendige navn for utøveren.

Det utføres JOIN-operasjoner mellom disse tabellene basert på deres relasjoner. JOIN brukes blant annet til å kombinere ukeOversikt-tabellen med brukere-tabellen basert på BrukerID, og JOIN brukes til å koble ukePlan_UkeDager-tabellen med økt-tabellen basert på øktens dato.

Dette view-et er utformet for å være allsidig, slik at man kan filtrere på ulike kriterier for å få en mer spesifikk visning av brukerens progresjon. For eksempel kan man filtrere på BrukerID og OvelseID for å få relevant informasjon om en bestemt utøver og eventuell øvelse og dermed deres fremgang i treningsprogrammet.

Utøver økt oversikt:

```
CREATE VIEW Utover_Okt_Overview AS
SELECT
    o.oktID,
    o.oktType,
    o.dato AS Dato,
    o.brukerID AS BrukerID,
    u.fornavn AS UtøverFornavn,
    u.etternavn AS UtøverEtternavn,
    ou.øvelseNavn AS Øvelse,
    ou.beskrivelse AS ØvelsesBeskrivelse
FROM
    okt o
JOIN
    brukere u ON o.brukerID = u.BrukerID
JOIN
    øvelsesLogg ol ON o.dato = ol.dato
JOIN
    øvelser ou ON ol.OvelseID = ou.OvelseID;
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

oktID	oktType	Dato	BrukerID	UtøverFornavn	UtøverEtternavn	Øvelse	ØvelsesBeskrivelse
1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Jogging	Løper i en moderat hastighet over en lengre per...
8	Kondisjonsøkt	2024-03-11	3	Guillermo	Homenick	Jogging	Løper i en moderat hastighet over en lengre per...
15	Styrkeøkt	2024-03-11	3	Guillermo	Homenick	Jogging	Løper i en moderat hastighet over en lengre per...
1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
8	Kondisjonsøkt	2024-03-11	3	Guillermo	Homenick	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
15	Styrkeøkt	2024-03-11	3	Guillermo	Homenick	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
8	Kondisjonsøkt	2024-03-11	3	Guillermo	Homenick	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
15	Styrkeøkt	2024-03-11	3	Guillermo	Homenick	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...

Ikt_Overview 1 x

Som man kan se over er "Utover_Okt_Overview" et view som viser en oversikt over øvelser som skal gjøres for hver enkelt utøver, men kan også brukes som en historikk. Det inneholder informasjon om brukerens ID, fornavn, etternavn, typen treningsøkt, datoen den skal gjennomføres (ev. har blitt gjennomført), øvelsene som skal gjøres og en kort beskrivelse av selve øvelsen. View-et gir en god oversikt over treningsaktivitetene til forskjellige brukere som er nyttig for både utøvere og trenere. Informasjonen i view-et kan brukes til å analysere treningsmønster, identifisere trender og planlegge fremtidige treningsøkter.

Da vi opprettet view-et "Utover_Okt_Overview" hentet vi som du kan se over, data fra tre tabeller for å gi en oversikt over når en bruker skal gjennomføre hvilke øvelser. View-et slår sammen informasjon om økter, brukere og øvelser for å danne en informativ tabell om treningsaktiviteter.

For å hente denne informasjonen koblet vi "okt"-tabellen med "brukere"- og "øvelsesLogg"-tabellene ved hjelp av bruker-ID og dato. Deretter koblet vi "øvelsesLogg" til "øvelser"-tabellen for å hente øvelsesnavn og beskrivelsen av øvelsen.

Select spørringer

Spørring:

```
SELECT b.BrukerID, b.fornavn, b.etternavn, o.oktID, o.oktType, o.dato
FROM brukere AS b
JOIN okt AS o ON b.BrukerID = o.BrukerID
WHERE o.BrukerID = '3';
```

Resultat:

Henter informasjon fra brukere og økt tabellen om brukeren som har bruker id 3.

	BrukerID	fornavn	etternavn	oktID	oktType	dato
▶	3	Guillermo	Homenick	15	Styrkeøkt	2024-03-10
	3	Guillermo	Homenick	16	Kondisjonsøkt	2024-03-11
	3	Guillermo	Homenick	17	Konkurranse	2024-03-12
	3	Guillermo	Homenick	18	Styrkeøkt	2024-03-13

Spørring:

```
SELECT
  (SELECT AVG(alder) FROM brukere WHERE brukerType = 'Trener') AS gjennomsnittsalderUtover,
  (SELECT AVG(alder) FROM brukere WHERE brukerType = 'Utover') AS gjennomsnittsalderTrener;
```

Resultat:

Henter ut gjennomsnittsalder for både trenere og utøvere i to separate kolonner.



	gjennomsnittsalderUtover	gjennomsnittsalderTrener
▶	39.6516	37.9297

Spørring:

```
SELECT OvelseID, ovelseNavn
FROM ovelser
WHERE OvelseID NOT IN (SELECT DISTINCT OvelseID FROM ovelsesLogg);
```

Resultat:

Henter ut alle øvelser med ID som ikke har blitt skrevet logg på.

Result Grid			 Filter Rows:
	OvelseID	ovelseNavn	
▶	2	Arm-og-beinsvinger	
	4	Løping	
	5	Yoga	
	6	Svømming	
	7	Spasertur	
	8	Sykling	
	14	Knebøy	
	15	Utfall	
	16	Svømming	

Spørring:

SELECT * **FROM** Utover_Okt_Overview **WHERE** BrukerID = 1 **and** dato = "2024-03-11";

Resultat:

Viser alle øvelsene som Nancy Gerlach (BrukerID 1) skal utføre den 11.03.2024. Her hentes informasjonen fra view-et som er vist over (Utover_Okt_Overview).

Result Grid		Filter Rows:		Export:		Wrap Cell Content:		
	oktID	oktType	Dato	BrukerID	UtøverFornavn	UtøverEtternavn	Øvelse	ØvelsesBeskrivelse
▶	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Jogging	Løper i en moderat hastighet over en lengre per...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Benkpress	Benkpress er en øvelse der du ligger på en flat ...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Roing-med-stang-eller-manualer	Roing med stang eller manualer innebærer å tre...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Roing-med-stang-eller-manualer	Roing med stang eller manualer innebærer å tre...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Roing-med-stang-eller-manualer	Roing med stang eller manualer innebærer å tre...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Skulderpress	Skulderpress er en styrkeøvelse der du løfter e...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Skulderpress	Skulderpress er en styrkeøvelse der du løfter e...
	1	Styrkeøkt	2024-03-11	1	Nancy	Gerlach	Skulderpress	Skulderpress er en styrkeøvelse der du løfter e...

Utøver Okt Overview 5 ×

Spørring:

SELECT UtøverNavn, ovelseNavn, forsteDato, sisteDato, minVekt, maxVekt, minRepetisjoner, maxRepetisjoner, (maxVekt - minVekt) **AS** vekt_forbedring
FROM bruker_progresjon
WHERE ovelseID = 9 **AND** BrukerID = 1;

Resultat:

Her vises fremgangen til Nancy Gerlach (BrukerID 1) for øvelsen Benkpress i perioden hun har trent. Her får en eventuell utøver god oversikt over vekt, repetisjoner og forbedringer når det kommer til vektmengde for øvelsen.

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	UtøverNavn	ovelseNavn	forsteDato	sisteDato	minVekt	maxVekt	minRepetisjoner	maxRepetisjoner	vekt_forbedring
	Nancy Gerlach	Benkpress	2024-03-11	2024-03-25	20	40	8	10	20

Spørring:

```
SELECT *  
FROM okt  
WHERE oktid = 1
```

Resultat:

Henter ut informasjon med øktID 1. Her får man oversikt over hvordan økt det var, når den var, hvem som gjennomførte den, og notatene som man har skrevet. Kan være greit hvis en utøver eller trener vil gå tilbake og lese hvordan det gikk.

Result Grid					
		Filter Rows:		Edit:	
				Export/Import:	
				Wrap Cell Content:	
	oktid	oktType	brukerID	dato	notater
▶	1	Styrkeøkt	1	2024-03-11	Styrkeøkt gikk bra, økte vekten på benkpress og følte meg sterk.
•	NULL	NULL	NULL	NULL	NULL

Forklaring av tabeller:

Brukere

Tabellen «brukere» er hvor alle brukere ligger registrert. Her finner man både trenere og utøvere. Alle brukere har sin egen ID i tillegg til nyttig informasjon som navn, alder og kjønn. I en «faktisk» situasjon ville for eksempel et treningssenter opprettet nye brukere her ved innmelding som enten utøver eller trener.

Økt:

Tabellen «økt» inneholder informasjon rundt økten som en bruker skal gjennomføre, eller har gjennomført. Hver økt har sin egen ID samt informasjon som økttype, bruker-id, dato og notater. Her er det meningen at en bruker kan se hvilke type økt hen skal gjennomføre, og skrive noen notater til hvordan hen syntes den gikk etter. Som sett over i view-et ligger mesteparten av denne informasjonen der, for oversiktighet, men føres inn i denne tabellen.

Øvelser:

Tabellen «øvelser» er rett og slett «biblioteket» for alle mulige øvelser som en bruker kan gjennomføre. Hver øvelse har egen ID i tillegg til navnet på øvelsen og en beskrivende tekst som forteller litt om den. Denne er for eksempel fin å bruke for de som er nye til trening og lurer på hvordan en øvelse skal gjøres.

Øvelseslogg:

Tabellen «øvelsesLogg» fungerer som en logg til hver øvelse en bruker har gjort. Hver logg har sin egen ID, samt; bruker-id, øvelse-id, dato og relevant informasjon til øvelsen som; repetisjoner, vekt, puls_sone og distanse. I et faktisk tilfelle fyller man da inn distanse på øvelser som løping, jogging og svømming, og repetisjoner og vekt på styrkeøvelser. Loggen legger grunnlaget for progresjonsoversikten til en bruker.

Programmer:

Tabellen «programmer» inneholder alle opprettede programmer som en utøver skal følge. Hvert program har sin egen ID, i tillegg til program-navn og informasjon om oppvarming og avslutning. Øvelsene som skal gjennomføres kommer i neste avsnitt. I vårt tilfelle kalte vi bare programmene ukedagene som ble oppgitt i oppgaveteksten.

Her kan derimot en eventuell trener navngi den noe annet, som for eksempel: «Leg day»

Program øvelser:

Tabellen «program_øvelser» henger sammen med «programmer» av den grunn at alle øvelsene som skal gjøres til hvert program legges her. Tabellen har derfor flere program-id-er som skal samsvare med én i «programmer». I kolonnen ved siden av ligger øvelses-id-ene som skal gjennomføres til hver program-id.

Uke oversikt:

Tabellen «ukeOversikt» viser en oversikt på syv dager om gangen med informasjon rundt hvilken ukeplan en utøver skal følge. Hver ukeoversikt har sin egen ID, og informasjon om hvilke utøver som skal følge den, samt hvilke plan utøveren skal følge, i tillegg til hvilke trener som har angitt dette, alternativt hvilke trener som skal følge utøveren den uken. Til slutt har man startdato og sluttdato for uken det gjelder.

Ukeplan:

Tabellen «ukePlan» er en oppslagstabell med to kolonner. Første kolonne er primærnøkkel og er «PlanID» i andre kolonne finner man «program_id». Plan-id brukes derfor til å si hvilke program man skal følge, som sett i uke oversikt.

Ukeplan ukedager:

Tabellen «ukePlan_UkeDager» henger sammen med «ukePlan» hvor denne tabellen viser hvilke planID som skal brukes på hvilken ukedag(dato). Her kan f.eks. en trener angi planer til ulike datoer, for deretter tildele BrukerID i «ukeOversikt»

Konklusjon:

Gjennom denne oppgaven har vi laget et databasesystem til trenere og utøvere ved å designe en strukturert treningsplanlegging- og registreringsdatabase. Dette oppnådde vi ved å utvikle et system som gir trenere og utøvere muligheten til å enkelt opprette, redigere og holde oversikt over treningsøkter og fremgang over tid.

Oppgaven vår inkluderer en grundig kravspesifikasjon, normalisering av databasen, design av en Entitet-Relasjon (ER) modell, implementering i MySQL og fylling av databasen med dummy data. Videre har vi utformet SQL-spørringer og views for å demonstrere systemets funksjonalitet, noe som illustrerer hvordan den eventuelt ville blitt brukt i «den virkelige verden».

Gjennom denne oppgaven har vi ført og fremst laget et databasesystem som samsvarer med utdelt case og dermed oppgavebeskrivelsen, samtidig som det har vært enormt lærerikt for oss. Vi har fått verdifull erfaring ved å designe dette databasesystemet og implementere det fra bunnen av. Den grundige normaliseringsprosessen og utviklingen av ER-modellen har gitt oss et bedre grunnlag for å forstå de underliggende prinsippene for effektiv databaseadministrasjon.

Referanseliste

OpenAI. (n.d.). *ChatGPT*. Chat.openai.com; OpenAI. Hentet 8. mars 2024, fra

<https://chat.openai.com/>

Kojevnikov, A. (n.d.). *Source code beautifier / syntax highlighter – convert code*

snippets to HTML «hilite.me. Hilite.me. Hentet 10. mars 2024, fra

<http://hilite.me/>

ChatGPT spørring 1:

Basert på dette, generer opp mer tilfeldig dummy data til tabellen, verdiene under kjon skal enten være "D" eller "M": INSERT INTO bruker (brukerID, fornavn, etternavn, brukerType, alder, kjonnn) VALUES (1, 'Nancy', 'Gerlach', 'Kunde', 26, 'D')

Svar:

Here's an SQL INSERT INTO script for the "bruker" table:

INSERT INTO bruker (brukerID, fornavn, etternavn, brukerType, alder, kjonnn) **VALUES**

INSERT INTO brukere (brukerID, fornavn, etternavn, brukerType, alder, kjonnn) **VALUES**

(1, 'Nancy', 'Gerlach', 'Kunde', 26, 'D'),

(2, 'Marcella', 'Parisian', 'Trener', 26, 'D'),

(3, 'Guillermo', 'Homenick', 'Kunde', 38, 'D'),

(4, 'Jared', 'West', 'Kunde', 29, 'M'),

(5, 'Janie', 'Hilpert', 'Kunde', 60, 'D'),

(6, 'Francis', 'Champlin', 'Trener', 24, 'M'),

(7, 'Ramon', 'Osinski', 'Trener', 27, 'M'),

(8, 'Stewart', 'Howell', 'Kunde', 21, 'M'),

(9, 'Sue', 'Gutkowski', 'Trener', 41, 'D'),

(10, 'Oscar', 'Kub', 'Trener', 37, 'M'),

(11, 'Frankie', 'Kilback', 'Trener', 56, 'D'),

(12, 'Jared', 'Walter', 'Trener', 34, 'D'),

(13, 'Kathleen', 'Kirlin', 'Kunde', 18, 'D'),

(14, 'Mike', 'Stoltenberg', 'Trener', 33, 'D'),

(15, 'Joanne', 'Padberg', 'Kunde', 42, 'M'),

(16, 'Jimmy', 'Weissnat', 'Trener', 31, 'D'),
(17, 'Sharon', 'Koss', 'Kunde', 59, 'D'),
(18, 'Norman', 'Rosenbaum', 'Kunde', 56, 'D'),
(19, 'Clark', 'Lesch', 'Trener', 35, 'D'),
(20, 'Gerald', 'Hamill', 'Kunde', 40, 'D'),

(21, 'Martin', 'Olsen', 'Trener', 27, 'M')

(22, 'Robert', 'Mccullough', 'Kunde', 45, 'M'),
(23, 'Dorothy', 'Matthews', 'Kunde', 32, 'D'),
(24, 'Alan', 'Simpson', 'Trener', 39, 'M'),
(25, 'Denise', 'Mcdonald', 'Kunde', 50, 'D'),
(26, 'Martin', 'Ferguson', 'Trener', 36, 'M'),
(27, 'Alice', 'Campbell', 'Kunde', 48, 'D'),
(28, 'Paul', 'Bowman', 'Trener', 29, 'M'),
(29, 'Lori', 'Huff', 'Kunde', 55, 'D'),
(30, 'Terry', 'Floyd', 'Kunde', 43, 'M'),
(31, 'Teresa', 'Olson', 'Kunde', 34, 'D'),
(32, 'Ralph', 'Gonzalez', 'Trener', 47, 'M'),
(33, 'Stephanie', 'Rice', 'Kunde', 30, 'D'),
(34, 'Lawrence', 'Parker', 'Kunde', 41, 'M'),
(35, 'Kathryn', 'Armstrong', 'Kunde', 37, 'D'),
(36, 'Louis', 'Black', 'Kunde', 52, 'M'),
(37, 'Melissa', 'Dean', 'Trener', 44, 'D'),
(38, 'Billy', 'Bates', 'Kunde', 26, 'M'),
(39, 'Christina', 'Reid', 'Kunde', 49, 'D'),
(40, 'Philip', 'Chavez', 'Trener', 33, 'M'),
(41, 'Mary', 'Wagner', 'Kunde', 31, 'D'),
(42, 'Geraldine', 'Harris', 'Kunde', 58, 'D'),
(43, 'Ronald', 'Rivera', 'Kunde', 46, 'M'),
(44, 'Melinda', 'Peters', 'Trener', 35, 'D'),
(45, 'Bruce', 'Mcguire', 'Kunde', 39, 'M'),
(46, 'Wanda', 'Hansen', 'Kunde', 53, 'D'),
(47, 'Harry', 'Hunt', 'Trener', 29, 'M'),
(48, 'Gloria', 'Fields', 'Kunde', 42, 'D'),

(49, 'Roger', 'Bryant', 'Trener', 37, 'M'),
(50, 'Evelyn', 'Duncan', 'Kunde', 44, 'D');

This script will insert all the provided data into the "bruker" table, assuming that the columns are correctly set up in your database.

ChatGPT spørring 2:

Kan du legge til dummydata i disse kommentarene maks 200 tegn. Egentlig bare en kort oppsummering på hvordan en hypotetisk treningsøkt gikk (etter dato) : 1 Styrkeøkt 1 2024-03-11 2 Kondisjonsøkt 1 2024-03-12 3 Kondisjonsøkt 1 2024-03-13 4 Styrkeøkt 1 2024-03-14 5 Kondisjonsøkt 1 2024-03-15 6 Kondisjonsøkt 1 2024-03-16 7 Konkurrans 1 2024-03-17 8 Kondisjonsøkt 2 2024-03-11 9 Styrkeøkt 2 2024-03-12 10 Konkurrans 2 2024-03-13 11 Kondisjonsøkt 2 2024-03-14 12 Styrkeøkt 2 2024-03-15 13 Konkurrans 2 2024-03-16 14 Styrkeøkt 2 2024-03-17 15 Styrkeøkt 3 2024-03-11 16 Kondisjonsøkt 3 2024-03-12 17 Konkurrans 3 2024-03-13 18 Styrkeøkt 3 2024-03-14 19 Kondisjonsøkt 3 2024-03-15 20 Konkurrans 3 2024-03-16 21 Konkurrans 3 2024-03-17

Svar:

- 1 Styrkeøkt gikk bra, økte vekten på benkpress og følte meg sterk.
- 2 Hadde en intensiv kondisjonsøkt, klarte å forbedre tidene mine.
- 3 Enda en god kondisjonsøkt, holdt en jevn intensitet gjennom hele økten.
- 4 Sterk styrkeøkt, økte vektene og forbedret formen min.
- 5 God kondisjonsøkt, fokuserte på utholdenhetstrening.
- 6 Fullførte en annen vellykket kondisjonsøkt, følte meg energisk.
- 7 Konkurrans gikk bra, presterte bra og følte meg tilfreds.
- 8 Solid kondisjonsøkt, jobbet med å forbedre min utholdenhet.
- 9 Styrkeøkt var utfordrende, økte vekter og fokuserte på form.
- 10 Konkurransen var tøff, men jeg klarte å presse meg gjennom.
- 11 Kondisjonsøkten var intens, klarte å slå mine personlige rekorder.
- 12 Sterk styrkeøkt, forbedret styrken og teknikken min.
- 13 Konkurransen var krevende, men jeg ga mitt beste.
- 14 Styrkeøkten var utfordrende, men jeg følte meg sterk og fokusert.
- 15 Gjennomførte en god styrkeøkt, økte vektene gradvis.

- 16 Kondisjonsøkten var bra, klarte å opprettholde en god intensitet.
- 17 Konkurransen gikk bra, presterte godt under press.
- 18 Styrkeøkten var intens, fokuserte på kjerneøvelser.
- 19 Jobbet med utholdenheten i dagens kondisjonsøkt.
- 20 Konkurransen var tøff, men jeg kjempet meg gjennom.
- 21 Konkurransen gikk bra, følte meg godt forberedt og fokusert.

ChatGPT spørring 3:

Basert på dette, generer opp mer tilfeldig dummy data til tabellen, hvis det er verdi i distansekolonene skal det ikke være verdier i repetisjoner og vekt kolonnen og visa versa:

```
INSERT INTO bruker (LogID, BrukerID, OvelseID, dato, repetisjoner, vekt, puls_sone,
distanse) VALUES (1, 1, 1, '2024-03-11', 'NULL', 'NULL', '120-140', '2km'), (2, 1, 9, '2024-03-11', 8, 20, '100-120', 'NULL')
```

Svar:

Here's an example of generating random dummy data for your table based on the criteria you provided:

```
INSERT INTO bruker (LogID, BrukerID, OvelseID, dato, repetisjoner, vekt, puls_sone,
distanse)
```

VALUES

```
(1, 1, 1, '2024-03-11', NULL, NULL, '120-140', '2km'),
(2, 1, 9, '2024-03-11', 8, 20, '100-120', NULL),
(3, 1, 9, '2024-03-11', 8, 30, '100-120', NULL),
(4, 1, 9, '2024-03-11', 10, 35, '100-120', NULL),
(5, 1, 10, '2024-03-11', 8, 20, '100-120', NULL),
(6, 1, 10, '2024-03-11', 10, 20, '100-120', NULL),
(7, 1, 10, '2024-03-11', 10, 20, '100-120', NULL),
(8, 1, 11, '2024-03-11', 8, 20, '100-120', NULL);
```

This script inserts dummy data into the bruker table based on the provided data, following the pattern where either repetisjoner and vekt columns have values or the distanse column has a value for each row. Adjust as needed for your specific requirements.