Musikkregister

1.

```
mycursor.execute("""
CREATE TABLE IF NOT EXISTS artist (
    id INT AUTO_INCREMENT PRIMARY KEY,
    navn VARCHAR(100) NOT NULL
)
""")

mycursor.execute("""
CREATE TABLE IF NOT EXISTS sang (
    id INT AUTO_INCREMENT PRIMARY KEY,
    tittel VARCHAR(100) NOT NULL,
    artist_id INT,
    FOREIGN KEY (artist_id) REFERENCES artist(id)
)
""")
```

```
+------------------+
| Tables_in_musikk |
+------------------+
| artist           |
| sang             |
+------------------+
2 rows in set (0.001 sec)

MariaDB [musikk]>
```

Denne koden oppretter en tabell for sanger og artister. Den oppretter også automatisk ved oppstart.

2.

```
def legg_til_artist():
    navn = input("Skriv inn artistnavn: ")
    mycursor.execute("INSERT INTO artist (navn) VALUES (%s)", (navn,))
    mydb.commit()
    print("Artist lagt til!\n")
```

For å opprette en kode som legger til artist må du lage en funksjon som legger til artisten. Vi lager en input, du skriver inn navnet. Denne inputten blir inserta inn i tabellen for artister og printer ut at artisten er lagt til.

```
while True:
    print("""
----- MUSIKKREGISTER -----
1. Legg til artist
2. Legg til sang
3. Vis artister
4. Vis sanger
5. Oppdater artist
6. Oppdater sang
7. Avslutt
""")
```

Du trenger hovedsakelig denne menyen, men vi starter med å bare sette søkelys på 1. Legg til artist.

```
MariaDB [musikk]> SELECT * FROM artist
    -> ;
+----+--------------+
| id | navn         |
+----+--------------+
|  1 | Drake        |
|  2 | Travis Scott |
|  3 | Pop Smoke    |
+----+--------------+
3 rows in set (0.001 sec)
```

Som du kan se har vi lagt til 3 artister ved hjelp av koden vi skrev.

3.

```
def legg_til_sang():
    tittel = input("Skriv inn sangtittel: ")
    artist_id = input("Skriv inn artist ID: ")
    mycursor.execute("INSERT INTO sang (tittel, artist_id) VALUES (%s, %s)", (tittel, artist_id))
    mydb.commit()
    print("Sang lagt til!\n")
```

Her er koden vi har lagd for å legge til sanger. Dette er også en funksjon. Du skriver inn sangtittel og artist id for å vise hvilke sanger som tilhører hvilke artister. Til slutt printer det ut at sanger har blitt lagt til.

```
Velg et tall 1-7: 2
Skriv inn sangtittel: For the night
Skriv inn artist ID: 3
Sang lagt til!
```

Her er hvordan det skal se ut.

```
ERROR 1146 (42S02): Table 'musikk.sanger' doesn't exist
MariaDB [musikk]> SELECT * FROM sang
    -> ;
+----+-----------------+-----------+
| id | tittel          | artist_id |
+----+-----------------+-----------+
|  2 | Passionfruit    |         1 |
|  3 | Fancy           |         1 |
|  4 | Butterfly Effect|         2 |
|  5 | Goosebumps      |         2 |
|  6 | Dior            |         3 |
|  7 | For the night   |         3 |
+----+-----------------+-----------+
6 rows in set (0.001 sec)
```

Her viser tabellen hvilke sanger som tilhører hvilken artist id.

4.

```
4
5  def vis_artister():
6      mycursor.execute("SELECT * FROM artist")
7      result = mycursor.fetchall()
8      print("\n--- ARTISTER ---")
9      for x in result:
0          print(f"{x[0]}: {x[1]}")
1      print("")
2
3
4
5  def vis_sanger():
6      mycursor.execute("""
7      SELECT s.id, s.tittel, a.navn
8      FROM sang s
9      JOIN artist a ON s.artist_id = a.id
0      """)
1      result = mycursor.fetchall()
2      print("\n--- SANGER ---")
3      for x in result:
4          print(f"{x[0]}: {x[1]} ({x[2]})")
5      print("")
6
```

Her er koden for å vise artister og sanger.  Første linjen viser tabellene. Deretter printer koden ut tabellen av artister og sanger. Til slutt går den gjennom alle x verdiene en etter en og lister dem. Skjermbildene over viser hvordan det ser listet ut.

5.

```
def oppdater_artist():
    artist_id = input("Skriv ID til artisten du vil endre: ")
    nytt_navn = input("Skriv nytt navn: ")
    mycursor.execute("UPDATE artist SET navn=%s WHERE id=%s", (nytt_navn, artist_id))
    mydb.commit()
    print("Artist oppdatert!\n")



def oppdater_sang():
    sang_id = input("Skriv ID til sangen du vil endre: ")
    ny_tittel = input("Skriv ny tittel: ")
    mycursor.execute("UPDATE sang SET tittel=%s WHERE id=%s", (ny_tittel, sang_id))
    mydb.commit()
    print("Sang oppdatert!\n")
```

Her er koden for å oppdatere artist og sang. Du skriver ID til artist/sang du vil endre og deretter skriver du nytt navn. Veldig simpel kode og dette kommer til å erstatte den forrige sangen/artisten. Mye av koden over består av inputs, siden det er dette vi bruker.

Skjermbildene viser også noen oppdaterte artister/sanger som jeg hadde byttet ut med tidligere.

En ting til jeg vil vise er hovedmenyen.

```
while True:
    print("""
----- MUSIKKREGISTER -----
1. Legg til artist
2. Legg til sang
3. Vis artister
4. Vis sanger
5. Oppdater artist
6. Oppdater sang
7. Avslutt
""")

    valg = input("Velg et tall 1-7: ")

    if valg == "1":
        legg_til_artist()
    elif valg == "2":
        legg_til_sang()
    elif valg == "3":
        vis_artister()
    elif valg == "4":
        vis_sanger()
    elif valg == "5":
        oppdater_artist()
    elif valg == "6":
        oppdater_sang()
    elif valg == "7":
        print("Programmet avsluttes...")
        break
    else:
        print("Feil valg, prøv igjen!\n")

mydb.close()
```

Her er koden for den.  Du får en input hvor du velger et tall mellom 1-7. Her bruker vi if og elif for å vise alternativer du kan velge. Hvis du velger 7 avslutter programmet og da bruker vi break. Hvis du ikke skriver et tall 1-7 så får du feil melding.

6.

```
MariaDB [musikk]> SELECT * FROM sang
    -> ;
+----+------------------+-----------+
| id | tittel           | artist_id |
+----+------------------+-----------+
|  2 | Passionfruit     |         1 |
|  3 | Fancy            |         1 |
|  4 | Butterfly Effect |         2 |
|  5 | Goosebumps       |         2 |
|  6 | Dior             |         3 |
|  7 | For the night    |         3 |
+----+------------------+-----------+
6 rows in set (0.001 sec)
```

Jeg restartet også programmet og som du kan se lagres alt i databasen. Du kan avslutte og restarte men viser du listene står det fortsatt alt som har blitt lagret fra før. (Dessverre kan jeg ikke vise mer siden jeg ikke kan koble til pien ettersom at jeg ikke er i klassen mens jeg skriver det her). Jeg har testet dette flere ganger.

Mye av koden her bruker vi input for det er slik vi gjør det. Noe annet vi bruker ofte er mysql spørringer som er viktig for denne oppgaven. Vi bruker simple spørringer som vi kan deretter skrive inn på command prompt og få opp alle tabellene. Slik som SELECT * FROM sang. Det viktigste vi bruker i dette programmet er mysql.connector.connect. Dette er slik vi kobler til databasen for at koden kan bruke mycursor og mydb for å sende sql kommandoer.  Man kobler først til databasen. Så oppretter man tabeller. Deretter funksjoner for å legge til, vise eller oppdatere. Til slutt hele menyen.