NTNU
Fakultet for informasjons-
teknologi og elektroteknikk
Institutt for datateknologi
og informatikk

Egen databaseløsning

Øving 8 – IDATT2103

Innlevert av:
Nicolai Thorer Sivesind
Erlend Rønning
Aleksander Brekke Røed

15. november, 2021

Foreleser: Tore Mallaug

Behovsbeskrivelse

I denne oppgaven skal vi opprette en database for å holde oversikt over Formel 1-lag og biloppsettet deres.

Et F1-lag består av tre typer ansatte: teamleder, sjåfør og øvrig ansatt. For at et lag skal være komplett må det ha nøyaktig én team-leder og to sjåfører. Et lag kan ellers ha et vilkårlig antall ansatte. I tillegg innehar et F1-lag opptil flere biler. Hver sjåfør må ha en funksjonell bil for at de skal kunne delta i et race.

En F1-bil består av 5 deler: et karosseri, et motorsystem, en vinge bak, en vinge foran og et sett med sidevinger. Alle skal registreres med vekt.

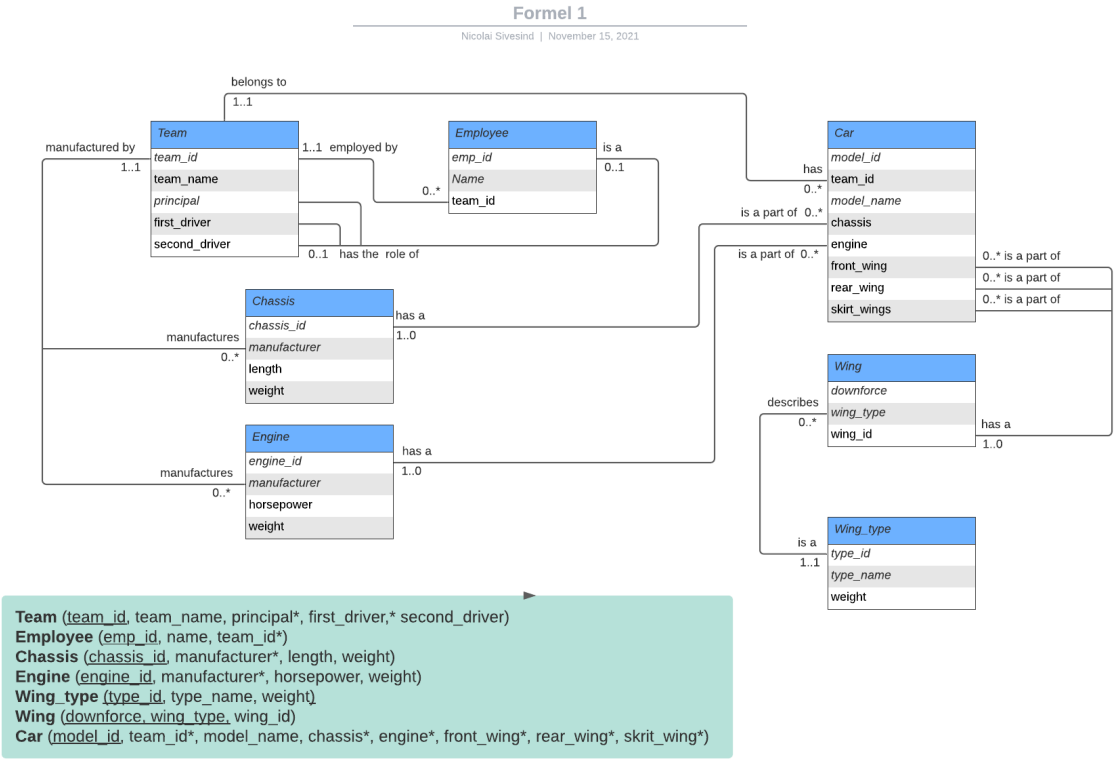
En type karosseri identifiseres ved et karosserinummer og tilhører kun ett lag. Et karosseri registreres med lengde. Flere biler kan ha samme karosseri.

Motorsystemet produseres også av lagene, men lag som ikke produserer egne motorsystem kan inngå avtale med produsentlag for å bruke deres motorer. En motor må registreres med antall hestekrefter.

Vi er kun ute etter noen spesifikke egenskaper ved vingene. Lagene kan derfor identifisere hva slags vinger de bruker ved en kombinasjon av marktrykk og vingetype for hver vinge. Det vil altså si at en vinge kan brukes av flere lag. Det finnes 3 typer vinger – en foran, en bak og et sett med sidevinger. Siden en vinges marktrykk er proporsjonal med vinkelen på bladene som utgjør vingen, er vekten derfor kun bestemt av hva slags type vinge det er.

Siden bildeler blir satt inn og byttet ut fortløpende, kan en bil være registrert uten deler. En bil er kun funksjonell hvis den består har alle 5 deler og har en total vekt som er høyere enn 750 kg.

ER-diagram



Team (team_id, team_name, principal*, first_driver,* second_driver)
Employee (emp_id, name, team_id*)
Chassis (chassis_id, manufacturer*, length, weight)
Engine (engine_id, manufacturer*, horsepower, weight)
Wing_type (type_id, type_name, weight)
Wing (downforce, wing_type, wing_id)
Car (model_id, team_id*, model_name, chassis*, engine*, front_wing*, rear_wing*, skrit_wing*)

Spørringer:

Hente ut praktisk team informasjon:

```
188    #  View of team information
189 •  CREATE VIEW team_info AS SELECT
190        team_name,
191        (SELECT full_name FROM employee WHERE emp_id = principal) as principal,
192        (SELECT full_name FROM employee WHERE emp_id = first_driver) as first_driver,
193        (SELECT full_name FROM employee WHERE emp_id = second_driver) as second_driver,
194        (SELECT COUNT(employee.emp_id) FROM employee WHERE employee.team_id = team.team_id) AS employees
195    FROM team;
196
```

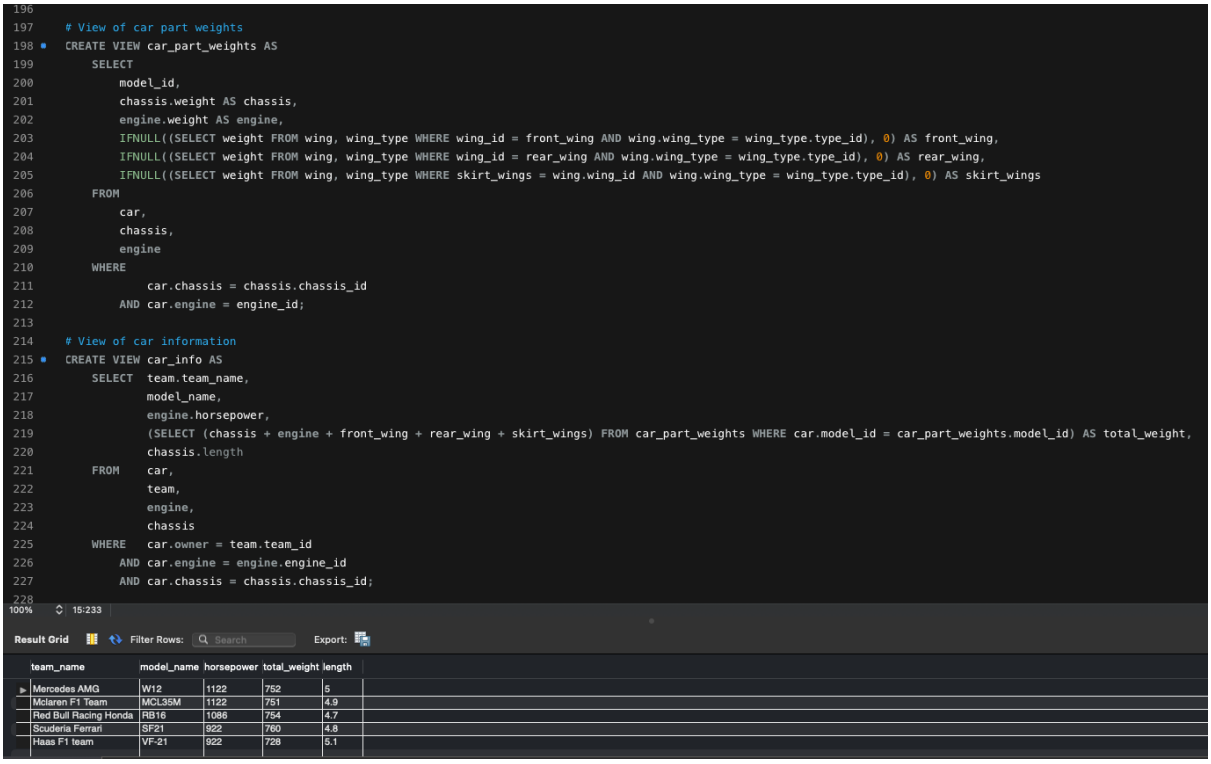
100% 18:230

Result Grid Filter Rows: Search Export:

team_name	principal	first_driver	second_driver	employees
Mercedes AMG	Toto Wolff	Lewis Hamilton	Valtteri Bottas	4
Mclaren F1 Team	Zak Brown	Lando Norris	Daniel Ricciardo	3
Red Bull Racing Honda	Christian Horner	Max Verstappen	Sergio Perez	3
Scuderia Ferrari	Mattia Binotto	Charles Leclerc	Carlos Sainz	3
Haas F1 team	Günter Steiner	Mick Schumacher	Nikita Mazepin	3

Her lager vi et VIEW som bruker de ulike fremmednøklene til å bytte ut identifikatorene med navnene til de ansatte som er tildelt roller. I tillegg legger vi til hvor mange ansatte hvert lag har. Bildet er en SELECT-spørring av dette viewet.

Hente ut praktisk informasjon om de ulike bilene:



```
196
197   # View of car part weights
198   CREATE VIEW car_part_weights AS
199       SELECT
200           model_id,
201           chassis.weight AS chassis,
202           engine.weight AS engine,
203           IFNULL((SELECT weight FROM wing, wing_type WHERE wing_id = front_wing AND wing.wing_type = wing_type.type_id), 0) AS front_wing,
204           IFNULL((SELECT weight FROM wing, wing_type WHERE wing_id = rear_wing AND wing.wing_type = wing_type.type_id), 0) AS rear_wing,
205           IFNULL((SELECT weight FROM wing, wing_type WHERE skirt_wings = wing.wing_id AND wing.wing_type = wing_type.type_id), 0) AS skirt_wings
206       FROM
207           car,
208           chassis,
209           engine
210       WHERE
211               car.chassis = chassis.chassis_id
212           AND car.engine = engine_id;
213
214   # View of car information
215   CREATE VIEW car_info AS
216       SELECT  team.team_name,
217               model_name,
218               engine.horsepower,
219               (SELECT (chassis + engine + front_wing + rear_wing + skirt_wings) FROM car_part_weights WHERE car.model_id = car_part_weights.model_id) AS total_weight,
220               chassis.length
221       FROM    car,
222               team,
223               engine,
224               chassis
225       WHERE   car.owner = team.team_id
226           AND car.engine = engine.engine_id
227           AND car.chassis = chassis.chassis_id;
228
100%  15:233
Result Grid  Filter Rows: Search  Export:
```

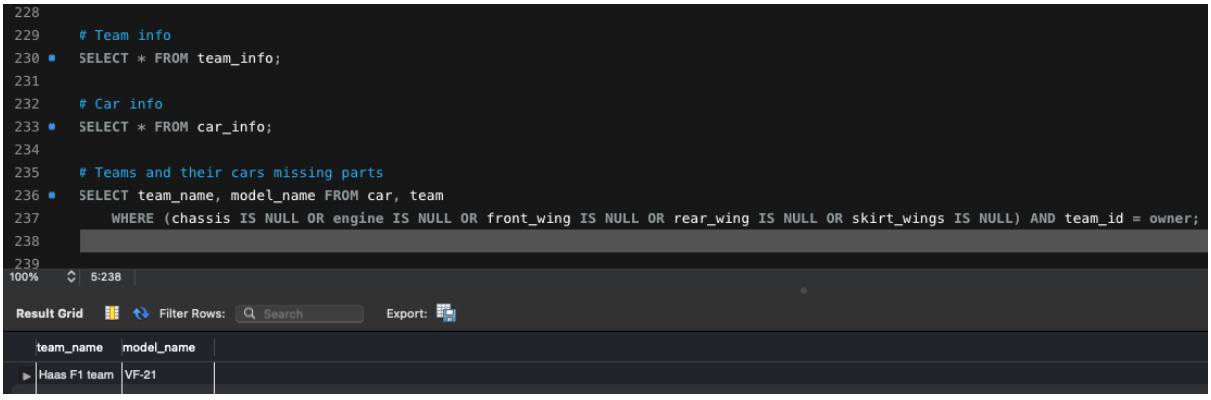
team_name	model_name	horsepower	total_weight	length
Mercedes AMG	W12	1122	752	5
Mclaren F1 Team	MCL35M	1122	751	4.9
Red Bull Racing Honda	RB16	1086	754	4.7
Scuderia Ferrari	SF21	922	760	4.8
Haas F1 team	VF-21	922	728	5.1

Her lager vi først et VIEW som gir oss en oversikt over hvor mye hver del på de ulike bilene veier.

Deretter lager vi enda et VIEW som gir oss praktisk informasjon om bilene. Her bruker vi ulike fremmed-nøkler til å hente ut relevant informasjon fra tabellene som blir referert til fra Bil-tabellen. Vi bruker også det første viewet vi lagde til å regne ut bilens totale vekt.

Bildet er en SELECT-spørring av det siste viewet.

Sjekke om det er noen biler som mangler deler:



```
228
229   # Team info
230   SELECT * FROM team_info;
231
232   # Car info
233   SELECT * FROM car_info;
234
235   # Teams and their cars missing parts
236   SELECT team_name, model_name FROM car, team
237       WHERE (chassis IS NULL OR engine IS NULL OR front_wing IS NULL OR rear_wing IS NULL OR skirt_wings IS NULL) AND team_id = owner;
238
239
100%  5:238
Result Grid  Filter Rows: Search  Export:
```

team_name	model_name
Haas F1 team	VF-21

Her sjekker vi om noen av bilene har minst ett av del-feltene sine tomme. I tillegg henter vi også ut navnet til laget som bilen tilhører.

Erfaringer:

Vi synes løsningen vår er robust og fleksibel. Som vi ser i noen av spørringene må vi gjøre litt ekstra arbeid for å hente ut den praktiske informasjonen. Dette er blant annet fordi vi jobber med mye informasjon som er spredt utover mange tabeller og har derfor mange fremmednøkler. Bil-tabellen har for eksempel 6 fremmednøkler. Ellers er informasjonen fordelt på en logisk måte som gjør at det er enkelt å legge til nye instanser i tabellene.

Istedenfor at vi lagrer vekt som et felt i 'vingetype', 'motor' og 'karosseri', kunne vi hatt en sterk entitetstype 'bildeler' som holder denne vekten og så la 'vingetype', 'motor' og 'karosseri' være svake entitetstyper som har bildel-identifikatoren som primær og fremmednøkkel. Ulempen med dette er at vi lagrer bildel-identifikatoren to steder. En gang i bildel og en gang som fremmednøkkel i de svake typene nevnt over. Vi risikerer ikke å få feilaktig informasjon, men det er likevel overflødig.

Både i Team-tabellen og bil-tabellen har vi flere kolonner som er fremmednøkler til samme tabell. Dette gjør det litt kronglete å hente ut instansspesifikk informasjon fra tabellen som blir referert til i flere felt. Vi må derfor bruke delspørringer. Fordelen med dette er at vi får et 0..1 forhold i disse tilfellene som er hva vi ønsker.

XML

Positive sider kan være at i et trelags klient/tjener-arkitektur kan XML brukes som:

- En god systemuavhengig måte å beskrive, eller representere, data som skal lagres.
- En systemuavhengig måte å sende data mellom klienter og webtjenester på.

En negativ side ved XML vil være at hvis en bruker relasjonsdatabaser, kan mapping ned til tabeller og SQL være nødvendig hvis databasesystemet ikke tilbyr tilstrekkelig funksjonalitet for XML.