A picture containing table

Description automatically generated

Foreleser: Tore Mallaug

Nicolai Thorer Sivesind

Erlend Rønning

Aleksander Brekke Røed

**Egen databaseløsning**

**Øving 8 – IDATT2103**

15. november, 2021

Innlevert av:

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

Diagram

Description automatically generated

Spørringer:

**Hente ut praktisk team informasjon:**

Graphical user interface

Description automatically generated

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:**

Text

Description automatically generated

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:**

A screenshot of a computer

Description automatically generated with medium confidence

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.