***Projektoplæg.***

Denne opgave omhandler implementering af et databaseinterface i Java. Databasen som der skal laves interface til er den samme som i opgave 2. Løsningen til denne opgave vil blive benyttet i det afsluttende projekt i forbindelse med kurset 02324 Videreg˚aende Programmering. Opgaven kan ogs˚a løses helt uafhængigt af 02324, hvis du ikke følger det kursus. De to opgaver skal være en del af jeres analyseafsnit i jeres rapport og skal forstås 99,9 % før i går videre. ☺

1. **Opgave.**

Betragt en database med følgende skema:

operatoer(opr\_id : integer; opr navn : string; ini : string; cpr : string; password : string)

raavare(raavare\_id : integer; raavare navn : string; leverandoer : string)

raavarebatch(rb\_id : integer; raavare\_id : integer; maengde : real)

recept(recept\_id : integer; recept\_navn : string)

receptkomponent(recept\_id : integer; raavare\_id : integer; nom\_netto : real; tolerance : real)

produktbatch(pb\_id : integer; recept\_id : integer; status : integer)

produktbatchkomponent(pb\_id : integer; rb\_id : integer; opr\_id : integer; tara : real; netto : real)

Databasen har følgende fremmednøgler:

* + Feltet *raavare\_id* er en fremmednøgle i raavarebatch med reference til raavare.
  + Feltet *raavare\_id* er en fremmednøgle i receptkomponent med reference til raavare.
  + Feltet *recept\_id* er en fremmednøgle i receptkomponent med reference til recept.
  + Feltet *recept\_id* er en fremmednøgle i produktbatch med reference til recept.
  + Feltet *pb\_d* er en fremmednøgle i produktbatchkomponent med reference produktbatch.
  + Feltet *rb\_id* er en fremmednøgle i produktbatchkomponent med reference til raavarebatch.
  + Feltet *opr\_id* er en fremmednøgle i produktbatchkomponent med reference til operatoer.

Databasen indeholder information relateret til en virksomhed som producerer forskellige varer efter en række givne recepter. Herunder beskrives først de enkelte relationer i databasen.

**Relationen** operatoer

N˚ar en vare skal produceres skal de enkelte ingredienser fra recepten afvejes af en operatør. Information om operatørerne ligger i relationen operatoer. En tuppel i denne relation repræsenterer en operatør angivet ved sit identifikationsnummer (*opr id* ), sit navn (*opr\_navn*), sine initialer (*ini* ), sit CPR-nummer (*cpr* ) og sit password (*password* ).

**Relationen** raavare

De varer som virksomheden producerer best˚ar af en række r˚avarer. Relationen raavare indeholder oplysninger om disse r˚avarer. En tuppel i raavare angiver en r˚avares identifikationsnummer (*raavare\_id* ), dens navn (*raavare\_navn*) og navnet p˚a leverandøren af r˚avaren (*leverandoer*)

**Relationen** raavarebatch

Lagerbeholdningen af de enkelte r˚avarer er angivet i relationen raavarebatch. En tuppel i denne relation angiver et parti (batch) af en bestemt r˚avare. Tuplen indeholder et identifikationsnummer for r˚avarebatchet (*rb\_id* ), et identifikationsnummer for r˚avaren (*raavare\_id* ) og en mængde (*maengde*). Hvis en tuppel (*i, j, k*) er indeholdt i raavarebatch betyder det at lageret indeholder et parti med *k* kilogram af r˚avaren *j* og at dette parti har det entydige identifikationsnummer *i*. Bemærk at r˚avaren er angivet ved sit *raavare\_id*, da dette er nøglen i relationen raavare. Hvis vi til relationen raavarebatch ønsker at tilføje information om navnene p˚a r˚avarerne og deres leverandører kan vi forme det naturlige join raavare *t><l* raavarebatch.

**Relationen** recept

Relationen recept indeholder en oversigt over navnene p˚a de recepter (opskrifter) som virksomheden benytter, n˚ar de skal producere deres varer. En tuppel i recept angiver en recepts identifikationsnummer (*recept\_id* ) samt dens navn (*recept\_navn*). Feltet *recept\_navn* angiver navnet p˚a den vare som recepten angiver opskriften p˚a.

**Relationen** receptkomponent

Relationen receptkomponent angiver de enkelte ingredienser (r˚avarer) der indg˚ar i re- cepterne. En tuppel i receptkomponent svarer til en enkelt ingrediens i en enkelt recept. Tuplen indeholder identifikationsnummeret for recepten (*recept\_id* ), identifikationsnummeret for ingrediensen (*raavare\_id* ), den nominelle vægt (*nom netto*) samt en vægttolerance (*tolerance*). Hvis en tuppel (*i, j, k, l*) er indeholdt i receptkomponent betyder det at der i recepten *i* indg˚ar *k* kilogram af r˚avaren *j* og at den højst tilladte vægtafvigelse er *l* ki- logram. Hvis vi til relationen receptkomponent ønsker at tilføje information om navnene p˚a de enkelte recepter kan vi forme det naturlige join recept *t><l* receptkomponent.

**Relationen** produktbatch

Virksomheden benytter recepterne repræsenteret i relationen recept *t><l* receptkomponent til at producere deres varer. N˚ar virksomheden producerer et parti (batch) af en bestemt vare bliver det registreret i relationen produktbatch. En tuppel i produktbatch angiver et partis identifikationsnummer (*pb\_id* ), identifikationsnummeret for den benyttede recept (*recept id* ) samt en statusangivelse (*status*). Feltet *status* har værdien 0 n˚ar produktio- nen af det p˚agældende parti endnu ikke er p˚abegyndt, værdien 1 n˚ar produktionen er p˚abegyndt men ikke afsluttet, og værdien 2 n˚ar produktionen er afsluttet.

**Relationen** produktbatchkomponent

Relationen produktbatchkomponent angiver de enkelte ingredienser anvendt til at producere partierne i produktbatch. En tuppel i produktbatchkomponent svarer til en enkelingrediens i et enkelt parti af en produceret vare. Alle ingredienser er taget fra lagerbeholdningen af r˚avarer, s˚a ingredienserne refereres til med r˚avarebatch identifikationsnumre. En tuppel i produktbatchkomponent indeholder identifikationsnummeret for produktpar- tiet (*pb\_id* ), identifikationsnummeret for r˚avarebatchet (*rb\_id* ), identifikationsnummeret for operatøren som har afvejet ingrediensen (*opr\_id* ), samt nettovægt (*netto*) og taravægt (*tara*) for afvejningen.

**Eksempel-instanser**

Du indlæse eksempel-instanserne ved, at downloade filen create all tables.sql fra fildeling og dernæst i MySQL udføre følgende kommando:

source create\_all\_tables.sql

**Opgaven**

Udtryk nedenst˚aende forespørgsler i SQL og afprøv dem i MySQL. De svar som MySQL giver p˚a dine forespørgsler skal indg˚a som del af opgavebesvarelsen. Husk at skrive forkla- rende tekst til hver af dine SQL-forespørgsler!

1. Bestem navnene p˚a de r˚avarer som indg˚ar i mindst to forskellige r˚avarebatches.
2. Bestem relationen som for hver receptkomponent indeholder tuplen (*i, j, k*) best˚aende af receptens identifikationsnummer *i*, receptens navn *j* og r˚avarens navn *k*.
3. Find navnene p˚a de recepter som indeholder mindst ´en af følgende to ingredienser (r˚avarer): skinke eller champignon.Find navnene p˚a de recepter som indeholder b˚ade ingrediensen skinke og ingrediensen champignon.
4. Find navnene p˚a de recepter som *ikke* indeholder ingrediensen champignon.
5. Find navnene p˚a den eller de recepter som indeholder den største nominelle vægt af ingrediensen tomat.
6. Bestem relationen som for hver produktbatchkomponent indeholder tuplen (*i, j, k*) best˚aende af produktbatchets identifikationsnummer *i*, r˚avarens navn *j* og r˚avarens nettovægt *k*.
7. Find identifikationsnumrene af den eller de produktbatches som indeholder

en størst nettovægt af ingrediensen tomat.

1. Find navnene p˚a alle operatører som har været involveret i at producere

partier af varen margherita.

1. Bestem relationen som for hver produktbatchkomponent indeholder

tuplen (*i, j, k,l, m, n*) best˚aende af produktbatchets identifikationsnummer *i*,

produktbatchets status *j*, r˚avarens navn *k*, r˚avarens nettovægt *l*, den

tilhørende recepts navn *m* og operatørens navn *n*.

Udtryk nedenst˚aende forespørgsler i SQL og afprøv dem i MySQL.

Q1. Angiv antallet af produktbatchkomponenter med en nettovægt p˚a mere end 10. *Hint* : Brug aggregatfunktionen COUNT.

Q2. Find den samlede mængde af tomat som findes p˚a lageret, dvs. den samlede mængde af tomat som optræder i raavarebatch-tabellen. *Hint* : Brug aggregatfunktionen SUM.

Q3. Find for hver ingrediens (r˚avare) den samlede mængde af denne ingrediens som findes p˚a lageret. *Hint* : Modific´er forespørgslen ovenfor. Brug GROUP BY.

Q4. Find de ingredienser (navne p˚a r˚avarer) som indg˚ar i mindst tre forskellige recepter.

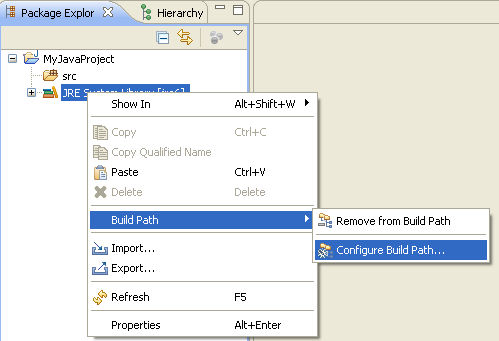
# Opgave 2

I det følgende forudsættes det at du benytter udviklingsmiljøet Eclipse og Windows. Benytter du et andet udviklingsmiljø kan der eventuelt være nogle afvigelser fra beskrivelsen nedenfor.

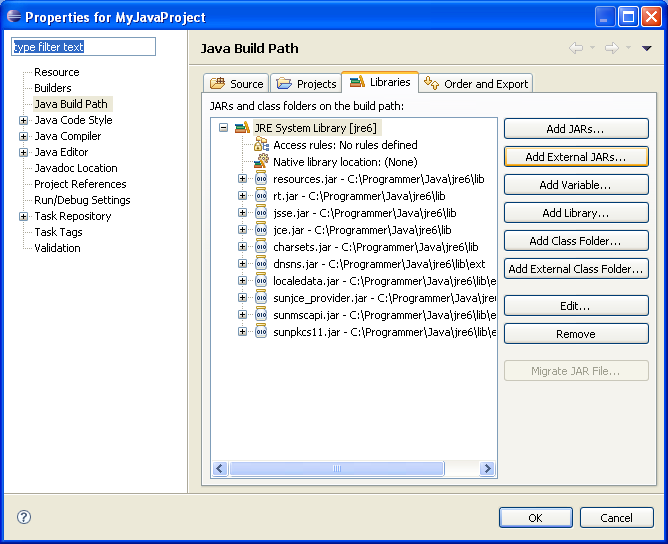
Hvis du ikke allerede har et passende Java-projekt under Eclipse til at indeholde koden til denne opgave, s˚a start med at oprette et nyt. Du skal nu have fat i JDBC-driveren og f˚a tilføjet den til Java-projektets bibliotek. For at gøre dette skal du først downloade driveren som ligger i samme katalog som denne hjemmeopgave under fildeling p˚a CampusNet. Filen hedder:

mysql-connector-java-3.0.17-ga-bin.jar

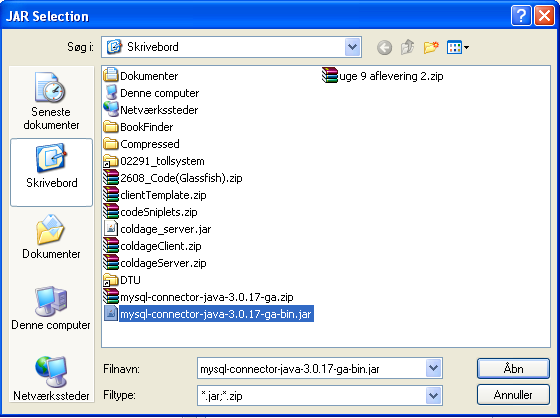
N˚ar du har downloadet filen s˚a højreklik p˚a kataloget ‘JRE System Library’ i dit Java- projekt under Eclipse. Vælg dernæst build path -> configure build path.



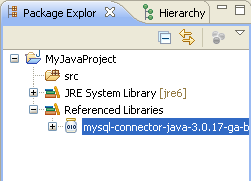
Vælg nu fanebladet ‘Libraries’.



Klik p˚a ‘Add External JARs...’ og navig´er dernæst hen til den jar-fil du netop har down- loadet fra fildeling.



Dit Java-projekt skulle nu gerne indeholde et katalog kaldet ‘Referenced Libraries’ med en reference til den hentede fil, hvilket betyder at du har f˚aet tilføjet JDBC-driveren til dit projekt.



Du skal nu downloade endnu en fil fra fildeling p˚a CampusNet:

LabDatabase.zip

Denne zip-fil indeholder en delvis implementation af databaseinterfacet. N˚ar du har down- loadet zip-filen (uden at pakke den ud!) skal du importere den i dit Java-projekt ved at højreklikke p˚a projekt-kataloget, vælge ‘Import’ og dernæst ‘General/Archive File’. Klik ‘Next’, navig´er hen til den downloadede fil og klik til slut ‘Finish’.

Du burde nu have f˚aet tilføjet følgende pakker til dit projekt-katalog:

connector01917 daoimpl01917 daointerfaces01917 dto01917

test01917

Pakken connector01917 indeholder connector-klassen Connector (jvf. ugens slides). Den indeholder desuden en klasse Constant. I klassen Constant skal du indsætte navnet p˚a database-serveren du vil skabe forbindelse til, navnet p˚a databasen du vil benytte, samt dit brugernavn og password. Du burde nu kunne eksekvere Main-klassen i pakken test01917. Afprøv det og se om det virker

Pakken dto01917 indeholder DTO’erne (data transfer objects) hørende til databasen. Der er en DTO-klasse hørende til hver af de 7 relationerne i den database som opgaven omhandler (jvf. databasens skema angivet i opgave 1 ovenfor):

OperatoerDTO ProduktBatchDTO

ProduktBatchKompDTO RaavareBatchDTO RaavareDTO ReceptDTO ReceptKompDTO

Kig p˚a koden til disse DTO-klasser og sørg for at du forst˚ar den.

De to pakker daointerfaces01917 og daoimpl01917 hører sammen. Den første in- deholder en række DAO-interfaces som daoimpl01917 skal implementere. I det udlevere- de kode er det kun interfacet OperatoerDAO som er implementeret. Se p˚a interfacene i daointerfaces01917 og implementationen af OperatoerDAO i daoimpl01917 og forsøg at skabe dig et overblik over tingene. Sørg især for at du forst˚ar alle detaljer i implementa- tionen MySQLOperatoerDAO af interfacet OperatoerDAO, da du ellers ikke vil være i stand til at løse denne opgave. Spørg eventuelt hjælpelæreren om hjælp.

Din opgave er nu følgende:

1. Implement´er de resterende interfaces fra daointerfaces01917. Implementationerne skal placeres i pakken daoimpl01917.
2. Lav en test-klasse som tester din implementation. En meget simpel test-klasse som tester implementationen af Operatoer er allerede lavet i Main-klassen af test01917- pakken.

Udover ovenst˚aende er du velkommen til at modificere i den udleverede kode med henblik p˚a at forbedre den.