Tredje undervisningsgang

Database

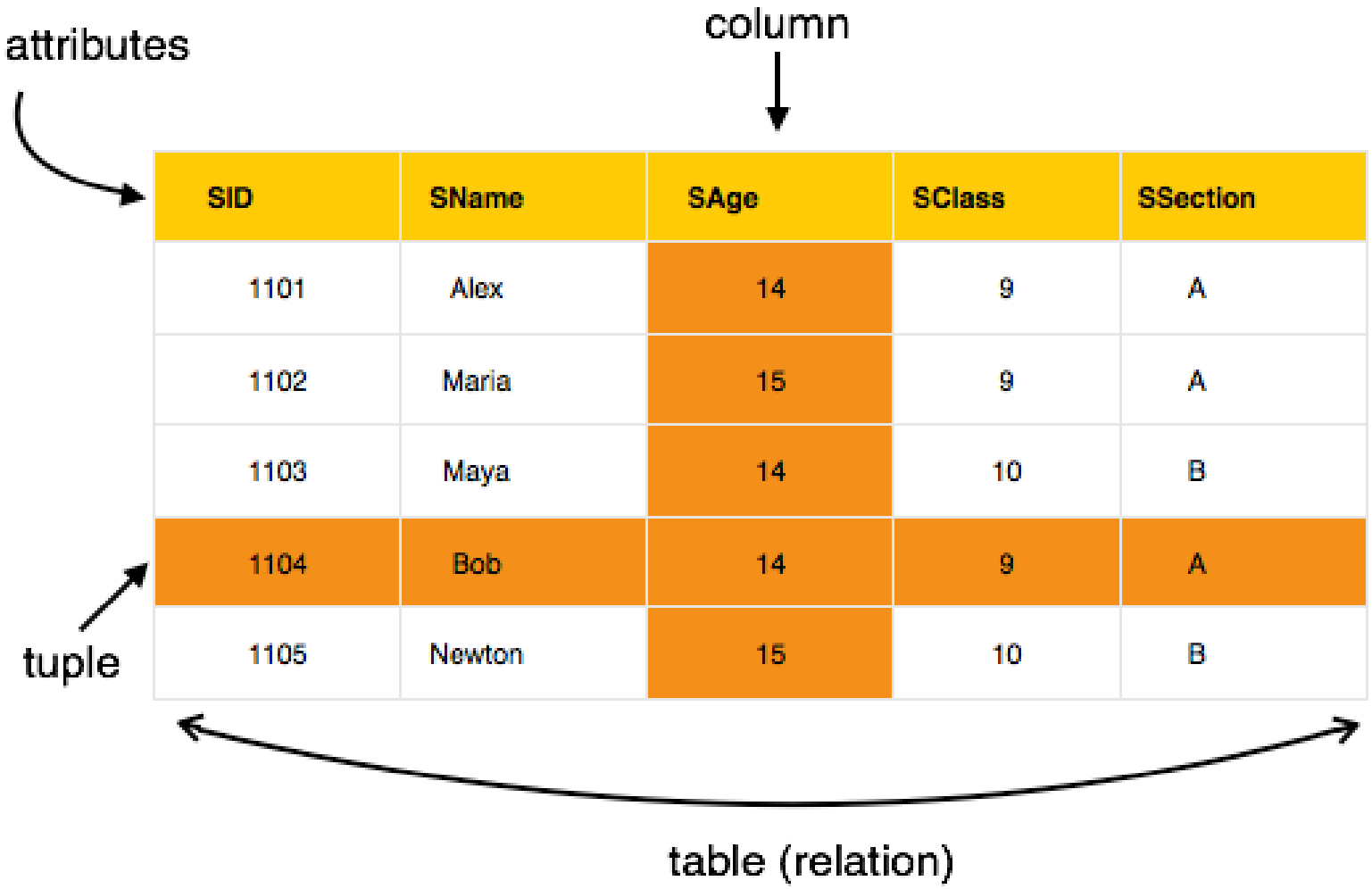
Denne undervisningsgang

- Lektier fra sidst
- Dagen eksempel database
  - Mere ER diagrammer
- Matematikken bag databaser
- Forbind data til ny sæt – praksis
- Keys og foreign keys - praksis
- At finde den rette data
- Transaktioner
- Databaser i web (PHP)
  - MySQLi og PDO
- Eksamen og krav

Lektier fra sidst

Det der er produceret vises og forklares til underviseren og resten af klassen

En tupel tydeliggjort



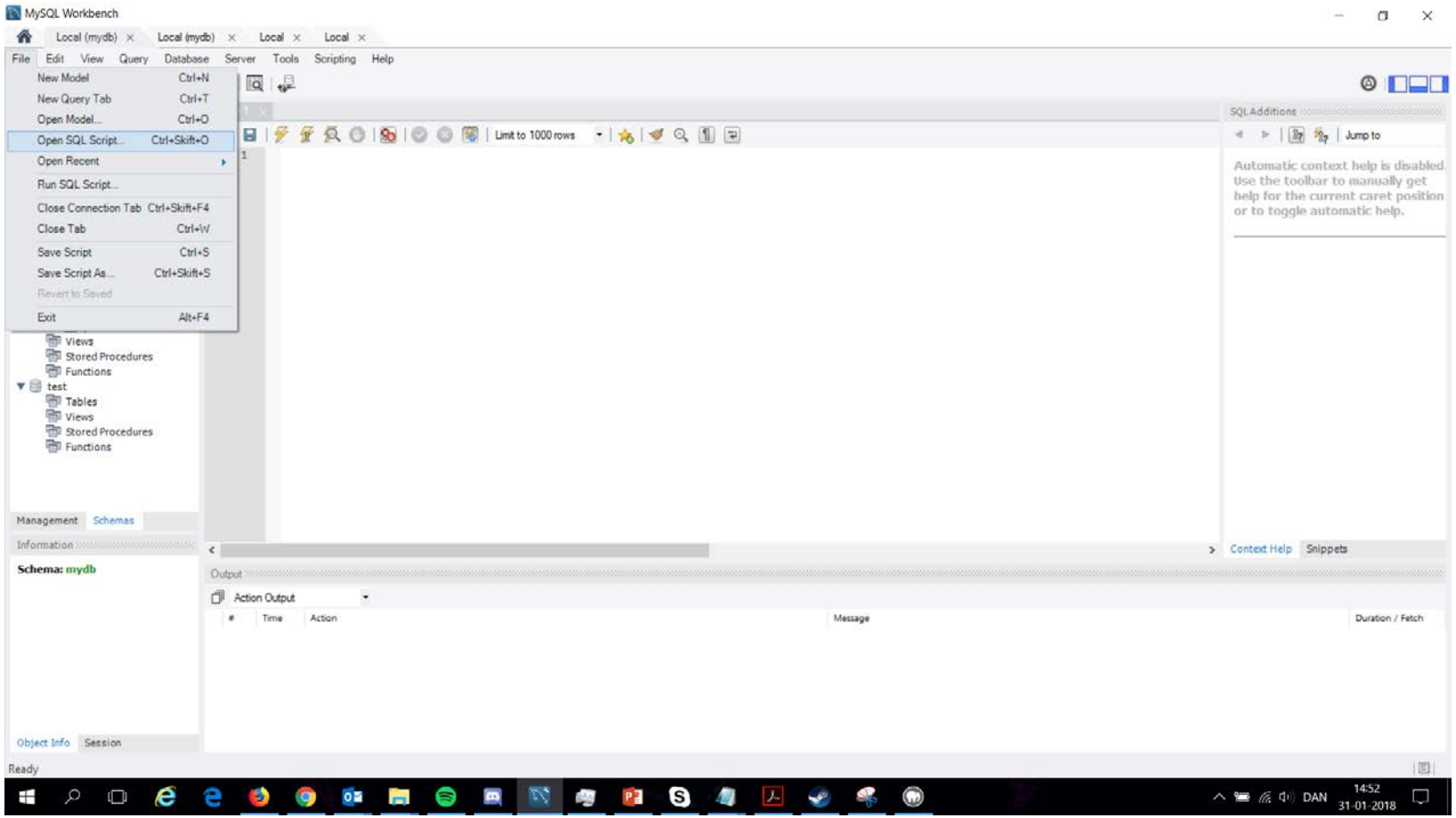
SID	SName	SAge	SClass	SSection
1101	Alex	14	9	A
1102	Maria	15	9	A
1103	Maya	14	10	B
1104	Bob	14	9	A
1105	Newton	15	10	B

Dagens eksempel database

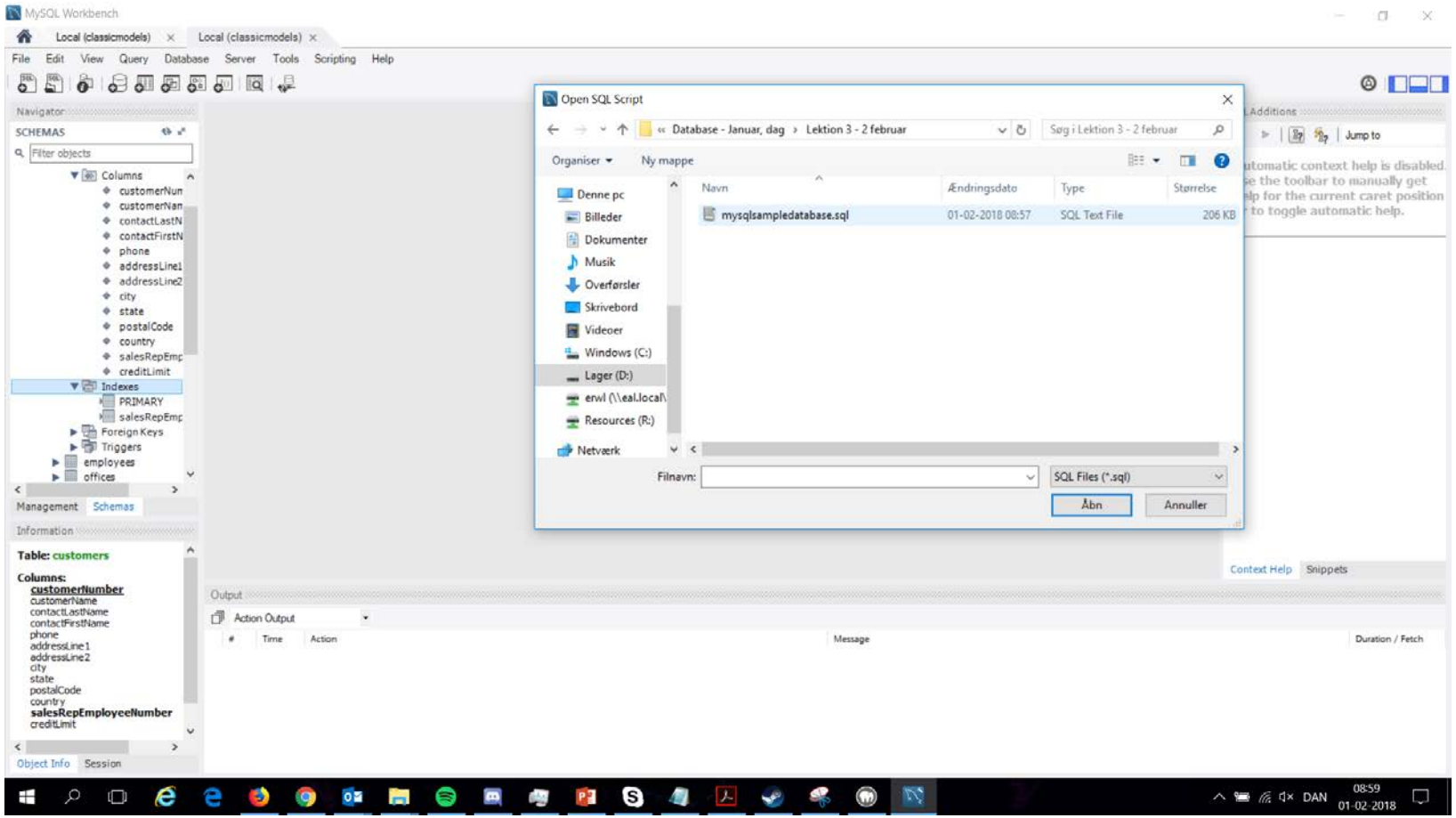
Hent mysqlsampledatabase.sql fra dagens lektion på Fronter og pak den ud

Start derefter MAMP og MySQL Workbench og åbn forbindelsen til din lokale databaseserver

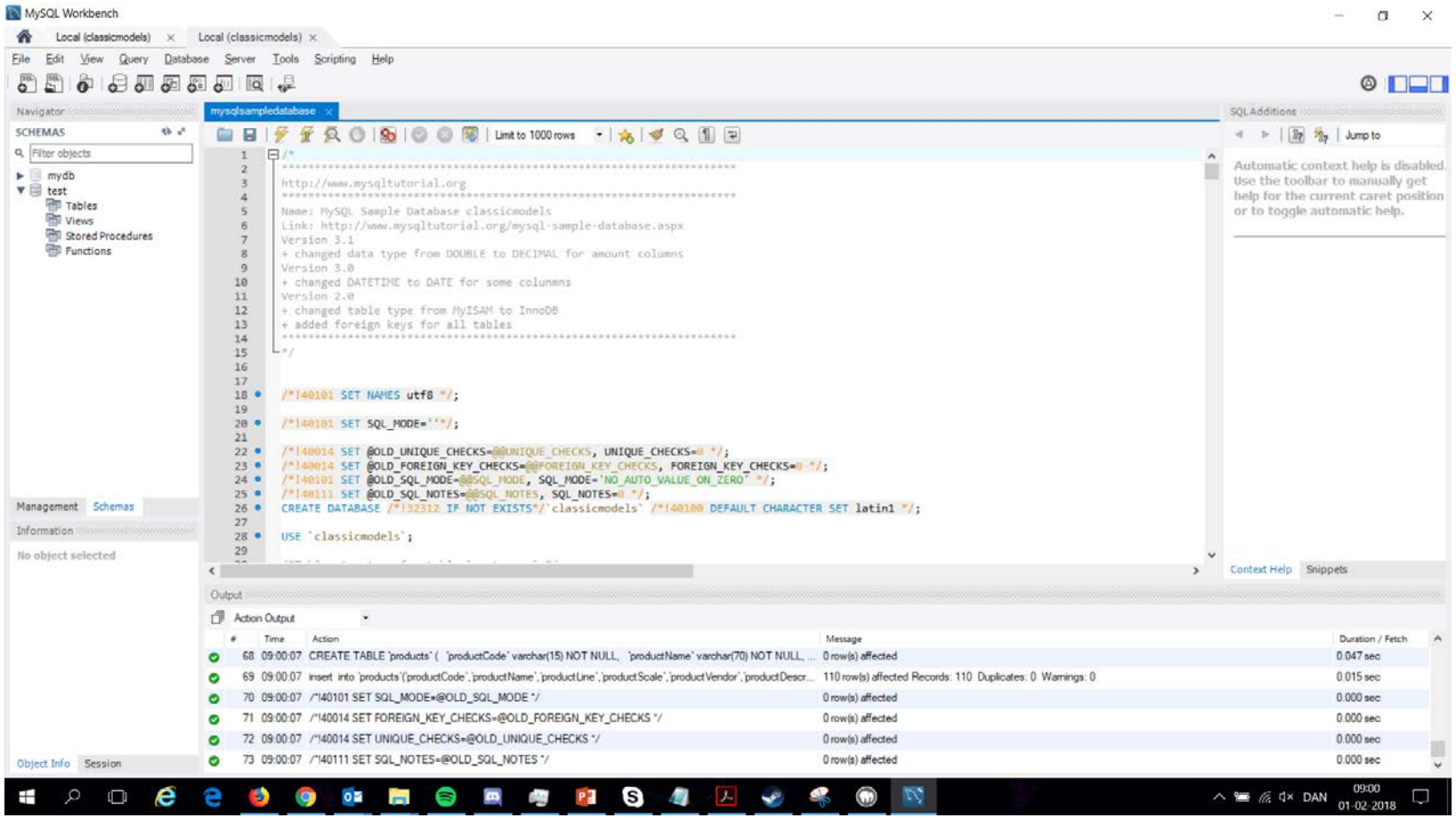
Åbn SQL Script



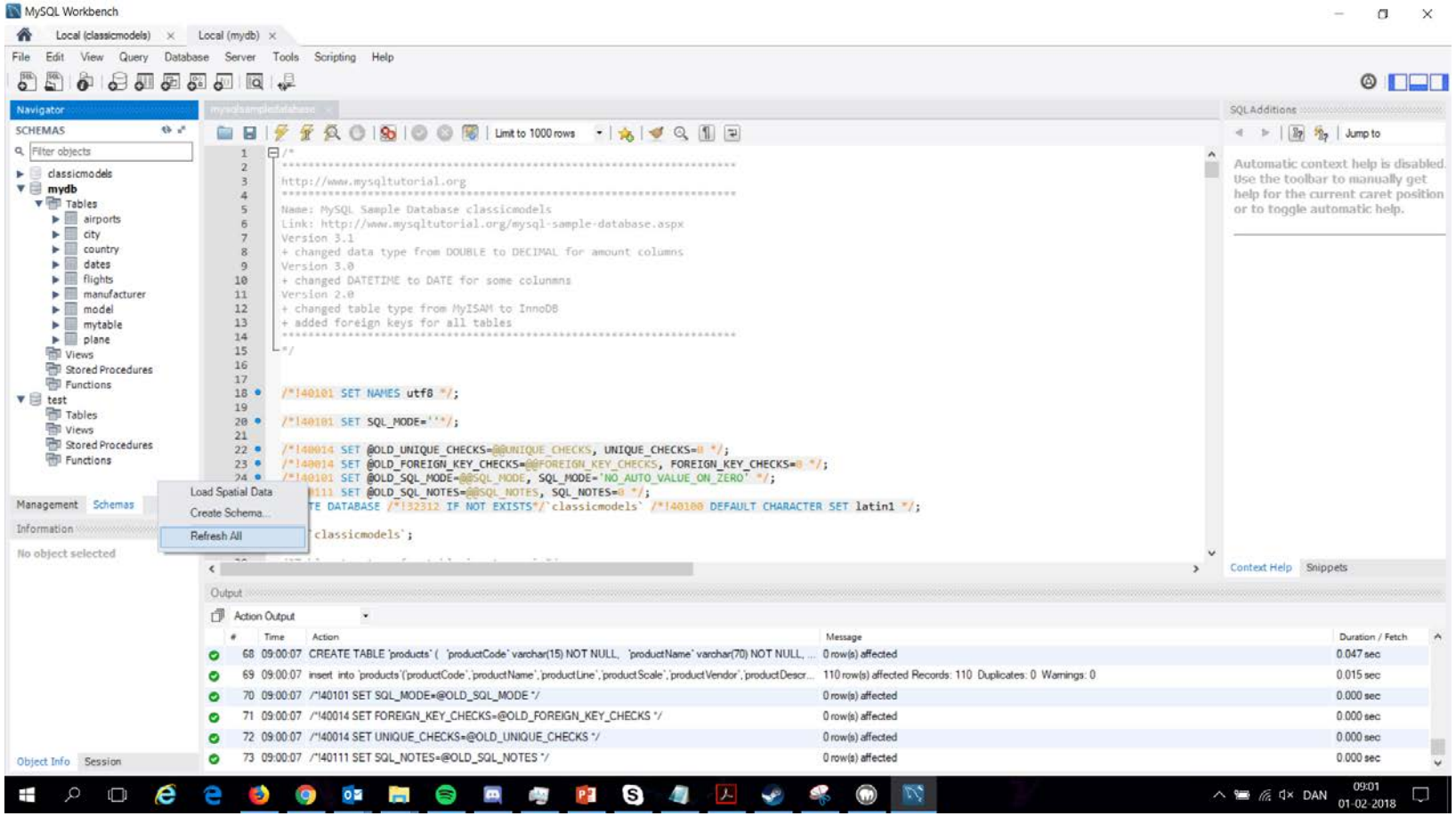
Vælg mysqlsampledatabase.sql



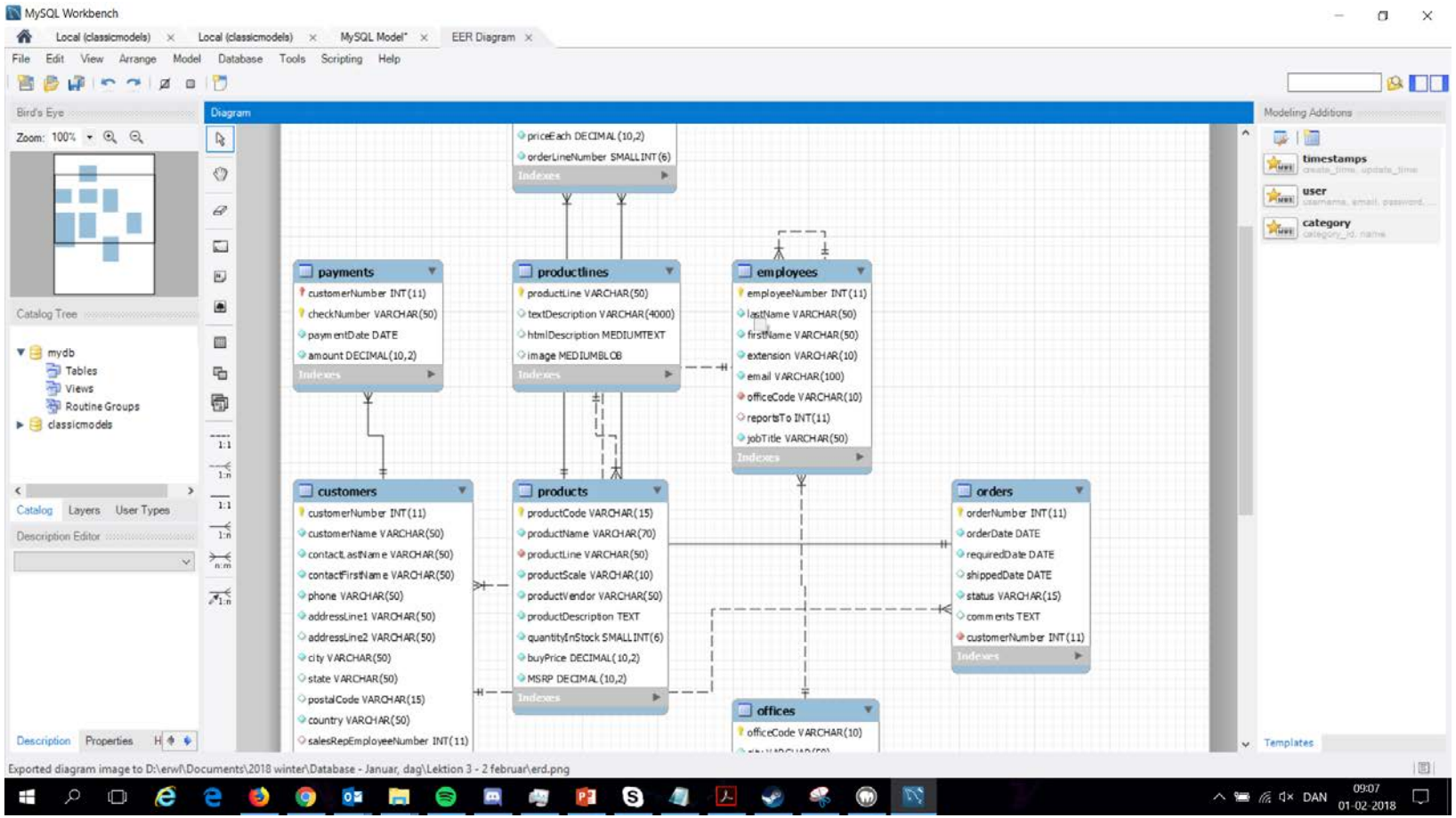
Kør scriptet



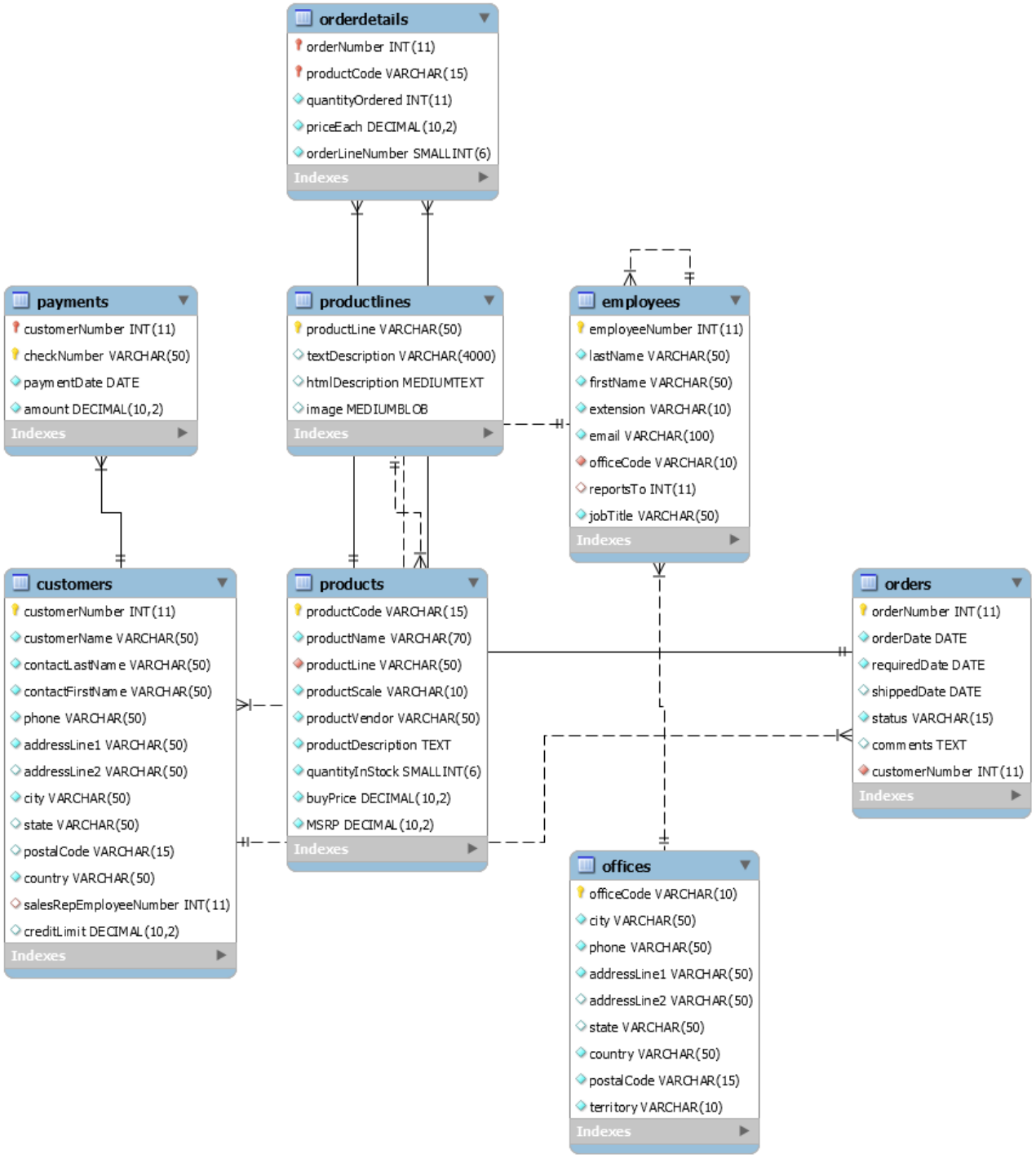
Hvis den ikke viser det ny classicmodels schema så bed den Refresh All



Du kan bede om at Reverse Engineer et ERD

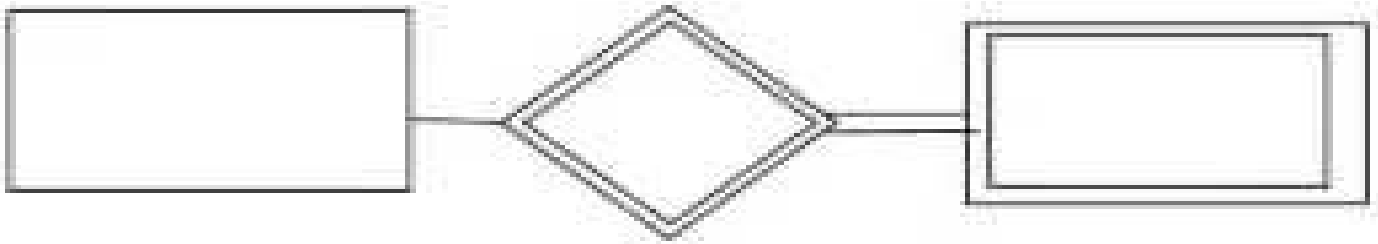


ERD


orderdetails
orderNumber INT(11)
productCode VARCHAR(15)
quantityOrdered INT(11)
priceEach DECIMAL(10,2)
orderLineNumber SMALLINT(6)
Indexes
payments
customerNumber INT(11)
checkNumber VARCHAR(50)
paymentDate DATE
amount DECIMAL(10,2)
Indexes
productlines
productLine VARCHAR(50)
textDescription VARCHAR(4000)
htmlDescription MEDIUMTEXT
image MEDIUMBLOB
Indexes
employees
employeeNumber INT(11)
lastName VARCHAR(50)
firstName VARCHAR(50)
extension VARCHAR(10)
email VARCHAR(100)
officeCode VARCHAR(10)
reportsTo INT(11)
jobTitle VARCHAR(50)
Indexes
customers
customerNumber INT(11)
customerName VARCHAR(50)
contactLastName VARCHAR(50)
contactFirstName VARCHAR(50)
phone VARCHAR(50)
addressLine1 VARCHAR(50)
addressLine2 VARCHAR(50)
city VARCHAR(50)
state VARCHAR(50)
postalCode VARCHAR(15)
country VARCHAR(50)
salesRepEmployeeNumber INT(11)
creditLimit DECIMAL(10,2)
Indexes
products
productCode VARCHAR(15)
productName VARCHAR(70)
productLine VARCHAR(50)
productScale VARCHAR(10)
productVendor VARCHAR(50)
productDescription TEXT
quantityInStock SMALLINT(6)
buyPrice DECIMAL(10,2)
MSRP DECIMAL(10,2)
Indexes
orders
orderNumber INT(11)
orderDate DATE
requiredDate DATE
shippedDate DATE
status VARCHAR(15)
comments TEXT
customerNumber INT(11)
Indexes
offices
officeCode VARCHAR(10)
city VARCHAR(50)
phone VARCHAR(50)
addressLine1 VARCHAR(50)
addressLine2 VARCHAR(50)
state VARCHAR(50)
country VARCHAR(50)
postalCode VARCHAR(15)
territory VARCHAR(10)
Indexes

Note om linje typer

- Vi tegner en massiv linje, hvis og kun hvis vi har et ID-afhængigt forhold; ellers ville det være en stiplet linje.
- Ved et svagt men ikke ID-afhængigt forhold tegnes en stiplet linje, fordi det er et svagt forhold.



Matematikken bag databaser

Relationel algebra og relationel calculus

Baggrund

- Relationelle databasesystemer forventes at være udstyret med et forespørgselssprog, som kan hjælpe deres brugere med at query (forespørge) databaserne forekomsterne.
- Der findes to slags forespørgselssprog – den relationelle algebra og de relationelle beregninger (calculus).

Den relationelle algebra

- Den relationelle algebra er et proceduremæssigt forespørgselssprog, der tager forekomster af relationer som input og giver forekomster af relationer som output.
- Den bruger operatører til at udføre forespørgsler.
  - En operatør kan enten være unær (unary) eller binær.
- Den accepterer relationer som input og giver relationer som output.
- Den relationelle algebra udføres rekursivt på en relation, og mellemresultater betragtes også som relationer.

Den relationelle algebra

De fundamentale operationer inden for den relationelle algebra er:

- Select
- Project
- Union
- Set difference
- Cartesian product
- Rename

Den relationelle algebra – Select operation (σ)

- Den vælger tupler, der tilfredsstiller det givne prædikat fra en relation.
- Notation – $\sigma_p(r)$
- σ står for valg prædikat og r står for relation.
- p er en præpositionel logisk formel, som kan bruge konnektorer som and, or, og not.
  - Disse udtryk kan bruge relationelle operatører som - =, ≠, ≥, <,>, ≤.

$\sigma_{subject = "database"}(Books)$

Output – Selects tuples from books where subject is 'database'.

$\sigma_{subject = "database" \text{ and } price = "450"}(Books)$

Output – Selects tuples from books where subject is 'database' and 'price' is 450.

$\sigma_{subject = "database" \text{ and } price = "450" \text{ or } year > "2010"}(Books)$

Output – Selects tuples from books where subject is 'database' and 'price' is 450 or those books published after 2010.

Den relationelle algebra – Project operation (∏)

- Den projekterer kolonne (r), der opfylder et givet prædikat.
- Notation - ΠA1, A2, An (r)
- Hvor A1, A2, An er attribut attributter af relation r.
- Duplicerende rækker elimineres automatisk, da forholdet er et sæt.

$$\Pi_{\text{subject, author}} (\text{Books})$$

Selects and projects columns named as subject and author from the relation Books.

Den relationelle algebra – Union operation (U)

- Udfører en binær union mellem to givne relationer og defineres som
  $r \cup s = \{ t \mid t \in r \text{ or } t \in s\}$
- Notation – r U s
- Hvor r og s er enten database relationer eller relations resultat sæt (midlertidig relation).
- For at en union skal være gyldig skal følgende betingelser opretholdes:
  - r og s skal have det samme antal attributter.
  - Attribut domæner skal være kompatible.
  - Duplikat tupler fjernes automatisk.

$$\prod_{\text{author}} (\text{Books}) \cup \prod_{\text{author}} (\text{Articles})$$

Output – Projects the names of the authors who have either written a book or an article or both.

Den relationelle algebra – Set difference (–)

- Resultatet af set difference er tupler, som er til stede i ét forhold, men ikke er i det andet forhold.
- Notation – r – s
- Finder alle de tupler, der er til stede i r, men ikke i s.

$$\prod_{\text{author}} (\text{Books}) - \prod_{\text{author}} (\text{Articles})$$

Output – Provides the name of authors who have written books but not articles.

Den relationelle algebra – Cartesian product (X)

- Kombinerer information af to forskellige relationer til en.
- Notation - r X s
- r og s er relationer, og deres output vil blive defineret som r X s = {q t | q ∈ r og t ∈ s}

$$\sigma_{author = 'tutorialspoint'}(Books \times Articles)$$

Output – Yields a relation, which shows all the books and articles written by tutorialspoint.

Den relationelle algebra – Rename operation (ρ)

- Resultaterne af den relationelle algebra er også relationer, men uden navn.
- Rename operationen giver os mulighed for at omdøbe outputforholdet.
- Rename operation er betegnet med det lille græsk bogstav rho ρ.
- Notation - ρ x (E)
- Hvor resultatet af udtryk E er gemt med navnet x.

Relationelle beregninger

- I modsætning til Relational Algebra er Relational Calculus et ikke-proceduremæssigt forespørgselssprog, det vil sige, det fortæller hvad man skal gøre, men forklarer aldrig, hvordan man gør det.
- Relationelle beregninger findes i to former:
  - Tuple Relational Calculus (TRC)
  - Domain Relational Calculus (DRC)

Relationelle beregninger - TRC

- Filtrering af variabelintervaller over tupler
- Notation - {T | Condition}
- Returnerer alle tupler T, der opfylder en tilstand.

```
{ T.name |  Author(T) AND T.article = 'database' }
```

Output – Returns tuples with 'name' from Author who has written article on 'database'.

- TRC kan kvantificeres. Vi kan bruge Existential (∃) og Universal Quantifiers (∀).

```
{ R| ∃T   ∈ Authors(T.article='database' AND R.name=T.name)}
```

Output – The above query will yield the same result as the previous one.

Relationelle beregninger - DRC

- I DRC bruger filtreringsvariablen domæne for attributter i stedet for hele tupel værdier (som udført i TRC, nævnt tidligere).
- Notation - {a1, a2, a3, ..., an | P (a1, a2, a3, ..., an)}
- Hvor a1, a2 er attributter og P står for formler bygget af indre attributter.

```
{< article, page, subject > |  ∈ TutorialsPoint ∧ subject = 'database'}
```

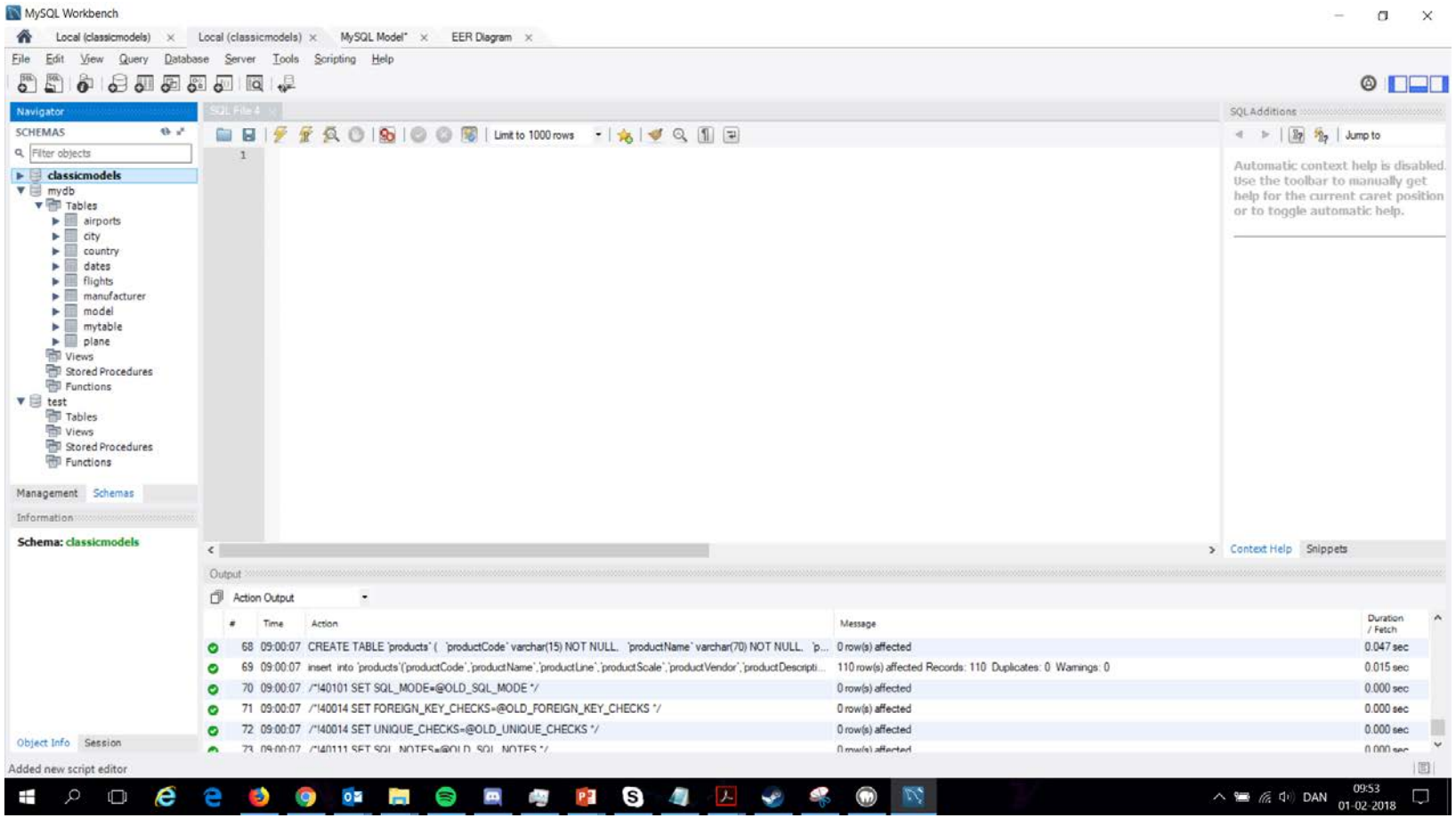
Output – Yields Article, Page, and Subject from the relation TutorialsPoint, where subject is database.

- Ligesom TRC kan DRC også skrives ved hjælp af eksistentielle og universelle kvantifikatorer. DRC involverer også relationelle operatører.
- Ekspressionsstyrken i Tuple Relation Calculus og Domain Relation Calculus svarer til Relational Algebra.

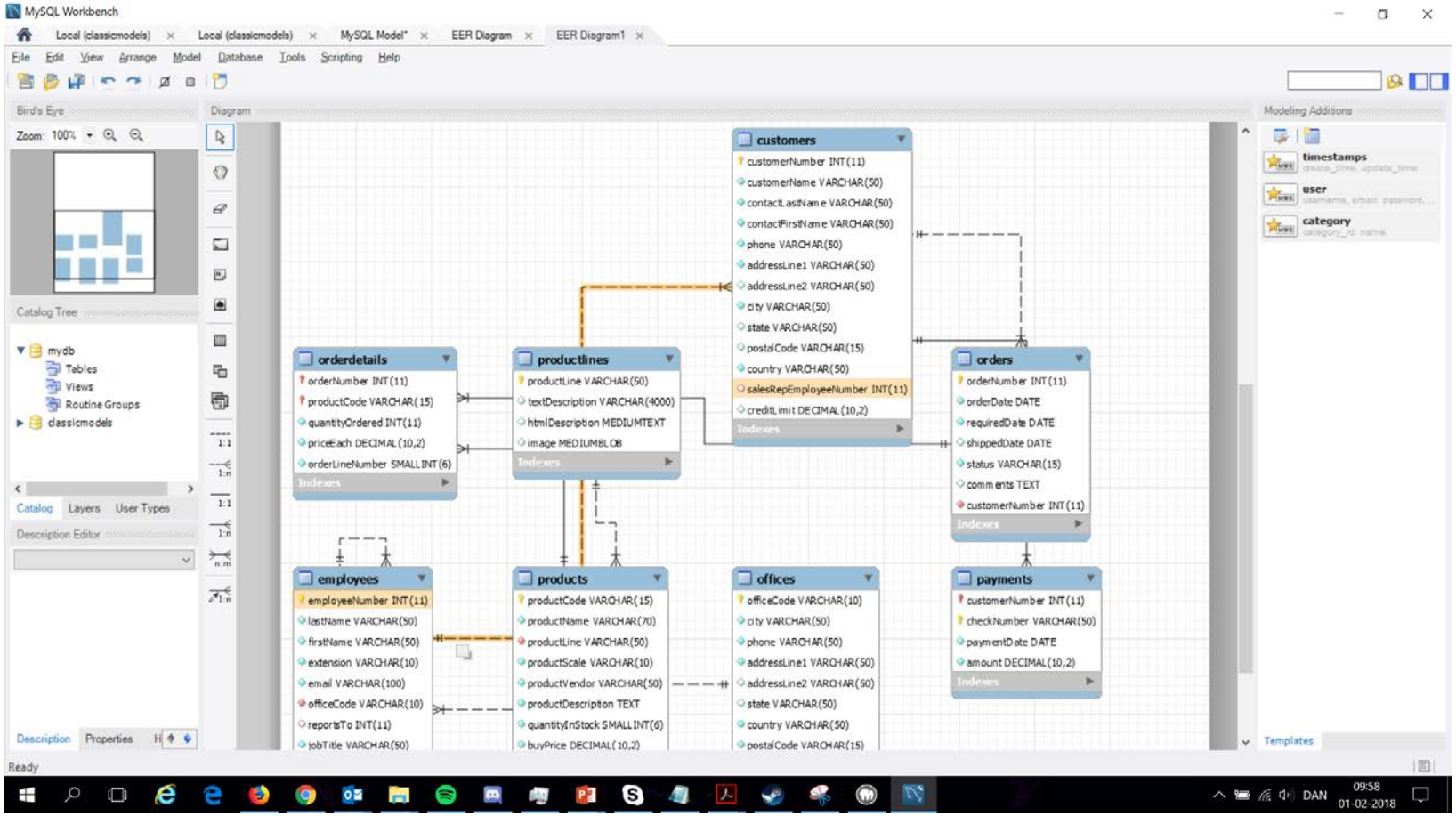
Forbind data til ny sæt - praksis

Joins i praksis

Åbn vores database og lav en ny quert (ctrl+t)



Vi vil gerne finde hvilken ansat der er tilknyttet den enkelte kunde



Vi vil sætte ansatte sammen med de relevante kunder, og kun hvor en ansat har kunder, så vi vælger et almindeligt join

```
SELECT * FROM employees
JOIN customers ON employees.employeeNumber =
customers.salesRepEmployeeNumber
```

- Læg mærke til at al data "hænger sammen" uden nogen former for null, da de kun parres hvor der er overlap

Vi vil alligevel have alle ansatte, men kun de kunder der er tilknyttet ansatte

```
SELECT * FROM employees
LEFT JOIN customers ON employees.employeeNumber
= customers.salesRepEmployeeNumber
```

- Læg mærke til at der nu er kolonner, hvor vi har felter med NULL

Nu sætter vi al info om ansatte sammen med kunder

```
SELECT * FROM employees
RIGHT JOIN customers ON employees.employeeNumber
= customers.salesRepEmployeeNumber
```

- Læg mærke til at der nu er andre kolonner, hvor vi har felter med NULL

Nu sætter vi al info om kunder sammen med ansatte

```
SELECT * FROM employees
INNER JOIN customers ON employees.employeeNumber
= customers.salesRepEmployeeNumber
```

- Læg mærke til de mange felter med NULL

Nu sætter vi al info om ansatte sammen med kunder hvor der er overlap

```
SELECT * FROM employees
INNER JOIN customers ON employees.employeeNumber
= customers.salesRepEmployeeNumber
```

- Læg mærke til de mange felter med NULL

MySQL understøtter ikke OUTER JOINS eller FULL JOINS

SQL File 3* | new_schema - Schema

Limit to 1000 rows

```
1  SELECT * FROM employees
2  OUTER JOIN customers ON employees.employeeNumber = customers.salesRepEmployeeNumber
3
```

```
1  SELECT * FROM employees
2  FULL JOIN customers ON employees.employeeNumber = customers.salesRepEmployeeNumber
```

10	10:30:29	SELECT * FROM employees FULL JOIN customers ON employees.employeeNumber = customers.salesRepEm...	Error Code: 1054. Unknown column 'employees.employeeNumber' in 'on clause'

Keys og foreign keys i praksis

Vi gennemgår dagens database

At finde den rette data

Filtrering og søgning i resultater

ORDER BY

- Indtil videre returnerer vi posterne fra vores databaser i den rækkefølge, de vises i databasen.
- I eksemplet har vi således sorteret efter employeeNumber.
- ORDER BY-klausulen kan ændre på dette.

```
SELECT firstName, lastName, email FROM employees
ORDER BY firstName ASC
```

- Prøv med DESC i stedet for ASC

ORDER BY med flere kolonner

- Nogle gange kan det være rart at sortere efter flere parametre hvis et domæne har gentagelser, således at man stadig har styr på indholdet ud fra en sekundær værdi

```
SELECT productLine, productName, quantityInStock
FROM productsORDER BY productLine DESC,
quantityInStock ASC
```

Begræns antal svar

- Nogle gange har man ikke behov for at hente *alt* data der opfylder ens krav ud
- Ofte vil det være mere praktisk at vise 10-20 svar ad gangen
- I nogle SQL varianter (f.eks. MS SQL Server / MS Access) vil koden være

```
SELECT TOP(5) productLine, productName,
quantityInStock FROM products
```

- Bemærk at man også kan angive % istedet for et tal

```
SELECT TOP(5) PERCENT productLine, productName,
quantityInStock FROM products
```

Begræns antal svar

- I MySQL virker TOP dog ikke, men så kan vi bruge LIMIT

```
SELECT productLine, productName, quantityInStock
FROM products
ORDER BY productLine DESC, quantityInStock ASC
LIMIT 5
```

- TOP og LIMIT fungerer, uanset hvilke poster der returneres fra resten af ens SELECT-kommando.
- Derefter vil ting som ORDER og filtrering ved hjælp af WHERE-klausulen først udføres. Når du har et sæt resultater, vil TOP/LIMIT-klausulen returnere XX antal poster fra det pågældende sæt.

Transaktioner

Samlinger af databasehandlinger

Hvad er en transaktion

- En transaktion er en arbejdsenhed, der udføres mod en database.
- Transaktioner er enheder eller sekvenser af arbejde udført i en logisk rækkefølge, uanset om den er manuelt skrevet af en bruger eller automatisk lavet af en slags databaseprogram.
- En transaktion er udbredelsen af en eller flere ændringer i databasen.
  - Hvis du for eksempel opretter en post eller opdaterer en post eller sletter en post fra tabellen, udfører du en transaktion på den pågældende tabel.
  - Det er vigtigt at kontrollere disse transaktioner for at sikre dataintegriteten og håndtere databasefejl.
- Praktisk set samler man mange SQL-forespørgsler i en gruppe og udføre dem alle sammen som en del af en transaktion.

Egenskaber for transaktioner - ACID

- Transactions have the following four standard properties, usually referred to by the acronym ACID.
- Atomicity – sikrer, at alle operationer inden for arbejdsenheden gennemføres med succes. Ellers afbrydes transaktionen i tilfælde af fejl, og alle tidligere operationer rulles tilbage til deres tidligere tilstand.
- Consistency – sikrer, at databasen korrekt ændrer states på en vellykket transaktion.
- Isolation – gør det muligt for transaktioner at fungere uafhængigt af og transparent for hinanden.
- Durability – sikrer, at resultatet eller effekten af en committed (forpligtet/indsendt) transaktion fortsætter i tilfælde af systemfejl.

Transaction

- Før man kan udnytte alle de førnævnte fordele skal man dog samle sin SQL i en transaktion, så den udføres samlet.

```
SET TRANSACTION [ READ WRITE | READ ONLY ];
```

- I nogle varianter som MySQL er det

```
START TRANSACTION [ READ WRITE | READ ONLY ];
```

- Man begynder med ovenstående og afslutter med COMMIT

Transaktions kontrol og kommandoer

- De følgende kommandoer bruges til at kontrollere transaktioner:
- COMMIT – til at gemme ændringerne.
- ROLLBACK – til at rule ændringerne tilbage.
- SAVEPOINT – skaber et sted i gruppen af transaktioner hvortil man kan forestage et ROLLBACK.
- SET TRANSACTION – giver et navn til en transaktion.

- Transaktionskontrolkommandoer bruges kun med DML-kommandoerne som f.eks. - INSERT, UPDATE og DELETE.
- De kan ikke bruges, når du opretter tabeller eller dropper dem, fordi disse operationer automatisk committes til databasen.

COMMIT kommandoen

- COMMIT-kommandoen er transaktionskommandoen, der bruges til at gemme ændringer, der påberåbes af en transaktion til databasen.
- COMMIT kommandoen gemmer alle transaktioner til databasen siden den sidste COMMIT eller ROLLBACK kommando.
- Syntaksen for COMMIT-kommandoen er blot:

```
COMMIT;
```

ROLLBACK kommandoen

- ROLLBACK-kommandoen er transaktionskommandoen, der bruges til at fortryde transaktioner, der ikke allerede er gemt i databasen.
- Denne kommando kan kun bruges til at fortryde transaktioner, siden den sidste COMMIT- eller ROLLBACK-kommando blev udstedt.
- Syntaksen for ROLLBACK-kommandoen er blot:

```
ROLLBACK;
```

SAVEPOINT

- Er SAVEPOINT er et punkt i en transaktion, hvor man kan rulle transaktionen tilbage til et bestemt punkt uden at skulle rulle hele transaktionen tilbage.
- Syntaksen for en SAVEPOINT-kommando er som vist nedenfor:

```
SAVEPOINT savepointname;
```

- For at rulle tilbage til et SAVEPOINT er syntaksen således:

```
ROLLBACK TO savepointname;
```

RELEASE SAVEPOINT

- Hvis man ikke vil bruge et savepoint mere kan man slippe det fri med følgende kommando

```
RELEASE SAVEPOINT savepointname;
```

- Herefter er det væk og man kan ikke rulle tilbage til det mere.

Samlet eksempel

```
START TRANSACTION;
BEGIN;
SAVEPOINT sp1;
ALTER TABLE customersDROP FOREIGN KEY customers_ibfk_1;
ROLLBACK TO SAVEPOINT sp1;
DELETE FROM employees WHERE officeCode = 6 AND
employeeNumber > 1600;
COMMIT;
```

Databaser i web

Sådan forbinder du og kommunikerer med en database i PHP

PHP generelt

- Den er tre måder man typisk forbinder til en database på med PHP, MySQL, MySQLi og PDO.
- Den ældste, MySQL, regnes som dog som forældet og understøttes ikke af de nyeste udgaver af PHP.
- MySQLi er dog "bare" en videreudvikling af MySQL, i'et står for Improved, så det er let at "oversætte" alle de gamle projekter man støder på hos kunder til den mere moderne standard.
- MySQLi kan kodes både procedural og objekt orienteret, hvor PDO udelukkende er objekt orienteret.
- Her er nogle eksempler på hvordan man forbinder til en database

PHP og MySQLi procedural

```
<?php
$servername = “localhost";
$username = "username";
$password = “password";
// Skab forbindelse
$conn = mysqli_connect($servername, $username, $password);
// Tjek forbindelse
if (!$conn) {
      die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully”;
```

- Man benytter herefter $conn variablen når man vil forbinde til database.

PHP og MySQLi objekt orienteret

```
<?php
$servername = “localhost";
$username = “username";
$password = “password”;
// Skab forbindelse
$conn = new mysqli($servername, $username, $password);
// Tjek forbindelse
if ($conn->connect_error) {
      die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully”;
```

- Her behandler vi $conn som et objekt, som vi så tjekker for forbindelses fejl.

PHP og PDO

```
<?php
$servername = “localhost";
$username = “username";
$password = “password”;
try {
      $conn = new PDO(“mysql:host=$servername;dbname=myDB", $username, $password);
      $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
      echo "Connected successfully";
}
catch(PDOException $e)
{
      echo "Connection failed: " . $e->getMessage();
}
```

- Her opretter man først et $conn objekt, som vi så giver attributten PDO::ERRMODE_EXCEPTION, som vi så fanger med en catch og returnerer fejl-beskeden i $e.

Hent data via SQL via MySQLi

```
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}
```

Hent data via SQL via PDO

```
$stmt = $conn->prepare("SELECT id, firstname, lastname FROM MyGuests");
$stmt->execute();

// set the resulting array to associative
$result = $stmt->setFetchMode(PDO::FETCH_ASSOC);
foreach(new TableRows(new RecursiveArrayIterator($stmt-fetchAll())) as $k=>$v)
{
        echo $v;
}
```

Luk forbindelsen

- Man bør huske at lukke forbindelsen når man er færdig med den for at frigive resurser og for sikkerhedens skyld. Den lukkes selvfølgelig ved scriptets afslutning, men man kan sagtens være færdig med at benytte den inden og bør derfor lukke den.
- **MySQLi procedural**

```
mysqli_close($conn);
```

- **MySQLi objekt orienteret**

```
$conn->close();
```

- **PDO**

```
$conn = null;
```

Lektier til næste gang

Læsning og opgaver

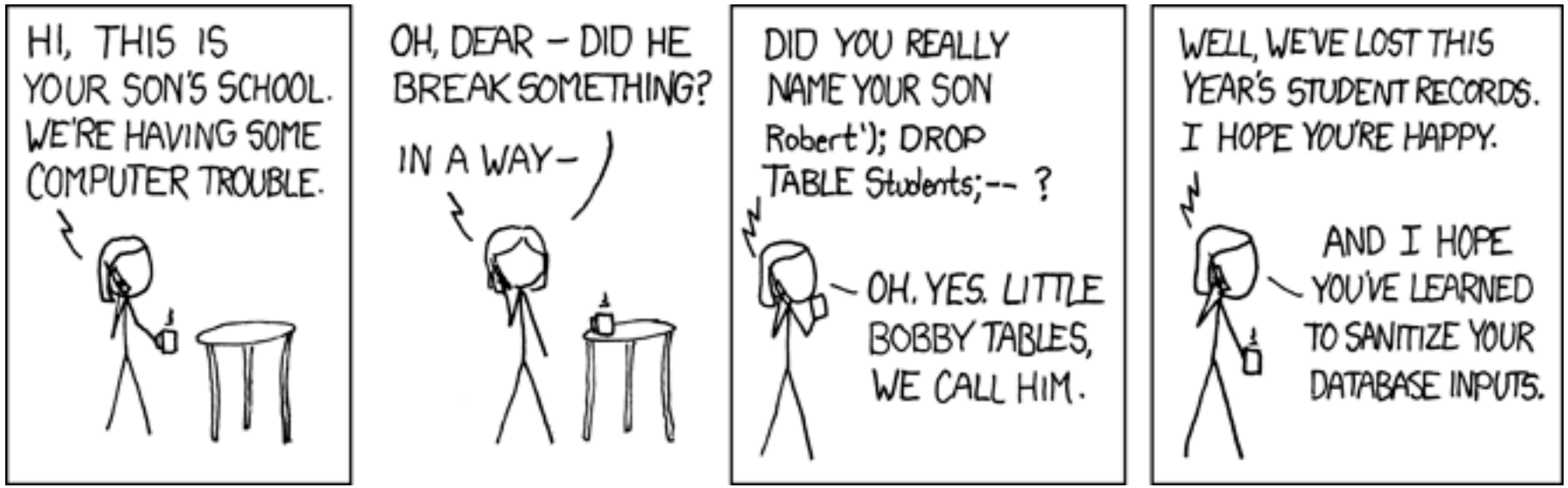
Læsning og opgaver til næste gang

- Læs side 93 – 145 i SQL - The Ultimate Beginners Guide
- Lav en kort skriftlig opsummering af de fire grupper af SQL kommandoer (se anden undervisnings gang)
  - Jeg hører jer kort I dem næste gang, og de skal bruges til eksamen
- I gør ER diagrammet over jeres selvvalgte database færdigt
- I laver SQL statements til at bygge og søge i jeres selvvalgte database

Ekstra

- Forklar hvorfor eksemplet med ROLLBACK ikke virker 100%

Forsmag på næste gang



Kilder

Materiale benyttet i denne undervisningsgang

Kilder

Dagens database

- http://www.mysqltutorial.org/mysql-sample-database.aspx
- https://youtu.be/AiTXe3cyuJM
- Matematikken bag databaser
- https://en.wikipedia.org/wiki/Boolean_algebra
- https://www.tutorialspoint.com/dbms/relational_algebra.htm
- https://becominghuman.ai/why-i-learned-boolean-algebra-and-invented-a-better-multi-pass-sql-engine-1a69a96904d9

Kilder

Forbind data til ny sæt - praksis

- https://community.modeanalytics.com/sql/tutorial/sql-joins-where-vs-on/
- https://www.tutorialspoint.com/sql/sql-full-joins.htm

At finde den rette data

- https://www.dnndev.com/learn/guide-legacy/sql-for-beginners/sorting-data
- https://www.dnndev.com/learn/guide-legacy/sql-for-beginners/limiting-results
- https://www.w3schools.com/sql/sql_top.asp

Kilder

At finde den rette data

- https://www.techonthenet.com/sql_server/select_top.php
- https://www.dnndev.com/learn/guide-legacy/sql-for-beginners/filtering-data

Transaktioner

- http://www.tutorialspoint.com/sql/sql-transactions.htm
- https://dev.mysql.com/doc/refman/5.7/en/commit.html
- https://dev.mysql.com/doc/refman/5.7/en/savepoint.html

Kilder

Eksempler på web sprog

- https://www.w3schools.com/php/php_mysql_select.asp