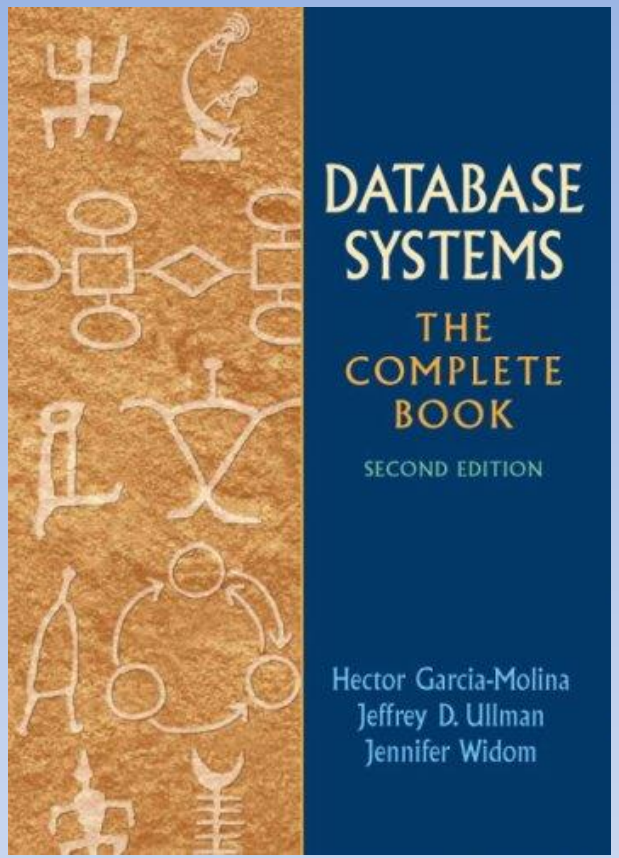


Database Systems
Chapter 6 & Section 2.3: SQL Intro

- SQL Intro
- Creating Tables
- Altering Table
- Dropping Tables
- Keys
- MySQL Examples

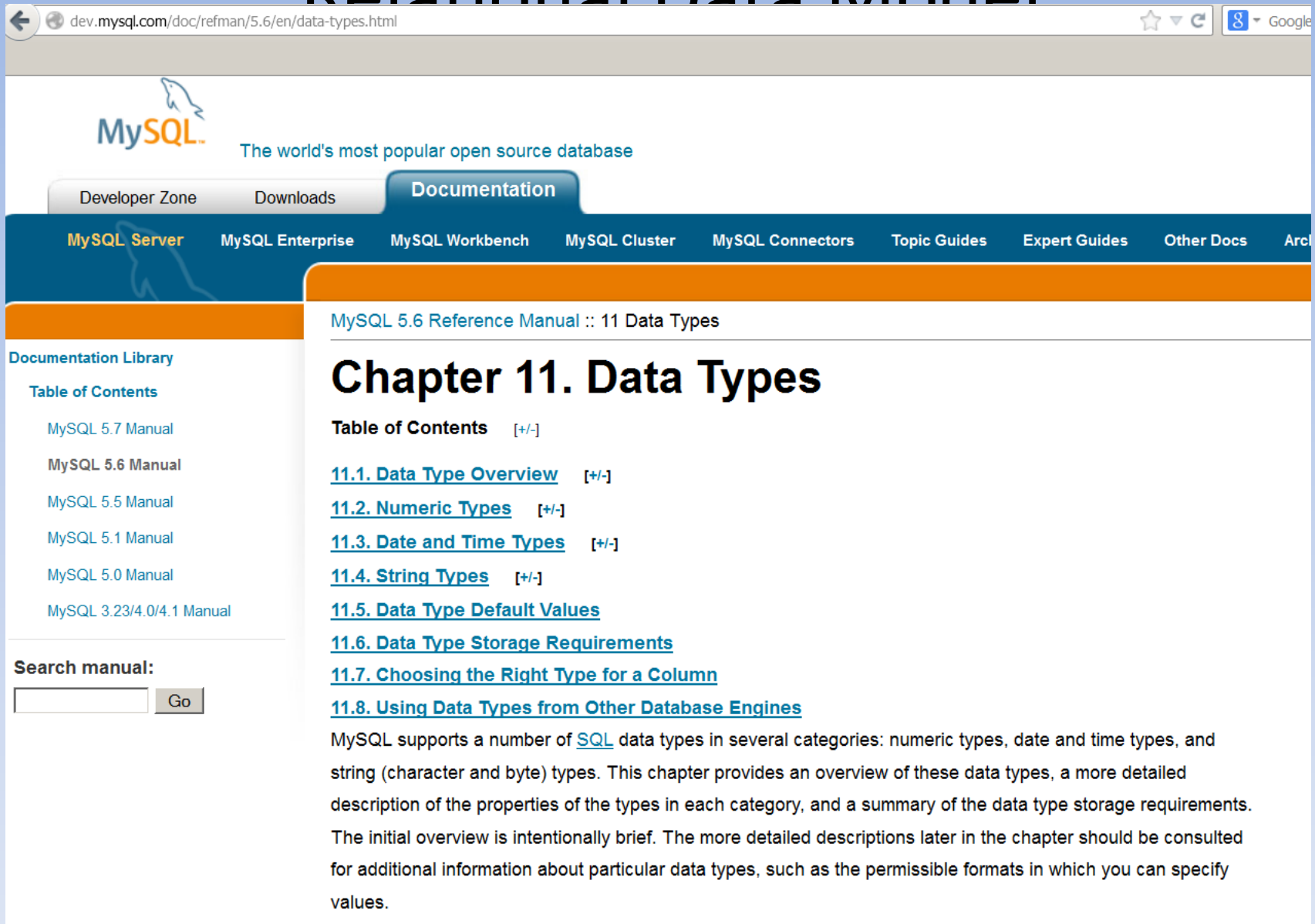
Relational Data Model: Part 2
Language: SQL

- Two Aspects to language
- Data Definition (ddl):
  - Declaring database schema: tables, constraints, indexes, views.
  - like declaring data/variables in programming language
- Data Manipulation (dml):
  - asking questions (querying) and modifying data.
  - Like executable code within programming language.

Relational Data Model
SQL (DDL aspects)

- Create table NAME(att1 type, att2 type …)
  - NAME: name of the newly created table
  - att1, att2, att3, att4, …: Attributes for table.
  - Type: data type for each of the attributes
    - See next slide!

Relational Data Model



Relational Data Model: SQL Creating Tables

```
CREATE TABLE Person(
   pid int,
   lName varchar(20),
   fName varchar(20),
   gender char(1),
   birth date);
```

Tables w/ Keys

- Method 1:
  - 'PRIMARY KEY' included after attribute declaration
  - Good only when key is a single attributed
- Method 2:
  - 'PRIMARY KEY (att1, att2, …)' included after all attribute declarations
- UNIQUE can be used instead of PRIMARY KEY:
  - UNIQUE keys can have NULL value.
  - PRIMARY KEY keys cannot be NULL.

Relational Data Model: SQL
Creating Tables w/ Keys
Method 1

```
CREATE TABLE Person(
   pid int primary key,
   lName varchar(20),
   fName varchar(20),
   gender char(1),
   birth date);
```

Relational Data Model: SQL
Creating Tables w/ Keys
Method 2

```
CREATE TABLE Person(
   pid int,
   lName varchar(20),
   fName varchar(20),
   gender char(1),
   birth date,
   primary key (pid));
```

StudentDB Example

Transcript(sid, semester, year,

CourseID, CourseDesc, units, grade)

sid	semester	year	courseID	courseDesc	units	grade
500	Fall	1980	English 1	Composition	3	B
500	Fall	1980	Chem 1A	Gen Qual Anal	5	C
500	Fall	1980	Math 20	Intro Comp Prog	2	A
500	Fall	1980	Math 75	Math Analysis I	4	A
500	Fall	1980	Hist 11	Amer Hst to 1865	3	A
500	Spring	1981	QM 64	Compu Lang - COBOL	3	A
500	Spring	1981	Phil 1	Into to Phil	4	B
500	Spring	1981	Chem 8	Elem Org Chem	3	C
500	Spring	1981	Math 76	Math Analysis II	4	B
500	Spring	1981	Math 114	Discrete Struct	3	B
500	Fall	1981	Art H 20	Modern World	3	B
500	Fall	1981	Fin 34	Personal Investing	3	A
500	Fall	1981	Math 77	Math Anal III	0	F
500	Fall	1981	Math 107	Intro Prob + Stat	3	A
500	Winter	1981	Econ 1a	Prin of Econ	3	A

Delete a Table

- To delete a table use DROP:

  DROP TABLE *tablename*;

- To just delete the data (see chapter 6):

  DELETE FROM *tablename*;

Delete a Table: Example StudentDB.sql

- Create Table:

```
CREATE TABLE Person(
    pid int primary key,
    lName varchar(20), fName varchar(20),
    gender char(1), birth date);
```

- Drop table:

```
DROP TABLE Person
```

Delete a Table: Example StudentDB.sql

```
CREATE TABLE Transcript(
   sid int, semester varchar(20), year int,
   CourseID varchar(20), CourseDesc varchar(20),
   units int, grade char(2))
```

- To delete a table use DROP:
  DROP transcript;

- To just delete the data (see chapter 6):
  DELETE FROM transcript;

Modifying Tables: ALTER

- The command ALTER is used to modify tables:
  - ALTER *tablename* ADD *att1 type1*
    - Adds the attribute *att1* with *type type1* to table *tablename*.
  - ALTER tablename DROP attribute

Computers

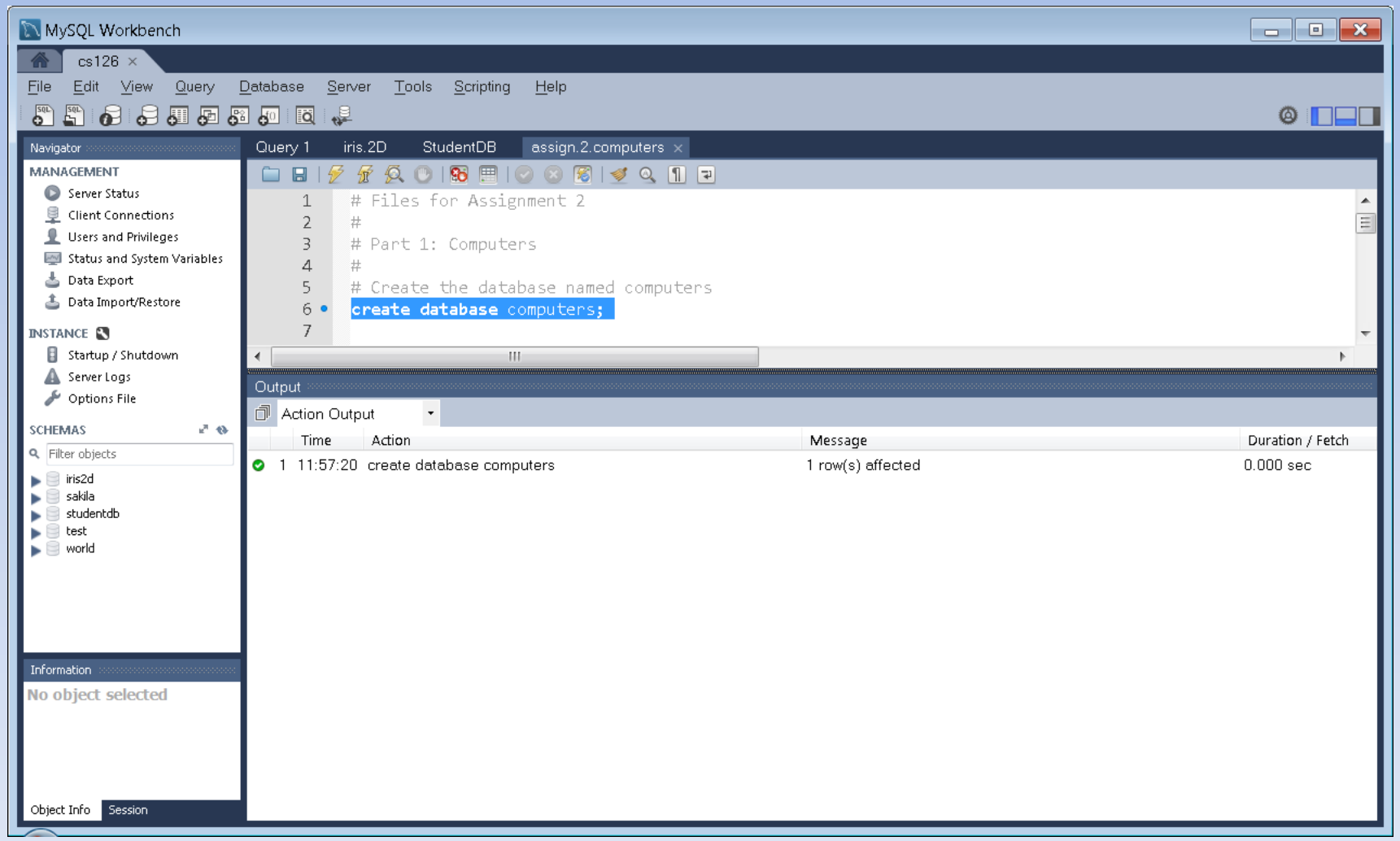
- ‘computers’ domain described in the text book
- Data shown in Figures 2.20 & 2.21

Computers
Creating a new Database Schema

- Create an empty database schema called ‘computers’

```
# #Create the database named computers
create database computers;
```

MySQL – Screenshot



Computers
Using new Database Schema

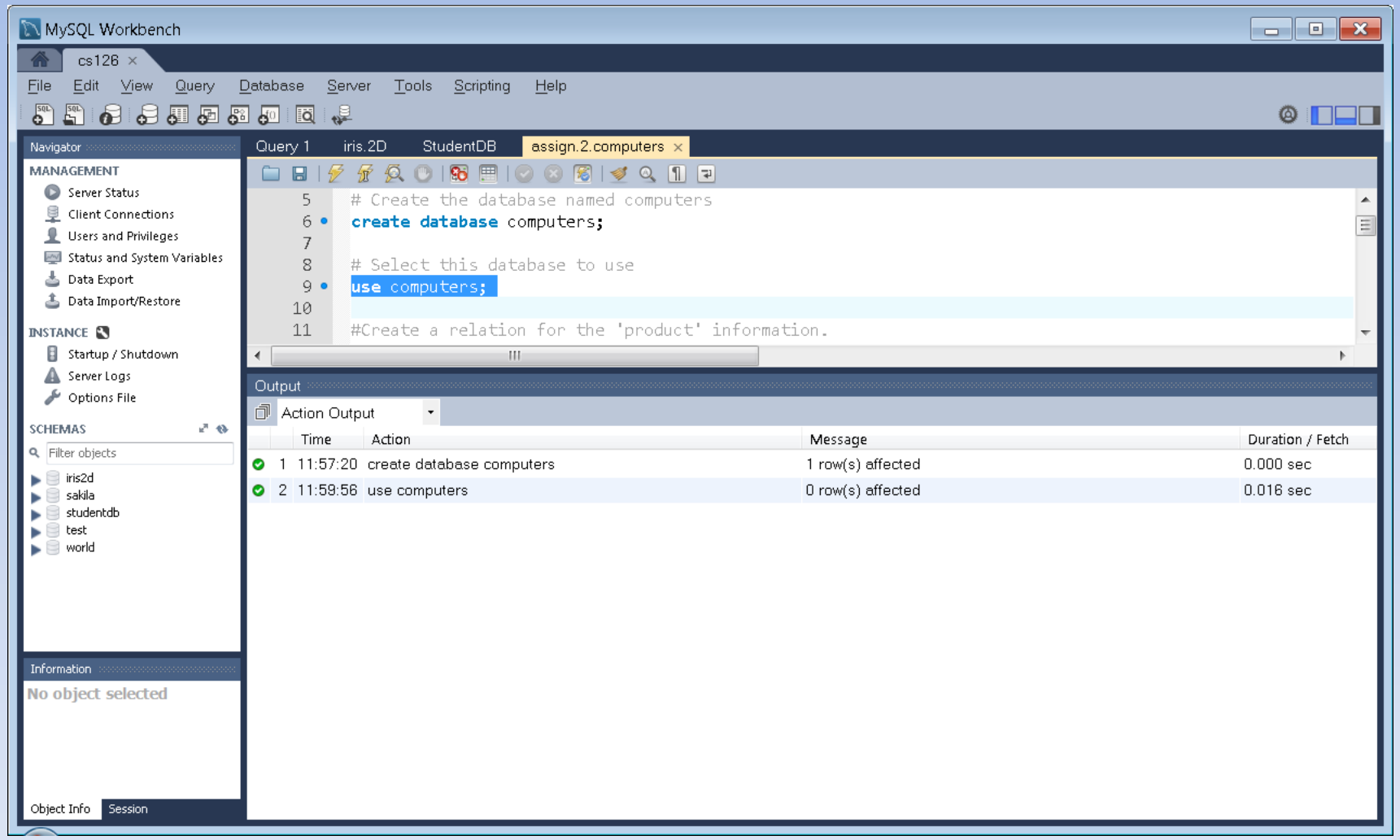
- Select the schema ‘computers’ for use.
- Executing SQL In Workbench:
  - Select Command
  - Enter: Ctrl-Shift-Enter (Simultaneously)

#Create the database named computers

```
create database computers;
```

<ctrl><shift><cr>

MySQL Screenshot

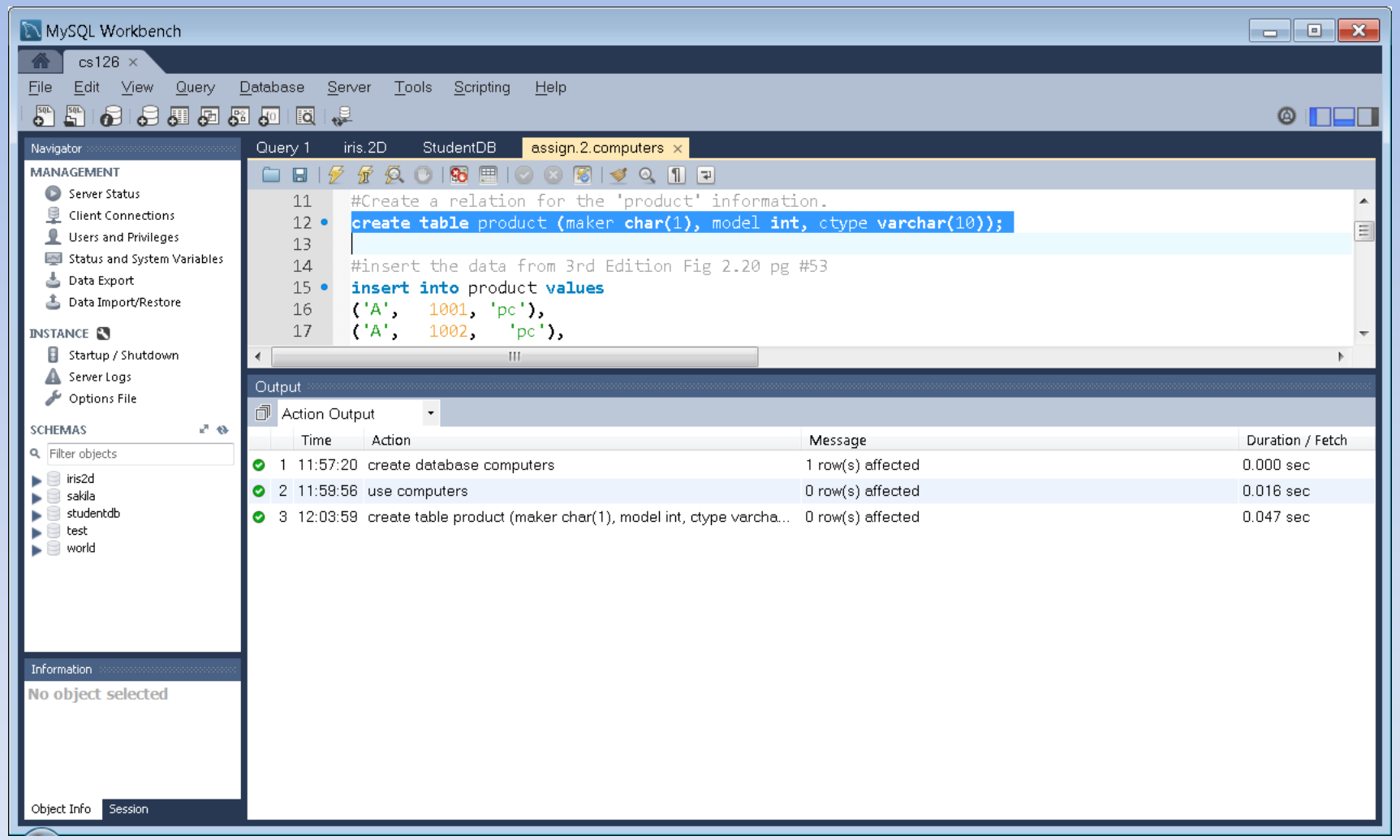


Computers
Creating relations in Computers Schema

- Select the schema ‘computers’ for use.

```
#Create a relation for the 'product' information.
create table product (maker char(1), model int,
   ctype varchar(10));
```

Screenshot

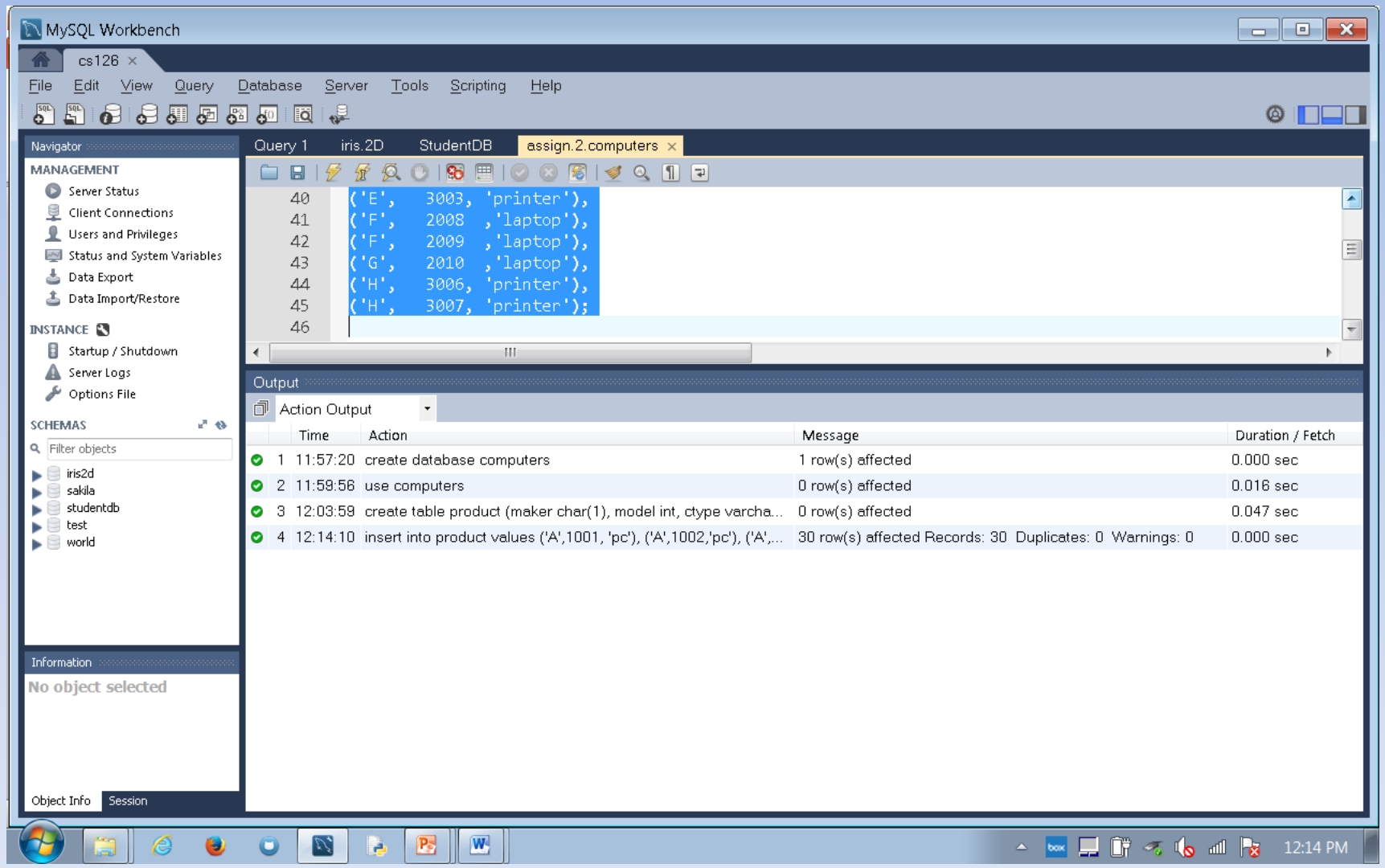

MySQL Workbench
cs126
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
SCHEMAS
Filter objects
iris2d
sakila
studentdb
test
world
Information
No object selected
Object Info
Session
Query 1
iris.2D
StudentDB
assign.2.computers
11 #Create a relation for the 'product' information.
12 create table product (maker char(1), model int, ctype varchar(10));
13
14 #insert the data from 3rd Edition Fig 2.20 pg #53
15 insert into product values
16 ('A', 1001, 'pc'),
17 ('A', 1002, 'pc'),
Output
Action Output
Time Action Message Duration / Fetch
1 11:57:20 create database computers 1 row(s) affected 0.000 sec
2 11:59:56 use computers 0 row(s) affected 0.016 sec
3 12:03:59 create table product (maker char(1), model int, ctype varcha... 0 row(s) affected 0.047 sec

Computers
Inserting data into relations

- Relations are empty, and need data

#insert the data from 3rd Edition Fig 2.20 pg #53
insert into product values
('A', 1001, 'pc'),
('A', 1002, 'pc'),
('A', 1003, 'pc'),
('A', 2004 , ‘laptop'),
('A', 2005 , 'laptop'),
('A', 2006 , 'laptop'),
('B', 1004, 'pc'),
('B', 1005, 'pc'),
('B', 1006, 'pc'),
('C', 1007, 'pc'));

MySQL Screenshot

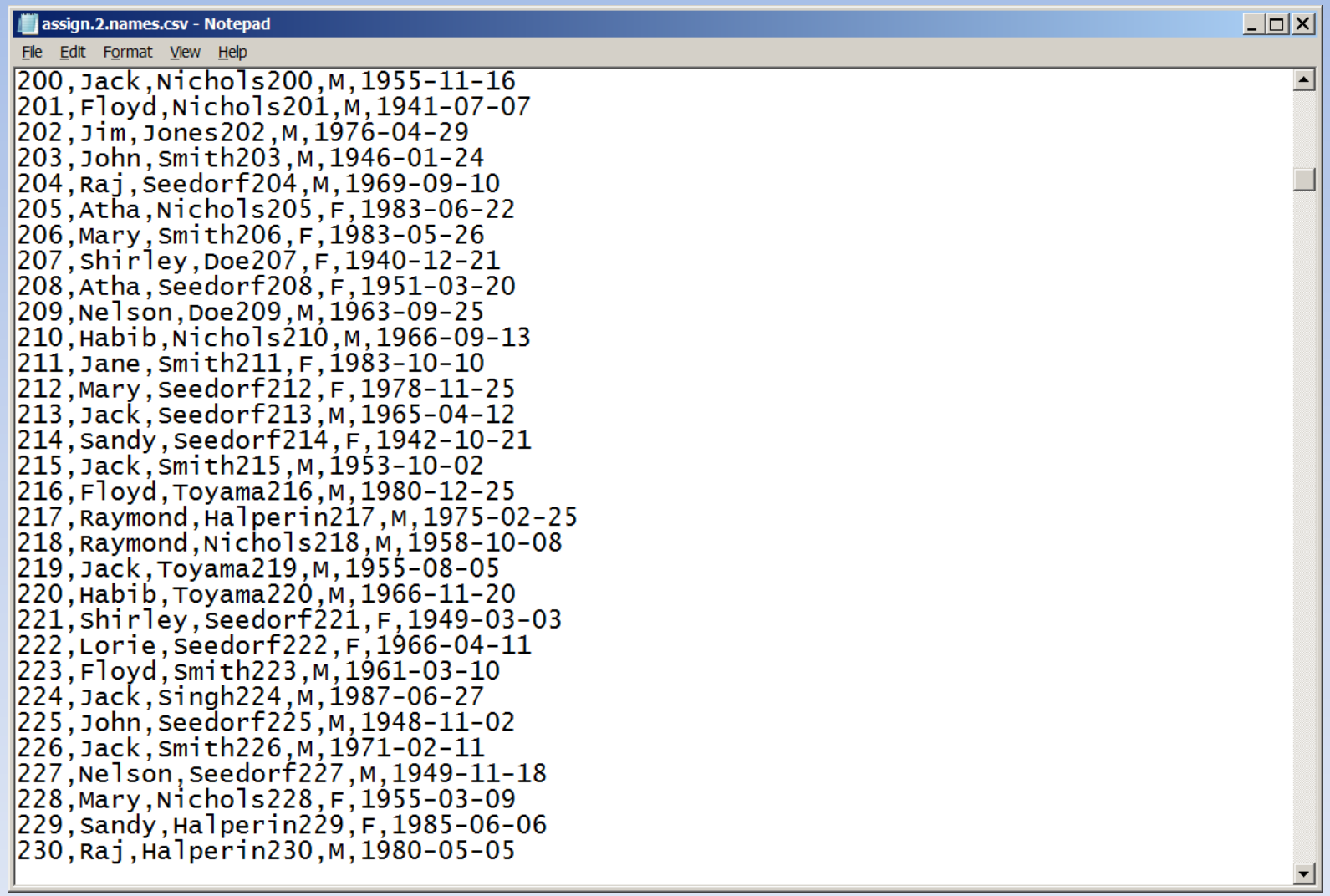


Inserting data from CSV

- Importing data from CSV file:

```
/* Load Data from CSV */
LOAD DATA
LOCAL INFILE
'D:/Documents/GitHub/Spring14CSci126/Assignments/CSci126.Assignment.2/assign.2.names.csv'
INTO TABLE person
FIELDS TERMINATED BY ',';
```

Inserting data from CSV



Generating Test Data w/ Python

```
import csv
import random

def RandomValue(items):
    return items[random.randrange(len(items))]
```

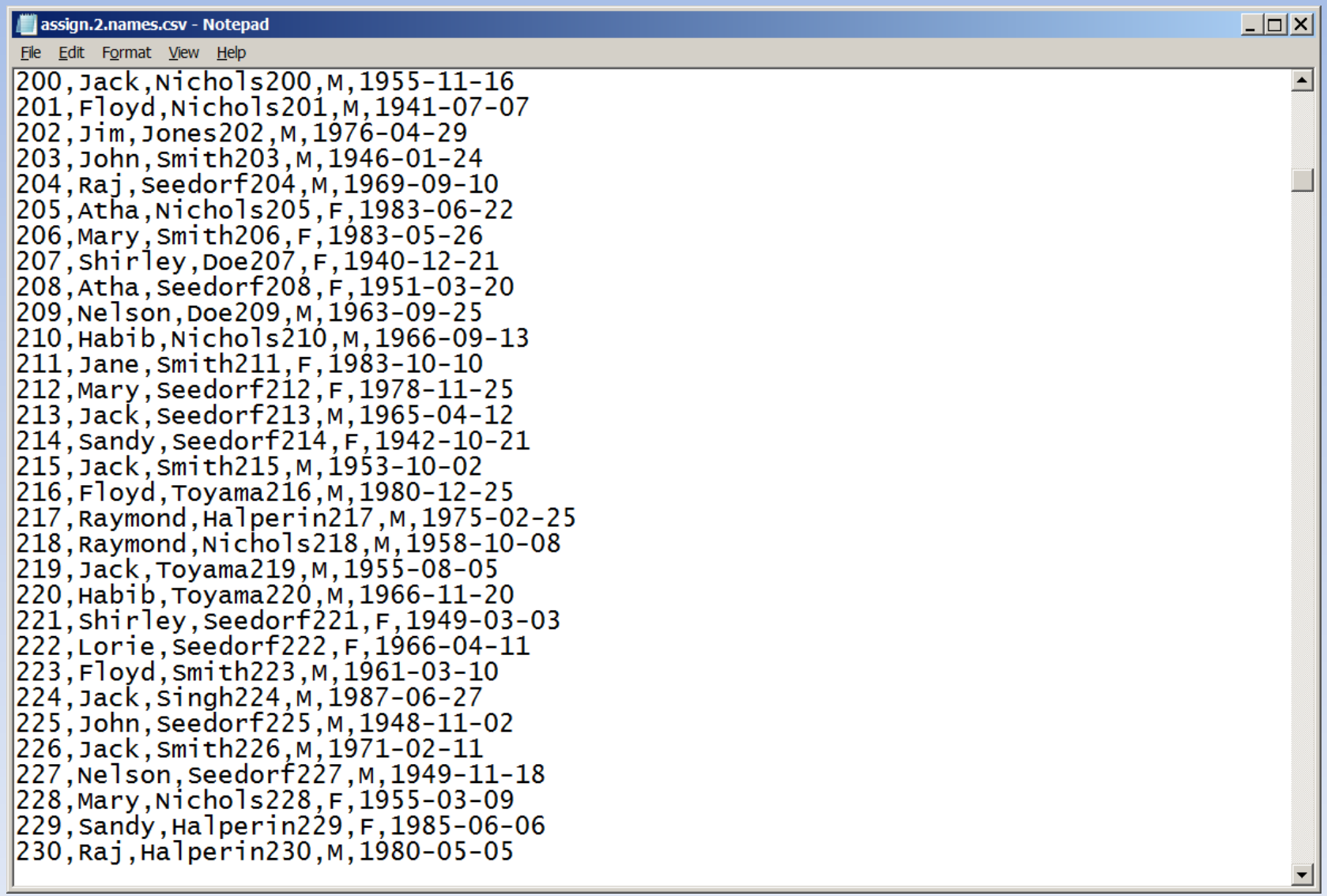
Generating Test Data w/ Python

```
def gen():
    random.seed(555)
    size = 1000
    idoffset = 100
    lastNames = ['Doe', 'Smith', 'Jones', \
                 'Seedorf', 'Nichols', 'Toyama', 'Singh', 'Halperin']
    firstNames = [['John', 'Jack', 'Jim', 'Raj', \
                   'Marius', 'Nelson', 'Habib', \
                   'Floyd', 'Pete', 'Raymond', 'Buryl'], \
                  ['Jane', 'Jocelyn', 'Jackie', 'Shirley', \
                   'Atha', 'Lorie', 'Sandy', \
                   'Ginger', 'Mary']]
    genders = ['M', 'F']
    months = ['01','02','03','04','05','06','07','08','09','10','11','12']
    days = ['01','02','03','04','05','06','07','08','09','10', \
      '11','12','13','14','15','16','17','18','19','20', \
      '21','22','23','24','25','26','27','28','29','30']
    syear = 1940
    yrange = 50
    #filename = '\\tmp\\names.csv'
    filename = 'names.csv'
    with open(filename, 'wb+') as csvfile:
        f = csv.writer(csvfile, delimiter=',',
                       quotechar='"', quoting=csv.QUOTE_MINIMAL)
```

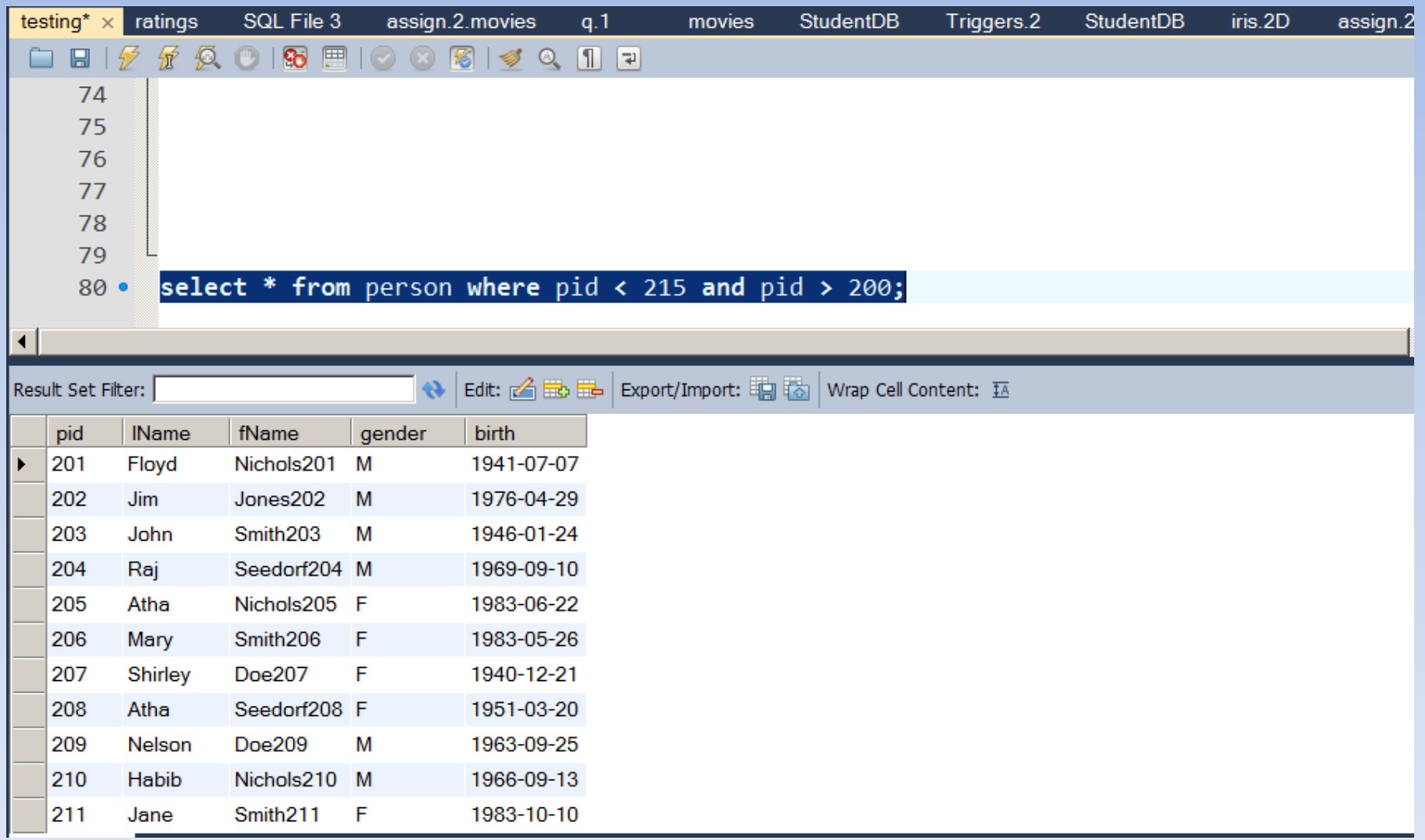
Generating Test Data w/ Python

```
with open(filename, 'wb+') as csvfile:
    f = csv.writer(csvfile, delimiter=',',
                   quotechar='"', quoting=csv.QUOTE_MINIMAL)
    for i in range(size):
        cid = idoffset+i
        g = random.randrange(len(genders))
        firstName = RandomValue(firstNames[g])
        lastName = RandomValue(lastNames) + str(cid)
        gender = genders[g]
        month = RandomValue(months)
        day = RandomValue(days)
        year = random.randrange(yrange) + syear
        bd = str(year) +'-'+ month + '-' + day
        f.writerow([cid, firstName, lastName, gender, bd ])
```

Generating Test Data w/ Python



Generating Test Data w/ Python



pid	lName	fName	gender	birth
201	Floyd	Nichols201	M	1941-07-07
202	Jim	Jones202	M	1976-04-29
203	John	Smith203	M	1946-01-24
204	Raj	Seedorf204	M	1969-09-10
205	Atha	Nichols205	F	1983-06-22
206	Mary	Smith206	F	1983-05-26
207	Shirley	Doe207	F	1940-12-21
208	Atha	Seedorf208	F	1951-03-20
209	Nelson	Doe209	M	1963-09-25
210	Habib	Nichols210	M	1966-09-13
211	Jane	Smith211	F	1983-10-10

Chapter 6: SQL

- Discussed Data Definition Language (DDL) within SQL.
- Now looking at Data Manipulation Language (DML) within SQL.
- Implementation of Relational Algebra
- SQL Allows DBMS to optimize actual implementation.

But First: Getting Data
A Brief Intro

- SQL Modify Commands are not like Queries , in that:
  - they do not return a result.
  - they do modify the contents of database.
- SQL Modify Commands:
  - Insert
  - Update
  - Delete

Database Mods: Basic Insert

- INSERT INTO table VALUE (…);
- INSERT INTO table VALUES (…), (…), (…), …
- LATER:
  - INSERT INTO table (SUBQUERY)

Insert Example

- Inserting a single value:

Relation Schema: product (maker, model, ctype);

```
INSERT INTO product VALUE
('Z', 5005, 'laptop' );
```

Insert Example

- Inserting multiple values:

Relation Schema: product (maker, model, ctype);

#insert the data from 3rd Edition Fig 2.20 pg #53
insert into product values
('A', 1001, 'pc'),
('A', 1002, 'pc'),
('A', 1003, 'pc'),
('A', 2004 , 'laptop'),
('A‘ ,2005 , 'laptop'),
('A‘ ,2006 , 'laptop'),
('B', 1004, 'pc'),
('B', 1005, 'pc'),
('B', 1006, 'pc'),
('B', 2007 ,'laptop'),
('C', 1007, 'pc');

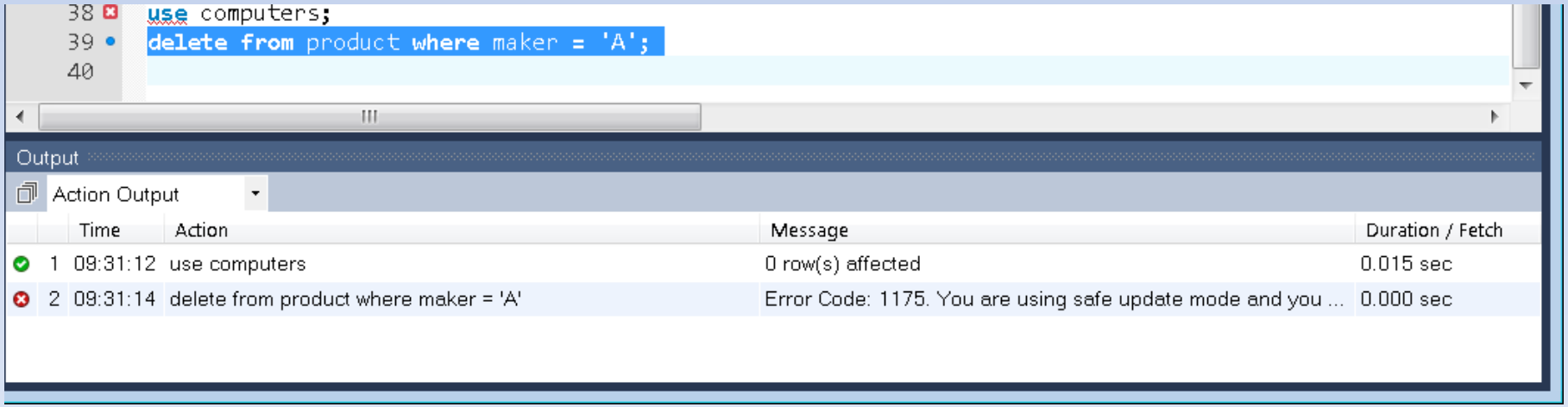
Database Mods: Delete

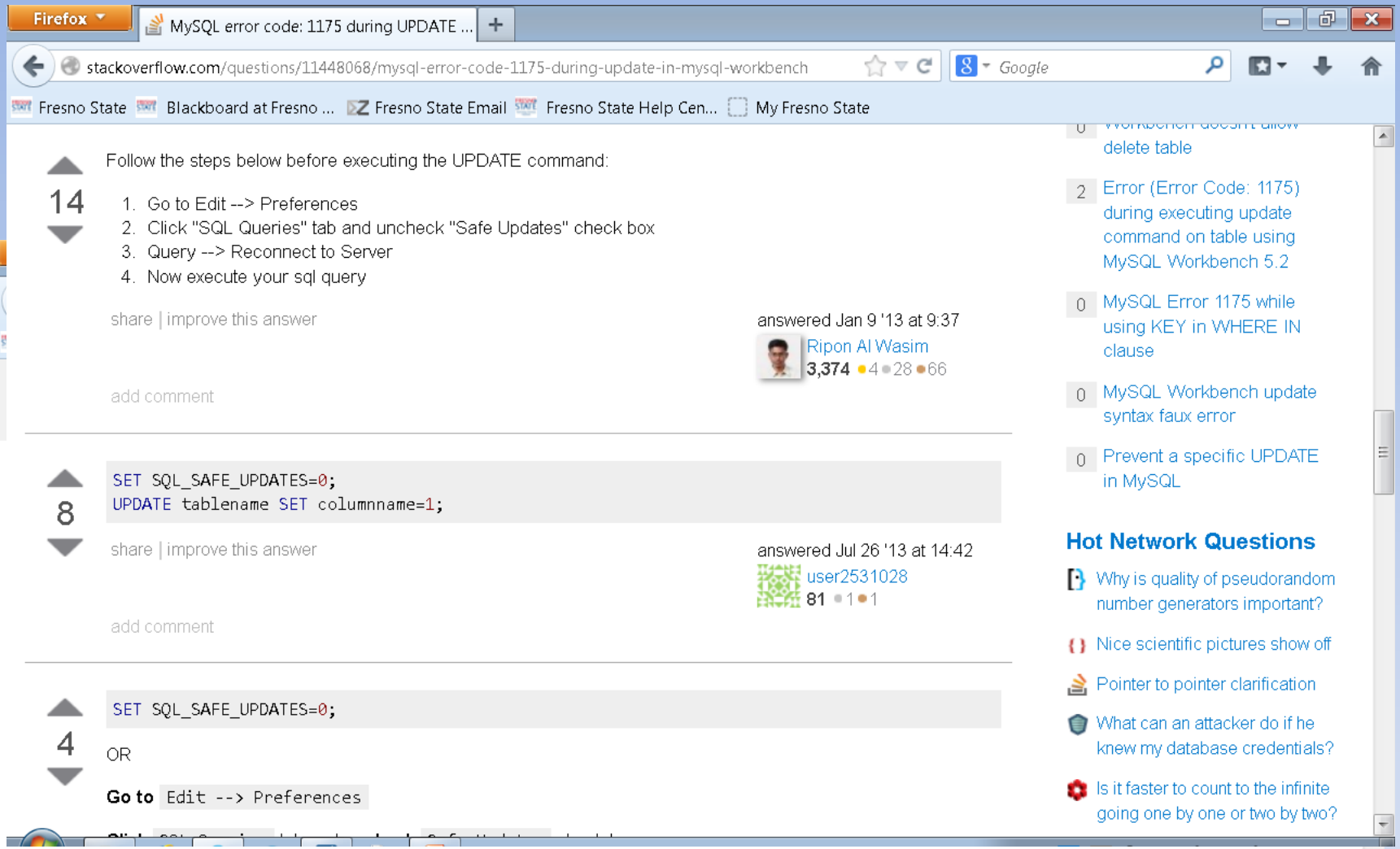
- DELETE FROM table;
- DELETE FROM table WHERE <cond>;

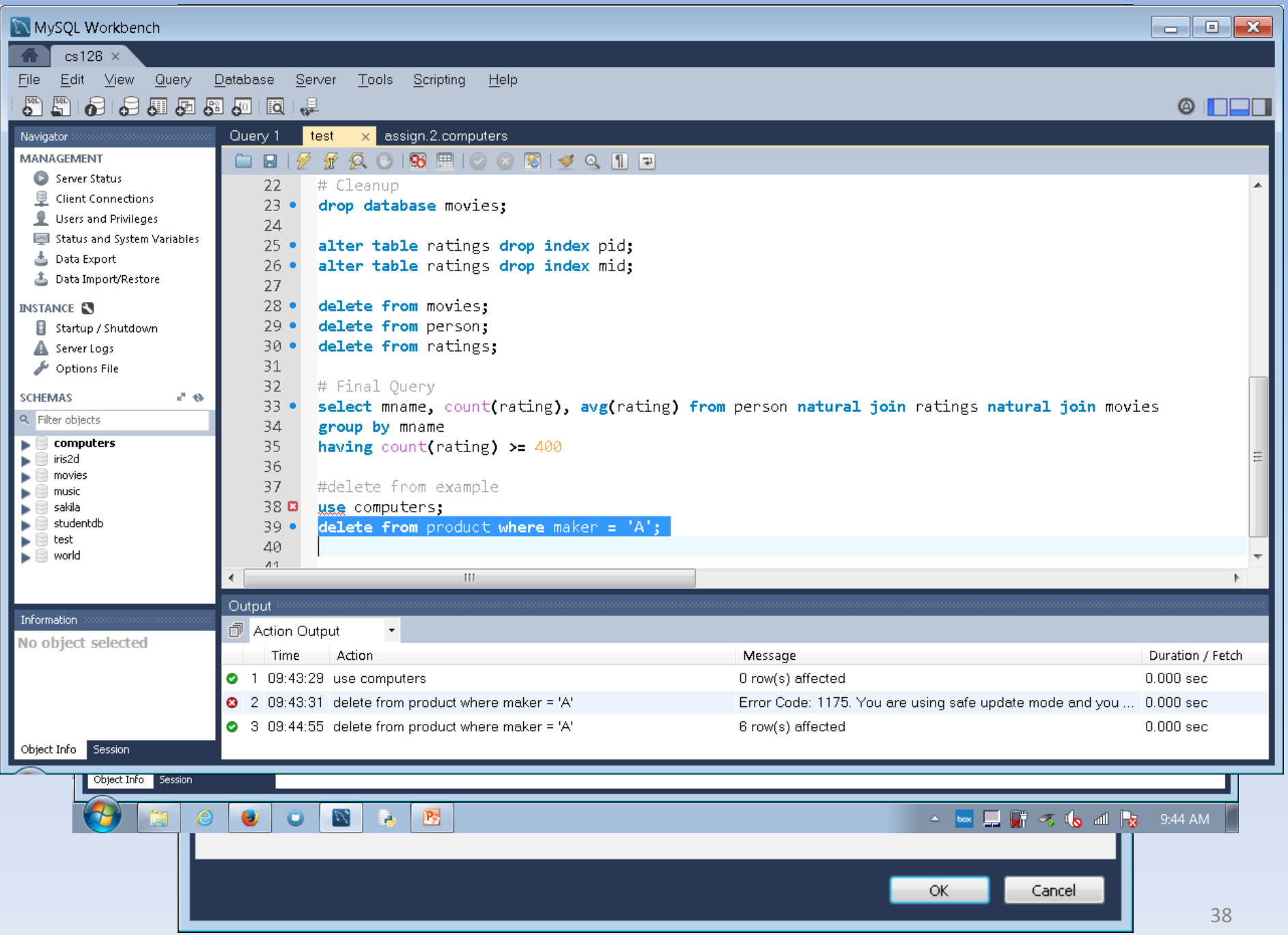
Database Mods: Delete

Relation Schema: product (maker, model, ctype);

Use computers;




Firefox
MySQL error code: 1175 during UPDATE ...
stackoverflow.com/questions/11448068/mysql-error-code-1175-during-update-in-mysql-workbench
Google
Fresno State
Blackboard at Fresno ...
Fresno State Email
Fresno State Help Cen...
My Fresno State
14
Follow the steps below before executing the UPDATE command:
1. Go to Edit --> Preferences
2. Click "SQL Queries" tab and uncheck "Safe Updates" check box
3. Query --> Reconnect to Server
4. Now execute your sql query
share | improve this answer
answered Jan 9 '13 at 9:37
Ripon Al Wasim
3,374 ●4 ●28 ●66
add comment
8
SET SQL_SAFE_UPDATES=0;
UPDATE tablename SET columnname=1;
share | improve this answer
answered Jul 26 '13 at 14:42
user2531028
81 ●1 ●1
add comment
4
SET SQL_SAFE_UPDATES=0;
OR
Go to Edit --> Preferences
delete table
2 Error (Error Code: 1175) during executing update command on table using MySQL Workbench 5.2
0 MySQL Error 1175 while using KEY in WHERE IN clause
0 MySQL Workbench update syntax faux error
0 Prevent a specific UPDATE in MySQL
Hot Network Questions
Why is quality of pseudorandom number generators important?
Nice scientific pictures show off
Pointer to pointer clarification
What can an attacker do if he knew my database credentials?
Is it faster to count to the infinite going one by one or two by two?


MySQL Workbench
cs128
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
SCHEMAS
Filter objects
computers
iris2d
movies
music
sakila
studentdb
test
world
Information
No object selected
Object Info
Session
Query 1
test
assign.2.computers
22 # Cleanup
23 drop database movies;
24
25 alter table ratings drop index pid;
26 alter table ratings drop index mid;
27
28 delete from movies;
29 delete from person;
30 delete from ratings;
31
32 # Final Query
33 select mname, count(rating), avg(rating) from person natural join ratings natural join movies
34 group by mname
35 having count(rating) >= 400
36
37 #delete from example
38 use computers;
39 delete from product where maker = 'A';
40
Output
Action Output
Time Action Message Duration / Fetch
1 09:43:29 use computers 0 row(s) affected 0.000 sec
2 09:43:31 delete from product where maker = 'A' Error Code: 1175. You are using safe update mode and you ... 0.000 sec
3 09:44:55 delete from product where maker = 'A' 6 row(s) affected 0.000 sec
Object Info
Session
9:44 AM
OK
Cancel

Database Mods: Update

- UPDATE table SET

  att1=value,

  att2=value

  WHERE <cond>;

Database Mods: Update

- R1(K int, B float, C float, primary key (K));

select * from r1;

K	B	C
2	1	7
3	2	0
5	1	7
7	2	2
9	1	8

Database Mods: Update

- R1(K int, B float, C float, primary key (K));

UPDATE r1 SET b = b*10, c = c/10.0 WHERE k < 5;

K	B	C
2	10	0.7
3	20	0
5	1	7
7	2	2
9	1	8

Back to Chapter 6: SQL

- Discussed Data Definition Language (DDL) within SQL.
- Now looking at Data Manipulation Language (DML) within SQL.
- Implementation of Relational Algebra
- SQL Allows DBMS to optimize actual implementation.
- One SQL Question at End;

Example Tables

- R1(K, A, B, C)
- R2(K, D, E)
- R3(A, A1, A2, A3)
- R4(B, B1, B2)
- R5(C, C1, C2, C3, C4, C5)
- w/ K a key value for R1 and R2.
- w/ A a key value for R3.
- w/ B a key value for R4.
- w/ C a key value for R5.

Example Tables: Data

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

R2

K	D	E
4	1	6
5	1	5
1	1	8
2	1	7
3	1	3

R4

B	B1	B2
0	0	0
3	9	27

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

Simplest SQL Operation: Projection from Relational Algebra

- $Y := \Pi_L (X)$
  - Select a set of attributes/columns of relation

Projection w/ SQL SELECT-FROM

SELECT attribute-names

FROM table-name **;**

Operators
Projection w/ SQL

- $Y := \pi_{C,C2}(X)$
- SQL:

```
SELECT C, C2
FROM X;
```

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

X

C	C2
4	0
5	0
1	3
2	3
3	3

SELECT C, C2
FROM X;

Operators
Projection w/ SQL

- Listing all tuples
- SQL:
  - SELECT *
  - FROM X

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

SELECT * FROM X

Operators
Selection

- $Y := \sigma_C(X)$
  - Select a set of rows of a relation
  - Based on conditional expression C
  - Operands in C are either attributes of relation X or constants.
  - Y includes only tuples that make C true.

Projection w/ SQL SELECT-FROM-WHERE

SELECT attribute-names

FROM table-name

WHERE condition**;**

Selection w/ SQL: WHERE Clause

- $Y := \sigma_{K<3}(X)$
- SQL:
  - SELECT *
  - FROM X
  - WHERE K<3;

X

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$\sigma_{K<3}(X)$

K	A	B	C
1	1	3	8
2	1	3	7

SELECT * FROM X WHERE K<3;

Selection & Projection w/ SQL: Select-From-Where Statements

- $Y := \sigma_{K<3}(X)$
- SQL:
  - SELECT K, A
  - FROM X
  - WHERE K<3;

X

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$\pi_{K,A}(\sigma_{K<3}(X))$

K	A
1	1
2	1

SELECT K, A FROM X WHERE K<3;

Selection & Projection w/ SQL: Select-From-Where Statements

- $Y := \sigma_{K<3}(X)$
- SQL:
  - SELECT K, A ($\pi_{K,A}$)
  - FROM X
  - WHERE K<3 ($\sigma_{K<3}$);

X

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$\pi_{K,A}(\sigma_{K<3}(X))$

K	A
1	1
2	1

SELECT K, A
FROM X
WHERE K<3;

SQL: Select-From-Where Statements

- Always remember, and never forget:
  - SELECT * FROM table ;
    - SELECT attributes
    - FROM table
    - WHERE attribute = value ;

SELECT desired attributes

FROM one or more tables

WHERE condition about tuples of the tables

Select-From-Where Statements

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

- Select C, C5 from R5;

C	C5
4	6
5	5
1	8
2	7
3	3

Select-From-Where Statements

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

- Select C, C5 from R5;
- $\pi_{C,C5}$ (R5)

C	C5
4	6
5	5
1	8
2	7
3	3

Select-From-Where Statements

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

- SELECT C, C5
- FROM R5
- WHERE C > 3;

C	C5
4	6
5	5

Select-From-Where Statements

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

- SELECT C, C5
- FROM R5
- WHERE C > 3;
- $\sigma_{C>3}(\pi_{C,C5}(R5))$

C	C2
4	6
5	5

Example:
Exercise – 2.4.1

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

a) What PC models have a speed of at least 3.00?

a) What PC models have a speed of at least 3.00?

- R1 := $\sigma_{speed \geq 3.00}$ (PC)
- R2 := π_{model}(R1)

model
1005
1006
1013

a) What PC models have a speed of at least 3.00?

- Step 1: Sele

```
1  #testing
2  use computers;
3
4
5  select * from pc;
6
7
```

sult Set Filter: Export: Wrap Cell Content:

model	speed	ram	hd	price
1001	2.66	1024	250	2114
1002	2.1	512	250	995
1003	1.42	512	80	478
1004	2.8	1024	250	649
1005	3.2	512	250	630
1006	3.2	1024	250	1049
1007	2.2	1024	250	510
1008	2.2	2048	250	770
1009	2	1024	250	650
1010	2.8	2048	300	770
1011	1.86	2048	160	959
1012	2.8	1024	160	649
1013	3.06	512	80	529

a) What PC models have a speed of at least 3.00?

- Step 2: Relational Algebra
  - $\sigma_{speed \geq 3.00}(PC)$

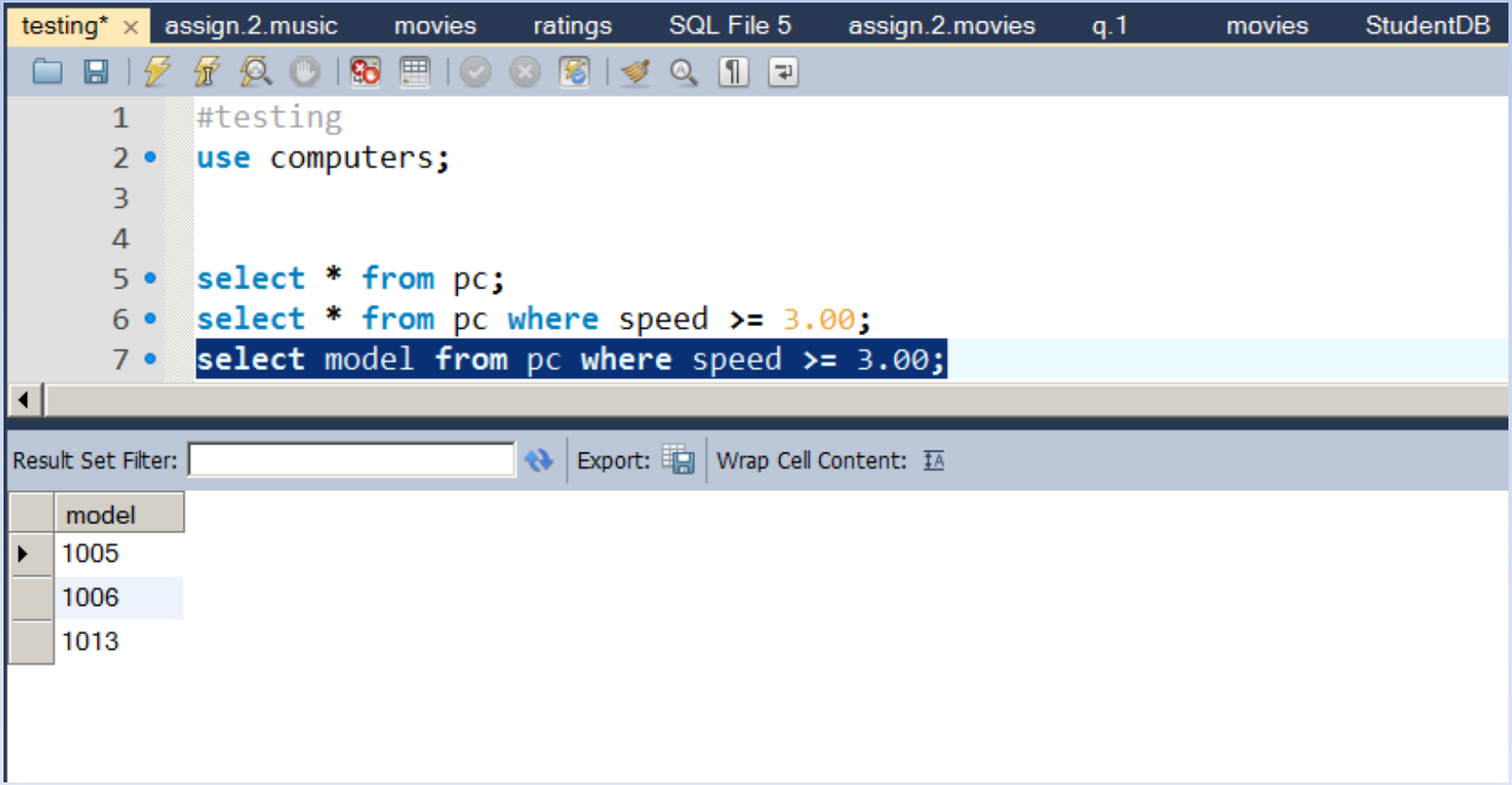
```
1  #testing
2  use computers;
3
4
5  select * from pc;
6  select * from pc where speed >= 3.00;
7
```

ult Set Filter: Export: Wrap Cell Content:

model	speed	ram	hd	price
1005	3.2	512	250	630
1006	3.2	1024	250	1049
1013	3.06	512	80	529

a) What PC models have a speed of at least 3.00?

- Step 3: Relational Algebra
  - R1 := $\sigma_{speed \geq 3.00}$ (PC)
  - R2 := π_{model}(R1)



Example:
Exercise – 6.1.3

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

a) Find the model number, speed, and hard-disk size for all PC's whose price is under $1000?

a) Find the model number, speed, and hard-disk size for all PC's whose price is under $1000?

- Step 1: Selection:
  - SELECT *
  - FROM PC
  - WHERE price < 1000 ;

a) Find the model number, speed, and hard-disk size for all PC's whose price

- Step
  - SE
  - FR
  - W
- Step
  - SE
  - FR
  - W

```
5  SELECT  model,
6          speed,
7          hd
8  FROM    PC
9  WHERE   price < 1000 ;
10
11
```

Result Set Filter: Export: Wrap Cell Content:

model	speed	hd
1002	2.1	250
1003	1.42	80
1004	2.8	250
1005	3.2	250
1007	2.2	250
1008	2.2	250
1009	2	250
1010	2.8	300
1011	1.86	160
1012	2.8	160
1013	3.06	80

PC 1

Output

Action Output

	Time	Action	Message
1	13:44:48	use computers	0 row(s) affected
2	13:44:54	SELECT model, speed, hd FROM PC WHERE price < 1000 LIMIT 0, 1000	11 row(s) returned

Renaming Attributes

- Attributes can be renamed:
  - Within Select Clause
  - Using Keyword: AS

SELECT **old-name** AS **new-name**

FROM table-name

Renaming Attributes

R5

C	C1	C2	C3	C4	C5
4	2	0	6	1	6
5	2	0	5	1	5
1	1	3	8	1	8
2	1	3	7	1	7
3	2	3	3	1	3

SELECT *C* AS x, *C2* AS y

FROM *R5*;

- $\rho_{x,y}(\pi_{c,c2}(R5))$

X	Y
4	0
5	0
1	3
2	3
3	3

Example:
Exercise – 6.1.3

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

b) Find the model number, speed, and hard-disk size for all PC's whose price is under $1000, but rename the *speed* column *gigahertz* and the *hd* column *gigabytes*?

b) Find the model number, speed, and hard-disk size for all PC's whose price is under $1000, but rename the *speed* column *gigahertz* and the *hd* column *gigabytes*?

- Step
- Step
- Step

testing* | Triggers.2 | StudentDB | iris.2D | assign.2.computers | searching | assign.2.computers | assign.2.music | movies

```
5  SELECT  model,
6          speed AS gigahertz,
7          hd    AS gigabytes
8  FROM    PC
9  WHERE   price < 1000 ;
10
11
```

Result Set Filter: | Export: | Wrap Cell Content:

model	gigahertz	gigabytes
1002	2.1	250
1003	1.42	80
1004	2.8	250
1005	3.2	250
1007	2.2	250
1008	2.2	250
1009	2	250
1010	2.8	300
1011	1.86	160
1012	2.8	160
1013	3.06	80

PC 2

Output

Action Output

	Time	Action	Message
1	13:44:48	use computers	0 row(s) affected
2	13:44:54	SELECT model, speed, hd FROM PC WHERE price < 1000 LIMIT 0, 1000	11 row(s) returned
3	13:55:27	SELECT model, speed AS gigahertz, hd AS gigabytes FROM PC WHERE pri...	11 row(s) returned

Constants & Expression w/ S Select Clause

- Select ‘title’, c + c1 AS total;

Example:
Exercise – 6.1.3

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

Find the model number and hard-disk size in megabytes for all PC's whose price is under $1000, and rename the *hd* column *megabytes*?

Find the model number and hard-disk size in megabytes for all PC's whose price is under $1000, and rename the *hd* column *megabytes*?

esting* × | Triggers.2 | StudentDB | iris.2D | assign.2.computers | searching | assign.2.computers | assign.2.music | movies | ratings | SC

```
3
4   SELECT  model,
5           hd*1000 AS megabytes
6   FROM    PC
7   WHERE   price < 1000 ;
8
```

esult Set Filter: | Export: | Wrap Cell Content:

model	megabytes
1002	250000
1003	80000
1004	250000
1005	250000
1007	250000
1008	250000
1009	250000
1010	300000
1011	160000
1012	160000
1013	80000

Matching Strings w/ LIKE's

- Comparing a string to a pattern:

  <Attribute> LIKE <pattern>

  <Attribute> NOT LIKE <pattern>

- *Pattern* is a quoted string with

  % = "any string."

  _ = "any character."

- SELECT * FROM person WHERE name LIKE "D%"
- SELECT * FROM person WHERE name LIKE "D_"

Matching Strings w/ Like

- SE
- FF
- W

```
91
92 • select * from person;
93 • select * from person where fname like 'Seedorf%';
94
```

ult Set Filter: Edit: Export/Import: Wrap Cell Content:

pid	lname	fname	gender	birth
116	Pete	Seedorf116	M	1985-06-13
120	Mary	Seedorf120	F	1947-04-30
122	Marius	Seedorf122	M	1960-07-21
124	Habib	Seedorf124	M	1941-12-23
132	Raj	Seedorf132	M	1972-02-22
133	Ginger	Seedorf133	F	1955-07-09
134	Shirley	Seedorf134	F	1947-05-21
139	Mary	Seedorf139	F	1943-01-18
143	Raj	Seedorf143	M	1964-10-18
162	Atha	Seedorf162	F	1957-09-21
167	Jane	Seedorf167	F	1981-09-06
169	Pete	Seedorf169	M	1969-02-16
170	Floyd	Seedorf170	M	1975-10-09
176	John	Seedorf176	M	1983-05-19
177	Mary	Seedorf177	F	1953-09-08
185	Jack	Seedorf185	M	1979-10-21

rson 5 ×

tput

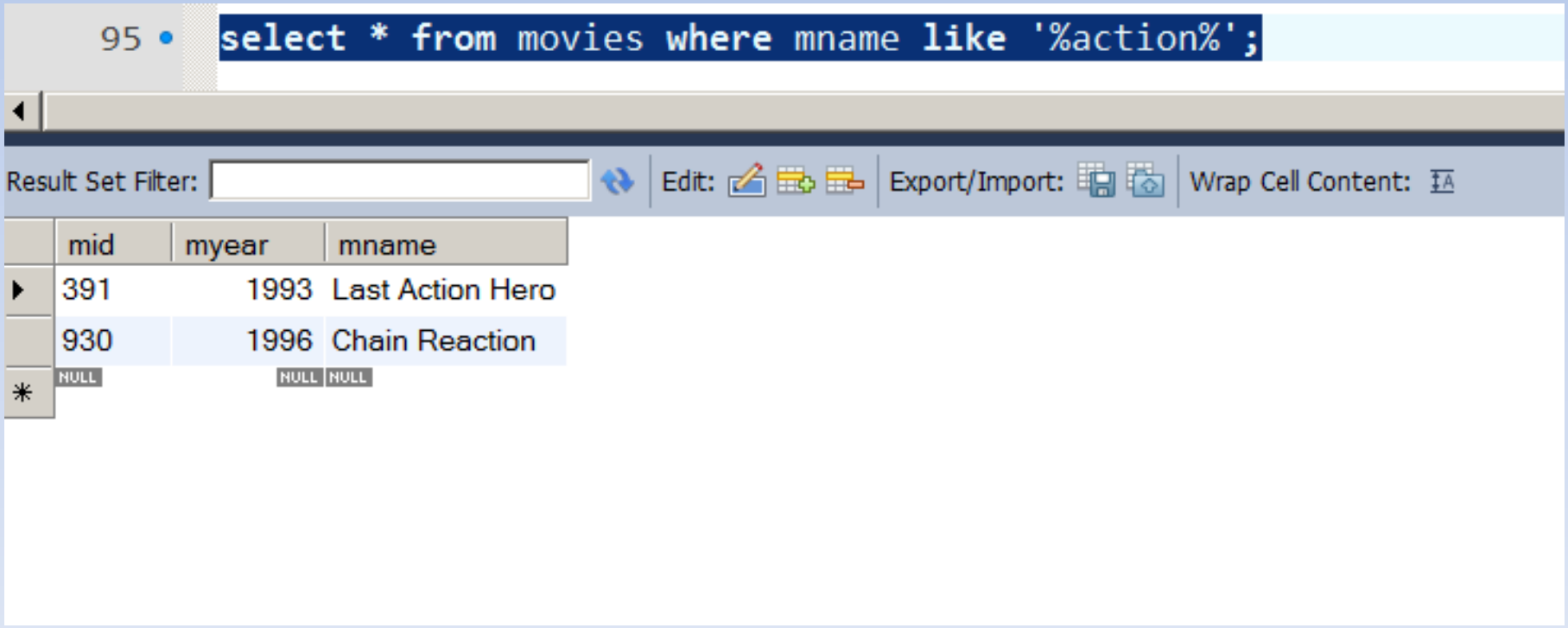
Action Output

	Time	Action	Message
16	09:53:49	alter table person change firstname lname varchar(20)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
17	09:53:52	select * from person where fname like 'Seedorf%' LIMIT 0, 1000	135 row(s) returned

Matching strings w/ Like

- Find all movies that have ‘action’ in title.

SELECT *



Matching strings w/ Like

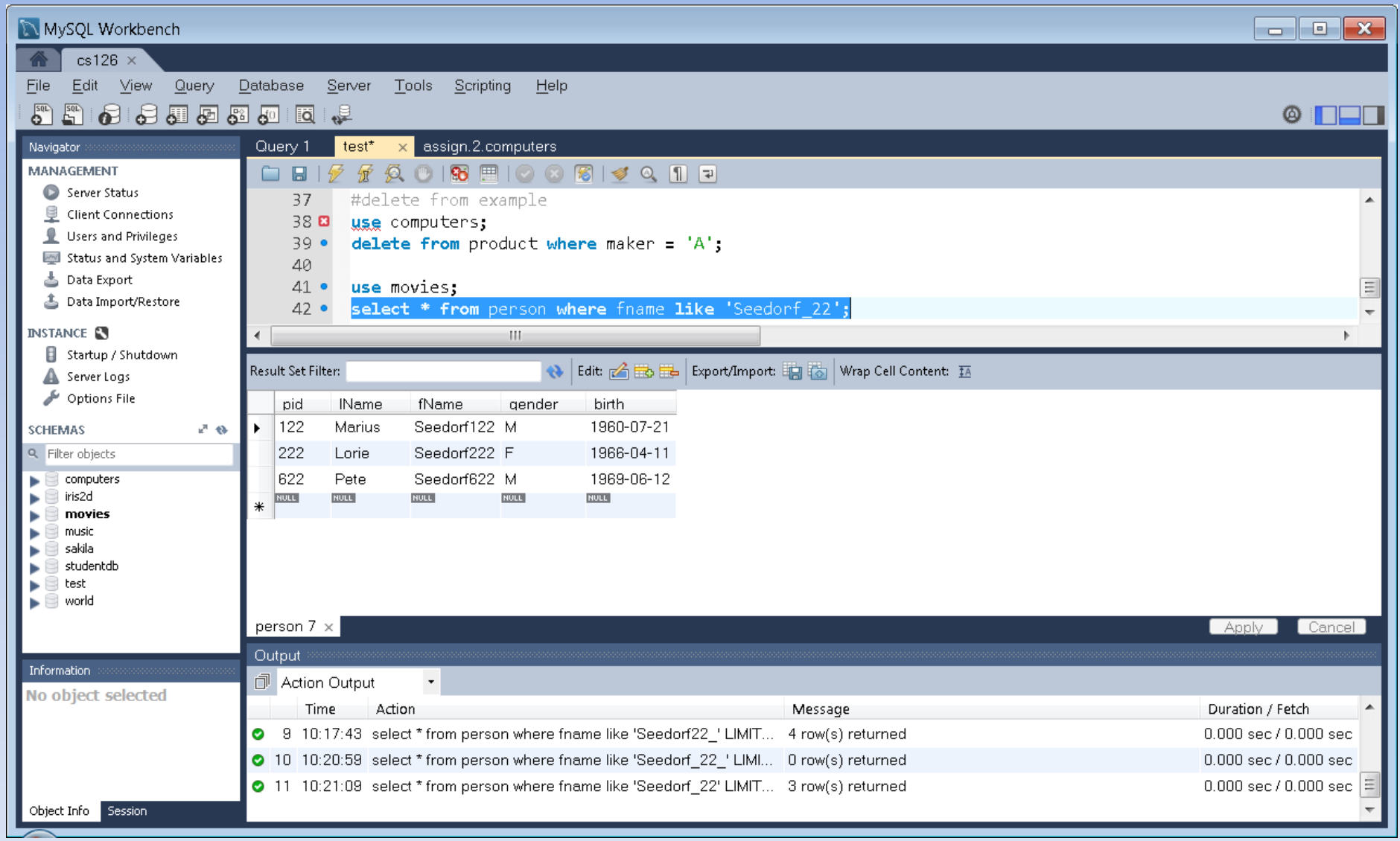
- Find all movies that have ‘action’ in title.

SELECT *

```
95 •  select * from movies where mname like '% action %';
```

Result Set Filter: Edit: Export/Import: Wrap Cell Conte

mid	myear	mname
391	1993	Last Action Hero
NULL	NULL	NULL


MySQL Workbench
cs126
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
SCHEMAS
Filter objects
computers
iris2d
movies
music
sakila
studentdb
test
world
Information
No object selected
Object Info
Session
Query 1
test*
assign.2.computers
37 #delete from example
38 use computers;
39 delete from product where maker = 'A';
40
41 use movies;
42 select * from person where fname like 'Seedorf_22';
Result Set Filter:
Edit:
Export/Import:
Wrap Cell Content:
pid lName fName gender birth
122 Marius Seedorf122 M 1960-07-21
222 Lorie Seedorf222 F 1966-04-11
622 Pete Seedorf622 M 1969-06-12
person 7
Apply
Cancel
Output
Action Output
Time Action Message Duration / Fetch
9 10:17:43 select * from person where fname like 'Seedorf22_' LIMIT... 4 row(s) returned 0.000 sec / 0.000 sec
10 10:20:59 select * from person where fname like 'Seedorf_22_' LIMI... 0 row(s) returned 0.000 sec / 0.000 sec
11 10:21:09 select * from person where fname like 'Seedorf_22' LIMIT... 3 row(s) returned 0.000 sec / 0.000 sec

Working w/ Dates

- DATE is a special string type in SQL:
  - ‘2014-02-04’
- YEAR(DATE):
  - Yields the year from the date value
  - SELECT YEAR(DATE '2014-02-04');
    - Yields ‘2014’

Assign.2.movies.sql

- CREATE TABLE Person(

  pid int primary key,

  lName varchar(20),

  fName varchar(20),

  gender char(1),

  birth date);

Assign.2.movies.sql

- Query: Find everyone born in 1970:

testing* × Triggers.2 StudentDB iris.2D assign.2.computers searching assign.2.computers assign.2.music movies ratings SQL File 11 assign.

```
29
30 •  SELECT *
31    FROM person
32    WHERE YEAR(birth) = 1970;
33
34
```

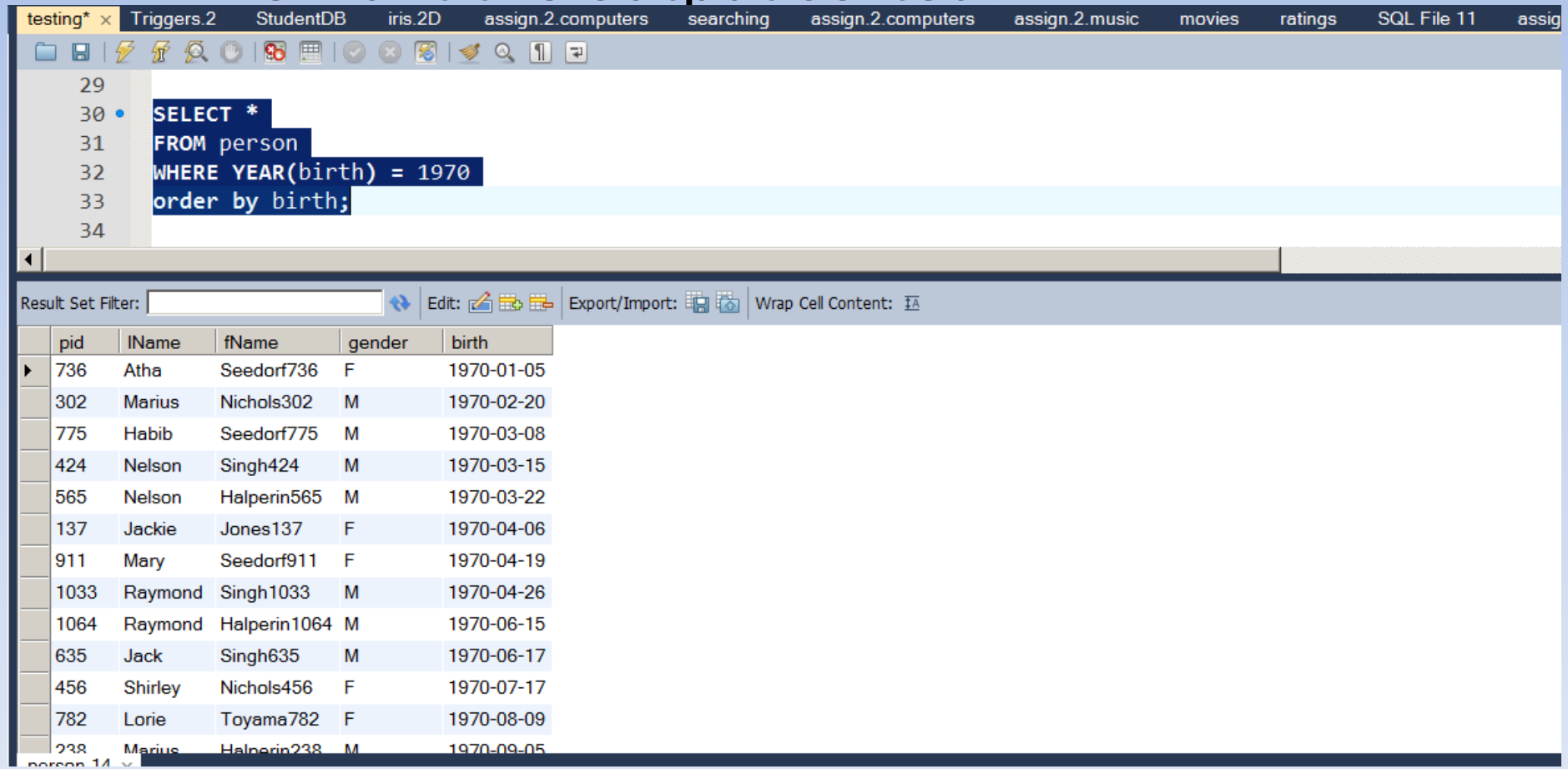
Result Set Filter: Edit: Export/Import: Wrap Cell Content:

pid	lName	fName	gender	birth
117	Sandy	Toyama117	F	1970-11-12
137	Jackie	Jones137	F	1970-04-06
238	Marius	Halperin238	M	1970-09-05
279	John	Nichols279	M	1970-11-29
302	Marius	Nichols302	M	1970-02-20
424	Nelson	Singh424	M	1970-03-15
456	Shirley	Nichols456	F	1970-07-17
565	Nelson	Halperin565	M	1970-03-22
635	Jack	Singh635	M	1970-06-17
669	Ginger	Seedorf669	F	1970-10-09
691	Buryl	Jones691	M	1970-09-11
736	Atha	Seedorf736	F	1970-01-05
775	Habib	Seedorf775	M	1970-03-08

person 13

Ordering Output

- If we want the output sorted:



pid	lName	fName	gender	birth
736	Atha	Seedorf736	F	1970-01-05
302	Marius	Nichols302	M	1970-02-20
775	Habib	Seedorf775	M	1970-03-08
424	Nelson	Singh424	M	1970-03-15
565	Nelson	Halperin565	M	1970-03-22
137	Jackie	Jones137	F	1970-04-06
911	Mary	Seedorf911	F	1970-04-19
1033	Raymond	Singh1033	M	1970-04-26
1064	Raymond	Halperin1064	M	1970-06-15
635	Jack	Singh635	M	1970-06-17
456	Shirley	Nichols456	F	1970-07-17
782	Lorie	Toyama782	F	1970-08-09
238	Marius	Halperin238	M	1970-09-05

Joining Tables

- SQL offers the ability to connect tables in several different ways.
- PRODUCT is the simplest, by including each relation in the FROM clause list (separated by commas)

Select-From-Where Statements

- SELECT * FROM R1, R4;

R4

B	B1	B2
0	0	0
3	9	27

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

R1 X R4

K	A	B	C	B	B1	B2
4	2	0	6	0	0	0
4	2	0	6	3	9	27
5	2	0	5	0	0	0
5	2	0	5	3	9	27
1	1	3	8	0	0	0
1	1	3	8	3	9	27
2	1	3	7	0	0	0
2	1	3	7	3	9	27
3	2	3	3	0	0	0
3	2	3	3	3	9	27

Select-From-Where Statements

- SELECT K, B2
  FROM R1, R4;

R4

B	B1	B2
0	0	0
3	9	27

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

K	B2
4	0
4	27
5	0
5	27
1	0
1	27
2	0
2	27
3	0
3	27

$\pi_{K,B2}$ (R1 X R4)

Product

Y1 X Y2

IN SQL:

```
SELECT *
FROM Y1, Y2;
```

Y1(

A,	B
1	2
3	4

)

Y2(

B,	C
5	6
7	8
9	10

)

(

A,	Y1.B,	Y2.B,	C
1	2	5	6
1	2	7	8
1	2	9	10
3	4	5	6
3	4	7	8
3	4	9	10

)

Select-From-Where Statements

- SELECT K, R1.B, R4.B, B2
  FROM R1, R4
  WHERE K < 3 ;

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$\sigma_{K<3}\,(\,\pi_{K,R1.B,R4.B,B2}\,(R1 \times R4)\,)$

K	R1.B	R4.B	B2
1	3	0	0
1	3	3	27
2	3	0	0
2	3	3	27

R4

B	B1	B2
0	0	0
3	9	27

Select-From-Where Statements

- SELECT K, B2
  FROM R1, R4
  WHERE K < 3 AND
    R1.B = R4.B ;

$$\sigma_{K<3 \text{ and } R1.B=R4.B}(\pi_{K,R1.B,R4.B,B2} (R1 \text{ X } R4))$$

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

K	R1.B	R2.B	B2
~~1~~	~~3~~	~~0~~	~~0~~
1	3	3	27
~~2~~	~~3~~	~~0~~	~~0~~
2	3	3	27

R4

B	B1	B2
0	0	0
3	9	27

Select-From-Where Statements

- SELECT K AS x, B2 AS y,
  FROM R1, R4
  WHERE K < 3 AND R1.B=R4.B;

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$$\rho_{x,y}(\sigma_{K<3}(\pi_{K,B2}(R1 \times R4)))$$

X	Y
1	27
2	27

R4

B	B1	B2
0	0	0
3	9	27

Select-From-Where Statements: Tuple Variables

- SELECT *
  FROM R1A, R1 B
  WHERE B.K < 3 AND A.K=B.C;

R1

K	A	B	C
4	2	0	6
5	2	0	5
1	1	3	8
2	1	3	7
3	2	3	3

$$\sigma_{B.K<3}\left(\rho_{A(K,A,B,C)}(R1) \times \rho_{B(K,A,B,C)}(R1) \right)$$

Example:
Exercise – 2.4.1

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

b) Which manufacturers make laptops with a hard disk of at least 100gb

b) Which manufactureers make laptops with a hard disk of at least

```
97  SELECT Product.maker
98  FROM Product, Laptop
99  WHERE Product.Model = Laptop.model AND
100    hd >= 100;
101
```

sult Set Filter: Export: Wrap Cell Content:

maker
A
B
E
F
G

maker
E
A
B
F
G

FROM Product, Laptop

WHERE Product.Model = Laptop.model AND

hd >= 100;

Example:
Exercise – 2.4.1

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

d) Find the model numbers of all color laser printers

Example

- Product(maker, model, type)
- PC(model, speed, ram, hd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)
- **Write the Relational Algebra to:**

Ex: 2.4.1.d) Find the model numbers of all color laser printers

R1 := $\sigma_{color = true\ AND\ type = laser}$ (Printer)

R2 := π_{model} (R1)

IN SQL:

```
SELECT model FROM printer
WHERE color and ctype='laser';
```

model
3003
3007

List all printer models and type, along with whether color or B&W

```
SELECT model, ctype,
   CASE color
      WHEN true THEN 'color'
      WHEN false THEN 'B&W'
      ELSE 'error'
   END as Color
FROM printer;
```

model	ctype	color
3001	ink-jet	color
3002	laser	B&W
3003	laser	color
3004	ink-jet	color
3005	laser	B&W
3006	ink-jet	color
3007	laser	color

List all printer makers, models, & prices

SELECT product.maker, product.model, printer.price

FROM product, printer

```
101
102 •  SELECT product.maker, product.model, printer.price
103    FROM product, printer
104    WHERE product.model = printer.model;
105
```

Result Set Filter: Export: Wrap Cell Content:

maker	model	price
D	3004	120
D	3005	120
E	3001	99
E	3002	239
E	3003	899
H	3006	100
H	3007	200

List all printer makers, models, & prices w/ TUPLE VARIABLES

SELECT m.maker, m.model, pr.price

```
101
102 • SELECT product.maker, product.model, printer.price
103   FROM product, printer
104   WHERE product.model = printer.model;
105
106 • SELECT m.maker, m.model, pr.price
107   FROM product m, printer pr
108   WHERE m.model = pr.model;
109
```

Result Set Filter: Export: Wrap Cell Content:

maker	model	price
D	3004	120
D	3005	120
E	3001	99
E	3002	239
E	3003	899
H	3006	100
H	3007	200

Example

- SQL Question :
- Schema:
  - Product(maker, model, type)
  - PC(model, speed, ram, hd, price)
  - Laptop(model, speed, ram, hd, screen, price)
  - Printer(model, color, type, price)
- Write SQL to find Maker, Model, Screen Size, and Price for laptops with screens larger than 15 inches.