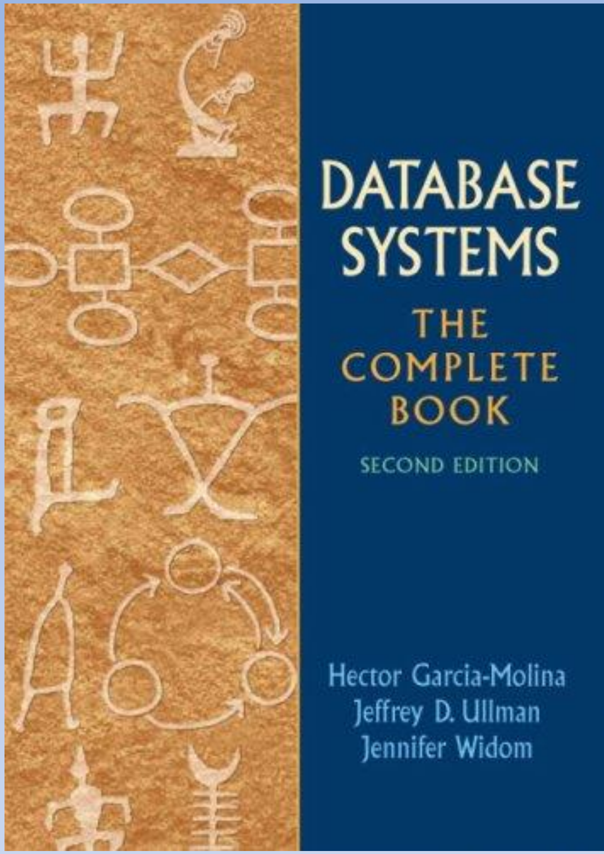


Database Systems
Chapter 2: Data Models

- Data Models defined.
- Look @ some Data Models
- Look @ Relational Data Model
- Look @ SQL in MySQL

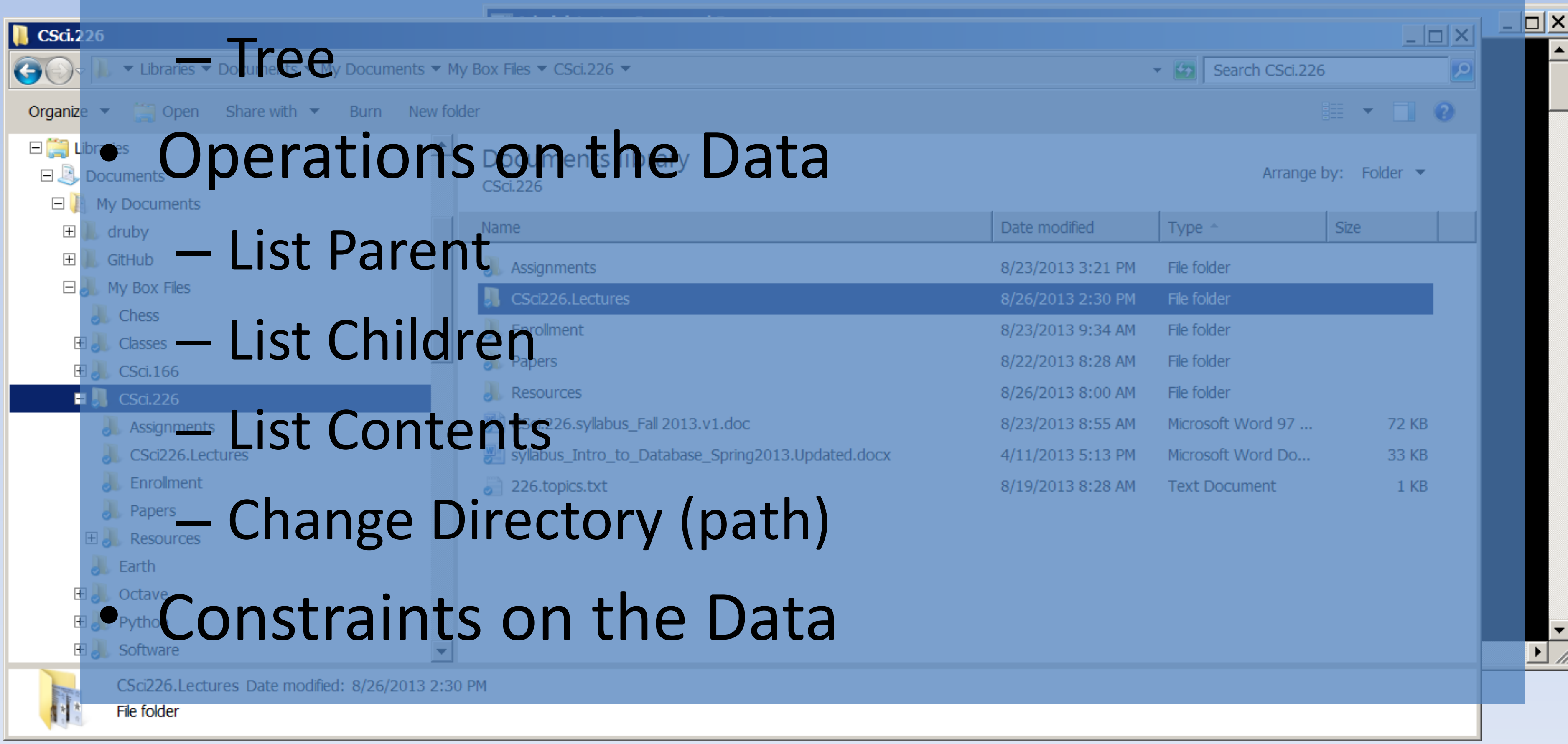
What’s a Data Model

- Structure of the Data
  - Tree, Graph
  - Semi-Structured (XML)
  - Probabilistic Graphs
  - Relational
- Operations on the Data
- Constraints on the Data

What’s do we get from Data Model

- Efficiency!
  - Provide us an abstraction for data.
  - Operations within abstractions optimized.
- Mathematical formalisms for proofs.

Tree Model
File System



- Structure of the Data
  - Tree
- Operations on the Data
  - List Parent
  - List Children
  - List Contents
  - Change Directory (path)
- Constraints on the Data

Semi-Structured Data (XML)

```
<Bookstore>
  <Book ISBN="ISBN-0-13-713526-2" Price="85" Edition="3rd">
    <Title>A First Course in Database Systems </Title>
    <Authors>
       <Auth> <FName>Jeffrey</FName> <LName>Ullman</LName>  </Auth>
       <Auth><FName>Jennifer</FName><LName>Widom</LName></Auth>
     </Authors>
  </Book>
    <Book ISBN="ISBN-0-13-815504-6" Price="100">
    <Remark> Buy this book bundled with "A First Course" - a great deal!</Remark>
    <Title>Database Systems: The Complete Book</Title>
    <Authors>
       <Auth><FName>Hector</First_Name><LName>Garcia-Molina</LName></Auth>
       <Auth><FName>Jeffrey</FName><LName>Ullman</LName></Auth>
       <Auth><FName>Jennifer</FName><LName>Widom</LName></Auth>
     </Authors>
   </Book>
</Bookstore>
```

XML

- Semantic Tags:
  - <Bookstore> *elements* </Bookstore>
- Attributes
  - <Book ISBN="ISBN-0-13-713526-2" Price="85" Edition="3rd"> </Book>
- Nested Elements:

```
<Book ISBN="ISBN-0-13-713526-2" Price="85" Edition="3rd">
  <Title>A First Course in Database Systems </Title>
  <Authors>
    <Auth> <FName>Jeffrey</FName> <LName>Ullman</LName>  </Auth>
    <Auth><FName>Jennifer</FName><LName>Widom</LName></Auth>
  </Authors>
</Book>
```

Data Model for Probability Distributions

- Probabilistic Graph Models
  - Designed for working with Complex Probability Distributions
- Example: Student Domain Variables
  - Course Difficulty (Difficulty)
  - Student grade in Course (Grade)
  - Quality of Recommendation Letter (Letter)
  - Student Intelligence (Intelligence)
  - Student SAT Score (SAT)
- Given a student's SAT score and Course Difficulty:
  - What is the probability student will receive an A.

Probabilistic Graph Models

- Bayesian Networks

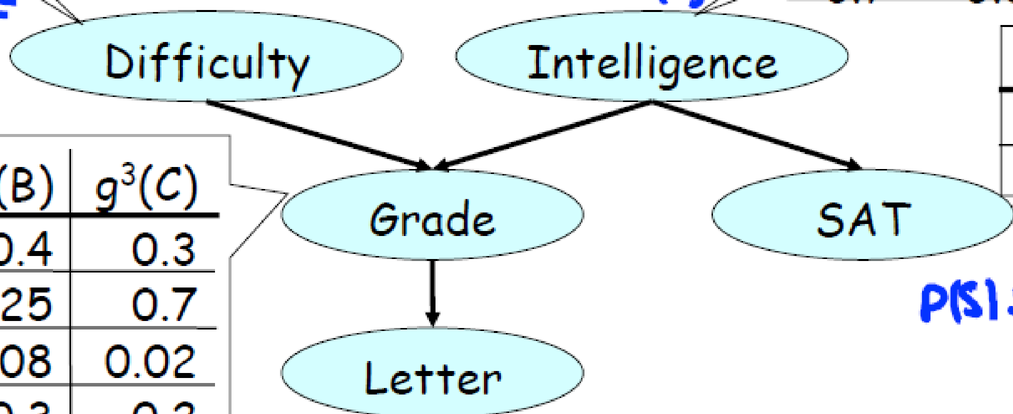
CPD = cond. prob. dist.

P(D)

d^0	d^1
0.6	0.4

P(I)

i^0	i^1
0.7	0.3



→P(G|I,D)

	g^1(A)	g^2(B)	g^3(C)
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

P(S|I)

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

P(L|G)

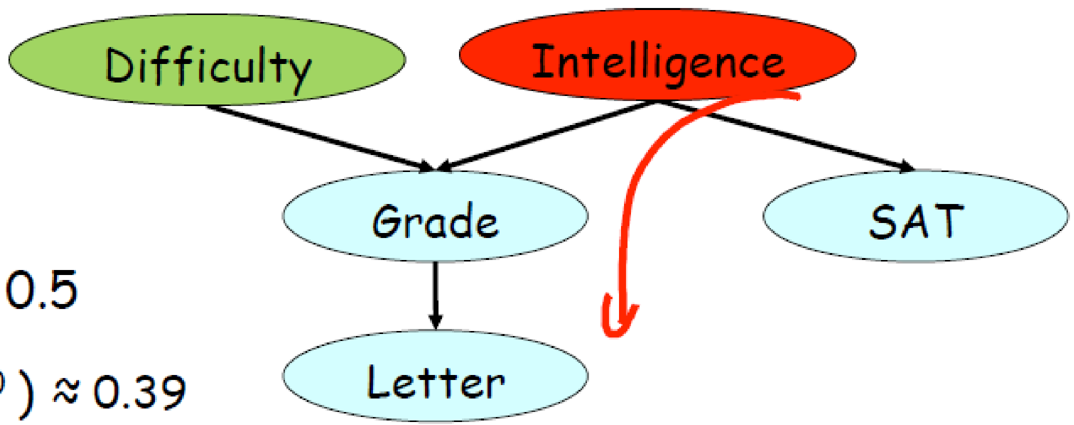
	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

Daphne Koller

Probabilistic Graph Models

- Causal Inference

Causal Reasoning



$P(l^1) \approx 0.5$

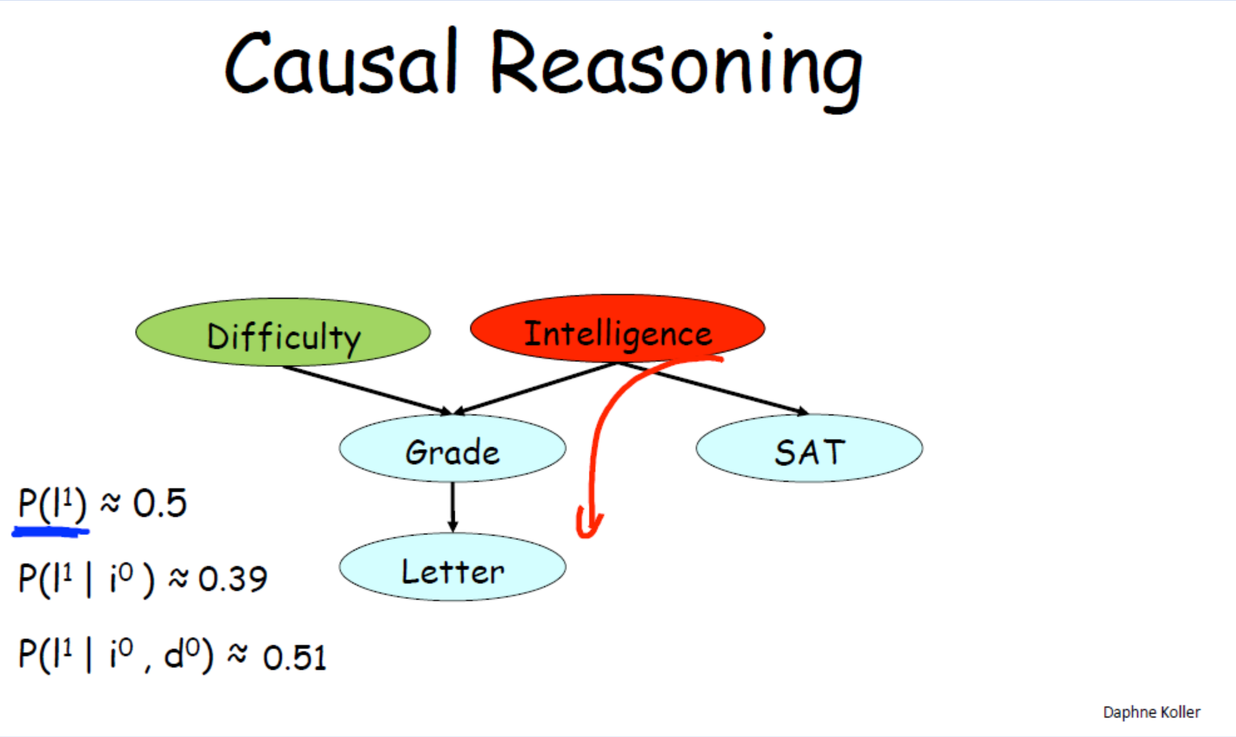
$P(l^1 \mid i^0) \approx 0.39$

$P(l^1 \mid i^0, d^0) \approx 0.51$

Daphne Koller

Probabilistic Graph Models

- Judea Pearl – Turing Award, 2011
  - For fundamental contributions to artificial intelligence through the development of a calculus for probabilistic and causal reasoning.



Relational Data Model

- Codd 1970
- Everything is a table
- Every row in a table has the same columns
- Relationships are implicit no pointers

Database Philosophy

God made the integers;
all else is the work of man.
(Leopold Kronecker, 19th Century Mathematician)

Codd made relations;
all else is the work of man.
(Raghu Ramakrishnan, DB text book author)

Map-Reduce Model

- Unlike the Relational Data Model and other models, it is a Programming Paradigm
- Existing MapReduce and Similar Systems
  - Google MapReduce
  - Support C++, Java, Python, Sawzall, etc.
  - Based on proprietary infrastructures and some open source libraries
- Hadoop Map-Reduce
  - Open Source!
  - HDFS, Map-Reduce, Pig, Zookeeper, HBase, Hive
  - Used by Yahoo!, Facebook, Amazon and Google-IBM NSF cluster
- Dryad
  - Proprietary, based on Microsoft SQL servers

Data Model - Relations

- Structure of the Data: Relations
  - Physical Data Model
  - Conceptual Data Model
- Operations on the Data:
  - Queries
  - Modification
  - Allows Optimizations
- Constraints on the Data

Relational Data Model

- Provides limited, yet useful, operations
- Provides framework for power of languages
  - SQL

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

Relational Data Model Terminology

- Attributes
  - Columns of relation
- Schema
  - Schema of Relation= Name + {Attributes}
  - Schema of Database = {Schema of Relations}

Relational Data Model Terminology

- Tuples:
  - Rows of relation
- Domains: Attributes have domains
- Relations are Sets of Tuples
- Instance of Relation
  - Set of tuples from relation
  - Current Instance = Set of tuples in relation NOW

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

- Attributes?
  - PersonID, FirstName, LastName, Gender, Birthdate

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

- Tuples?

(100, John, Doe100, M, 1960-10-10)

(101, John, Doe101, M, 1955-01-10)

(102, John, Doe102, F, 1955-10-01)

(103, John, Doe103, M, 1970-09-10)

(104, John, Doe104, F, 1977-07-07)

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

- Components of the first tuple?

  100 -> PersonID

  John -> FirstName

  Doe100 -> LastName

  M -> Gender

  1960-10-10 -> Birthdate

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

- The Relation Schema for Person?

Person(personID, firstName, lastName, gender, birthdate)

Relation Example
Person Relation

personID	firstName	lastName	gender	birthdate
100	John	Doe100	M	1960-10-10
101	John	Doe101	M	1955-01-10
102	Jane	Doe102	F	1955-10-01
103	John	Doe103	M	1970-09-10
104	Jane	Doe104	F	1977-07-07

- Suitable Domain for each attribute
  - personID: integer
  - firstName, lastName: string (varchar in MySQL)
  - gender: Character(1)
  - birthDate: Date

Person Relation: Equivalent Representation

personID	lastName	firstName	birthdate	gender
100	Doe100	John	1960-10-10	M
101	Doe101	John	1955-01-10	M
102	Doe102	Jane	1955-10-01	F
103	Doe103	John	1970-09-10	M
104	Doe104	Jane	1977-07-07	F

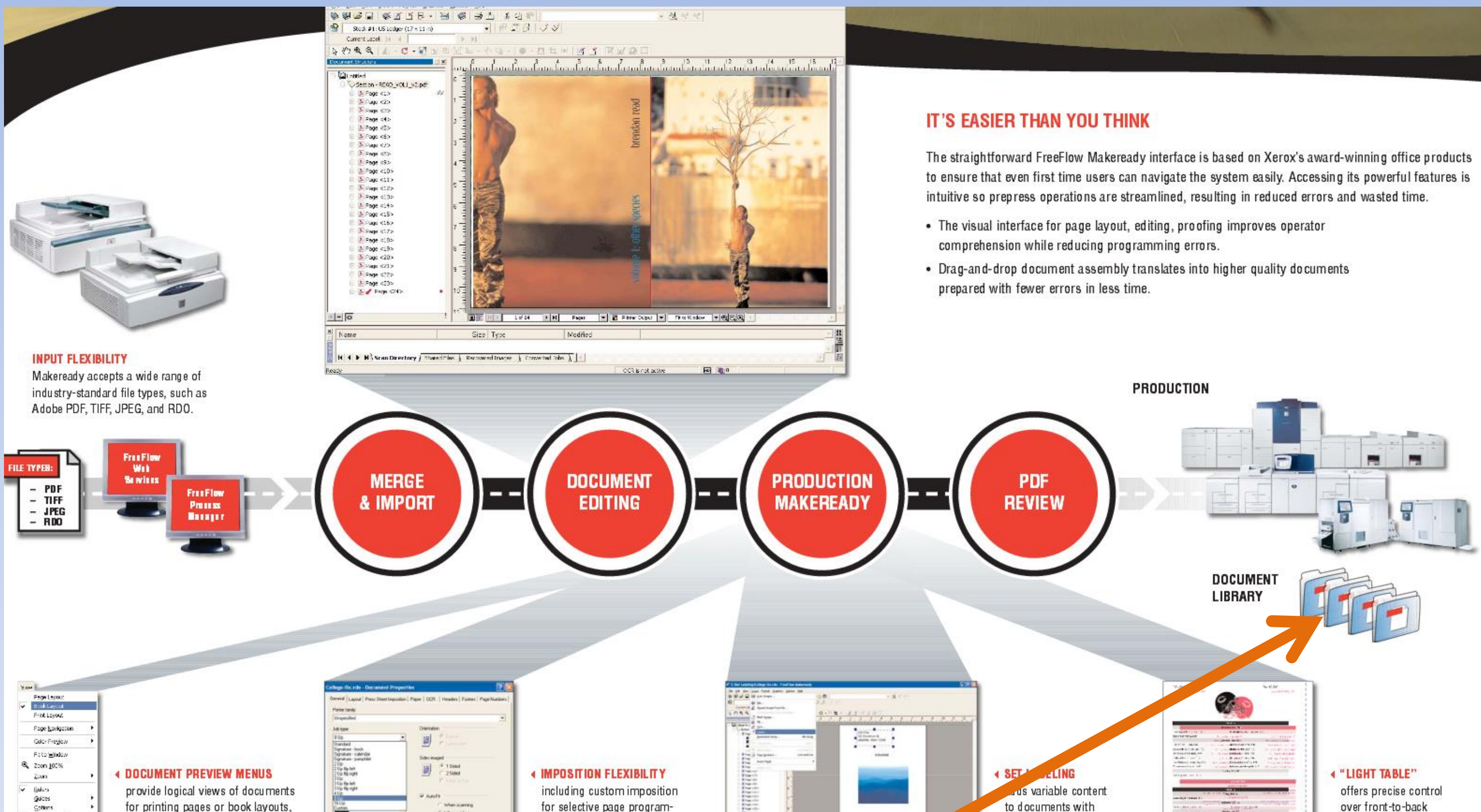
Relational Data Model Terminology

- Key Constaint
  - Attribute (or set) that are GUARANTEED unique.
- Key Examples:
  - SSN
  - Vehicle VIN
  - Student ID
  - Book ISBN

Relational Data Model: Part 2
Language: SQL (se.quel)

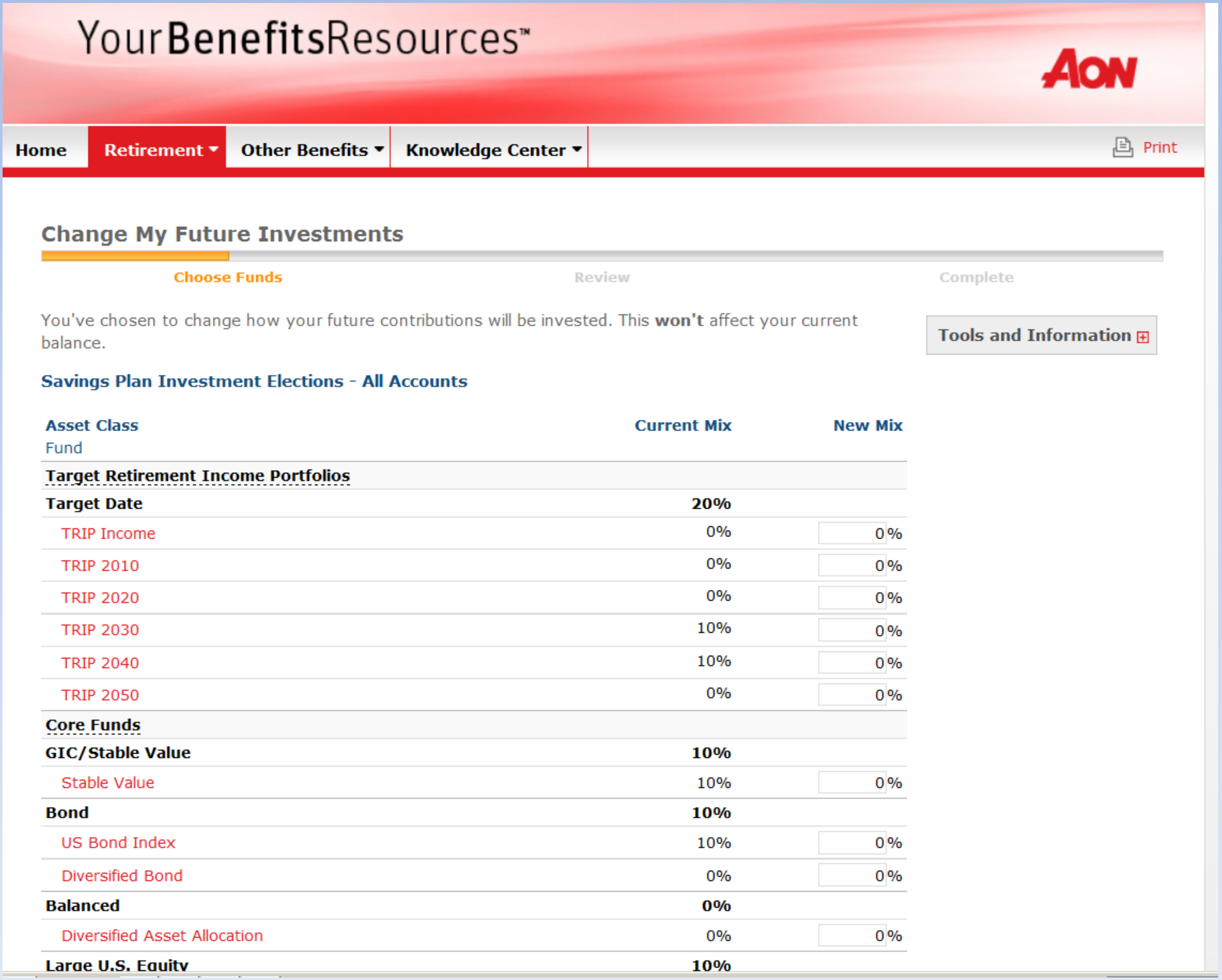
- Two Aspects to language
- Data Definition (ddl):
  - Declaring database schema: tables, constraints, indexes, views.
  - like declaring data/variables in programming language
- Data Manipulation (dml):
  - asking questions (querying) and modifying data.
  - Like executable code within programming language.

Database Example w/ Xerox FreeFlow



DataBase

Database Examples



YourBenefitsResources™

AON

Home | Retirement ▾ | Other Benefits ▾ | Knowledge Center ▾ | Print

Change My Future Investments

Choose Funds — Review — Complete

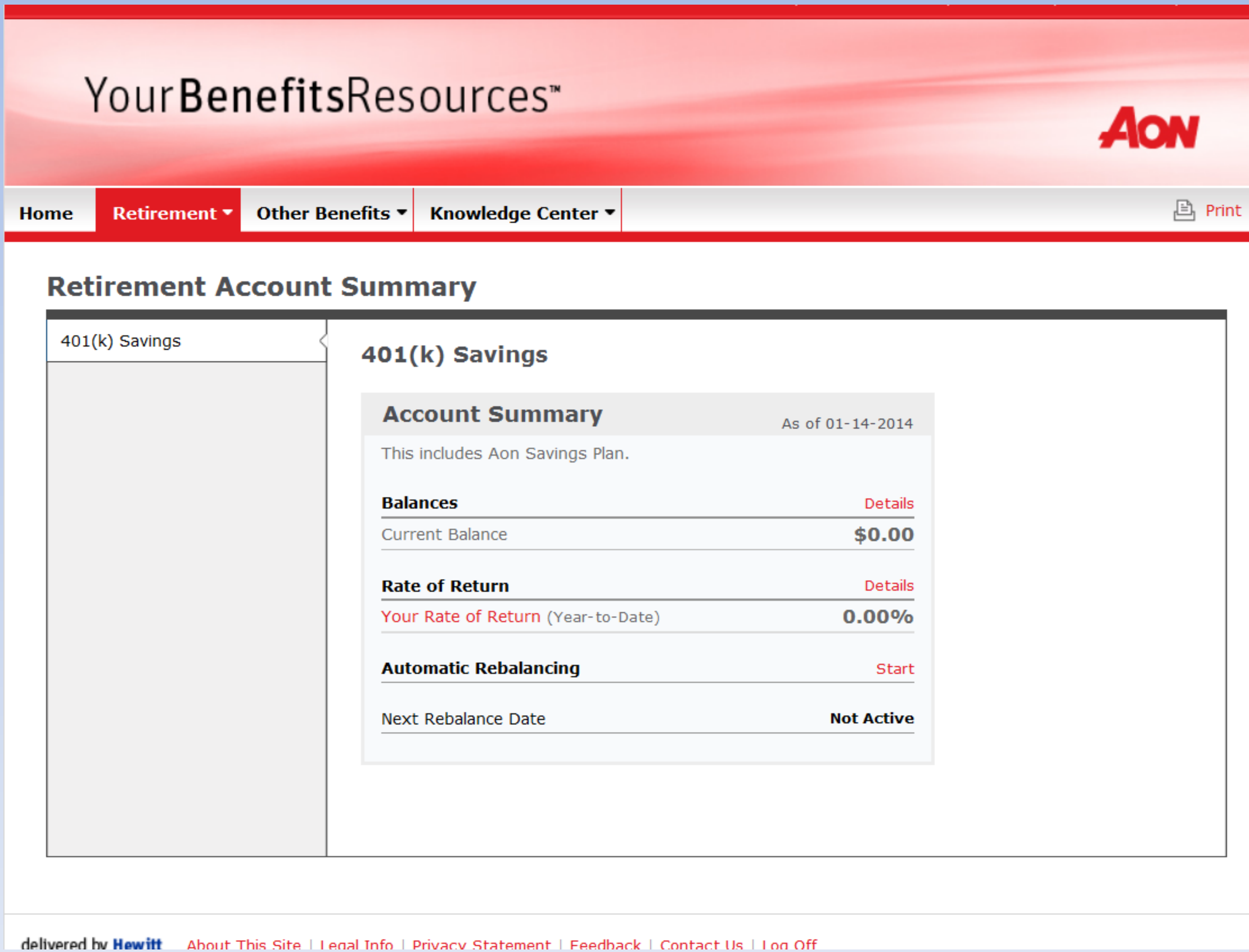
You've chosen to change how your future contributions will be invested. This **won't** affect your current balance.

Tools and Information

Savings Plan Investment Elections - All Accounts

Asset Class Fund	Current Mix	New Mix
Target Retirement Income Portfolios		
Target Date	**20%**	
TRIP Income	0%	0 %
TRIP 2010	0%	0 %
TRIP 2020	0%	0 %
TRIP 2030	10%	0 %
TRIP 2040	10%	0 %
TRIP 2050	0%	0 %
Core Funds		
GIC/Stable Value	**10%**	
Stable Value	10%	0 %
Bond	**10%**	
US Bond Index	10%	0 %
Diversified Bond	0%	0 %
Balanced	**0%**	
Diversified Asset Allocation	0%	0 %
Large U.S. Equity	**10%**	

Database Experience



Chapter 2: Relational Algebra

- Data Manipulation Language Still Needed
- Enter Relational Algebra
  - In commercial systems not used directly
  - SQL (in commercial systems used directly) has Relational Algebra as it’s center.
  - SQL query gets translated to Relational Algebra
  - Limited expressiveness a virtue.

Algebra

- What is Algebra?
  - Operands & Operators
  - Closure is usually needed.
- Let’s take a look @ Some Algebras

Algebra - Arithmetic

- What is Algebra?
  - Operands & Operators
  - Closure is usually needed.
- Algebra of Arithmetic
  - Operands include: variables (x, y, z, …), and constants (1, 4, 5, …)
  - Operators include: addition, subtractions, division, …

Algebra - Arithmetic

- What is Algebra?
  - Operands & Operators
  - Closure is usually needed.
- Algebra of Arithmetic
  - Operands include: variables (x, y, z, …), and constants (1, 4, 5, …)
  - Operators include: addition, subtractions, division, …

- **Linear Algebra**
  - Operands: Matrix variables and constants
  - Operators include Dot Product, Determinant, Transpose
  - Operations returns matrices, allowing operator composition
    - Building Expressions

Relation Algebra

- Operands:
  - Relations
  - Variables representing relations
- Operators:
  - Set operations
  - Slicing Relations
  - Gluing Relations
  - Renaming Relations
- Closure is Needed

Example Relation Schemas

- 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.

Current Instances for Relation Examples

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

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

R4

B	B1	B2
0	0	0
3	9	27

Set Operators Union/Intersection/Difference

- $X \cap Y$
- $X \cup Y$
- $Y - X$
  - Schemas must be identical

Operators
Union

- X ∪ Y

K	D	E
1	1	8
2	1	7

X

K	D	E
4	1	6
5	1	5

Y

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

X ∪ Y

Operators
Intersection

- $X \cap Y$

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

X

K	D	E
1	1	8
2	1	7
3	1	3

Y

K	D	E
1	1	8
2	1	7

$X \cap Y$

Operators
Difference

- X – Y:
- Set of elements in X but NOT IN Y
- Element of Y ALSO IN X are removed

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

X

K	D	E
1	1	8
2	1	7
3	1	3

Y

K	D	E
4	1	6
5	1	5

X – Y

Combining Operators

- Since each operation returns a Relation (closure) it can feed other operations.
- Can be viewed as an Expression Tree

Operators
Intersection as Difference

- $X \cap Y$: $X - (X - Y)$

X

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

Y

K	D	E
1	1	8
2	1	7
3	1	3

$X - Y$

K	D	E
4	1	6
5	1	5

$X - (X - Y)$

$X \cap Y$

K	D	E
1	1	8
2	1	7

Operators
Selection

- Y := σ_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.

Operators
Selection

- $Y := \sigma_{K<3}(X)$

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

Operators
Projection

- $Y := \pi_{L}(X)$
  - Select a set of attributes/columns of relation

Operators
Projection

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

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

$\pi_{C,C2}(X)$

Product

- Z := X **X** Y
  - Each rows of X attached to Each Possible row of Y

Product

- Z := R1 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

R4

B	B1	B2
0	0	0
3	9	27

Product

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

Product

Y3 := Y1 X Y2

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
5	6
7	8
9	10

Y3(

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

Natural Join

- Usually want to join tuples that in some way *match.*
- Natural Join requires matching attributes have matching values.
- R3 := R1 $\bowtie$ R2.
  - Take Product: R1 X R2
  - Take Result: σ_C
  - C is same named attributes are equal
  - Remove redundant attributes
- Dangling Tuples are tuples from one relation that have no match in the other tuple.

Product – Changed Instances

Y3 := Y1 X Y2

Y1(

A,	B)
1	**2**
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

A,	Y1.B,	Y2.B,	C)
1	2	2	11
1	2	5	9
1	2	7	2
3	4	2	11
3	4	5	9
3	4	7	2

Natural Join

$Y3 := Y1 \bowtie Y2$

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

A,	B,	C)
1	2	11

Natural Join:
Product & Selection & Projection

Y3 := Y1 X Y2

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

A,	Y1.B,	Y2.B,	C)
1	2	2	11
1	2	5	9
1	2	7	2
3	4	2	11
3	4	5	9
3	4	7	2

STILL NEED:

1. Take Result where matching attributes are equal: $Y4=\sigma_{Y1.b=Y2.b}$ (Y3)
2. Remove redundant attributes: $\pi_{A,Y1.B,C}$ (Y4)

Natural Join: Dangling Tuples

Y3 := Y1 X Y2

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	**9**
7	**2**

Y3(

A,	Y1.B,	Y2.B,	C)
1	2	2	11
1	**2**	**5**	**9**
1	**2**	**7**	**2**
3	**4**	**2**	**11**
3	**4**	**5**	**9**
3	**4**	**7**	**2**

STILL NEED:

1. Take Result where matching attributes are equal: $Y4 = \sigma_{Y1.b=Y2.b}$ (Y3)
2. Remove redundant attributes: $\pi_{A,Y1.B,C}$ (Y4)

Theta-Join

- R3 := R1 $\bowtie_C$ R2
  - Take Product: R1 X R2
  - Take Result: σ_C
  - C is boolean condition

Theta Join:
Product & Selection

$$Y3 := \sigma_{A<C}(Y1 \times Y2)$$

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

A,	Y1.B,	Y2.B,	C)
1	2	2	11
1	2	5	9
1	2	7	2
3	4	2	11
3	4	5	9
~~3~~	~~4~~	~~7~~	~~2~~

Theta Join:

Product & Selection

$$Y3 := \sigma_{A+B<C}\,(Y1 \text{ X } Y2)$$

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

A,	Y1.B,	Y2.B,	C)
1	2	2	11
1	2	5	9
~~1~~	~~2~~	~~7~~	~~2~~
3	4	2	11
3	4	5	9
~~3~~	~~4~~	~~7~~	~~2~~

Renaming

- R1 := $\rho_{R1(A1,\ldots,An)}$(R2)
  - makes R1 be a relation with attributes A1,…,A*n* and the same tuples as R2.

Renaming

$$Y3 := \rho_{R1(A,B1,B2,C)}(Y1 \times Y2)$$

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

R1(

A,	B1,	B2,	C)
1	2	2	11
1	2	5	9
1	2	7	2
3	4	2	11
3	4	5	9
3	4	7	2

Renaming & Set Operations

$$Y3 := \rho_{X,Y}(Y1) \cup \rho_{X,Y}(Y2)\ )$$

Y1(

A,	B)
1	2
3	4

Y2(

B,	C)
2	11
5	9
7	2

Y3(

X,	Y)
1	2
3	4
2	11
5	9
7	2

Precedence

- Precedence of relational operators:
  1. [σ, π, ρ] (highest).
  2. [X, $\bowtie$].
  3. $\cap$.
  4. [$\cup$, —]

Relational Algebra for Constraints

- Relation $R \neq \emptyset$ OR $R = \emptyset$
- Key Constraints
- Foreign Key Constraints
- Domain Value Constraints

Example 1: Key Constraint

- PC1 = PC2 = PC

$Y := \sigma_{PC1.model=PC2.model \text{ AND } PC.maker \neq PC2.maker}(PC1 \text{ X } PC2)$

- Key Constraint: $Y = \emptyset$

Example 2
Referential Integrity Constraint

$Y1 := \pi_{model}$ (Product)

$Y2 := \pi_{model}$ (PC)

- Referential Integrity Constraint: $Y2 \subseteq Y1$

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?

maker	model	type
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
D	1008	pc
D	1009	pc
D	1010	pc
D	3004	printer
D	3005	printer
E	1011	pc
E	1012	pc
E	1013	pc
E	2001	laptop
E	2002	laptop
E	2003	laptop
E	3001	printer
E	3002	printer
E	3003	printer
F	2008	laptop
F	2009	laptop
G	2010	laptop
H	3006	printer
H	3007	printer

Figure 2.20: Sample data for Product

model	speed	ram	hd	price
1001	2.66	1024	250	2114
1002	2.10	512	250	995
1003	1.42	512	80	478
1004	2.80	1024	250	649
1005	3.20	512	250	630
1006	3.20	1024	320	1049
1007	2.20	1024	200	510
1008	2.20	2048	250	770
1009	2.00	1024	250	650
1010	2.80	2048	300	770
1011	1.86	2048	160	959
1012	2.80	1024	160	649
1013	3.06	512	80	529

(a) Sample data for relation PC

model	speed	ram	hd	screen	price
2001	2.00	2048	240	20.1	3673
2002	1.73	1024	80	17.0	949
2003	1.80	512	60	15.4	549
2004	2.00	512	60	13.3	1150
2005	2.16	1024	120	17.0	2500
2006	2.00	2048	80	15.4	1700
2007	1.83	1024	120	13.3	1429
2008	1.60	1024	100	15.4	900
2009	1.60	512	80	14.1	680
2010	2.00	2048	160	15.4	2300

(b) Sample data for relation Laptop

model	color	type	price
3001	true	ink-jet	99
3002	false	laser	239
3003	true	laser	899
3004	true	ink-jet	120
3005	false	laser	120
3006	true	ink-jet	100
3007	true	laser	200

(c) Sample data for relation Printer

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?

$$\pi_{model}$$

$$\sigma_{speed \geq 3.00}$$

$$PC$$

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)

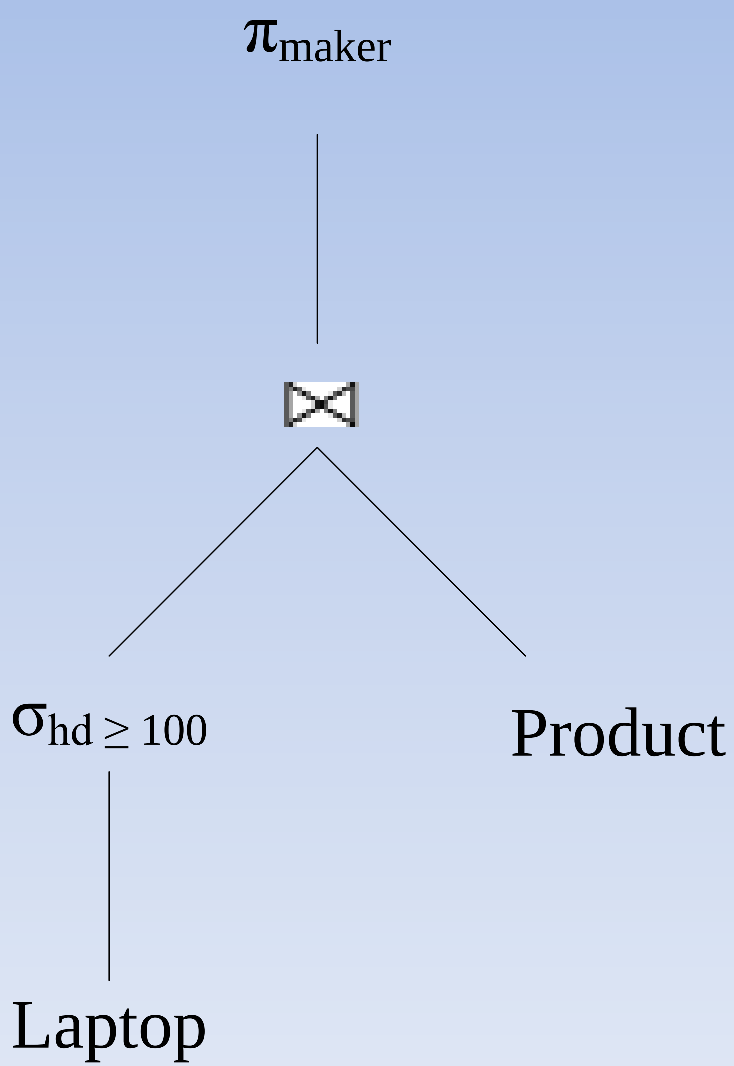
b) Which manufacturers make laptops with a hard disk of at least 100gb

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

- R1 := $\sigma_{hd \geq 100}$ (Laptop)
- R2 := Product (R1)
- R3 := π_{maker} (R2)

maker
E
A
B
F
G

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



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)

c) Find all model number and price of all products (of any type) made by manufacturer B

c) Find all model number and price of all products (of any type) made by manufacturer B

- $R1 := \sigma_{maker=B}\,(Product \bowtie PC)$
- $R2 := \sigma_{maker=B}\,(Product \bowtie Laptop)$
- $R3 := \sigma_{maker=B}\,(Product \bowtie Printer)$
- $R4 := \pi_{model,price}\,(R1)$
- $R5 := \pi_{model,price}\,(R2)$
- $R6: = \pi_{model,price}\,(R3)$
- $R7 := R4 \cup R5 \cup R6$

model	price
1004	649
1005	630
1006	1049
2007	1429

In-Class 2b
Ullman & Widom pg. 52

- 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

Ex: 2.4.1.e) Find those manufacturers (maker) that sell laptops, but not PC's.

maker	model	type
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
D	1008	pc
D	1009	pc
D	1010	pc
D	3004	printer
D	3005	printer
E	1011	pc
E	1012	pc
E	1013	pc
E	2001	laptop
E	2002	laptop
E	2003	laptop
E	3001	printer
E	3002	printer
E	3003	printer
F	2008	laptop
F	2009	laptop
G	2010	laptop
H	3006	printer
H	3007	printer

model	speed	ram	hd	price
1001	2.66	1024	250	2114
1002	2.10	512	250	995
1003	1.42	512	80	478
1004	2.80	1024	250	649
1005	3.20	512	250	630
1006	3.20	1024	320	1049
1007	2.20	1024	200	510
1008	2.20	2048	250	770
1009	2.00	1024	250	650
1010	2.80	2048	300	770
1011	1.86	2048	160	959
1012	2.80	1024	160	649
1013	3.06	512	80	529

(a) Sample data for relation PC

model	speed	ram	hd	screen	price
2001	2.00	2048	240	20.1	3673
2002	1.73	1024	80	17.0	949
2003	1.80	512	60	15.4	549
2004	2.00	512	60	13.3	1150
2005	2.16	1024	120	17.0	2500
2006	2.00	2048	80	15.4	1700
2007	1.83	1024	120	13.3	1429
2008	1.60	1024	100	15.4	900
2009	1.60	512	80	14.1	680
2010	2.00	2048	160	15.4	2300

(b) Sample data for relation Laptop

model	color	type	price
3001	true	ink-jet	99
3002	false	laser	239
3003	true	laser	899
3004	true	ink-jet	120
3005	false	laser	120
3006	true	ink-jet	100
3007	true	laser	200

(c) Sample data for relation Printer

- **Write the Relational Algebra to:**
  - Ex: 2.4.1.d) Find the model numbers of all color laser printers
  - Ex: 2.4.1.e) Find those manufacturers (maker) that sell laptops, but not PC's.

Answers Follow

In-Class 2b

- 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

Ex: 2.4.1.e) Find those manufacturers (maker) that sell laptops, but not PC’s.

In-Class 2b

- 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 := $\sigma_{\text{color = true AND type = laser}}$ (Printer)
R2 := π_{model} (R1)

CORRECT

model
3003
3007

In-Class 2b

- 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

π_{model} ($\sigma_{color\ =\ true\ AND\ type\ =\ laser}$ (Printer))

CORRECT

model
3003
3007

In-Class 2b

- 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 := $\sigma_{\text{color = true } \cancel{\text{AND type = laser}}}$ (Printer)
R2 := π_{model} (R1)

Partially Correct

model
3003
3007

In-Class 2b

- 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 := $\sigma_{color\cancel{=true}}$ (Printer)
R2 := $\sigma_{type\cancel{=laser}}$ (Printer)
R3 := R1 ⋈ R2

~~R3 := R1 ∩ R2~~
~~R4 := π_{model} (R3)~~

Partially Correct

model
3003
3007

In-Class 2b

- 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 := $\sigma_{color\cancel{=true}}$ (Printer)

Partially Correct

model
3003
3007

In-Class 2b

- 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 := $\pi_{model,color}$(Printer)
R2 := σ_{model} (R1)

Incorrect

R1

model	Color
3002	False
3003	True
3007	True

In-Class 2b

- 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.e) Find those manufacturers (maker) that sell laptops, but not PC's.

R1 := $\sigma_{type=laptop}$ (Product)
R2 := $\sigma_{type=PC}$(Product)
R3 := π_{maker}(R1)
R4 := π_{maker}(R2)
R5 := R3 – R4

CORRECT

maker
F
G

In-Class 2b

- 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.e) Find those manufacturers (maker) that sell laptops, but not PC's.

R1 := $\sigma_{type=laptop}$ (Product)
R2 := $\sigma_{type=PC}$(Product)
$\cancel{\pi_{maker}(R1\text{-}R2)}$

Correct: $\pi_{maker}(R1)-\pi_{maker}(R2)$

R1

maker	model	type
A	2004	laptop
A	2006	laptop
F	2008	laptop

R2

maker	model	type
A	1002	PC
A	1003	PC

In-Class 2b

- 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:**

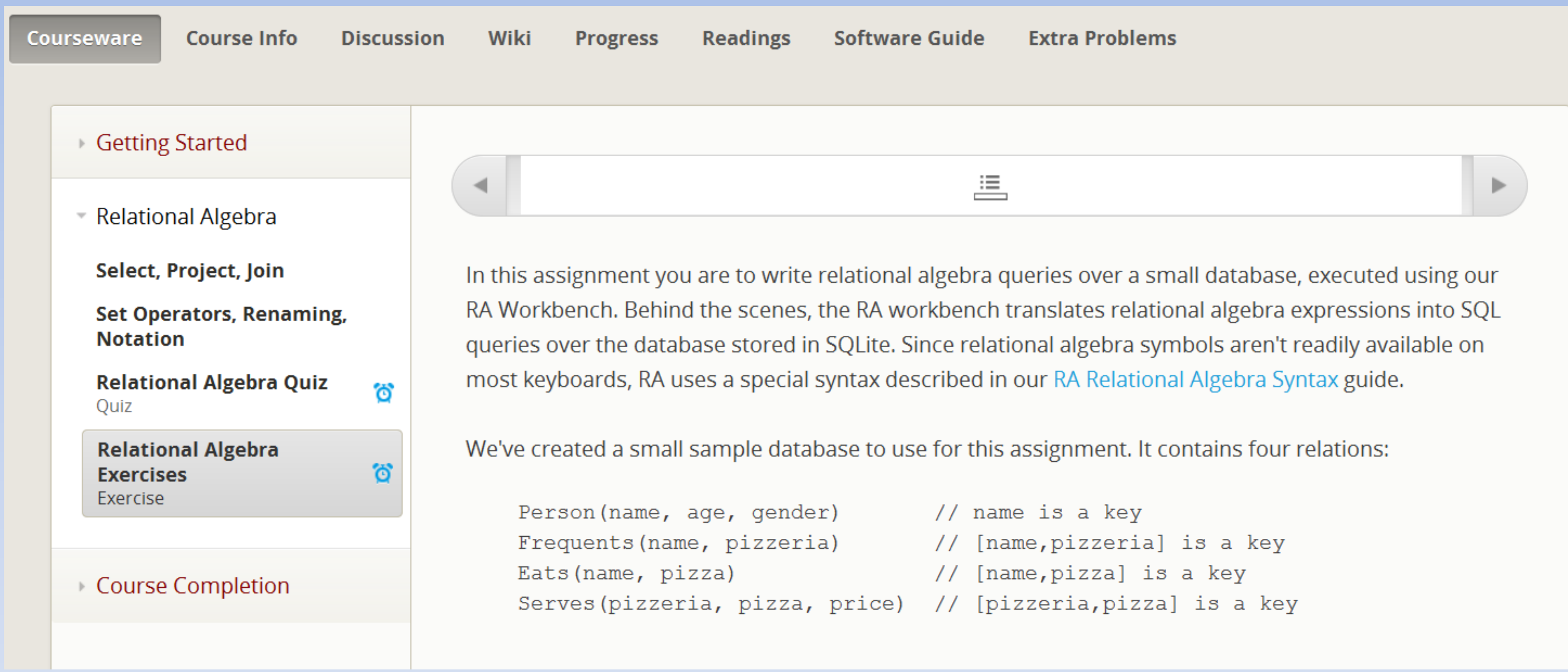
Ex: 2.4.1.e) Find those manufacturers (maker) that sell laptops, but not PC's.

~~$R1 := \sigma_{type=laptop}(Product)$~~
~~$R2 := \sigma_{type=PC}(Product)$~~
$R3 := \sigma_{maker}(laptop)$
$R4 := \sigma_{maker}(PC)$
$R5 := R3 – R4$

Partially Correct

maker
F
G

DB4



Courseware | Course Info | Discussion | Wiki | Progress | Readings | Software Guide | Extra Problems

Getting Started

Relational Algebra

Select, Project, Join

Set Operators, Renaming, Notation

Relational Algebra Quiz
Quiz

Relational Algebra Exercises
Exercise

Course Completion

In this assignment you are to write relational algebra queries over a small database, executed using our RA Workbench. Behind the scenes, the RA workbench translates relational algebra expressions into SQL queries over the database stored in SQLite. Since relational algebra symbols aren't readily available on most keyboards, RA uses a special syntax described in our RA Relational Algebra Syntax guide.

We've created a small sample database to use for this assignment. It contains four relations:

```
Person(name, age, gender)       // name is a key
Frequents(name, pizzeria)       // [name,pizzeria] is a key
Eats(name, pizza)               // [name,pizza] is a key
Serves(pizzeria, pizza, price)  // [pizzeria,pizza] is a key
```

RA: A Relational Algebra Interpreter

http://www.cs.duke.edu/~junyang/ra/

Introduction

RA is a simple relational algebra interpreter written in Java. It is built on top of an SQL-based relational database system. It implements relational algebra queries by translating them into SQL queries and executing them on the underlying database system through JDBC. RA is packaged with SQLiteJDBC, so you can use RA as a standalone relational-algebra database system. Alternatively, you can use RA as a relational-algebra frontend to other database systems.

RA Operators

- \select_{*cond*}
- \project_{*attr_list*}
- \join_{*cond*}
- \join
- \cross
- \union, \diff, and \intersect are the relational union, difference, and intersect operators.
- \rename_{*new_attr_name_list*} is the relational rename operator,

RA Operators

- \select_{*cond*}
  - is the relational selection operator.
  - For example, to select Drinker tuples with name Amy or Ben, we can write
    - \select_{name = 'Amy' or name = 'Ben'} Drinker;.
  - Syntax for *cond* follows SQL.
  - Note that string literals should be enclosed in ***single*** quotes, and you may use boolean operators and, or, and not.
  - Comparison operators <=, <, =, >, >=, and <> work on both string and numeric types.
  - For string match you can use the SQL LIKE operator;
    - \select_{name like 'A%'} drinker;
    - finds all drinkers whose name start with A, as % is a wildcard character that matches any number of characters.

RA Operators

- \project_{*attr_list*}
  - is the relational projection operator,
  - *attr_list* is a comma-separated list of attribute names.
  - For example, to find out what beers are served by Talk of the Town (but without the price information), we can write :

\project_{bar, beer} (\select_{bar = 'Talk of the Town'} Serves);

RA Operators

- \join_{*cond*}
  - is the relational theta-join operator.
  - For example, to join Drinker(name, address) and Frequents(drinker, bar, times_a_week) relations together using drinker name, we can write
    - Drinker \join_{name = drinker} Frequents;.
  - Syntax for *cond* again follows SQL; see notes on \select for more details.
- \join
  - is the relational natural join operator.
  - For example, to join Drinker(name, address) and Frequents(drinker, bar, times_a_week) relations together using drinker name, we can write
    - Drinker \join \rename_{name, bar, times_a_week} Frequents;.
    - Natural join will automatically equate all pairs of identically named attributes from its inputs (in this case, name), and output only one attribute per pair.
    - Here we use \rename to create two name attributes for the natural join; see notes on \rename below for more details.
- \cross
  - is the relational cross product operator.
  - For example, to compute the cross product of Drinker and Frequents, we can write Drinker
    - \cross Frequents;.

RA Operators

- \union, \diff, and \intersect are the relational union, difference, and intersect operators.
  - For a trivial example, to compute the union, difference, and intersection between Drinker and itself, we can write
    - Drinker \union Drinker;
      - which would return Drinker itself
    - Drinker \diff Drinker;
      - an empty relation
    - Drinker \intersect Drinker;
      - Drinker itself
- \rename_{*new_attr_name_list*}
  - is the relational rename operator, where *new_attr_name_list* is a comma-separated list of new names, one for each attribute of the input relation.
  - For example, to rename the attributes of relation Drinker and compute the cross product of Drinker and itself, we can write
    - \rename_{name1, address1} Drinker \cross \rename_{name2, address2} Drinker;

Example RA Expression

Here is an example of a complex query, which returns beers liked by those drinkers who do not frequent James Joyce Pub:

```
\project_{beer} (

  ((\project_{name}             // all drinkers
     Drinker)
   \diff
   (\rename_{name}              // rename so we can diff
      \project_{drinker}        // drinkers who frequent JJP
        \select_{bar = 'James Joyce Pub'}
          Frequents))

  \join_{drinker = name}        /* join with Likes to find beers */

  Likes
```

Example 2: RA Expression

We've created a small sample database to use for this assignment. It contains four relations:

```
Person(name, age, gender)         // name is a key
Frequents(name, pizzeria)         // [name,pizzeria] is a key
Eats(name, pizza)                 // [name,pizza] is a key
Serves(pizzeria, pizza, price)    // [pizzeria,pizza] is a key
```

View the database. (You can also download the schema and data.)

```
\project_{pizza, pizzeria}(Serves)
```

✖ Incorrect

Q1 (1 point possible)

Find all pizzas eaten by at least one female over the age of 20.

- View the RA Relational Algebra Syntax guide
- If you generate an error, you will see the message from the underlying SQLite system -- apologies for the lack of better error messages

```
1 \project_{pizza, pizzeria}(Serves)
```

Incorrect

Your Query Result:

cheese	Chicago Pizza
cheese	Dominos
cheese	Little Caesars
cheese	New York Pizza
cheese	Pizza Hut
cheese	Straw Hat
mushroom	Dominos
mushroom	Little Caesars
pepperoni	Little Caesars
pepperoni	New York Pizza
pepperoni	Pizza Hut
pepperoni	Straw Hat
sausage	Little Caesars

Expected Query Result:

cheese
mushroom
supreme

Reset Submit

Q9 (1 point possible)

Find all pizzerias that serve every pizza eaten by people over 30.

(This query is very challenging; extra congratulations if you get it right.)

- View the RA Relational Algebra Syntax guide
- If you generate an error, you will see the message from the underlying SQLite system -- apologies for the lack of better error messages

```
Enter your RA query here
```