

# Introduction to Data Science

Lecture 4; November 2<sup>nd</sup>, 2016

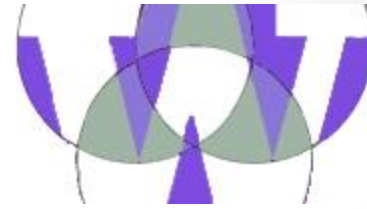
Ernst Henle

[ErnstHe@UW.edu](mailto:ErnstHe@UW.edu)

Skype: ernst-henle

( 1 )

# Agenda



- Announcements
  - The social component is a course requirement:
    - Contribute on LinkedIn and collaborate on homework!
  - Guest Lecture in November
    - Business Side of Data Science by Marius Marcu on November 16<sup>th</sup> 2016
- Review Homework (Accuracy Measures)
- Quiz 05a Confusion Matrix And Accuracy
- Data Structures
- Break
- Relational Algebra I
- Quiz 05b Relational Algebra Intro
- Relational Algebra II
- Break
- Relational Algebra II continued
- Quiz 05c Product Join Division
- Midpoint Retrospective
- Predictive Faux Pas
- Data as Sparse Matrices (Time Permitting)
- Assignment. See assignment slides at the end of the deck. (Complete all assignments items from all assignment slides. Submit by Saturday 11:57 PM)

# Accuracy Measures Review

# Homework Review

## 1. Training vs Test Data

- a) Question: In general, for any modeling data, why are performance metrics better on training data than on test data? Answer: Because the model was optimized for, or trained on, training data
- b) Question: Given modeling data, how do you determine which of this data will become training data and which data will become test data? Answer: Randomly partition the data into mutually exclusive datasets.
- c) You are given two datasets where one was the training data and the other is the test data. How can you determine which dataset is which? Answer: The data that result in better accuracy measures are probably the training data.

# Homework Review

- The Confusion Matrix
  - Calculate the accuracy measures including the F-measure for the Homework. Positive and negative are just points-of-view:
    - Illness is positive (as in a test to determine if one is ill)
    - Health is positive (as in: it's positive to be healthy)

# Homework Review

- A model was trained on 300 individuals where 149 had the cold and 151 were healthy.
  - These numbers are irrelevant.
  - The accuracy measures are assessed by predictions and the test data.
  - Accuracy is not assessed with the training data.
- The model was tested on 100 individuals where 10 were ill.
  - Total population: 100
  - Support for ill: 10
  - Therefore, support for healthy: 90
- The model correctly predicted that 85 of the healthy individuals were indeed healthy
  - Correct predictions of healthy: 85
  - Therefore, incorrect prediction of ill (they were actually healthy): 5
  - (90 healthy - 85 correct predictions of healthy -> 5 healthy that were not predicted as healthy)
- and correctly predicted that 7 of the ill individuals were indeed ill.
  - Correct predictions of ill: 7
  - Therefore, incorrect prediction of healthy (they were actually ill): 3
  - (10 ill - 7 correct predictions of ill -> 3 ill that were not predicted as ill)

# Homework Review

**85 predicted healthy and were healthy**  
**3 predicted healthy but were ill**  
**5 predicted ill but were healthy**  
**7 predicted ill and were ill**

- A model was trained on 300 individuals where 149 had the cold and 151 were healthy.
  - These numbers are irrelevant.
  - The accuracy measures are assessed by predictions and the test data.
  - Accuracy is not assessed with the training data.
- The model was tested on 100 individuals where 10 were ill.
  - Total population: 100
  - Support for ill: 10
  - Therefore, support for healthy: 90
- The model correctly predicted that 85 of the healthy individuals were indeed healthy
  - Correct predictions of healthy: 85
  - Therefore, incorrect prediction of ill (they were actually healthy): 5
  - (90 healthy - 85 correct predictions of healthy -> 5 healthy that were not predicted as healthy)
- and correctly predicted that 7 of the ill individuals were indeed ill.
  - Correct predictions of ill: 7
  - Therefore, incorrect prediction of healthy (they were actually ill): 3
  - (10 ill - 7 correct predictions of ill -> 3 ill that were not predicted as ill)

# Homework: Confusion Matrix

85 predicted healthy and were healthy
3 predicted healthy but were ill
5 predicted ill but were healthy
7 predicted ill and were ill



# Homework: Confusion Matrix

85 predicted healthy and were healthy  
3 predicted healthy but were ill  
5 predicted ill but were healthy  
7 predicted ill and were ill

Positive and negative are just points-of-view:

- Illness could be positive (as in a test to determine if one is ill)
- Health could be positive (as in: it's a positive thing to be healthy)

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
3 predicted healthy but were ill  
5 predicted ill but were healthy  
7 predicted ill and were ill

Health is Positive

		Actual	
		P	N
Predicted	P'	TP	FP
	N'	FN	TN

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
3 predicted healthy but were ill  
5 predicted ill but were healthy  
7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7
		Health is Positive	

		Actual	
		P	N
Predicted	P'	TP	FP
	N'	FN	TN

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7
		Health is Positive	

Illness is Positive

		Actual	
		P	N
Predicted	P'	TP	FP
	N'	FN	TN

		P	N
Predicted	P'	TP	FP
	N'	FN	TN

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

		Actual	
		P	N
Predicted	P'	TP	FP
	N'	FN	TN

		Actual	
		P	N
Predicted	P'	TP	FP
	N'	FN	TN

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
3 predicted healthy but were ill  
5 predicted ill but were healthy  
7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
3 predicted healthy but were ill  
5 predicted ill but were healthy  
7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

- True Positive: 85
- True Negative: 7
- False Positive: 3
- False Negative: 5

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

- True Positive: 85
- True Negative: 7
- False Positive: 3
- False Negative: 5

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

- True Positive: 7
- True Negative: 85
- False Positive: 5
- False Negative: 3



# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

- True Positive: 85
- True Negative: 7
- False Positive: 3
- False Negative: 5

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

- True Positive: 7
- True Negative: 85
- False Positive: 5
- False Negative: 3

- Sensitivity\*:  $tp / (tp + fn)$
- Specificity:  $tn / (tn + fp)$
- Accuracy:  $(tp + tn) / (tp + fp + tn + fn)$
- Precision :  $tp / (tp + fp)$
- Recall\*:  $tp / (tp + fn)$
- F-measure:  $2tp / (2tp + fn + fp)$

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

- True Positive: 85
- True Negative: 7
- False Positive: 3
- False Negative: 5

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

- True Positive: 7
- True Negative: 85
- False Positive: 5
- False Negative: 3

- Sensitivity\*:  $tp / (tp + fn)$
- Specificity:  $tn / (tn + fp)$
- Accuracy:  $(tp + tn) / (tp + fp + tn + fn)$
- Precision :  $tp / (tp + fp)$
- Recall\*:  $tp / (tp + fn)$
- F-measure:  $2tp / (2tp + fn + fp)$

- Sensitivity\*: 0.94
- Specificity: 0.7
- Accuracy: 0.92
- Precision: 0.97
- Recall\*: 0.94
- F-measure: 0.95

# Homework: Confusion Matrix

85 predicted healthy and were healthy  
 3 predicted healthy but were ill  
 5 predicted ill but were healthy  
 7 predicted ill and were ill

		Actual	
		P	N
Predicted	P'	85	3
	N'	5	7

Health is Positive

- True Positive: 85
- True Negative: 7
- False Positive: 3
- False Negative: 5

- Sensitivity\*:  $tp / (tp + fn)$
- Specificity:  $tn / (tn + fp)$
- Accuracy:  $(tp + tn) / (tp + fp + tn + fn)$
- Precision :  $tp / (tp + fp)$
- Recall\*:  $tp / (tp + fn)$
- F-measure:  $2tp / (2tp + fn + fp)$

- Sensitivity\*: 0.94
- Specificity: 0.7
- Accuracy: 0.92
- Precision: 0.97
- Recall\*: 0.94
- F-measure: 0.95

		Actual	
		P	N
Predicted	P'	7	5
	N'	3	85

Illness is Positive

- True Positive: 7
- True Negative: 85
- False Positive: 5
- False Negative: 3

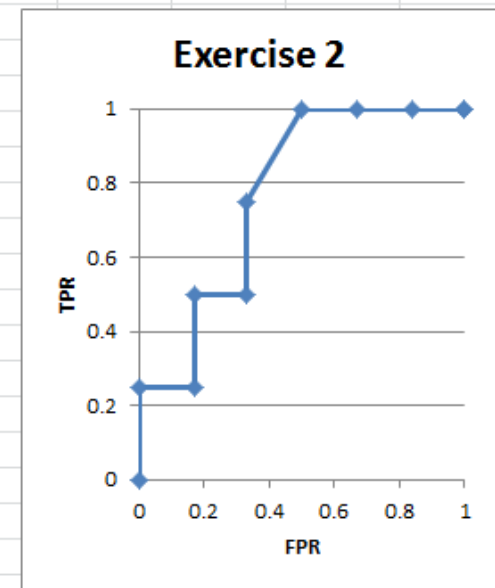
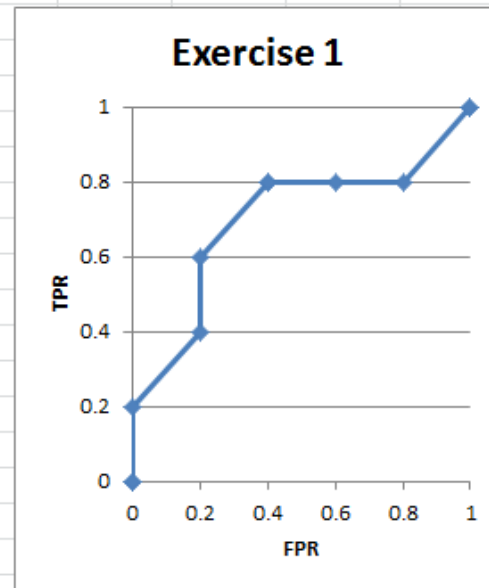
- Sensitivity\*: 0.7
- Specificity: 0.94
- Accuracy: 0.92
- Precision: 0.58
- Recall\*: 0.7
- F-measure: 0.63

# Homework: Make an ROC

- [HowToMakeAnROC\\_Results.xls](#)

Results: Exercise 1	
FPR	TPR
1	1
1	1
0.8	0.8
0.6	0.8
0.4	0.8
0.4	0.8
0.2	0.6
0.2	0.4
0	0.2
0	0
0	0

Results: Exercise 2	
FPR	TPR
1	1
1	1
0.84	1
0.67	1
0.5	1
0.33	0.75
0.33	0.5
0.17	0.5
0.17	0.25
0	0.25
0	0

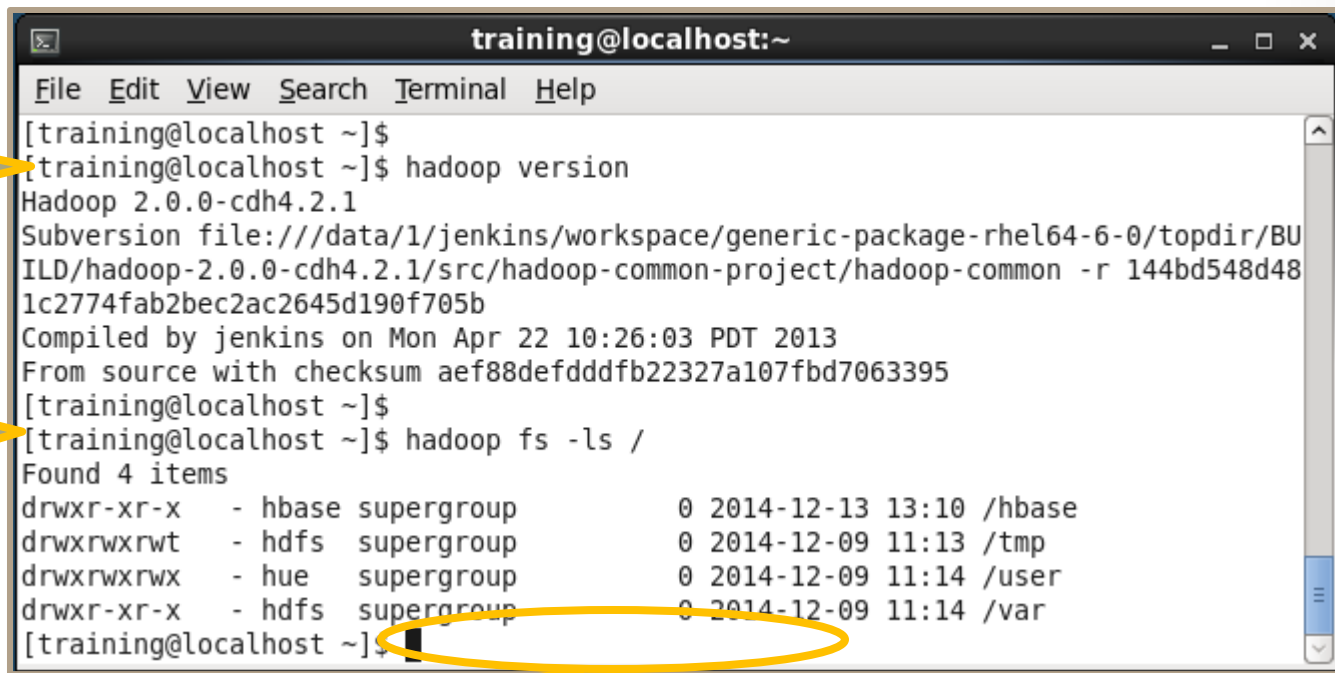


# Homework: Prepare VM

- Enter into Console: `hadoop version`
- Enter into Console: `hadoop fs -ls /`

Check that Hadoop is installed

List directories in HDFS



A terminal window titled 'training@localhost:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[training@localhost ~]$  
[training@localhost ~]$ hadoop version  
Hadoop 2.0.0-cdh4.2.1  
Subversion file:///data/1/jenkins/workspace/generic-package-rhel64-6-0/topdir/BU  
ILD/hadoop-2.0.0-cdh4.2.1/src/hadoop-common-project/hadoop-common -r 144bd548d48  
1c2774fab2bec2ac2645d190f705b  
Compiled by jenkins on Mon Apr 22 10:26:03 PDT 2013  
From source with checksum aef88defdddfb22327a107fbd7063395  
[training@localhost ~]$  
[training@localhost ~]$ hadoop fs -ls /  
Found 4 items  
drwxr-xr-x - hbase supergroup 0 2014-12-13 13:10 /hbase  
drwxrwxrwt - hdfs supergroup 0 2014-12-09 11:13 /tmp  
drwxrwxrwx - hue supergroup 0 2014-12-09 11:14 /user  
drwxr-xr-x - hdfs supergroup 0 2014-12-09 11:14 /var  
[training@localhost ~]$
```

A yellow oval highlights the prompt '[training@localhost ~]\$' at the bottom of the terminal window.

Type your name into the console and take a screen shot

# Homework:

## Classification Accuracy (0)

- ClassificationAccuracy.R

# Homework:

## Classification Accuracy (1)

- # Problem statement
  - # I A Classification is tested on 1000 cases.
  - # II The false positive rate is 0.4
  - # III The true positive rate is 0.8.
  - # IV The accuracy is 0.7.
- 
- # Problem statement expressed using TP, FP, FN, TN
  - # I  $N = TP + FP + FN + TN = 1000$
  - # II  $FPR = FP / (FP + TN) = 0.4$
  - # III  $TPR = TP / (TP + FN) = 0.8$
  - # IV  $(TP + TN) / (TP + FP + FN + TN) = 0.7$
- 
- # Problem statement expressed as linear equations
  - # I  $1*TP + 1*FP + 1*FN + 1*TN = 1000$
  - # II  $0*TP + 3*FP + 0*FN - 2*TN = 0$
  - # III  $1*TP + 0*FP - 4*FN + 0*TN = 0$
  - # IV  $-3*TP + 7*FP + 7*FN - 3*TN = 0$

# Homework:

## Classification Accuracy (2)

- # Problem statement expressed as linear equations
- # I  $1*TP + 1*FP + 1*FN + 1*TN = 1000$
- # II  $0*TP + 3*FP + 0*FN - 2*TN = 0$
- # III  $1*TP + 0*FP - 4*FN + 0*TN = 0$
- # IV  $-3*TP + 7*FP + 7*FN - 3*TN = 0$

- # Problem statement expressed in terms of linear algebra:

- # We want to solve the linear equation:  $Ax = b$

- # Where:

- # A is the matrix

- # x is a vector of TP, FP, FN, TN

- # b is the right-hand side of the linear equation

```

# -----
#   matrix A           vector b
# -----
# TP   FP   FN   TN   | b
# -----
#   1    1    1    1   | 1000
#   0    3    0   -2   | 0
#   1    0   -4    0   | 0
#  -3    7    7   -3   | 0
# -----

```

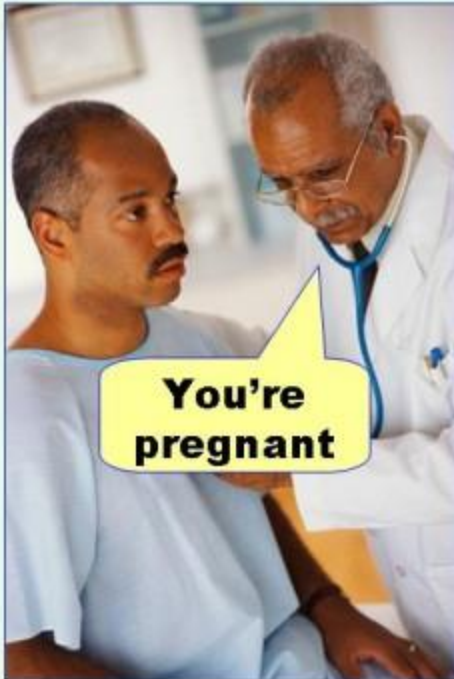


# Accuracy

- Links
  - [http://en.wikipedia.org/wiki/Accuracy\\_and\\_precision](http://en.wikipedia.org/wiki/Accuracy_and_precision)
  - [http://en.wikipedia.org/wiki/F1\\_score](http://en.wikipedia.org/wiki/F1_score)
  - [http://en.wikipedia.org/wiki/Precision\\_and\\_recall](http://en.wikipedia.org/wiki/Precision_and_recall)
- Exercise
  - Question 1: Why is the following statement both correct and useless? “My pregnancy test has a 89% accuracy”.
  - Question 2: What is the precision of the pregnancy test with the following measures?
    - A pregnancy test correctly predicted pregnancy 80% of the time among pregnant women.
    - 10% of all the women were predicted pregnant but were actually not pregnant.
    - The accuracy of the test was 89%.

# Pregnancy Test Exercise

**Type I error**  
(false positive)



**Type II error**  
(false negative)



# Pregnancy Test Exercise

- Question 1: Accuracy does not address Recall or Precision. For instance, 89% Accuracy could mean 89% TN and 0% TP. Both Recall and Precision would be 0%
- Question 2: Use ClassificationAccuracy.R (homework) as a template to complete PregnancyExercise.R
- **PregnancyExercise.R**
- Problem Statement
  - I  $TP + FN + FP + TN = 1$
  - II  $TP / (TP + FN) = \text{Recall} = 0.80$
  - III  $(TP + TN) / (TP + FP + TN + FN) = 0.89$
  - IV  $FP = 0.1$
- Algebra on statements II and III
  - II  $FN = TP * 0.20 / 0.80$
  - III\*I  $TN = 0.89 - TP$
- Substitute FN, TN, and FP:
  - I,II,III,
  - $TP * 0.$
  - $TP = 0$
- Results:
  - $TP = 0$
  - Precision

# Quiz 05a Confusion Matrix And Accuracy

- Confusion Matrix and Accuracy Measures
- Last question is similar to the last question of the homework review. Complete PregnancyExercise.R by using ClassificationAccuracy.R as an example

**An Expert is a person who tells you a simple thing in a confused way in such a fashion as to make you think the confusion is your own fault.**

*William Castle*

# Accuracy Measures Review

# Data Structures

# Data Structures (1)

## Terminology and Concepts

- Data
  - Dataset is a set of Data. A set implies a commonality. The commonality is expressed as a type or a relation.
  - A data type provides structure and meaning to the data. Just like there is no such thing as un-structured data, there is no such thing as un-typed data. Data can be insufficiently typed and structured.
- Rectangular Data
  - Datasets are often 2D matrices, which are organized into rows and columns. The column and row order is not important .
  - Columns are named with a header; A columns may be also referred to as an attribute or field. The number of columns is often called the dimensionality of the data.
  - Rows are not named. A row is often referred to as a case or observation. Number of rows in a category is called support.
- Data dimensionality
  - A data frame or a table can be considered a sparse multi-dimensional matrix
  - The dimensionality for un-supervised learning is #columns
  - The dimensionality for supervised learning is #columns - 1 because one column represents the value and not the dimension. This structure is very similar to a star schema

# Data Structures (2)

## Terminology and Concepts

- Predictive Analytics (Machine Learning , Artificial Intelligence)
  - Algorithms (often called Methods)
    - Supervised Learning
      - Classification
      - Estimation
    - Unsupervised Learning
      - Clustering
      - Association (Market-basket analysis)
      - Anomaly detection
    - Time Series
      - Forecasting (Arima)
      - Regression with time lags
      - Survival analysis



# Data Structures (3)

## Terminology and Concepts

- Supervised Learning Algorithms
  - Classification Algorithms predict classes or categories
    - Logistic Regression (Deterministic)
    - Decision Trees (Deterministic)
    - Naïve Bayes (Deterministic)
    - Neural Net (Non-Deterministic)
    - Random Forest (Non-Deterministic)
  - Estimation Algorithms predict continuous (numeric) values
    - Generalized Linear Modeling abbreviated: GLM (Deterministic)
      - Linear Regression
      - Logistic Regression
    - Regression Trees (Deterministic)
    - Neural Net (Non-Deterministic)

# Data Structures (4)

## Terminology and Concepts

- Un-Supervised Learning Algorithms
  - Segmentation Algorithms, also called Clustering, create clusters or segments. These clusters can be thought of as categories.
    - Mixture of Gaussians aka Probabilistic (Deterministic)
    - Hierarchical (Deterministic)
    - K-Means (Non-Deterministic)
  - Association Algorithms associate or link items by a common attribute called the transaction ID.
    - Market Basket Analysis (Deterministic)
    - Affinity Analysis (Deterministic)
  - Anomaly Detection is used to find unusual or anomalous data like outliers

# Data Structures (5)

## Terminology and Concepts

- Forecasting (Time Series) is used to estimate future values based on past behaviors.
  - ARIMA / Auto ARIMA
  - Survival Analysis

# Data Structures (6)

## Major types of Data Sets

- Univariate
- Rectangular
- Time Series
- Nested
- Graphs (later in the course)

# Data Structures (7)

## Univariate

- A collection of data. The data do not have a particular order. Example: Students' age. This type of data is often (mistakenly) called unstructured data, especially when the values are strings of indeterminate length. (Ragged Array)
- Example usage: anomaly detection.

# Data Structures (8)

## Univariate

<u>Parent Income</u>
40,000
53,000
60,000

# Data Structures (9)

## Rectangular Data

- The data set has columns and rows. Each cell has a value or is null.
- A Rectangular dataset is often called a matrix, data frame, or table.
- Example usage: classifications and estimations

# Data Structures (10)

## Rectangular Data

- Columns have descriptive headers like: Name, Age, Height, Weight of each student.
- Columns are also called attributes and fields.
- All values within a column have the same data type



# Data Structures (11)

## Rectangular Data

- Rows generally do not have names. If a row has a name, then the names could be considered another column.
- Rows are also called observations or cases
- The number of rows in a category is called support.

# Data Structures (12)

## Rectangular Data

<u>ID</u>	<u>IQ</u>	<u>Parent Income</u>	<u>Moral Support</u>	<u>Gender</u>	<u>College Plans</u>
835	107	40,000	Yes	Female	Applied
016	99	53,000	Yes	Male	Applied
490	105	60,000	No	Male	Did not apply

# Data Structures (13)

## Time Series

- A rectangular data set where the independent variable is time. The observations are sorted by time.
- Example usage: forecasting.

# Data Structures (14)

## Time Series

<u>Date</u>	<u>Red Wine Sales</u>	<u>White Wine Sales</u>	<u>Rose Sales</u>
1/22/13	\$103.00	\$300.50	\$19.00
1/23/13	\$35.50	\$204.00	\$44.00
1/24/13	\$217.50	\$74.50	\$80.00

# Data Structures (15)

## Nested

- A rectangular data set where a cell contains a table. The nested structure can have a flat representation that is not nested.
- Example usage: associations (shopping basket analyses).

# Data Structures (16)

## Nested

<u>Transaction ID</u>	<u>Item</u>
1	<div>Milk</div> <div>Sugar</div>
2	<div>Lumber</div>
3	<div>Milk</div> <div>Sugar</div> <div>Flour</div>

Each cell in Item contains a  
table of items

# Data Structures (17)

## Nested

<u>Transaction ID</u>	<u>Item</u>
1	Milk
1	Sugar
2	Lumber
3	Milk
3	Sugar
3	Flour

Replicate the transaction IDs

# Data Structures (18)

## Nested

<u>Transaction ID</u>	<u>Item</u>
---------------------------	-------------

1	Milk
1	Sugar

2	Lumber
---	--------

3	Milk
3	Sugar
3	Flour

Flatten the table



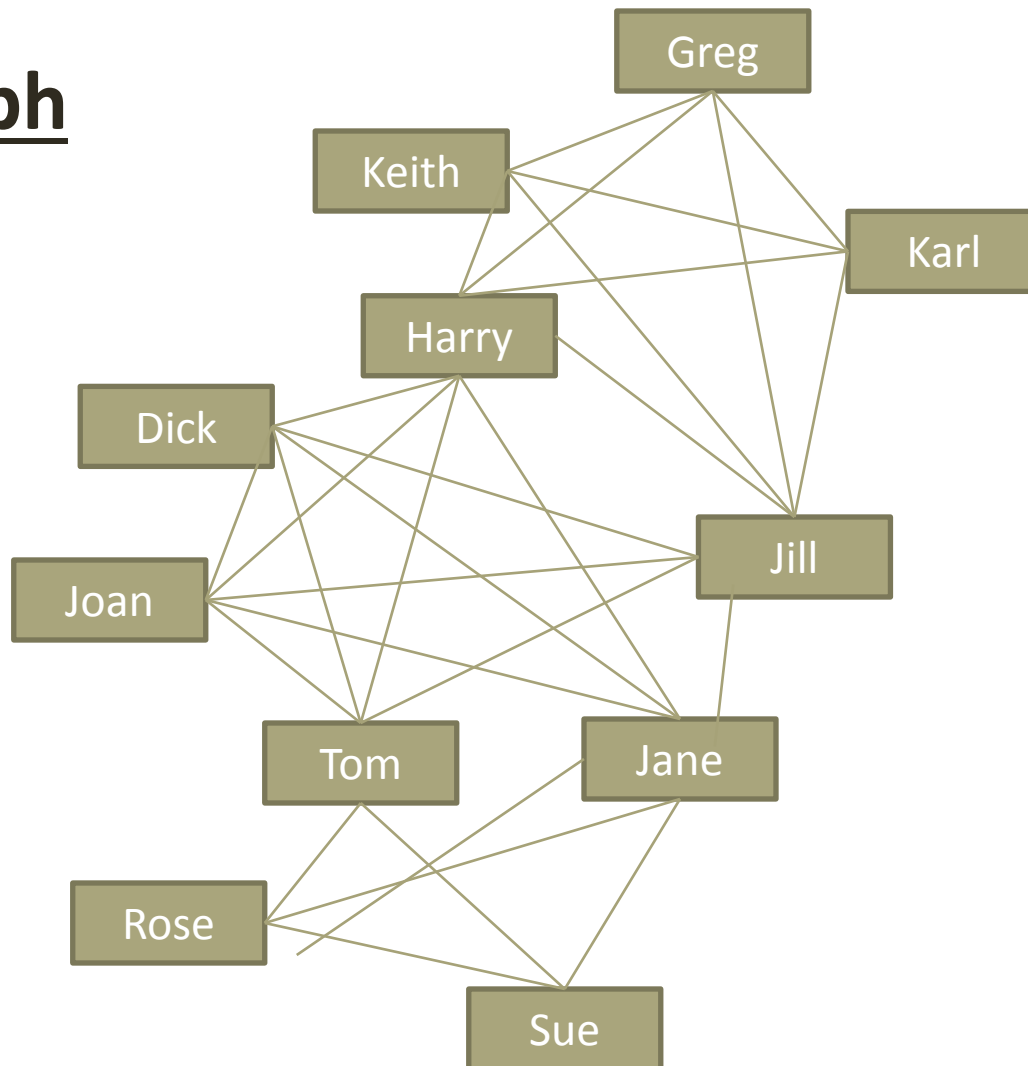
# Data Structures (19)

## Nested

<u>Transaction ID</u>	<u>Item</u>
1	Milk
1	Sugar
2	Lumber
3	Milk
3	Sugar
3	Flour

# Data Structures (20)

## Graph



# Data Structures

# Break



# Relational Algebra

The Theory behind Relational Databases

# Relational Algebra: What and Why

- Ted Codd introduced relational algebra to databases and created the relational model.
- Relational algebra provides a theoretical foundation for relational databases, and particularly for query languages like SQL.
- Why do you want a theoretical foundation?
  - If you want to optimize a query or a database
  - If you are thinking about using NOSQL, then you should be aware of the limitations and advantages of NOSQL data management. In other words, relational algebra assists in comparing SQL with NOSQL (NOT-SQL, Not-Oonly-SQL, KNOW-SQL, [http://www.youtube.com/watch?v=sh1YACOK\\_bo](http://www.youtube.com/watch?v=sh1YACOK_bo))
- Use these files as examples for this lecture:
  - RelationalAlgebraAndSQL.pdf
  - RelationalAlgebraAndSQL.sql

# New Terminology (1)

Term	Comments
<u>Table</u>	Part of a database
<u>Relation</u>	A table where rows are unique. Operand in Relational Algebra/Calculus
<u>Tuple</u>	<u>single</u> , <u>double</u> , <u>triple</u> , <u>quadruple</u> , <u>quintuple</u> , <u>sextuple</u> ; Like a row in a table
<u>Arity</u>	<u>unary</u> , <u>binary</u> , <u>ternary</u> , <u>quaternary</u>
<u>Closure</u>	Operation on a type produces a value of that same type. Natural Numbers have closure under + and * ( $3 * 5 = 15$ ) Natural Numbers do not have closure under – or /; $5 - 3 = -2$

# New Terminology (2)

Term	Comments
<u>Procedural</u>	Step-by-step solution to solving problem or achieving goal. I will drive to Bellevue, enter the class room and listen to the lecture. (Relational Algebra is <u>procedural</u> or <u>imperative</u> )
<u>Declarative</u>	Stating what one wants in non-ambiguous terms without describing how one is to achieve ones goal. Example: I want to know what was said in class last week. I don't care if you use the slide deck, your memory, or the recording to get me that information. (SQL is <u>declarative</u> )
<u>Relational Algebra</u>	The algebra that describes relations as operands and results
<u>Relational Calculus</u>	The calculus that uses relations as operands and results (SQL)



# New Terminology (3)

Operation	Symbols	Comments
<u>Selection</u>	$\sigma$ (sigma); $\sigma_{\phi}(R)$ ;	SELECT * FROM <table name> <u>WHERE</u> <u>Column1 = 1</u>
<u>Projection</u>	$\pi$ (pi); $\pi_{c_1, c_2, \dots, c_n}(R)$	SELECT <u>Column1, Column 2</u> FROM <table name>
<u>Rename</u>	$\rho$ (rho)	
<u>Union</u>	$\cup$	$A \cup B$ ; $A = \{1, 2, 3, 5\}$ ; $B = \{0, 2\}$ ; $\{1, 2, 3, 5\} \cup \{0, 2\} = \{0, 1, 2, 3, 5\}$
<u>Intersection</u>	$\cap$	$A \cap B$ ; $A = \{1, 2, 3, 5\}$ ; $B = \{0, 2\}$ ; $\{1, 2, 3, 5\} \cap \{0, 2\} = \{2\}$
<u>Difference</u>	$\setminus, -$	$B \setminus A = B - A$ ; $\{0, 2\} - \{1, 2, 3, 5\} = \{0\}$

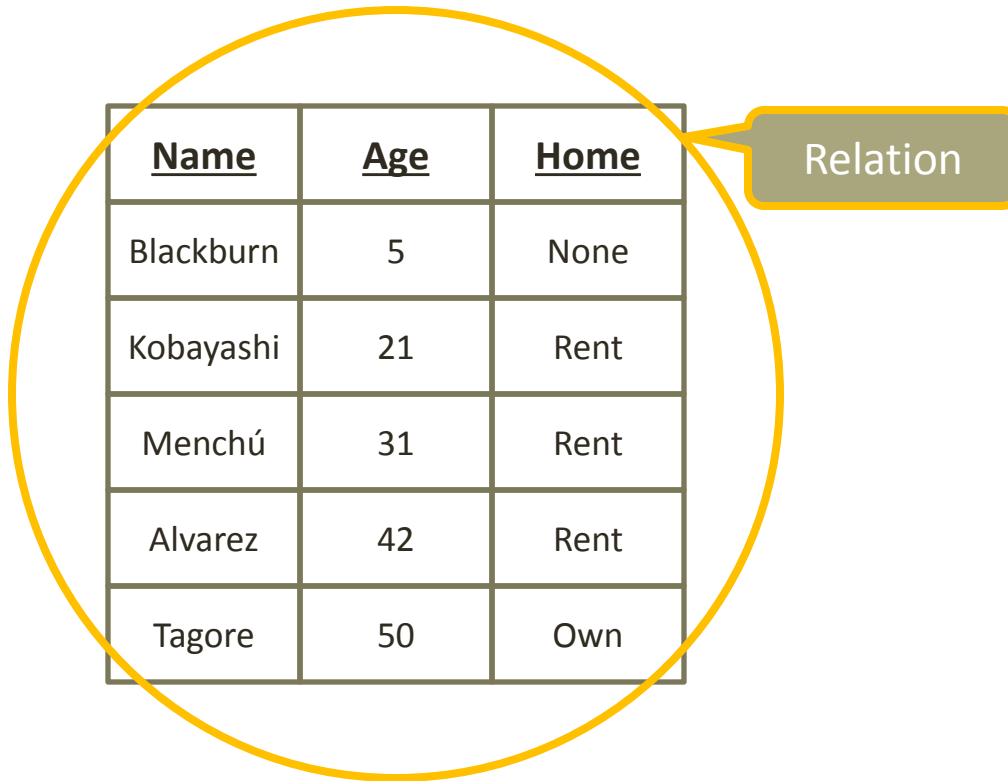
# New Terminology (4)

Operation	Symbols	Comments
<u>Product</u>	$\times$	$A \times B$ $A=\{1,2,3,5\}$ ; $B=\{0,2\}$ ; $\{1,2,3, 5\} \times \{0,2\} = \{\{1,0\}, \{2,0\}, \{3,0\}, \{5,0\}, \{1,2\}, \{2,2\}, \{3,2\}, \{5,2\}\}$
<u>Join</u>	$\bowtie_{\varphi}$	$B \bowtie_{\varphi} A$ ; $\varphi: A > B$ ; $A=\{1,2,3,5\}$ ; $B=\{0,2\}$ ; $\{1,2,3,5\} \bowtie_{\varphi} \{0,2\} = \{\{1,0\}, \{2,0\}, \{3,0\}, \{3,2\}, \{5,0\}, \{5,2\}\}$
<u>Division</u>	$\div$	$A \div B = C$ ; Project to show me the columns in A that are not in B; Select to show me the tuples in A that are a superset of the a tuple in B.

# Relational Algebra

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

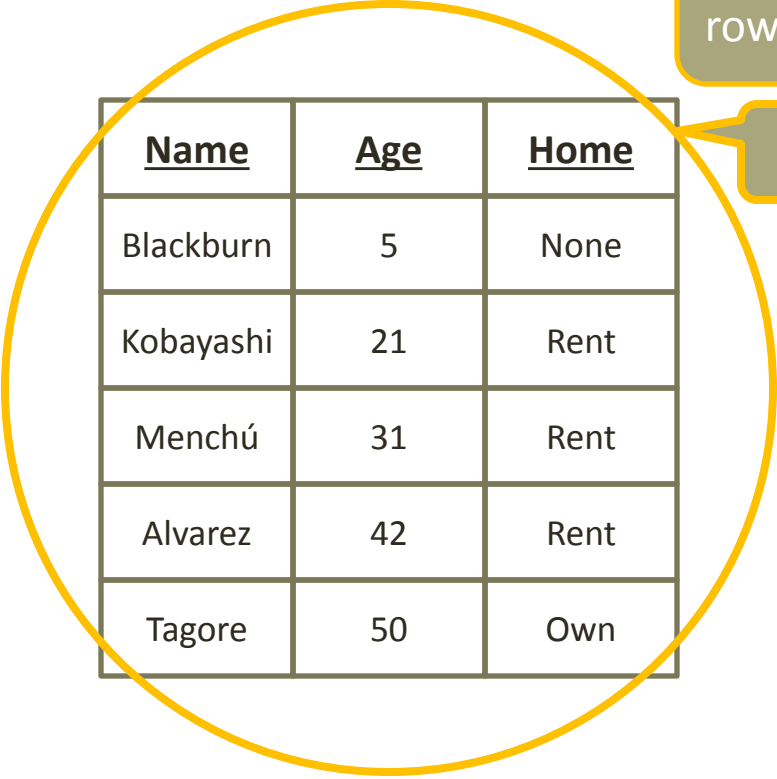
# Relational Algebra: Relation



<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

# Relational Algebra: Relation

Relation is like a table except that each row must be unique like in a set



<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Relation

# Relational Algebra: Attribute

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Attribute

# Relational Algebra: Attribute

## Attribute:


Must be of the same data type.  
Have a name

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Attribute

# Relational Algebra: Tuple

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



tuple



# Relational Algebra: Tuple

**tuple** from: singlele, doublele, triplele,  
quadruple, quinttuple  
**arity** from: unary, binary, ternary

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

tuple with arity of 3

# Relational Algebra: Operands and Simple Operations

- Operand
  - Relation (Table)
- Operations
  - UNION
  - INTERSECT
  - PROJECT
  - SELECT
  - PRODUCT
  - DIVISION

# Relational Algebra: Union

Combine Relations

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

# Relational Algebra: Union

Combine Relations

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Relational Algebra Union:  
 $R \cup S$

# Relational Algebra: Union

Combine Relations

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

SQL Statement:

```
SELECT * FROM MyTableR UNION  
SELECT * FROM MyTableS
```

Relational Algebra Union:  
 $R \cup S$

# Relational Algebra: Union

Combine Relations

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Relational Algebra Union:  
 $R \cup S$

# Relational Algebra: Intersect

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Tagore	50	Own

Same Rows

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

# Relational Algebra: Intersect

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Tagore	50	Own

Same Rows

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



# Relational Algebra: Intersect

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Tagore	50	Own

Same Rows

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Relational Algebra Intersection:  
 $R \cap S$

# Relational Algebra: Intersect

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Tagore	50	Own

Same Rows

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

SQL Statement:

```
SELECT * FROM MyTableR  
INTERSECT  
SELECT * FROM MyTableS
```

Relational Algebra Intersection:  
 $R \cap S$

# Relational Algebra: Intersect

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Tagore	50	Own

Same Rows

<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



<u>Name</u>	<u>Age</u>	<u>Home</u>
Menchú	31	Rent
Tagore	50	Own

Relational Algebra Intersection:  
 $R \cap S$

# Relational Algebra: Examples

- $R \cup S$ 
  - `SELECT * FROM MyTableR UNION SELECT * FROM MyTableS`
- `SELECT * FROM MyTableR UNION SELECT * FROM MyTableS`
  - $R \cup S$  or  $S \cup R$
- $R \cap S$ 
  - `SELECT * FROM MyTableR INTERSECT SELECT * FROM MyTableS`
- `SELECT * FROM MyTableR INTERSECT SELECT * FROM MyTableS`
  - $R \cap S$  or  $S \cap R$
- In General:
  - An operation with  $\cup$  or  $\cap$  produces a relation
  - $R \cup S = S \cup R$
  - $R \cap S = S \cap R$
  - $(R \cup S) \cap T = (R \cap T) \cup (S \cap T)$
  - $(R \cap S) \cup T = (R \cup T) \cap (S \cup T)$

# Relational Algebra: Project

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Vertical partition

# Relational Algebra: Project

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Vertical partition

Relational Algebra Project:

$\pi_{c1, c2, \dots, cn}(R)$

where

$c1, c2, \dots, cn$ : Age, Home

R: MyTable

# Relational Algebra: Project

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

SQL Statement:

```
SELECT Age, Home FROM  
MyTable
```

Vertical partition

Relational Algebra Project:

$$\pi_{c1, c2, \dots, cn}(R)$$


where

$c1, c2, \dots, cn$ : Age, Home

R: MyTable

# Relational Algebra: Project

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



<u>Age</u>	<u>Home</u>
5	None
21	Rent
31	Rent
42	Rent
50	Own

Relational Algebra Project:

$\pi_{c1, c2, \dots, cn}(R)$

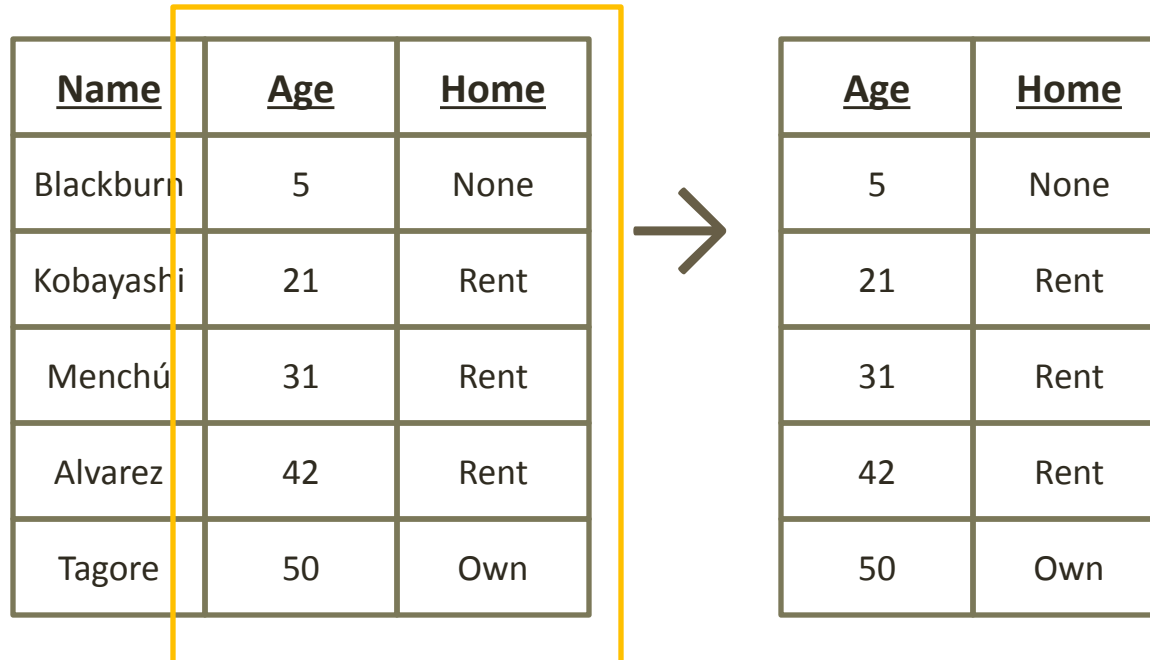
where

$c1, c2, \dots, cn$ : Age, Home

R: MyTable



# Relational Algebra: Project



The result of a projection is a relation with 0 to n attributes where n is the number of attributes in the operand

Relational Algebra Project:

$\pi_{c1, c2, \dots, cn}(R)$   
where

c1, c2, ..., cn: Age, Home

R: MyTable

# Relational Algebra: Select

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Horizontal partition

# Relational Algebra: Select

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Horizontal partition

Relational Algebra Select:

$\sigma_{\varphi}(R)$   
where

$\varphi$ : Home = "Rent"

R: MyTable

# Relational Algebra: Select

SQL Statement:

```
SELECT * FROM MyTable WHERE  
Home = "Rent"
```

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own

Horizontal partition

Relational Algebra Select:


$\sigma_{\varphi}(R)$   
where

$\varphi$ : Home = "Rent"

R: MyTable

# Relational Algebra: Select

<u>Name</u>	<u>Age</u>	<u>Home</u>
Blackburn	5	None
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent
Tagore	50	Own



<u>Name</u>	<u>Age</u>	<u>Home</u>
Kobayashi	21	Rent
Menchú	31	Rent
Alvarez	42	Rent

The result of a selection is a relation with 0 to n tuples where n is the number of tuples in the operand

Relational Algebra Select:

$\sigma_{\varphi}(R)$   
where

$\varphi$ : Home = "Rent"

R: MyTable

# Relational Algebra: Examples

- $\pi_{\text{Age, Home}}(R)$ 
  - SELECT Age, Home FROM MyTable
- $\sigma_{\text{Home}=\text{"Rent"}}(R)$ 
  - SELECT \* FROM MyTable WHERE Home = "Rent"
- SELECT Age, Home FROM MyTable WHERE Home = "Rent"
  - $\pi_{\text{Age, Home}}(\sigma_{\text{Home}=\text{"Rent"}}(R))$  or  $\sigma_{\text{Home}=\text{"Rent"}}(\pi_{\text{Age, Home}}(R))$
- In General:
  - An operation with  $\sigma$  produces a relation
  - An operation with  $\pi$  produces a relation
  - $\sigma_{\varphi_1}(\sigma_{\varphi_2}(R)) = \sigma_{\varphi_2}(\sigma_{\varphi_1}(R))$
  - $\pi_{[c1]}(\pi_{[c2]}(R)) = \pi_{[c2]}(\pi_{[c1]}(R))$
  - $\pi_{[c]}(\sigma_{\varphi}(R)) = \sigma_{\varphi}(\pi_{[c]}(R))$  (**only if**  $\varphi$  is not dependent on  $[c]$ )

# Quiz 05b Relational Algebra

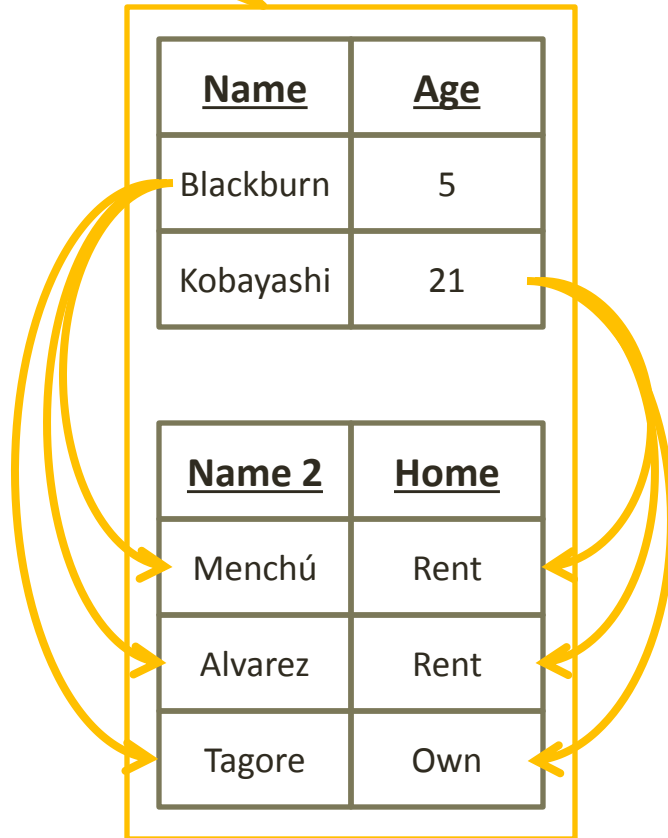
## Intro

- Quiz 06b (Relational Algebra)



# Relational Algebra: Product

Combine Rows

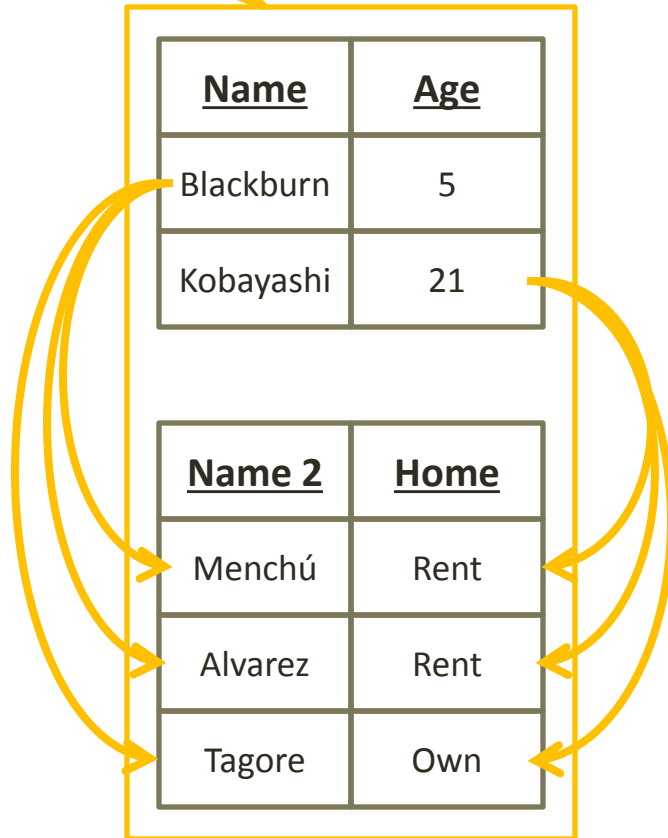


Relational Algebra Product:  
 $R \times S$



# Relational Algebra: Product

Combine Rows



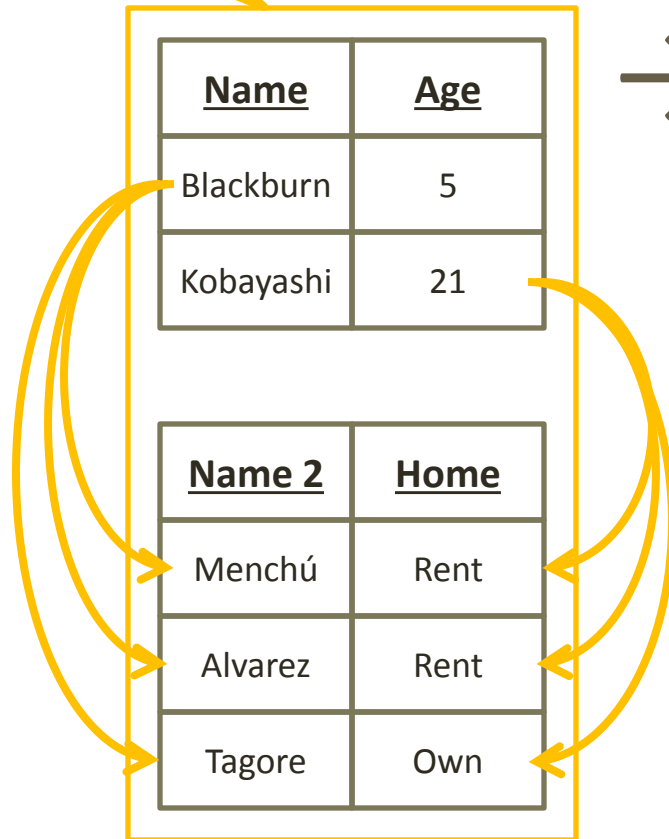
SQL Statement:

```
SELECT * FROM TableR, TableS
```

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
-------------	------------	---------------	-------------

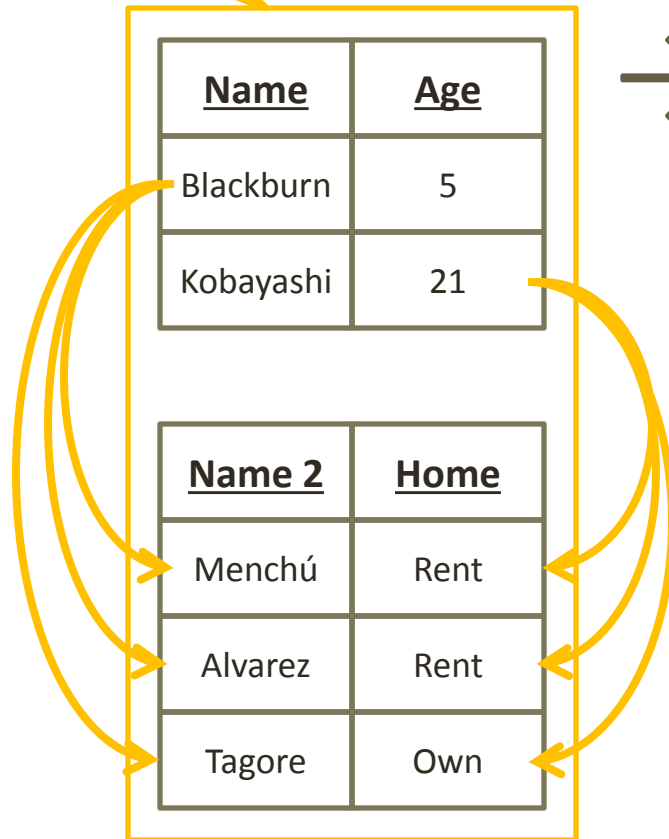
Blackburn	5	Menchú	Rent
		Alvarez	Rent
		Tagore	Own

Kobayashi	21	Menchú	Rent
		Alvarez	Rent
		Tagore	Own

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
-------------	------------	---------------	-------------

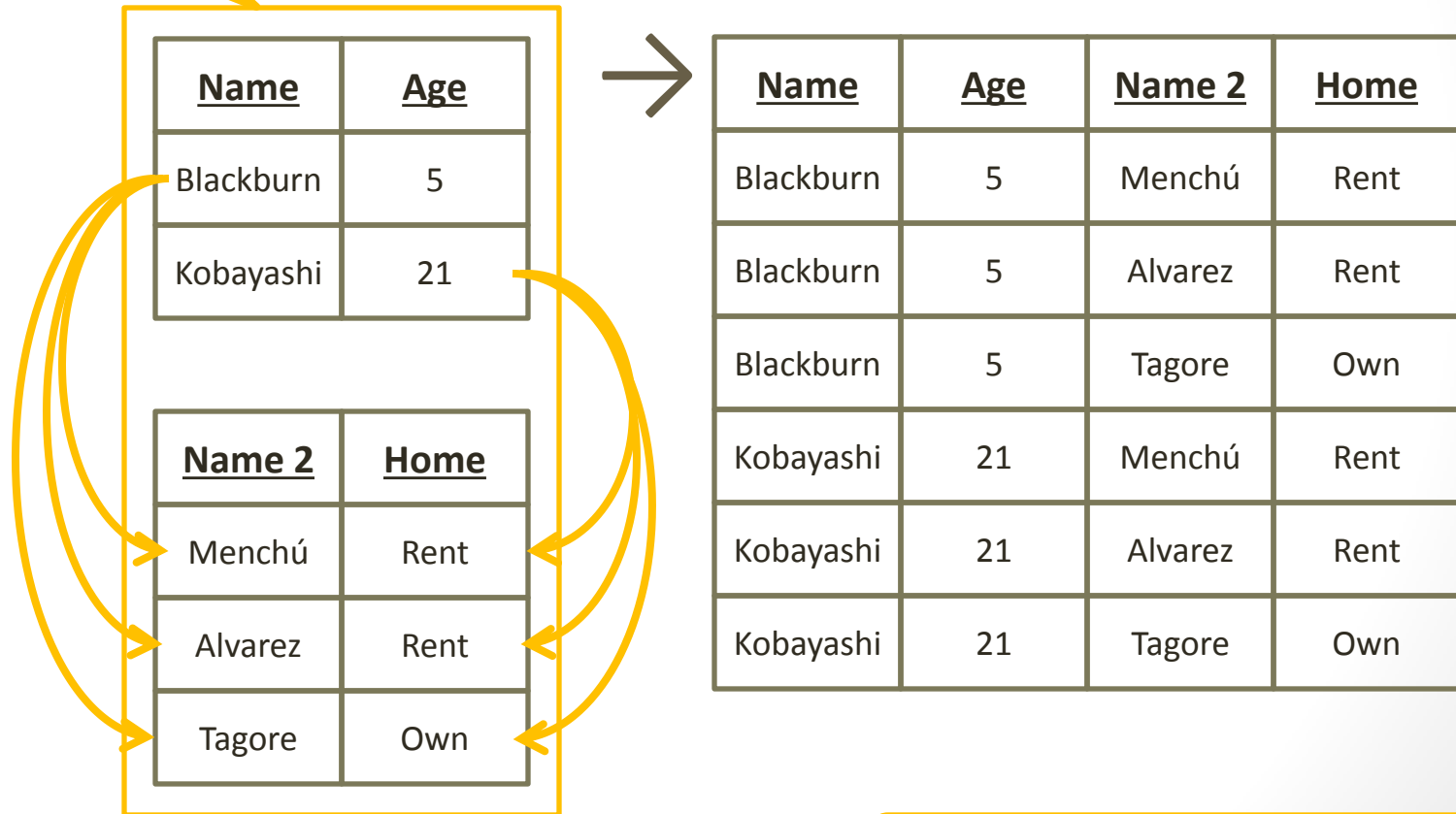
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own

Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

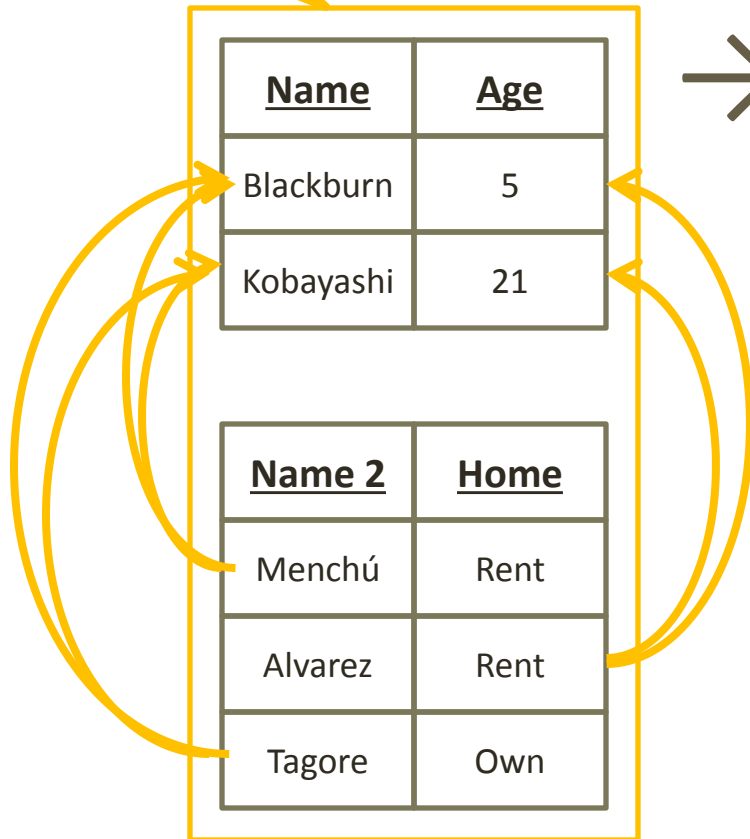
Combine Rows



Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21		
Blackburn	5	Alvarez	Rent
Kobayashi	21		
Blackburn	5	Tagore	Own
Kobayashi	21		

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
-------------	------------	---------------	-------------

Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent

Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent

Blackburn	5	Tagore	Own
Kobayashi	21	Tagore	Own

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent
Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Tagore	Own

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Product

Combine Rows

The result of a product is a relation with  $n \times m$  tuples where  $n$  and  $m$  are the number of tuples in the operands. The arity of the result is  $i + j$  where  $i$  and  $j$  are the arities of the operands

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent
Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Tagore	Own

Relational Algebra Product:  
 $R \times S$



# Relational Algebra: Product

Combine Rows

The result of a product is a relation with  $n*m$  tuples where  $n$  and  $m$  are the number of tuples in the operands. The arity of the result is  $i + j$  where  $i$  and  $j$  are the arities of the operands

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own

Relational Algebra Product:  
 $R \times S$

# Relational Algebra: Join

Combine Rows

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent
Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Tagore	Own

Relational Algebra Product with Select:  
 $\sigma_{\varphi}(R \times S)$  where  $\varphi: \text{Home} = \text{"Rent"}$   
Relational Algebra Join:  
 $R \bowtie_{\varphi} S$  where  $\varphi: \text{Home} = \text{"Rent"}$

# Relational Algebra: Join

Combine Rows

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent
Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Tagore	Own

Relational Algebra Product with Select:  
 $\sigma_{\varphi}(R \times S)$  where  $\varphi: \text{Home} = \text{"Rent"}$   
Relational Algebra Join:  
 $R \bowtie_{\varphi} S$  where  $\varphi: \text{Home} = \text{"Rent"}$

# Relational Algebra: Join

Combine Rows

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



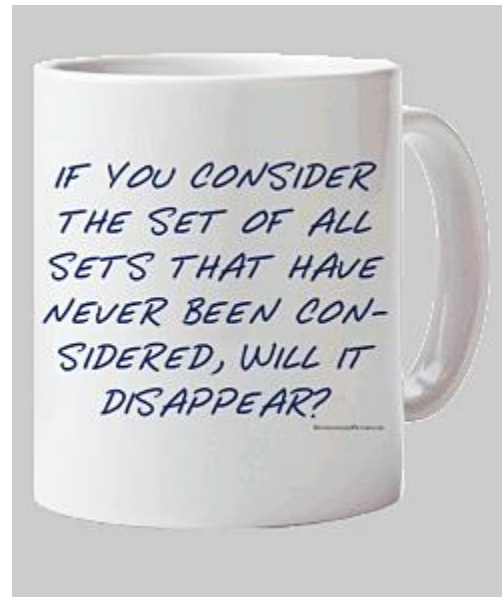
<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Kobayashi	21	Menchú	Rent
Blackburn	5	Alvarez	Rent
Kobayashi	21	Alvarez	Rent

Relational Algebra Product with Select:  
 $\sigma_{\varphi}(R \times S)$  where  $\varphi: \text{Home} = \text{"Rent"}$   
Relational Algebra Join:  
 $R \bowtie_{\varphi} S$  where  $\varphi: \text{Home} = \text{"Rent"}$

# Relational Algebra: Join

- A Join is a Product with a select statement
- Product followed by Select
  - `SELECT * FROM TableR, TableS WHERE Home = "Rent"`
  - $\sigma_{\varphi}(R \times S)$  where  $\varphi: \text{Home} = \text{"Rent"}$
- JOIN
  - `SELECT * FROM TableR JOIN TableS ON Home = "Rent"`
  - $R \bowtie_{\varphi} S$  where  $\varphi: \text{Home} = \text{"Rent"}$

# Break



# Relational Algebra: Division

This was a Product  
Operand

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own

This was a Product Operand

This was the result  
of a Product

<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own

Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

A Division is sort of like the reverse of a Product

This was a Product  
Operand

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21

This was the result  
of a Product

<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own

<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own

This was a Product  
Operand

Relational Algebra Division:  
 $R \div S$



# Relational Algebra: Division

A Division is sort of like the reverse of a Product

This was a Product  
Operand

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own

This was a Product Operand

This was the result  
of a Product

<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own



Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own
Sancar	54	Tagore	Own

Add another row to this table that did not result from the product.

Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

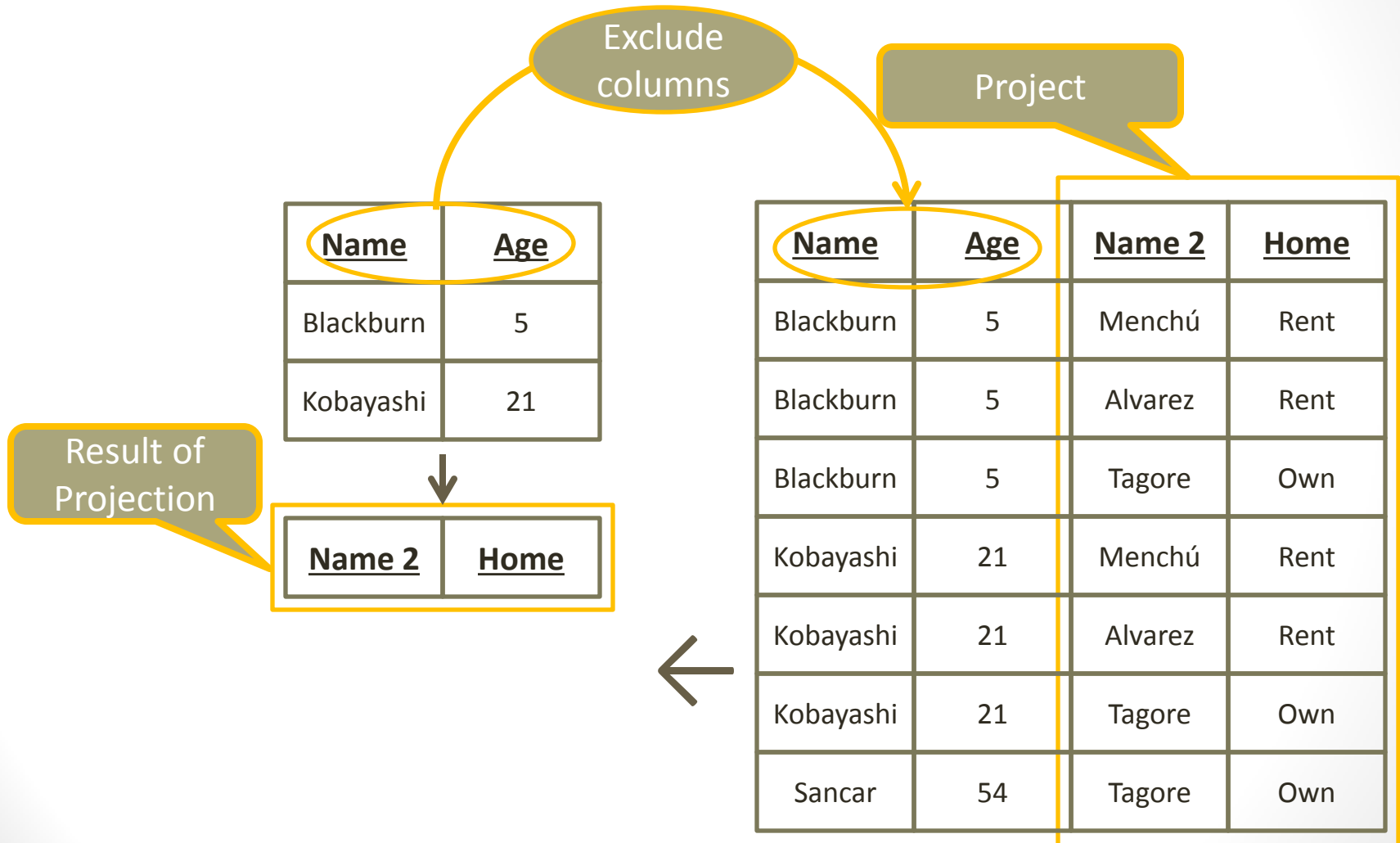
<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own
Sancar	54	Tagore	Own

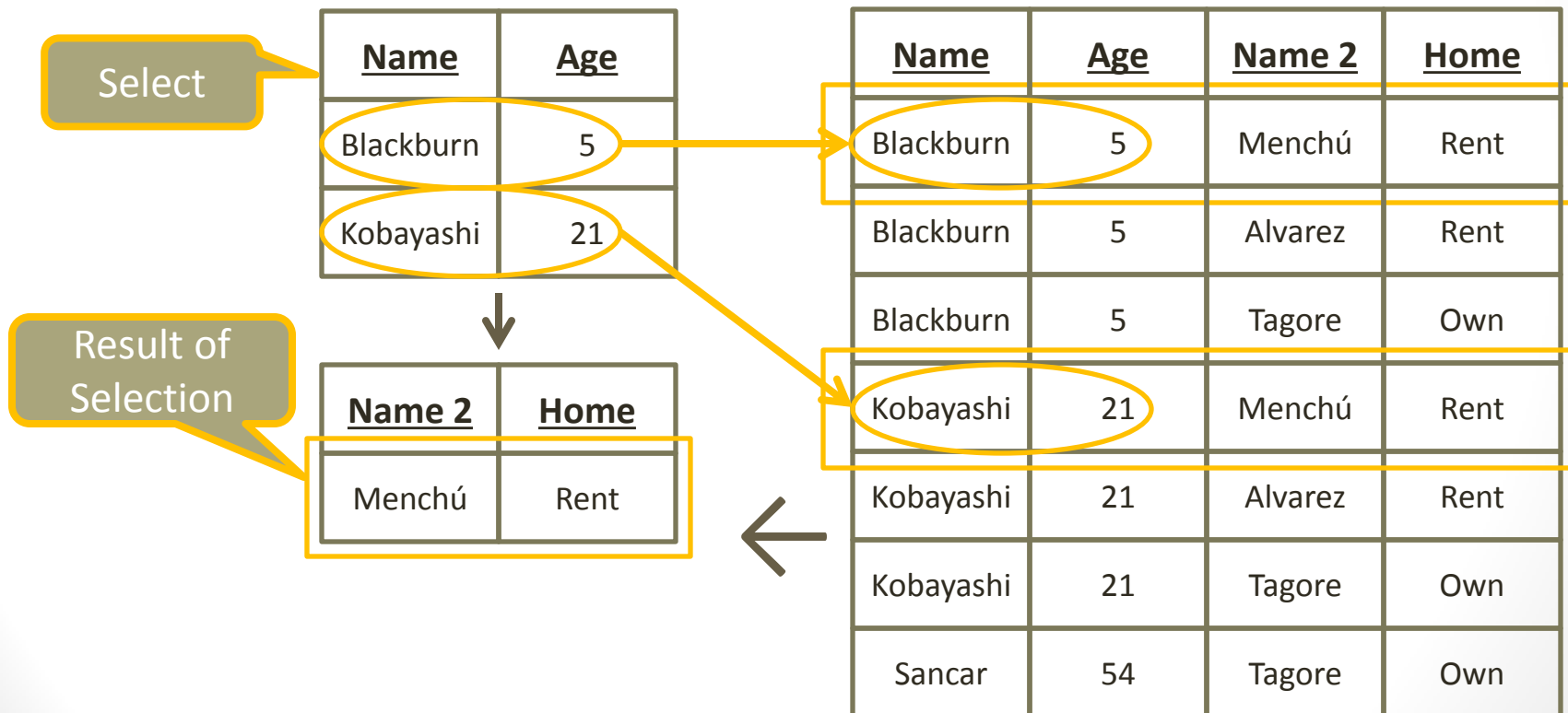
Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division



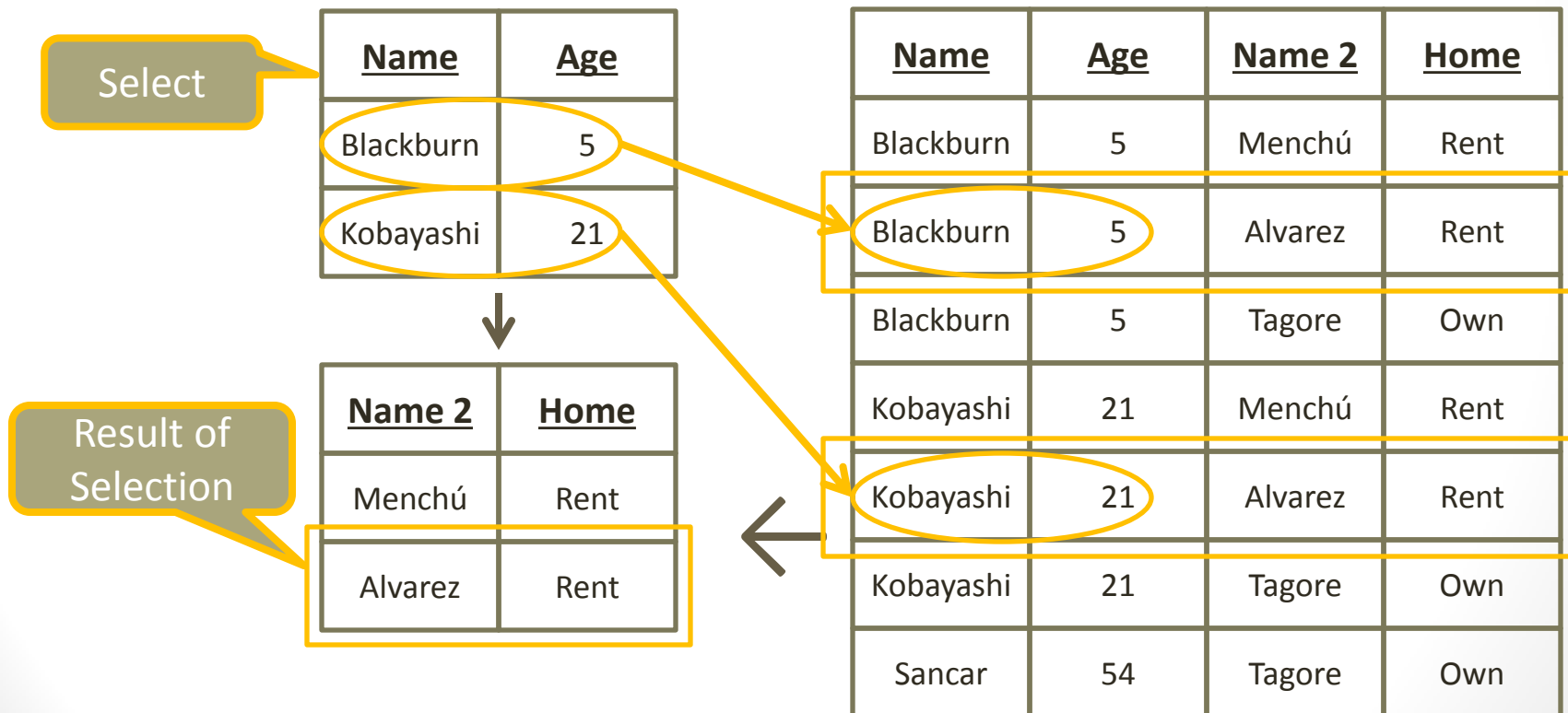
Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division



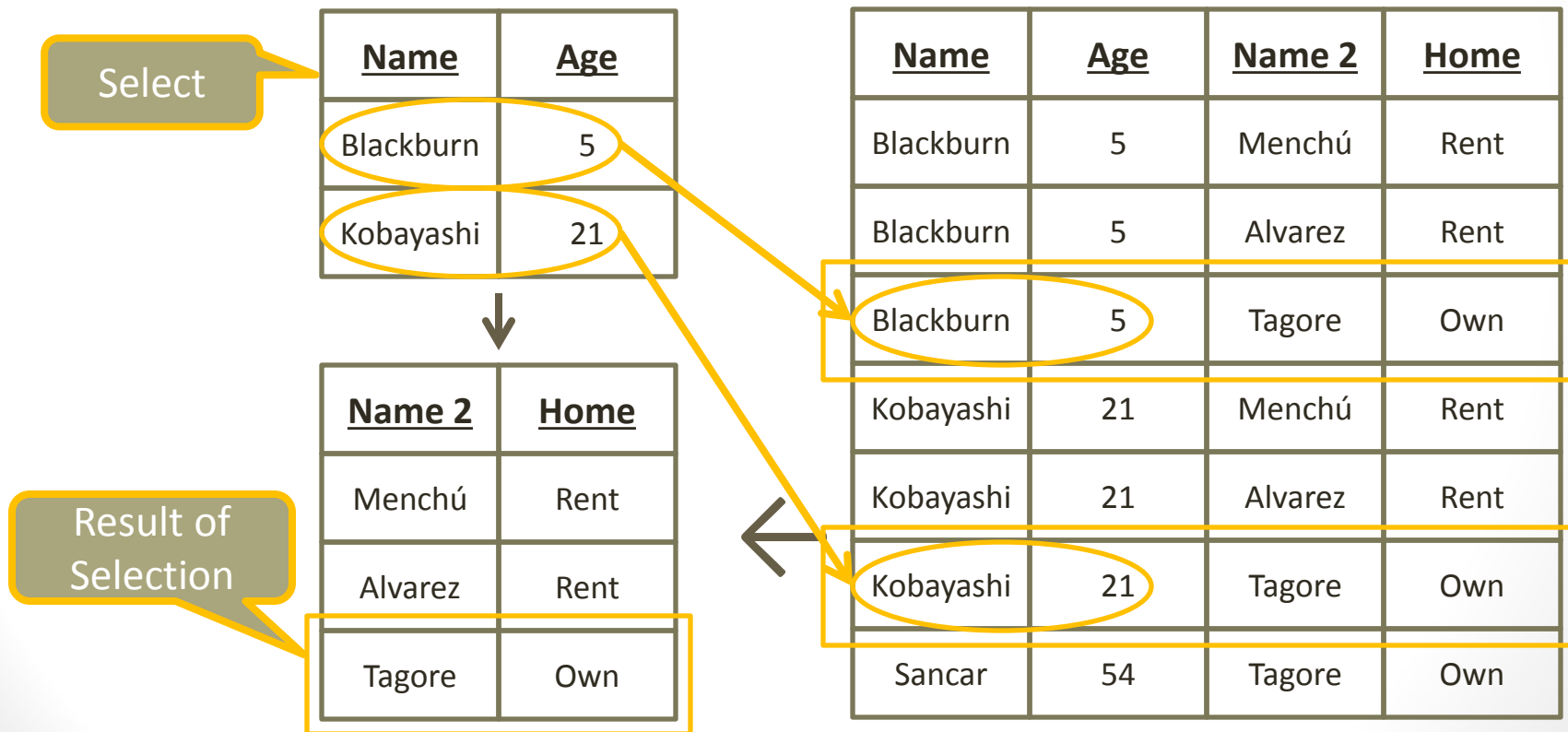
Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division



Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division



Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

[Menchú, Rent] is in the same tuple as  
[Blackburn, 5] and [Kobayashi, 21]

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own
Sancar	54	Tagore	Own

Relational Algebra Division:  
 $R \div S$



# Relational Algebra: Division

[Alvarez, Rent] is in the same tuple as  
[Blackburn, 5] and [Kobayashi, 21]

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own



<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own
Sancar	54	Tagore	Own

Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

[Tagore, Own] is in the same tuple as  
[Blackburn, 5] and [Kobayashi, 21]

<u>Name</u>	<u>Age</u>
Blackburn	5
Kobayashi	21



<u>Name 2</u>	<u>Home</u>
Menchú	Rent
Alvarez	Rent
Tagore	Own

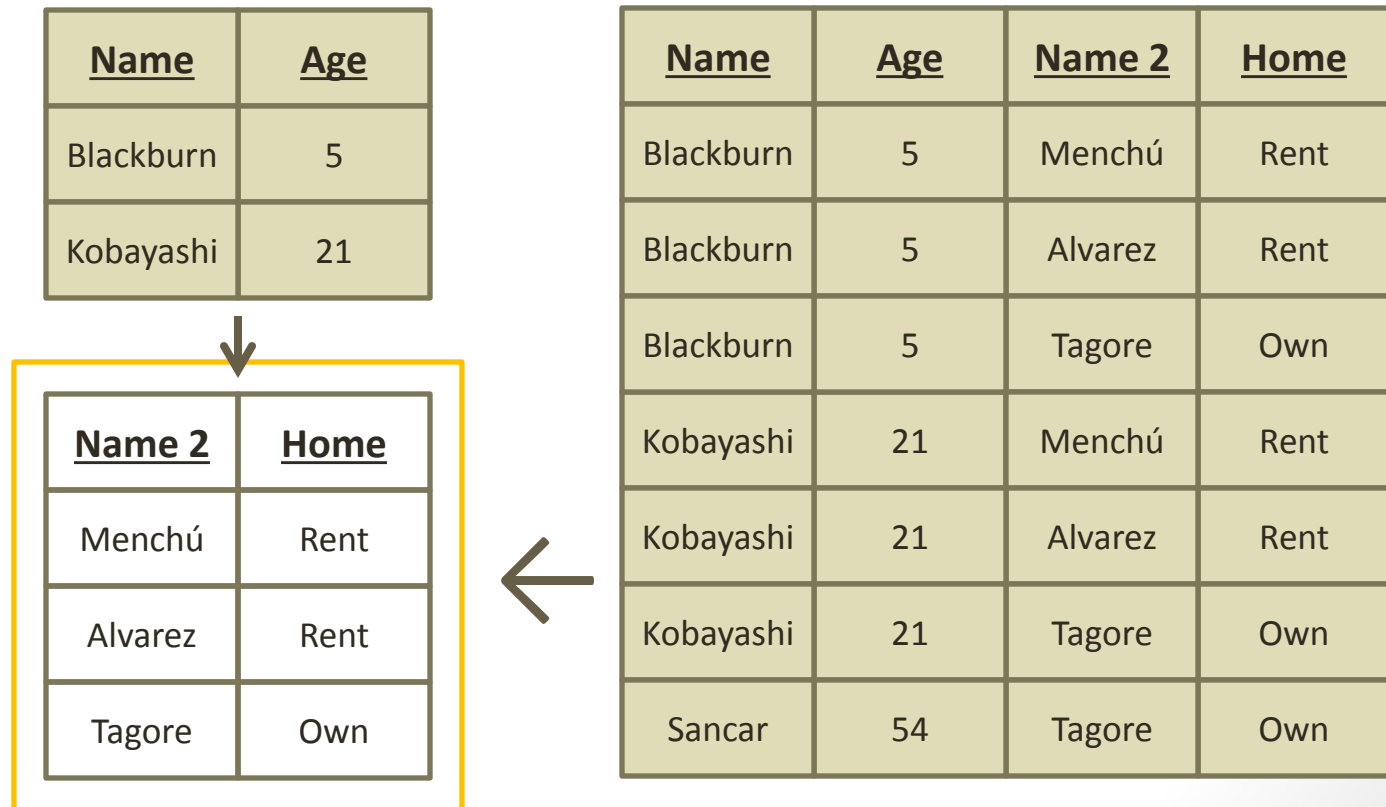


<u>Name</u>	<u>Age</u>	<u>Name 2</u>	<u>Home</u>
Blackburn	5	Menchú	Rent
Blackburn	5	Alvarez	Rent
Blackburn	5	Tagore	Own
Kobayashi	21	Menchú	Rent
Kobayashi	21	Alvarez	Rent
Kobayashi	21	Tagore	Own
Sancar	54	Tagore	Own

Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Division

The result of a division is a relation with  $n$  tuples of arity  $l$  where the dividend operand has at least  $n \cdot m$  tuples of arity  $i + j$  and the divisor operand has exactly  $m$  tuples of arity  $j$  that are a subset of the of the dividend tuples.



Relational Algebra Division:  
 $R \div S$

# Relational Algebra: Resources

- Examples for Relational Algebra and SQL
  - RelationalAlgebraAndSQL.pdf
  - RelationalAlgebraAndSQL.sql
- Links for definitions and concepts:
  - [http://en.wikipedia.org/wiki/Cartesian\\_product](http://en.wikipedia.org/wiki/Cartesian_product)
  - [http://en.wikipedia.org/wiki/Commutative\\_property](http://en.wikipedia.org/wiki/Commutative_property)
  - [http://en.wikipedia.org/wiki/Associative\\_property](http://en.wikipedia.org/wiki/Associative_property)
  - [http://en.wikipedia.org/wiki/Closure\\_\(mathematics\)](http://en.wikipedia.org/wiki/Closure_(mathematics))
  - [http://en.wikipedia.org/wiki/Relational\\_calculus](http://en.wikipedia.org/wiki/Relational_calculus)
  - [http://en.wikipedia.org/wiki/Relational\\_algebra](http://en.wikipedia.org/wiki/Relational_algebra)
  - [http://en.wikipedia.org/wiki/Edgar\\_F.\\_Codd](http://en.wikipedia.org/wiki/Edgar_F._Codd)
  - [http://en.wikipedia.org/wiki/Relational\\_model](http://en.wikipedia.org/wiki/Relational_model)
  - [http://en.wikipedia.org/wiki/Relational\\_database](http://en.wikipedia.org/wiki/Relational_database)
  - [http://en.wikipedia.org/wiki/Query\\_language](http://en.wikipedia.org/wiki/Query_language)
  - <http://en.wikipedia.org/wiki/SQL>
  - <http://en.wikipedia.org/wiki/NoSQL>

# Quiz 05c Product Join Division

- Quiz 06c (Product, Join, Division)



# Relational Algebra

The Theory behind Relational Databases

# Midpoint Perspective

# Midpoint Perspective

- The class has done excellent work. I haven't commented enough on your excellent work and obvious talent.
- Philosophy of instruction: The point of a class like ours, as opposed to a MOOC, is the personal feedback and dedicated community.
  - The feedback from the instructor and students are the most valuable learning tool.
  - The community is special, because you know everyone and everyone has committed to participate in the group.



# Midpoint Perspective

- Some topics we covered (Focus on Analysis)
  - Data Preparation for Data Science
  - Introduction to R
  - Predictive Analytics:
    - Unsupervised learning: K-Means
    - Supervised learning: Classification
  - Classification Statistics
  - Data Structures
  - Relational Algebra
- Some topics we will cover (Focus on Persistence)
  - NoSQL (Scale out and CAP)
  - RDBMS
  - Graph Data
  - SPARQL
  - EAV and Sparse Matrices
  - Hadoop (HDFS and MapReduce)
  - Spark

# Midpoint Perspective

# New Terminology

- Hadoop
- Master Node
- Data Node
- Cluster
- Hive
- Impala
- MapReduce
- HDFS
- Doug Cutting
- Scalability
- AWS
- Elastic Cloud
- NoSQL
- CAP Theorem
- Consistency (CAP)
- Availability (CAP)
- Partition Tolerance (CAP)
- Eric Brewer
- RDBMS
- ACID
- Atomic (ACID)
- Consistent (ACID)
- Isolation (ACID)
- Durability (ACID)
- BASE
- Eventual Consistency
- Paxos
- Sqoop
- CouchDB
- Shared Data
- Stale Data
- Scale-out
- Scale-up
- Grace Hopper
- Data Replication
- Horizontal Partitioning
- Vertical Partitioning
- Heartbeats
- Multi-Version Concurrency Control
- EAV
- Relational Algebra
- Relational Calculus
- Relational Model
- Ted Codd
- Codd's Theorem
- Transaction Shell
- Column-oriented DBMS
- Row-oriented
- SPARQL

# Assignment (1)

1.  $\{a, b, c\}$  is a relation that contains the tuples  $a$ ,  $b$ , and  $c$ . In the following cases the tuples have arity of 1. Calculate the following:
  - a.  $(\{1, 2, 3\} \cup \{5, 7, 11\}) \cap \{2, 4, 6, 8, 10\}$
  - b.  $(\{1, 2, 3\} \cap \{2, 4, 6, 8, 10\}) \cup (\{5, 7, 11\} \cap \{2, 4, 6, 8, 10\})$
2. A relation exists with 4 columns, named Column1, Column2, Column3, and Column4. Column1 is of type text. Column2, Column3, and Column4 are of type int:
  - a. Use relational algebra to fulfill the intent of the following SQL.
    - **SELECT Column1, Column3 FROM MyTable WHERE Column2 = Column3**
  - b. Reverse the order of projection and selection in your algebraic formulation from item 2a. What is the result of the new algebraic expression?
3.  $\pi_{c1, c2}(\sigma_{\varphi1}(\sigma_{\varphi2}(\pi_{c1, c2, c3, c5}(R))))$   
Where
  - $\varphi1: C1 = C5;$
  - $\varphi2: C5 = \text{"Test"};$
  - $R: \text{MyTable};$
  - a. Write a SQL statement that declares the intent of the algebraic notation
  - b. Simplify the algebraic statement. Simplification means minimize the number of parentheses and terms.

# Assignment (2)

4. `SELECT * FROM T1 JOIN T2 ON T1.C1 = T2.C1`
  - a. Write out an equivalent in relational algebra using the join operator
  - b. Write out an equivalent in relational algebra without using the join operator
5.  $\pi_{S.C1, R.C2}(\sigma_{\varphi_1}(R) \bowtie_{\varphi_2} S)$   
where
  - $\varphi_1 = (R.C2 = 'A')$
  - $\varphi_2 = (R.C1 = S.C2)$
  - Write out equivalent SQL and test this SQL using relations R and S that you create for this example. The relations R and S in RelationalAlgebraAndSQL.pdf and RelationalAlgebraAndSQL.sql don't quite work because their column types do not match for this assignment.
6. Write a function in R that uses sqldf package to create a full outer join. The function is called OuterJoin and the function declaration is: ***OuterJoin <- function(tableA, tableB, keyA, keyB)***. tableA and tableB are data frames that will be joined. keyA and keyB are of type character and contain the names of the column that you join on. The function returns a data frame that is the outer join of the two input data frames. The SQL statement in sqldf is available on the web. In order to point out the difference between tables and sets, do not use the UNION keyword from sqldf, instead use rbind from {base}. rbind, by itself, doesn't do exactly what UNION does and you may need more code.

# Assignment (3)

7. Install and setup the Hadoop VM according to SetupVirtualMachine.pdf. Make sure that you get results similar to those shown in SetupVirtualMachine.pdf. If you completed this item last week, then you do not need to do it again. Otherwise, submit the requested screenshot to Canvas.
8. Submit answers to items 1 through 6 in a txt, doc/docx, or sql file. I will need to copy and paste the SQL statements. Submit the screen shot from item 7, if you didn't do it last week. Submission due date is Saturday 11:57 PM.
9. Discuss a topic in the class LinkedIn group. The topic should be related to something here:
  - The preview section in the class overview including the quiz preview.
  - The New Terminology slide in this deck.
  - Read: Google file system:  
<http://static.googleusercontent.com/media/research.google.com/en/us/archive/gfs-sosp2003.pdf>
  - Read MapReduce:  
<http://static.googleusercontent.com/media/research.google.com/en/us/archive/mapreduce-osdi04.pdf>

# Introduction to Data Science