5A – INTRODUCTION TO THE RELATIONAL MODEL AND SQL

CS 1656 Introduction to Data Science

Alexandros Labrinidis – http://labrinidis.cs.pitt.edu
University of Pittsburgh

RELATIONAL MODEL

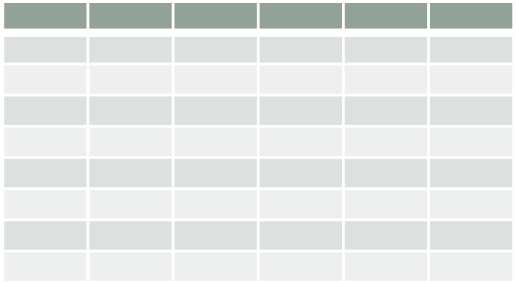
The Relational Model

- It is the most popular data model
  - Simplest, most uniform data structures
  - Most formal (algebra to describe operations)

- Introduced in **1970** (by E. F. Codd – Turing Award/1981)
  - Before: records, pointers, sets, etc
  - Hierarchical Data Model (IBM IMS, 1966-68)
  - Network Data Model (CODASYL DBTG, 1969)
- Everything from real world is represented by **relations**
  - i.e., tables
- Each table has multiple rows and columns
  - Row in a table "binds" values together (row = tuple)

Data stored in tables

- Everything is stored in tables under the relational model!



=

Let's store some data: List of products

- **Product Number = 557**
  - **Name = Fleece Pullover (in English)**
  - **Colors = navy, black**
  - **Department = Women's**
- **Product Number = 563**
  - **Name = Floppy Sun Hat (in English)**
  - **Department = Accessories**
- **Product Number = 443**
  - **Name = Deluxe Travel Bag (in English)**
  - **Department = Accessories**
- **Product Number = 784**
  - **Name = Cotton Dress Shirt (in English)**
  - **Colors = white, gray**
  - **Department = Men's**
  - **Description: Our favorite shirt!**

Store as XML document (catalog.xml)

```
<catalog><product dept="WMN">
                <number>557</number>
                <name language="en">Fleece Pullover</name>
                <colorChoices>navy black</colorChoices>
        </product>
        <product dept="ACC">
                <number>563</number>
                <name language="en">Floppy Sun Hat</name>
        </product>
        <product dept="ACC">
                <number>443</number>
                <name language="en">Deluxe Travel Bag</name>
        </product>
        <product dept="MEN">
                <number>784</number>
                <name language="en">Cotton Dress Shirt</name>
                <colorChoices>white gray</colorChoices>
                <desc>Our <i>favorite</i> shirt!</desc>
        </product>
</catalog>
```

Store as JSON object

```
{ "products":
        [ {"number": 557, "name": "Fleece Pullover", "language":"en",
          "colors": "navy, black", "dept": "WMN"},

          {"number": 563, "name": "Floppy Sun Hat", "language":"en",
          "dept": "ACC"},

          {"number": 443, "name": "Deluxe Travel Bag", "language":"en",
          "dept": "ACC"},

          {"number": 784, "name": "Cotton Dress Shirt", "language":"en",
          "colors": "white, gray", "dept": "MEN",
          "desc": "Our <i>favorite</i> shirt!"},
        ]
}
```

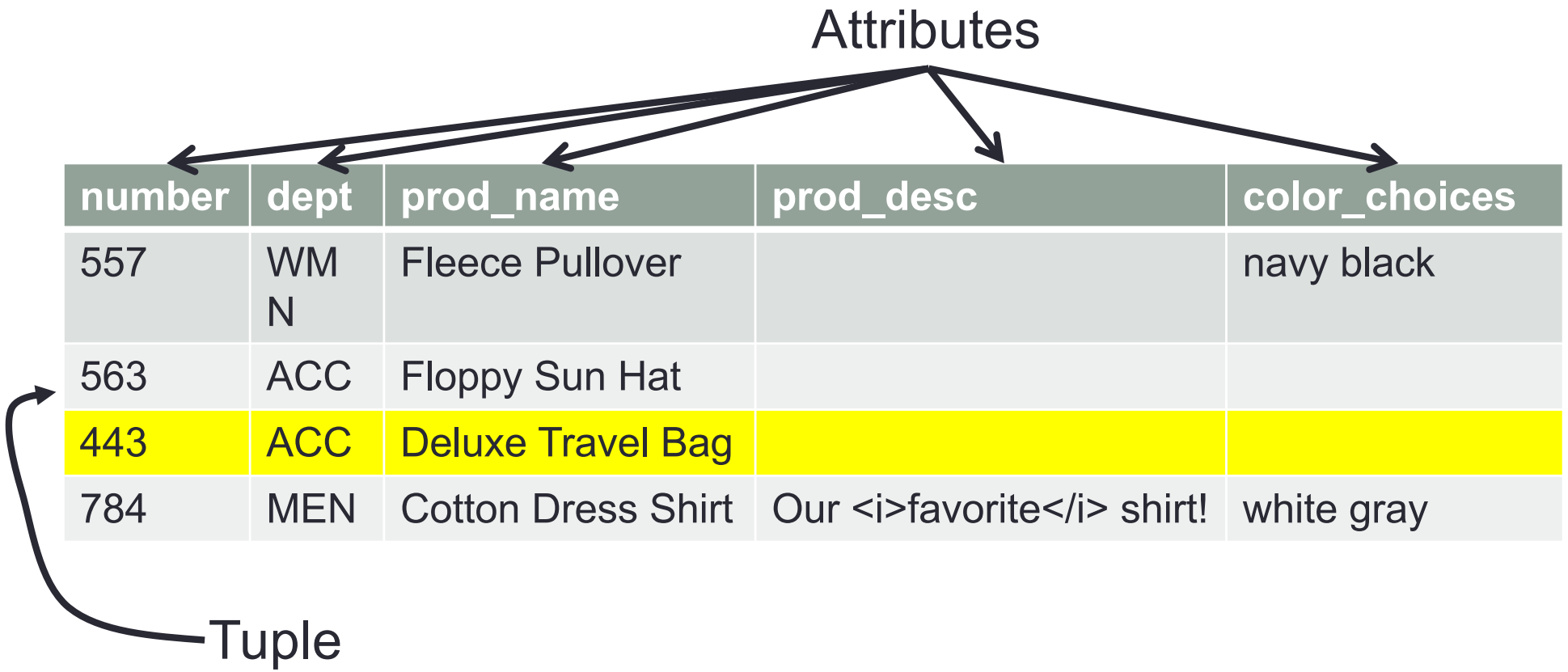
Store as JSON object – take 2

```
{ "products":
        [ {"number": 557, "name": "Fleece Pullover", "language":"en",
          "colors": ["navy", "black"], "dept": "WMN"},

          {"number": 563, "name": "Floppy Sun Hat", "language":"en",
          "dept": "ACC"},

          {"number": 443, "name": "Deluxe Travel Bag", "language":"en",
          "dept": "ACC"},

          {"number": 784, "name": "Cotton Dress Shirt", "language":"en",
          "colors": ["white","grey"], "dept": "MEN",
          "desc": "Our <i>favorite</i> shirt!"},
        ]
}
```

Sample relation: Catalog

number	dept	prod_name	prod_desc	color_choices
557	WMN	Fleece Pullover		navy black
563	ACC	Floppy Sun Hat		
443	ACC	Deluxe Travel Bag		
784	MEN	Cotton Dress Shirt	Our favorite shirt!	white gray

Sample relation: Catalog



number	dept	prod_name	prod_desc	color_choices
557	WMN	Fleece Pullover		navy black
563	ACC	Floppy Sun Hat		
443	ACC	Deluxe Travel Bag		
784	MEN	Cotton Dress Shirt	Our favorite shirt!	white gray

Sample XML document: order.xml

```
<order num="00299432" date="2006-09-15" cust="0221A">
        <item dept="WMN" num="557" quantity="1" color="navy"/>
        <item dept="ACC" num="563" quantity="1"/>
        <item dept="ACC" num="443" quantity="2"/>
        <item dept="MEN" num="784" quantity="1" color="white"/>
        <item dept="MEN" num="784" quantity="1" color="gray"/>
        <item dept="WMN" num="557" quantity="1" color="black"/>
</order>
```

Sample relation: Orders

OrderNum	Date	Cust	Dept	ProdNum	Quant	Color
00299432	2006-09-15	0221A	WMN	557	1	navy
00299432	2006-09-15	0221A	ACC	563	1	
00299432	2006-09-15	0221A	ACC	443	2	
00299432	2006-09-15	0221A	MEN	784	1	white
00299432	2006-09-15	0221A	MEN	784	1	gray
00299432	2006-09-15	0221A	WMN	557	1	black

Q: Something is off with this table. What?

A: Too much redundancy!

Sample relation: Orders, OrderItems

OrderNum	Date	Cust
00299432	2006-09-15	0221A

Orders

OrderNum	Dept	ProdNum	Quant	Color
00299432	WMN	557	1	navy
00299432	ACC	563	1	
00299432	ACC	443	2	
00299432	MEN	784	1	white
00299432	MEN	784	1	gray
00299432	WMN	557	1	black

OrderItems

Sample XML document: prices.xml

```
<prices>
        <priceList effDate="2006-11-15">
        <prod num="557">
                <price currency="USD">29.99</price>
                <discount type="CLR">10.00</discount>
        </prod>
        <prod num="563">
                <price currency="USD">69.99</price>
        </prod>
        <prod num="443">
                <price currency="USD">39.99</price>
                <discount type="CLR">3.99</discount>
        </prod>
        </priceList>
</prices>
```

Sample relation: Prices

Eff_Date	Prod_Num	Currency	Price	Disc_Type	Discount
2006-11-15	557	USD	29.99	CLR	10.00
2006-11-15	563	USD	69.99		
2006-11-15	443	USD	39.99	CLR	3.99

Sample relation: Catalog (revisited)

number	dept	prod_name	prod_desc	color_choices
557	WMN	Fleece Pullover		navy black
563	ACC	Floppy Sun Hat		
443	ACC	Deluxe Travel Bag		
784	MEN	Cotton Dress Shirt	Our favorite shirt!	white gray

Under relational model can store only “atoms” in a cell

➔ cannot do queries for part of a cell

➔ must break up color_choices

Sample relation: Catalog (option 1)

number	dept	prod_name	prod_desc	choice1	choice2
557	WMN	Fleece Pullover		navy	black
563	ACC	Floppy Sun Hat			
443	ACC	Deluxe Travel Bag			
784	MEN	Cotton Dress Shirt	Our favorite shirt!	white	gray

Sample relation: Catalog (option 2)

number	dept	prod_name	prod_desc
557	WMN	Fleece Pullover	
563	ACC	Floppy Sun Hat	
443	ACC	Deluxe Travel Bag	
784	MEN	Cotton Dress Shirt	Our favorite shirt!

Catalog

number	color_choice
557	navy
557	black
784	white
784	gray

ColorChoices

SQL

Structured Query Language

About SQL

- SQL stands for Structured Query Language
- ANSI Standard on 1986, ISO Standard on 1987

[Source: http://en.wikipedia.org/wiki/SQL]

- **Query Language**:
  - Allows manipulation and retrieval of data from a database

- **Declarative**:
  - Specify **WHAT** is to be retrieved
    - E.g., show me all products from “ACC” department
  - **HOW** to retrieve it is responsibility of system
    - This allows for optimizations!

General form of SQL statements

select attribute1, attribute2, attribute3, …

from table1 [, table2, …]

where condition1 and/or condition2 …

Select

- Select is used to specify which columns to include as part of the results
  - i.e., acts as a filter on the columns
- Example:

Select Dept, ProdNum
From OrderItems

OrderNum	Dept	ProdNum	Quant	Color
00299432	WMN	557	1	navy
00299432	ACC	563	1	
00299432	ACC	443	2	
00299432	MEN	784	1	white
00299432	MEN	784	1	gray
00299432	WMN	557	1	black

Select (results)

- Select is used to specify which columns to include as part of the results
  - i.e., acts as a filter on the columns
- Example:

Select Dept, ProdNum
From OrderItems

Dept	ProdNum
WMN	557
ACC	563
ACC	443
MEN	784
MEN	784
WMN	557

- Number of columns is called **Arity**

Where

- Where is used to specify which rows to include as part of the results
  - i.e., returned tuples must satisfy the predicate

- Example:

Select *
From OrderItems
Where Dept = “ACC”

OrderNum	Dept	ProdNum	Quant	Color
00299432	WMN	557	1	navy
00299432	ACC	563	1	
00299432	ACC	443	2	
00299432	MEN	784	1	white
00299432	MEN	784	1	gray
00299432	WMN	557	1	black

Where (results)

- Where is used to specify which rows to include as part of the results
  - i.e., returned tuples must satisfy the predicate
- Example:

OrderNum	Dept	ProdNum	Quant	Color
00299432	ACC	563	1	
00299432	ACC	443	2	

Select *

From OrderItems

Where Dept = “ACC”

- Asterisk (*) in select means all columns
- Number of rows is called cardinality

What about combinations?

Query:

Select Dept, ProdNum

From OrderItems

Where Dept = “ACC”

OrderNum	Dept	ProdNum	Quant	Color
00299432	WMN	557	1	navy
00299432	ACC	563	1	
00299432	ACC	443	2	
00299432	MEN	784	1	white
00299432	MEN	784	1	gray
00299432	WMN	557	1	black

What about combinations? (results)

Query:

Select Dept, ProdNum

From OrderItems

Where Dept = “ACC”

Dept	ProdNum
ACC	563
ACC	443

Everything is a relation (revisited)

The elegance of the relational model:
when performing an operation, the results of the operation can be an input to another operation

In other words: Operations can be trivially pipelined

Everything is a relation (revisited)

```
Select  Dept, ProdNum
From    OrderItems
Where   Dept = "ACC"
```

=

```
Select Dept, ProdNum
From
    Select  *
    From    OrderItems
    Where Dept = "ACC"
```

```
select Dept, ProdNum
from (select * from OrderItems where Dept="ACC")
```

```
Select  Dept, ProdNum
From    OrderItems
Where   Dept = "ACC"
```

=

```
Select *
From
    Select  Dept, ProdNum
    From    OrderItems
Where Dept = "ACC"
```

```
select *
from (select Dept, ProdNum from OrderItems)
where Dept="ACC"
```

Understanding Question

(Q1) How many rows will the output of the following query have (i.e., what is its cardinality)?

```
select prod_name, dept
from Catalog
where number > 500
```

(Q2) How many columns will the output of the following query have (i.e., what is its arity)?

```
select prod_name, dept
from Catalog
where number > 500
```

(Q3) How many columns will the output of the following query have (i.e., what is its arity)?

```
select prod_name, dept
from Catalog
where number > 1000
```

JOINS

Or, how to combine information from different tables

Cartesian Product

```
select *
from relationA, relationB
```

- Columns of result:
  - All columns of relationA plus all columns of relationB
- Rows of results:
  - Combination of all tuples from relationA with all tuples of relationB

- Cardinality of result = cardinality(A) * cardinality(B)
- Arity of result = arity(A) + arity(B)

Cartesian Product Example

People

Name	Number
Alice	4231
Bob	56
Chris	363

Offices

Name	Building	ROOM
Alice	SENSQ	5432
Bob	CL	32123
Chris	BENDM	105

```
select * from people, offices
```

Cartesian Product Example - Results

P.name	P.number	O.name	O.Building	O.ROOM
Alice	4231	Alice	SENSQ	5432
Alice	4231	Bob	CL	32123
Alice	4231	Chris	BENDM	105
Bob	56	Alice	SENSQ	5432
Bob	56	Bob	CL	32123
Bob	56	Chris	BENDM	105
Chris	363	Alice	SENSQ	5432
Chris	363	Bob	CL	32123
Chris	363	Chris	BENDM	105

select * from people, offices

However…

- It makes more sense to only combine RELATED tuples

- **In other words**:
  - for tables that share a common attribute
  - for cases where the values for the common attribute are the same

Cartesian Product No More

P.name	P.number	O.name	O.building	O.ROOM
Alice	4231	Alice	SENSQ	5432
Bob	56	Bob	CL	32123
Chris	363	Chris	BENDM	105

```
select *
from people, offices
where people.name = offices.name
```

This is called a JOIN

Join

- Different flavors of joins exist

- Most important, different variants:
  - **Equijoin**
  - Natural join
  - Outerjoin

Equijoin

```
select *
from people, offices
where people.name = offices.name
```

In general:

- Predicate must have equality condition
- Check condition before combine tuples
- All attributes are retained, from both relations

Understanding Question

(Q4) How many rows will the output of the following query have (i.e., what is its cardinality)?

```
select *
from Catalog, Prices
```

(Q5) How many columns will the output of the following query have (i.e., what is its arity)?

```
select *
from Catalog, Prices
```

ABOUT KEYS

Keys

- **Superkey**
  - Set of one or more attributes that, taken collectively, uniquely identify a tuple within the relation
    - E.g., {firstName, lastName, zipcode}, {firstName, lastName}, {ssn}, {ssn, zipcode}
- **Candidate key**
  - Is a superkey for which no proper subset is superkey (i.e. minimal)
    - E.g., {ssn}
- **Primary key**
  - Candidate key chosen by database designer as principal means of identifying tuples within relation

Identifying the key

- What is the key in relation
  GRADUATES = (SID, Degree, Major, Year) ?

SID	*Degree*	*Major*	*Year*
123	BS	CS	1992
123	MS	CS	1993
064	BA	History	1991
445	PhD	CS	1999
123	BS	Math	1992
123	MS	Math	1992

Foreign Key

- A relation X may include among its attributes the primary key of another relation, Y
- This attribute is called **foreign key** from X referencing Y
- X is called the referencing relation
- Y is called the referenced relation
- **Example**:
  - OrderNum in table OrderItems is a foreign key, referencing OrderNum in table Orders

Aliasing

```
select *
from people as p, offices as o
where p.name = o.name
```

In general:

- Aliasing can happen for relation names OR for attribute names

```
select  people.name as pn
from people, offices as o
where people.name = o.name
```

Natural Join

- Similar to equijoin, but:
  - Equality predicate is “forced” on all common attributes between the two relations
  - Common attributes are only included once in the results
- Example:

```
select *
from people NATURAL JOIN offices
```

People

Name	Number
Alice	4231
Bob	56
Chris	363

Offices

Name	Building	ROOM
Alice	SENSQ	5432
Bob	CL	32123
Chris	BENDM	105

Natural Join – Results

P.name	P.number	O.building	O.ROOM
Alice	4231	SENSQ	5432
Bob	56	CL	32123
Chris	363	BENDM	105

```
select *
from people NATURAL JOIN offices
```

Natural Join 2

- What if instead of room, we have ‘number’ in Offices?

```
select *
from people NATURAL JOIN offices
```

People

Name	Number
Alice	4231
Bob	56
Chris	363

Offices

Name	Building	Number
Alice	SENSQ	5432
Bob	CL	32123
Chris	BENDM	105

Understanding Question

(Q6) How many rows will the output of the following query have (i.e., what is its cardinality)?

select *

from Catalog NATURAL JOIN Prices

(Q7) How many columns will the output of the following query have (i.e., what is its arity)?

select *

from Catalog NATURAL JOIN Prices