

SQL Introduction

- SQL is a standard language for querying and manipulating data
- SQL is a very high-level programming language
 - This works because it is optimized well!
- Many standards out there:
 - ANSI SQL, SQL92 (a.k.a. SQL2), SQL99 (a.k.a. SQL3),
 - Vendors support various subsets
 - We focus on the most commonly used constructs in SQL

SQL stands for
Structured Query Language

NB: Probably the world's most successful **parallel** programming language (multicore?)

SQL Has Three Major Sub-Languages

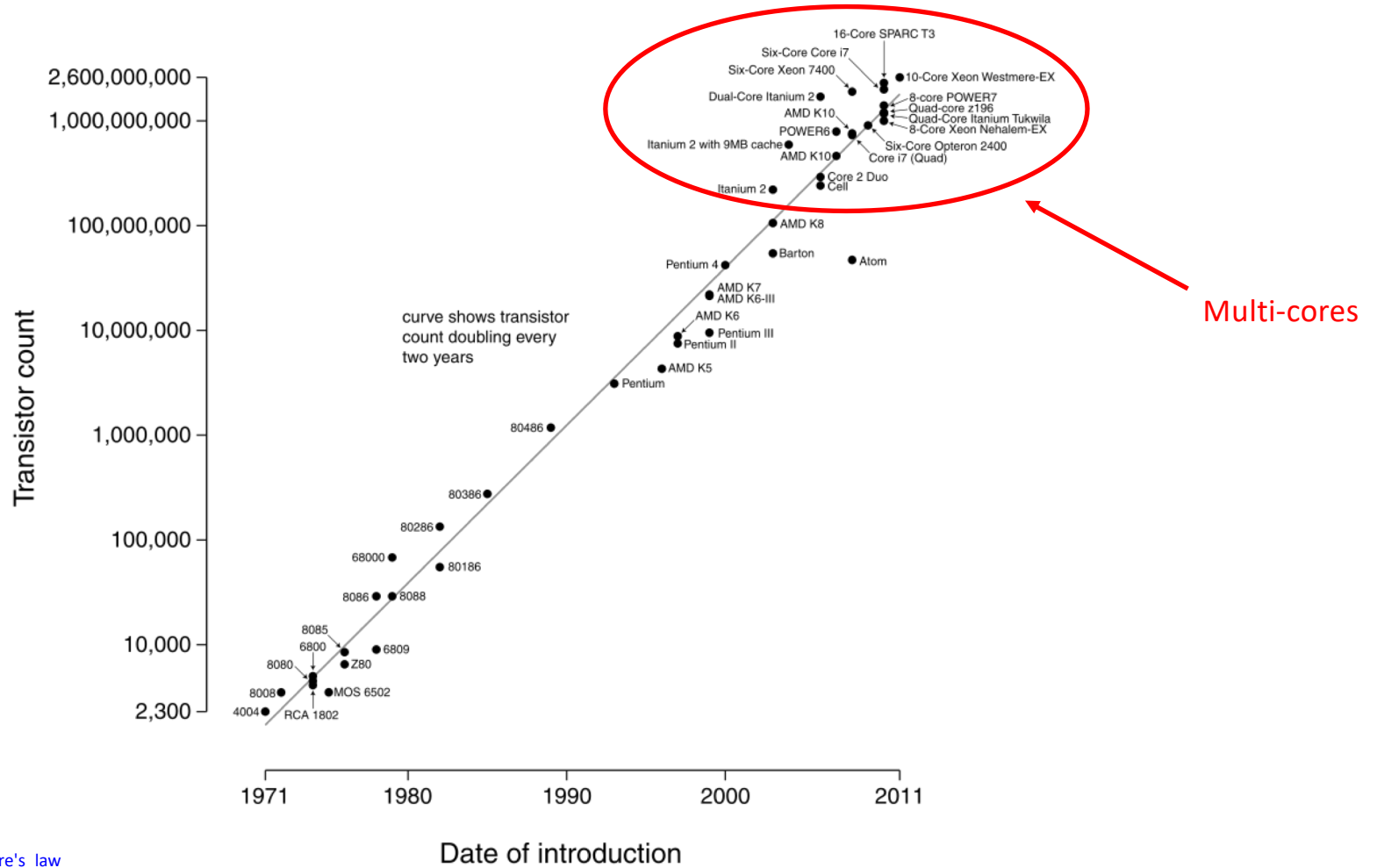
- Data Manipulation Language (DML)
 - Insert/delete/modify tuples in tables
 - Commands that maintain and query a database (our main focus!)
- Data Definition Language (DDL)
 - Define a relational schema (create, alter, and drop tables; establish constraints)
 - Create/alter/drop tables and their attributes
- Data Control Language (DCL)
 - Commands that control a database, including administering privileges and committing data

Most spectacular these days: theoretic potential for perfect scaling!

- perfect scaling
 - given sufficient resources, performance does not degrade as the database becomes larger
- key: parallel processing
- cost: number of processors polynomial in the size of the DB
- all (most) relational operators highly parallelizable

Moore's law

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Source: http://en.wikipedia.org/wiki/Moore's_law

What is SQL?

The Positives

- It's a simple language
- There are only a few key words that you have to learn – it's fairly simple
- It's major purpose is to communicate with a database and ask a database for data
- It's a declarative language (you define what to do)

The Challenges

- Simplicity has it's cost – it gets complex quickly
 - Imagine only having 2 verbs (go, put, wait) to express all you do in a lifetime
 - It's either infeasible or you have to combine a lot basic actions to construct a more complex action
(e.g. skydiving = put parachute into backpack, put the backpack on your back, go airplane, wait until airplane is at 14k feet, go to open door, go outside airplane, ...)
- Declarative programming is perceived as non-intuitive (well, decide for yourself 😊)

Compare semantics between Excel and Database tables

Excel

	A	B	C	D	
1	PName	Price	Category	Manufacturer	table heading
2	Gizmo	19.99	Gadgets	GizmoWorks	
3	PowerGizmo	29.99	Gadgets	GizmoWorks	
4	SingleTouch	149.99	Photography	Canon	
5	MultiTouch	203.99	Household	Hitachi	row

column

Database¹

Table name

TABLE

Product

Search

Show All

rowid	PName	Price	Category	Manufacturer
1	Gizmo	19.99	Gadgets	GizmoWorks
2	PowerGizmo	29.99	Gadgets	GizmoWorks
3	SingleTouch	149.99	Photography	Canon
4	MultiTouch	203.99	Household	Hitachi

attribute name

tuple/ entity/
record/ row

attribute/ field/
column

¹ A Database (DB) is simply a system that holds multiple tables (like Excel has multiple sheets)

Tables in SQL

Attribute names

Table name

Key

Product			
<u>PName</u>	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

Tuple / row
(Entity)

Attribute

Data Types in SQL

- Atomic types
 - Character strings: CHAR(20), VARCHAR(50)
 - Numbers: INT, BIGINT, SMALLINT, FLOAT
 - Others: MONEY, DATETIME, ...
- Record (aka tuple)
 - Every attribute must have an atomic type
- Table (aka relation)
 - A set of tuples (hence tables are flat!)

Table Schemas

- The schema of a table is the table name, its attributes, and their types:

```
Product(Pname: string, Price: float,  
Category: string, Manufacturer: string)
```

- A key is an attribute whose values are unique; we underline a key
 - A key may also be a set of attributes (more later)

Basic SQL

SQL Query

- Basic form (there are many many more bells and whistles)

```
SELECT <attributes>  
FROM   <one or more relations>  
WHERE  <conditions>
```

Call this a SFW query.

Simple SQL Query

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

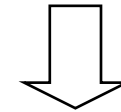
```
SELECT *  
FROM Product  
WHERE category='Gadgets'
```

Simple SQL Query

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

```
SELECT *  
FROM Product  
WHERE category='Gadgets'
```



Selection

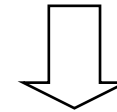
PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks

Simple SQL Query

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

```
SELECT pName, price, manufacturer
FROM Product
WHERE price > 100
```



Selection
& Projection

PName	Price	Manufacturer
SingleTouch	\$149.99	Canon
MultiTouch	\$203.99	Hitachi

Selection vs. Projection

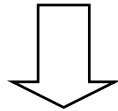
Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

One **projects** onto some attributes (columns)
-> happens in the **SELECT** clause

One **selects** certain entires=tuples (rows)
-> happens in the **WHERE** clause
-> acts like a **filter**

```
SELECT pName, price
FROM   Product
WHERE  price > 100
```



PName	Price
SingleTouch	\$149.99
MultiTouch	\$203.99

SQL: A Few Details on Syntax

- SQL commands are case insensitive:
 - SELECT = Select = select
 - Product = product, Category = category
- But values are not:
 - Different: 'Gadgets', 'gadgets'
 - (Notice: in general, but default settings will vary from DBMS to DBMS. E.g. MySQL is case insensitive. Just to be safe, always assume values to be case sensitive!)

```
WHERE LOWER(Category)='gadgets'
```

- Use single quotes for constants:
 - 'abc' - yes
 - "abc" - no (except MySQL and SQLite)

Eliminating Duplicates

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
PowerGizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

Set vs. Bag
semantics

```
SELECT category  
FROM Product
```



Category
Gadgets
Gadgets
Photography
Household

```
SELECT DISTINCT category  
FROM Product
```



Category
Gadgets
Photography
Household

Ordering the Results

```
SELECT pName, price, manufacturer
FROM   Product
WHERE  category='Gadgets'
      and price > 10
ORDER BY price, pName
```

- Ties in attribute *price* broken by attribute *pname*
- Ordering is ascending by default. Descending:

```
... ORDER BY price ASC, pname DESC
```

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

```
SELECT DISTINCT category
FROM   Product
ORDER BY category
```



?

```
SELECT category
FROM   Product
ORDER BY pName
```



?

```
SELECT DISTINCT category
FROM   Product
ORDER BY pName
```



?

Product

PName	Price	Category	Manufacturer
Gizmo	\$19.99	Gadgets	GizmoWorks
Powergizmo	\$29.99	Gadgets	GizmoWorks
SingleTouch	\$149.99	Photography	Canon
MultiTouch	\$203.99	Household	Hitachi

```
SELECT DISTINCT category
FROM Product
ORDER BY category
```



Category
Gadgets
Household
Photography

```
SELECT category
FROM Product
ORDER BY pName
```



Category
Gadgets
Household
Gadgets
Photography

```
SELECT DISTINCT category
FROM Product
ORDER BY pName
```



Syntax error on large more "principled" DBMSs
(Oracle, PostgreSQL, SQL server) /
unpredictable results on others(MySQL, SQLite)

"ORDER BY items must appear in the select list if SELECT DISTINCT is specified."

Some history


Some "birth-years"

- 2004: Facebook
- 1998: Google
- 1995: Java, Ruby
- 1993: World Wide Web
- 1991: Python
- 1985: Windows
- 1974: SQL

SQL: Declarative Programming

SQL

```
select (e.salary / (e.age - 18)) as comp  
from employee as e  
where e.name = "Jones"
```



Declarative Language: you say what you want
without having to say how to do it.

Procedural Language: you have to specify exact
steps to get the result.

SQL: was not the only Attempt

SQL

```
select (e.salary / (e.age - 18)) as comp  
from employee as e  
where e.name = "Jones"
```

QUEL

```
range of e is employee  
retrieve (comp = e.salary / (e.age - 18))  
where e.name = "Jones"
```

Commercially not used anymore since ~1980

DBMSs we discuss in this class

- PostgreSQL (Required)
 - popular and powerful open source database

We prefer PostgreSQL over MySQL because it has a more principled interpretation of SQL (and a powerful EXPLAIN command)