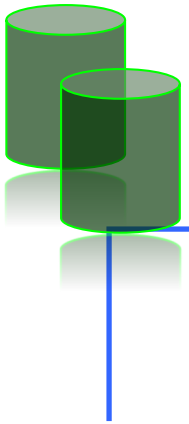


CZ2007
Introduction to Databases

NANYANG TECHNOLOGICAL UNIVERSITY

Querying Relational Databases using SQL
Part-1

Cong Gao

Professor
School of Computer Science and Engineering
Nanyang Technological University, Singapore

Why Should You Study Databases?

NANYANG TECHNOLOGICAL UNIVERSITY

- **Make more $$$:**
  - Startups need DB talent right away
  - Massive industry…

- **Intellectual (Research):**
  - Science: data poor to data rich
    - No idea how to handle the data!
  - Fundamental ideas to/from all of CS:
    - Systems, theory, AI, logic, stats, analysis….

Many great computer systems ideas started in DB.

Ask Questions!

NANYANG TECHNOLOGICAL UNIVERSITY

The important thing is not to stop questioning.

Albert Einstein

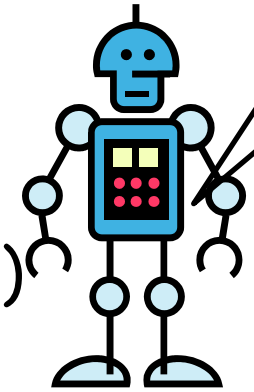
Database and DBMS

Database

What is the average annual income of a Singapore tax payer?

SGD
S$5,596

Database Management System (DBMS)



User

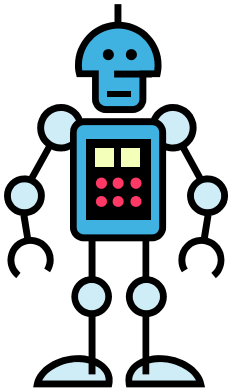
Tables, Relations, Relational Model

Database

Taxpayer_ID	Annual_Income
51248297	100000
33891634	50000
…	…

Income_Table

Database Management System



User

Tables, Relations, Relational Model

Database

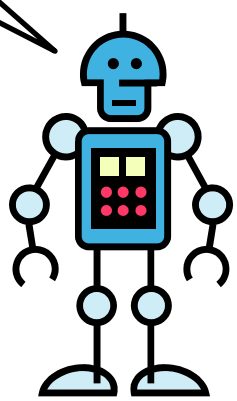
Taxpayer_ID	Annual_Income
51248297	100000
33891634	50000
…	…

Income_Table

???

What is the average annual income of a Singapore tax payer?

Database Management System



User

Structured Query Language (SQL)

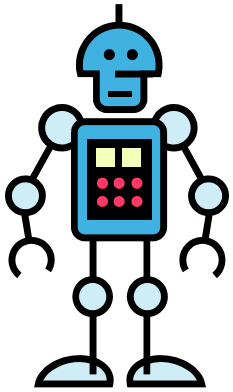
Database

Taxpayer_ID	Annual_Income
51248297	100000
33891634	50000
…	…

Income_Table

```
SELECT avg(Annual_Income)
FROM   Income_Table
```

Database Management System



User

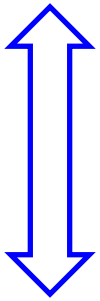
Structured Query Language (SQL)

Database

Taxpayer_ID	Annual_Income
51248297	100000
33891634	50000
…	…

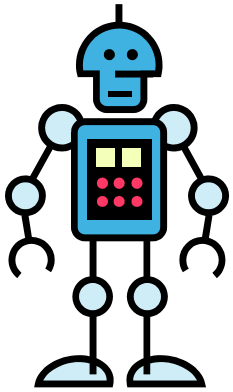
Income_Table

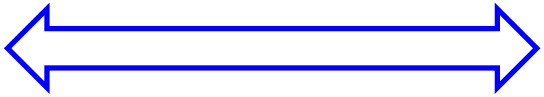
More details about SQL will be covered in the course



```
SELECT avg(Annual_Income)
FROM   Income_Table
```

Database Management System





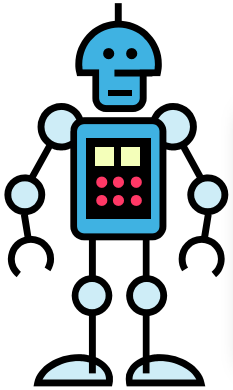
User

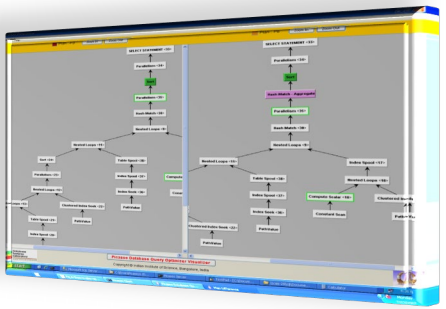
Querying RDBMS
(Relational Database Management System)

SQL

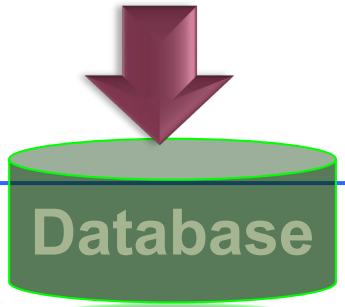
DBMS Interface

Database Management System





Relational Algebra



Roadmap

NANYANG TECHNOLOGICAL UNIVERSITY

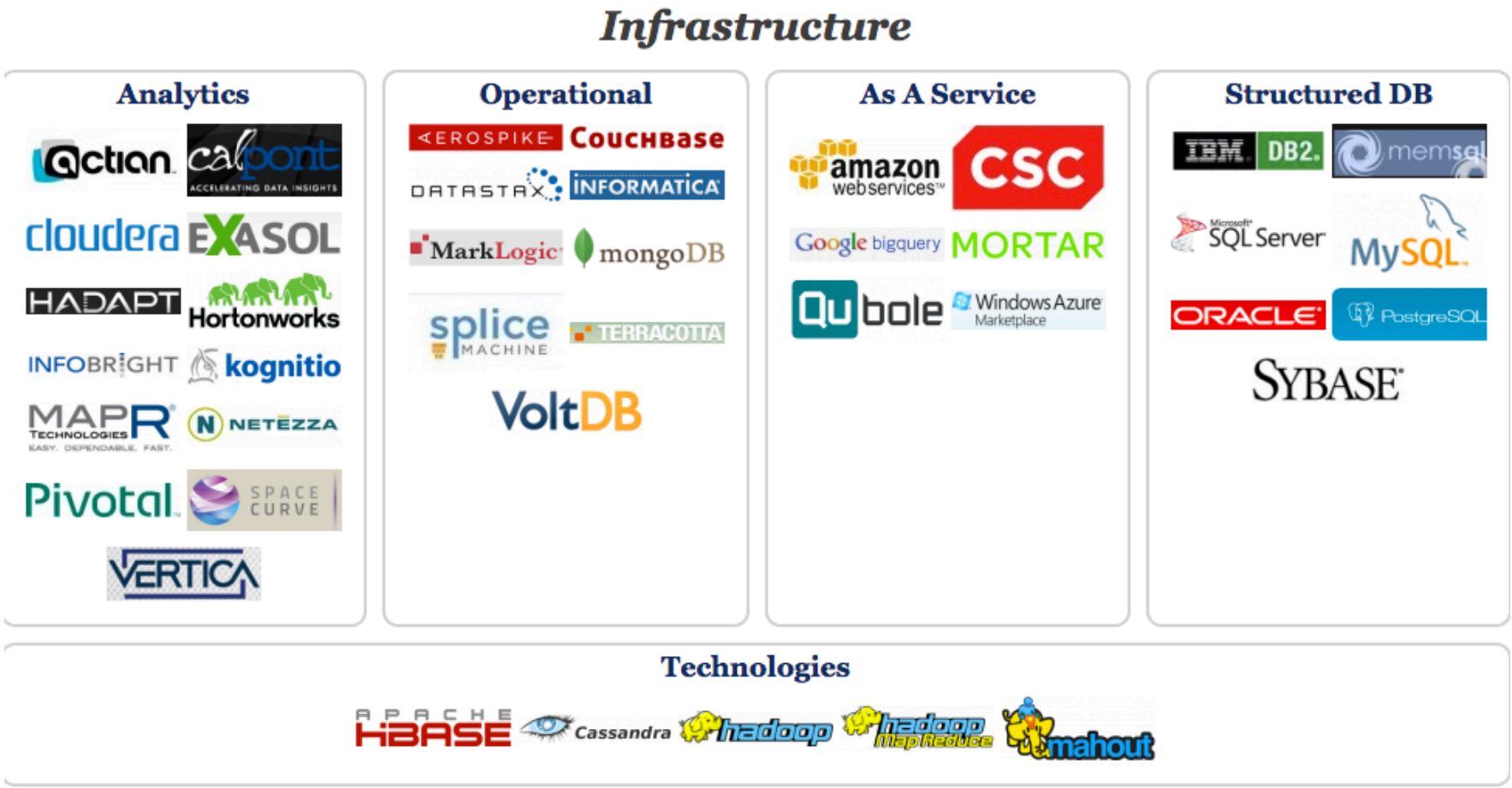
- Next 9 lectures
  - Introduction to SQL
  - SELECT
    FROM
    WHERE
  - Eliminating duplicates
  - Renaming attributes
  - Expressions in SELECT Clause
  - Patterns for Strings
  - Ordering
  - Joins
  - …

What is SQL?

- **Structured Query Language** (SQL) – standard query language for relational databases. Pronounced "**S-Q-L**" or "**sequel**"

- **A brief history:**
  - First proposal of SEQUEL (IBM Research, System R, 1974)
  - First implementation in SQL/DS (IBM) and Oracle (1981)
  - Around 1983 there is a "de facto standard"
  - Became official standard in 1986 – defined by the American National Standards Institute (ANSI), and in 1987 – by the International Organization for Standardization (ISO)
  - ANSI SQL89
  - ANSI SQL92 (SQL2)
  - ANSI SQL99 (SQL3)
  - ANSI SQL 2003 (added OLAP, XML, etc.)
  - ANSI SQL 2006 (added more XML)
  - … …. …
  - ANSI SQL 2016 (added pattern matching, JSON, etc.)

Present Days: Big Data



New technology. *Same* SQL Principles

What SQL we shall study?

NANYANG TECHNOLOGICAL UNIVERSITY

- All major database vendors (Oracle, IBM, Microsoft, Sybase) conform to SQL standard

- Although database companies have added "proprietary" extensions (different dialects)

- Commercial systems offer features that are not part of the standard
  - Incompatibilities between systems
  - Incompatibilities with newer standards (e.g. triggers in SQL:1999)

– We concentrate more on the principles

– (mostly) We will study SQL92 - a basic subset

Good Practice for learning SQL

- Install a DBMS in your machine, such as PostgreSQL, mySQL, etc
  - Set up a database and tables with example data
  - Run SQL, debug yourself
- SQL server in school lab
- Google error (stackoverflow)
- Consult textbook
- Other sources:
  - An easy to use website:
    - https://www.w3schools.com/sql/default.asp
    - Can try to run SQL using the example database there
  - Comparison of different SQL implementations – by Troels Arvin (http://troels.arvin.dk/db/rdbms/)

What we want to do with SQL?

NANYANG TECHNOLOGICAL UNIVERSITY

- Manage and query the database (a set of relations / tables)

What we want to do on the relations?
• Retrieve
• Insert
• Delete
• Update

More about SQL

NANYANG TECHNOLOGICAL UNIVERSITY

Declarative Language

- SQL is a ***declarative language*** (non-procedural).
- A SQL query specifies *what* to retrieve but not *how* to retrieve it.

What is a procedural language ??

- Procedure/ Functions
- Write instructions on *how* to do it
- C, C++, Java

SQL is Not a complete programming language

- It does not have control or iteration commands.

Stuffs supported by SQL

Data Manipulation Language (DML)

- Perform queries
- Perform updates (add/ delete/ modify)

Data Definition Language (DDL)

- Creates databases, tables, indices
- Create views
- Specify integrity constraints

Embedded SQL

Wrap a high-level programming language around DML to do more sophisticated queries/updates

We shall not study this !

Tables in SQL

- A **relation** or **table** is a multiset of tuples (rows) having the attributes specified by the schema
  - Schema: the name of a relation + the set of attributes

A **multiset** is an unordered list (or: a set with multiple duplicate instances allowed)

List: [1, 1, 2, 3]
Set: {1, 2, 3}
Multiset: {1, 1, 2, 3}

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

Attributes (Columns) in a Table

An **attribute** (or **column**) is a typed data entry present in each tuple in the relation

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

Tuples (Rows) in a Table

A **tuple** or **row** is a single entry in the table having the attributes specified by the schema

*Also referred to sometimes as a **record***

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

Data Types in SQL

- Character strings
  - CHAR(20)
  - VARCHAR(50)
  - …
- Numbers
  - INT
  - FLOAT
  - …
- Others
  - BOOLEAN
  - DATETIME
  - …

Every attribute must have an type

Product

PName	Price	Category
iPhone x	888	Phone
iPad	668	Tablet
Mate 10	798	Phone
EOS 550D	1199	Camera

Key of a Table

NANYANG TECHNOLOGICAL UNIVERSITY

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

- A **key** is an attribute whose values are unique; we underline a primary key

Product(Pname, Price, Category, Manufacturer)

Principle Form of SQL

NANYANG TECHNOLOGICAL UNIVERSITY

Basic Structure of SQL

SELECT desired attributes (***A1, A2, … , An)***
FROM one or more tables (***R1, R2, … , Rm)***
WHERE condition about tuples of the tables (***P)***

Mapping to Relational Algebra

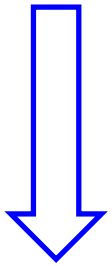
$$\Pi_{A1, A2, \ldots, An} (\sigma_P (R1 \times R2 \times \ldots \times Rm))$$

Simple SQL Query

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  Category = 'Phone'
```



"selection"

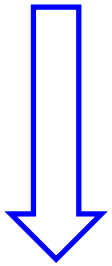
PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
Mate 10	798	Phone	Huawei

Simple SQL Query

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  Category <> 'Phone'
```



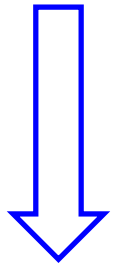
PName	Price	Category	Manufacturer
iPad	668	Tablet	Apple
EOS 550D	1199	Camera	Canon

Simple SQL Query

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  Category = 'Phone' AND Price > 800
```



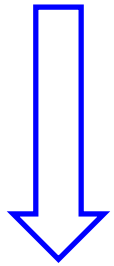
PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

Simple SQL Query

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  Category = 'Tablet' OR Price > 1000
```



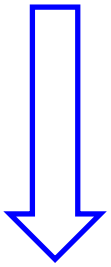
PName	Price	Category	Manufacturer
iPad	668	Tablet	Apple
EOS 550D	1199	Camera	Canon

Simple SQL Query (selection and projection)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Price, Manufacturer
FROM    Product
WHERE   Price > 800
```



"selection and projection"

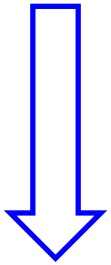
PName	Price	Manufacturer
iPhone x	888	Apple
EOS 550D	1199	Canon

Simple SQL Query (WHERE Clause: BETWEEN)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Price, Manufacturer
FROM    Product
WHERE   Price  BETWEEN 800 AND 1200
```



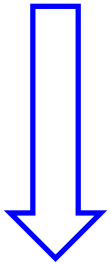
PName	Price	Manufacturer
iPhone x	888	Apple
EOS 550D	1199	Canon

Simple SQL Query (WHERE Clause: IN)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Price, Manufacturer
FROM    Product
WHERE   Manufacturer  IN ('Huawei', 'Canon')
```



PName	Price	Manufacturer
Mate 10	798	Huawei
EOS 550D	1199	Canon

SQL Syntax

- There is a set of *reserved words* that cannot be used as names for table name or attribute name. For example, SELECT, FROM, WHERE, etc.
- Use single quotes for constants:
  - 'abc' – Okay
  - "abc" – Not okay
- SQL is generally *case-insensitive*.
  - Exception: is string constants. 'FRED' not the same as 'fred'.
- White-space is ignored
- All statements end with a semicolon (;)

Summary and Roadmap

- Introduction to SQL
- SELECT
  FROM
  WHERE

- Next
  - Eliminating duplicates
  - Renaming attributes
  - Expressions in SELECT Clause
  - Patterns for Strings
  - Ordering
  - Joins
  - …

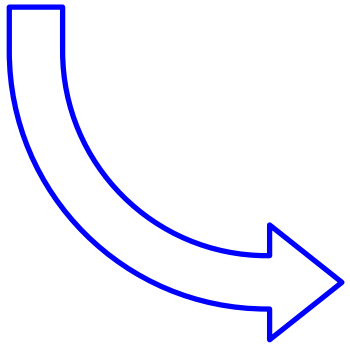
Reference: Chapter 6.1 of the Book "Database Systems: The Complete Book; Hector Garcia-Molina Jeffrey D. Ullman, Jennifer Widom

Eliminating Duplicates

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  Category
FROM    Product
```



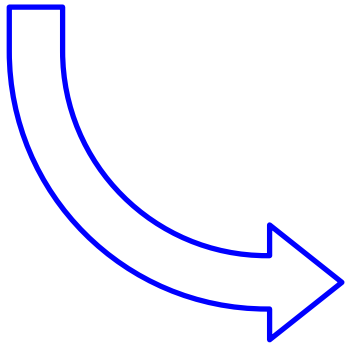
Category
Phone
Tablet
Phone
Camera

Eliminating Duplicates (cont.)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT DISTINCT Category
FROM   Product
```



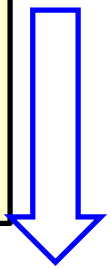
Category
Phone
Tablet
Camera

AS: Renaming Attributes

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT PName AS Product, Price AS Cost,Manufacturer
FROM   Product
WHERE  Category = 'Phone'
```



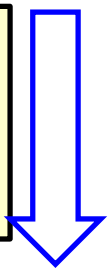
Product	Cost	Manufacturer
iPhone x	888	Apple
Mate 10	798	Huawei

Expressions in SELECT Clause

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT PName, Price*1.4 AS Cost_IN_SGD, Manufacturer
FROM   Product
WHERE  Category = 'Phone'
```



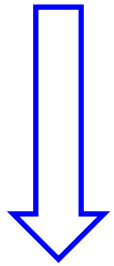
PName	Cost_IN_SGD	Manufacturer
iPhone x	1243.2	Apple
Mate 10	1117.2	Huawei

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName LIKE 'iPh%'
```



PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

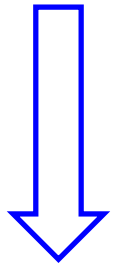
% stands for "any string"

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName LIKE '%Phone x%'
```



PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

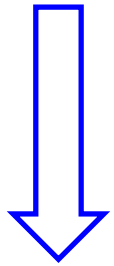
% stands for "any string"

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName LIKE '%P%e%'
```



PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

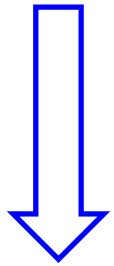
% stands for "any string"

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName LIKE '_Phone x'
```



PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

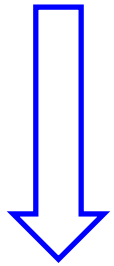
_ stands for "any character"

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName LIKE '_Phone__'
```



PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple

_ stands for "any single character"

Patterns for Strings

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT *
FROM   Product
WHERE  PName NOT LIKE '_Phone__'
```

More example patterns

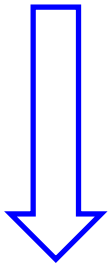
- '_ _ _' - Matches any string of exactly three characters
- '_ _ _%' - Matches any string of at least three characters
- 'ab\%cd%' - Match all strings beginning with "ab%cd"

Ordering the Results (cont.)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Price
FROM    Product
WHERE  Price < 800
ORDER BY  PName DESC
```



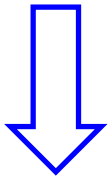
PName	Price
iPad	668
Mate 10	798

Ordering the Results (cont.)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Category
FROM    Product
WHERE  Price < 1000
ORDER BY  Category, PName
```



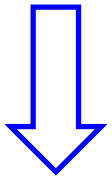
PName	Category
Milestone	Phone
iPhone x	Phone
iPad	Tablet

Ordering the Results (cont.)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Category
FROM    Product
WHERE   Price < 1000
ORDER BY  Category DESC,
                PName
```



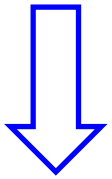
PName	Category
iPad	Tablet
Mate 10	Phone
iPhone x	Phone

Ordering the Results (cont.)

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  PName, Category
FROM    Product
WHERE   Price < 1000
ORDER BY  Category DESC,
                PName DESC
```



PName	Category
iPad	Tablet
iPhone x	Phone
Mate 10	Phone

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Motorola
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  Category
```

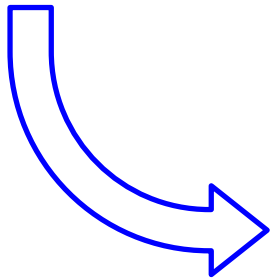
?

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  Category
```



Category
Camera
Phone
Tablet

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  Category
WHERE  Price < 1000
```

?

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  Category
WHERE  Price < 1000
```

Error!

- "WHERE" should always proceed "ORDER BY"

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  PName
```

?

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

```
SELECT  DISTINCT Category
FROM    Product
ORDER BY  PName
```

Error!

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

Joins

Company

CName	StockPrice	Country
Canon	45	Japan
Huawei	1	China
Apple	374	USA

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

- A user wants to know the names and prices of all products by Japan companies. How?

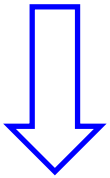
Joins

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

Company

CName	Stock Price	Country
Apple	374	USA
Huawei	1	China
Canon	45	Japan



- SELECT PName, Price
  FROM Product, Company
  WHERE Country = 'Japan'
  AND Manufacturer = CName

PName	Price
EOS 550D	1199

Meaning (Semantics) of Join – An Example

```
SELECT R.A
FROM   R, S
WHERE  R.A = S.B
```

R

A
1
3

S

B	C
2	3
3	4
3	5

Cross Product

A	B	C
1	2	3
1	3	4
1	3	5
3	2	3
3	3	4
3	3	5

Apply Selections / Conditions

A	B	C
3	3	4
3	3	5

Apply Projection

Output

A
3
3

Summary of Meaning (Semantics) of Join

```
SELECT R.A
FROM   R, S
WHERE  R.A = S.B
```

1. Take **cross product**:

$$X = R \times S$$

2. Apply **selections / conditions**: = Filtering!

3. Apply **projections** to get final output:

= Returning only *some* attributes

Remembering this order is critical

How Join is Actually Executed in a Database System?

- The preceding slides show what a join means (i.e., semantics)
- Not actually how the DBMS executes it under the covers

We shall not study it in this course – will be discussed in CZ 4031

Joins

Person

PName	Address	WorksFor
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA
- SELECT PName
  FROM Person, Company
  WHERE Country = 'USA'
  AND WorksFor = CName

Joins

Person

PName	Address	WorksFor
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT PName, Address
  FROM Person, Company
  WHERE Country = 'USA'
  AND WorksFor = CName

Error!

Joins

Person

PName	Address	WorksFor
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT PName, Company.Address
  FROM Person, Company
  WHERE Country = 'USA'
  AND WorksFor = CName

Joins

Person

PName	Address	CName
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT PName, Company.Address
  FROM Person, Company
  WHERE Country = 'USA'
  AND CName = CName

Error!

Joins

Person

PName	Address	CName
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT PName, Company.Address
  FROM Person, Company
  WHERE Country = 'USA'
  AND Person.CName = Company.CName

Joins

Person

PName	Address	CName
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT X.PName, Y.Address
  FROM Person AS X, Company AS Y
  WHERE Y.Country = 'USA'
  AND X.CName = Y.CName

Joins

Person

PName	Address	CName
…	…	…

Company

CName	Address	Country
…	…	

- Find the names of the persons who work for companies in USA, as well as their company addresses
- SELECT X.PName, Y.Address
  FROM Person X, Company Y
  WHERE Y.Country = 'USA'
  AND X.CName = Y.CName

Exercise

Company

CName	StockPrice	Country
...	...	...

Product

PName	Price	Category	Manufacturer
...	...	...	...

- Exercise: Find the names of the companies in China that produce products in the 'tablet' category
- SELECT DISTINCT CName
  FROM Company, Product
  WHERE Manufacturer = CName
  AND Country = 'China'
  AND Category = 'Tablet'

Exercise

Company

CName	StockPrice	Country
...	...	...

Product

PName	Price	Category	Manufacturer
...	...	...	...

- Exercise: Find the names of the companies in China that produce products in the 'tablet' or 'phone' category
- SELECT DISTINCT CName
  FROM Company, Product
  WHERE Manufacturer = CName
  AND Country = 'China'
  AND (Category = 'Tablet'
  OR Category = 'Phone')

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

- Exercise: Find the manufacturers that produce products in both the 'tablet' and 'phone' categories
- SELECT DISTINCT Manufacturer
  FROM Product
  WHERE Category = 'Tablet'
  AND Category = 'Phone'

Error!

Exercise

Product

PName	Price	Category	Manufacturer
iPhone x	888	Phone	Apple
iPad	668	Tablet	Apple
Mate 10	798	Phone	Huawei
EOS 550D	1199	Camera	Canon

- Exercise: Find the manufacturers that produce products in both the 'tablet' and 'phone' categories
-
```
SELECT  DISTINCT X.Manufacturer
FROM    Product AS X, Product AS Y
WHERE   X.Manufacturer = Y.Manufacturer
        AND X.Category = 'Tablet'
        AND Y.Category = 'Phone'
```

Summary and roadmap

- Introduction to SQL
- SELECT
  FROM
  WHERE
- Eliminating duplicates
- Renaming attributes
- Expressions in SELECT Clause
- Patterns for Strings
- Ordering
- Joins

- Next
  - Subquery
  - Aggregations
  - UNION, INTERSECT, EXCEPT
  - NULL
  - Outerjoin
  - …

Reference: Chapter 6.1&6.2 of the Book "Database Systems: The Complete Book; Hector Garcia-Molina Jeffrey D. Ullman, Jennifer Widom