

# DATABASE METHODOLOGY

## Relational Database Theory

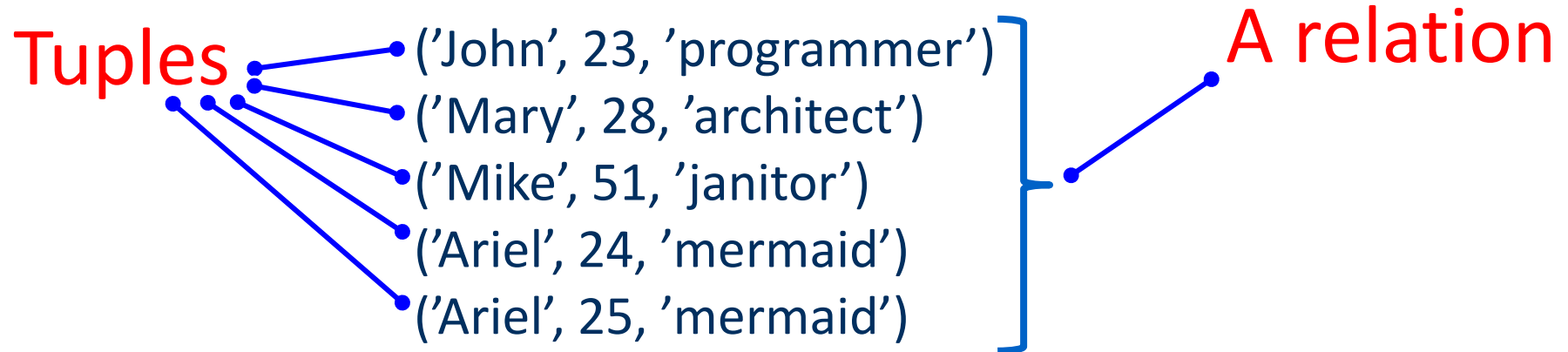
### The Relational Model Part 2 - Terminology

# The Relational Model - Terminology

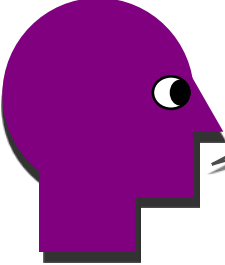
- In this module you will learn about the basic terminology of the Relational Model:
  - Relations defined, tuples
  - Relation schemas, attributes, domains, degree, cardinality
  - Relations vs. tables, relation schemas vs. table definitions, attributes vs. columns, tuples, vs. rows
  - Relational database schemas (models)

# Relations Defined

- A relation is a *set* of tuples
  - So, each of those tuples is an *element* of that set
  - Each tuple is in turn an *ordered list* of one or more values
  - All tuples in the set/relation have the same structure
  - Each tuple is **unique**, because sets do not have duplicate elements!
  - The order between *the tuples* is irrelevant, since the elements in a set are unordered!



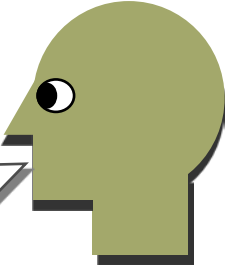
# Storing Relations



Very well, so the tuple order is irrelevant according to the Relational Model, but when we implement a database using this data model, then there must be some *physical* data order, right?

Yes, you're right, on a database server's hard drive(s), the data is stored in various memory addresses, which might be in sequence, or might not be.

However, the users of the database should normally and preferably not have to be aware of those lower details.



# Relation Schemas

A relation's structure is characterized (described) in a **Relation schema**

## Relation schema (name)

A relation schema is identified by its name

Person

|             |              |              |                 |
|-------------|--------------|--------------|-----------------|
| ssn: String | name: String | age: Integer | salary: Integer |
|-------------|--------------|--------------|-----------------|

## Attribute (name)

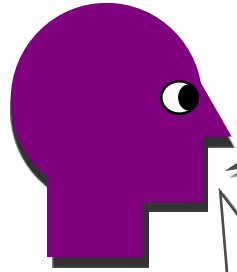
A label describing some property (attribute) of the tuples in the relation schema

## Domain (name)

A domain is a named set of rules defining the *allowable* values for one or more attributes. Often just a **datatype**

The relation schema is shown as a header on top of the tuples of the relation but is not part of the relation itself: It is *meta* data, there to aid in understanding the data

# More Terminology



The **intension** of a relation is its structure, i.e. its relation schema

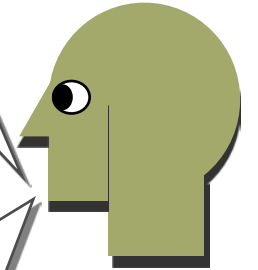
The **degree** of a relation is the *number of attributes* it has.

The degree is usually fixed, but is changed if attributes are added or deleted

The **extension** of a relation is all the tuples currently in the relation

The **cardinality** of a relation is the *current number of tuples* in the relation

The cardinality is not a fixed number, it changes as the number of tuples in the relation changes



# Relations Vs. Tables

- Relations are *implemented* using **tables**
- Tuples become **rows**
- Relation schemas become **table definitions**
- Attributes become **columns**
- Each column gets the attribute's domain

Textual syntax for **table definitions**, *including* the domains:

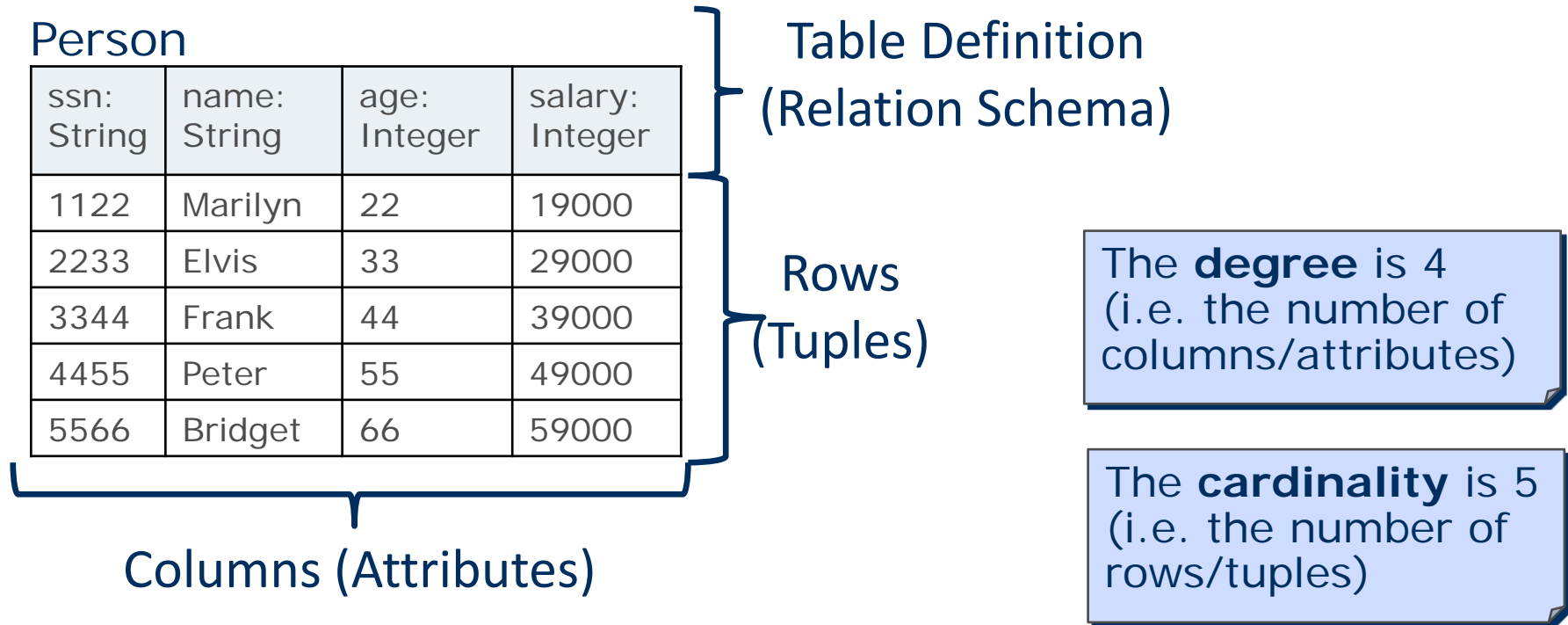
```
TableName(col_1:domain, col_2:domain, ..., col_N:domain)
```

Textual syntax for **table definitions**, *without* the domains:

```
TableName(col_1, col_2, ..., col_N)
```

# Relations Vs. Tables, continued

- A relation implemented as a table
  - 5 tuples  $\Rightarrow$  5 rows





# Attribute/Column Order?

- **Formally (in the Relational Model):**
  - The order of attributes in a relation is irrelevant
- **In practise (in a relational database):**
  - At “run time”, we might have to know which column order that was chosen at “design time”

Formally, the same relation!

| name  | ssn         | weight |
|-------|-------------|--------|
| Ollie | 111111-1111 | 81     |
| Peter | 222222-2222 | 59     |
| Lisah | 999999-9999 | 63     |

| ssn         | name  | weight |
|-------------|-------|--------|
| 111111-1111 | Ollie | 81     |
| 222222-2222 | Peter | 59     |
| 999999-9999 | Lisah | 63     |

# Relational Database Schemas (Models)

- A relational *database* consists of tables/relations
  - Tens/hundreds/thousands of tables/relations
  - Logically connected (i.e. logically related)
- Each of the tables/relations
  - Has a table definition/relation schema
    - With a *distinct* name (unique name)
- A **Relational *database* schema (model)**
  - Describes the set of tables/relations in the database
    - And the logical connections between them

# An Example Relational DB Schema

*In textual notation (there are others)*

Person(ssn, name, address, age)

Car(regNo, brand)

Ownership(person, car) Ownership.person is FK to Person.ssn, Ownership.car is FK to Car.regNo

# What Was Learnt In This Module?

- Now you know the basics about
  - Relations
  - Relation schemas
  - How tables implement relations
  - Relational database schemas (models)
- *But there's more!*

# Medverkande

Anders Thelemyr – Lärare

Lars In de Betou – Mediepedagog

Inspelat 2015-08-31

Institutionen för data- och systemvetenskap, DSV



Stockholms  
universitet