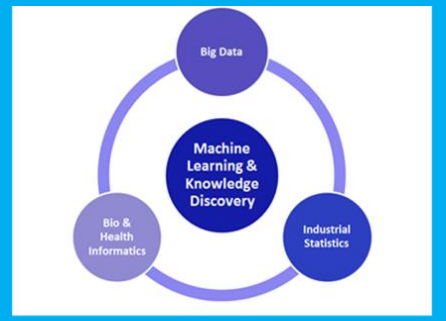
521156S
TOWARDS DATA MINING

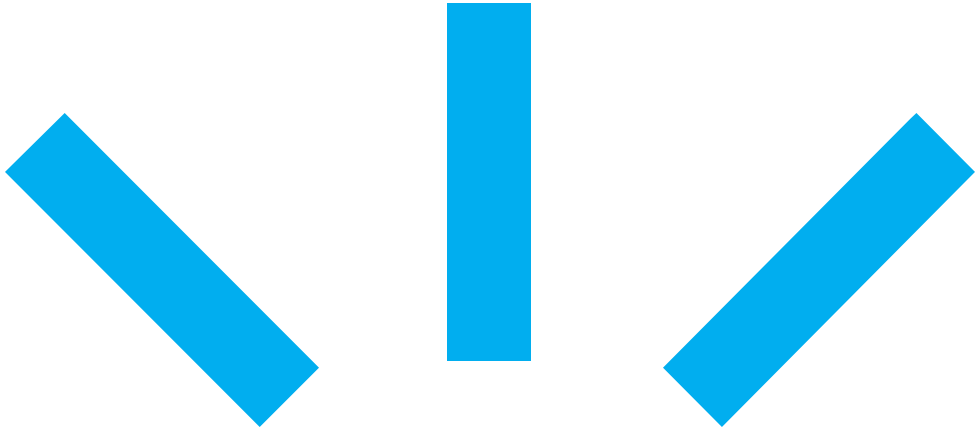
MATKALLA TIEDONLOUHINTAAN

UNIVERSITY OF OULU

BISG Oulu



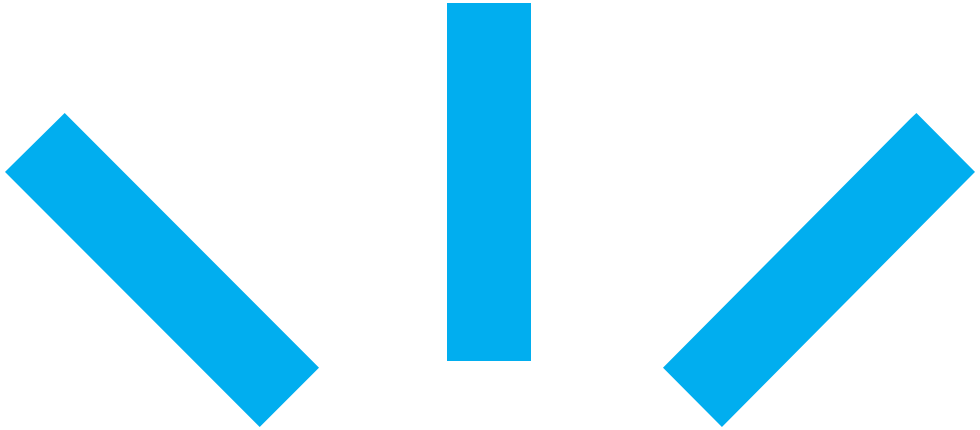
Data Analysis and Inference Group



Ethical Considerations in Data Mining, Information Security and Privacy, Open Data – Using and producing, Data Management and Databases, Relational data modeling

Anna-Mari Wartiainen

Material written by
Lauri Tuovinen, Postdoctoral Researcher
DataAI/BISG, University of Oulu



Ethical Considerations in Data Mining

Why Data Ethics?

- When working with digital data, you're part scientist, part engineer – two distinct sets of professional ethics at play
- As a scientist, you're accountable to the scientific community (and society/humanity in general)
- As an engineer, you're accountable to your employer, your clients and any other stakeholders
- If it's personal data you're using, you're also accountable to the individuals concerned

Knowledge Is Power

- …and power corrupts? Not necessarily, but it's a possibility
- Data is not knowledge, but it holds the *potential* for knowledge – that's the whole point of data mining
- Therefore, having some data and the means to mine it puts you in a position of power; data ethics is about what you do with that power

Types of Ethical Issues

- **Ethical issues in data mining may be related to:**
  - **Motives:** pursuing questionable objectives, **or…**
  - **Methods:** employing unacceptable means, **or…**
  - **Consequences:** causing harmful effects
  - You need to be honest and open about what you are planning to do
  - You need to carefully consider who may be affected by your actions and how

Basic Guidelines

- **Data is like software: it may be public domain, but usually it's under some sort of license**
  - Be aware of who owns the data and of the terms under which they're willing to share it with you
- **In particular, data that's yours to analyse is not necessarily yours to distribute**
  - Being permitted to disclose the data doesn't necessarily mean that you should – consider the potential implications from the point of view of everyone involved, including yourself
- **If you're collecting the data yourself and it involves other people, make sure they are completely and accurately informed on what data you're collecting and how you will use it**

Whose Data Is It Anyway?

- **Generally speaking, data is owned by someone, and the owner gets to decide who has access to the data**
  - Therefore the first thing you need to find out is who owns the data you want to use
- **If you collected it, it's yours – look out for your own interests first**
  - It's okay to be a little selfish here, even if you're going to release the data later

Terms and Conditions

- **If you got the data from someone else, there are likely to be some terms under which you are allowed to use it – be aware of them and do not violate them**
- At the very least, if you publish your results, you should credit the sources of the data by way of citation or acknowledgment
- **Even if it's your data, remember that there may be other people who have a say in what you can do with it**

Likely Stakeholders

- Your employer: are there policies you need to observe when dealing with research data?
- Other organisations: are there business or other interests at stake that you need to take into account?
- Individuals: are there reservations concerning information privacy that you need to honour?

Institutional Ethics

- Some organisations (e.g. universities) have their own internal ethics review boards appointed to uphold the ethical principles of the organisation
- If you’re mining data that’s personal or otherwise sensitive, the ethics board may want to evaluate your procedures
- A statement from the relevant ethics board may also be required as an appendix to the proposal when applying for a grant

Mining Personal Data

- Personal data is a special case: it's not just who you share the data with that you need to be wary of, it's also what the data might reveal to you
- Your data subjects have a right to privacy – you need to design your data mining effort to accommodate it
- Key concepts: anonymity and consent

Informed Consent

- Originally a principle of medical ethics, but can be extended to cover mining of personal data
- Consent: you need permission from each individual whose data you wish to use
- Informed: consent only counts when based on a complete understanding of what data will be collected and how it will be used
- If you’re doing the collecting, you’re responsible for ensuring this

Consent Is Not Carte Blanche

- **Informed consent is not the end of ethical responsibility – you need to take care of your end of the bargain**
- **Remember, consent is given under the assumption that certain constraints apply; these may include:**

- The data is kept separate from any information that might identify individual subjects
- The data is not combined with additional information on the subjects obtained from other sources
- The data is only mined for specific named purposes
- The data will be deleted without delay once the stated purposes have been fulfilled

Possible Harmful Outcomes

- Unexpected identification of individuals from a combination of non-identifying attributes
- Unexpected discovery of sensitive information from a combination of non-sensitive attributes
- Taking action based on a false positive, leading to a violation of rights – think data mining for crime prevention

Protecting Your Subjects

- **The main defense against unintentional breach of privacy is anonymity: the data should be purged of any information that's too closely associated with the identity of an individual subject**
- **This may not be as easy as it sounds, since variables that ordinarily would not be considered identifying, may turn out to be identifying when subjected to data mining**
- **To avoid harmful consequences from false positives, the output of a data mining algorithm should not just be taken at face value – the more severe the consequences, the more important it is to understand the logic behind the result**
- Depending on the method used, this may or may not be feasible

Legal Perspective

- Handling digital data is regulated on both national and EU level
- Ownership of datasets and the exclusive rights of owners are governed by intellectual property laws
- The proper conduct when processing personal data and the rights of data subjects are governed by data protection laws

Finnish Laws and Regulations

- **Data Protection Act (Tietosuojalaki 1050/2018): the main act of parliament governing the processing of personal data**
  - Specifies and supplements the General Data Protection Regulation of the European Union
  - Repeals the old Personal Data Act of 1999
- **Copyright Act (Tekijänoikeuslaki 404/1961): Section 49 grants exclusive control over replication and publication to the producer of a catalogue or database**
  - Individual elements of the catalogue or database may be copyrightable in themselves
- **More information: Avointiede.fi, Finlex**

EU Laws and Regulations

- **General Data Protection Regulation (GDPR), implemented on 25 May, 2018**
  - Applies to the processing of personal data of data subjects residing in the EU, regardless of where the processor is located
  - Strengthens the conditions for consent and expands the rights of data subjects
  - As an EU regulation, takes effect automatically – does not need to be enacted by national governments
- **More information:**
  EUGDPR.org
  https://ec.europa.eu/info/law/law-topic/data-protection_en

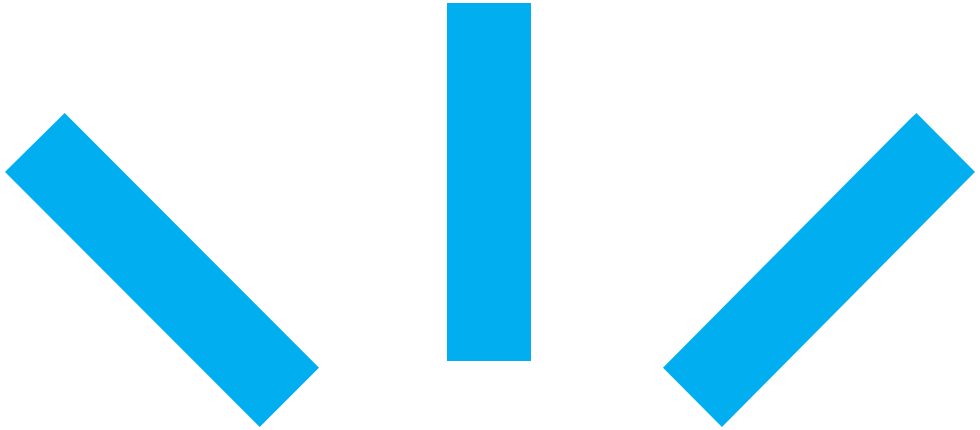
Summary: Ethics in Data Mining

- **When you're working with a dataset, you're seldom the only one with a legitimate interest in the data; data ethics is about knowing who the other stakeholders are and respecting their rights**
- **Ethical misconduct may take place at any stage of the work**
  - Collecting data
  - Analysing data
  - Sharing data or analysis results
- **Particular care must be taken when handling a dataset that contains personally identifiable or otherwise sensitive information**

Further Reading

- **Computer ethics:**

- Deborah G. Johnson, *Computer Ethics* (Pearson)
- Giannis Stamatellos, *Computer Ethics: A Global Perspective* (Jones & Bartlett Learning)
- Robert N. Barger, *Computer Ethics: A Case-Based Approach* (Cambridge University Press)



Information Security and Privacy

What Is Information Security?

- The public image of information security is somewhat distorted: hackers and malware programmers get most of the press
- Sure, hacking and malware are threats to be taken seriously, but not at the cost of neglecting to address other, more mundane risks
- To understand how to protect your data, let us first take a look at what it needs to be protected from

The Big Three

The main aspects of information security are summarised by the CIA triad: confidentiality, integrity, availability

Confidentiality: the data cannot be accessed by anyone not authorised to

Integrity: the data cannot be accidentally or maliciously deleted or corrupted

Availability: an authorised user cannot be prevented from accessing the data

Information Security Threat Examples

- **Hardware or software failure**
  - E.g. disk drive crash, virus infection
- **Negligent or incompetent administration**
  - E.g. inappropriate access privileges, obsolete system software
- **Accidental misbehaviour by authorised user**
  - E.g. accidental data deletion
- **Malicious activity by authorised user**
  - E.g. deliberate data leakage
- **Malicious activity by unauthorised user**
  - E.g. denial-of-service attack, eavesdropping on communications

Not Your Job, or Is It?

- Security is everyone’s job, but there are things you have no control over unless you’re the system administrator, which you probably aren’t
- If you’re using data storage facilities provided by your employer, things such as firewalls and regular backups are (hopefully) taken care of for you
- If not, your best bet is to choose a third-party storage service known to be reliable

Things You Can, and Should, Do

- It doesn't hurt to **keep redundant backups**, if you have somewhere to store them
- If your data is particularly sensitive, you may want to consider **encrypting** it, especially if you're using a third-party service to store it
- If you're in charge of a database with multiple users, **learn the access privilege scheme** of your database management system and use it – **everyone should have precisely those privileges they need, no more**
- Use strong passwords, especially for the root user!

The Concept of Information Privacy

- **Information privacy is about who is entitled to view personally identifiable information concerning a given individual**
- **Basic principle: each individual should have the power to determine for themselves what data is recorded about them and who is allowed to see it**
  - Exemptions are made for public authorities
- **This particularly holds for sensitive information such as medical records**

Classical vs Information Privacy

- Privacy in the classical sense is essentially freedom from surveillance, ”being home alone with the curtains closed”
- Information privacy is more complex; it’s not something you either have or don’t have at a given point in time
- Leaving information traces of yourself is an inevitable part of living in a modern society; information privacy is your ability to control where you leave such traces, what information they contain and how long they last

Threats to Information Privacy

- **Total information privacy is not attainable in practice; there are a number of deteriorating factors, only some of which an individual can affect**
- **Examples:**

- Government-mandated data collection
- Illegitimate data collection, data theft
- Unethical handling of legitimately obtained data
- Indifference concerning one's own privacy

We Do It to Ourselves

- Again, the most ”popular” threats are not necessarily the ones you should be worrying about the most
- Big Brother laws are the subject of hot debate, but in the meantime, people are quite happy to let various businesses track them in exchange for discounts and services
- Most people either don’t realise the implications or don’t consider them severe enough to give up all those nice perks

Data Mining as a Privacy Risk

- Consensually trading personal information for something else is not necessarily a problem; how much people value their own privacy is ultimately their own business
- What makes this a potential problem is that data mining challenges the traditional idea of informed consent
- If even the people doing the mining don’t have a clear idea of what they may find out, how can they inform you properly?

Profiling with Proficiency

- **If you have collected a large quantity of data representing an individual's activities, with data mining you can achieve a sometimes unnervingly detailed and accurate profile of them**
- You can find out their preferences with respect to products, services, politics etc., which is why this sort of data is gold to anyone hoping to sell you something, whether it's a phone or a presidential candidate
- **The profile may include some extremely personal information such as pregnancy, which may lead to some rather awkward situations if you haven't told your family yet**

They Know Who You Are

- Even if you think you've done your best to safeguard the privacy of your subjects, accidents can happen
- A dataset that contains no personally identifiable information may still reveal facts that, when combined with some additional information, can be used to identify an individual subject
- Simple example: from an anonymous dataset of locations, an individual subject's approximate home address and employer can be deduced; employee names can be retrieved from the employer's website and cross-referenced with a public phone directory

Got Something to Hide?

- **Because of the special nature of data mining, even if you won't publish your data, you may need to hide parts of it**
- **Standard practice is to replace identifying information with randomly or sequentially generated identifiers that cannot be traced to the person**
- Remember that data leakage is always a possibility, so anonymisation is also a pre-emptive form of damage control – if you need to keep the identifying information at all, store it separately
- **Proper anonymisation is generally required to get an ethical permit, and also necessary to get consent from the individuals concerned**
- It's not enough that you trust yourself to not abuse what you learn from the data – you need to gain the trust of others

Balancing Act

- Besides removing direct identifiers, it may be necessary to reduce the information content of background variables that could, when suitably combined, be used to identify an individual
- This does not necessarily mean removing the risky variables altogether; often it is enough to substitute classes for exact values
- For example, instead of giving the exact age of each individual, an age class representing a ten-year range could be used
- Each class should have a reasonably large number of members, or the anonymising effect may be compromised

Anonymising Qualitative Data

- If your data contains free-text fields (e.g. questionnaire answers), you need to pay attention to their content – there may be identifying information
- If a person's name is mentioned, it can be redacted altogether or replaced with a non-identifying characterisation (e.g. "Lauri Tuovinen" → "a researcher from Oulu")
- This is particularly important because the information may concern someone who isn't even part of the study
- The Finnish Social Science Data Archive (Tietoarkisto) has a good info package on anonymising both quantitative and qualitative data

Advanced Data Modification

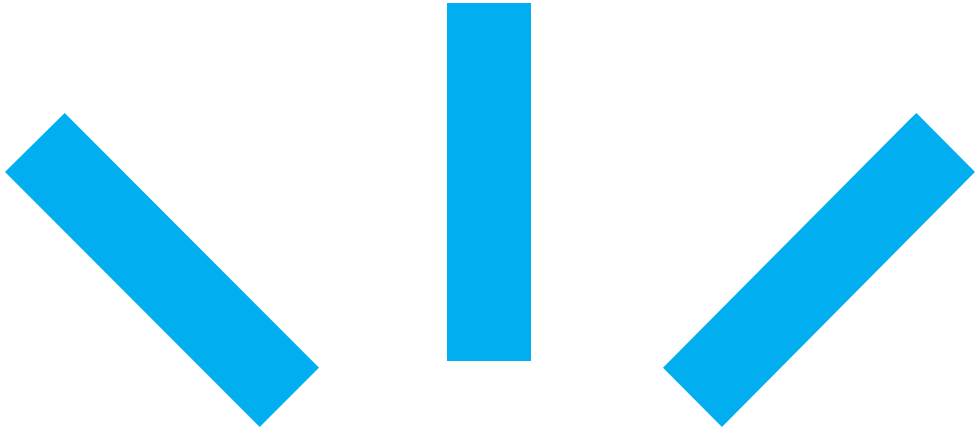
- The anonymisation techniques discussed above pertain also to studies using more traditional analysis methods
- Sometimes it is necessary to employ more advances techniques designed specifically to address the privacy risks of data mining
- The goal is to obfuscate the dataset on the level of individuals without distorting its properties on higher levels of abstraction

Privacy Preservation Methods

- **Heuristic-based selective data blocking or perturbation**
  - Estimate the likelihood of a given variable proving sensitive, hide or modify the most likely candidates
  - Hiding may also be applied to the mining results, e.g. association rules
- **Reconstruction of distributions at an aggregate level**
  - Effectively destroy the data at the individual level while preserving some of its utility
- **Cryptography-based techniques for secure multiparty computations**
  - Allows pooling data for mining without revealing your data to the other parties

Summary: Security and Privacy

- Securing data means protecting its confidentiality, integrity and availability
- Privacy is related to confidentiality, but even if the confidentiality of a dataset is preserved, privacy may still be violated internally
- To prevent disclosure of sensitive details to anyone – even legitimate users of the data – datasets containing personal information are rendered anonymous
- To achieve proper anonymisation, it is often necessary to employ a more elaborate approach than simply removing trivial identifiers such as names



Open Data – Using and producing

Defining Open Data

- **The Open Definition by Open Knowledge International:**
  - ”Open means anyone can freely access, use, modify, and share for any purpose (subject, at most, to requirements that preserve provenance and openness)”
- **Open Data Institute:**
  - ”Open data is data that anyone can access, use or share”

What Does It Mean?

- **Trivially, the data should be freely accessible somewhere – not hidden behind a paywall or anything**
  - You may be required to register as a user, though
- **You can share it, but you may be obliged to keep it open and tell everyone where it originally came from**
- **It should come with sufficient metadata to enable anyone to use it meaningfully**

Open Data as a Service

- Some open datasets are available through an online query API instead of or in addition to being downloadable
- This is particularly useful for data that changes in real time, enabling the development of third-party applications
- Classic example: route planners and other apps based on open public transport data (e.g. digitransit.fi, rata.digitraffic.fi)

Sharing Is Caring

- **The most obvious benefit of open data is that it allows more overall value to be extracted from a given dataset**
  - Unless the original producer has already exhausted all of its potential, others can take over
- **From the user's perspective, using open data saves resources that would otherwise have to be spent on collecting data or obtaining access to non-open datasets**
- **For the producer of the data, the benefit is more abstract, but especially if the data collection is publicly funded, it's often in the interest of the funder to ensure openness**

A Scientific Perspective

- Besides the obvious one, openness also has a more specific benefit pertaining to scientific research
- For peer review to work, the reviewers should have access to as much information about your research as possible – including your data
- The most important parts you include in your paper, but generally it's not feasible to include everything
- This doesn't mean opening up your entire dataset to the whole world – just enough to facilitate proper reviewing of your work

FAIR Principles

Findable

Accessible

Interoprable

Re-usable

https://www.go-fair.org/fair-principles/

https://www.force11.org/group/fairgroup/fairprinciples

https://fairdata.fi/en

Losing Control

- The main argument against making your data open is that once it's out there, you can no longer (fully) control who gets to use it and how
- From a corporate perspective, this may mean losing business opportunities or giving them to competitors, so if the data collection was privately funded, this can be a strong incentive to keep the data under lock and key
- In a sense, lack of control is also an argument against using open data – you're stuck with whatever the original owner has chosen to release, whether or not it suits your needs

Open Data Policies

- **Funding agencies increasingly encourage – or even expect – their grantees to release their research material as open data**
  - Besides the policy of your funder, check that of your employer, as well as any agreements pertaining to the specific project in which the data was obtained
- **You don’t want to shoot yourself in the foot by releasing too early, but if you sit on a dataset you’re not using, you’re letting a potentially valuable resource go to waste**

Academy of Finland Policy

- Project proposals must include a data management plan, including a description of how the research data obtained in the project will be made available to other researchers
- The Academy recommends that principal investigators use major public research data archives to store and distribute their data
- See, for example, the Open Science and Research Initiative of the Ministry of Education and Culture: http://openscience.fi

University of Oulu Policy

- The university has a research data policy that recommends registration and storage of digital datasets in relevant databanks
- The rights of researchers (as well as other stakeholders) to protect their interests are acknowledged
- The university will provide counseling for researchers preparing proposals on how to write the data management plan and budget the costs of data management

Open Data Licensing

- The license of an open dataset is what determines the terms and conditions that apply to using it
- It’s possible to write the license terms yourself, but this can be quite tedious if you want to get all the legalese exactly right
- Therefore it’s usually preferable to opt for an existing licensing scheme

Licensing Schemes

- **The Creative Commons (CC) licensing framework is popular, for obvious reasons: it's convenient and easy to understand**
  - Public domain (CC0): "no rights reserved"
  - Attribution (CC BY): distribution and derivative works permitted, as long as the original creator is credited
  - Modifiers to the basic BY license: ShareAlike (SA), NoDerivs (ND), NonCommercial (NC)
- **Open Data Commons offers an alternative scheme, the Open Database License (ODbL)**
  - Designed specifically for open data – distinguishes between the database and its contents, enables a different license to be applied to the contents

Using Open Data

- **The first step is the big one: finding the dataset that's right for you**
  - Has the information you want
  - Large enough quantity
  - In a format you can work with
  - Sufficiently permissive license
  - Etc.
- **Once you've located a dataset that suits you, getting it may be as simple as clicking on a download link**
- **After that, you proceed as you would if you had collected the data yourself**
  - Some extra effort may be required in preparing it, e.g. converting it to a more convenient representation format

Where to Find It

- Unless you know the URL for a specific dataset, you'll need a search engine – you can try Google, but there are also more specialised ones available

  https://www.google.com/publicdata/directory

- Avoindata.fi (https://www.avoindata.fi) stores and distributes open datasets released by various organisations, mainly public sector
- Tietoarkisto (http://www.fsd.uta.fi) is an archival service for social science datasets
- Etsin (http://etsin.avointiede.fi) searches for research datasets based on metadata shared by the data owners; the actual data may or may not be open

Things to Remember

- Open data is not public domain unless explicitly marked as such; keep the license term in mind and respect them
- Remember that the dataset as a whole and individual elements of it may have different owners and licenses – e.g. a dataset consisting of photographs
- Even if not explicitly required, it's common courtesy to give proper credit when using someone else's data in a publication; many open datasets come with a convenient identifier that you can use to cite the data

Contributing Open Datasets

- **When you've made the decision to publish your dataset, the next step is to prepare it for publication**
- **The point is to ensure that the dataset can be discovered and used by those who may find it useful**
- **Don't make too many assumptions about who is going to use the data and how – choose a distribution format that enables the data to be imported into a wide variety of tools, either directly or via conversion**
- Standard formats are good, open formats are better – steer clear of obscure proprietary formats
- **Above all, use plenty of metadata to tell everyone what the dataset contains and how it's structured**

Openness Is Not a Binary Thing

- Judge the right time to release your data – it doesn’t have to go public the minute you have it
- Making a dataset open doesn’t necessarily mean releasing it in its entirety – you may need to redact sensitive details
- Choose a license that represents your values and those of other stakeholders; for example, a share-alike license if you intend that all derivative works should also be open

Where to Publish

- You can use, for example, your organisation's website or a general-purpose file hosting service
- If you're working for a Finnish university or with funding from the Academy of Finland, IDA (http://avointiede.fi/ida) is a good option
- Exactly where the data is hosted is not terribly important; what matters is that it is discoverable, so submit your metadata to relevant indexing services such as Avoindata.fi and Etsin
- Avoindata.fi also provides hosting for open datasets (max. 150 MB)

Things to Remember

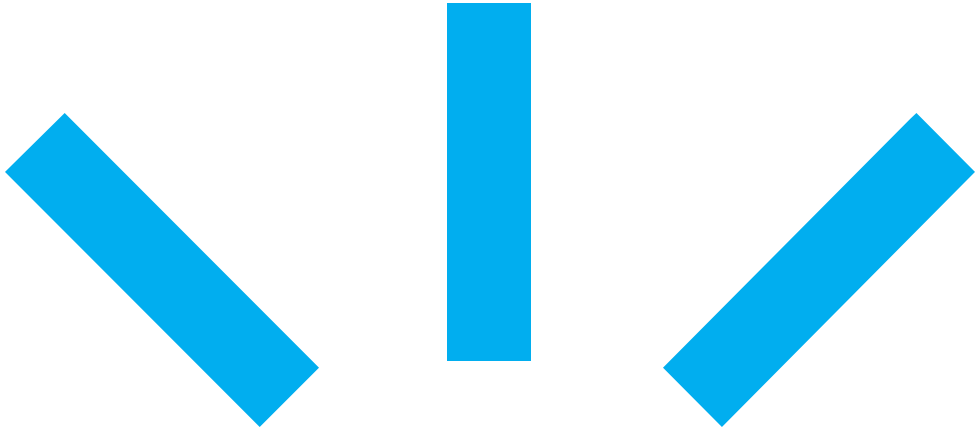
- Before you publish, make sure to clear everything with everyone who has a say in what may or may not be done with the data
- Familiarise yourself with the quality criteria of the hosting/indexing services you're planning to use and see to it that your data and metadata satisfy them
- Choose a license that you're sure you can live with – once the data has been released, you can't just change the terms at will

Summary: Open Data

- Publishing a dataset as open data is a way to ensure that its full potential gets utilised
- As a user, your main problem is finding a dataset that suits your purposes in every way; there are services that can help you do this
- As a producer, you need to weigh your own interests against the common good; you may want to finish your own work before sharing your data with others
- When publishing data, it is important to choose an appropriate license and to provide sufficient metadata to enable others to find and understand your dataset

Further Reading

- Rob Kitchin, *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences* (SAGE Publications)



Data Management and Databases

Why Data Management?

- **Data management is *everything you to do ensure that your data is available to you when you need to process it***
  - Having the data stored persistently
  - Knowing what data you have and what it is about
  - Being able to extract the specific subset you need
  - Being able to use the data in your tool of choice
- **Choosing the right data management solution for your purposes can help you a great deal with your research**

Aspects of Data Management

- **Persistent storage**
  - Physical storage media
  - Software platform
- **Curation and utilisation**
  - Procedures, interfaces for adding new data
  - Procedures, interfaces for retrieving data
- **Security**
  - Assurance of availability and integrity
  - Protection against illicit access
- **Indexing and metadata**

Metadata: Data about Data

- **In order to make sense of your data, you need to be aware of its *syntax* (representation) and *semantics* (meaning)**
- **Database management systems can enforce some basic rules concerning representation, but they cannot ensure that the data is meaningful**
- **Having good metadata is especially important if you're going to transfer your data to someone else – things you take for granted may not be so obvious to them**
- If available, use a standard representation format appropriate for the type of data you have, such as the DDI standard for data collected by surveys

Data Management Options

- **If you have a single dataset that can be easily stored in a single file, doing everything manually is a perfectly feasible option**
- **However, there are many situations where this soon gets pretty inconvenient**
  - Not wanting to import all of the data into memory at once
  - Having multiple data files that you need to cross-reference
  - Working with more complex data types than simple numbers or strings

Data Management Options (cont.)

- **A wide variety of database management systems are available to reduce your burden**
- **Common features:**
  - Ensuring integrity and consistency of data
  - A query language for manipulating data and extracting subsets
  - APIs/drivers for accessing the database from programming languages and data analysis tools
- **One size does not fit all – we will look at the pros and cons of some popular choices**

Plaintext Data Files

- Or CSV files, or whatever you want to call them – files with records (rows) separated by newlines and fields (columns) separated by tabs or commas (usually)
- Dead simple and completely portable – whatever tools you're using, the odds are you can export and import these
- However, these benefits are soon lost if your dataset has a more complex internal structure

Other File Formats

- **Examples:**
  - XML documents
  - Excel spreadsheets
  - MATLAB .mat files
  - SPSS .sav files
- **These enable you to express more complex data structures, generally at the cost of being less simple to generate and less portable**
  - Otherwise, from a data management perspective, they are equivalent to CSV

Example: Activity Monitoring

- As long as the data file contains only the actual activity data, no problem

Date	Active minutes	Steps	Calories
2016-08-01	67	7922	2775
2016-08-02	61	8217	2854
2016-08-03	28	5166	2742
2016-08-04	15	3657	2511
2016-08-05	37	6943	2873

Example: Activity Monitoring

- What happens, though, if you also want to record information about the people monitored? Maybe also the devices used?

Name	Age	Device	Date	Active minutes	Steps	Calories
Melissa Schulz	24	Fitbit Flex	2016-08-01	67	7922	2775
Melissa Schulz	24	Fitbit Flex	2016-08-02	61	8217	2854
Melissa Schulz	24	Fitbit Flex	2016-08-03	28	5166	2742
Melissa Schulz	24	Fitbit Flex	2016-08-04	15	3657	2511
Melissa Schulz	24	Fitbit Flex	2016-08-05	37	6943	2873

Relational Databases

- A relational database is a *collection of relational tables*
- Formally, a relation is a set of *tuples*; a tuple is an ordered finite sequence of elements (variable values)
- Informally, a relational table is like a data file: the tuples represent the rows and the tuple elements the columns

Why Use One Then?

- **A relational database system provides a powerful query engine for selecting and extracting the specific combination of rows and columns that you currently need**
- **Query examples:**

- Select rows from a table based on the value of a given column
- Join two tables based on a column signifying the same thing in both tables
- Group table rows base on a column value and compute an aggregate value for each group

Querying Relational Databases

- **The lingua franca of relational databases is SQL – all relational systems implement it**
  - In theory, you could drop in any relational system to replace any other without modifying your queries, although in practice it's not quite that simple
- **SQL is a *functional language*: you only need to tell the query engine what output you want, and the engine will figure out the most efficient way to generate it**

Example: Activity Monitoring

- **Now, instead of storing all your data in a single file, we can have one relational table for the activity measurements, one for the people and one for the monitoring devices**
  - Special key attributes link each activity record to the corresponding person and device
- **If we want, for example, the measurements for a specific person using a specific device on a specific day, we can achieve that by querying a join of all three tables**

Example: Activity Monitoring

personId	firstName	lastName	gender	age
1	Andrew	Parkinson	male	28
2	Melissa	Schulz	female	24
3	Joe	Chang	male	41

monitorId	brand	model
1	Polar	Loop
2	Fitbit	Flex

personId	monitorId	measurementDate	activeMinutes	steps	calories
1	1	2016-08-01	26	2964	1069
2	2	2016-08-01	67	7922	2775
3	1	2016-08-01	75	5763	2583
1	1	2016-08-02	11	3895	2548
2	2	2016-08-02	61	6638	2854

Still Going Strong

- The RDB concept was proposed in the 70s and became the dominant type of database in the world in the 80s – it probably still is
- Popular systems include Oracle Database, Microsoft SQL Server, IBM DB2, MySQL/MariaDB, PostgreSQL, SQLite
- Relational is the default choice if your data conforms easily to a predefined schema; if not, you may want to consider a NoSQL database instead

NoSQL Databases

- A loosely defined group of database systems, characterised mainly by the underlying data model not being (strictly) relational
- Technically, the first databases were NoSQL since the relational model did not exist at the time, but the term is considerably more recent
- Popular types include document stores, column stores and key-value stores

MongoDB

- **One of the most popular NoSQL database management systems today, used by Facebook, eBay, CERN, Expedia, SAP, Ticketmaster and many others**
- **Stores data in the form of BSON (”binary JSON”) documents**
  - JSON = JavaScript Object Notation, a language-independent data interchange format based on the JavaScript programming language
- **Has its own query language, with SQL-like capabilities**

Example: Activity Monitoring

```
{
  '_id' : 29,
  'personId' : 3,
  'monitorId' : 1,
  'date' : '2016-08-01',
  'activeMinutes' : 75,
  'steps' : 5763,
  'calories' : 2583
}
```

```
{
  '_id' : 3,
  'firstName' : 'Joe',
  'lastName' : 'Chang',
  'gender' : 'male',
  'age' : 41
}
```

```
{
  '_id' : 1,
  'brand' : 'Polar',
  'model' : 'Loop',
}
```

MongoDB vs Relational Databases

- **A BSON document is like a row in a relational table: a set of field-value pairs**
  - Compound values are permitted – the value of a field can be an array or an embedded document
- **A collection of BSON documents is like a relational table**
  - No fixed schema – any document can be inserted into any collection, whether or not it has the same structure as the other documents in the same collection

Pros and Cons

- **On the plus side:**
  - Flexibility of the database schema and data types
  - More natural mapping between programming language objects and database records
  - Scalability and distributability
- **On the minus side:**
  - Less strict consistency guarantees
  - No standardisation – you're pretty much stuck with the system you've chosen, whereas relational systems are more or less interchangeable

Wait, There's More!

- **Object databases, for an even more direct mapping between an object-oriented language and a database**
  - E.g. Objectivity/DB
- **XML document stores, for XML-formatted data**
  - E.g. BaseX
- **Big data stores, for very large quantities of sparse data**
  - E.g. Apache HBase

Direct vs Indirect

- Data files can generally be imported and exported directly (although a format conversion may be required)
- With databases, data can be imported and exported directly if there's a suitable database connector available for the data mining tool
- An alternative is to import and export indirectly via an intermediate container that both the data mining tool and the database can read from and write to

- *Importing*: reading a dataset from persistent storage into a data mining tool
- *Exporting*: writing a dataset from data mining tool into persistent storage
- Persistent storage may refer to either a data file or a database

Import/Export with Data Files

- **Virtually all data analysis tools provide functions for importing and exporting CSV-style data files**
  - MATLAB: `readtable`, `writetable`
  - R: `read.table`, `write.table`
- **For other widely used formats there is typically a conversion available**
  - For example, data in Excel format can be converted to CSV by opening the file in Excel and saving it in CSV format
  - Note that this may result in surprises concerning the formatting of certain types of values – be careful with things like decimal and thousands separators

Data Files as Intermediate Containers

- Many database systems similarly provide functions for importing and exporting CSV files, so you can use them as containers for indirect import/export
- For example, in MySQL you can use the `SELECT INTO OUTFILE` statement to export data from a table (or tables) into a file that you can then import into a data mining tools
- To import data from a file into MySQL, you can use the `LOAD DATA INFILE` statement
- These are not universal, even among relational databases – if you're using a different system, these (probably) won't work

Database Connectors

- **To access a database directly from a data mining tool, you need a suitable database connector or driver**
- **Relational databases being as popular as they are, any tool that you can take at all seriously is likely to have connectors for them available**
  - If there isn't one for the specific product you're using, there probably is for ODBC, which can be used to access most relationals (and with the right kind of driver, even data files)
- **With NoSQL databases, you may or may not be in luck, depending on which combination of database and data mining tool you're using**

Direct Import/Export with MySQL

- **MATLAB:**
  - The Database Toolbox provides both a graphical app and a command-line interface for accessing relational databases
  - Other options available on MathWorks File Exchange – search for MySQL
- **R:**
  - The DBI package defines a generic interface for relational database access, and the RMySQL package implements it for MySQL and MariaDB
  - You can find all the necessary packages on CRAN

Direct Import/Export with MongoDB

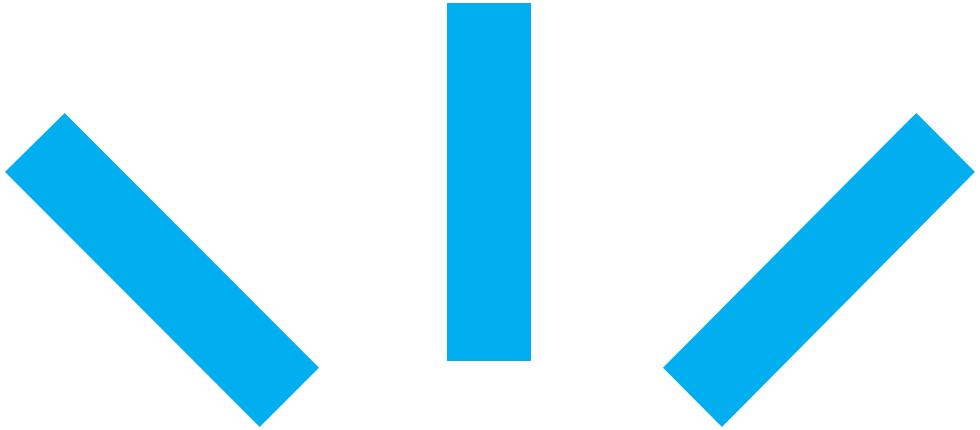
- **MATLAB:**
  - The MongoDB Java driver can be used, MATLAB as a built-in Java Virtual Machine
  - Alternatively, try MongoMatlabDriver, built on top of the MongoDB C driver, available on GitHub
- **R:**
  - The rmongodb package available on CRAN provides an interface to MongoDB using the C driver

Summary: Data Management Basics

- **Data management comprises the activities done to ensure that you can access the data you need, when you need it**
- **The most important decision you need to make is how you are going to represent and store your data**
  - Metadata is also important, especially if the data will be used by others
- **File-based data management is a possibility, but database management systems offer many beneficial functionalities**

Further Reading

- C. J. Date, *An Introduction to Database Systems* (Addison-Wesley)
- Avi Silberschatz, Henry F. Korth & S. Sudarshan, *Database System Concepts* (McGraw-Hill)
- Ramez Elmasri & Shamkant B. Navathe, *Fundamentals of Database Systems* (Pearson)
- David M. Kroenke & David J. Auer, *Database Processing: Fundamentals, Design, and Implementation* (Pearson)



Relational data modeling and a crash course in sql

What Is a Data Model?

- A data model is an abstract representation of an application domain – more specifically, a representation of data to be processed and stored
- A static model: does not say anything about the behaviour of the application
- Also does not say anything about what technology will be used to implement the data store

Reasons for Data Modeling

- **By first creating a data model, you can start designing your database without committing yourself to any specific type of database system**
- The modeling process may help you understand which type would best suit your purposes
- **Visualising the domain is useful for understanding relationships among domain concepts, and for spotting any omissions you may have made**
- **If you're working as a team, models are always a great way to establish and communicate a shared understanding of whatever it is you're working on**

Entity-Relationship Modeling

- **Entity-relationship (ER) notation is a graphical language for designing the data model**
- **An ER model consists of:**
  - *Entities*: principal concepts in the application domain
  - *Relationships*: named connections among entities
  - *Attributes*: properties of entities or relationships
- **With extended ER (EER) notation, object-oriented concepts such as inheritance can be represented**

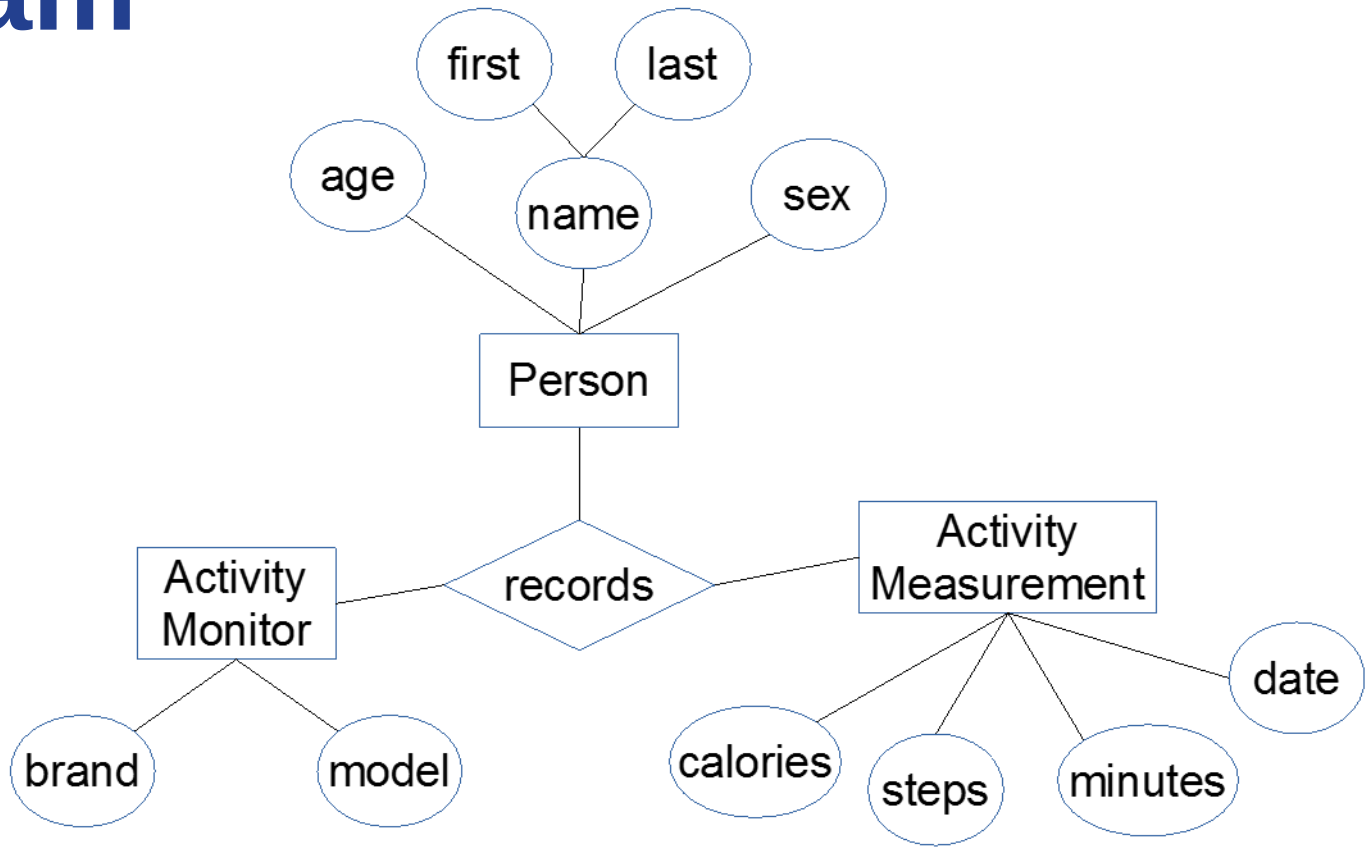
ER Notation

- **An ER model is usually drawn as a diagram, using the following notation**
  - Boxes denote entities
  - Ellipses denote attributes
  - Diamond shapes denote relationships
  - Line segments connect associated shapes
  - Strings and numbers attached to relationships denote roles and cardinalities
- **Although ER notation is typically used for relational modeling, it's not exclusive to relational databases**

Example: Activity Monitoring

- **We can represent our previous example using three entities:**
  - Person, with attributes *first name*, *last name*, *gender* and *age*
  - Activity Monitor, with attributes *brand* and *model*
  - Activity Measurement, with attributes *date*, *active minutes*, *steps* and *calories*
- **A three-way relationship connects the entities**

Simple ER Diagram



Converting an ER Model into Tables

- **There is a fairly straightforward mapping between ER model elements and relational tables:**
  - Entities become tables
  - Attributes become table columns
  - Relationships become either foreign keys or separate tables
- **In the activity monitoring example, we get three tables; the relationship is represented by foreign keys in the ActivityMeasurement table**

Table Specifications

Person	
personId	int unsigned
firstName	varchar
lastName	varchar
gender	enum(’male’, ’female’)
age	tinyint unsigned

ActivityMonitor	
monitorId	int unsigned
brand	varchar
model	varchar

ActivityMeasurement	
measurementId	int unsigned
personId	int unsigned
monitorId	int unsigned
measurementDate	date
activeMinutes	int unsigned
steps	int unsigned
calories	int unsigned

Summary: Relational Data Modeling

- Data modeling can be used to create an abstract design for a database before making decisions about implementation
- A commonly used graphical language in data modeling is entity-relationship (ER) notation
- Once the data model has been created using ER, it can be translated into specifications for relational database tables by a fairly straightforward procedure

Relational Algebra

- Recall that mathematically, relational tables are sets of tuples
- Relational algebra defines operations on these sets; these include standard set theoretical operations, but also ones that are specific to relations
- Relational algebra, as proposed by E. F. Codd in 1970, forms the mathematical basis of SQL queries

Operators in Relational Algebra

- **When it comes to understanding how SQL works, four operators get you pretty far**
- ***Projection*: extracting a subset of the attributes (columns) of a relation**
- ***Selection* (or *restriction*): extracting a subset of the tuples (rows) of a relation**
- ***Rename*: giving a relation or attribute a different name**
- ***Natural join*: merging tuples from two different relations having common attributes**
  - The result is a new relation where each tuple is formed by joining tuples from the operand relations based on the common attributes having the same values

The `SELECT` Statement

- **The most basic query of all: `SELECT * FROM [table];`**

- Returns all the data stored in `[table]`
- The `*` is shorthand for every column in `[table]`
- You can change the ordering of results by appending `ORDER BY [column] [ASC | DESC]`
- `;` terminates the query string and tells the query processor to start parsing

Example

- `SELECT * FROM Person ORDER BY lastName ASC, firstName ASC;`

personId	firstName	lastName	gender	age
3	Joe	Chang	male	41
6	Stephanie	Greene	female	37
4	Rosita	Martinez	female	33
5	Sean	McMillan	male	30
1	Andrew	Parkinson	male	28
2	Melissa	Schulz	female	24

Projection

- `SELECT [column1], [column2], … , [columnN] FROM [table];`
- Returns all the data in `[table]`, but only for the columns named

Example

- `SELECT firstName, lastName, age FROM Person;`

personId	firstName	lastName	gender	age
1	Andrew	Parkinson	male	28
2	Melissa	Schulz	female	24
3	Joe	Chang	male	41
4	Rosita	Martinez	female	33
5	Sean	McMillan	male	30
6	Stephanie	Greene	female	37

Selection

- `SELECT * FROM [table] WHERE [expression];`
- Returns only those tuples from `[table]` for which the conditions specified by `[expression]` hold

Example

- `SELECT * FROM Person WHERE gender='male' AND age<=30;`

personId	firstName	lastName	gender	age
1	Andrew	Parkinson	male	28
2	Melissa	Schulz	female	24
3	Joe	Chang	male	41
4	Rosita	Martinez	female	33
5	Sean	McMillan	male	30
6	Stephanie	Greene	female	37

Building Selection Conditions

- **Arithmetic operators: `+`, `-`, `*`, `/`, `DIV`, `MOD`**
- **Comparison operators: `=`, `<`, `<=`, `>`, `>=`, `<>`**
- **Pattern matching operators: `LIKE`, `REGEXP`**
  - `LIKE` for simple wildcard matching, `REGEXP` for more complex patterns
- **Logical operators: `AND`, `OR`, `XOR`, `NOT`**
  - Use to build compound expressions of arbitrary complexity

Combining Projection and Selection

- `SELECT firstName, lastName, age FROM Person WHERE gender='male' AND age<=30;`

personId	firstName	lastName	gender	age
1	Andrew	Parkinson	male	28
2	Melissa	Schulz	female	24
3	Joe	Chang	male	41
4	Rosita	Martinez	female	33
5	Sean	McMillan	male	30
6	Stephanie	Greene	female	37

Multi-Table Queries (Joins)

- **The `FROM` clause may specify an arbitrary number of tables**
- **`SELECT * FROM [table1], [table2], … , [tableN] WHERE [expression]`**

- The tables are joined by using foreign key columns in the selection conditions
- You can give the tables shorthand names to make the query string easier to write

Example

- `SELECT measurementDate, steps FROM ActivityMeasurement A, Person P WHERE A.personId = P.personId AND lastName='Martinez';`

measurementDate	steps
2016-08-01	4535
2016-08-02	9715
2016-08-03	11270
2016-08-04	8388
2016-08-05	9483

Alternative: Inner Join

- **`SELECT … FROM [table1] INNER JOIN [table2] ON [table1].[column1]= [table2].[column2] WHERE … ;`**
- Equivalent to `SELECT … FROM [table1], [table2] WHERE [table1].[column1]= [table2].[column2] AND … ;`
- Different syntax, same operation – there should be no difference in performance, choosing one or the other is a matter of personal preference
- **There are other types of joins as well, you can learn about them yourself if you're interested**

Aggregation

- **`SELECT [column], [expression] FROM [table] GROUP BY [column];`**
- Groups the rows in `[table]` by the value of `[column]` and computes the value of `[expression]` for each group
- A number of statistical functions are available for [expression]: `COUNT`, `SUM`, `AVG`, `MIN`, `MAX` etc.

Example

- **`SELECT gender, COUNT(*) FROM Person GROUP BY gender;`**
- `SELECT COUNT(*) FROM Person;` without the `GROUP BY` clause returns the total row count of the table

gender	count(*)
male	3
female	3

Subqueries

- A `SELECT` query can be nested within another query by placing it in parentheses
- The `IN` operator tests for set membership; it's often useful with a subquery, to see whether a column value in one table occurs in the result set of a query on another table

Example

- ```
  SELECT firstName, lastName FROM
  Person WHERE personId IN (SELECT
  DISTINCT personId FROM
  ActivityMeasurement);
  ```

firstName	lastName
Andrew	Parkinson
Melissa	Schulz
Joe	Chang
Rosita	Martinez
Sean	McMillan
Stephanie	Greene
  ```

Other Subquery Operators

- **Standard comparison operators can be used to compare e.g. the value of a column to the (scalar) value of a subquery**
- **`ANY` and `ALL` are used in conjunction with a comparison operator to test whether the comparison is true for some or all of the values returned by the subquery**
  - `SOME`, an alias for `ANY`, is sometimes used, mainly in cases where it clarifies the meaning of the query
- **`EXISTS` and `NOT EXISTS` are used to test whether a subquery returns any rows**

Data Manipulation Statements

- Insert a new row: `INSERT INTO [table]([column1], … , [columnN]) VALUES([value1], … , [valueN]);`
- Change existing rows: `UPDATE [table] SET [column1]= [value1], … , [columnN]= [valueN] WHERE [expression];`
- Delete existing rows: `DELETE FROM [table] WHERE [expression];`

Creating a Table

```
- CREATE TABLE Person(
  personId INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  firstName VARCHAR(255),
  lastName VARCHAR(255),
  gender ENUM('male', 'female'),
  age TINYINT UNSIGNED
  );
```

Notes on `CREATE TABLE`

- `INT` is a 4-byte integer, `TINYINT` a single-byte one; `UNSIGNED` means only non-negative values are accepted
- `VARCHAR` is a string of variable length
- `ENUM` is a string that has a limited set of permitted values
- `PRIMARY KEY` means that each row must have a unique non-null value for the column
- `AUTO_INCREMENT` means the database engine will automatically generate values for the column

Other Data Definition Statements

- **For tables:**
  - `ALTER TABLE`
  - `RENAME TABLE`
  - `TRUNCATE TABLE`
  - `DROP TABLE`
- **For databases:**
  - `CREATE DATABASE`
  - `ALTER DATABASE`
  - `DROP DATABASE`
- **Similarly for functions, procedures, indexes etc.**

Summary: Introduction to SQL

- **SQL is the standard language for communicating with relational database management systems**
- **With SQL, you can select data to retrieve from a database, manipulate data and define data structures**
  - Data selection is based on algebraic operations on relations
- **As a functional language, SQL only expresses the output you want – the query engine takes care of the rest**

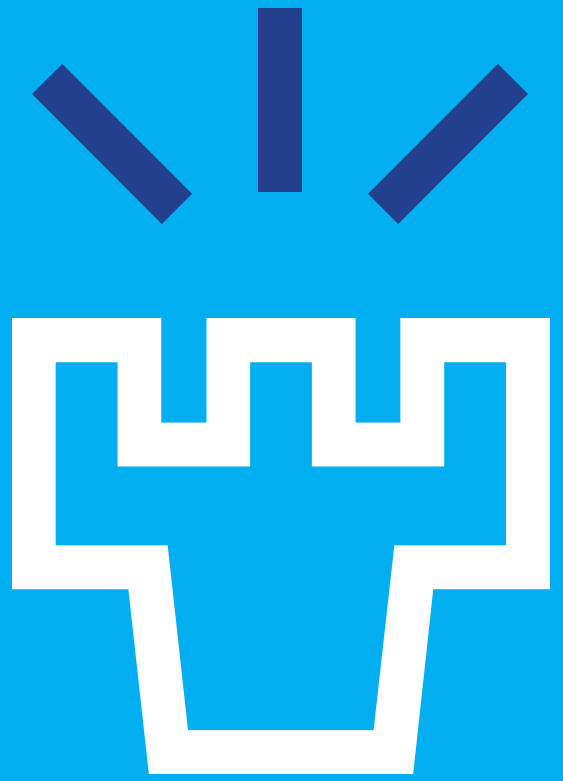
Next steps

This week

- **Exercise lab on Wednesday 12.15-14.00**
  - Open data concepts and practice
  - Data management using relational databases

Next week

- **Pre-class assignment for week 3 due next Monday 23:59:59**
- **Exercise submissions due next Tuesday 23:59:59**



UNIVERSITY
OF OULU