# Database Theory and Knowledge Representation

## 1st Lecture

David Carral

University of Montpellier

October 21, 2021

# INTRODUCTION

## Personal Information

- Name: David Carral
- Job Title: CRCN Researcher at Inria
- Research Team: GraphIK
- Email: david.carral@inria.fr
- Webpage: `https: //www-sop.inria.fr/members/David.Carral/`
- Photo:

## Contact, Questions, and Course Materials

- I will only teach 3 lectures: 14/11, 21/11, and 28/11

- You can ask me questions after each lecture or via email. If this is not enough, we can always arrange a meeting at the LIRMM:

    Salle 3.129 - Batîment 5
    Campus Saint Priest - Universite Montpellier
    860 Rue St - Priest, 34090 Montpellier

- I will upload all slides in Moodle and provide you with additional study materials (see the following slides).

Obtain an understanding of database theory with a focus on different query languages:

- Relational data model
- Query languages: first-order, conjunctive, and Datalog
- Expressive power of query languages
- Complexity of query answering

## Remark

We will not only discuss theoretical results but will also do some exercises to understand how to use the above languages.

1. Turing machines
2. First-order logic: syntax and semantics
3. Computational complexity

## Remark

I will reintroduce most of the concepts from computational complexity that are applied in this lecture (if necessary).

## Additional Materials

1. Database Theory:
   - ▶ *Foundations of Databases*
     Serge Abiteboul, Richard Hull, Victor Vianu
     Available at `http://webdam.inria.fr/Alice/`
   - ▶ *Database Theory* Course at TU Dresden
     Available at `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2020)/en`
2. Computational Complexity:
   - ▶ *Foundations of Complexity Theory* Course at TU Dresden
     Available at `https://iccl.inf.tu-dresden.de/web/Complexity_Theory_(WS2020)/en`
   - ▶ *Introduction to the Theory of Computation*
     Michael Sipser
   - ▶ *Computational Complexity: A Modern Approach*
     Boaz Barak and Sanjeev Arora

# Acknowledgements

This lecture is heavily inspired by the *Database Theory* lectures at TU Dresden, which were created by Prof. Markus Krötzsch:



## Additional links to lectures from previous years:

- 2016: `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2016)/en`
- 2018: `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2018)/en`
- 2019: `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2019)/en`

## (Tentative) Outline

1. What is a database?
2. The relational data model
3. Relational algebra queries
4. First-order queries
5. Conjunctive queries
6. Tree-like queries
7. Datalog queries

# 1. WHAT IS A DATABASE?

## WHAT IS A DATABASE?

A Database Management System (DBMS) is a software to manage collections of data.

⤳ highly important class of software systems
⤳ major role in industry and in research
⤳ extremely wide variety of concepts and implementations

General three-level architecture of DBMS:

- **External Level:** application-specific user views
- **Logical Level:** abstract data model, independent of implementation, conceptual view
- **Physical Level:** data structures and algorithms, platform-specific

In this lecture: focus on logical view for relational data model

Basic functionality of DBMS:

- **Schema definition:** specify how do we logically organise data
- **Update:** insert/delete/update stored data
- **Query:** retrieve stored data or information derived from it
- **Administration:** user rights management, configuration, recovery, data export, etc.

Many related concerns:

- **Persistence:** data retained when DBMS is shut down
- **Optimisation:** ensure maximal efficiency
- **Scalability:** cope with increasing loads by adding resources
- **Concurrency:** support update/query operations in parallel
- **Distribution:** combine data from several locations
- **Interfaces:** APIs, query languages, update languages, etc.

**In this lecture:** schema and query languages

# 2. The Relational Data Model

# Database = Collection of Tables

Lines:

| Line | Type |
|------|------|
| 85 | bus |
| 3 | tram |
| F1 | ferry |
| … | … |

Stops:

| SID | Stop | Accessible |
|-----|------|------------|
| 17 | St-Guilhem | true |
| 42 | Foch | true |
| 57 | Comédie | true |
| 123 | Av. Liberté | false |
| … | … | … |

Connect:

| From | To | Line |
|------|-----|------|
| 57 | 42 | 85 |
| 17 | 789 | 3 |
| … | … | … |

Every table has a schema:

- Lines[Line:string, Type:string]
- Stops[SID:int, Stop:string, Acc:bool]
- Connect[From:int, To:int, Line:string]

## Remark

Answering queries requires integrating info from different tables.

## Towards a Formal Definition of "Table"

A table row has one value for each column. Hence, a row can be represented by a function from attributes to values.

Example: The row

| SID | Stop | Accessible |
|-----|------|------------|
| ... | ... | ... |
| 42 | Comédie | true |
| ... | ... | ... |

can be represented by the function:

$$f : \{\mathsf{SID} \mapsto 42, \mathsf{Stop} \mapsto \texttt{"Comédie"}, \mathsf{Accessible} \mapsto \mathsf{true}\}$$

### Remark

The above is an "abstract data model" and not a "data structure". We focus in the logical level of the DBMS architecture.

# The Domain of a Database

In order to give a definition for a database instance, we need to introduce the universe of elements that appear in the tables.

## Definition: Domain

Let **dom** ("domain") be the (possibly infinite) set of conceivable values in tables.

## Remark

For simplicity, we drop the datatypes of database columns and assume that each column uses the same datatype that supports all values in **dom**.

# DATABASE = SET OF TABLES

## Definition: Named Perspective

- A relation schema $R[U]$ consists of a relation name $R$ and a finite set $U$ of attributes ($|U|$ is the arity of $R[U]$)
- A table for $R[U]$ is a finite set of functions from $U$ to **dom**
- A database instance $\mathcal{I}$ is a finite set of tables

## Remark

Note that we disregard the order and multiplicity of rows.

Under the "set of tables" perspective, tables are also called *relation instances*. The table with relation schema $R[U]$ in the database instance $\mathcal{I}$ is written $R^{\mathcal{I}}$.

## Example 2.5

Consider the database instance $\mathcal{I}$ that contains the tables:

Lines:

| Line | Type |
|------|------|
| 85   | bus  |
| 33   | tram |

Stops:

| SID | Stop       | Accessible |
|-----|------------|------------|
| 17  | St-Guilhem | true       |

Under the "Set of Tables" perspective, the database instance $\mathcal{I}$ is the set $\{\text{Lines}^{\mathcal{I}}, \text{Stops}^{\mathcal{I}}\}$ where:

$$\text{Lines}^{\mathcal{I}} = \{\{\textbf{Line} \mapsto 85, \textbf{Type} \mapsto \text{bus}\},$$
$$\{\textbf{Line} \mapsto 33, \textbf{Type} \mapsto \text{tram}\}\}$$
$$\text{Stops}^{\mathcal{I}} = \{\{\textbf{SID} \mapsto 17, \textbf{Stop} \mapsto \text{St-Guilhem}, \textbf{Accessible} \mapsto \text{true}\}\}$$

# DATABASE = SET OF FACTS

Another way to encode the rows in a database is using facts:

$$\text{Lines}(85, \texttt{"bus"}) \qquad \text{Stops}(42, \texttt{"Comédie"}, \text{true})$$
$$\text{Lines}(F1, \texttt{"ferry"})$$

## Definition: Unnamed Perspective

A fact is an expression $p(t_1, \ldots, t_n)$ where
- $p$ is an $n$-ary predicate symbol
- $t_1, \ldots, t_n$ are constant symbols

A database instance is a finite set of facts.

## Remark

Constant symbols in facts are elements of **dom**.

## Example

Consider the database instance $\mathcal{I}$ that contains the tables:

Lines:

| Line | Type |
|------|------|
| 85   | bus  |
| 33   | tram |

Stops:

| SID | Stop       | Accessible |
|-----|------------|------------|
| 17  | St-Guilhem | true       |

Under the "Set of Facts" perspective, the database instance $\mathcal{I}$ is the set of facts:
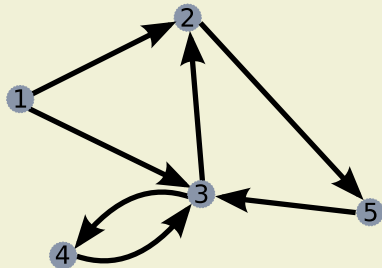
$$\mathcal{I} = \{\text{Lines}(85, \text{bus}), \text{Lines}(33, \text{tram}),$$
$$\text{Stops}(17, \text{St-Guilhem}, \text{true})\}$$

## Example

Binary relations (sets of pairs) can be viewed as directed graphs.

| Source | Target |
|--------|--------|
| 1 | 2 |
| 1 | 3 |
| 2 | 5 |
| 3 | 2 |
| 3 | 4 |
| 4 | 3 |
| 5 | 3 |



Many binary tables in one graph? Use table name to label edges!

What to do with tables of arity $\neq 2$?
$\rightsquigarrow$ Generalise graphs to <span style="color:red">hypergraphs</span>!

## Definition 2.10: Hypergraph Perspective

A hypergraph is a triple $\langle V, E, \rho \rangle$, where
- $V$ is a set of vertices
- $E$ is a set of edge names
- $\rho$ maps each edge name $e \in E$ to an $n$-ary relation $\rho(e) \subseteq V^n$

In other words: finite hypergraphs are databases!

We consider three different definitions for a database:

- A *set of tables*, which are sets of functions from the corresponding set of attributes to **dom**.
- A *set of facts* wherein the predicates are the table names and the constants are elements of **dom**.
- A *hypergraph* where hyperedges are labeled with table names and vertices are elements of **dom**.

# 3. Relational Algebra Queries

Query language based on a set of operations on databases.

Each operation takes one or more tables as input and produces another table as output

(we often simplify notation and write a table name rather than a table instance)

Main operations:

- Selection $\sigma$
- Projection $\pi$
- Join $\bowtie$
- Renaming $\delta$
- Difference $-$
- Union $\cup$
- Intersection $\cap$

### Remark

The RA is the theoretical language underpinning SQL.

## SELECTION

"Find all bus lines"

$$\sigma_{\text{Type}="bus"}\text{Lines}$$

"Find all connections that begin and end in the same stop"

$$\sigma_{\text{From}=\text{To}}\text{Connect}$$

### Definition 3.1: Selection Operator

The selection operator has the form $\sigma_{n=m}$

- $n$ is an attribute name
- $m$ is an attribute name or a constant value

Consider a table $R^{\mathcal{I}}$ for $R[U]$.

- For $m$ constant value: $\sigma_{n=m}(R^{\mathcal{I}}) = \{f \in R^{\mathcal{I}} \mid f(n) = m\}$
- For $m$ attribute name: $\sigma_{n=m}(R^{\mathcal{I}}) = \{f \in R^{\mathcal{I}} \mid f(n) = f(m)\}$

This is only defined if $U$ contains the required attribute names.

## PROJECTION

"Find all possible types of lines"

$$\pi_{\text{Type}}\text{Lines}$$

"Find all pairs of adjacent stops on line 85"

$$\pi_{\text{From,To}}(\sigma_{\text{Line}=\texttt{"85"}}\text{Connect})$$

### Definition 3.2: Projection Operator

The **projection operator** has the form $\pi_{a_1,\ldots,a_n}$ where each $a_i$ is an attribute name.

Consider a table $R^{\mathcal{I}}$ for $R[U]$.

$$\pi_{a_1,\ldots,a_n}(R^{\mathcal{I}}) = \{f_{\{a_1,\ldots,a_n\}} \mid f \in R^{\mathcal{I}}\}$$

where $f_{\{a_1,\ldots,a_n\}}$ is the restriction of $f$ to the domain $\{a_1,\ldots,a_n\}$; that is, the function $\{a_1 \mapsto f(a_1),\ldots,a_n \mapsto f(a_n)\}$. Of course, this projection is only defined if $a_i \in U$ for each $a_i$.

"Find all connections and their type of line":

Connect:

| From | To | Line |
|------|-----|------|
| 57 | 42 | 85 |
| 17 | 789 | 3 |
| … | … | … |

Lines:

| Line | Type |
|------|------|
| 85 | bus |
| 3 | tram |
| F1 | ferry |
| … | … |

Connect $\bowtie$ Lines:

| From | To | Line | Type |
|------|-----|------|------|
| 57 | 42 | 85 | bus |
| 17 | 789 | 3 | tram |
| … | … | … | … |

### Definition 3.3: Natural Join Operator

The natural join operator has the form $\bowtie$.

Consider tables $R^{\mathcal{I}}$ for $R[U]$ and $S^{\mathcal{I}}$ for $S[V]$.

$$R^{\mathcal{I}} \bowtie S^{\mathcal{I}} = \{f : U \cup V \to \mathbf{dom} \mid f_U \in R^{\mathcal{I}} \text{ and } f_V \in S^{\mathcal{I}}\}$$

where $f_U$ ($f_V$) is the restriction of $f$ to elements in $U$ ($V$) as before

"Find all lines that depart from an accessible stop"

Stops:

| SID | Stop | Accessible |
|-----|------|------------|
| 57 | Comédie | true |
| 123 | St-Guilhem | false |
| … | … | … |

Connect:

| From | To | Line |
|------|-----|------|
| 57 | 42 | 85 |
| 17 | 789 | 3 |
| … | … | … |

We need to join Stops.SID with Connect.From $\rightsquigarrow$ use renaming

$$\pi_{Line}\big(\sigma_{\text{Accessible}="\texttt{true}"}(\text{Stops} \bowtie \delta_{\text{From,To,Line}\rightarrow\text{SID,To,Line}}(\text{Connect}))\big)$$

## Definition 3.3: Renaming Operator

The renaming operator has the form $\delta_{a_1,\ldots,a_n \rightarrow b_1,\ldots,b_n}$ with all $a_i$ mutually distinct attribute names, and likewise for all $b_i$.

Consider a table $R^{\mathcal{I}}$ for $R[\{a_1, \ldots, a_n\}]$.

$$\delta_{a_1,\ldots,a_n \rightarrow b_1,\ldots,b_n}(R^{\mathcal{I}}) = \{f \circ g \mid f \in R^{\mathcal{I}} \text{ and } g : \{b_i \mapsto a_i\}_{1 \leq i \leq n}\}$$

where $f \circ g$ is function composition: $(f \circ g)(x) = f(g(x))$

Binary operators on tables of the same relational schema, defined like the usual set operations.

"Find all stops where line 3 departs, but line 8 does not depart."

"Find all stops where either line 3 or line 8 departs."

"Find all stops where both line 3 and line 8 depart."

It is sometimes convenient to define constant tables in queries.

"Find all stops near Helmholtzstr. (SID 42), including Helmholtzstr."

$$\delta_{\text{To}\to\text{StopId}}(\pi_{\text{To}}(\sigma_{\text{From}="42"}\text{Connect})) \cup \{\{\text{StopId} \mapsto 42\}\}$$

One can generalise this to constant tables with more than one column or more than one table.

Generalising the previous example:

"Stops that are Helmholtzstr."

$$R_0 = \{\{\text{From} \mapsto 42\}\}$$

"Stops that are next to Helmholtzstr."

$$R_1 = \delta_{\text{To} \to \text{From}}(\pi_{\text{To}}(\text{Connect} \bowtie R_0))$$

"Stops at distance 2 from Helmholtzstr."

$$R_2 = \delta_{\text{To} \to \text{From}}(\pi_{\text{To}}(\text{Connect} \bowtie R_1))$$

Stops reachable from Helmholtzstr. with a short-distance ticket:

$$R_0 \cup R_1 \cup R_2 \cup R_3 \cup R_4$$

What about all stops reachable from Helmholtzstr.?
⇝ see upcoming lectures …

Films

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

1. Who is the director of "The Imitation Game"?

$$\pi_{Director}(\sigma_{Title=\text{"The Imitation Game"}}(Films))$$

2. Which cinemas feature "The Imitation Game"?

$$\pi_{Cinema}(\sigma_{Title=\text{"The Imitation Game"}}(Program))$$

Films

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

3. What are the address and phone number of "Schauburg"?

$$\pi_{Address,Phone}(\sigma_{Cinema=\text{"Schauburg"}}(Venues))$$

4. *Boolean query:* Is a film directed by "Smith" playing?

$$\pi_{\emptyset}(\sigma_{Director=\text{"Smith"}}(Films) \bowtie Program)$$

Films

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

5. List the pairs of persons such that the first directed the second in a film, and vice versa.

$$\pi_{Director,D}(\sigma_{Director=A}(\sigma_{Actor=D}(\delta_{Title,Director,Actor \to T,D,A}(Films) \bowtie Films)))$$

6. List the names of directors who have acted in a film they directed.

$$\pi_{Director}(\sigma_{Actor=Director}(Films))$$

Films

| Title | Director | Actor |
|---|---|---|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | | |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|---|---|---|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|---|---|---|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

7. Return $\{\text{Title} \mapsto \text{"Ap. Now"}, \text{Director} \mapsto \text{"Coppola"}\}$ as the answer.

$$\{\{\text{Title} \mapsto \text{"Ap. Now"}\}\} \bowtie \{\{\text{Director} \mapsto \text{"Coppola"}\}\}$$

8. Find the actors cast in at least one film by "Smith".

$$\pi_{\text{Actor}}(\sigma_{\text{Director}=\text{"Smith"}}(\text{Films}))$$

Films

| Title | Director | Actor |
|---|---|---|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | | |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|---|---|---|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|---|---|---|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

9. Find the actors that are NOT cast in a movie by "Smith."

$$\pi_{Actor}(Films) - \pi_{Actor}(\sigma_{Director=\text{"Smith"}}(Films))$$

10. Find all pairs of actors who act together in at least one film.

$$q = \pi_{Actor',Actor}[\delta_{Actor \to Actor'}(Films) \bowtie Films]$$
$$q - \sigma_{Actor=Actor'}(q)$$

Consider the following identities and decide for each whether it is true or false. If true, prove your answer using the definitions from the lecture; if false, give a counterexample.

1. $R \bowtie S = S \bowtie R$
2. $R \bowtie (S \bowtie T) = (R \bowtie S) \bowtie T$
3. $\pi_X(R \circ S) = \pi_X(R) \circ \pi_X(S)$ for all $\circ \in \{\cup, \cap, -, \bowtie\}$
4. $\sigma_{n=m}(R \circ S) = \sigma_{n=m}(R) \circ \sigma_{n=m}(S)$ for all $\circ \in \{\cup, \cap, -\}$.
5. $\sigma_{n=m}(R \bowtie S) = \sigma_{n=m}(R) \bowtie S$, for $n$ and $m$ attributes of $R$ only.

### Discussion

Why are these identities of interest?

# Bonus Exercise: DIY

The set of operations $\{\sigma, \pi, \cup, -, \bowtie, \delta\}$ can express all queries of relational algebra. (Note that $\cap$ is syntactic sugar.) Show that it is not possible to reduce this set any further.

## Tip

For each operator, define a query that can be expressed using that operator. Then, show that this query cannot be expressed using the remaining operators.

## Remark

Video solutions available at: `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2020)/en`

# 4. First-order Queries

The relational queries considered so far produced a result table from a database.

Other query languages can be completely different, but they usually agree on this:

### Definition

- Syntax: a *query expression q* is a word from a query language (algebra expression, logical expression, etc.)
- Semantics: a *query mapping M[q]* is a function that maps a database instance $\mathcal{I}$ to a database table $M[q](\mathcal{I})$

## Generic Queries

We only consider queries that do not depend on the concrete names given to constants in the database:

### Definition

A query $q$ is *generic* if, for every bijective renaming function $\mu : \textbf{dom} \rightarrow \textbf{dom}$ and database instance $\mathcal{I}$:

$$\mu(M[q](\mathcal{I})) = M[\mu(q)](\mu(\mathcal{I})).$$

In this case, $M[q]$ is *closed under isomorphisms*.

### Example

A Non-generic query using datatypes:

$$\sigma_{Age>50}(\textit{Employees})$$

Lines:

| Line | Type |
|------|------|
| 85 | bus |
| 3 | tram |
| F1 | ferry |
| … | … |

Stops:

| SID | Stop | Accessible |
|-----|------|------------|
| 17 | St-Guilhem | true |
| 42 | Foch | true |
| 57 | Comédie | true |
| 123 | Av. Liberté | false |
| … | … | … |

Connect:

| From | To | Line |
|------|------|------|
| 57 | 42 | 85 |
| 17 | 789 | 3 |
| … | … | … |

Every table has a *schema*:

- Lines[Line:string, Type:string]
- Stops[SID:int, Stop:string, Accessible:bool]
- Connect[From:int, To:int, Line:string]

Idea: database instances are finite first-order interpretations
⤳ use first-order formulae as query language
⤳ use the fact perspective (more natural here)

Examples (using schema as in previous lecture):

- Find all bus lines: *Lines*($x$, bus)
- Find all possible types of lines: $\exists y.\text{Lines}(y, x)$
- Find all lines that depart from an accessible stop:

$$\exists y_{\text{SID}}, y_{\text{Stop}}, y_{\text{To}}.\left(\text{Stops}(y_{\text{SID}}, y_{\text{Stop}}, \text{true}) \land \text{Connect}(y_{\text{SID}}, y_{\text{To}}, x_{\text{Line}})\right)$$

# First-order Logic with Equality: Syntax

Basic building blocks:

- **Predicate names** with an arity $\geq 0$: $p$, $q$, Lines, Stops
- **Variables**: $x$, $y$, $z$
- **Constants**: $a$, $b$, $c$
- **Terms** are variables or constants: $s$, $t$

Formulae of first-order logic are defined as usual:

$$\varphi ::= p(t_1, \ldots, t_n) \mid t_1 \approx t_2 \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \exists x.\varphi \mid \forall x.\varphi$$

where $p$ is an $n$-ary predicate, $t_i$ are terms, and $x$ is a variable.

- An **atom** is a formula of the form $p(t_1, \ldots, t_n)$
- A **literal** is an atom or a negated atom
- Occurrences of variables in the scope of a quantifier are **bound**; other occurrences of variables are **free**

We use the usual shortcuts and simplifications:

- flat conjunctions ($\varphi_1 \wedge \varphi_2 \wedge \varphi_3$ instead of ($\varphi_1 \wedge (\varphi_2 \wedge \varphi_3)$)))
- flat disjunctions (similar)
- flat quantifiers ($\exists x, y, z.\varphi$ instead of $\exists x.\exists y.\exists z.\varphi$)
- $\varphi \rightarrow \psi$ as shortcut for $\neg\varphi \vee \psi$
- $\varphi \leftrightarrow \psi$ as shortcut for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
- $t_1 \not\approx t_2$ as shortcut for $\neg(t_1 \approx t_2)$

But we always use parentheses to clarify nesting of $\wedge$ and $\vee$.

For instance, "$\varphi_1 \wedge \varphi_2 \vee \varphi_3$" is not allowed!

First-order formulas are evaluated over sets of facts $\mathcal{I}$.[1]

To interpret formulas with free vars, we need a variable assignment $\mathcal{Z} : \text{Var} \rightarrow \textbf{dom}$. For an atom $\alpha$, let $\mathcal{Z}(\alpha)$ be the atom that results from replacing every variable $x$ in $\alpha$ with $\mathcal{Z}(x)$.

A formula $\varphi$ is satisfied by $\mathcal{I}$ and $\mathcal{Z}$, written $\mathcal{I}, \mathcal{Z} \models \varphi$, if:

- $\mathcal{I}, \mathcal{Z} \models p(t_1, \ldots, t_n)$ if $\mathcal{Z}(p(t_1, \ldots, t_n)) \in \mathcal{I}$
- $\mathcal{I}, \mathcal{Z} \models t_1 \approx t_2$ if $\mathcal{Z}(t_1 = t_2)$
- $\mathcal{I}, \mathcal{Z} \models \neg\varphi$ if $\mathcal{I}, \mathcal{Z} \not\models \varphi$
- $\mathcal{I}, \mathcal{Z} \models \varphi \wedge \psi$ if $\mathcal{I}, \mathcal{Z} \models \varphi$ and $\mathcal{I}, \mathcal{Z} \models \psi$
- $\mathcal{I}, \mathcal{Z} \models \varphi \vee \psi$ if $\mathcal{I}, \mathcal{Z} \models \varphi$ or $\mathcal{I}, \mathcal{Z} \models \psi$
- $\mathcal{I}, \mathcal{Z} \models \exists x.\varphi$ if there is $c \in \textbf{dom}$ with $\mathcal{I}, \mathcal{Z} \cup \{x \mapsto c\} \models \varphi$[2]
- $\mathcal{I}, \mathcal{Z} \models \forall x.\varphi$ if for all $c \in \textbf{dom}$ we have $\mathcal{I}, \mathcal{Z} \cup \{x \mapsto c\} \models \varphi$

---

[1] Reminder: a set of facts is also a database.

[2] W.l.o.g., assume that variables are quantified by at most one quantifier.

# First-order Logic Queries

## Definition: FO Queries

An $n$-ary first-order query $q$ is an expression $\varphi[x_1, \ldots, x_n]$ where $x_1, \ldots, x_n$ are exactly the free variables of $\varphi$ (in a specific order).

## Definition: FO Query Answering

An answer to $q = \varphi[x_1, \ldots, x_n]$ over an interpretation $\mathcal{I}$ is a tuple $\langle a_1, \ldots, a_n \rangle$ of constants such that

$$\mathcal{I} \models \varphi[x_1/a_1, \ldots, x_n/a_n]^3$$

where $\varphi[x_1/a_1, \ldots, x_n/a_n]$ is $\varphi$ with each free $x_i$ replaced by $a_i$.

The result of $q$ over $\mathcal{I}$ is the set of all answers of $q$ over $\mathcal{I}$.

---

[3]Note that $\varphi[x_1/a_1, \ldots, x_n/a_n]$ does not feature answer variables.

A **Boolean query** is a query of arity 0
$\rightsquigarrow$ we simply write $\varphi$ instead of $\varphi[]$
$\rightsquigarrow$ $\varphi$ is a closed formula (a.k.a. sentence)

What does a Boolean query return?

Two possible cases:

- $\mathcal{I} \not\models \varphi$, then the result of $\varphi$ over $\mathcal{I}$ is $\emptyset$ (the empty table)
- $\mathcal{I} \models \varphi$, then the result of $\varphi$ over $\mathcal{I}$ is $\{\langle\rangle\}$ (the unit table)

Interpreted as Boolean check with result true or false (i.e., match or no match)

Films

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

1. Who is the director of "The Imitation Game"?

$$\exists y_A. \text{Films}(\texttt{"The Imitation Game"}, x_D, y_A)[x_D]$$

2. Which cinemas feature "The Imitation Game"?

$$\exists y_T. \text{Program}(x_C, \texttt{"The Imitation Game"}, y_T)[x_C]$$

Films

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

3. What are the address and phone number of "Schauburg"?

$$\text{Venues}(\texttt{"Schauburg"}, x_A, x_P)[x_A, x_P]$$

4. *Boolean query:* Is a film directed by "Smith" playing?

$$\exists y_T, y_A, y_C, z_T.\ \text{Films}(y_T, \texttt{"Smith"}, y_A) \wedge \text{Program}(y_C, y_T, z_T)$$

**Films**

| Title | Director | Actor |
|-------|----------|-------|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

**Venues**

| Cinema | Address | Phone |
|--------|---------|-------|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

**Program**

| Cinema | Title | Time |
|--------|-------|------|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

5. List the pairs of persons such that the first directed the second in a film, and vice versa.

$$\exists y_T, z_T. \ \text{Films}(y_T, x_D, x_A) \wedge \text{Films}(z_T, x_A, x_D)[x_D, x_A]$$

6. List the names of directors who have acted in a film they directed.

$$\exists y_T. \ \text{Films}(y_T, x_D, x_D)[x_D]$$

**Films**

| Title | Director | Actor |
|---|---|---|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

**Venues**

| Cinema | Address | Phone |
|---|---|---|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

**Program**

| Cinema | Title | Time |
|---|---|---|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

7. Return $\{\text{Title} \mapsto \texttt{"Ap. Now"}, \text{Director} \mapsto \texttt{"Coppola"}\}$ as the answer.

$$\{\texttt{DirectedBy}(\texttt{"Apocalypse Now"}, \texttt{"Coppola"})\}$$

8. Find the actors cast in at least one film by "Smith".

$$\exists y_T.\ \texttt{Films}(y_T, \texttt{"Smith"}, x_A)[x_A]$$

Films

| Title | Director | Actor |
|---|---|---|
| The Imitation Game | Tyldum | Cumberbatch |
| The Imitation Game | Tyldum | Knightley |
| … | … | … |
| Internet's Own Boy | Knappenberger | Swartz |
| Internet's Own Boy | Knappenberger | Lessig |
| Internet's Own Boy | Knappenberger | Berners-Lee |
| … | … | … |
| Dogma | Smith | Damon |
| Dogma | Smith | Affleck |

Venues

| Cinema | Address | Phone |
|---|---|---|
| UFA | St. Petersburger Str. 24 | 4825825 |
| Schauburg | Königsbrücker Str. 55 | 8032185 |
| … | … | … |

Program

| Cinema | Title | Time |
|---|---|---|
| Schauburg | The Imitation Game | 19:30 |
| Schauburg | Dogma | 20:45 |
| UFA | The Imitation Game | 22:45 |

9. Find the actors that are NOT cast in a movie by "Smith."

$$\exists y_T, y_D.\text{Films}(y_T, y_D, x_A) \land$$
$$\forall x_T, x_D.(\text{Films}(x_T, x_D, x_A) \to x_D \not\approx \text{"Smith"}[x_A]$$

10. Find all pairs of actors who act together in at least one film.

$$\exists y_T, y_D, y_D'. \text{Films}(y_T, y_D, x_A) \land \text{Films}(y_T, y_D', x_{A'}) \land x_A \not\approx x_{A'}[x_A, x_{A'}]$$

# Summary

We have covered the following topics:

- The relational data model
- Relational queries
- First-order queries

### Future Content:

- Query expressivity: comparing RA and FO queries
- Complexity of query answering