Multimedia (MM) Databases

Query Languages (QL) / Query Processing

Prof. (FH) PD Dr. Mario Döller

Table of Contents

Part 1:

- MMQL in General
- History of MMQL
- Categories of MMQL
- 1st Category: Representative Examples
  - MOQL
  - SQL/MM

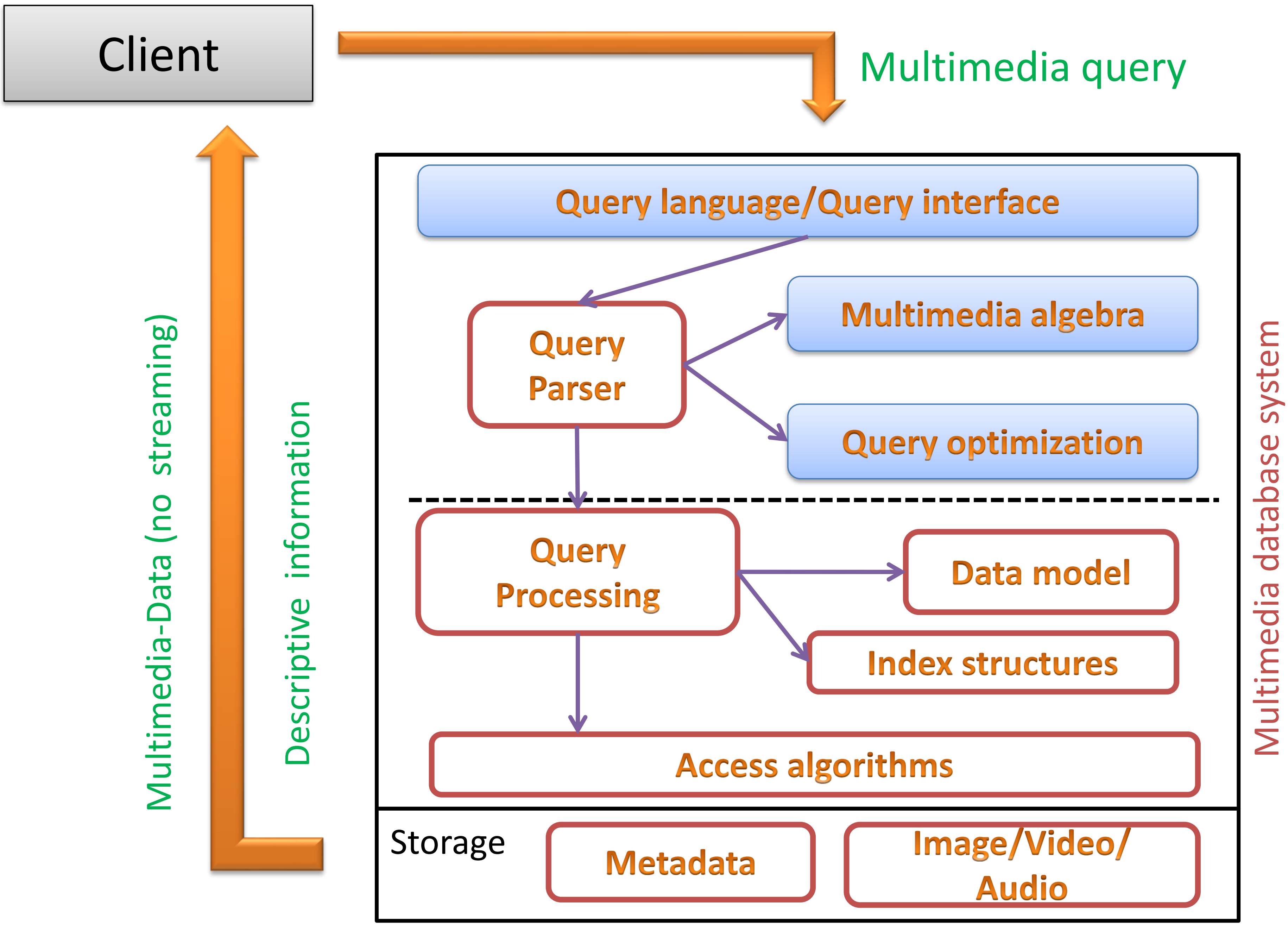
Part 2

- 2nd Category: Representative Example
  - MPEG Query Format
- Result Presentation
- Query Processing and Optimization

Table of Contents

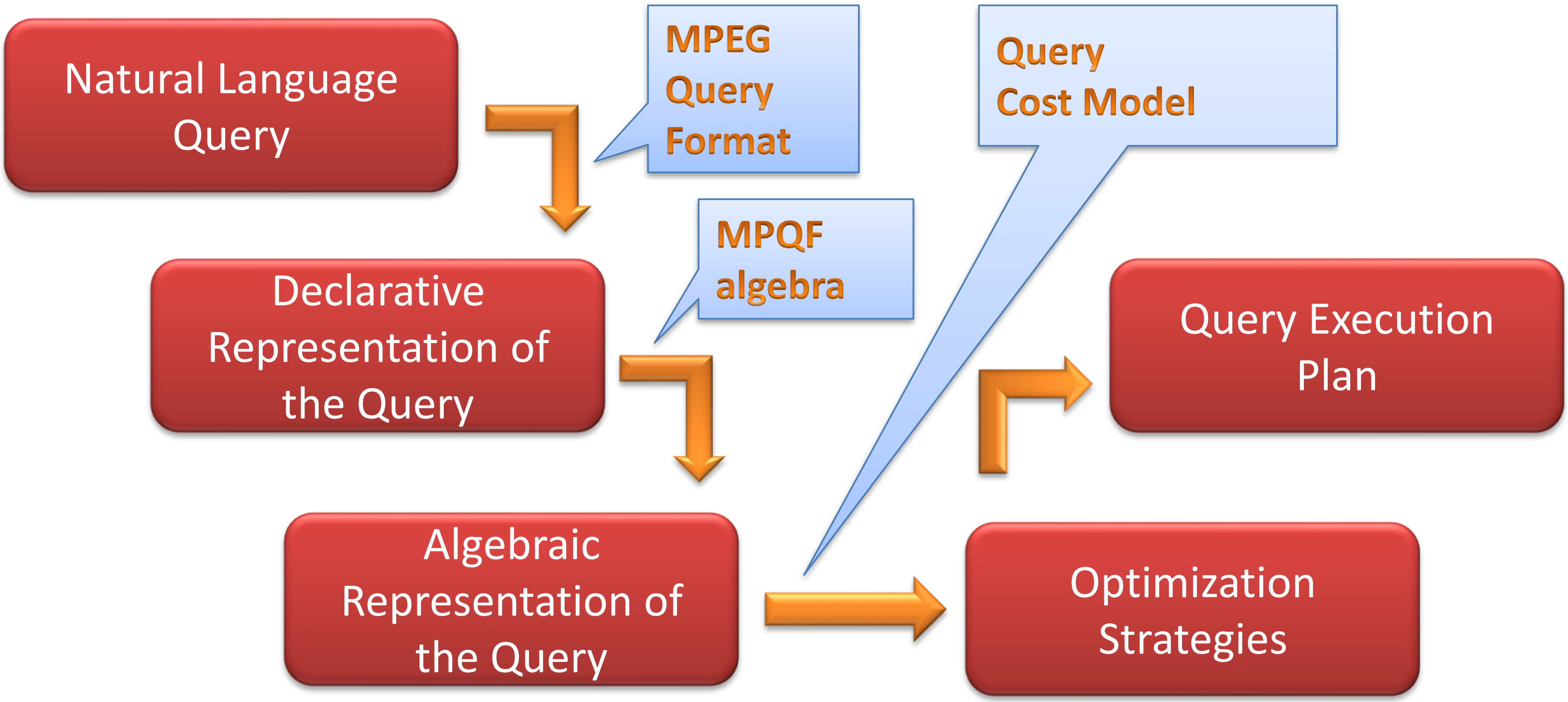
Part 1 :

Part 2


Client
Multimedia query
Multimedia-Data (no streaming)
Descriptive information
Query language/Query interface
Query Parser
Multimedia algebra
Query optimization
Query Processing
Data model
Index structures
Access algorithms
Multimedia database system
Storage
Metadata
Image/Video/ Audio

Query Types in MM Systems: Classification

- Exemplary end-to-end workflow of a multimedia database system



Query Types in MM Systems: Example

Natural language query: example

Give me all images and their titles, which are similar to my example image and were taken in Berlin, whereupon the similarity to the example image is much more important than its association to Berlin.

In addition, the data size of the selected images should not exceed the value 2048Kb.

Query Types in MM-Systems I

- Classical exact queries
  - Targeting non multimedia attributes

- Semantic queries
  - Determination of the query result based on descriptions of the semantic content (occurrence of specific objects, persons)

- Syntactic queries
  - Targeting basic characteristics of the media
  - ex.: resolution, framerate

Query Types in MM-Systems II

- Similarity queries (content based)
  - Applied on low-level features of the media (ex.: color distributions) and look for media with similar features („give me all images similar to my query image“).

- Correlation queries
  - Try to identify spatial and temporal correlations in media („give me all images in which a red ball is next to a yellow one“).

Requirements for a MMQL

- The main general requirements for a MM query are the following:
  - Universality (also support querying „classical“ database attributes)
  - Content-based (semantic) queries
  - Spatial queries
  - Temporal queries
  - Content-based similarity queries
  - Fuzzy queries
  - Presentation

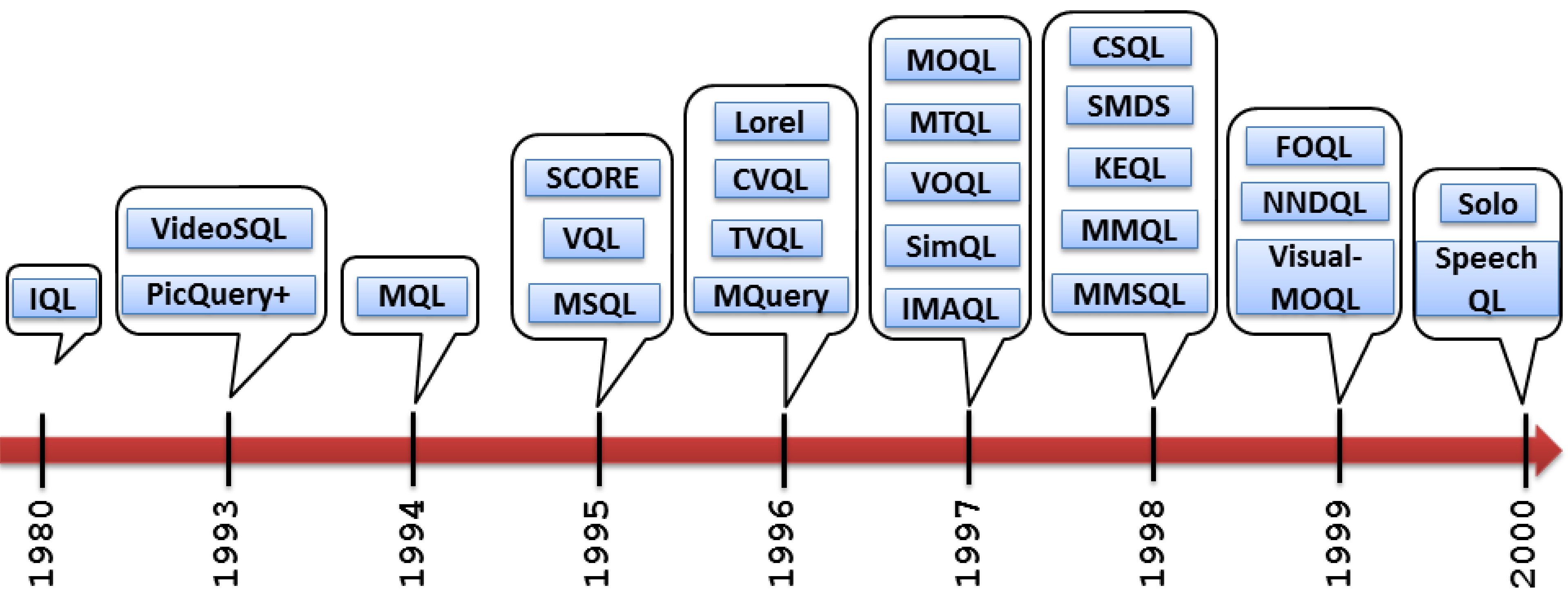
Table of Contents

Part 1 :

Part 2

History of MMQL (1980 - 2000)

- Focus on image data (medical images)
- Mainly spatial and similarity-based queries
- Mainly extensions of existing languages (SQL, OQL)



History of MMQL (2001 - 2011)

- Multimedia in general (also multimodal),
- Temporal queries, Relevance Feedback,
- New standards (SQL/MM, MPQF),
- Fuzzy logic, user preferences, thresholds, …

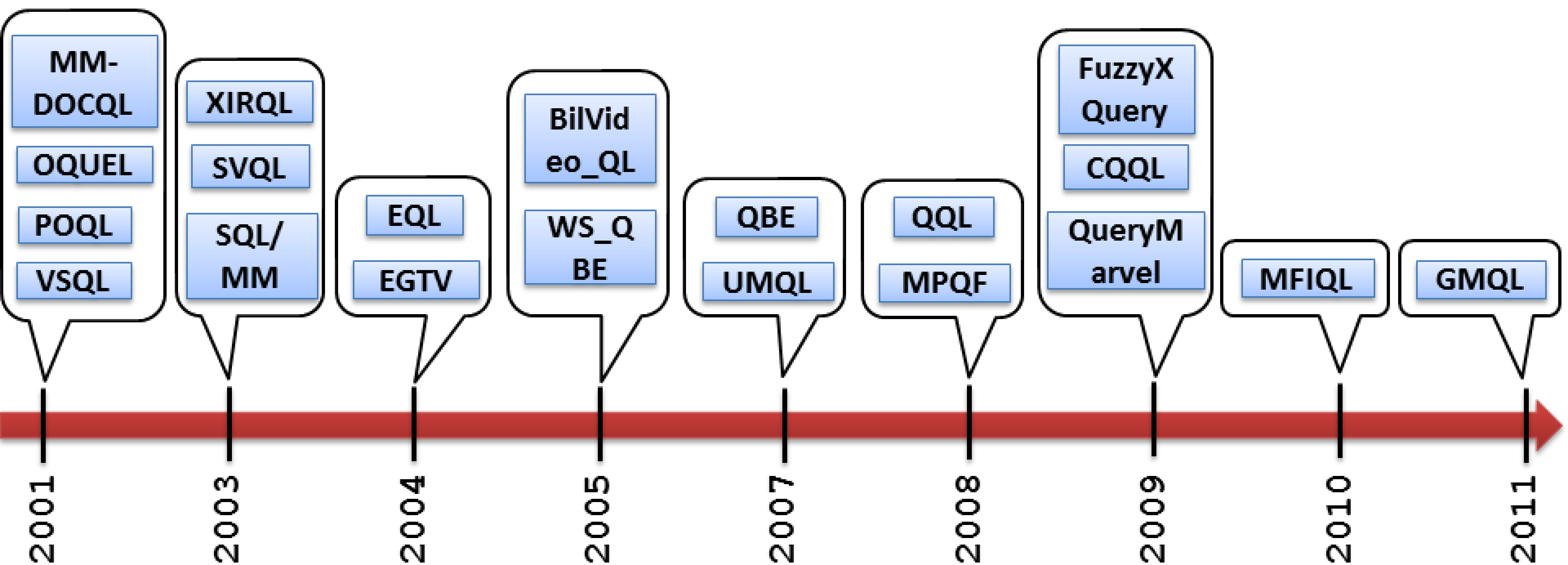


Table of Contents

Part 1 :

Part 2

Categories of MMQL

1st Extension of SQL and OQL

- SQL/MM and MOQL (in the following slide outlined as examples)

2nd „From scratch“

- ex.: VideoSQL, MPEG Query Format

Extensions of OQL/SQL

- Several approaches for standard extensions of OQL and SQL
  - MOQL for OQL (attempt to integrate into OQL)
  - SQL/MM for SQL-99 (standardized by ISO/IEC Working Group, SQL of the JTC 1/SC 32)
  - De facto standards of individual providers, ex.: in Oracle MultiMedia.

Table of Contents

Part 1 :

- MMQL in General
- History of MMQL
- Categories of MMQL
- 1st Category: Representative Examples
  - MOQL
  - SQL/MM

Part 2

- 2nd Category: Representative Example
  - MPEG Query Format
- Result Presentation
- Query Processing and Optimization

Overview

- A general query language?
- Object Query Language
- Multimedia extensions (MOQL)
  - Spatial relations
  - Temporal relations
  - Result presentation
- VisualMOQL / DISIMA Project
- Summary

A General Query language ?

- Importance of the problem of acceptation by potential users
- OQL or SQL-Syntax very successful/ well-known
- Object-orientation desirable

Idea:

Extend an existing Query language,
concretely: OQL (Object Query Language)

OQL

- based on the ODMG object model

- Similar to SQL-92;
  - object-oriented extensions:
  - complex objects, object identity, path expressions, polymorphisms, function calls, Late Binding

- Embedded in query languages

OQL

- Basic query construct:

```
select [ distinct ] projection_attributes
from   query  [ [ as ] identifier ]
    {,   query  [ [ as ] identifier ] }
where query
```

MOQL (Multimedia Object Query Language)

- Extensions in the where clause of OQL queries:
  - spatial relations (spatial_expression).
  - temporal relations (temporal_expression).
  - ‚contains' relation (contains_predicate)
  - Presentation functions using 'present' clause

Spatial Predicates

Return Value	Point	Line	Region (circle, rectangle)
Point	nearest, farthest	within, midpoint	centroid, inside
Line	cross	intersect	inside (contains), cross
Region (...)	cover	cover (coveredBy), cross	topological_predicate, directional_predicate

Directions:

left, right, above, below, front, back, north, south, west, east, northwest... and combinations with front/back (front_left, back_north ...)

Spatial Functions

Return value	Point	Line	Region	Value
Point	nearest, farthest		region	
Line	intersect	intersect	region	length, slope
Region	centroid		interior, exterior, mbr	area, perimeter

```
select   lake, area(lake.region)
from     Lakes lake
where lake.region coveredBy SachsenAnhalt
and      area(lake.region) > 10
```

Temporal Relations

- By time intervals:
  - equal, before, after, meet, metBy, overlap, overlappedBy, during, include, start, startedBy, finish, finishedBy

- Time intervals have a start and an end

- A time point is a time interval for which start=stop

Time constructs: year, month, day, hour, minute, second, ms

Temporal Continuous Media

- Functions (only video data): (universal: timeStamp)

Return value	Frame	clip	video
Frame	prior, next	clip	
Clip	firstFrame, lastFrame, nth	prior, next	video
Video		firstClip, lastClip, nth	

Predicates (camera motions):
zoomIn, zoomOut, panLeft, panRight, tiltUp, tiltdown, cut, fade, wipe, dissolve

Example of Video Query

- „Find the first film segment with person *MrX* from the video *JamesB*„

```
select   firstClip(
    select c from JamesB.clips c
    where c contains MrX
    order  by  lowerBound(c.timestamp)
 )
```

Presentation Functions

- New present clause

```
select ... from ... where
where  layout   { and  layout }
```

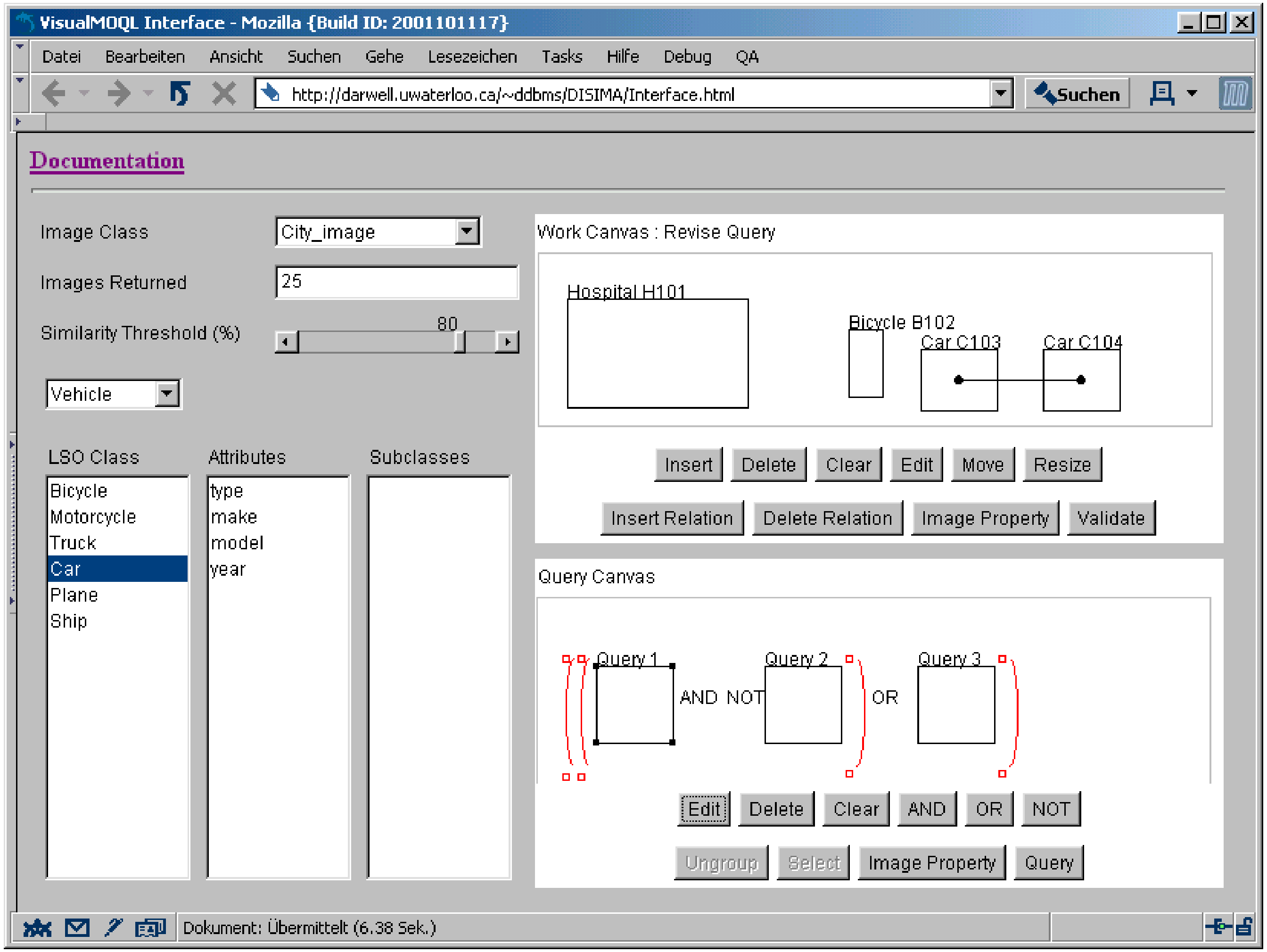
- The Layout is made of spatial and temporal entries, or of a user-defined ‚scenario‘.
- Entries/functions: atWindow, play, parStart, display

VisualMOQL

- Implementation of the image part of MOQL

- Part of the DISIMA project (Distributed Image Database Management System)
  - Content-based Queries (‚salient‘ objects)
  - Declarative queries

VisualMOQL



VisualMOQL Interface - Mozilla {Build ID: 2001101117}
Datei Bearbeiten Ansicht Suchen Gehe Lesezeichen Tasks Hilfe Debug QA
http://darwell.uwaterloo.ca/~ddbms/DISIMA/Interface.html
Suchen
Documentation
Image Class
City_image
Images Returned
25
Similarity Threshold (%)
80
Vehicle
LSO Class
Bicycle
Motorcycle
Truck
Car
Plane
Ship
Attributes
type
make
model
year
Subclasses
Work Canvas : Revise Query
Hospital H101
Bicycle B102
Car C103
Car C104
Insert Delete Clear Edit Move Resize
Insert Relation Delete Relation Image Property Validate
Query Canvas
Query 1
AND NOT
Query 2
OR
Query 3
Edit Delete Clear AND OR NOT
Ungroup Select Image Property Query
Dokument: Übermittelt (6.38 Sek.)

VisualMOQL

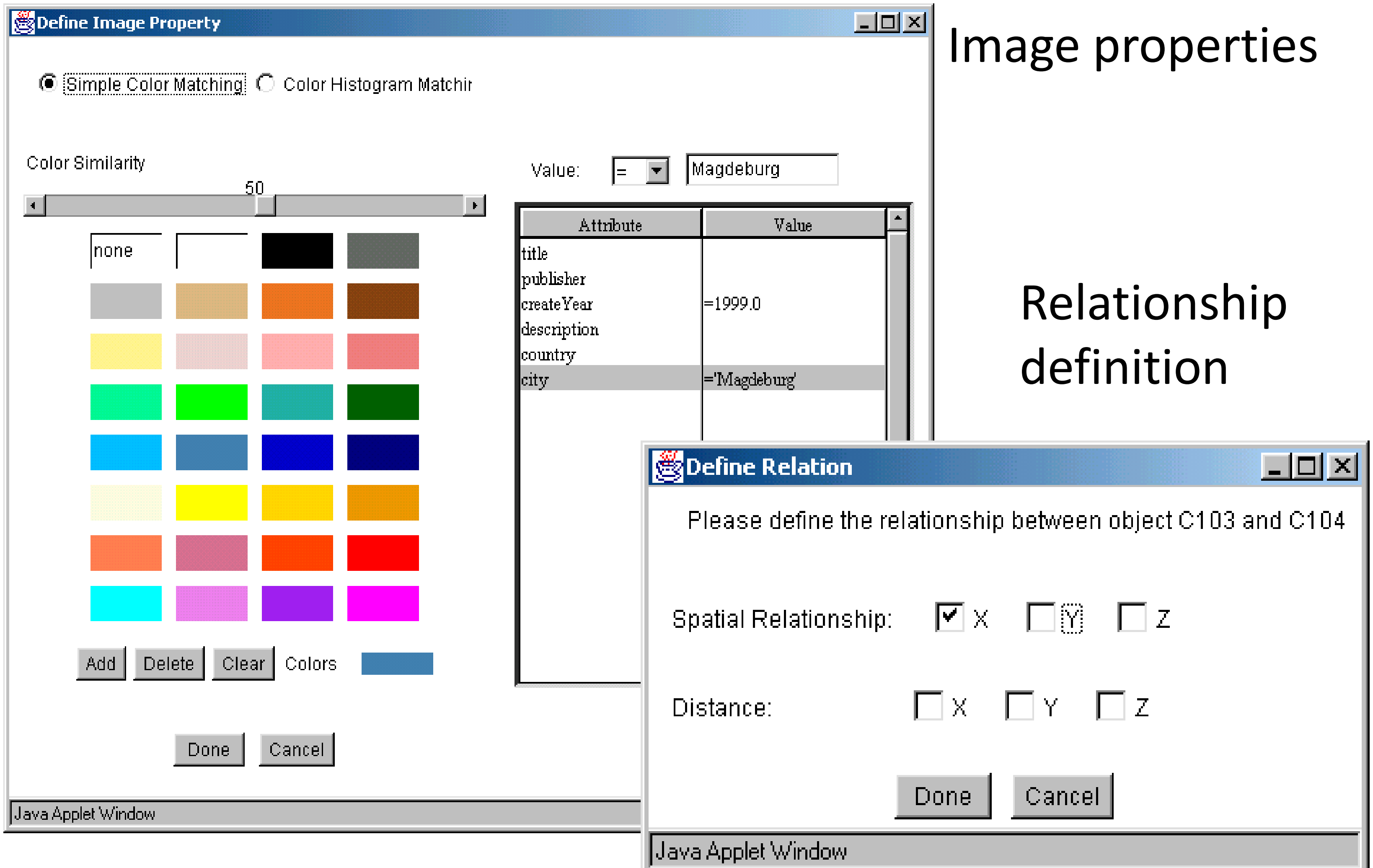
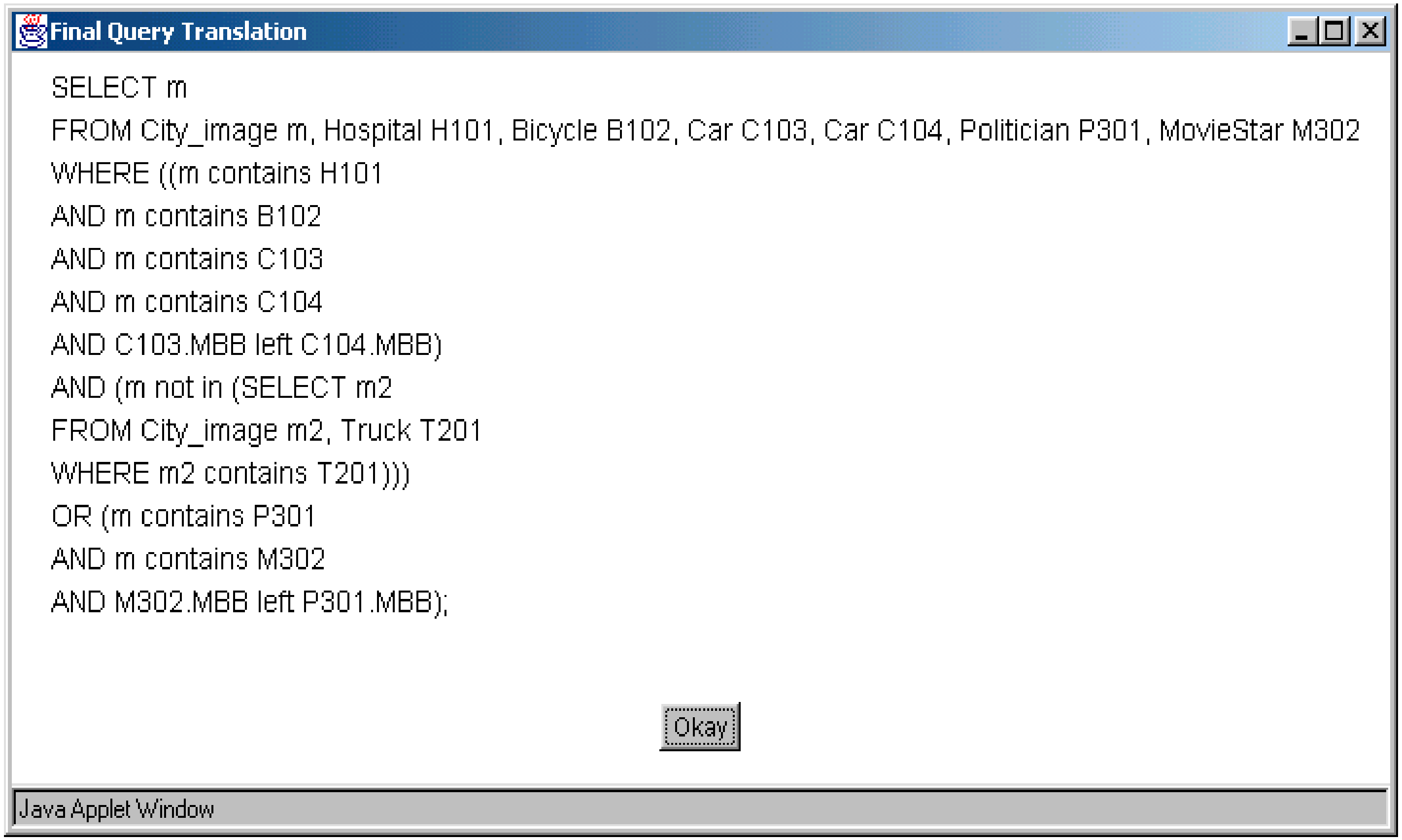


Image properties

Relationship definition

VisualMOQL



Result presentation: File list with preview images

Summary

- MOQL extends the established Object Query Language.
- Supports in theory all requirements of a general MM Query language.
- No support for audio (yet).
- Until now only a prototype implemented on ObjectStore.

Table of Contents

Part 1 :

Part 2

SQL/MM (MM for MultiMedia)

- ISO/IEC-Standard (ISO/IEC Working Group, SQL of the JTC 1/SC 32), which defines several “classes libraries” on SQL object types.
- The structured types defined in these libraries are first-class SQL types, which can be expressed with SQL:1999 instructions.
- International standard since 2002.

SQL/MM Overview

- Belongs to the SQL standard, is however self-contained
  - SQL: ISO/IEC 9075, SQL/MM: ISO/IEC 13249
- Composed of several parts
  - Part 1: SQL/MM **Framework (**IS Nov. 2002)
  - Part 2: SQL/MM **Full Text (**IS Okt. 2000)
  - Part 3: SQL/MM **Spatial (**IS Dez. 1999)
  - Part 5: SQL/MM **Still Image** (IS Mai 2001)
- Part 1 provides an overview and specifies conformity
- Each further part:
  - Deals with a specific media data type
  - Is composed of UDT's, methods and functions defined according to SQL:1999

SQL/MM Full Text

- Version of 10.12.2001
- Specifies
  - UDT **FullText** for text data and
  - UDT **FT_Pattern** for search patterns
- FullText:
  - Two search methods:
    - **Contains**: Boolean search ⇒ result: yes/no
    - **Rank**: Ranking ⇒ result: implem.-dependent real value
  - Two constructors (String, String+ language)
  - Function **FullText_to_Character** to create a string

SQL/MM Full Text: Search pattern for Contains and Rank

- Contextual pattern

```
aText.Contains ('
    ("Abschnitt") near "Standard"
    within 0 sentences in order
') = 1
```

- Conceptual pattern

```
aText.Contains ('
    is about "Internationaler Standard zur
    Volltextsuche"
') = 1
```

  – Single sentence, count of single word patterns, sets of sentences, patterns with Boolean operators (I, &, NOT)

- Example of query:

```
select * from myDocs
where Doc.Rank(' "Standard" ') > 0.8
```

SQL/MM Spatial

- Version of 10.12.2001, 581 pages
- Corresponds to the type Graphic
- Specifies UDT's for
  - 2D-Data (Point, Line, Area)
  - Collections of such data items
- Defines routines for
  - Manipulation, search and comparison of spatial data
  - Conversion between the UDT's and string or binary representations
- For each geometry object (ST_Geometry)
  - an **SRID** (spatial reference system identifier) specifies the spatial reference system

SQL/MM Spatial: Types

- 0-dim: ST_Point
- 1-dim: ST_Curve
  - Subtypes defined by interpolation between individual points
  - **ST_LineString**: linear interpolation
  - **ST_CircularString**: circular interpolation
  - **ST_CompoundString**: mixed
- 2-dim: ST_Surface
  - **ST_CurvePolygon**: 1 external + n internal ST_CompoundString-Umrandungen
  - **ST_Polygon**: only ST_LineString sides
- Collection objects
  - Same reference system for all elements
  - **ST_MultiPoint**
  - **ST_MultiCurve, ST_MultiLineString**
  - **ST_MultiSurface, ST_MultiPolygon**

SQL/MM Still Image

- Version of 10.12.2001
- specifies
  - UDT **SI_StillImage** for image data,
  - UDT **SI_Feature** for features and
  - UDT **SI_FeatureList** for lists of features
- SI_StillImage:
  - two constructors (BLOB, BLOB + Format)
  - two mutator methods: BLOB replacement + format change
  - two Observer to create thumbnails
- internal representation left free
  - No data dependencies!

SQL/MM Still Image: UDT SI_StillImage

```
create type SI_StillImage as (
    SI_content binary large object(SI_MaxContLength),
    SI_contentLength integer,
    SI_format character varying(8),
    SI_height integer,
    SI_width integer,
    …
    )
```

- SI_content:
  - also covers registration data (Header, color tables etc.)
  - Container for the whole image
- SI_format:
  - Supported formats
    - the DBS can read them and extract image properties
  - User defined formats

SQL/MM Still Image: features (Features) I

- Basis type SI_Feature has the following subtypes:
  - **SI_AverageColor**: a single color for the whole image
  - **SI_ColorHistogram**: frequencies of color groups
  - **SI_PositionalColor**: division of the image in rectangles with the corresponding average color
  - **SI_Texture**: size, illumination variation, dominant direction of repeating patterns (i.e. textures)

- All features have a method SI_Score, which
  - Computes the distance between an image and a feature value and
  - Returns a real value between 0 and 1.

SQL/MM Still Image: features (Features) II

- Alle subtypes of SI_Feature have a corresponding feature extraction method.

- Objects of types SI_AverageColor and SI_ColorHistogram can be directly instantiated (from constants).

- CBR functionality: the polymorphic SI_Score methode compares two signature vectors.

```
SELECT p1, p2
FROM Picture1 p1, Picture2 p2
WHERE
  p1.photo1_color.SI_Score(p2. photo2) > 0.5 AND
  p1.photo1_texture.SI_Score(p2.photo2)  > 0.4
```

Similarity Comparison in OR Databases

- Oracle's Multimedia and IBM DB2 Extenders (cf. later) base on the SQL/MM concepts for similarity comparison
- However: different SQL Syntax and no polymorphic ScoreFunction.
- Same query as before for Oracle:

```
SELECT p1.description, p2.description
FROM Picture p1, Picture p2
WHERE
ORDSYS.IMGSimilar(p1.photo1_sig, p2.photo2_sig,
     'color="0,6" texture="0,2" shape="0,1" location="0,1"', 20)=1;
```

Table of Contents

Part 1:

Part 2

Table of Contents

Part 1 (F. Sadiku):

- MMQL in General
- History of MMQL
- Categories of MMQL
- 1st Category: Representative Examples
  - MOQL
  - SQL/MM

Part 2 (M. Döller)

- 2nd Category: Representative Example
  - MPEG Query Format
- Result Presentation
- Query Processing and Optimization

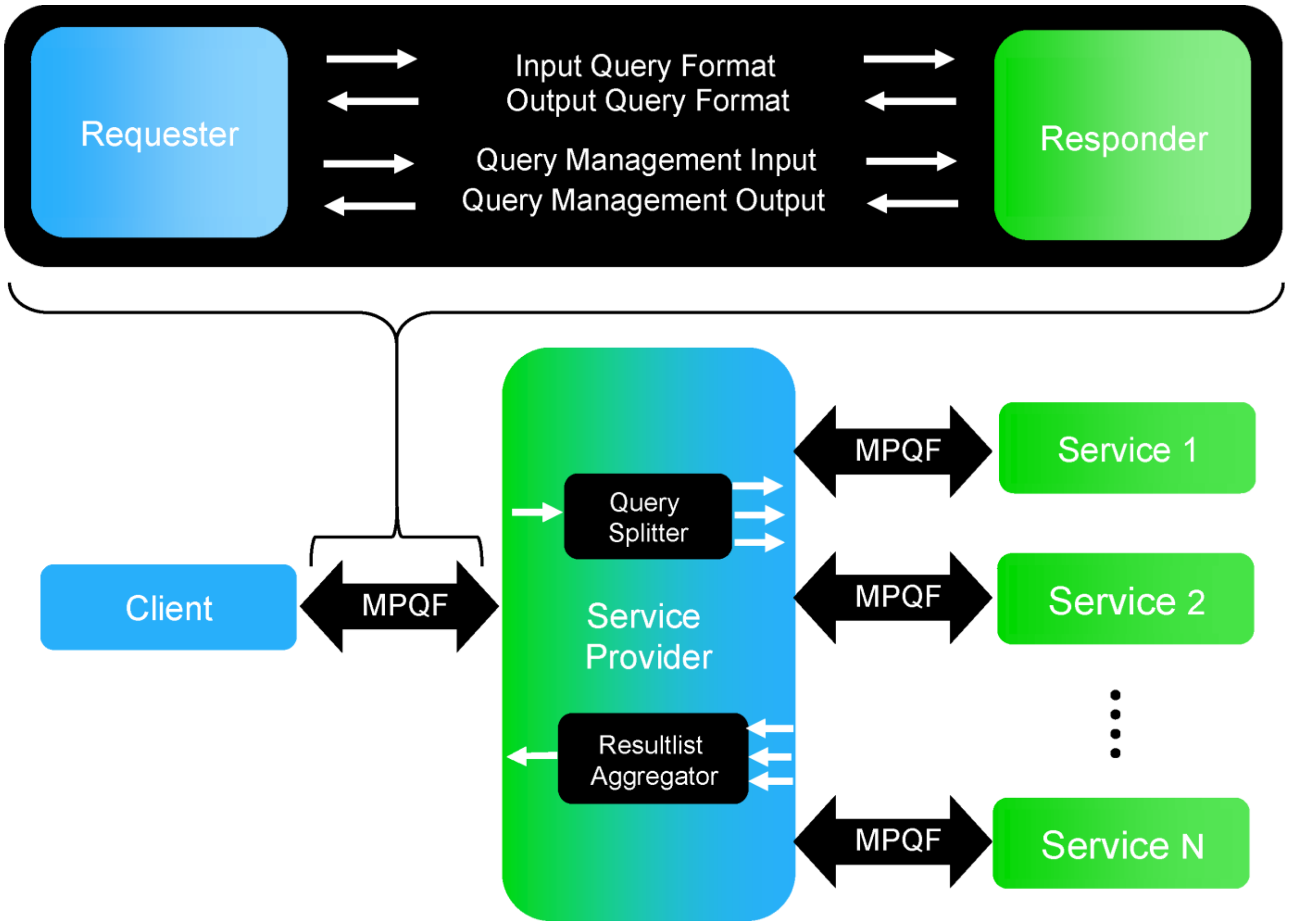
The MPEG Query Format (MPQF)

- International standard since the end of 2008, Part 12 of thMPEG-7 standard

- General concepts
  - Based on XML, defined using an XML Schema
  - Decoupled from a specific metadata standard (also MPEG-7)
  - Supports all XML based metadata descriptions
  - Integrates limited XQuery functionality

- MPQF is composed of three main categories:
  - Management
  - Input Query Format
  - Output Query Format

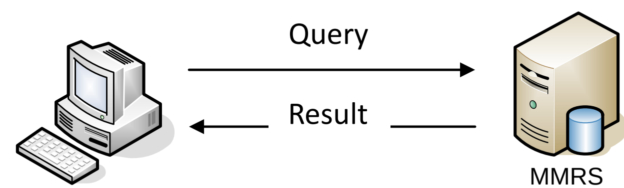
MPQF Scenario



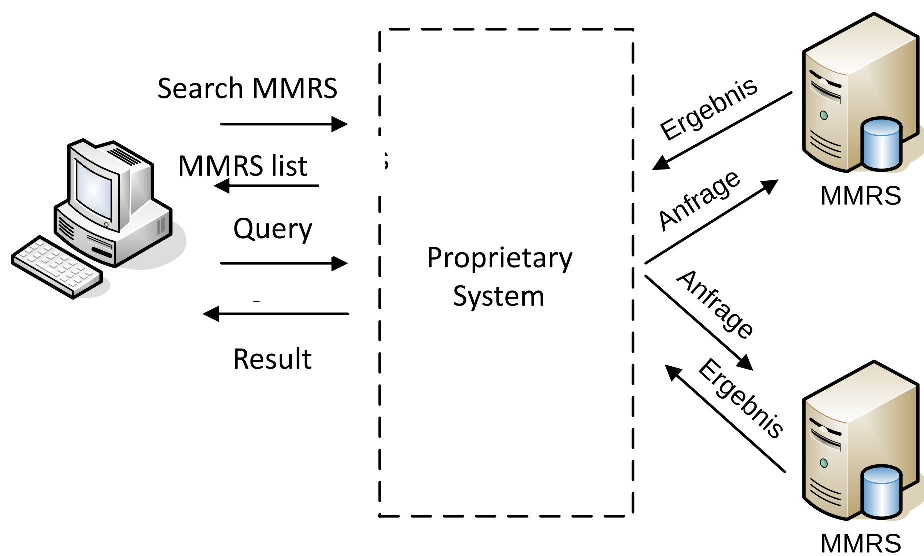
MPQF Concepts
Management

- How to find the right Multimedia search engine?
  - 2 Scenarios
    - MMRS is known to the user
    - MMRS(s) are unknown. How to find the right one(s) for my search

1. Scenario



2. Scenario

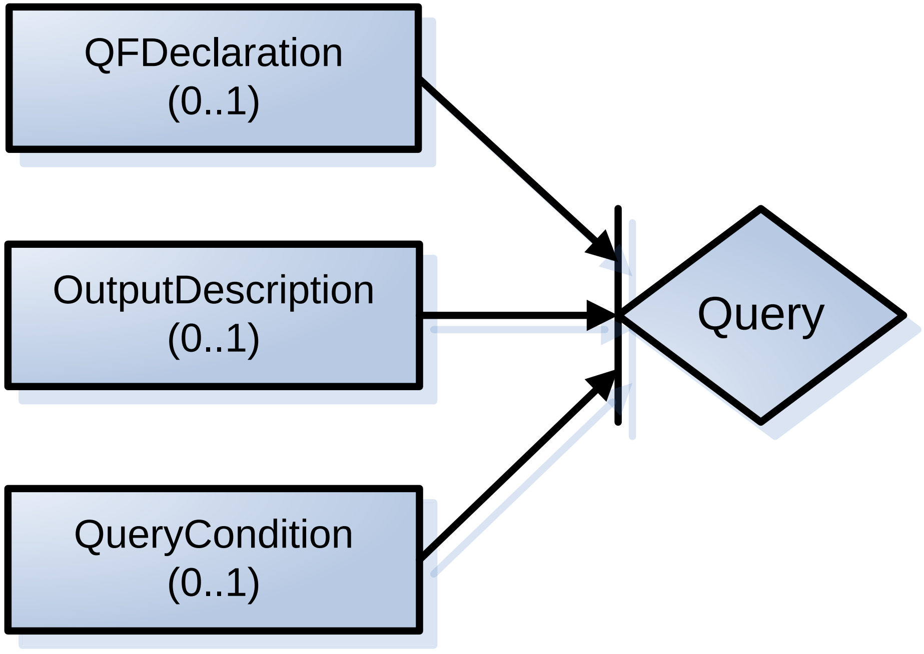


MPQF Concepts
Query I

- Parts of a Query
  - QFDeclaration
  - OutputDescription
  - QueryCondition

- MPQF supports:
  - Synchronous/Asynchronous mode
  - Timeout Functionality

General Query structure



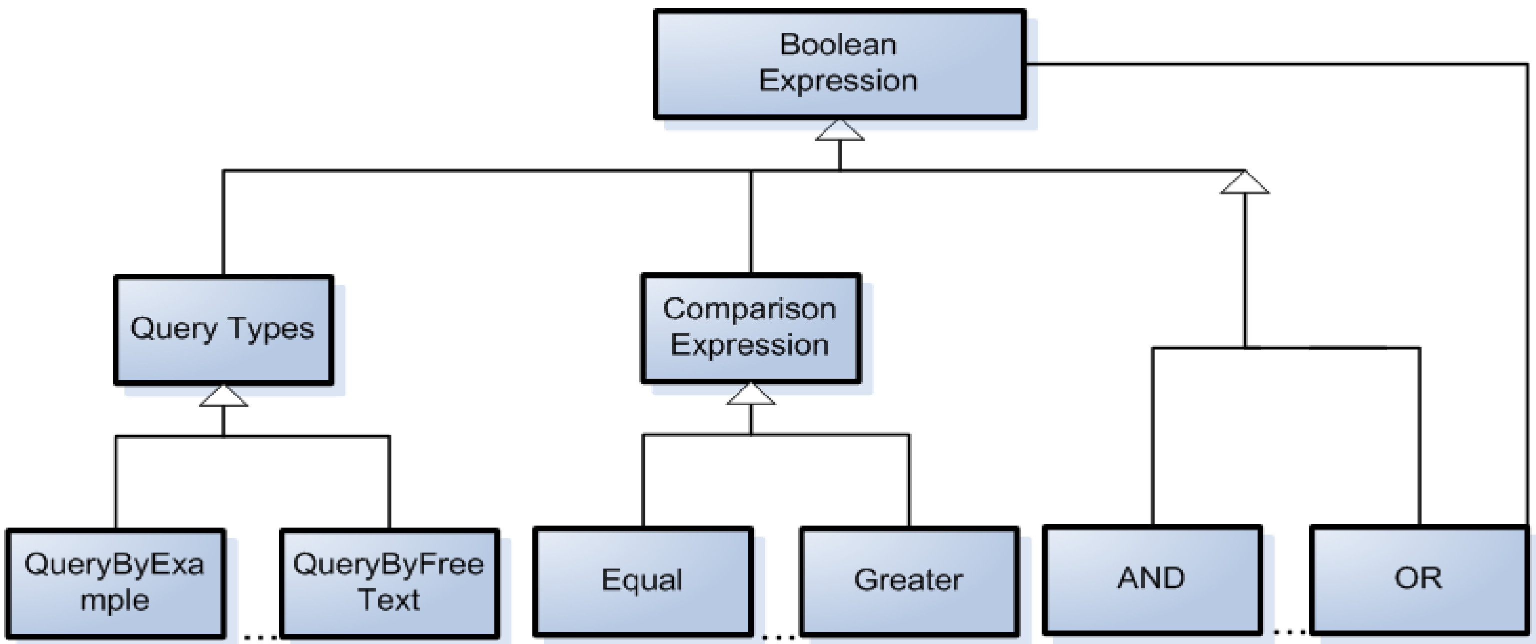
MPQF Concepts
Query II

- QFDeclaration
  - Declaration of resources for query conditions
    - The following resources are available: (structured) text, media or their metadata description (ex.:: DominantColorType of MPEG-7)

- OutputDescription
  - Defines the Content as well as the structure of the result set
  - Uses XPath to select elements of the metadata description
    - Supports absolute and relative addressing
  - Description independent
  - Provides grouping and sorting functionality
  - Also restriction and paging of the result set

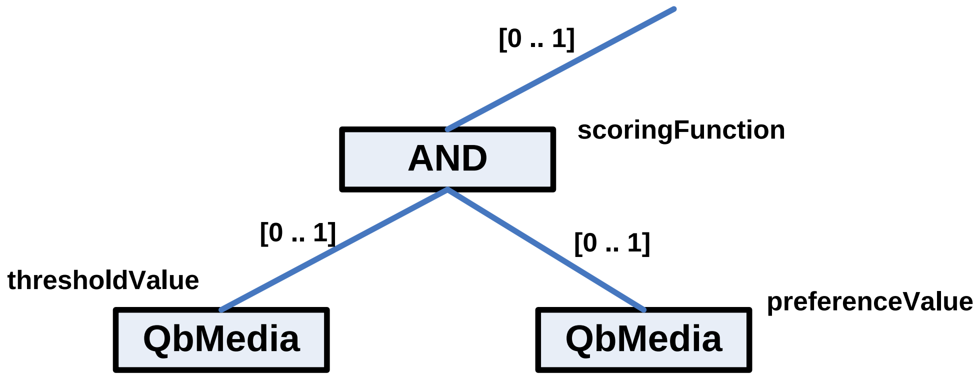
MPQF Concepts
Query conditions I

- Modular filter architecture
- *TargetMediaType* for encoding filtering
- Join functionality



MPQF Concepts

Query Conditions II



- *PreferenceValue* and *thresholdValue* for each condition.
- *ScoringFunction* for each „Fuzzy Boolean Operator“ (AND, OR, XOR) (It is recommended that the functions comply to t-norm or t-conorm rules).
- Results in a rank and confidence evaluation for each element.

Table of Contents

Part 1:

- MMQL in General
- History of MMQL
- Categories of MMQL
- 1st Category: Representative Examples
  - MOQL
  - SQL/MM

Part 2

- 2nd Category: Representative Example
  - MPEG Query Format
- Result Presentation
- Query Processing and Optimization

Result Presentation

- Very important for MMDBS.
- More complex than for traditional DB.
- Spatial and temporal information necessary (ex.: order of execution).
- Different options:
  - Media composition
  - Interactive playout
  - Synchronization

Query and presentation: SQL+D

- *SQL+D is* a Multimedia and presentation extension for object-relational SQL.
  - Enables the user to specify the display layout of an SQL-Query to control the presentation of the results.

```
SELECT a,v FROM MONUM
  WHERE country='USA'
  DISPLAY panel main
  WITH a AS audio A, v AS video V ON main.Center(Overlay),
  SHOW V,A
```

Query Processing in SQL+D

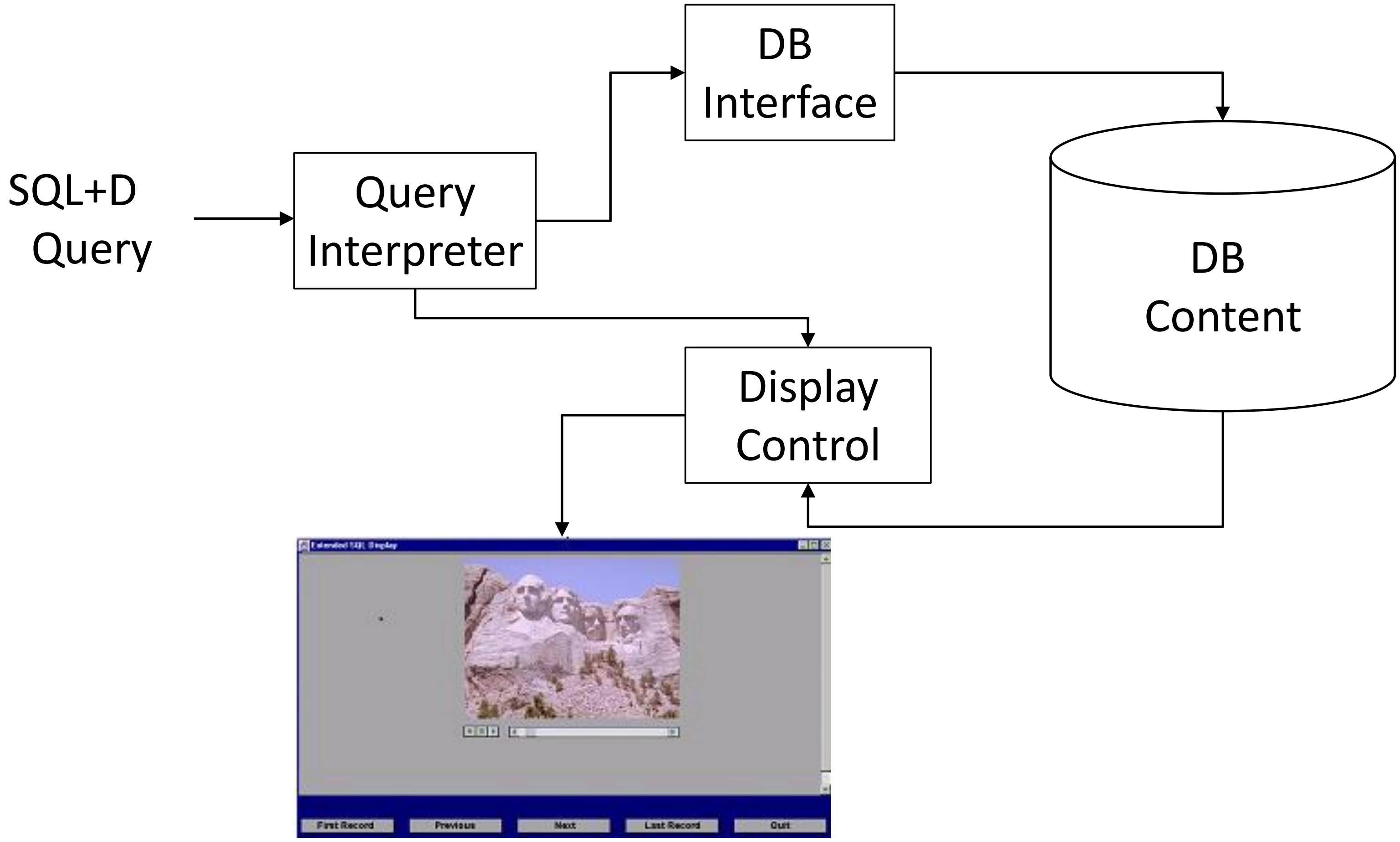


Table of Contents

Part 1:

Part 2

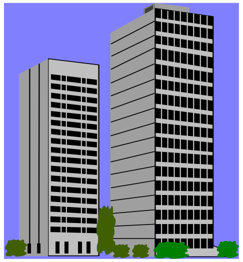
Overview

- Multimedia Query Examples
- Multimedia Query Processing and Optimization Requirements
- Image Data Modeling
- A Similarity-based Algebra
- Multimedia Query Optimization

Reference:

S. Atnafu and L. Brunie and H. Kosch: "Similarity-Based Operators and Query Optimization for Multimedia Database Systems", In International Database Engineering and Applications Symposium (IDEAS) 2001. IEEE CS Press, pp. 346-355, Grenoble (France), July 2001.

Multimedia Query Processing I



SI (Photo, FV, Time, Date) – images taken by the surveillance camera
EMP (Photo, FV, Name, Occupation) – employees

- Query 1: for a specific image of a person in SI, find its most similar image in EMP.
  - Query usually supported by existing systems
  - Can be expressed in MOQL, also partially in SQL/MM

Multimedia Query Processing II

- for some scenarios SI alone does not deliver enough information.

- Query 2: for pictures of individual persons in SI, taken the day before between 4 pm and 6 pm, find their most similar image in EMP as well as the corresponding names and addresses.

  - Such a query is not supported by existing systems,
  - It requires a relational selection in the SI table and a "similarity-based Join" of SI and EMP,

Multimedia Query Processing and Optimization

- Requirements of multimedia query processing and optimization:
  - Definition of a data model to manipulate multimedia files
  - Definition of multimedia operations, such as for example: “*Similarity-Based Selection and Join*”
  - Development of a formal algebra for MMDB query operations
  - Development of strategies for multimedia query optimization

General Goals

A

DBMS:

- Metadata for image description/query*:*
  (already available since ~ 30 years)
- Operations for keyword matching (relational)

  but,
    - incomplete representation of images,
    - subjective descriptions,
    - time consuming.

B

Computer view:

- very promising CBIR methods:
  (already available since ~ 20 years)
- automatic extraction of content,
- Description: color, texture, shape, etc.,
- **but,**
    - similarity-based (non-exact).
    - current approaches insufficient

C

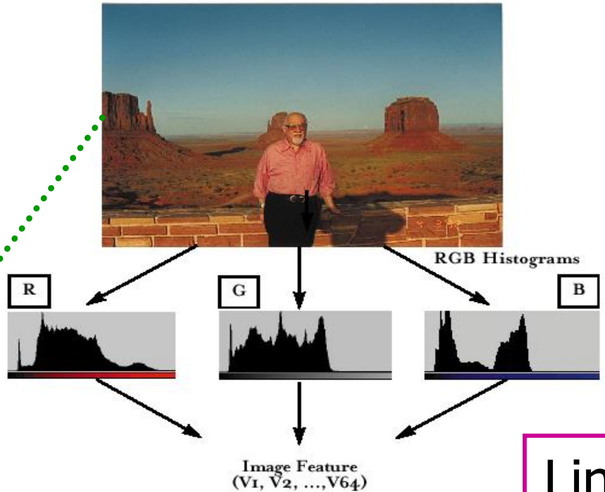
A system with multi-criteria-queries on images:

- **makes use of OR model,**
- **convenient model for image data storage,**
- **bases on new similarity algebra,**
- **query optimization possible,**
- **Protoype.**

Image Data Modeling

Image Data Storage Model

An OR model:



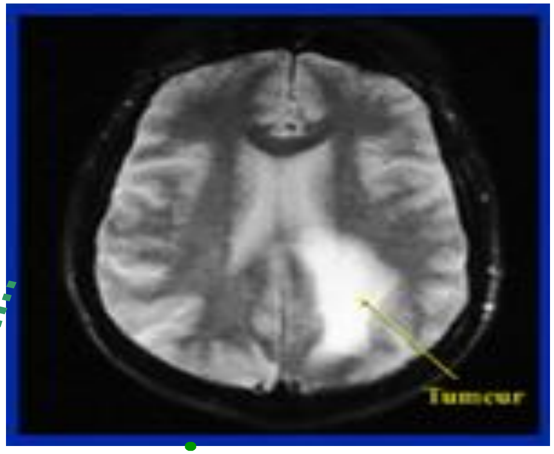
Link to other tables

M(id, O, F, A, P)

Image semantic and contextual data

Salient Object Data Support

Management of salient objects in relation with their source images.



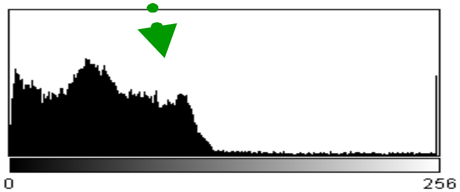
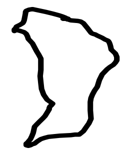


Image Feature
(V1, V2, ….,Vn)



Fv

$S(id_s, F_s, A_s)$

Semantic data

$M(id, O, F, A, P)$

Semantic, Contextual and
Spatial data

Query Example:

- Considering Query 2
  - SI(id, *Surv_Photo*, F, A', P), where A' is a composite object defined as A' = {Date, Time} and *Surv_Photo* is the picture taken by the surveillance camera.
  - EMP(id, Photo, F, A*, P), where A* is a composite extracted from the schema A*(ename, eaddress, Dept, Occupation) and *Photo* is the picture of the employee.
- Query in SQL/MM-like syntax:

```
SELECT *
FROM SI, EMP
WHERE  SI.F ≈ EMP.F
    AND SI.A'.Date = 31-12-1999
    AND SI.A'.time BETWEEN (4:00 AND 6:00PM).
```

- The symbol “≈” is associated with the similarity-based join operation “$\otimes_{\varepsilon}$” *(cf. later)*.

Similarity-based Queries

Definition (ε-similarity)

Given a set of images S, a query image Q, a positive number ε, and the feature vectors f_q and f_s, the ε-similarity is defined as:

$$\varepsilon - similarity(S, q, f_q, f_S, \varepsilon) = X \Leftrightarrow X \subseteq S \wedge \forall x \epsilon X \cdot \parallel q.f_q - x.f_S \parallel \leq \varepsilon.$$

Similarity-based Queries

Definition (k-NN-similarity):

Given a set of images S, a query image Q, a positive number ε, and the feature vectors f_q and f_s, the k-NN-similarity is defined as:

$$k - NN - similarity(S, q, f_q, f_S, k) = X \iff X \subseteq S \wedge Card(X) = k \wedge \forall x \in X; y \in (X \backslash S) \cdot \| x.f_S - q.f_q \| \leq \| y.f_S - q.f_q \|$$

The Similarity-based Algebra
Content-based Retrieval Methods

- Optimization properties

	k-NN	Range Query(ε)
Number of returned images	K	Depends on ε
Setting the values k and ε	Easy	Not easy
Symmetric join operation possible?	No	Yes
Easy to optimize?	No	Yes

Similarity-based Algebra II

- A similarity-based join operator:
  - A binary operator on image tables applied on F,

Definition: for $M_1(id_1, O_1, F_1, A_1, P_1)$ and $M_2(id_2, O_2, F_2, A_2, P_2)$, M_1, M_2, and $\varepsilon > 0$

$$M_1 \otimes^{\varepsilon} M_2 = \{(id_1, o_1, f_1, a_1, p_1') \mid (id_1, o_1, f_1, a_1, p_1) \in M_1$$
$$\wedge\, p'_1 = p_1 \cup (M_2, \{(id_2, \|o_1 - o_2\| < \varepsilon)\})$$
$$\wedge\, p'_1 \neq Null\}$$

Similarity-based Algebra III

- **Problem:** $\otimes^{\varepsilon}$ is non-symmetric
- **Necessary:** A "*Symmetric Similarity-based join operator*" ex.: Query optimization

M'$_1$

id	O	F	A	P
id_1^1	o_1^1	f_1^1	a_1^1	p_1^1
id_1^2	o_1^2	f_1^2	a_1^2	p_1^2
id_1^3	o_1^3	f_1^3	a_1^3	p_1^3
:	:	:	:	:
id_1^n	o_1^n	f_1^n	a_1^n	p_1^n

M_2

Referred id	Sim_Score
id^3_2	5.68
id^5_2	13.05
id^9_2	29.56
id^{13}_2	46.67
id^{25}_2	49.90

$(M_2, \{(id^i_2, sim\_Score^i)\})$

Similarity-based Algebra IV

- **Additive unit**

<u>**Definition (Additive unit)**</u>**: Let $M_1(id_1, O_1, F_1, A_1, P_1)$ and $M_2(id_2, O_2, F_2, A_2, P_2)$ two image tables.**

The *additive unit* of M_1 and M_2 is defined as:

$$M_1 \uplus M_2 = \{(id, o, f, a, p) \mid (id, o, f, a, p) \in M_1 \lor (id, o, f, a, p) \in M_2\}$$

Similarity-based Algebra V

- **Symmetric similarity-based join operator:**

	id	O	F	A	P	
M_1	id_1^1	o_1^1	f_1^1	a_1^1	p_1^1	
	⋮	⋮	⋮	⋮	⋮	
	id_1^j	o_1^j	f_1^j	a_1^j	p_1^j	$\rightarrow (M_2, \{(id^i_2, sim\_Score^i)\})$
	⋮	⋮	⋮	⋮	⋮	
	id_1^n	o_1^n	f_1^n	a_1^n	p_1^n	
M_2	id_2^1	o_2^1	f_2^1	a_2^1	p_2^1	
	⋮	⋮	⋮	⋮	⋮	
	id_2^t	o_2^t	f_2^t	a_2^t	p_2^t	$\rightarrow (M_1, \{(id^k_1, sim\_Score^k)\})$
	⋮	⋮	⋮	⋮	⋮	
	id_2^m	o_2^m	f_2^m	a_2^m	p_2^m	

Definition: $M_1 \oplus^{\varepsilon} M_2 = (M_1 \otimes^{\varepsilon} M_2) \uplus (M_2 \otimes^{\varepsilon} M_1)$

Property:

$\oplus^{\varepsilon}$ is symmetric.

Similarity-based Algebra VI

- The Mine operator: uses the components P_1 of $M_1 \otimes^{\varepsilon} M_2$ and creates table $M_2 \otimes^{\varepsilon} M_1$
  - The cost of Mine is negligible compared to similarity-based operations,
  - Mine is an operator, which uses the features of range queries.

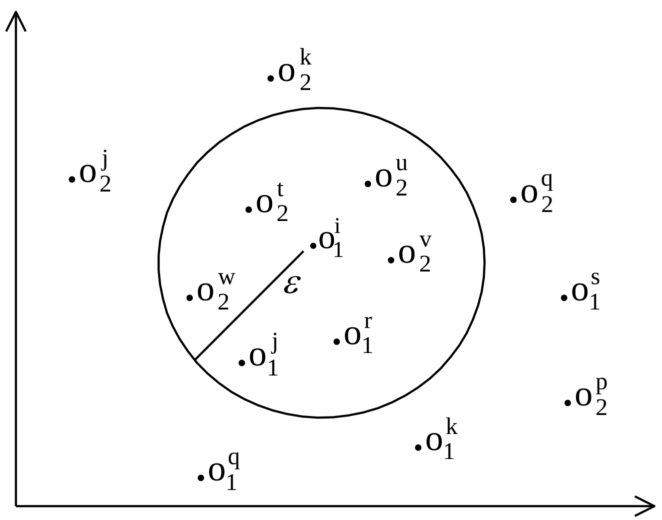
Definition (Mine Operator):

$$\mathit{Mine}(M_1 \otimes^{\varepsilon} M_2) = M_2 \otimes^{\varepsilon} M_1$$

$$\mathit{Mine}(M_2 \otimes^{\varepsilon} M_1) = M_1 \otimes^{\varepsilon} M_2$$

Similarity-based Algebra VII

- **Advantage:**
  - Once $M_1 \otimes^{\varepsilon} M_2$ has been computed, $M_2 \oplus^{\varepsilon} M_1$ can be derived with lower costs,
  - Very useful for query optimization (i.e. a query optimizer can determine which from $M_1 \otimes^{\varepsilon} M_2$ and $M_2 \otimes^{\varepsilon} M_1$ has the lowest cost and let the Mine operator perform the other).



Similarity-based Algebra VIII

- Other relevant content-based operators
  - Multiple similarity-based join,
  - Given two image tables, M1 and M2:
    - asymmetric similarity-based cross product ($\cap^{\varepsilon}$) ,
    - asymmetric similarity-based union ($\cup^{\varepsilon}$),
    - Similarity-based difference ($-^{\varepsilon}$),
    - Cartesian product of two image tables

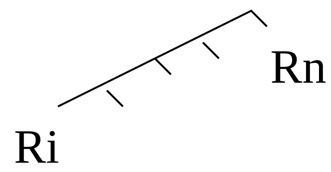
- Given a relational table R and an image table M, one can define:
  - A relational selection from an image table
  - A relational join of two image tables,
  - A relational join of an image table and a relational table

Multimedia Query Optimization

- Traditional optimization techniques.
- Specific problems of multimedia query optimization.
- Algebraic transformation rules for query optimization.

Query Optimization in traditional DB I

- Join order selection
  - R1 $\bowtie$ R2 $\bowtie$ R3 $\bowtie$ … $\bowtie$ Rn
  - Construction of a join tree



  - Dynamic programing
    - Computation of the best plan for each subset of relations
      - Best plan (R1, .., Rn) = min cost plan of (

        R1 $\bowtie$ Best plan(R2, .., Rn)

        R2 $\bowtie$ Best plan(R1, R3, .., Rn)

        ….

        Rn $\bowtie$ Best plan(R1, .., Rn-1))

Query Optimization in traditional DB II

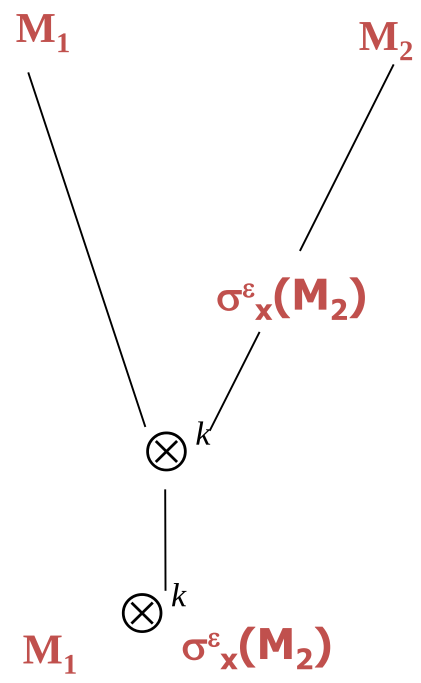
- Selection and projection are reduced to the smallest possible space
- For each join, the order of operators and possible permutations with other joins must be evaluated!
- Sorting order
  - Join may be less costly when the input is first sorted by the join attribute
  - => Best plan(set-of-relations, sort-order)
- Cost-based optimization
  - Costs for join and selection are well-defined: CPU + I/O costs
  - Evaluation of the results of intermediary relations by sampling/histograms
- Heuristic and randomized optimization strategies (ex.: iterative improvement)

Optimization for Multimedia DB

- Expensive predicates/functions in selections/projections/joins
  - Selection based on image manipulations or similarity searches
  - Is the selection more expensive than the join or not?

- The usual heuristic "Move selection predicates to lowest possible level'' does not work in the general case.

Query Optimization: Example Join Tree



New dimension of query optimization: should M_1 be the left or right input of the join?

Similarity-based Query Optimization

- Algebraic transformation rules:
  - Selection

$$\delta^{\varepsilon}_{q}(M_1 \overset{+}{\cup} M_2) = \delta^{\varepsilon}_{q}(M_1) \overset{+}{\cup} \delta^{\varepsilon}_{q}(M_2);$$

$$\delta^{\varepsilon}_{q}(M_1 \otimes^{\varepsilon} M_2) = \delta^{\varepsilon}_{q}(M_1) \otimes^{\varepsilon} M_2;$$

$$\delta^{\varepsilon}_{q}(M_1 \oplus^{\varepsilon} M_2) = \delta^{\varepsilon}_{q}(M_1) \otimes^{\varepsilon} M_2 \overset{+}{\cup} \delta^{\varepsilon}_{q}(M_2) \otimes^{\varepsilon} M_1$$

  - Join

$$M_1 \otimes^{\varepsilon} M_2 \rightarrow \mathrm{Mine}(M_2 \otimes^{\varepsilon} M_1)$$

Inverse the input operators of the join?
-> cost model !

Query Optimization: Range Query

- Approximation of the selectivity of multimedia range queries
  - Nearest-neighbor and range queries are very frequent in MMDBs

  - Performance can be improved by using index structures.
  - The use of the index structure at query processing and the expected resulting selectivity are approximated by a cost model.
    - The selectivity of a nearest-neighbor is trivial: K
    - The selectivity of a range query is a research question!

Mario Döller, Harald Kosch,
IEEE International Conference on Multimedia & Expo,
July 2005, pp:382-385, Amsterdam (The Netherlands)

Query Optimization – Range Query

Related Work I

- Important properties for approximating the selectivity of a range query:
  - Efficiency of the computation of the approximation
  - Exactitude of the approximation
  - Adaptability to other data sets (no assumption on the distribution of the data)
  - Maintainability of the approach wrt. update operations on the data set (Insert/Delete, …)

Query Optimization – Range Queries

Related Work II

- Selectivity approximation for ε-similarity (range queries)

Method	Properties
Sampling	• Random choice of example data from the data set, • Easy to implement, • Bad properties for multidimensional data, • Good results by frequent accesses to hard drives, • Problem: Ratio between best possible exactitude to smallest amount of example data.
Histogram-based approaches	• Split of the data set in fixed buckets and computation of the data density/size. • Problem: Size of histogram increase exponentially with dimensionality. • Good with update operations

Query Optimization – Range Queries

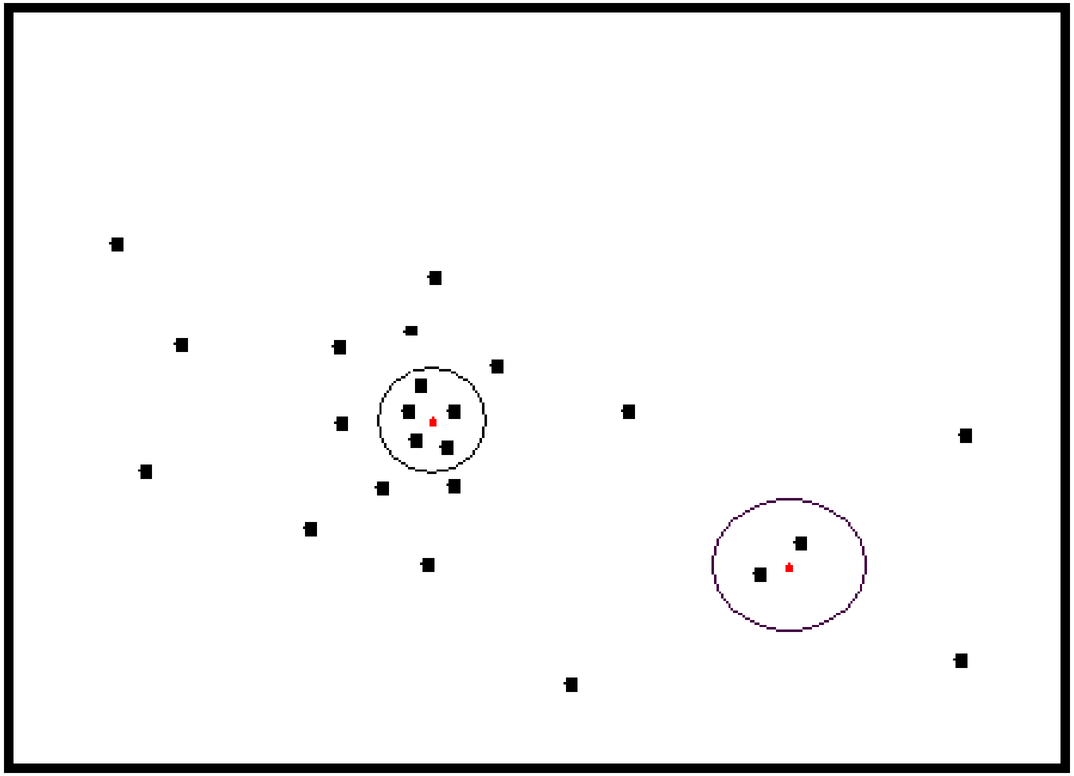
Related Work III

- Selectivity approximation for ε-similarity (range queries)

Method	Properties
Wavelet, DCT, …	• Improvement on histogram approaches • Transformation of data to obtain important coefficients, • Updating a wavelet transformation is difficult, • In order to compute the approximation the transformation must be partially inversed.
Data density methods	• Approximation of the density of the data using clusters/functions etc., • Better exactitude than wavelet approaches, • Efficient for predicting, • High computation time.

Query Optimization – Range Queries

- Observation: the density in the data space determines the selectivity.



Query Optimization – Range Queries

- Algorithm:

1. Cluster data set with a density-based clustering algorithm (ex. DBScan)

2. Determine the MBR (Minimum Bounding Region) for each cluster

3. Compute the density of each cluster C_i with:

$$Density(C) = \frac{\#P\ of\ C}{Vol(C)}$$

where Vol(C) is the *hypervolume* of the MBR and #P is the number of points in the cluster

Query Optimization – Range Queries

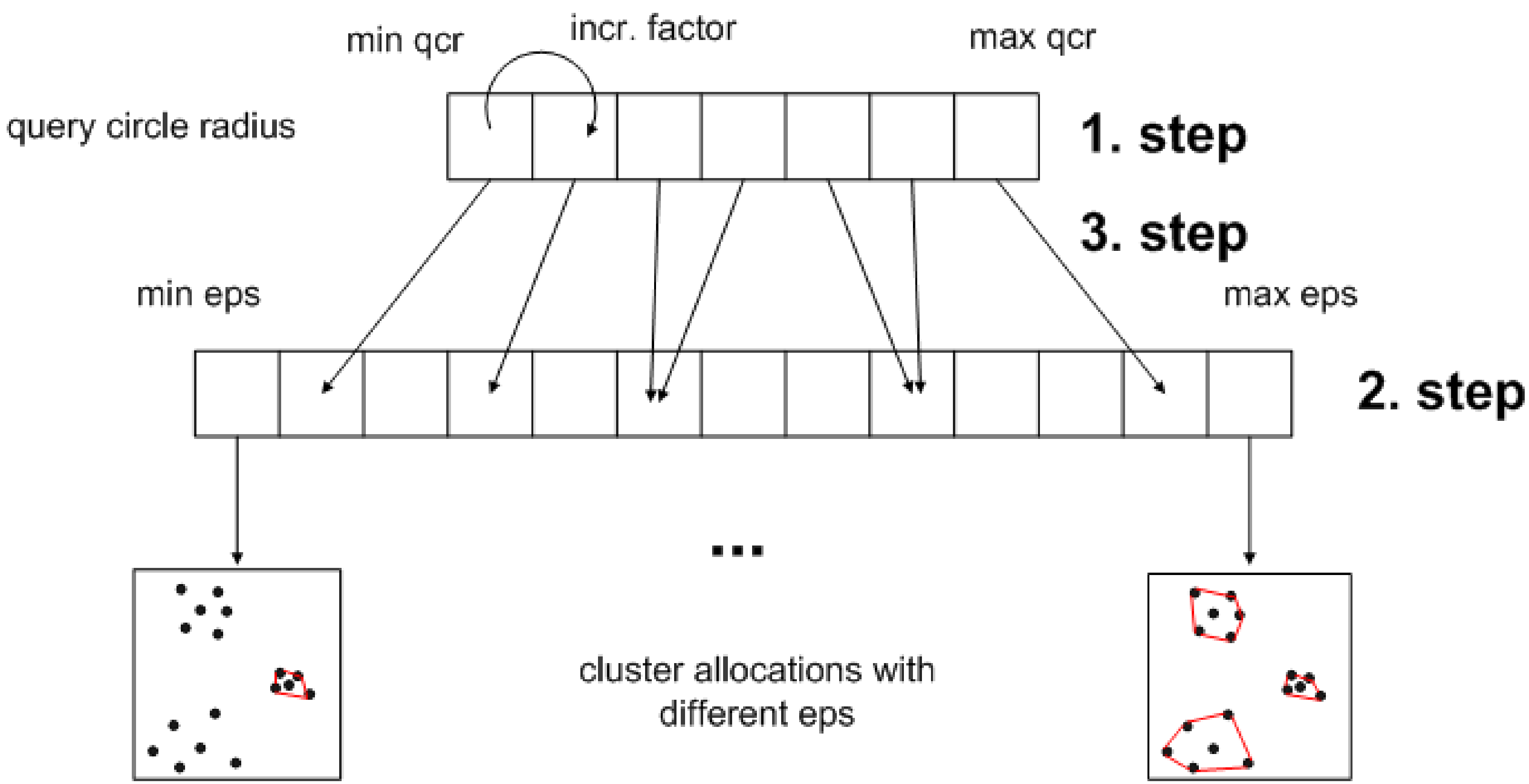
4. Approximate the selectivity of query point Q with radius r with:

$$approx.selectivity = \begin{cases} Density(Ci) * Vol(Q) & , Q \in Area(Ci) \\ MinPts - 1 & , otherwise \end{cases}$$

where Vol(Q) is the *hypervolume* of the enclosing area, defined by Q with radius r

Query Optimization – Range Queries

5. Identify the error-minimizing clustering of a given set with the query radius



Query Optimization – Range Queries

Query:

```
explain plan SET STATEMENT_ID = 'selectivity'
for
select count(*) from location l, city c where (rangeQueryOp(l.location, '50.0 10.0', 2, 0.22) = 1
                                               OR c.name LIKE 'A%')
                                               AND l.cityid = c.id;
```

Two query plans:

without an approximation	with an approximation

QUERY PLAN
------------------------------------ 	
SELECT STATEMENT
  SORT AGGREGATE
    NESTED LOOPS
      TABLE ACCESS FULL CITY
      TABLE ACCESS FULL LOCATION
	QUERY PLAN
	------------------------------------ 	
	SELECT STATEMENT
	  SORT AGGREGATE
	    NESTED LOOPS
	      TABLE ACCESS FULL LOCATION
	      TABLE ACCESS FULL CITY

Query execution times:

89.2s

45.8s with clustering

The End