

# Personalization of Queries in Database Systems

Georgia Koutrika, Yannis Ioannidis

Department of Informatics and Telecommunications  
University of Athens, Hellas

Presented by: Linas Baltrunas and Tenzin Choesang

## 1 Introduction

- Authors
- Summary of the Approach
- Related Approaches

## 2 Preference Selection

- Preference Selection Algorithm
- Preference Integration

## 3 Experimental Results

# Introduction of Authors

## Yannis Ioannidis. University of Athens

- Interests

- ▶ Query Optimization in Database and Information Systems
- ▶ Heterogeneous Systems
- ▶ Human-Computer Interaction Database User Interfaces

- Known as one of the key researcher in Histograms

- CV

- ▶ Department of Informatics, Univ. of Athens, Hellas (Associate Professor)
- ▶ Computer Sciences Department, Univ. of Wisconsin (Associate Professor)
- ▶ 1986 Computer Science, Univ. of California, Berkeley
- ▶ 1983 Applied Mathematics (Computer Science), Harvard University

## Georgia Koutrika

- Interests
  - ▶ various aspects of personalization
  - ▶ user profiling
  - ▶ data mining
- Ph.D. student of Ioannidis

# Motivation of Query Personalization

**Goal: to personalize the query**

Which movie is shown tonight?

```
select MV.title
from  MOVIE MV, PLAY PL
where MV.mid=PL.mid and PL.date='2/7/2003'
```

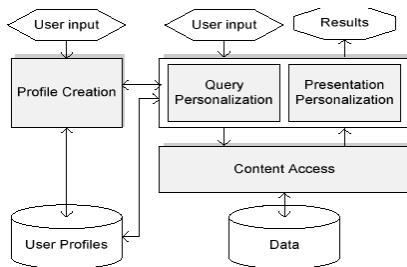
Julie

```
select MV.title
from  MOVIE MV, PLAY PL, GENRE GN
where MV.mid=PL.mid and PL.date='2/7/2003' and
MV.mid=GN.mid and (GN.genre='comedy' or
GN.genre='thriller')
```

Rob

```
select MV.title
from  MOVIE MV, PLAY PL, GENRE GN, CAST CA, ACTOR AC
where MV.mid=PL.mid and PL.date='2/7/2003' and
MV.mid=GN.mid and MV.mid=CA.mid and
CA.mid=AC.mid and (GN.genre='sci-fi' or
AC.name='J. Roberts')
```

# Summary trough Architecture



# Related Approaches

## Content Based Approaches

- **Query-based approaches:** Select content on the basis of query
- **Filter-based approaches:** Select content from stored user profile
- **Personalized approaches:** Select content based on user query and preferences stored in profile

# User Preference Model

- Director.name = "W. Allen"

## **Preference on one imply preference on the other**

- Movie.mid = Directed.mid and Directed.did=Director.did and Director.name="W. Allen"

# Atomic User Preferences

Personalization Graph  $G(V,E)$

## Nodes (G):

- Relation Nodes
- Attribute Nodes
- Value Nodes

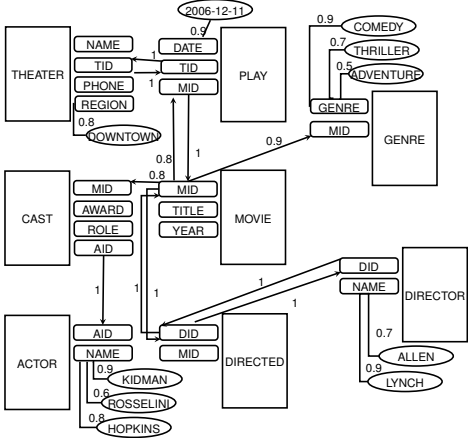
## Edges (E):

- Selection Edges
- Join Edges

# Atomic User Preferences

THEATER.tid=PLAY.tid, 1  
THEATER.REGION="DOWNTOWN", 0.8  
PLAY.tid=THEATER.tid, 1  
PLAY.mid=MOVIE.mid, 1  
MOVIE.mid=PLAY.mid, 0.8  
MOVIE.mid=GENRE.mid, 0.9  
MOVIE.mid=DIRECTED.mid, 1  
MOVIE.mid=CAST.mid, 0.8  
ACTOR.name="HOPKINS", 0.8  
ACTOR.name="KIDMAN",0.9  
ACTOR.name="ROSSELLINI",0.6  
GENRE.genre="COMEDY", 0.9  
GENRE.genre="THRILLER", 0.7  
GENRE.genre="ADVENTURE", 0.5  
DIRECTOR.name="ALLEN", 0.7  
DIRECTOR.name="LYNCH",0.9  
DIRECTED.did=DIRECTOR.did, 1  
DIRECTOR.did=DIRECTED.did, 1  
CAST.AID=ACTOR.AID, 1

# Personalization Graph



# Implicit/Transitive User Preferences

- **Transitive Join:** Between two attribute nodes
- **Transitive Selection:** From an attribute node to a value node. It can be decomposed into a transitive join and an atomic selection.

**Transitive query element is defined as the conjunction of the constituent atomic ones**

# Implicit/Transitive User Preferences

## Transitive Preference

- $D_N = d_i | d_i$ : degree of interest in  $P_i \in P_N, i=1 \dots N$
- $f_{\otimes}(D_N) \leq \min(D_N)$ .
- $f_{\otimes}(D_N) = d_1 d_2 \dots d_N$ .

Where,  $P_N$  is a set of N composable atomic preferences  $D_N$  is corresponding degree of interest

- **Note:** The degree of interest in a transitive preference decreases as the length of the corresponding directed path increases

## Example

- Julie likes actress N. Kidman Actor.name="N. Kidman" 0.9
- then, she also likes movies starring the same actress, which is implicitly expressed as

MOVIE.mid=CAST.mid and  
CAST.aid=ACTOR.aid and  
ACTOR.name="N.Kidman"

Degree of interest on transitive preference is

$$0.8 * 1 * 0.9 = 0.72$$

# Logical Combination of User Preferences

- **Conjunctive Preferences:**  $f_{\wedge}(D_N) \geq \max(D_N)$ .  $f_{\wedge}(D_N) = 1 - (1 - d_1)(1 - d_2) \dots (1 - d_N)$  The degree of interest in multiple preferences satisfied together increases with the number of these preferences
- **Disjunctive Preferences:**  $\min(D_N) \leq f_{\vee}(D_N) \leq \max(D_N)$   
 $f_{\vee}(D_N) = (d_1 + d_2 + \dots + d_N) / N$  The degree of interest in satisfying one of several preference is between the highest and the lowest degree of interest among the original preferences.

## Example: transitive selection on Julies profile

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did  
AND DIRECTOR.name="W. Allen"  
MOVIE.mid=GENRE.mid and GENRE.genre="comedy"

- Comedies directed by W. Allen is  $1 - (1 - 1 * 1 * 0.7)(1 - 0.9 * 0.9) = 0.943$
- Either a comedy or a W. Allen movie is  $(0.7 + 0.81) / 2 = 0.755$

# Theorem

Let  $\Omega$ : Set of all conditions that represent logical combinations of preferences in  $P_N$  according to the personalization model.

- For any  $\omega_1, \omega_2$  in  $\Omega$  with degrees of interest  $d_1$  and  $d_2$ , if  $\text{Result}(\omega_1) \subseteq \text{Result}(\omega_2)$ , then  $d_1 \geq d_2$ .
- This theorem captures the intuition that strictly smaller query answers are of strictly higher interest to the user.

# Preference Selection

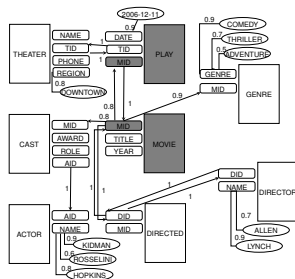
## Parameters for preference selection

- Top K preferences derived from user profile (Defined by  $C_I$ )
- No. Of M preferences that are mandatory constrains ( $0 \leq M \leq K$ )
- No. Of L preferences that should at least be met by the results.

## Extracted preferences has following properties

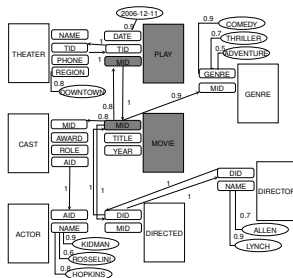
- It is related to a query
- It is not conflicting with a query

# Query Representation as Personalization Graph Subgraph



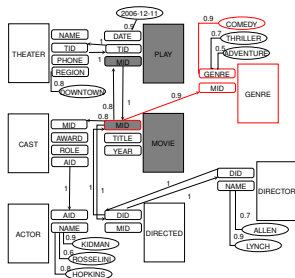
- *SELECT \* FROM MOVIE MV, PLAY PL WHERE MV.MID = PL.MID*
- Query graph is connected

# Related Preferences



- Preference is syntactically related to a query, if it maps to a path that is attached to the query graph.

# Related Preferences



- Preference is syntactically related to a query, if it maps to a path that is attached to the query graph.
- **MOVIE.MID=GENRE.MID AND GENRE.genre="COMEDY"**

# Syntactically Conflicting Preferences

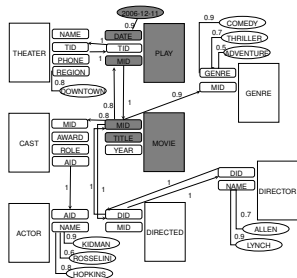
- Preference is syntactically conflicting with a query if it is conflicting with a condition already there.
- Two conditions are conflicting if there is selection condition on the attribute and preference gives different condition on the same attribute
- Example: THEATER.REGION='downtown' conflicts with THEATER.REGION='uptown'

# Preference Selection Algorithm (Idea)

- Take personalization graph and query
- Gradually construct directed paths in decreasing order of their degree of interest
  - ▶ Paths starts from query graph (are syntactically related)
  - ▶ Expands outwards query subgraph

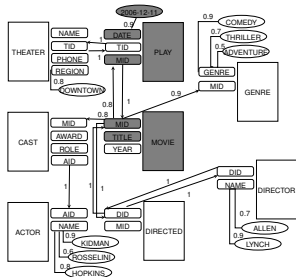


# Preference Selection Algorithm (Example)



- Construct example query as subgraph of personalization graph
- `SELECT mv.title FROM movie, play WHERE movie.mid=pl.mi AND play.date="2006-12-11";`

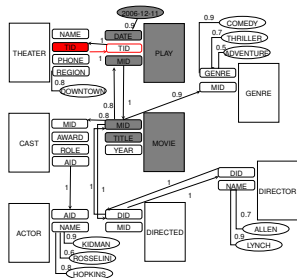
# Preference Selection Algorithm (Example)



## First step of algorithm

- For every atom property check if it is related and not conflicting with query

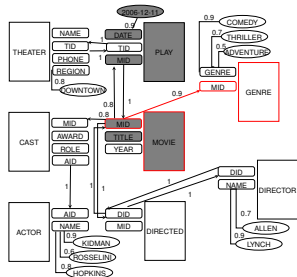
# Preference Selection Algorithm (Example)



## First step of algorithm

- For every atom property check if it is related and not conflicting with query
- **THEATER.tid=PLAY.tid, 1**
- Not related. Remember the order/directions matters!

# Preference Selection Algorithm (Example)



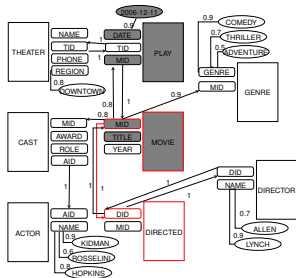
$QP = \{ \text{MOVIE.mid} = \text{GENRE.mid}, 0.9 \}$

## First step of algorithm

- For every atom property check if it is related and not conflicting with query
- **MOVIE.mid=GENRE.mid, 0.9**
- Related. Adding To *QP*



# Preference Selection Algorithm (Example)

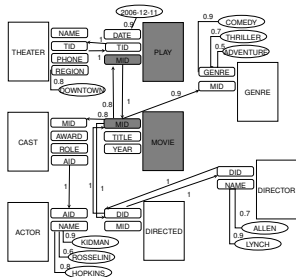


## First step of algorithm

- For every atom property check if it is related and not conflicting with query
- **MOVIE.mid=DIRECTED.mid, 1**
- Related. Added to the *QP*
- *QP* is always **ordered** by interest

$QP = \{ \text{MOVIE.mid} = \text{DIRECTED.mid}, 1 \}$   
 $\{ \text{MOVIE.mid} = \text{GENRE.mid}, 0.9 \}$

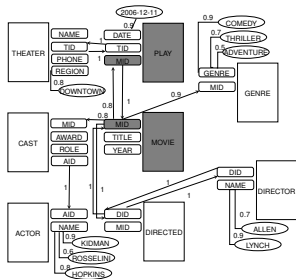
# Preference Selection Algorithm (Example)



Ordered list  $QP$  after first step of algorithm

PLAY.tid=THEATRE.tid, 1  
MOVIE.mid=DIRECTED.mid, 1  
MOVIE.mid=GENRE.mid, 0.9  
MOVIE.mid=CAST.mid, 0.8

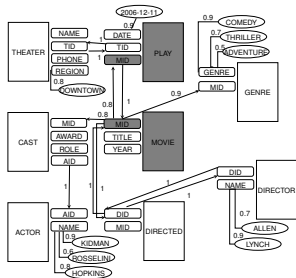
# Preference Selection Algorithm (Example)



Next, algorithm proceeds as follows

- While  $QP$  is not empty:
  - ▶ Take first preference (P) from ordered list  $QP$
  - ▶ Try to expand it with one more atom
  - ▶ If not conflicting, store it into  $QP$

# Preference Selection Algorithm (Example)



$P_K$  :

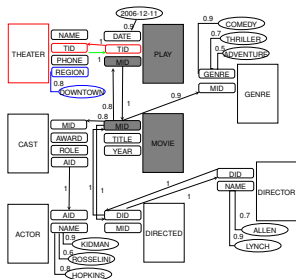
## Join preference expansion

- We take head of  $QP$
- Identify that it is a join (not selection)
- Expand into longer paths which are placed (if satisfies conditions) into  $QP$

$QP$  :

PLAY.tid=THEATRE.tid, 1  
MOVIE.mid=DIRECTED.mid, 1  
MOVIE.mid=GENRE.mid, 0.9  
MOVIE.mid=CAST.mid, 0.8

# Preference Selection Algorithm (Example)



$P_K$  :

## Join preference expansion

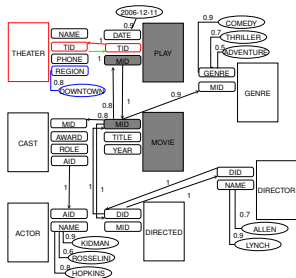
- Preference  $PLAY.tid=THEATRE.tid,1$
- Expanded with tow atoms  $THEATER.tid=PLAY.tid,$   
 $THEATER.REGION="DOWNTOWN"$
- Into  $\{PLAY.tid=THEATRE.tid \text{ AND } THEATER.tid=PLAY.tid, 1*1=1\}$   
 $\text{ AND } \{PLAY.tid=THEATRE.tid \text{ AND } THEATER.REGION="DOWNTOWN", 1*0.8=0.8\}$
- $\{PLAY.tid=THEATRE.tid \text{ AND } THEATER.tid=PLAY.tid, 1\}$  **Dropped**, because expands to a relation belonging to  $QP$  (Also Cycles are formed)

$QP$  :

$PLAY.tid=THEATRE.tid, 1$   
 $MOVIE.mid=DIRECTED.mid, 1$   
 $MOVIE.mid=GENRE.mid, 0.9$   
 $MOVIE.mid=CAST.mid, 0.8$

# Preference Selection Algorithm (Example)

## Join preference expansion



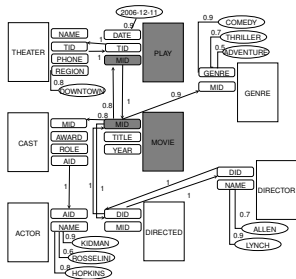
$P_K$  :

- Preference **PLAY.tid=THEATRE.tid,1**
- Expanded with tow atoms THEATER.tid=PLAY.tid, THEATER.REGION="DOWNTOWN"
- Into **{PLAY.tid=THEATRE.tid AND THEATER.tid=PLAY.tid, 1\*1=1}** AND **{PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 1\*0.8=0.8}**
- **{PLAY.tid=THEATRE.tid AND THEATER.tid=PLAY.tid, 1}** **Dropped**, because expands to a relation belonging to *QP* (Also Cycles are formed)
- **{PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8}** is stored to *QP*

*QP* :

PLAY.tid=THEATRE.tid, 1  
 MOVIE.mid=DIRECTED.mid, 1  
 MOVIE.mid=GENRE.mid, 0.9  
 MOVIE.mid=CAST.mid, 0.8  
**PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8**

# Preference Selection Algorithm (Example)



*QP* after first iteration we have *QP* :

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did, 1  
 MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81  
 MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID, 0.8  
 PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8  
 MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63  
 MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45

$P_K$  :

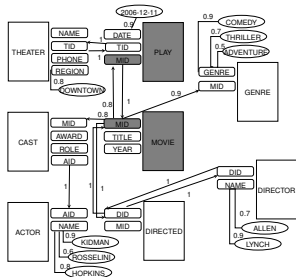




# Preference Selection Algorithm (Example)

## Selection preference expansion

- And now first time we have **selection preference**
- {MOVIE.mid=DIRECTED.mid AND  
DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="LYNCH", 0.9}

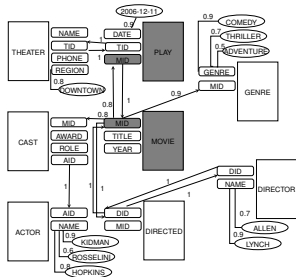


$P_K$ :

*QP*:

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="LYNCH", 0.9  
MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81  
MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID, 0.8  
PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8  
MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="ALLEN", 0.7  
MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63  
MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45

# Preference Selection Algorithm (Example)



$P_K$  :

MOVIE.mid=DIRECTED.mid AND  
DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="LYNCH"

## Selection preference expansion

- And now first time we have **selection preference**
- {MOVIE.mid=DIRECTED.mid AND  
DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="LYNCH",0.9}
- It satisfies CI (preference with degree of interest > 0.43) criteria and we add it to  $P_K$ !

$QP$  :

MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81  
MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID, 0.8  
PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8  
MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND  
DIRECTOR.name="ALLEN", 0.7  
MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63  
MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45

# Preference Selection Algorithm (Example)

## Operation we do

Move `MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY"`, 0.81  $P_K$

$QP$  :

`MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY"`, 0.81

`PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN"`, 0.8

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN"`, 0.7

`MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER"`, 0.63

`MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE"`, 0.45

$P_K$  :

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH"`, 0.9

# Preference Selection Algorithm (Example)

## Operation we do

Expand `MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID, 0.8`

$QP$  :

`MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID, 0.8`

`PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8`

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7`

`MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63`

`MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45`

$P_K$  :

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9`

`MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81`

# Preference Selection Algorithm (Example)

## Operation we do

Move `PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8` to  $P_K$

$QP$  :

`PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8`

`MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID, 0.8`

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7`

`MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63`

`MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45`

$P_K$  :

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9`

`MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81`

# Preference Selection Algorithm (Example)

## Operation we do

**Expand** `MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID, 0.8`

$QP$  :

`MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID, 0.8`

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7`

`MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63`

`MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45`

$P_K$  :

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9`

`MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81`

`PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8`

# Preference Selection Algorithm (Example)

## Operation we do

Expand `MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID, 0.8` Note, that expansion of above lead to three terms. But,  $CI$  is set to accept  $P$  with degrees  $>0.43$  **Discard** `MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID`

`AND ACTOR.name="ROSSELLINI", 0.42`

$QP$  :

`MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="KIDMAN", 0.72`

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7`

`MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="HOPKINS", 0.64`

`MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63`

`MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45`

$P_K$  :

`MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9`

`MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81`

`PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8`

# Preference Selection Algorithm (Example)

## Operation we do

What is left are **selection** preferences. Move all to  $P_K$

$QP$  :

MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="KIDMAN", 0.72  
MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7  
MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="HOPKINS", 0.64  
MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63  
MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45

$P_K$  :

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9  
MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81  
PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8

# Preference Selection Algorithm (Example)

Operation we do

Return  $P_K$

$QP$  :

$P_K$  :

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="LYNCH", 0.9

MOVIE.mid=GENRE.mid AND GENRE.genre="COMEDY", 0.81

PLAY.tid=THEATRE.tid AND THEATER.REGION="DOWNTOWN", 0.8

MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="KIDMAN", 0.72

MOVIE.mid=DIRECTED.mid AND DIRECTED.did=DIRECTOR.did AND DIRECTOR.name="ALLEN", 0.7

MOVIE.mid=CAST.mid AND CAST.AID=ACTOR.AID AND CAST.AID=ACTOR.AID AND ACTOR.name="HOPKINS", 0.64

MOVIE.mid=GENRE.mid AND GENRE.genre="THRILER", 0.63

MOVIE.mid=GENRE.mid AND GENRE.genre="ADVENTURE", 0.45

# Preference Integration

## Single Query

```
SELECT DISTINCT MV.title
FROM MOVIE MV, PLAY PL, CAST CA, ACTOR AC,
GENRE GN, DIRECTED DD, DIRECTOR DI
WHERE MV.mid=PL.mid and PL.date="2/7/2003" and
(
((MV.mid=GN.mid and GN.genre="comedy"
and MV.mid=CA.mid and CA.aid=AC.aid
and AC.name="N. Kidman")) or
((MV.mid=CA.mid and CA.aid=AC.aid
and AC.name="N. Kidman"
and MV.mid=DD.mid and DD.did=DI.did
and DI.name="D. Lynch")) or
((MV.mid=GN.mid and GN.genre="comedy"
and MV.mid=DD.mid and DD.did=DI.did
and DI.name="D. Lynch"))))
```

- Conjunction of the mandatory conditions
- Disjunction of all possible conjunctions of L conditions from the remaining K-M ones

# Preference Integration

## Multiple Queries

```
SELECT MV.title
FROM ((SELECT DISTINCT MV.title
FROM MOVIE MV, PLAY PL, MGENRE GN
WHERE MV.mid=PL.mid and
PL.date="2/7/2003" and
MV.mid=GN.mid and
GN.genre="comedy")
UNION ALL
( SELECT DISTINCT MV.title
FROM MOVIE MV, PLAY PL,
CAST CA, ACTOR AC
WHERE MV.mid=PL.mid and
PL.date="2/7/2003" and
MV.mid=CA.mid and
CA.aid=AC.aid and
AC.name="N. Kidman")
UNION ALL
( SELECT DISTINCT MV.title
FROM MOVIE MV, PLAY PL,
DIRECTED DD, DIRECTOR DI
WHERE MV.mid=PL.mid and
PL.date="2/7/2003" and
MV.mid=DD.mid and
DD.did=DI.did and
DD.name="D. Lynch")) TEMP
GROUP BY MV.title
HAVING count(*) >= 2
```

- A set of K-M queries, each one containing a simpler qualification, which is a conjunction of the mandatory conditions, and one condition from the remaining K-M ones.

# Evaluation of Query Personalization Framework

## Implementation issues

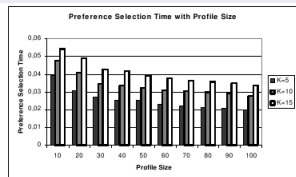
- Implementation is done on top of Oracle 9i
- Data comes from Internet Movies Database (imdb.com). Information about 340000 movies
- Profiles are partially generated, partially extracted from data

## Experimentation type

- Experiments show computational costs (Off-line evaluation)
- Do not provide On-line evaluation

# Effect of Profile Size on Preference Selection Time

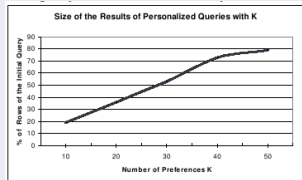
## Execution time dependency on profile size



- Bigger profile, faster computations!
  - ▶ Top k atomic preferences are more likely to be integrated
  - ▶ Do not need to compute other preferences
- Smaller K, faster computations.

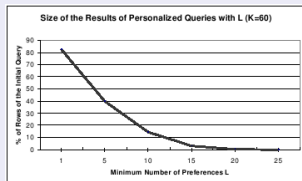
# Size of the Results of Personalized Queries

## Size of the results of personalized queries with K



- Unintuitive!
- Increasing number of preferences K, gives more data

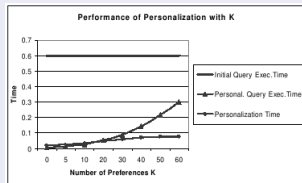
## Size of the Results of Personalized Queries with L (K=60)



- Bigger L, less data.
  - ▶ More preferences should be fulfilled. More constraints.

# Performance of Personalization

## Execution time dependency on profile size



- Faster with personalization???
- Time of transferring data takes more than algorithm execution
- Raw query
  - ▶ **Select \* FROM MOVIE;**
  - ▶ **A LOT OF DATA** to access/transfer
- Personalization
  - ▶ **Select \* FROM MOVIE WHERE ... AND ... AND ... AND ...;**
  - ▶ **LITTLE OF DATA** to access/transfer

Algorithm + Execution of complex query **takes less time** than transfer a lot of data

# Our Opinion

- It is nice theoretical approach which gives effortless integration of profile to the query
- Only off-line evaluation
- **No serendipity.** Only profile specific results will be given.