Model Management and Schema Mappings: Theory and Practice

Phil Bernstein
Microsoft Research

Howard Ho
IBM Research

September 27, 2007

Microsoft Research

Part One

Phil Bernstein
Microsoft Research

September 27, 2007

Data Programmability

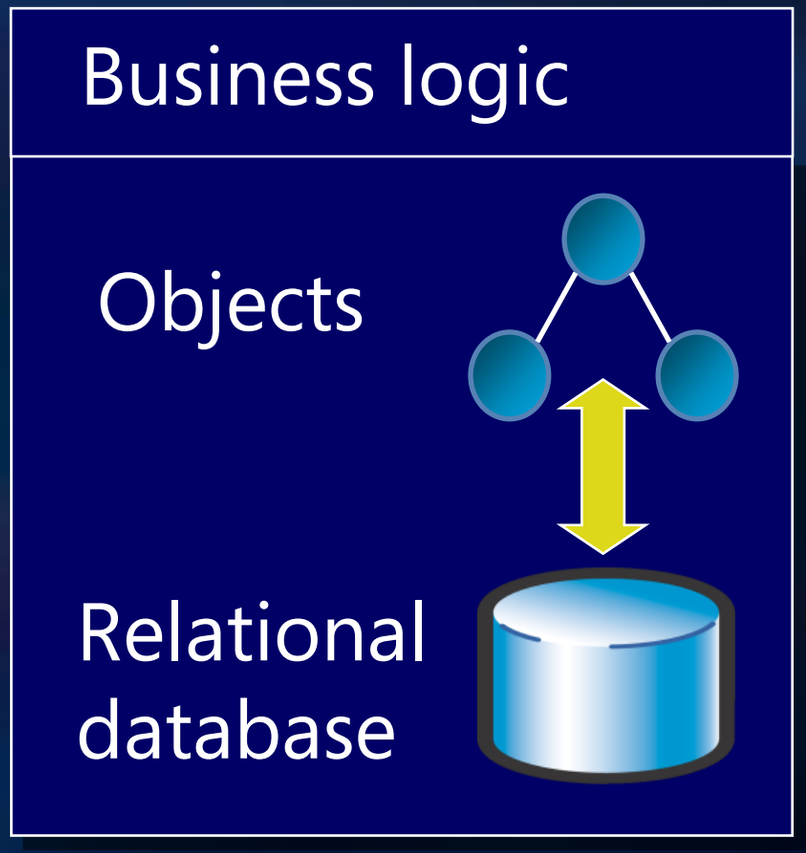
- Make it easier to write programs that access databases
- Traditionally, for large IT departments
- Much progress, but it's still ~40% of the work
- Core problem is developing and using complex mappings between schemas

Mapping Problems are Pervasive
And it's a Growth Industry

- Data translation
- XML message mapping
- Data warehouse loading
- Query mediators
- Forms managers
- Report writers
- Query designers
- Object-relational wrappers
- Portal generation from DB
- OLAP databases
- Application integration
- Composing web services

Object-Relational Wrappers

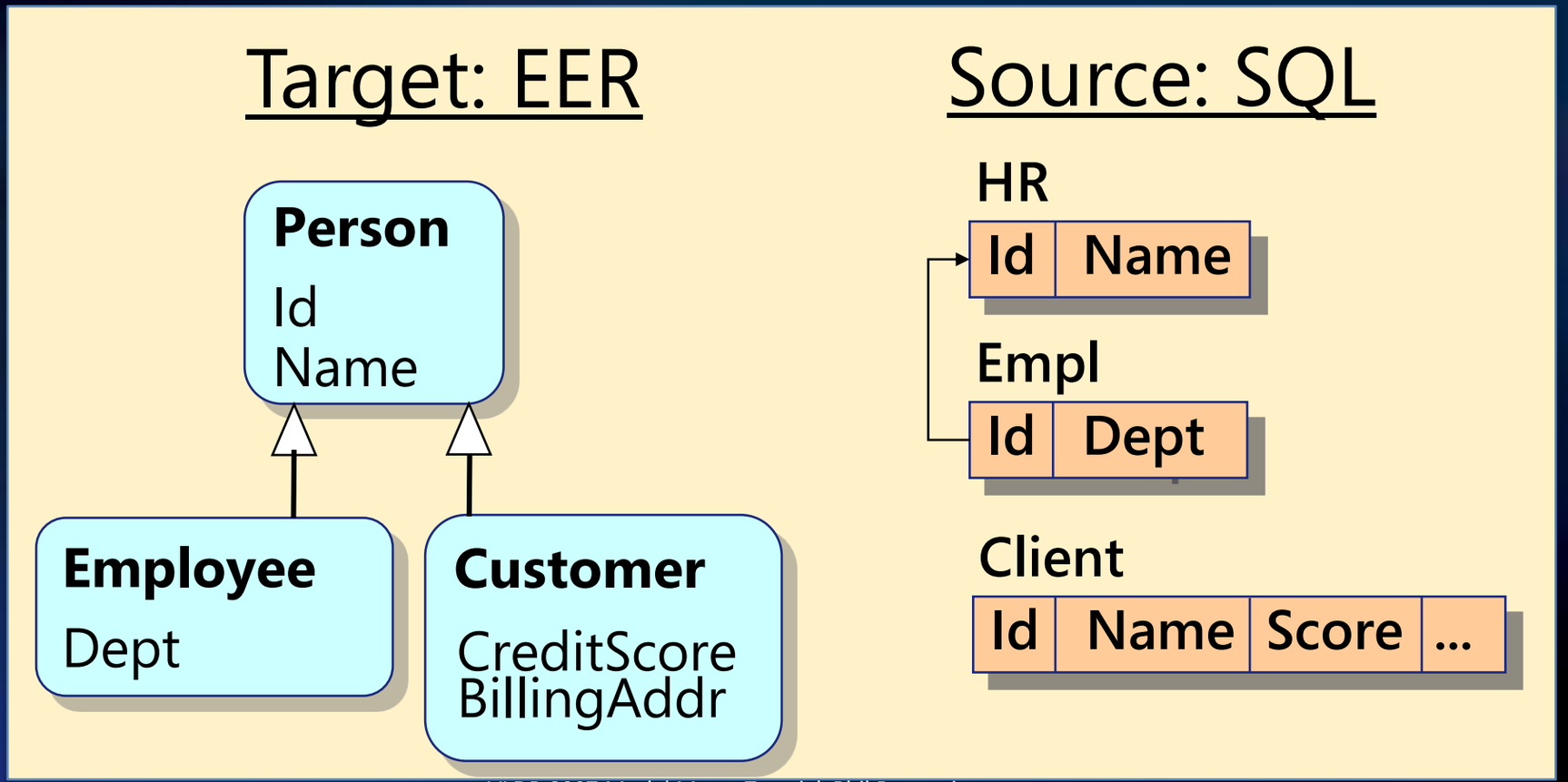
- Most packaged business apps need to access an OO view of relational data
- Requires an OR wrapper
- App developer specifies a high-level mapping
- A tool translates the mapping into executable code



An Example Mapping

- Person = HR $\cup$ $\pi_{ID,Name}$(Client)
- Employee = HR $\bowtie$ Empl
- Customers = Client

Specified by app developer



Executable Code for Persons

[Melnik, Adya, Bernstein, SIGMOD 07]

```
SELECT VALUE
    CASE
      WHEN (T5._from2 AND NOT(T5._from1))  THEN Person(T5.Person_Id, T5.Person_Name)
      WHEN (T5._from1 AND T5._from2)
        THEN Employee(T5.Person_Id, T5.Person_Name,  T5.Employee_Dept)
        ELSE Customer(T5.Person_Id, T5.Person_Name,  T5.Customer_CreditScore,
             T5.Customer_BillingAddr)
    END
FROM (  (SELECT T1.Person_Id, T1.Person_Name,  T2.Employee_Dept,
                CAST(NULL AS SqlServer.int) AS Customer_CreditScore,
                CAST(NULL AS SqlServer.nvarchar) AS Customer_BillingAddr, False AS _from0,
                (T2._from1 AND T2._from1 IS NOT NULL) AS _from1,  T1._from2
          FROM  ( SELECT  T.Id AS Person_Id, T.Name AS Person_Name, True AS _from2
                  FROM HR AS T) AS T1
                LEFT OUTER JOIN (
                  SELECT  T.Id AS Person_Id,  T.Dept AS Employee_Dept, True AS _from1
                  FROM dbo.Empl AS T) AS T2
                ON T1.Person_Id = T2.Person_Id )
          UNION ALL (
                SELECT  T.Id AS Person_Id,  T.Name AS Person_Name,
                        CAST(NULL AS SqlServer.nvarchar) AS Employee_Dept,
                        T.Score AS Customer_CreditScore,  T.Addr AS Customer_BillingAddr,
                        True AS _from0,  False AS _from1,  False AS _from2
                FROM Client AS T)
    ) AS T5
```

The Theme

- The main benefit
  - It's easier to design mappings than to write code
- The main problems
  - Help the user develop mappings
  - Translate mappings into code

Outline

- ✓ Motivation
- Solution Space
- Model Management 1.0
- Model Management 2.0
- Operators & Scenarios

Most slides come from SIGMOD 07 Keynote, SIGMOD 07 “Bridging Apps & DB”, and ICDE 2004 tutorial, all joint with Sergey Melnik

Metadata-Speak

Metadata-Speak	English Example
meta-meta-model = meta- meta-meta data	Built-in types (usually hard-coded)
metamodel = meta-meta data	Schema for "Table," "Column", "Key," …
model = meta data	Schema for the Employee Table
data	Employee Table

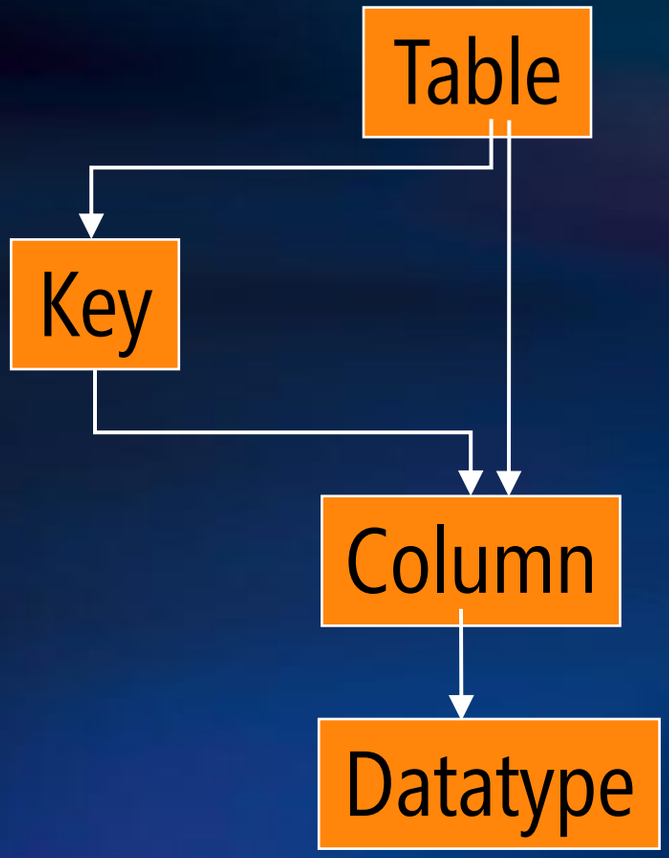
Solution Template (1)

1 Get a data manager for models and mappings

- Usually, it's an object manager
  - OO programming language, OODB, etc.
- Hence, meta-metamodel is the object manager's built-in types
  - Classes, attributes, methods, objects
  - Plus operators to manipulate them, such as NewClass, NewAttribute, NewObject, WriteAttribute

Solution Template (2)

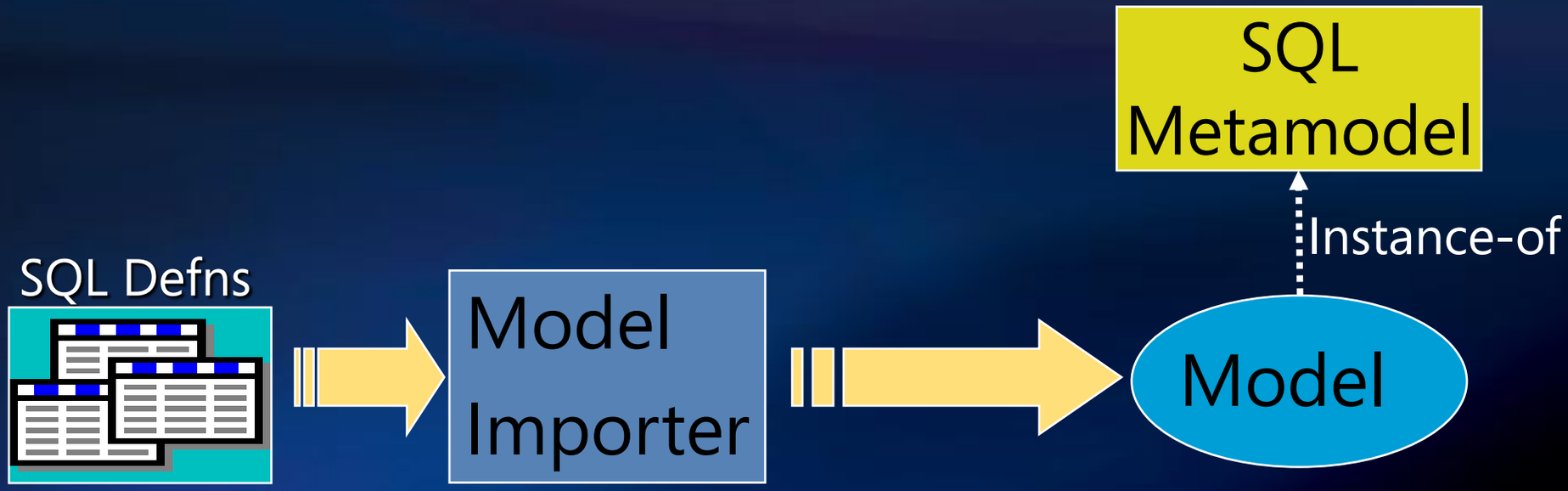
1. Get a data manager for models and mappings
2. Design metamodel(s) (e.g., for SQL schemas)



If the meta-metamodel is OO, then the metamodel consists of class definitions

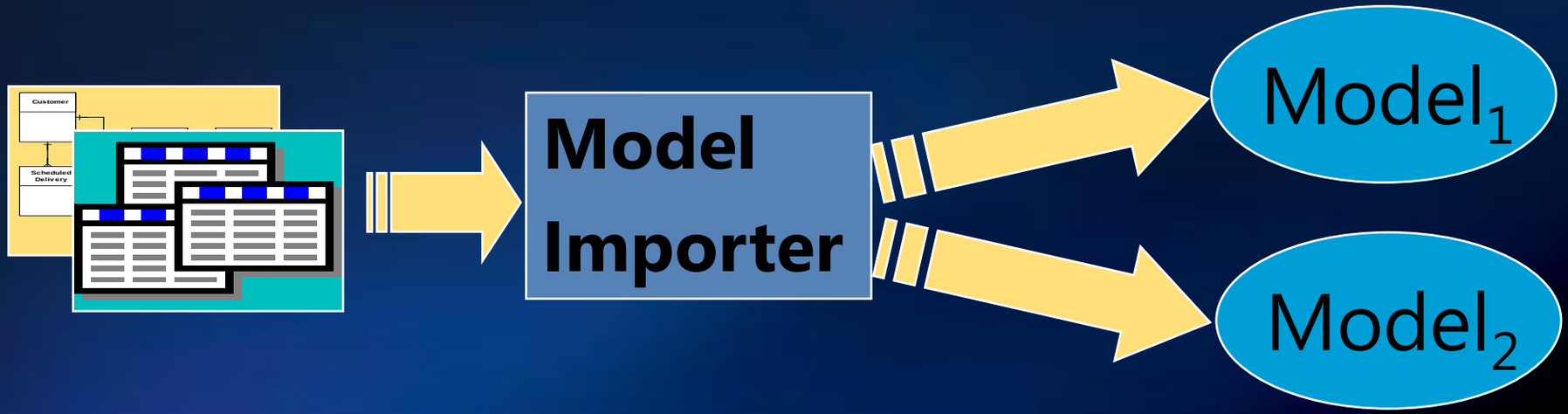
Solution Template (3)

1. Get a data manager for models and mappings
2. Design metamodel(s) (e.g., for SQL schemas)
3. Build a model importer for each metamodel



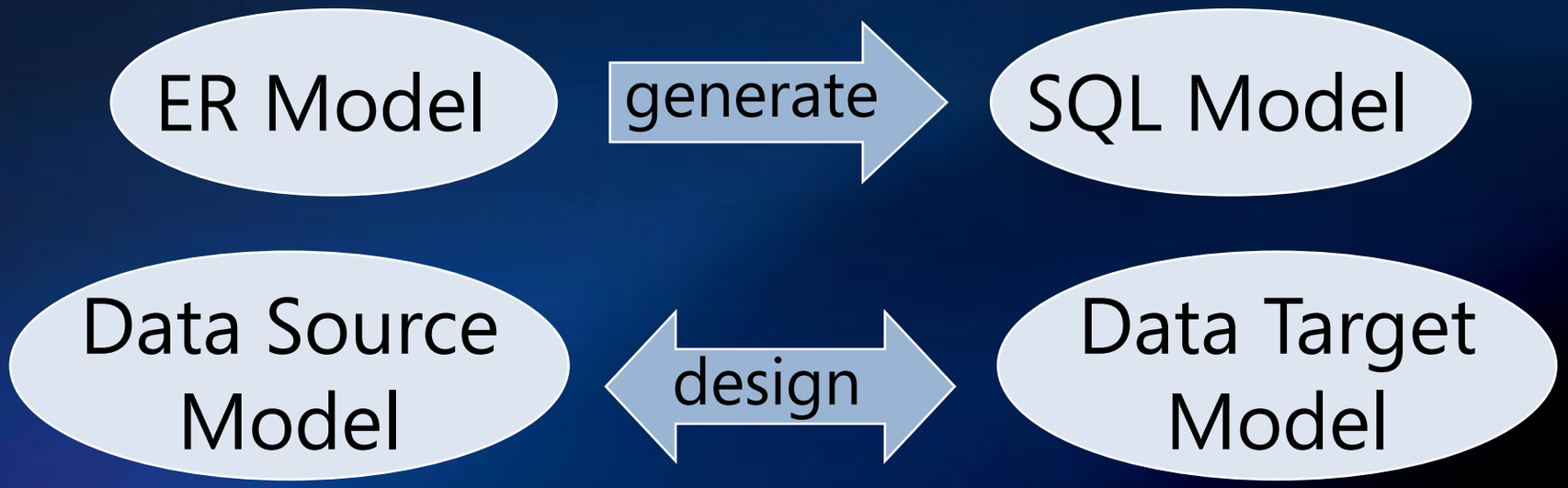
Solution Template (4)

1	Get a data manager for models and mappings
2	Design metamodel(s) (e.g., for SQL schemas)
3	Build a model importer for each metamodel
4	Invoke model importer(s)



Solution Template (4)

1	Get a data manager for models and mappings
2	Design metamodel(s) (e.g., for SQL schemas)
3	Build a model importer for each metamodel
4	Invoke model importer(s)
5	Generate or design mappings



Solution Template (5)

1	Get a data manager for models and mappings
2	Design metamodel(s) (e.g., for SQL schemas)
3	Build a model importer for each metamodel
4	Invoke model importer(s)

Problem	**$Model_1$**	**$Model_2$**
OR Mapping	object schema	relational schema
Data translation	source schema	target schema
Msg translation	source format	target format
DW loading	source schema	DW schema

Solution Template (6)

1	Get a data manager for models and mappings
2	Design metamodel(s) (e.g., for SQL schemas)
3	Build a model importer for each metamodel
4	Invoke model importer(s)
5	Generate or design mappings
6	Transform mappings into executable code

- SQL views, XSLT, ETL programs, etc.

Why is mapping hard?

[Haas, ICDT 07]

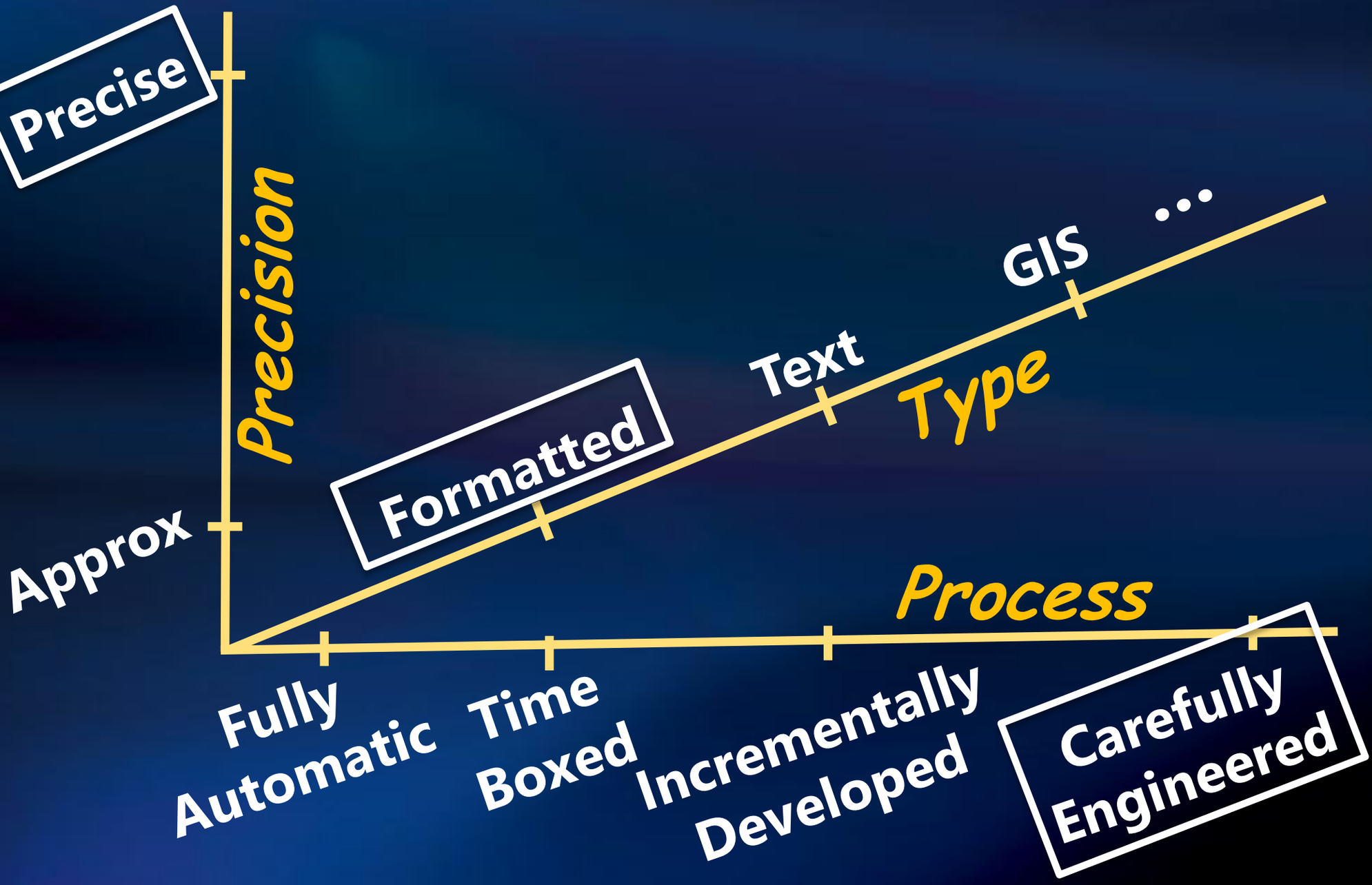
- Heterogeneity
- Impedance mismatch
- Insufficient abstraction
- Potpourri of tools

And It's Getting Harder

- More data models
  - Java, ODMG, XSD, .NET
  - RDF, OWL, EDM, SML
- More programming languages
- More types of tools
- More schema sources
  - Web site wrappers
  - Google Base
  - Generic info extractors [Gubanov, Bernstein WebDB 06]

Mapping Space

Info Integration Workshop
http://db.cis.upenn.edu/iiworkshop/



Outline

- ✓ Motivation
- ✓ Solution Space
- Model Management 1.0
- Model Management 2.0
- Operators & Scenarios

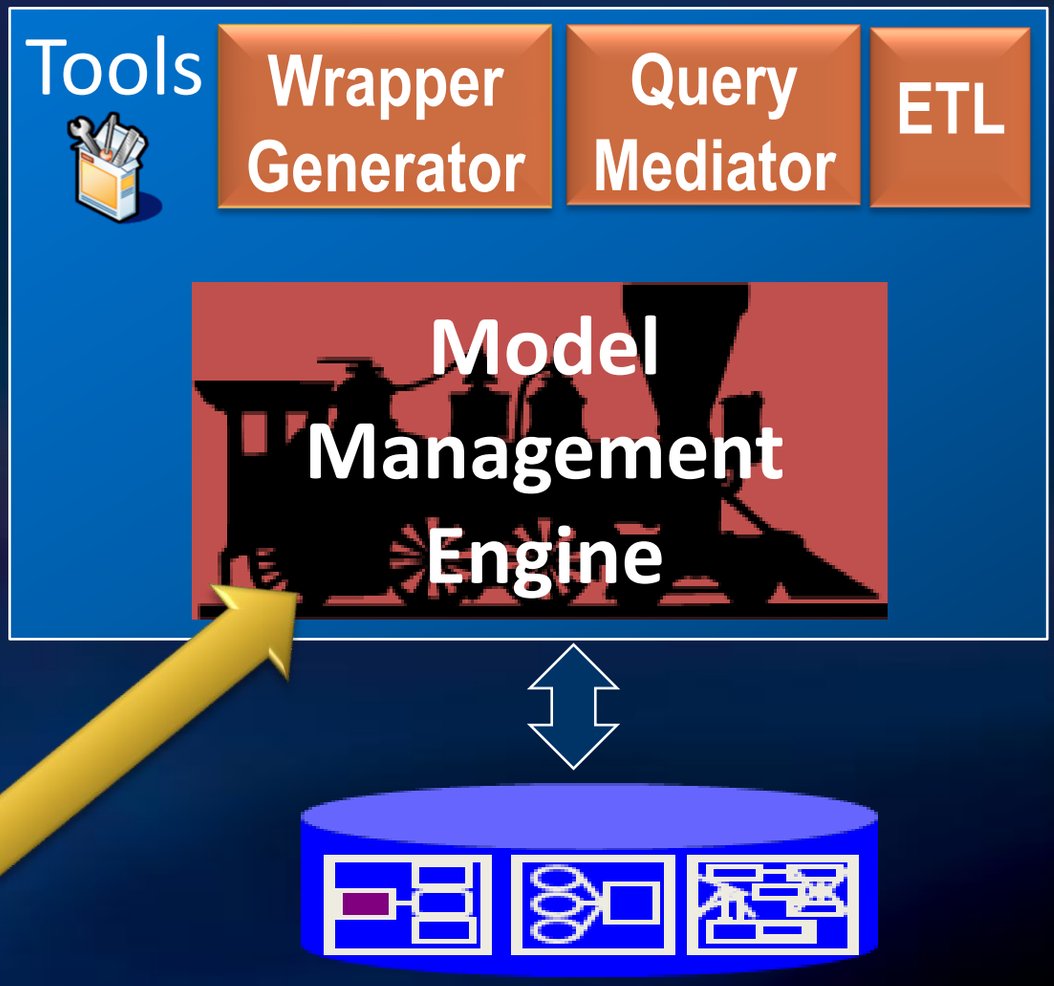
Model Management 1.0

[Bernstein, Halevy, Pottinger SIGMOD Record 00]

Manipulate
models & mappings
as bulk objects

Meta-model independent
- relational, ER, OO, XML, RDF, OWL, SML, …

Operations
- Match
- Compose
- Merge
- Diff/Extract
- ModelGen
- Inverse



Model Management
Getting Started

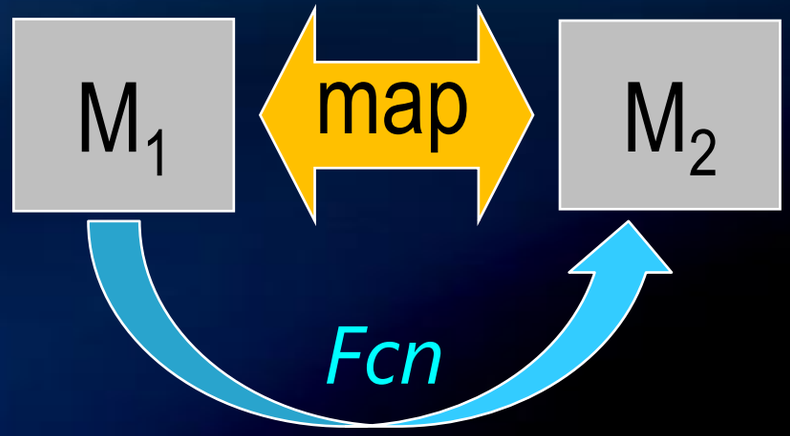
- Choose a schema definition language
- Choose a mapping language

Model Mgmt Operators

map = Match(M_1 , M_2)

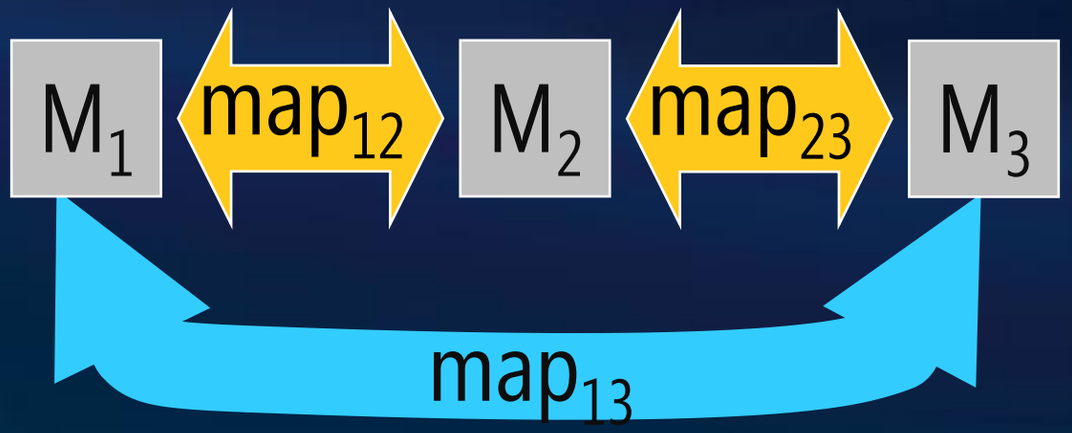
⟨M_2 , *map*⟩ =
ModelGen(M_1 , *metamodel*$_2$)

Fcn = TransGen(*map*)

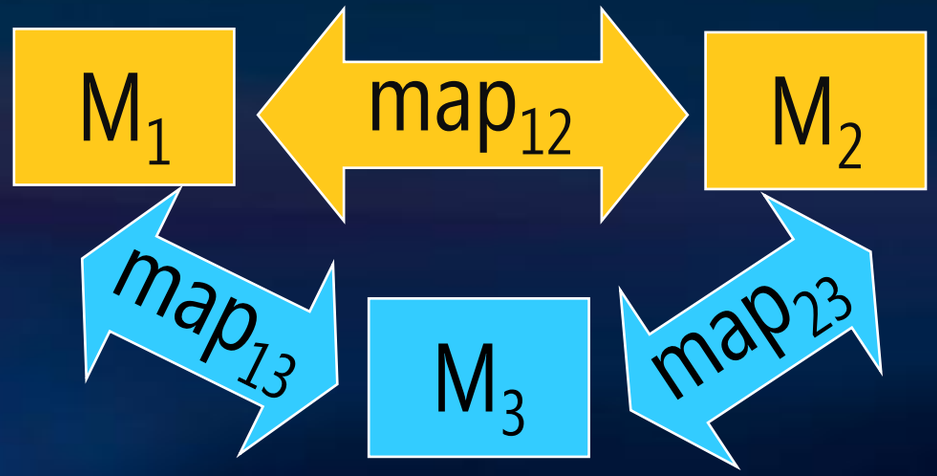


Model Mgmt Operators (cont'd)

Compose:
$map_{13} = map_{12} \bullet map_{23}$



$\langle M_3 , map_{13} , map_{23} \rangle$ =
Merge (M_1 , M_2 , map_{12})



$\langle M_2 , map_2 \rangle$ =
Diff(M_1 , map_1)

Plus a few more …

Model Management 1.0 Benefits

- More research focus on primary operations
  - More powerful operations
  - Hence better tools
- More leverage from tool investments
- More uniform behavior across tools

Good News / Bad News

- Good News
  - Lots of progress on operations
  - Some practical applications
  - A lot has been learned
- Bad News
  - Still waiting for the first reasonably-complete practical implementation
- Good news
  - A lot of research left to do

Outline

- ✓ Motivation
- ✓ Solution Space
- ✓ Model Management 1.0
- Model Management 2.0
  - What has changed: Use richer mappings
  - What has changed: Include the runtime
- Operators & Scenarios

What Has Changed?
Use Richer Mappings

2000
Structural mappings

- Mappings are structural
  - Relate schemas, *not* data
- Operations oblivious to mapping semantics
- Semantics is a plug-in

2007
Semantic mappings

- Mappings are semantic
  - Relate schemas *and* data
- Operations sensitive to mapping semantics
- Semantics is built-in

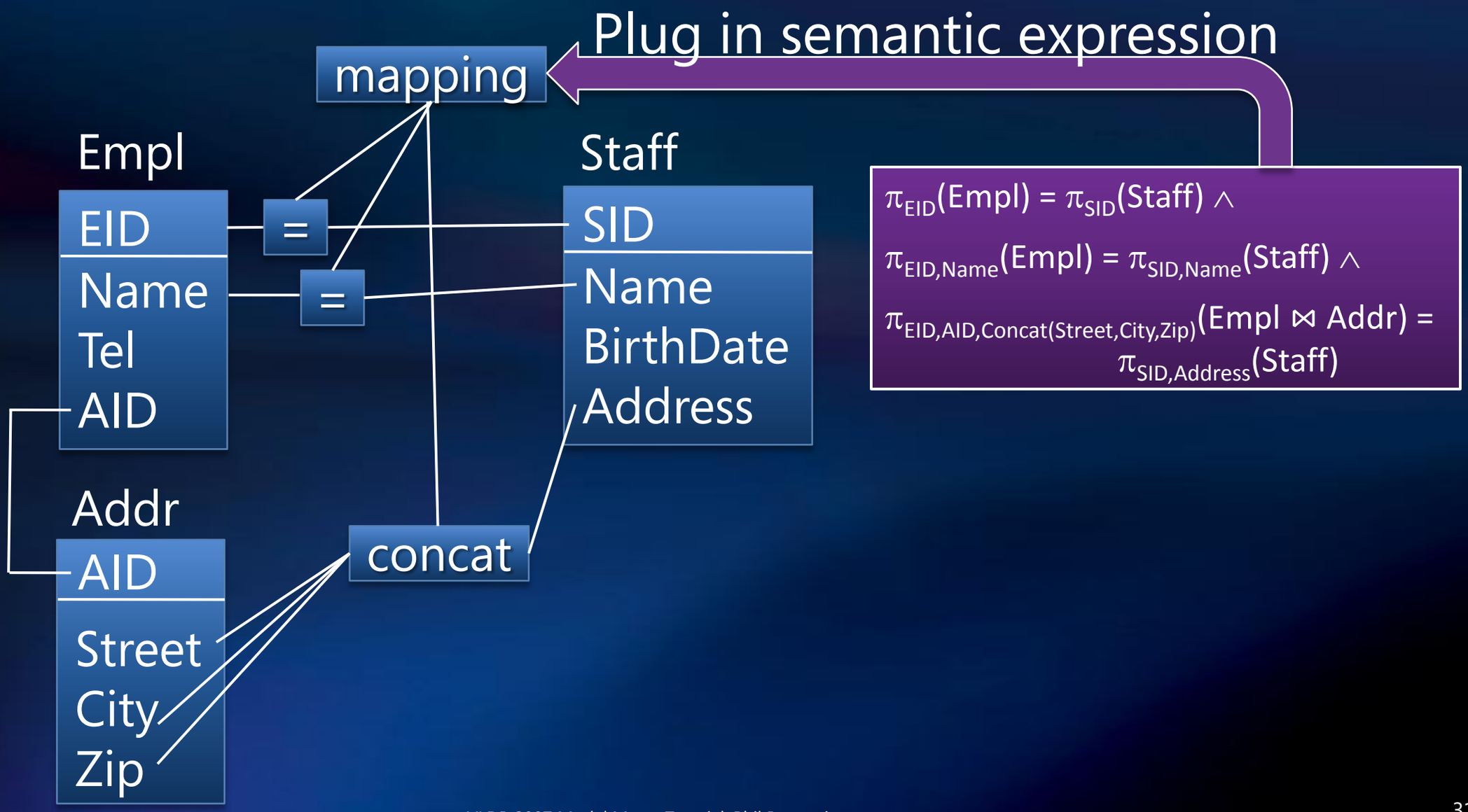
Semantic Mapping

- $\mathbb{I}(S_1)$ are the instances of schema S_1
  - Each d in $\mathbb{I}(S_1)$ is a database (e.g., a set of relations)
- $\mathbb{I}(S_2)$ are the instances of schema S_2
- $\text{map}_{12} \subseteq \mathbb{I}(S_1) \times \mathbb{I}(S_2)$
- Usually, we represent a mapping by an expression
  - V= R ⋈ S
  - R ⋈ S = T ⋈ U

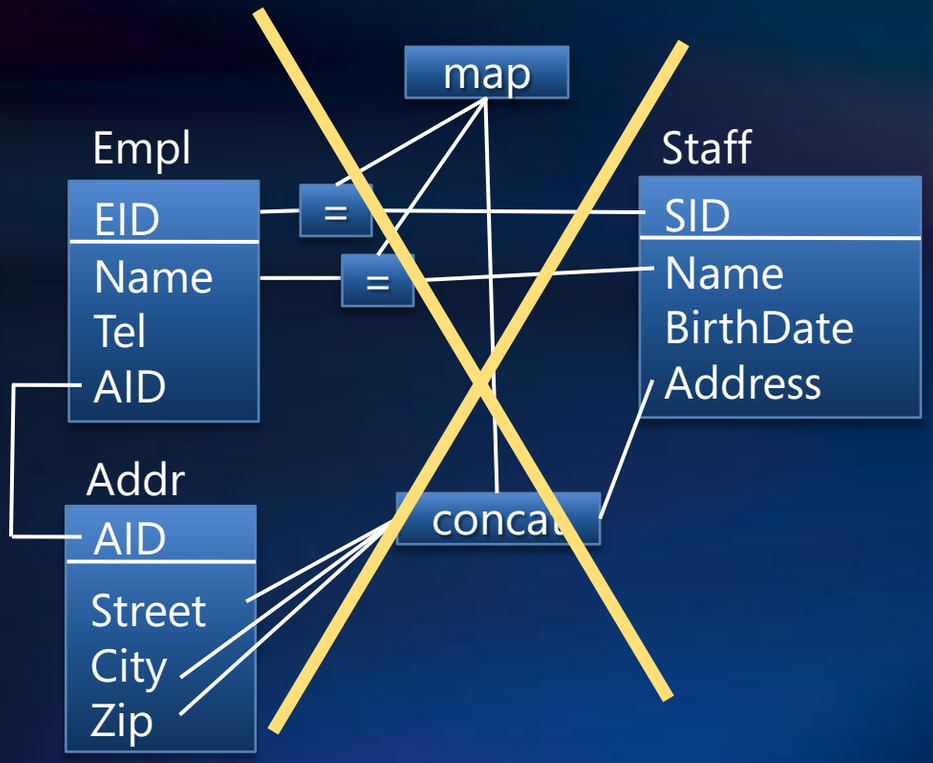
Example

[Bernstein. CIDR 03]

In 2000, mapping is a structure



Example
In 2007, just use the expression



$\pi_{EID}(Empl) = \pi_{SID}(Staff) \wedge$

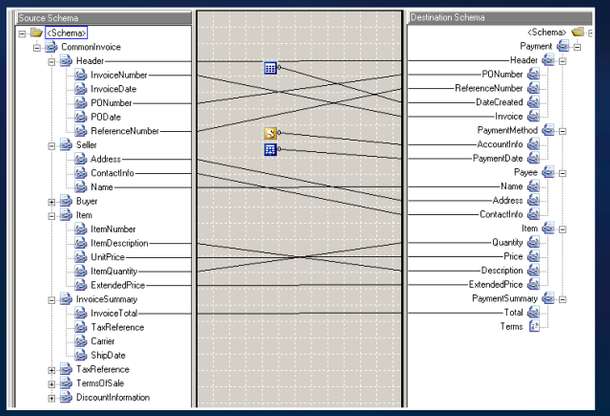
$\pi_{EID,Name}(Empl) = \pi_{SID,Name}(Staff) \wedge$

$\pi_{EID,AID,Concat(Street,City,Zip)}(Empl \bowtie Addr) = \pi_{SID,Address}(Staff)$

Mappings

[Casanova, Vidal. PODS 83] [Biskup, Convent. SIGMOD 86] [Catarci, Lenzerini. J. CoopIS 93] [Miller, Haas, Hernandez. VLDB 00]

- Element correspondences
  - First step in aligning schemas
  - For lineage & impact analysis
  - Weak or no formal semantics



- Mapping constraints relate instances of schemas
  - E.g., equality of relational expressions

```
SELECT Id, Name, Dept  =  SELECT Id, Name, Dept
FROM   Employee           FROM HR JOIN Empl ON Id
```

- Transformation is an executable mapping constraint
  - Constructs target instances from source instances
  - E.g., SQL query, XSLT, C# program

Mapping Expressiveness

- What we want: first-order logic with
  - negation
  - aggregation
  - set and bag semantics
  - regular expressions
  - nested collections and lists
  - rich type constructors (e.g., to construct XML fragments),
  - user-defined functions
  - deduplication and other heuristic functions
- What can we handle? … Much less.

Parallel Evolution

Clio Project

- IBM, Univ. of Toronto, U.C. Santa Cruz
- Miller, Haas, Hernandez, Fagin, Ho, Popa, Tan, …

Model Management

- Microsoft, Univ. of Washington, Univ. of Leipzig
- Bernstein, Halevy, Pottinger, Rahm, Madhavan, Melnik, …

Build a design tool for semantic mappings

Build model management operations with plug-in semantics

Study model management operations with semantics

What Has Changed?
Include the Runtime

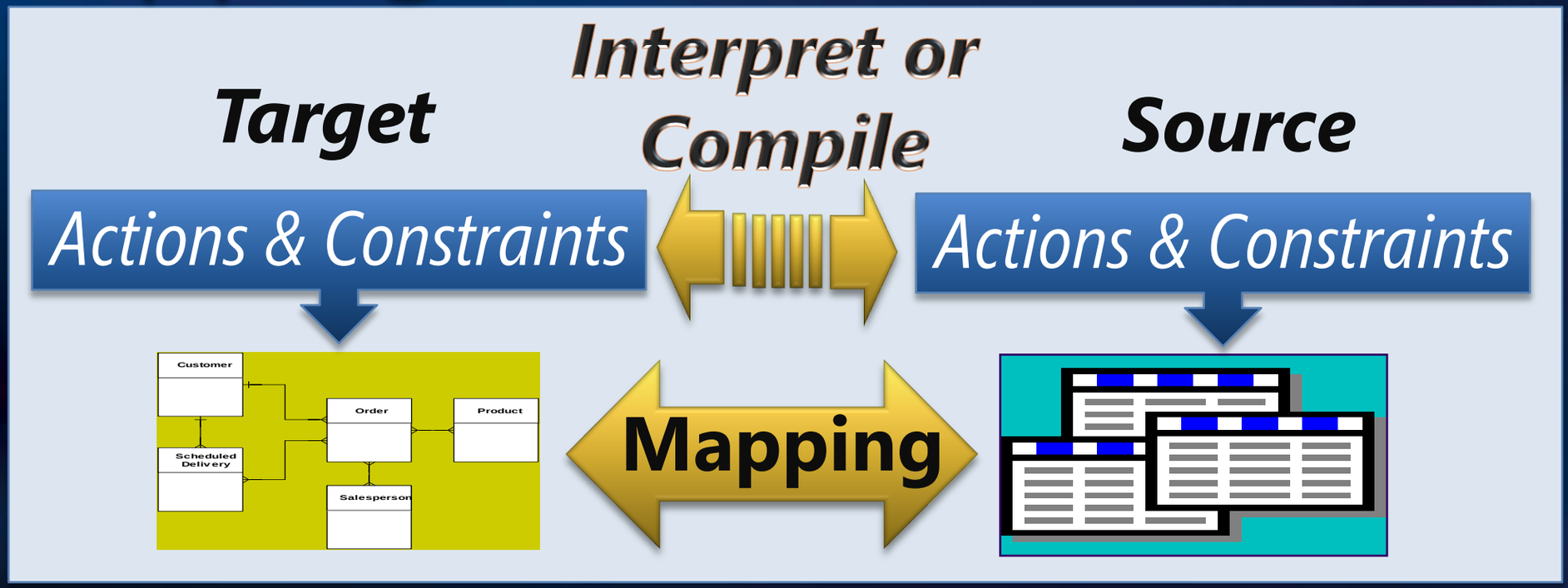
2000
Design-time

- Model management is independent of run-time
- No special run-time functionality

2007
Run-time

- Model management is tied to a run-time
- Run-time functions are sensitive to mapping expressiveness and model mgt capabilities

Mapping Runtime



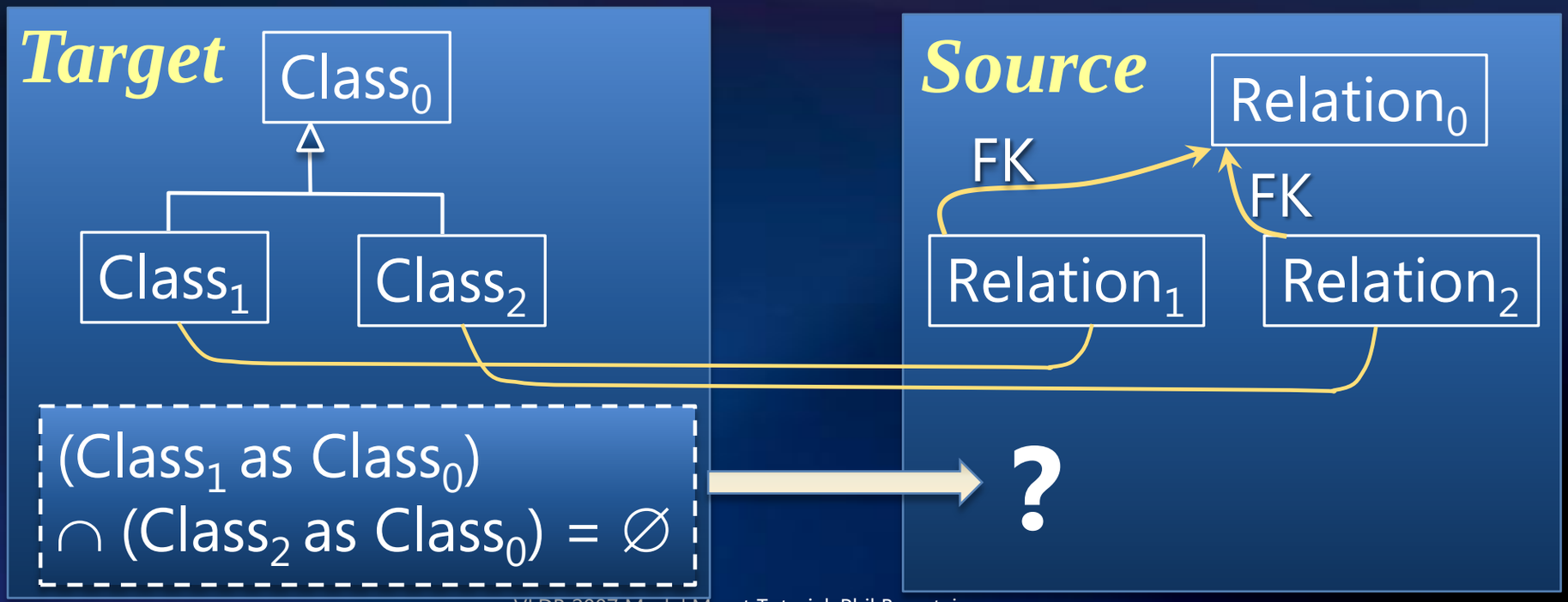
- Queries
- Updates
- Peer-to-peer
- Provenance
- Access Control

- Integrity constraints
- Synch logic
- Business logic
- Debugging

- Errors
- Indexing
- Notifications
- Batch loading
- Data exchange

Mapping Runtime Examples

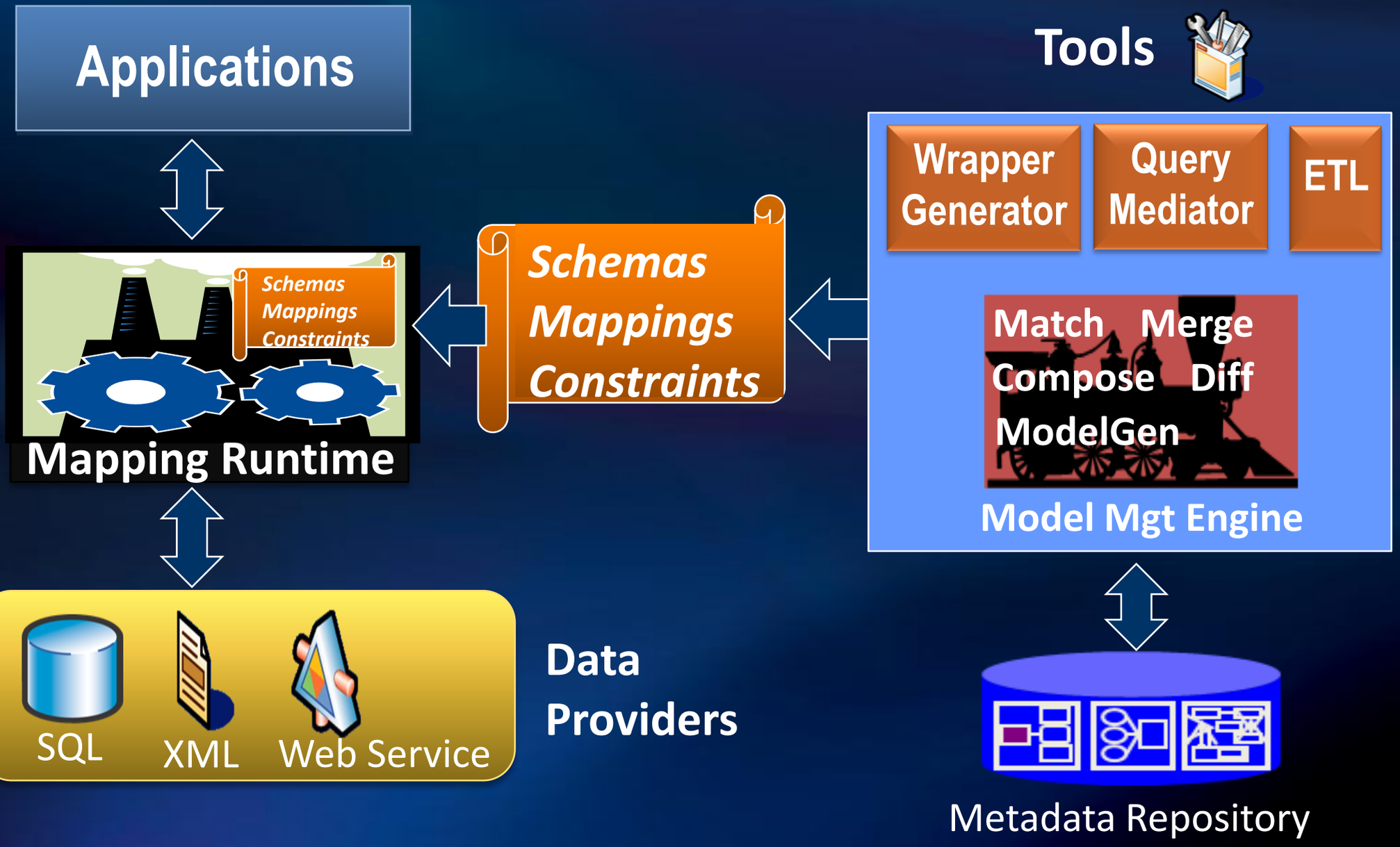
- Integrity constraints
  - Integrity constraints on target T are enforced by a combination of constraints enforced by the source and by the target runtime.
  - Feasibility - some constraints on T may not be expressible in source S.



Mapping Runtime Examples (2)

- Provenance [Cui, Widom, Weiner. TODS 00] [Baghwat, Chiticariu, Tan, Vijayvargia. VLDB J. 05] [Buneman, Chapman, Cheney. SIGMOD 06]
  - User moves data from source S to target T
  - Which source data contributed to a particular target data item?
- Errors
  - A data access via T is translated into an access on S that generates an error
  - The error needs to be passed back through the mapping in a form that is understandable in the context of T.

Model Management 2.0



Scenarios

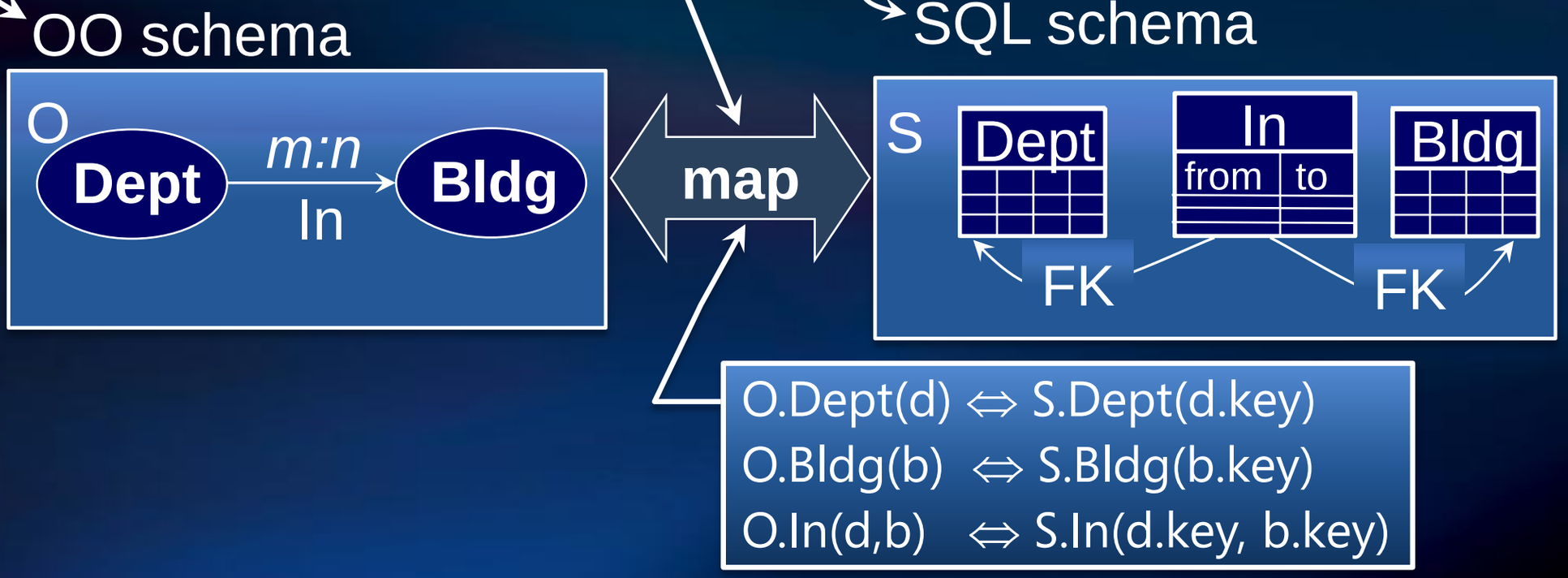
1. **Create mappings**
   - ModelGen
   - Match
   - ConstraintGen
   - TransGen
2. **Evolve mappings**
   - Compose
   - Diff
   - Merge
   - Inverse

ModelGen: Schema Translation

Input
- source model
- target metamodel

Output
- target model
- constraints

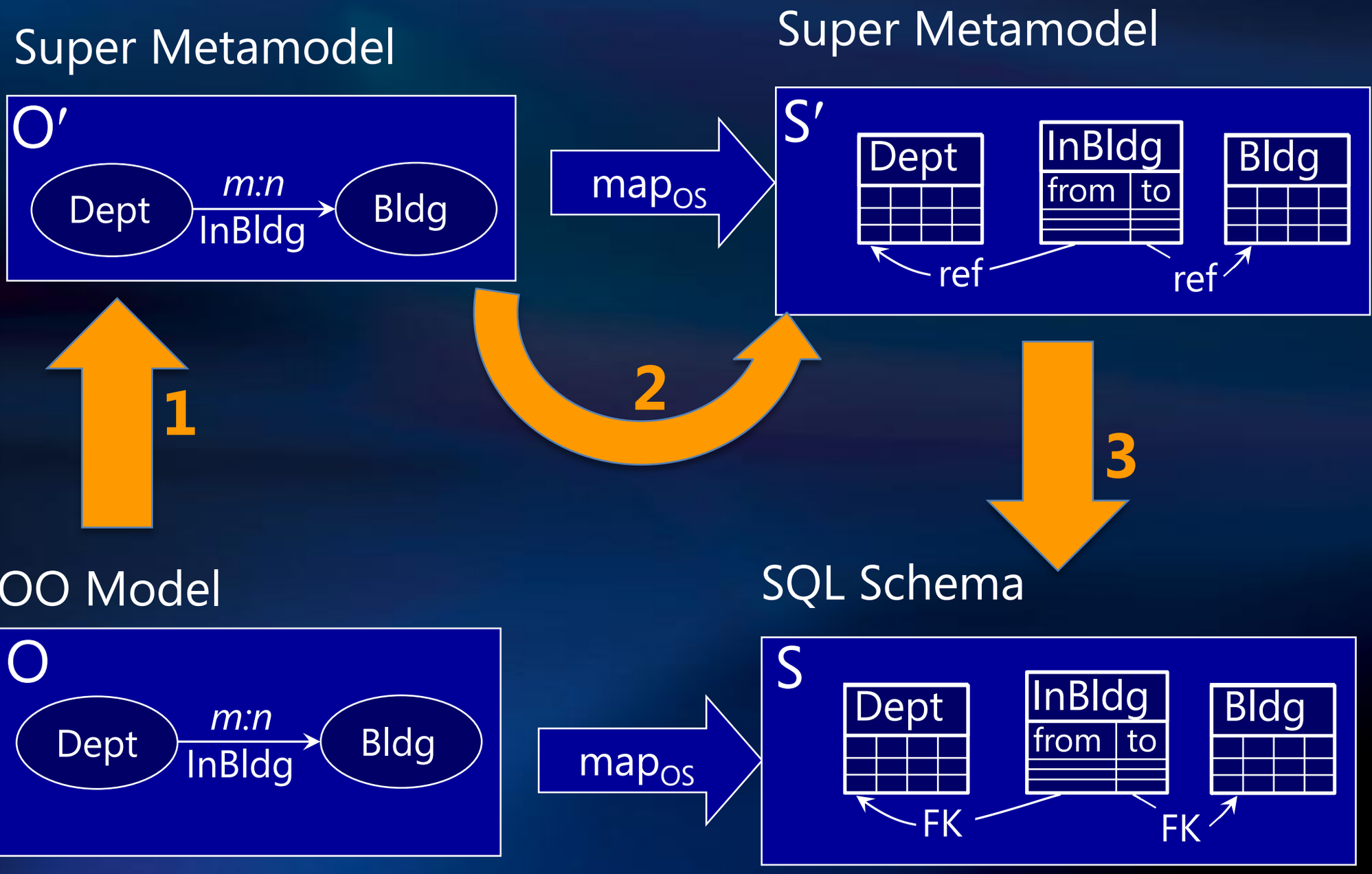
[Atzeni, Torlone. EDBT 96]
[Bernstein, Melnik, Mork. VLDB 05]
[Atzeni, Cappellari, Bernstein. EDBT 06]



- There are several credible prototypes
  - Don't know of products, yet

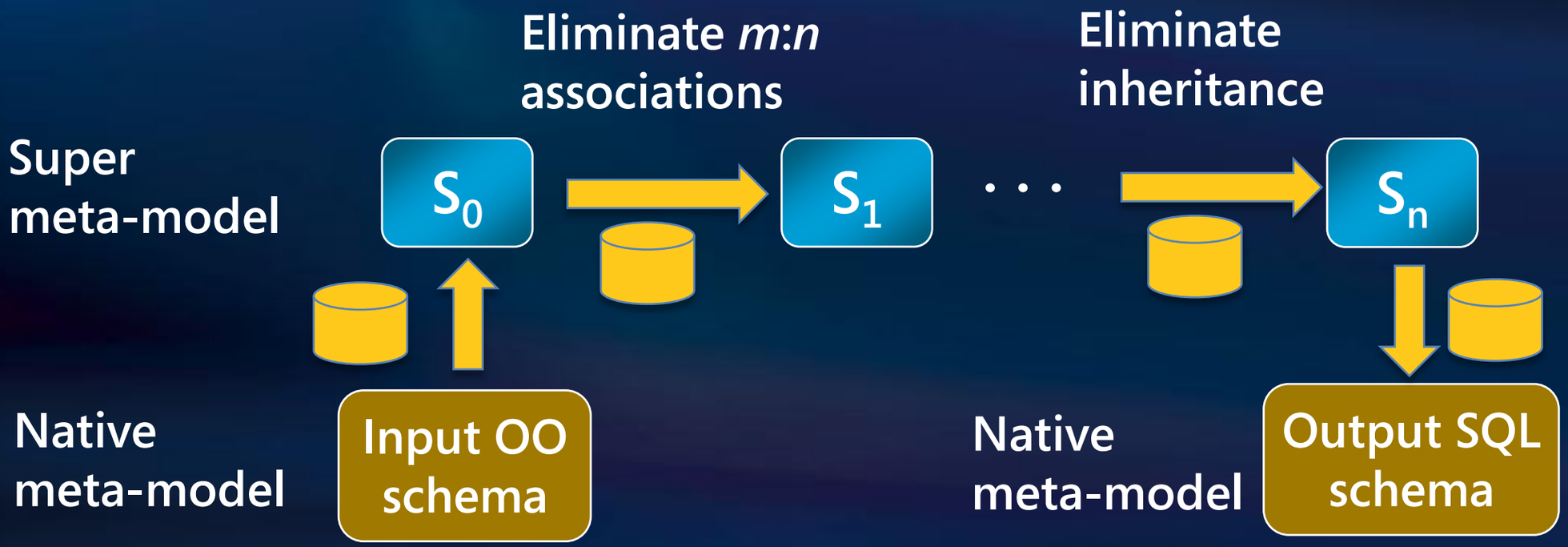
Implementation Strategy

[Atzeni & Torlone, EDBT '96]



Moving Data via ModelGen

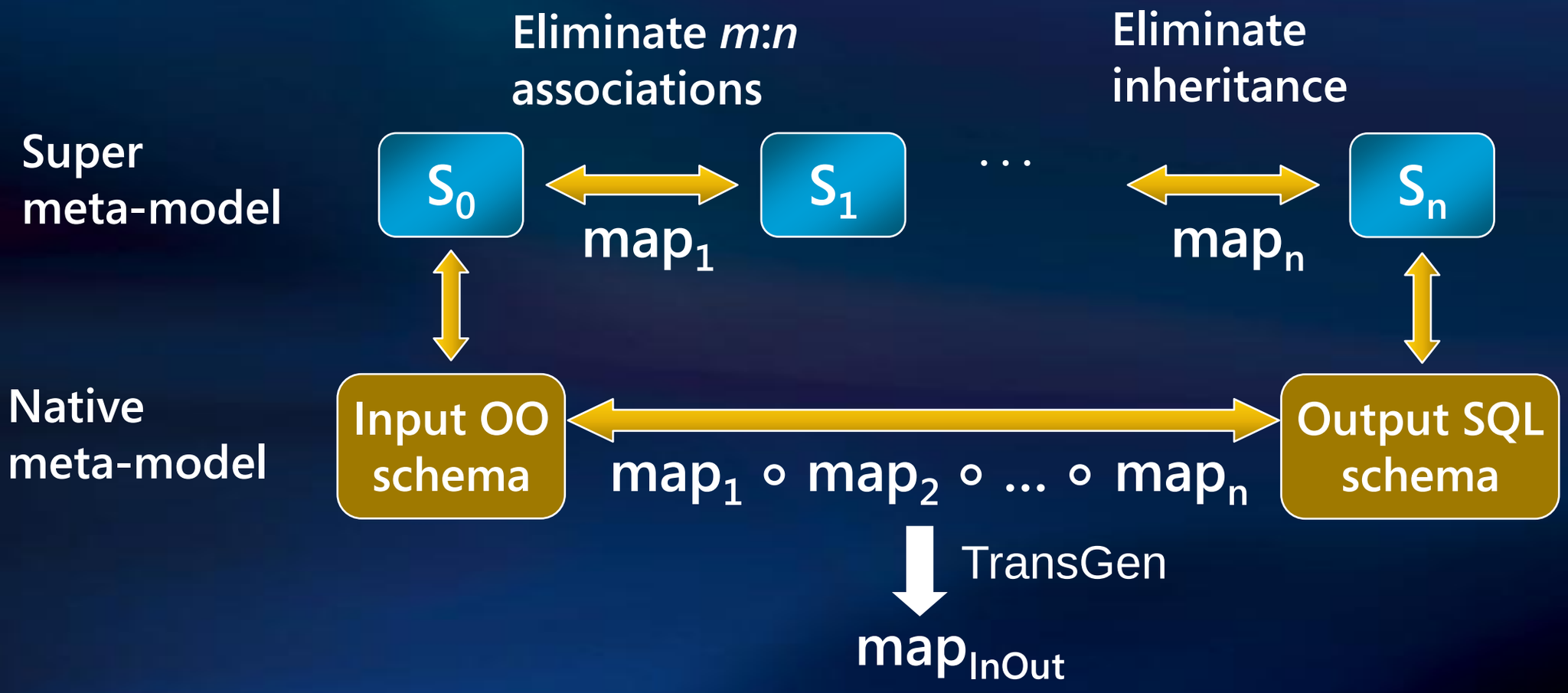
[Papotti, Torlone]
[Atzeni, Cappellari]



- Data is transferred to super-metamodel DB
- Data is transformed within super-metamodel DB
- Data is transferred to output schema's database

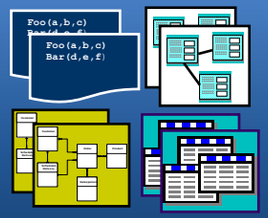
Obtaining Mappings From ModelGen

[Bernstein, Melnik, Mork VLDB'05, ER'07]



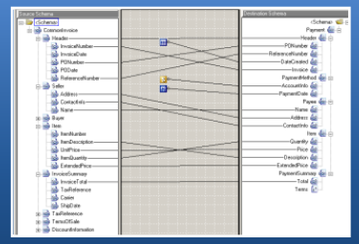
- Leverages Compose operator
- Each map_i roundtrips data

Code Generation Scenarios [Miller, Haas, & Hernandez, VLDB 00]



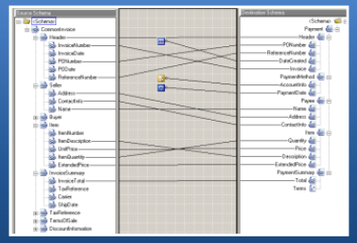
Models

Match



Correspondences

ModelGen



Correspondences

ConstraintGen

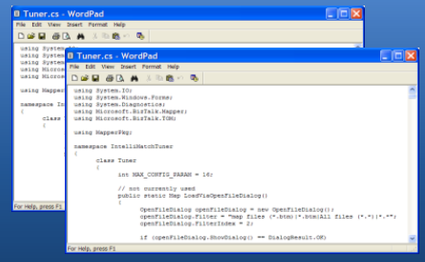
Select ord#, prod#, cust#
From Shipped
=
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

Select ord#, prod#, cust#
From Shipped
⊆
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

TransGen



Transformations

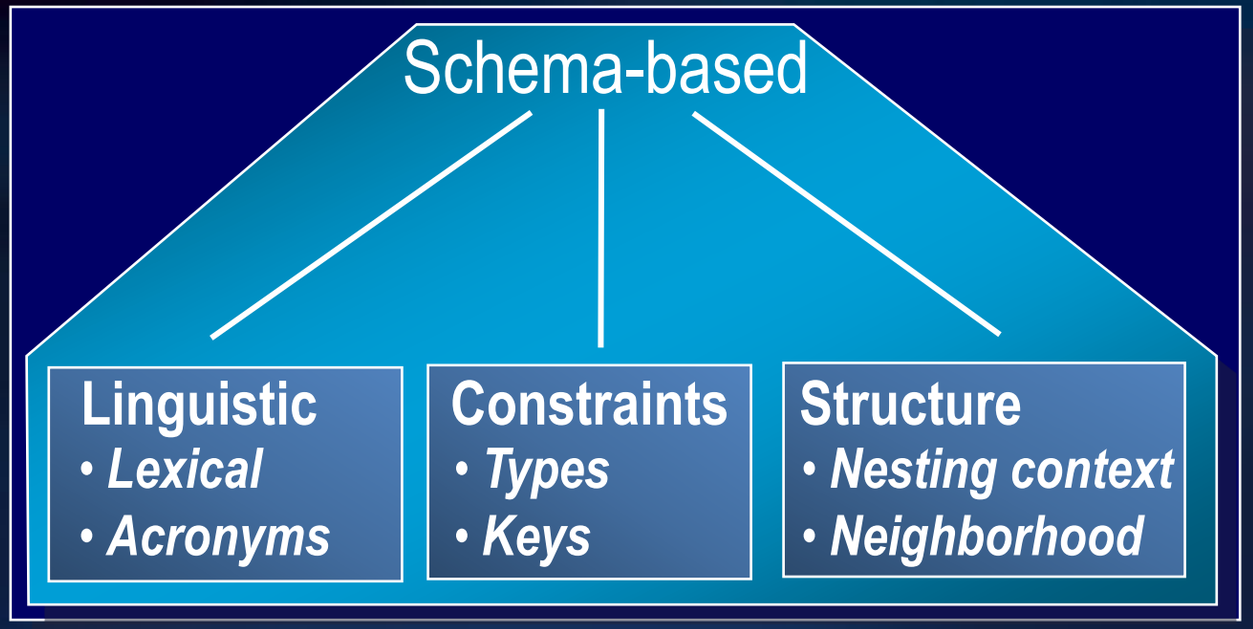
Schema Matching

- Exploit lexical analysis of element names, schema structure, data types, thesauri, value distributions, ontologies, instances, and previous matches
- Past Goal - improved precision & recall
  - Big productivity gains are unlikely
- Better goals
  - Return top-k, not best overall match
  - Avoid the tedium. Manage work.
  - HCI – handle large schemas.
  - User studies – what would improve productivity?

Schema Matching Algorithms

[Rahm & Bernstein, VLDB Journal 01]

- correspondences = **Match**($Schema_1$, $Schema_2$)
- Goal: Find good candidate matches
- Educated guesses, using multiple heuristics



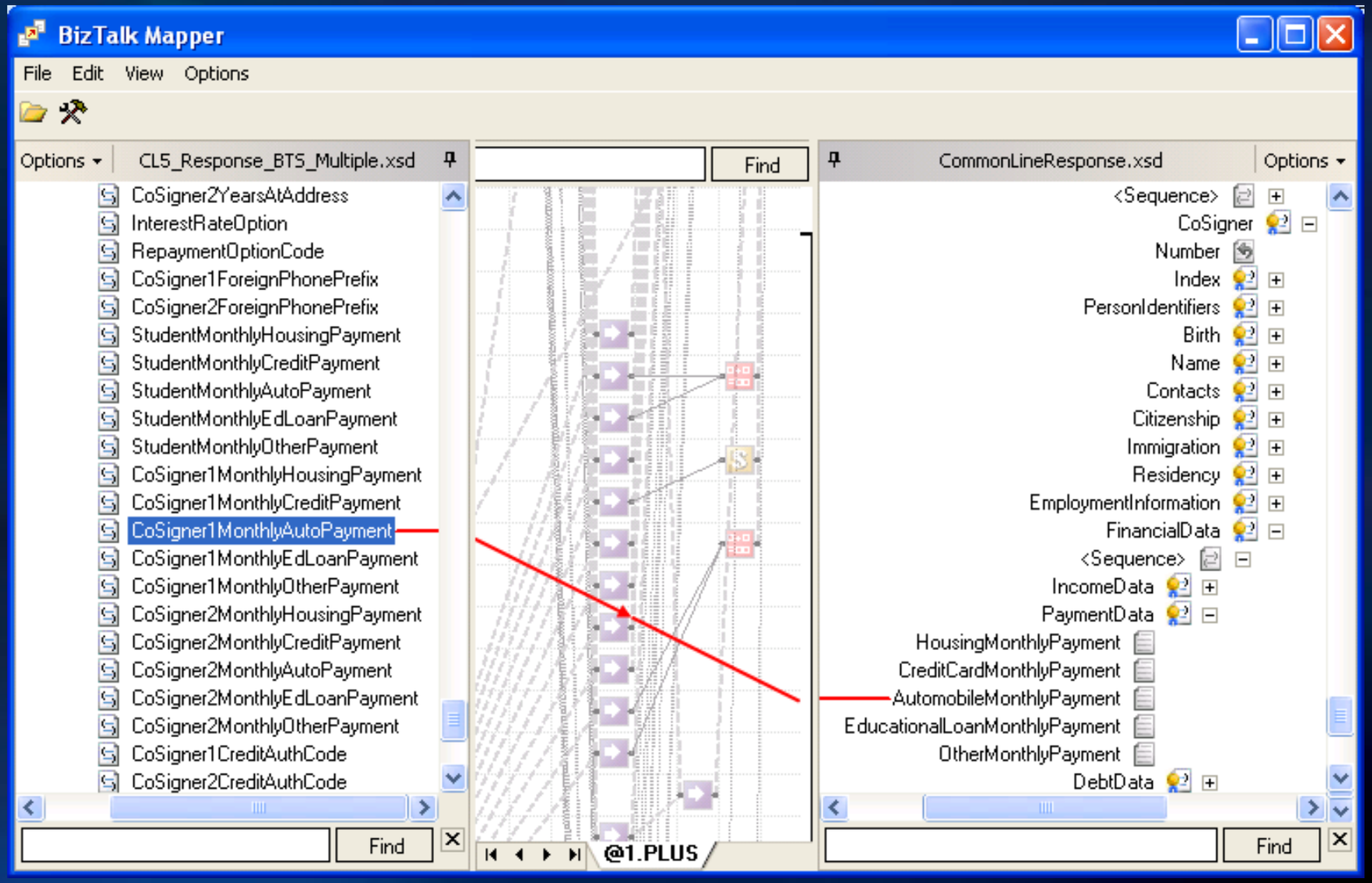
Reuse-based
- *Thesaurus*
- *Validated matches*

Content-based
- *Values*
- *Value Patterns*
- *Machine learning*

Incremental Matching for BizTalk Mapper

BizTalk Mapper: Demo [Bernstein, Melnik & Churchill, VLDB 06]

1. **Press SHIFT**
2. **Best candidate highlighted**
3. **Arrows navigate candidates**
4. **ENTER confirms**



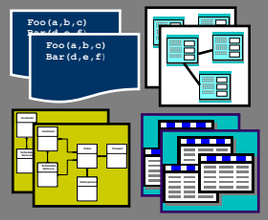
Screenshot: top match candidate (right) is element in context of CoSigner (as opposed to Borrower)

Cast of Thousands

- AnHai Doan
- Alon Halevy
- Pedro Domingos
- Phil Bernstein
- Erhard Rahm
- Sergey Melnik
- Jayant Madhavan
- Jeffrey Naughton
- Jaewoo Kang
- Tova Milo
- Pavel Shvaiko
- Fausto Giunchiglia
- Sonia Bergamaschi
- Silvana Castano
- Bin He
- Kevin Chang
- Namyoun Choi
- Il-Yeol Song
- Hyoil Han
- Domenico Ursino
- Luigi Palopoli
- Dominico Sacca
- Georgio Terracina
- David Embley
- David Jackman
- Li Xu
- Yihong Ding
- Jacob Berlin
- Amihai Motro
- Hong Hai Do
- Fabien Duchateau
- Zohra Bellahsene
- Ela Hunt
- Toralf Kirsten
- Andreas Thor
- Alexander Bilke
- Avigdor Gal
- Michalis Petropoulos
- Christoph Quix
- Chris Clifton
- Arnie Rosenthal
- Wen-Syan Li
- Hector Garcia-Molina
- Sagit Zohar
- Gio Wiederhold
- Anna Zhdanova
- Jerome Euzenat
- Prasenjit Mitra
- Natasha Noy
- Anuj Jaiswal
- Mikalai Yatskevich
- Nuno Silva
- Joao Rocha
- David Aumueller
- Sabine Massmann
- Felix Naumann

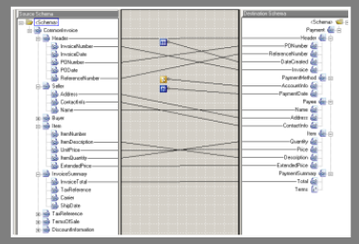
- Need better UI, workflow support, not (just) better precision & recall

Code Generation Scenarios [Miller, Haas, & Hernandez, VLDB 00]

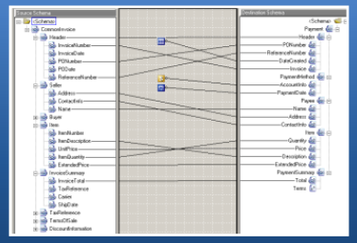


Models

Match



Correspondences



Correspondences

ConstraintGen

Select ord#, prod#, cust#
From Shipped
=
Select ord#, prod#, cust#
From Order Join Item
on ord#

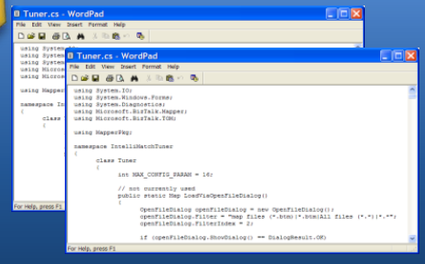
Constraints

?

Select ord#, prod#, cust#
From Shipped
⊆
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

TransGen



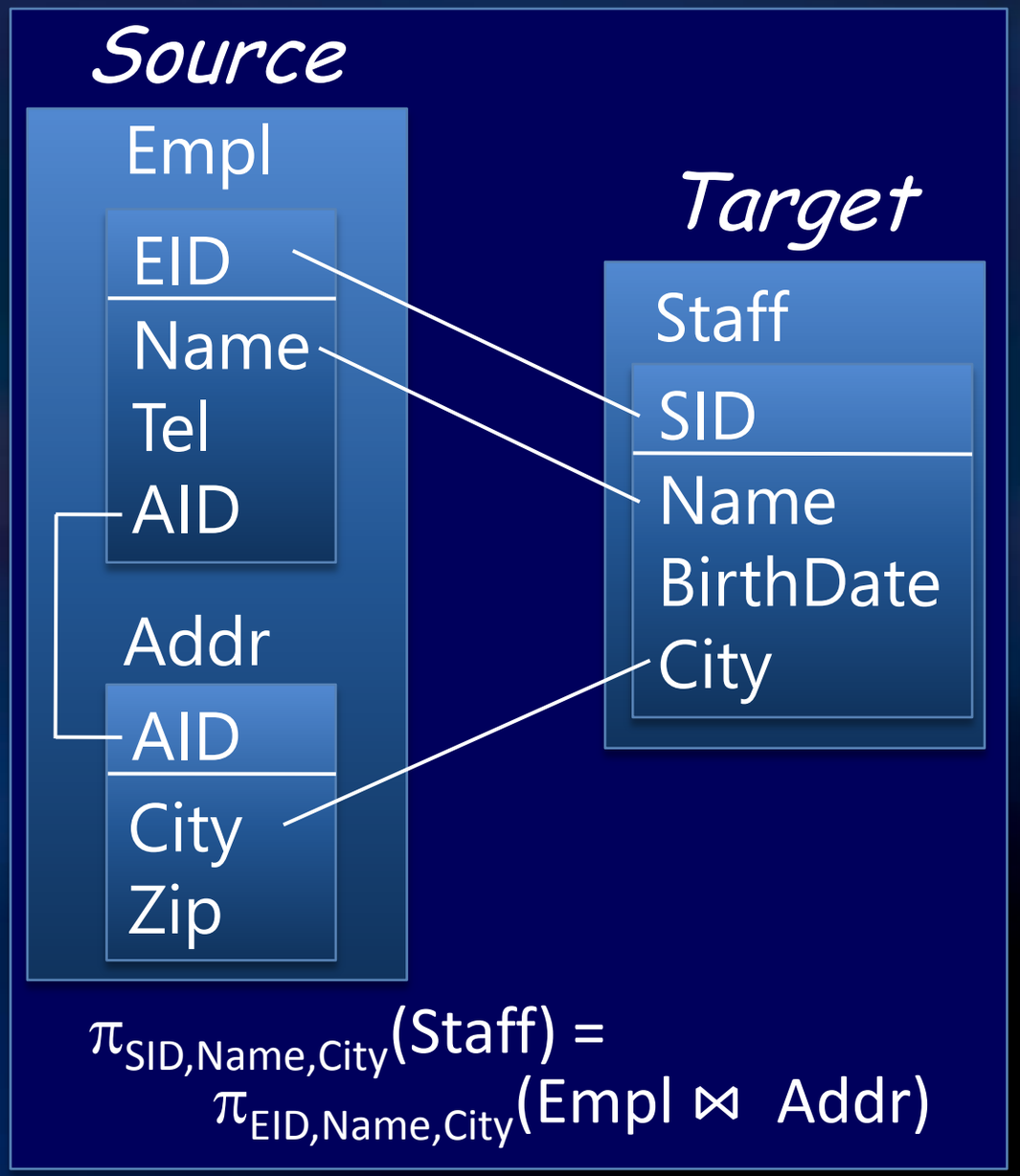
Transformations

Correspondences → Transformations

[Popa, Velegrakis, Miller, Hernandez, Fagin. VLDB 02]
[Velegrakis. PhD thesis 2005]

For a given target element

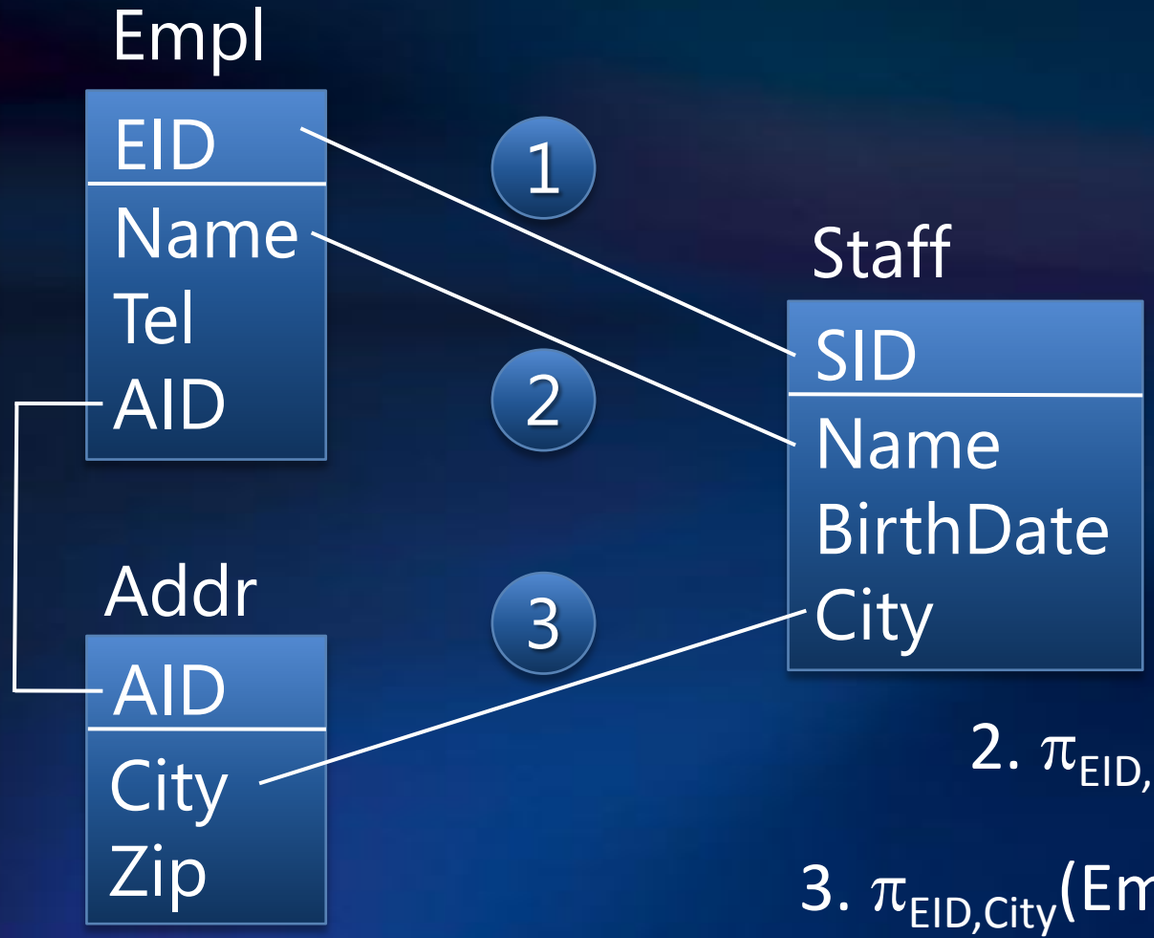
- Find all source elements linked by correspondences
- Find all ways that source elements are related
- Choose one of them and generate the transformation



Correspondences → Constraints

[Melnik, Bernstein, Halevy, & Rahm, SIGMOD 05]

- Directly interpret correspondences as mapping constraints
- If it's a tree schema and keys correspond



1. $\pi_{EID}(Empl) = \pi_{SID}(Staff)$

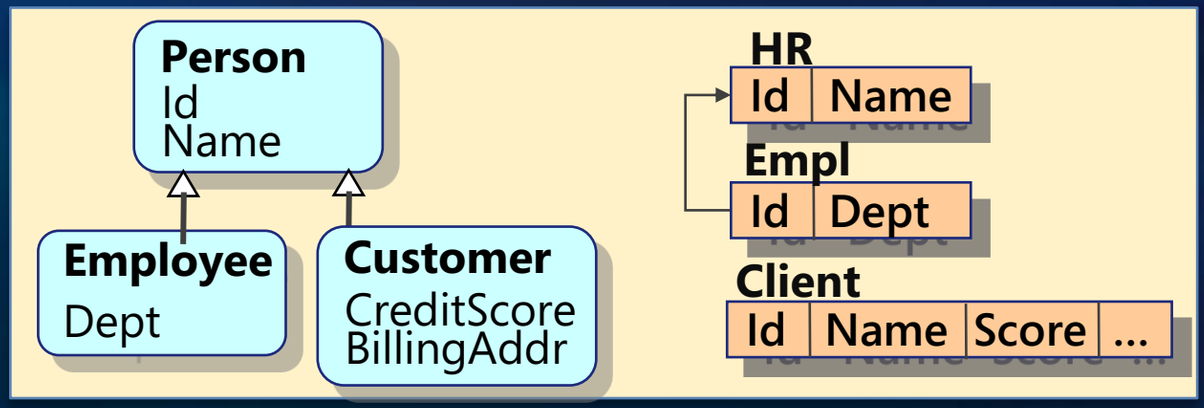
2. $\pi_{EID,Name}(Empl) = \pi_{SID,Name}(Staff)$

3. $\pi_{EID,City}(Empl \bowtie Addr) = \pi_{SID,City}(Staff)$

Constraints in ADO.NET

[Melnik, Adya, Bernstein, SIGMOD 07]

Source: EER
Target: SQL



Mapping Constraints

```
SELECT p.Id, p.Name
FROM Persons AS p
WHERE p IS OF (ONLY Person)
  OR p IS OF (ONLY Employee)
```

=

```
SELECT Id, Name
FROM dbo.HR
```

```
SELECT e.Id, e.Dept
FROM Persons AS e
WHERE e IS OF Employee
```

=

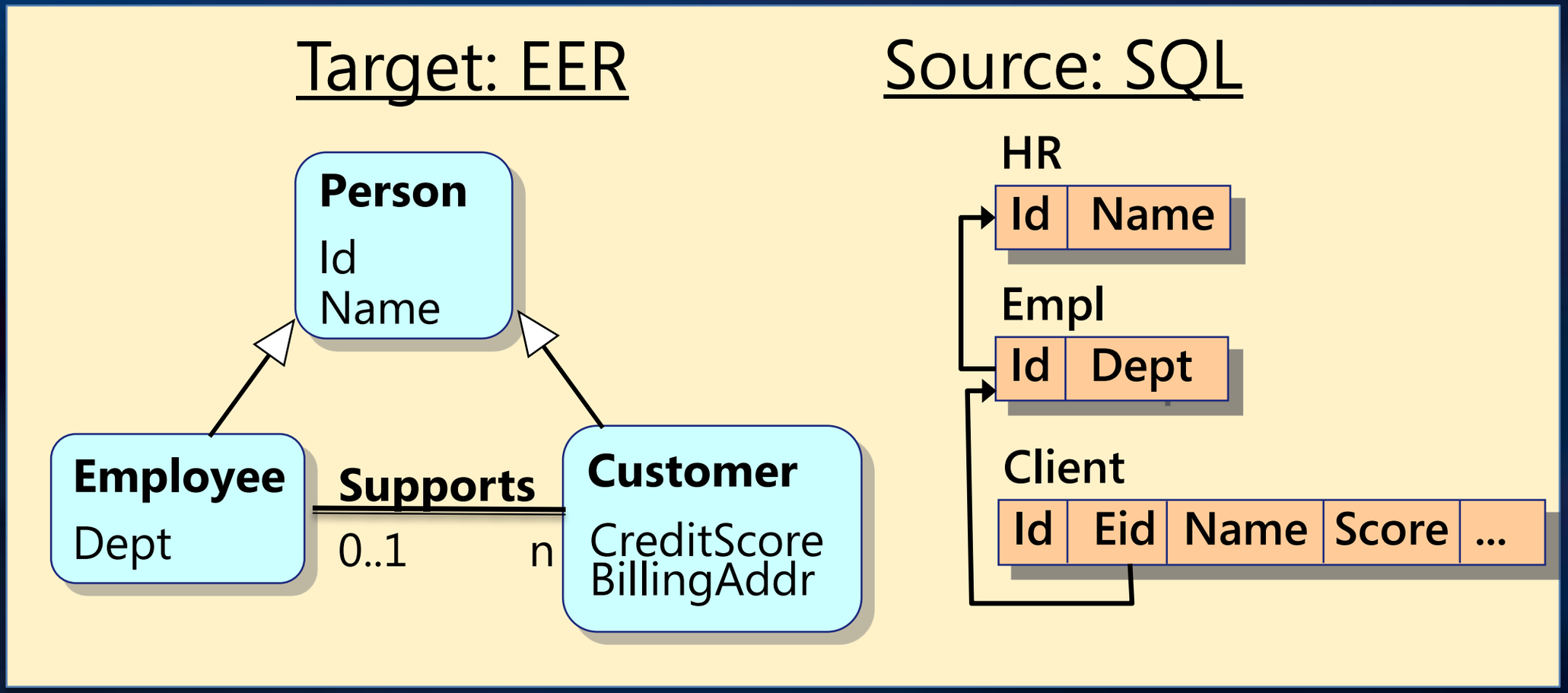
```
SELECT Id, Dept
FROM dbo.Empl
```

```
SELECT c.Id, c.Name,
   c.CreditScore, c.BillingAddr
FROM Persons AS c
WHERE c IS OF Customer
```

=

```
SELECT Id, Name,
          Score, Addr
FROM dbo.Client
```

A Relationship Constraint

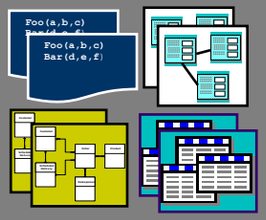



```
SELECT Key(s.Customer).Id,
       Key(s.Employee).Id
FROM Supports s
```

=

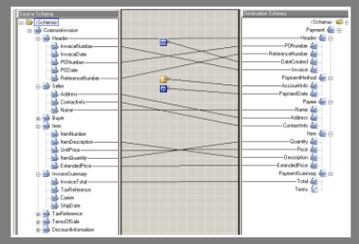
```
SELECT Cid, Eid
FROM Client
WHERE Eid IS NOT NULL
```

Code Generation Scenarios

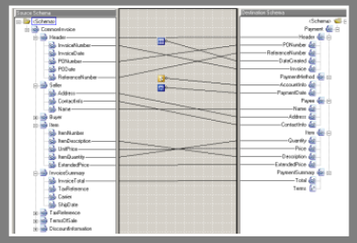


Models

Match



Correspondences



Correspondences

ConstraintGen

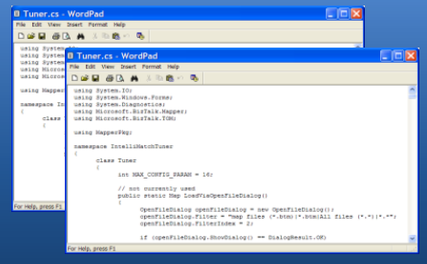
Select ord#, prod#, cust#
From Shipped
=
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

Select ord#, prod#, cust#
From Shipped
⊆
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

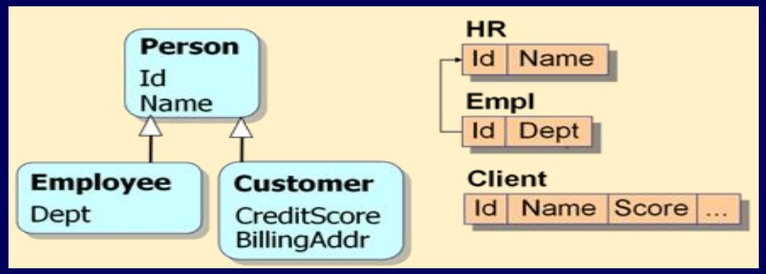
TransGen



Transformations

Constraints → Transformations

[Melnik, Adya, Bernstein, SIGMOD 07]




```
SELECT p.Id, p.Name
FROM Persons AS p
WHERE p IS OF (ONLY Person)
  OR p IS OF (ONLY Employee)
```

=

```
SELECT Id, Name
FROM dbo.HR
```

```
SELECT e.Id, e.Dept
FROM Persons AS e
WHERE e IS OF Employee
```

=

```
SELECT Id, Dept
FROM dbo.Empl
```

```
SELECT c.Id, c.Name,
   c.CreditScore, c.BillingAddr
FROM Persons AS c
WHERE c IS OF Customer
```

=

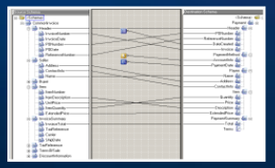
```
SELECT Id, Name,
       Score, Addr
FROM dbo.Client
```

```
SELECT VALUE
    CASE
        WHEN (T5._from2 AND NOT(T5._from1))  THEN Person(T5.Person_Id,
T5.Person_Name)
        WHEN (T5._from1 AND T5._from2)
            THEN Employee(T5.Person_Id, T5.Person_Name,  T5.Employee_Dept)
            ELSE Customer(T5.Person_Id, T5.Person_Name,  T5.Customer_CreditScore,
                T5.Customer_BillingAddr)
    END
  FROM (   (SELECT T1.Person_Id, T1.Person_Name,  T2.Employee_Dept,
                CAST(NULL AS SqlServer.int) AS Customer_CreditScore,
                CAST(NULL AS SqlServer.nvarchar) AS Customer_BillingAddr, False AS
                  _from0,
                (T2._from1 AND T2._from1 IS NOT NULL) AS _from1,  T1._from2
            FROM  ( SELECT  T.Id AS Person_Id, T.Name AS Person_Name, True AS
                  _from2
                FROM HR AS T) AS T1
                LEFT OUTER JOIN (
                  SELECT  T.Id AS Person_Id,  T.Dept AS Employee_Dept, True AS
                  _from1
                  FROM dbo.Empl AS T) AS T2
                ON T1.Person_Id = T2.Person_Id )
            UNION ALL (
                SELECT  T.Id AS Person_Id,  T.Name AS Person_Name,
                    CAST(NULL AS SqlServer.nvarchar) AS Employee_Dept,
                    T.Score AS Customer_CreditScore,  T.Addr AS
Customer_BillingAddr,
                    True AS _from0,  False AS _from1,  False AS _from2
                FROM Client AS T)
        ) AS T5
```

Constraints → Transformations (2)

- Difficulty depends on
  - Whether the constraints are functions
  - The transformation language (e.g., SQL, XSLT)
  - Expressiveness of constraints
  - Optimization required
- IBM's Clio translates constraints into data exchange programs in SQL or XSLT

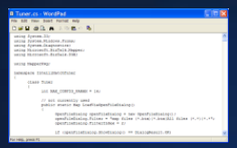
ADO.NET O/R Mapping [Melnik, Adya, Bernstein SIGMOD 07]



Correspondences

Select ord#, prod#, cust#
From Shipped
⊆
Select ord#, prod#, cust#
From Order Join Item
on ord#

Constraints

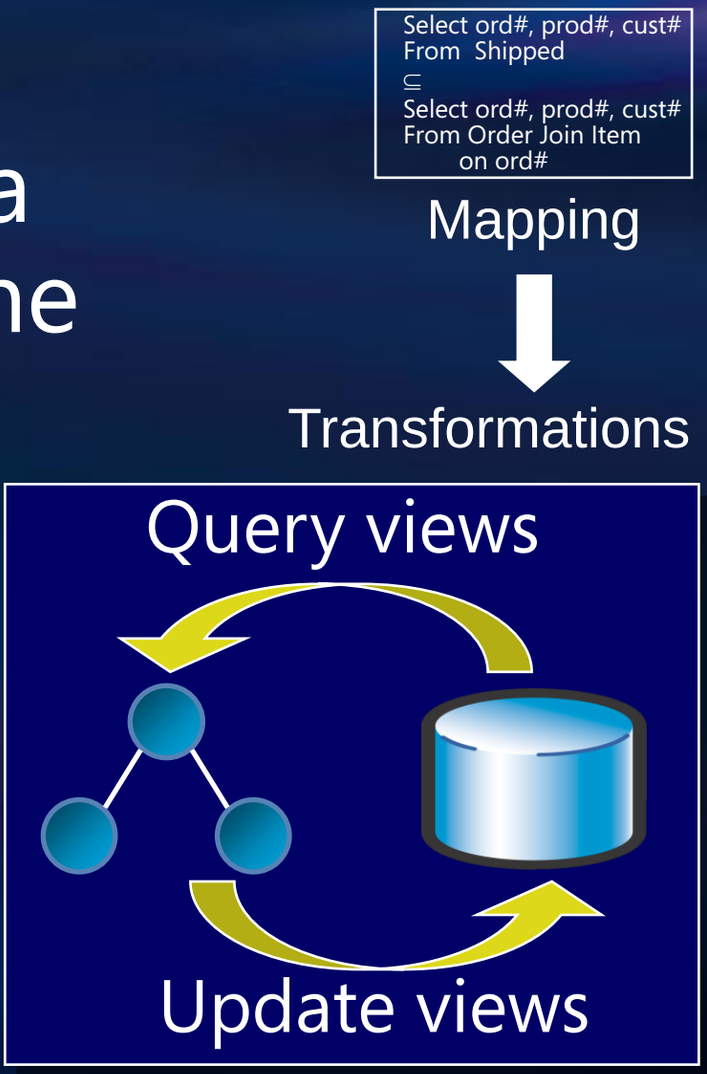


Transformations

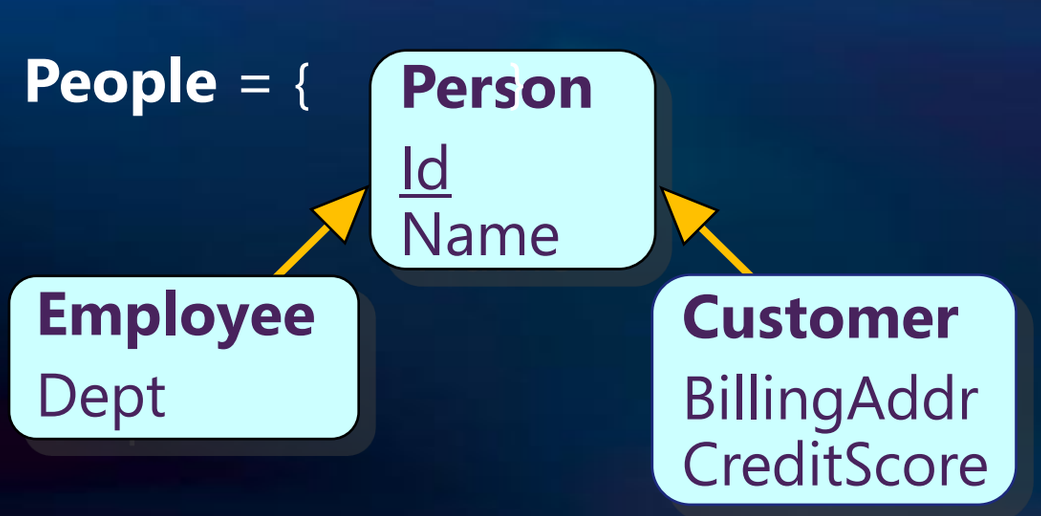
- Declarative mapping language
  - Allows non-expert users to specify complex mappings
  - Formal semantics
- Bidirectional views
  - Uniform, efficient runtime
  - Simplifies dev & test
- Updates via view maintenance
  - Arbitrary updates
  - Uses view maintenance technology

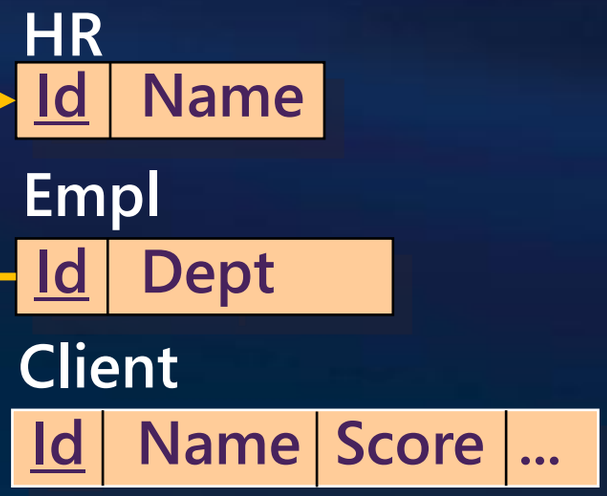
Bidirectional Views

- Mapping is compiled into data transformations driving runtime
- Views for querying
  - Objects = QueryViews(Tables)
  - Queries by view unfolding
  - "View update" problem
- New: update views
  - Tables = UpdateViews(Objects)
- Correctness: roundtripping
  - Objects = QueryViews(UpdateViews(Objects))
  - Enforced by view generation algorithm



Example – Update Views





HR = SELECT p.Id, p.Name FROM **People** p
WHERE p IS OF (ONLY Person) OR
p IS OF (ONLY Employee)

Empl = SELECT e.Id, e.Dept FROM **People** e
WHERE e IS OF Employee

Client = SELECT c.Id, c.Name, ... FROM **People** c
WHERE c IS OF Customer

U

Compiling Constraints (1)

- A mapping is a set of constraints:

  $\{ Q_{C1} = Q_{S1}, \quad \ldots, \quad Q_{Cn} = Q_{Sn} \}$

- E.g.,

```
SELECT e.Id, e.Dept
FROM Persons AS e
WHERE e IS OF Employee
```

=

```
SELECT Id, Dept
FROM dbo.Empl
```

- Goal: generate query and update views

Compiling Constraints (2)

- Mapping: $\{Q_{C1}=Q_{S1}, \ldots, Q_{Cn}=Q_{Sn}\}$
- E.g., f: SELECT p.Id, p.Name FROM **Persons** p = g: SELECT Id, Name FROM ClientInfo
- f: $V_1=Q_{C1} \cup$ $V_2=Q_{C2} \cup$ ... $V_n=Q_{Cn}$
- g: $V_1=Q_{S1} \cup$ $V_2=Q_{S2} \cup$ ... $V_n=Q_{Sn}$

Composition (1)

$$S_1 \xrightarrow{map_{12}} S_2 \xrightarrow{map_{23}} S_3$$

$$map_{12} \bullet map_{23}$$

$\mathbb{I}(S_1)$ are the instances of schema S_1

$map_{12} \subseteq \mathbb{I}(S_1) \times \mathbb{I}(S_2)$ $\quad$ $map_{13} \subseteq \mathbb{I}(S_2) \times \mathbb{I}(S_3)$

$map_{13} = \{ <s_1 \in S_1, s_3 \in S_3> \mid$

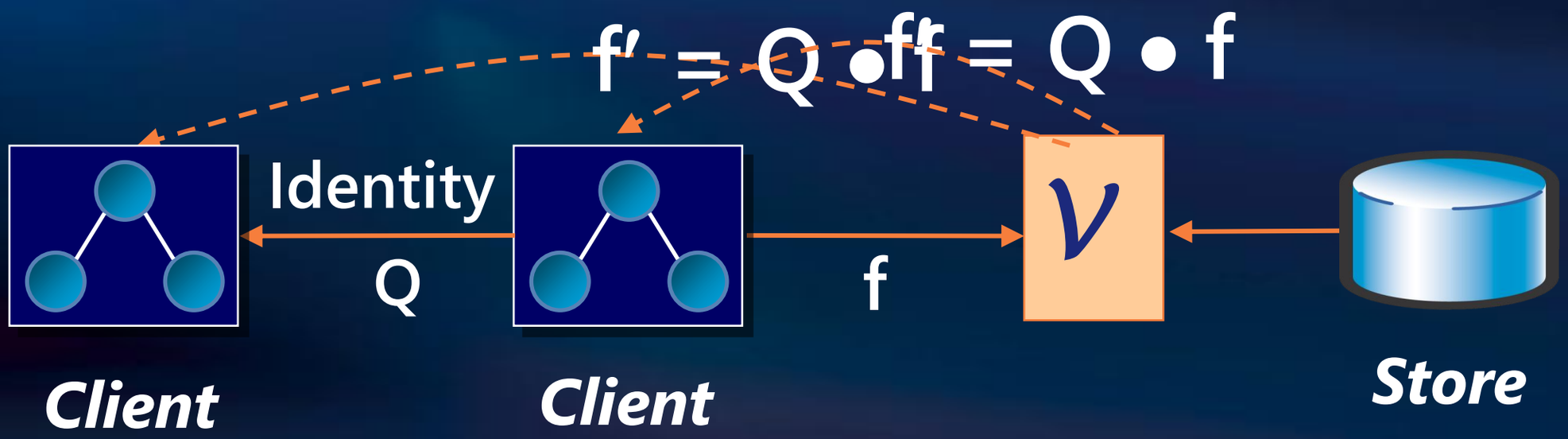
$(\exists\, s_2 \in S_2)\ (<s_1, s_2> \in map_{12})$

$\wedge\ (<s_2, s_3> \in map_{23})\ \}$

Well known examples

- View unfolding $S_1 \xrightarrow{v} S_2 \xrightarrow{q} S_3$
- Answering queries using views $S_1 \xleftarrow{v} S_2 \xrightarrow{q} S_3$

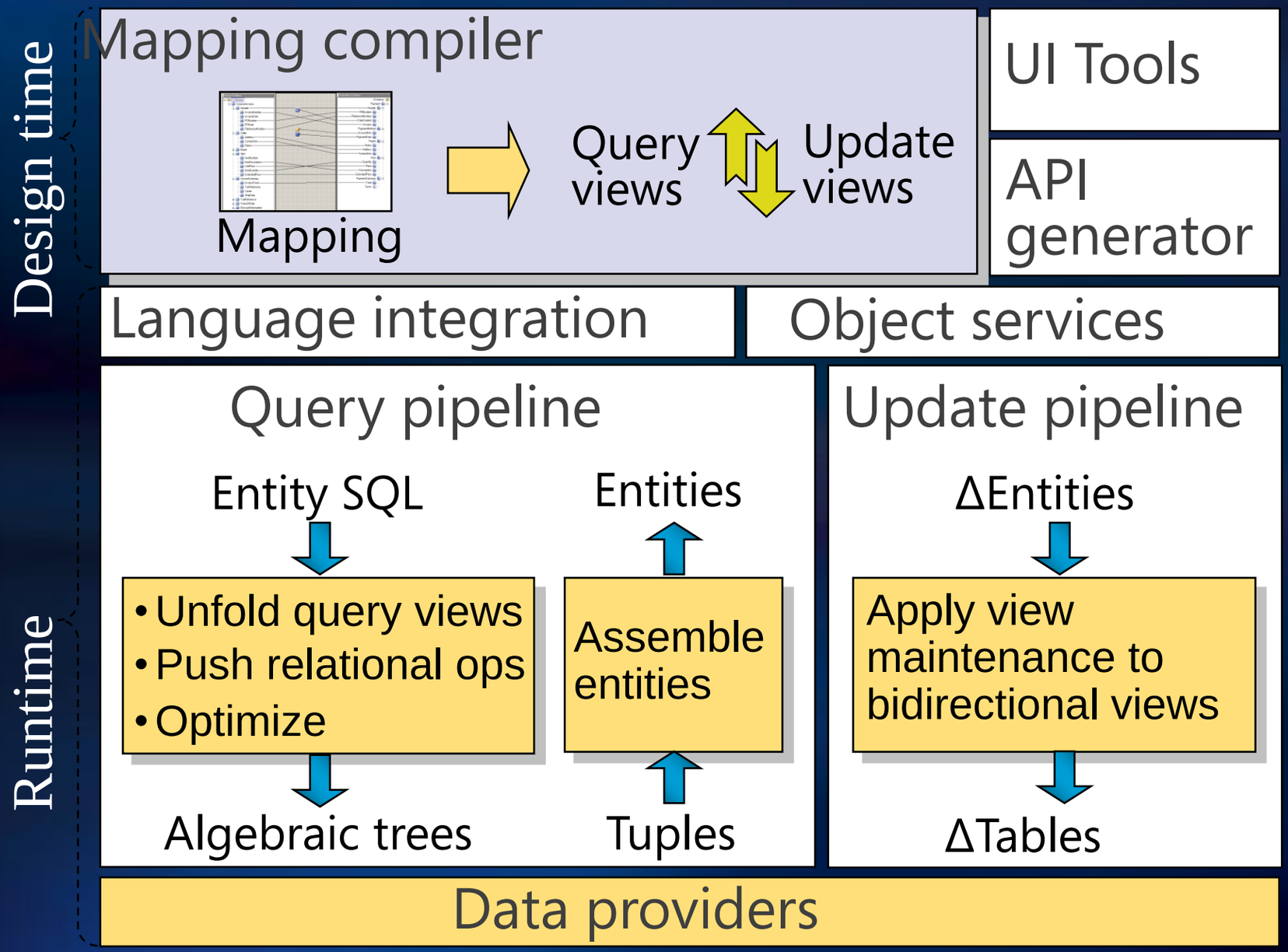
Computing f′ from f



- Use query rewriting to compute f′ from f
- This is mapping composition.

Architecture

Adya, Blakeley, Melnik, Muralidhar & ADO.NET Team (SIGMOD'07)

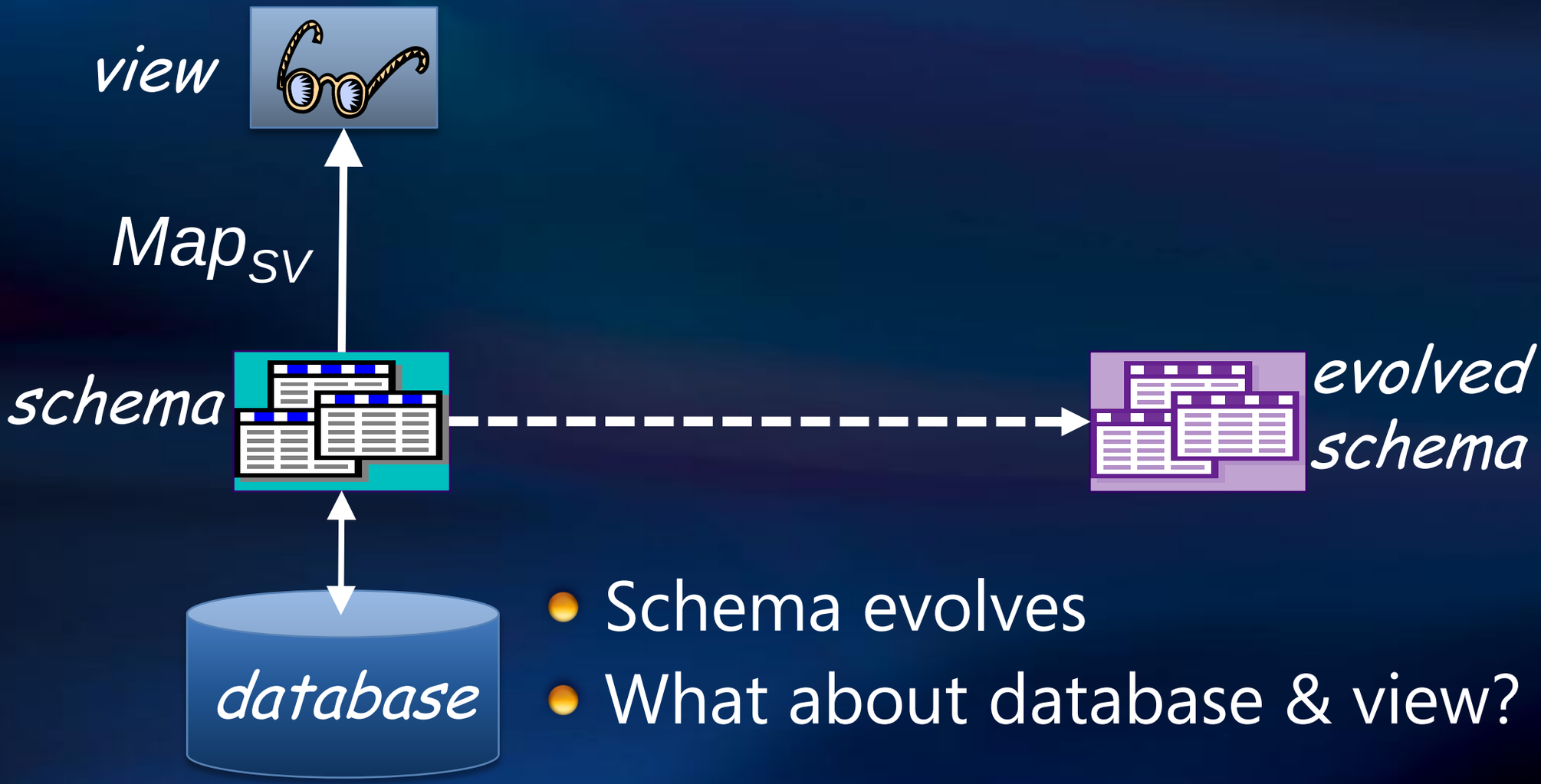


Scenarios

1. Create mappings
   - ✓ Match
   - ✓ ConstraintGen
   - ✓ TransGen
   - ✓ ModelGen

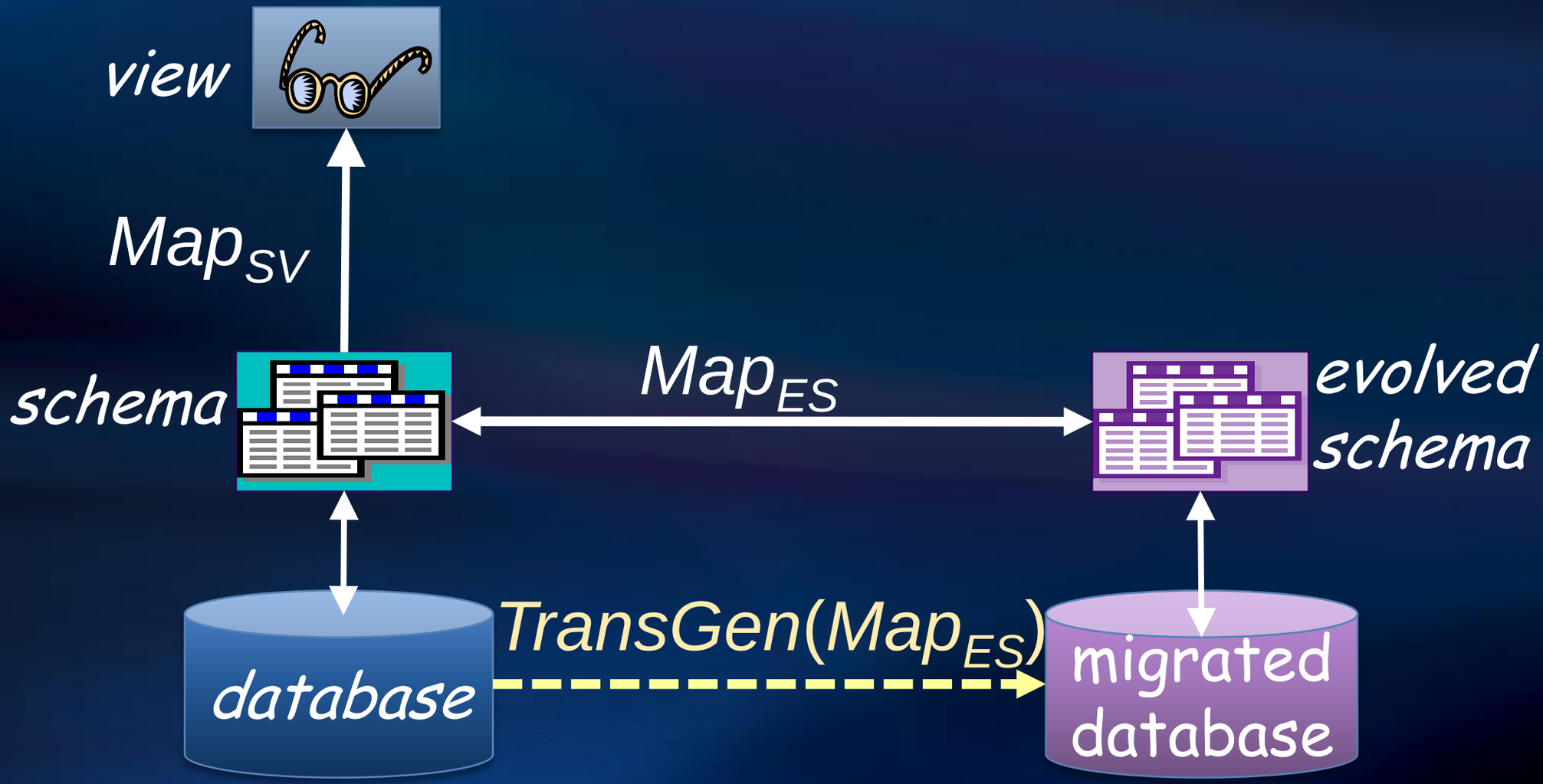
2. **Evolve mappings**
   - Compose
   - Diff
   - Merge
   - Inverse

Schema Evolution [Rahm, Bernstein. SIGMOD Rec. Dec 06]



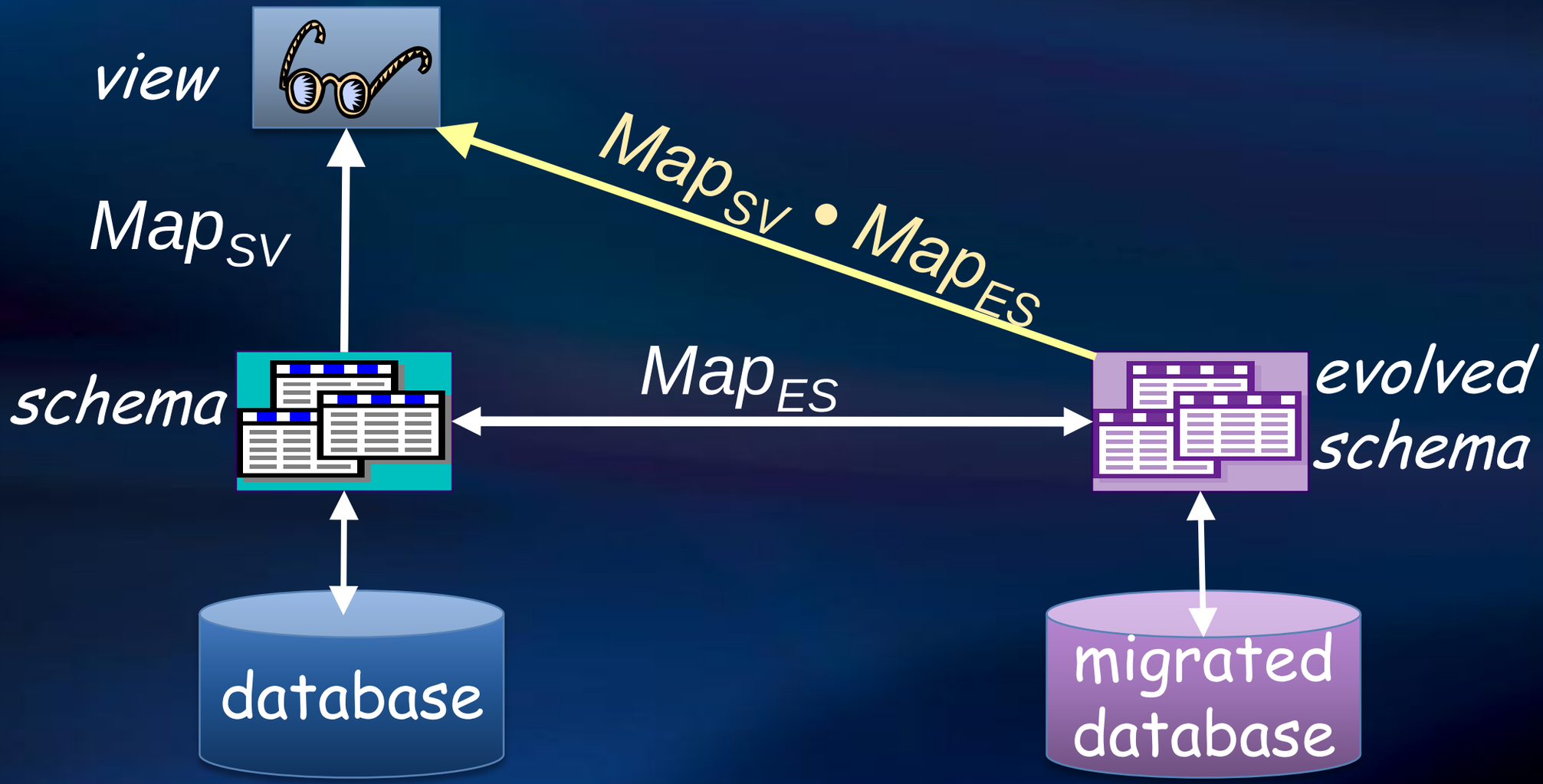
- Schema evolves
- What about database & view?

Data Migration



1. Create mapping: *schema* ⇔ *evolved schema*
2. Generate a transformation

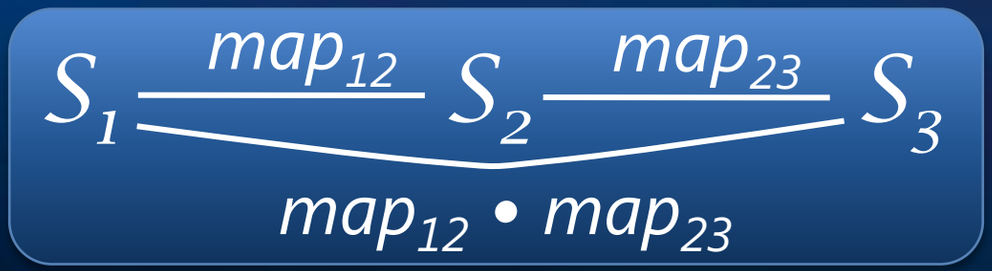
View Migration



- Compose Map_{SV} and Map_{ES} to connect *view* to *evolved schema*

Composition (2)

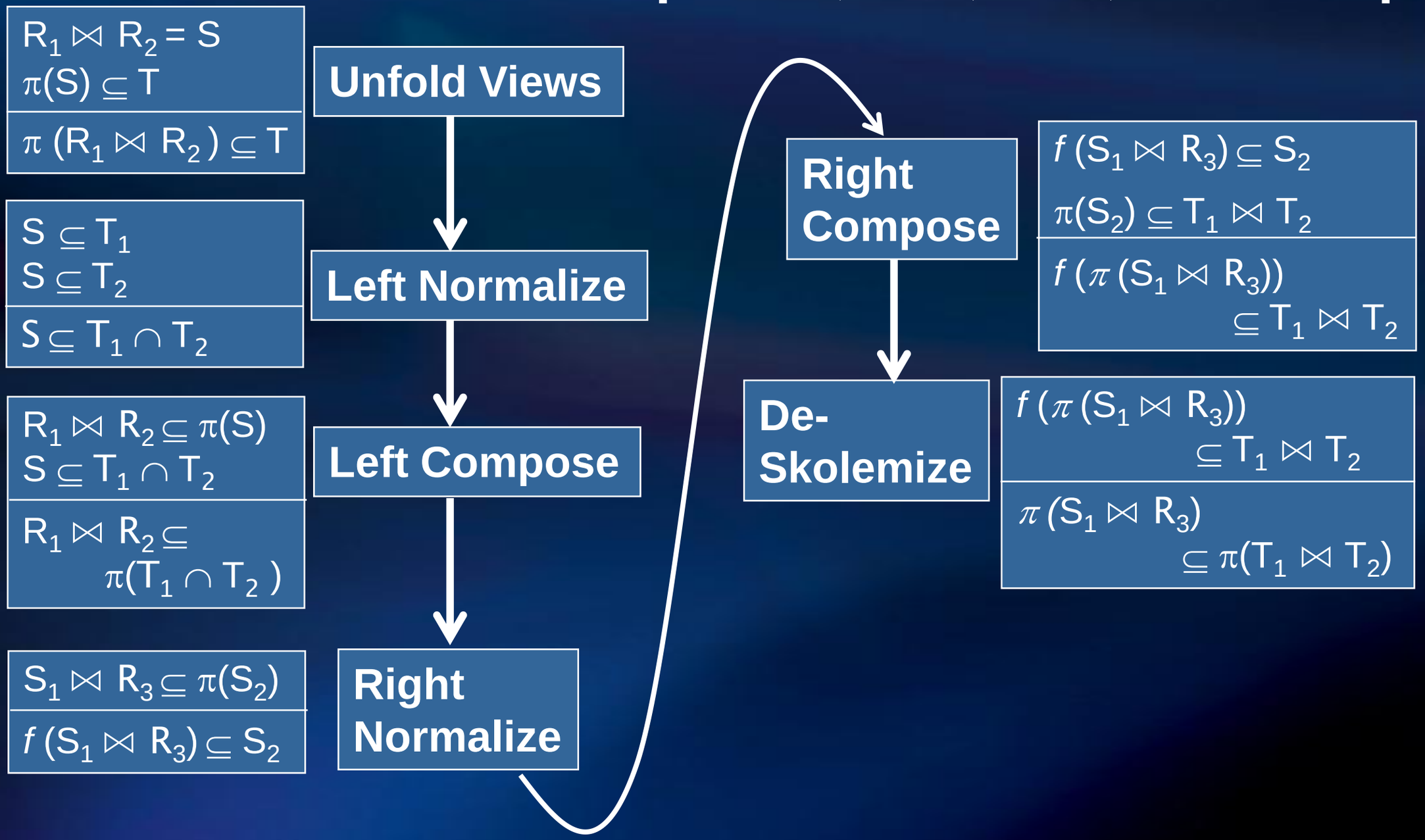
[Fagin, Kolaitis, Popa, Tan. TODS 05]
[Nash, Bernstein, Melnik. TODS 07]
[Yu, Popa. VLDB 05]
[Bernstein, Green, Melnik, Nash. VLDB 06]



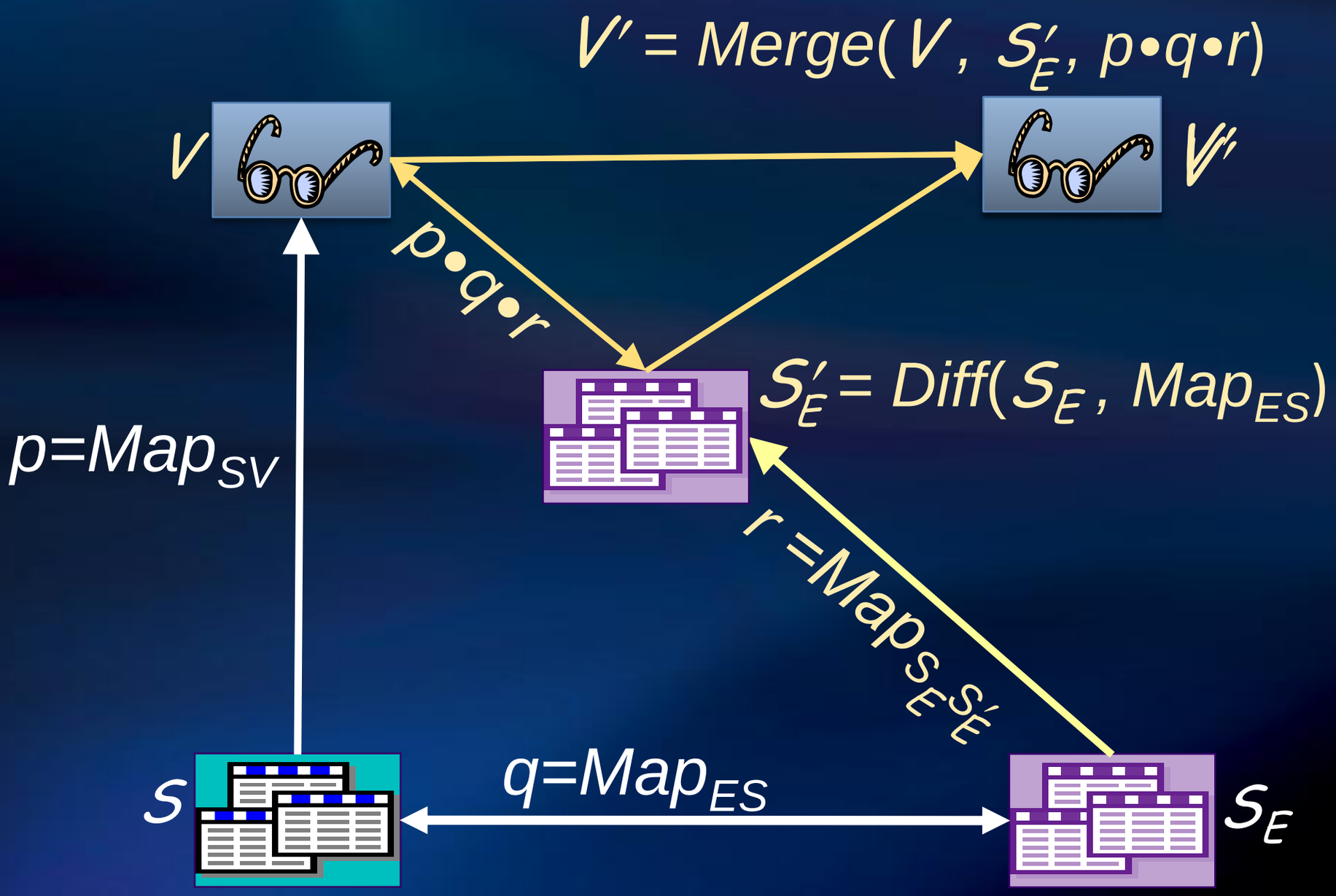
- Some natural 1st-order mapping languages are not closed under composition
  - Sometimes, it's undecidable whether the composition is expressible in the input language
  - Can settle for a partial solution over 1st-order mappings
- Or you can use a 2nd-order mapping language that's closed under composition
  - There's a composition algorithm to compute it
- Some prototype implementations reported
  - Practical applications needed

Our Composition System

[Bernstein, Green, Melnik, Nash VLDB06]



Augment View with S_E's new data



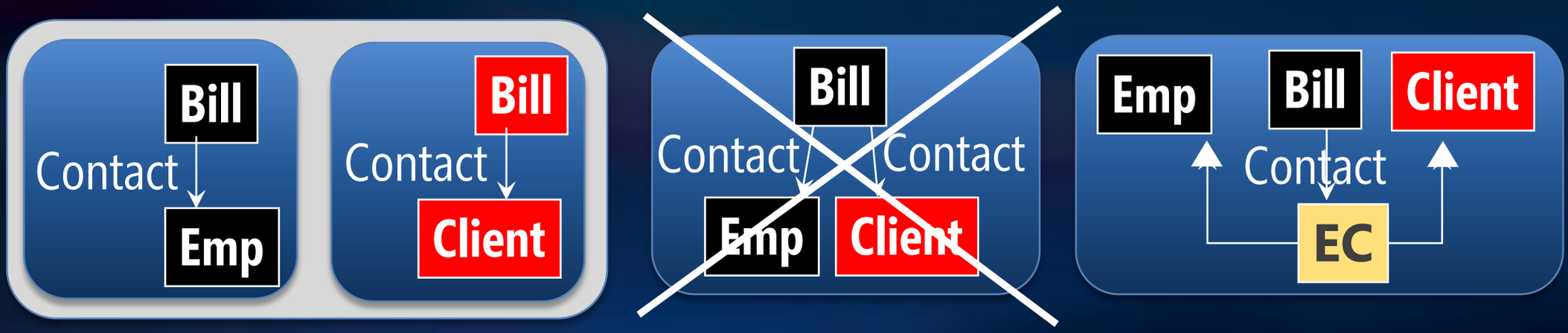
Extract & Diff

- [S'', $map_{S''-S'}$] = Extract(S', $map_{S'-S}$)
  - S'' is a maximal sub-schema of S' that can be populated with data from S via $map_{S'-S}$
  - Related to the materialized view selection problem: S'' is the minimal view needed to populate S
- Diff(S', $map_{S'-S}$) is the complement of Extract
  - It's the view complement problem [Bancillon & Spyratos, TODS 81]
  - An algorithm for select-project-join views is in [Lechtenbörger, Vossen. TODS 03]

Merge

[Casanova, Vidal. PODS 83] [Spaccapietra, Parent. TKDE 94]
[Biskup, Convent. SIGMOD 86] [Pottinger, Bernstein. VLDB 03]
[Buneman, Davidson, Kosky. EDBT 92]

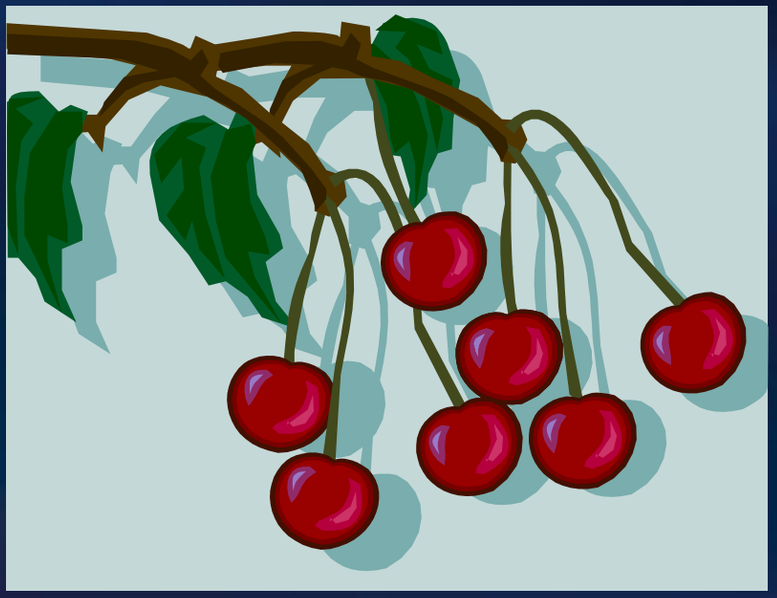
- Take disjoint union of schemas and constraints and then optimize
- Merge algorithms for structural mappings



- Extension: input map is a first-class model
- Not much known for semantic mappings

Low Hanging Fruit

- More surveys
  - Solutions to data programability problems
  - Products that address these problems (e.g. runtimes)
- More case studies
  - Using published solutions and products to solve mapping problems



Other Challenges

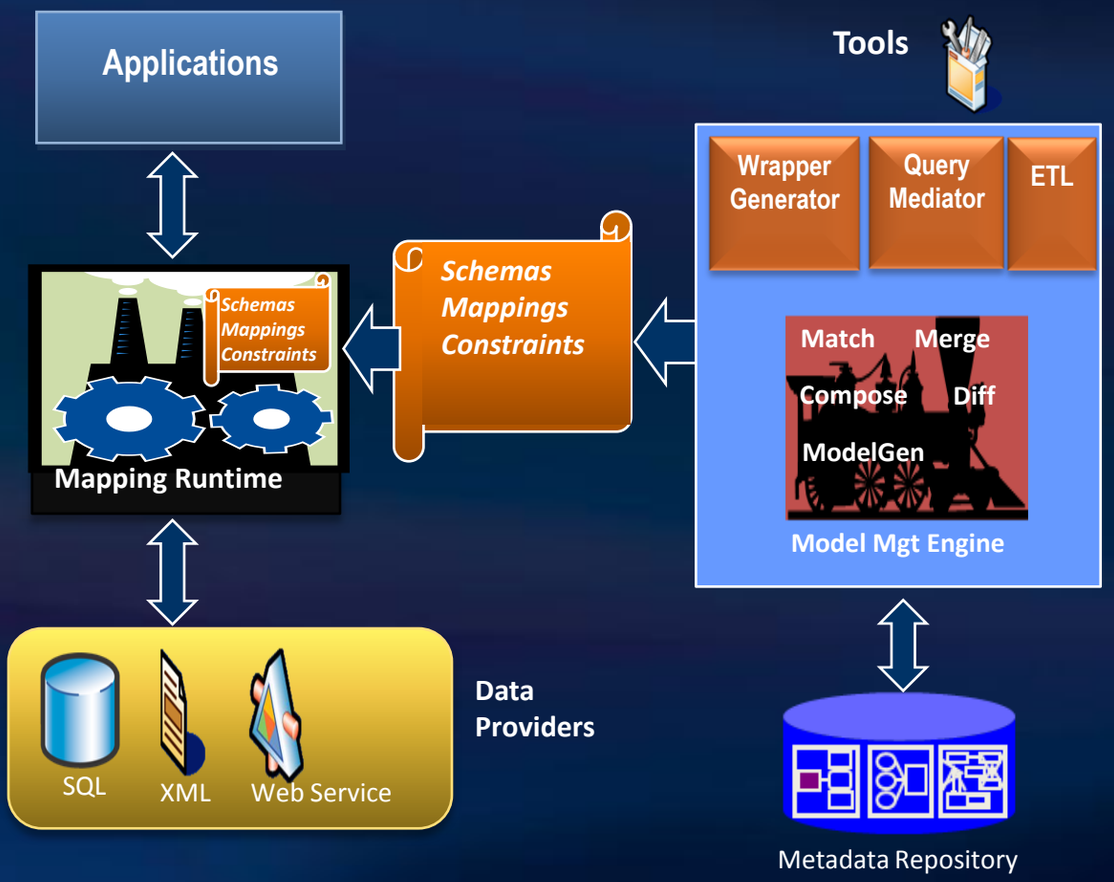
- Semantics and algorithms of operators with more expressive mappings
- Translating behavior on target via mapping to behavior on source

Foundation of operators

- View & schema integration (Merge)
  - Batini et al: ACM Comp. Surveys, 18 (4), ‘86
  - Beeri, Milo: ICDT ‘99
  - Biskup, Convent: SIGMOD ‘86
  - Buneman et al: EDBT ‘92
  - Casanova, Vidal: SIGMOD ‘83
  - Lenzerini: PODS ‘02
  - Motro: IEEE TSE, 13:7, ‘87
  - Pottinger, Berstein: VLDB ‘03
  - Rosenthal, Reiner: TODS 19 (2), ‘94
  - Spaccapietra, Parent: TKDE 6 (2), ‘94
- Composition of queries, views, GLAV maps; query processing
  - Abiteboul, Vianu, Hull: Addison-Wesley, 1995
  - Fernandez et al: TODS 27 (4), ’02
  - Madhavan, Halevy: VLDB’03
  - Papakonstantinou et al, SIGMOD ‘99
  - Shanmugasundaram et al, VLDB ‘01
  - Shu et al, TODS 2 (2), ‘77
- View selection, answering queries using views (Extract, Confluence)
  - Agrawal et al: SIGMOD ’01
  - Chen et al: IDEAS ’02
  - Chirkova et al: VLDB ’01
  - Gupta et al: PODS ’03
  - Halevy: VLDB J. 10:4, ‘01
  - Li et al: ICDT ‘01
  - Theodoratos et al: DKE 39 (3), ‘01
- View update, view complement (Diff)
  - Bancilhon, Spyratos: TODS 6:4, ’81
  - Cosmadakis, Papadimitrou: J. ACM ‘84
  - Dayal, Bernstein: VLDB ’78
  - De Amo et al: IDEAS ‘00
  - Hegner: J. Comp. Sys. Sci., ‘94
  - Keller, Ullman: SIGMOD ‘84
  - Lechtenbörger, Vossen: TODS 28:2, ‘03

Model Management System

- Is it still a goal to build a MMS?
- Or is it just a set of techniques to be applied?



Summary

- There's a big market looking for solutions
- Limited known about run-time scenarios
  - Mostly just for queries
  - Some updates, provenance, integrity constraints
  - Much work needed for synch logic, errors, indexing, notifications, batch loading, ….
- There's progress on many operators
  - But it's incomplete
  - For mappings with limited expressiveness
  - Little known about merge, diff, extract

Microsoft®

Your potential. Our passion.™