

ORACLE[®]

SOFTWARE POWERS THE INTERNET

Integrating Biometric Solutions With Database Management Systems – The Oracle RDBMS Approach

Seema Sundara, Timothy Chorma, Ying Hu, Jagannathan Srinivasan
Oracle Corporation
New England Development Center
Nashua, New Hampshire
{Seema.Sundara, Timothy.Chorma, Ying.Hu, Jagannathan.Srinivasan}@oracle.com

ORACLE®

Biometric Application Overview

- **Enrollment** involves
 - Capturing one or more biometric samples using a sensor device
 - Extracting salient features to generate the biometric template
 - Storing the biometric template along with related information
- **Verification** involves
 - Generating the biometric template as in enrollment
 - Matching the biometric template against a specific biometric template(1:1)
- **Identification** differs from verification in that
 - Match done against a set of enrolled biometric templates (1:n)
 - Possibly returns a set of candidates, each with a matching score

Need For RDBMS in Biometric Solutions

- Every large-scale Biometric Solution requires a RDBMS for efficient storage and access of data
- Example
 - AFIS system¹ - (400 million fingerprints)
 - Point-of-Sale biometric identification system² - (goal of 100 million users)

Using the RDBMS

- Loose Integration
 - Use the RDBMS only for storage of templates
 - Match performed against in-memory structures created from the stored templates
 - Users use Biometric vendor-specific API or BioAPI
- Tight Integration
 - Use the RDBMS for storage of templates as well as for performing the match
 - Users use SQL queries directly against database tables

Loose Integration

- Biometric data is loaded from a database table into memory
- Matching done on custom-built memory-based structures
 - (+) Results in fast matching
 - (-) The solution is memory-bound
- Further scalability, achieved by using Server Farms
 - (-) Vendor-centric solution
 - (-) Can not be easily extended to support multi-modal systems

Tight Integration

- Template matching is implemented within the RDBMS and performed using SQL
- Allows Biometric Vendor to exploit full capabilities of RDBMS including
 - Security
 - Scalability and availability (including Real-Application-Clusters of Oracle 9i)
 - Parallelism

Tight Integration using Oracle's Extensibility Framework

- Object types
- User-defined operators, including notion of *primary* and *ancillary* operators
- User-defined Indexing Schemes (*Indextypes*)

Available in Oracle8i onwards

Tight Integration – Template Storage

- A Biometric Template can be stored in a table column as
 - RAW data type
 - Simple Object data type
 - XML data type
 - Full Common Biometric Exchange File Format-compliant (CBEFF) data type

Tight Integration – Basic Approach

- Biometric Vendors define SQL operators
 - *IdentifyMatch()* Given an input template, returns all the templates which match the input within a certain threshold (defined as primary operator)
 - *Score()* Returns the degree of match of the input template with a stored template (defined as ancillary to *IdentifyMatch* operator)
- Biometric Vendors define implementations for these operators which are specific to their biometric

Tight Integration – Indexing

- Biometric Vendors define an indexing scheme (indextype) for fast evaluation of the *IdentifyMatch()* operator
- Defining an indexing scheme involves
 - Developing a filter(s) which will quickly eliminate a large number of non-matching templates
 - An exact match is performed against the resulting (smaller) set of templates

A Fingerprint Example

- Create a table to store employee data along with their fingerprint template

```
CREATE TABLE Employees  
(name VARCHAR2(128), employee_id INTEGER,  
dept VARCHAR2(30), fingerprint_template RAW(1024));
```

- Index the column storing fingerprint data, for faster access

```
CREATE INDEX FingerprintIndex ON  
Employees(fingerprint_template)  
INDEXTYPE IS FingerprintIndexType;
```

A Fingerprint Example (cont.)

- Retrieve the names and match scores for all employees whose fingerprint matches the input fingerprint

```
SELECT name, Score(1) FROM Employees  
WHERE IdentifyMatch(fingerprint_template, <input>, 1) > 0;
```

Fingerprint Indexing

- Possible indexing approach involves
 - classifying the fingerprints as (Left Loop, Right Loop, Whorl, and other) types
- Query involves
 - classifying the input fingerprint into one of these classes
 - performing exact matches against fingerprints of that class

Basic Indexing Approach

- Build an auxiliary structure (table) that stores extracted portions of the template information along with the unique row identifiers of the base table
- Build native bitmap or B-tree indexes on the auxiliary structure
- A query on this table models the filter that returns a set of row identifiers for which the pair-wise match is performed

Indexing Challenges

- It may not always be possible to develop filter(s) to reduce the search space
- It might be difficult to beat in-memory matching algorithm

Using SQL to Combine Predicates

- SQL can be seamlessly used to integrate biometric predicates with other user-defined relational predicates
- Relational predicates can reduce the search space

Retrieve the names and match scores for employees in the CIS dept, whose fingerprint matches the input fingerprint

```
SELECT name, Score(1) FROM Employees  
WHERE IdentifyMatch(fingerprint_template, <input>, 1) > 0  
AND dept = 'CIS';
```

Supporting Multi-modal Biometric Applications

- Why multi-modal biometrics?
 - Accuracy of a single biometric may be less than desired
 - If one of the traits is altered, user can still be recognized based on other traits

Combining Scores in Multi-modal Applications

```
CREATE TABLE Employees  
(id INTEGER, fingerprint_template RAW(1024),  
face_template RAW(1024));
```

```
SELECT Score(1) , Score(2) FROM Employees  
WHERE IdentifyMatch(fingerprint_template, <input-fp>, 1) >0 AND  
IdentifyMatch(face_template, <input-face>, 2) > 0;
```

```
SELECT Score(1) , Score(2) FROM Employees  
WHERE IdentifyMatch(fingerprint_template, <input-fp>, 1) >0 OR  
IdentifyMatch(face_template, <input-face>, 2) > 0;
```

Combining Scores in Multi-modal Applications (cont.)

```
CREATE TABLE Employees  
(id INTEGER, fingerprint_template RAW(1024),  
face_template RAW(1024));
```

```
SELECT Score(1) , Score(2) FROM Employees  
WHERE (IdentifyMatch(fingerprint_template, <input-fp>, 1) >0 OR  
IdentifyMatch(face_template, <input-face>, 2) > 0) AND  
Score(1) + Score(2) >1;
```

Alternate Approach to Supporting Multi-modal Applications

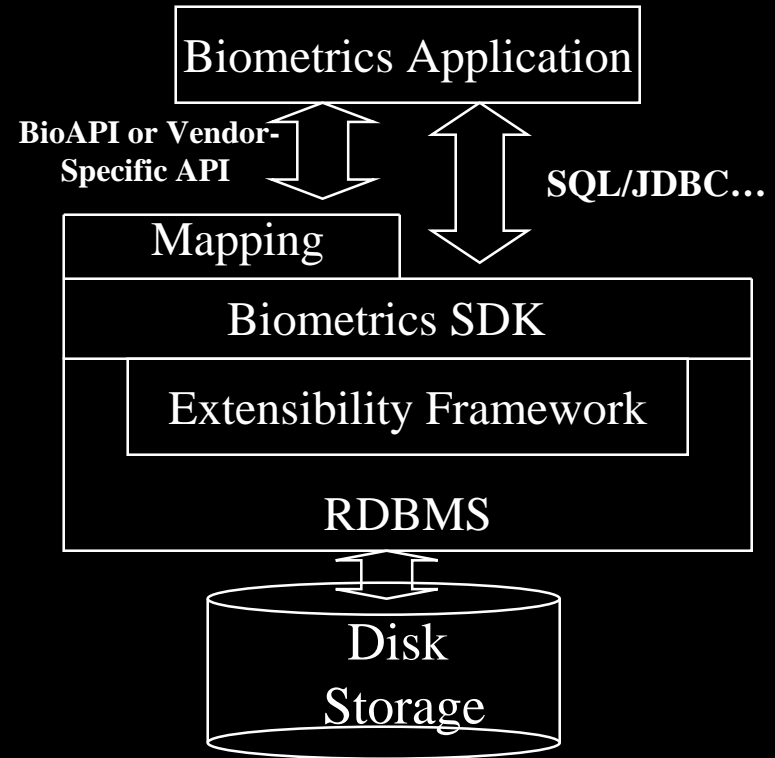
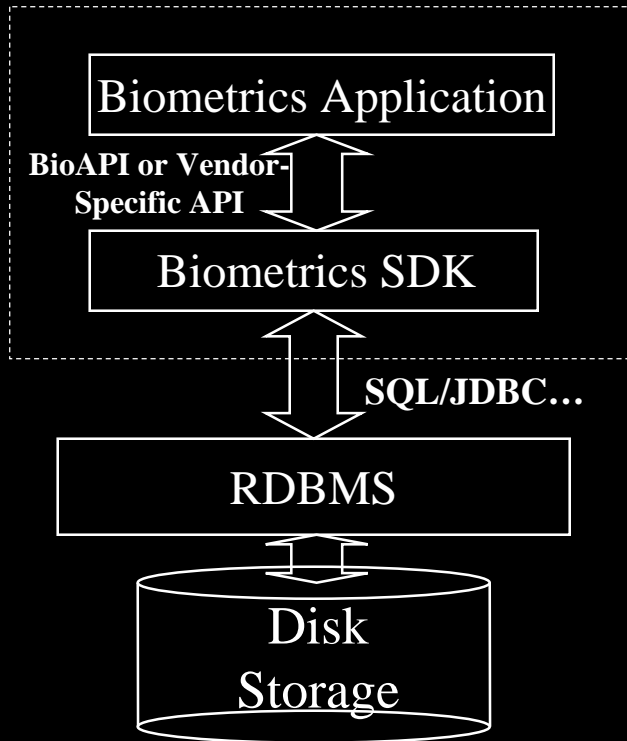
- Create a composite_biometric type to hold both fingerprint and face templates

```
CREATE TABLE Employees  
(id INTEGER, biom CompositeBiometric);
```

```
CREATE INDEX FingerprintFaceIndex ON  
Employees(biom)  
INDEXTYPE IS FingerprintFaceIndexType;
```

```
SELECT Score(1) FROM Employees WHERE  
IdentifyMatch(biom, CompositeBiometric(<input-fp>, <input-face>), 1) > 0 ;
```

Loose Vs. Tight Integration



Loose Vs. Tight Integration (cont.)

Loose

- Memory-based solution; can be fairly efficient and make use of pointers
- Memory bound
- Must custom-build features for large scale handling

Tight

- Caching tables/indexes can help; however incurs buffer cache overhead
- Not memory bound
- Can exploit the features of RDBMS, such as Partitioning, Parallelism, and Security

Loose vs. Tight Integration (cont.)

- Does not need to know about additional RDBMS features
- Index structures can be pure memory-based structures
- Difficult to combine relational predicates
- Difficult to support multi-modal applications
- Requires understanding of DBMS Extensibility features
- Coming up with index structures can be challenging
- Can combine with relational predicates
- Easily extends to handle multi-modal applications

Conclusions

- Biometric-based applications are growing with a need for large-scale deployment
- Current large-scale biometric applications do loose integration with RDBMS
- Tight integration possible using Oracle's Extensibility Framework
 - Enables applications to exploit Oracle's security, scalability, and parallelism features
 - Enables combining relational predicates with biometric predicates
 - Supports multi-modal solutions
- Challenges
 - Developing indexing schemes to reduce search space for various biometrics

Bibliography

- [1] www.integratedbiometric.com/links.htm
- [2] Douglas J. Buettner : A Large-scale Biometric Identification System at the Point of Sale, Biometric Consortium, BCfeb2002

ORACLE[®]

SOFTWARE POWERS THE INTERNET